Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Петрозаводский государственный университет»
Институт математики и информационных технологий

Отчет по дисциплине «Верификация программного обеспечения»

Программная система картирования местности с препятствиями для мобильного наземного робота

Выполнил студент группы 22407:
Мельников Дмитрий Сергеевич

Направление подготовки:
Программная инженерия

Руководитель:
К.А. Кулаков, к.ф.–м.н., доцент

оценка

дата

Петрозаводск —- 2023

Содержание

1 Объект тестирования

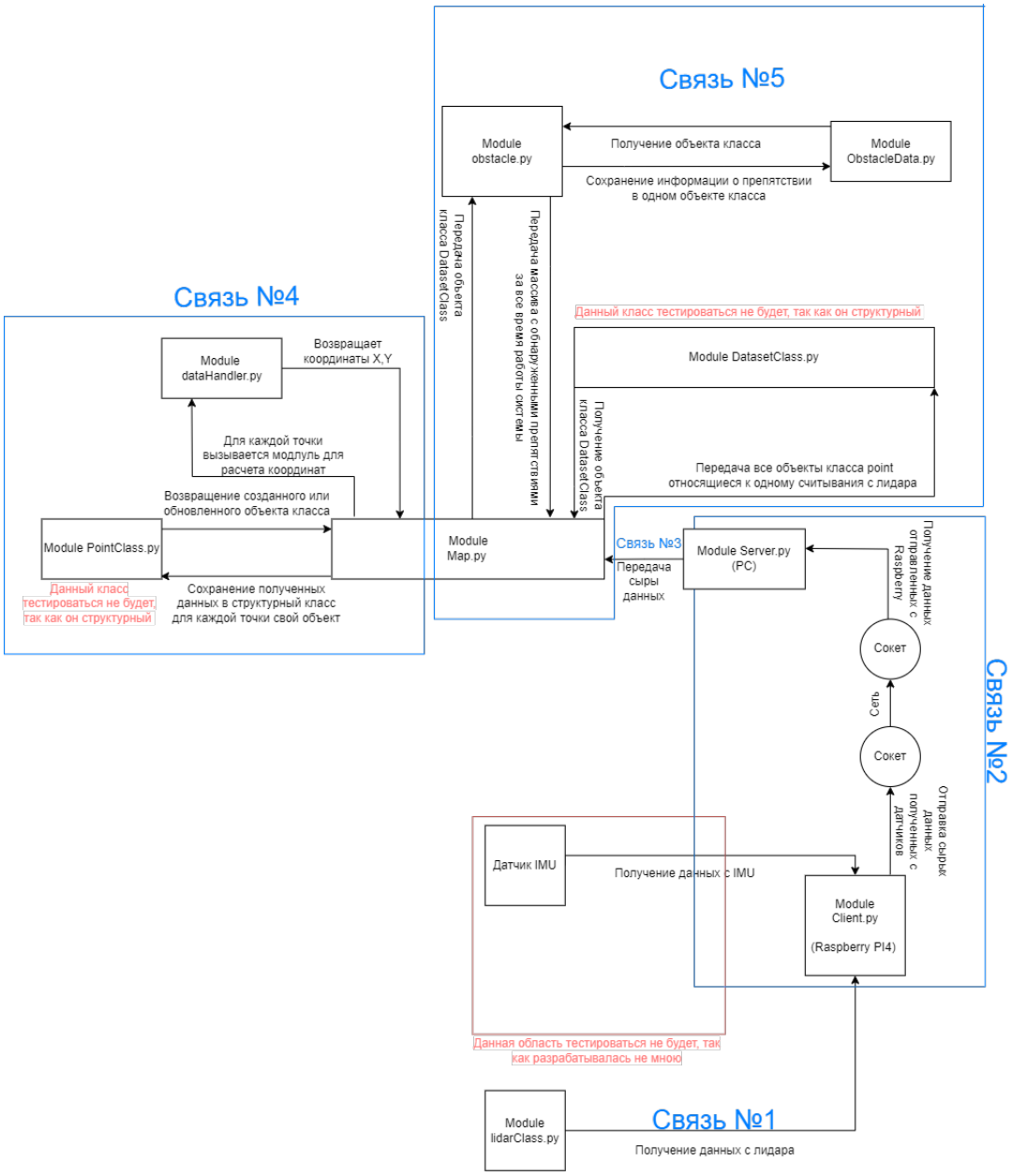
1.1 Описание системы

Программная система - картирования местности с препятствиями, то есть построение карты местности на основе датчиков: дальномеров-лидаров, IMU датчика, стереокамеры для построение карты глубины. В мои задачи входит обработка данных с лидара и построение карты местности. Программная система состоит из 9 файлов, 7 классов, 9 модулей и 10 функций написанных на Python.

1.2 Функциональные требования системы

1. Система предоставляет возможность получать данные с IMU датчика. Данный пункт не будет тестироваться так как разрабатывался не мною.
2. Система предоставляет возможность получать данные с лидара.
3. Система предоставляет возможность передачи данных на внешнее (более мощное) вычислительное устройство с помощью сокетов.
4. Система предоставляет возможность визуализацию обнаруженных препятствий в реальном времени отображая их на 2D карте.
5. Система предоставляет возможность сохранение текущей карты в файл и загрузку старой карты из файла.
6. Пользовательское приложение должно предоставлять возможность просмотра карты, изменение ее масштаба, перемещение по ней и остановку показа карты.
7. Пользовательское приложение должно предоставлять возможность просмотра кол-во нанесенных препятствий на карту, возможность выбирать конкретное препятствие.
8. Пользовательское приложение должно предоставлять возможность просмотра длины периметра выбранного препятствия и кол-во точек этого препятствия.
9. Пользовательское приложение должно предоставлять возможность изменять частоту опроса лидара и частоту отрисовки карты.
10. Пользовательское приложение должно предоставлять возможность просмотра расстояние до препятствия относительно текущего положения робота.

1.3 Архитектура



2 Стратегия тестирования

2.1 Описание модулей

2.1.1 Module lidarClass.py

Модуль отвечает за подключение к лидару и получение данных с него. Для реализации используется 1 класс Lidar() с двумя методами описанных в файле lidarClass.py :
Конструктор данного класа имеет след. поля:

1. self.laser = hokuyo.Hokuyo(port) Передаем порт подключенного датчика и создаем экземпляр класса.
   P.S. В данном поле используется чужая библиотека, которая для меня является черным ящиком.

2. self.laser.laser_on() Здесь так же используется метод из чужой библиотеки. Данный метод выполняет запуск лидара.

3. self.measurement_rate=1. Значение данного поля отвечает за частоту опроса датчика лидар, по умолчанию 1 секунда.

Методы данного класса:

1. Метод GetMeasurementData(self). Возвращает данные полученные с лидара, а именно 720 точек состоящих из расстояние до препятствия и угла относительно лидара.

2. Метод SetMeasurementRate(self, miliseconds:float). Устанавливает частоту опроса данных с датчика. На вход подается 1 аргумент:

   - miliseconds в данный аргумент пользователь указывает частоту опроса датчика в секундах.

2.1.2 Module Client.py

На стороне клиента используется файл client.py, в котором в потоке отправляются данные с лидара, путем создание класса Lidar() и вызова методов GetMeasurementData(self) и SetMeasurementRate(self, miliseconds:float) описанных в прошлом модуле.

2.1.3 Module Server.py

На стороне сервера в файле для реализации используется 1 класс Server() в файле Server.py с методом Data(), в которой происходит подключение к сокету и запись полученных данных.
В данном методе система получет массив кортежей(точек) состоящих из float(угла) с округлением до 2-х знаков, int (расстояния).

2.1.4 Module dataHandler.py

Модуль отвечает за обработку сырых данных, а именно рассчет координат x,y для каждой точки на основе полученных сырых данных с лидара угла и координат.
Для реализации используются класс DataHandler() в файле dataHandler.py и статический метод GetCoordinates(distance:float, angle:float, yaw:float).
Метод GetCoordinates(...) на вход получает след. параметры:

1. distance тип int (расстояние до препятствия).
2. angle тип float с округлением до двух знаков (угол под которым находится препятствие относительно лидара).
3. yaw тип float с округлением до двух знаков (угол рассчитанный датчиком IMU при повороте).

Полученные данные записываются в класс Point(), который будет описан в след. модуле.

2.1.5 Module PointClass.py

Данный модуль не будет тестироваться так как является структурным.
Для реализации используется 1 структурный класс Point() в файле PointClass.py со следующими полями:

1. self.X Координата точки по оси X.
2. self.Y Координата точки по оси Y.
3. self.Length Расстояние до препятствия.
4. self.Angle Угол под которым находится препятствие относительно лидара.

2.1.6 Module DatasetClass.py

Данный модуль не будет тестироваться так как является структурным.

Для реализации используется 1 структурный класс Dataset() в файле DatasetClass.py со следующими полями:

1. self.Points массив классов Point()
2. self.Roll Крен полученный от датчика IMU и рассчитанный через фильтр калмана.
3. self.Pitch Тангаж полученный от датчика IMU и рассчитанный через фильтр калмана.
4. self.Yaw Рысканье полученный от датчика IMU и рассчитанный через фильтр калмана.

2.1.7 Module obstacle.py

Для реализации данного модуля используется 1 класс ObstacleClassifier() в Файле obstacle.py.

Данный модель содержит след.методы:

1. Метод ICP(datasets) Данный метод корректирует значение с датчика IMU, а именно текущее местоположение и поворот в пространстве сравнивая два облака точек.
   На вход данный метод получает массив из двух эелементов(предыдущее и текущее считывание) класса Dataset() (описаном в пункте 2.1.6).
   На выходе возвращает рассчитанный угол поворота робота в сравнении с предыдущим считыванием, пройденный путь и скорректированное второе облако точек.
2. Метод GetObstaclesPoints(dataset)
   На вход метод получает получившегося облока точек после работы ICP(), а возвращает массив получившихся препятствий, которые не были на несены на карту. Массив препятствий представляет из себя множество объектов класса ObstacleData() данный класс будет описан в след. модуле.
3. Метод GetMergeObstacles(obstacles)
   На вход метод получает массив препятствий, а возвращает массив получившихся объединенных препятствий, которые уже и будут отрисованны на карте.

4. Метод GetWidth(Point1,Point2)

   На вход метод получает две точки, а возвращает рассчитанное расстояние между точками.

5. Метод EveryGetWidth(obstacle)

   На вход метод получает объект класса ObstacleData(), а возвращает рассчитанную сумму расстояня между всеми точками препятствия.

2.1.8 Module ObstacleData().py

Данный модуль не будет тестироваться так как является структурным.

Для реализации используется 1 структурный класс ObstacleData() в файле ObstacleData().py со следующими полями:

1. self.Points массив классов Point()

2. self.Color Хранит в себе цвет для отрисовки на карте.

3. self.Width_EP Хранит в себе длину периметра препятствия. (сумма расстояний между всеми точками).

2.1.9 Module Map.py

Данный модуль отвечает за отрисовку препятствий на Canvas. Для реализации данного модуля используется 1 файл Map.py имеет 10 функций. Представляет из себя тестовое приложение показанного на рис.2

1. Функция IncreaseKoef() вызывается по кнопке и увеличивает масштаб карты.

2. Функция DecreaseKoef() вызывается по кнопке и уменьшает масштаб карты.

3. Функция DrawPoints() отвечает за отрисовку точек на Canvas путем раскладывания списка препятствий на точки. У каждого препятствия свой различный цвет, которым и отрисовываются точки. Также каждое обновление карты показывается кол-во препятствий на ней.

4. Функция FuncOn() останавливает обновления карты, обработку и считывания данных с лидара.

5. Функция FuncOff() возобновляет обновления карты, обработку и считывания данных с лидара.

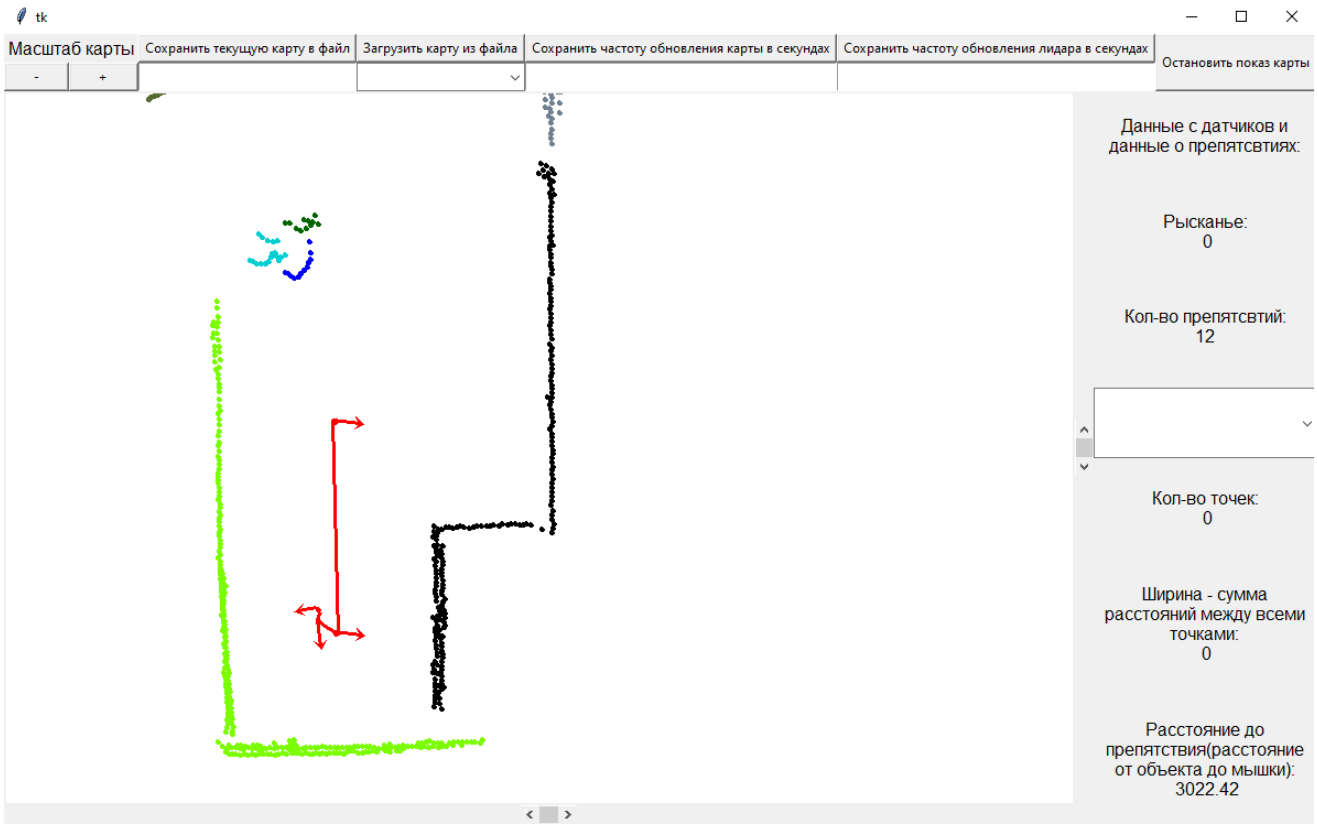
6. Функция move() высчитывает расстояние до препятствия путем наведенния мышкой на препятствие, которое отображенна на карте.

7. Функция FuncMeasurementRateKart() путем ввода значения в секундах в поле и нажатия на кнопку устонавливает частоту обновления карты.

8. Функция FuncMeasurementRateLidar() путем ввода значения в секундах в поле и нажатия на кнопку устонавливает частоту считывания данных с лидара.

9. Функция FuncLoadKart() подгружает уже построенную карту из файла в глобальну переменную obstacles файла MashtabMap.py.

10. Функция FuncSaveKart() сохраняет текущую построенную карту в файл из глобальной переменной obstacles файла MashtabMap.py.



2.2 Стратегия блочного тестирования

Для выполнения блочного тестирования будут использоваться библиотека PyTest. Блочные тесты предполагают тестирование отдельных функций и модулей программы отдельно друг от друга. Будут протестированы все функции, методы, указанные в разделе «Описание модулей».

2.3 Стратегия интеграционного тестирования

При интеграционном тестировании проверяется взаимодействие частей программы между собой, корректное получение исходных данных и отправка результатов. Будет проведено тестирование всех модулей на основе связей интеграции указанные на рис. 1:

1. Связь 1.

   Module «lidarClass.py» с помощью метода GetMeasurementData() получаем даныне с лидара массив кортежей со значениями типа Float и передаем в модуль Client.py и сохраняем в переменную mass

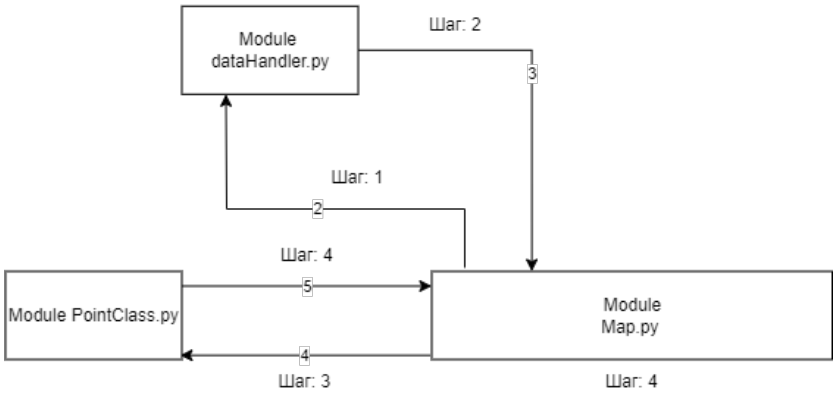
2. Связь 2.

   Module «Client.py» подключается к сокету и передает данные, а именно массив кортежей со значениями типа Float округленных до 2-х знаков в сокет.

   Module <Server.py» считывает данные переданные клиентом. Модуль должен получить массив кортежей со значениями типа Float округленных до 2-х знаков и сохранить в переменнуб points_arr

3. Связь 3. Данная связь не будет тестироваться, так как связь слишком простая.

4. Связь 4.



Шаг 1:

Module «Map.py» пробегает по массив points_arr и поочередно вызывает Module «dataHandler.py» и передает в него (массив кортежей со значениями типа Float округленных до 2-х знаков)

Шаг 2:

Module «dataHandler.py» в себе модуль рассчитывает координаты x,y типа float. После чего передает эти координаты в Module «Map.py».

Шаг 3:

Module «Map.py» создает объекты классов Point их кол-во зависит от длины массива points_arr.

Шаг 4:

Module «PointClass.py» Для каждого объекта класса Point() заполняются данные о точки считанной с лидара в соотвутствующие поля

1. Поле distance(дистанция до объекат) типа float с округлением до двух знаков

1. Поле angle(угол относительно лидара) типа float с округлением до двух знаков

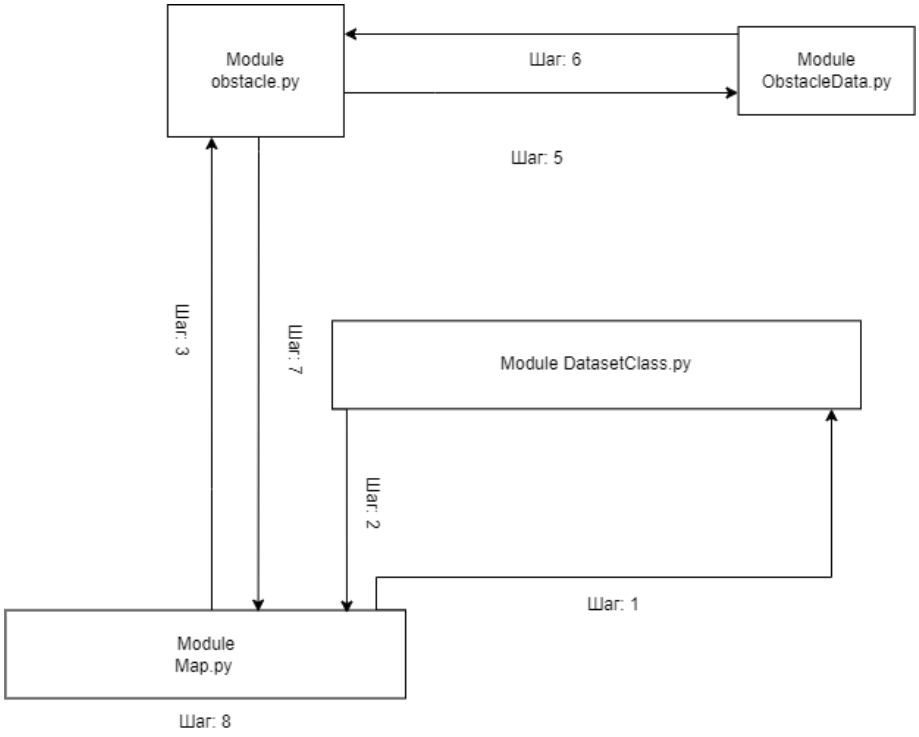
1. Поле X(координата X) типа float

1. Поле Y(координата Y) типа float

Шаг 5:

Module «Map.py» сохраняет все созданные объекты классов в переменную points типа list

5. Связь 5.



Шаг 1:

Module «Map.py» создает объект класса Dataset().

Шаг 2:

Module «DatasetClass.py» В первое поле (Points) сохраняет массив points и возвращает объект класса в Module «Map.py» .

Шаг 3:

Module «Map.py» передает объект класса Dataset() в Module «obstacle.py»

Шаг 4:

Module «obstacle.py» добавляет к массиву datasets полученный объект класса Dataset().

Шаг 5:

Module «obstacle.py» во время работы данного модуля для каждого обнаруженного препятствия создается объект класса ObstacelData(). После работы Module <ObstacelData> добавляет в переменную типа list объект класса.

Шаг 6:

Module <ObstacelData> Записывает данные о препятствии в соотвутствующие поля:

1. Поле points(массив точек относящихся к данному препятствию) типа list

2. Поле Width(рассчитанную длину периметра для этого препятствия) тип float

3. Поле color(цвет данного препятствия) тип string

Шаг 7:

Module «obstacle.py» передает полученные препятствия хранящиейся в переменной obstacles(содержит в себе массив объектов классов ObstacelData()) в Module «Map.py»

Шаг 8:

Module «Map.py» получает и сохраняет в переменной obstacles(массив объектов классов ObstacelData())

2.4 Стратегия аттестационного тестирования

Аттестационное тестирование будет проводиться вручную, согласно скриптам для тестирования. В ходе тестирования будут проверяться все функциональные требования. Указанные в пункте 1.2

2.5 Стратегия нагрузочного тестирование

В ходе нагрузочного тестирования будет проверяться способность основных компонент алгоритмов выдавать результат работы за конечный период времени без ошибок переполнения памяти. При построении карты работа алгоритма не должна превышать 2-ух секунд. В зависимости от уже на несенных на карту точек посчитаем ожидаемое время работы алгоритма.

Количество точек	Ожидаемое время
700-1400	менее 1 секунды.
1400-3000	менее чем 1.5 секунды.
3000<	менее 2 секунд.

3 Детальный план тестов

3.1 Блочное тестирование

ID Теста	Б1
Объект тестирования	Module lidarClass.py, метод GetMeasurementData().
Цель тестирования	Проверка возможности получения данных с лидара
Входные данные	Лидар должен быть подключен к устройству. Импортируем Класс Lidar(), создаем объект класса и передаем в конструктор класса порт /dev/ttyACM0(соответствующий лидару) на устройстве Rasspbery PI4. mass=lidar.GetMeasurementData()
Ожидаемый результат	Переменная mass заполнится более чем шестьюстами точками(дистанция, угол) считанных с лидара
Тип	Позитивный

ID Теста	Б2
Объект тестирования	Module lidarClass.py, метод GetMeasurementData().
Цель тестирования	Проверка случая, когда при считывание данных с лидара произошла ошибка.
Входные данные	Лидар не подключен к устройству или сломан. Импортируем Класс Lidar(), создаем объект класса и передаем в конструктор класса порт /dev/ttyACM0(соответствующий лидару) на устройстве Rasspbery PI4. mass=lidar.GetMeasurementData()
Ожидаемый результат	Вывод сообщения пользователю «Неверный порт» и команда завершается
Тип	Негативный

ID Теста	Б3
Объект тестирования	Module lidarClass.py, метод SetMeasurementRate(self, miliseconds:float)
Цель тестирования	Проверка возможности установки частоты считывание данных с лидара
Входные данные	Лидар должен быть подключен к устройству. Импортируем Класс Lidar(), создаем объект класса и передаем в конструктор класса порт /dev/ttyACM0(соответствующий лидару) на устройстве Rasspbery PI4. miliseconds=0.1, timer=0, timer2=0 lidar.SetMeasurementRate(miliseconds) timer= time.time() mass=lidar.GetMeasurementData() timer2= time.time() timer=timer2-timer
Ожидаемый результат	Переменная timer больше 10-ти секунд, но меньше 11
Тип	Позитивный

ID Теста	Б4
Объект тестирования	Module lidarClass.py, метод SetMeasurementRate(self, miliseconds:float).
Цель тестирования	Проверка случая, когда методу были переданны не допустимые значения.
Входные данные	Лидар должен быть подключен к устройству. Импортируем Класс Lidar(), создаем объект класса и передаем в конструктор класса порт /dev/ttyACM0(соответствующий лидару) на устройстве Rasspbery PI4. miliseconds=-0.1 lidar.SetMeasurementRate(miliseconds)
Ожидаемый результат	Вывод сообщения пользователю «Частота считывания должна быть числовым и положительным значением»
Тип	Негативный

ID Теста	Б5
Объект тестирования	Module lidarClass.py, метод SetMeasurementRate(self, miliseconds:float).
Цель тестирования	Проверка случая, когда методу были переданны не допустимые значения.
Входные данные	Лидар должен быть подключен к устройству. Импортируем Класс Lidar(), создаем объект класса и передаем в конструктор класса порт /dev/ttyACM0(соответствующий лидару) на устройстве Rasspbery PI4. miliseconds="two" lidar.SetMeasurementRate(miliseconds)
Ожидаемый результат	Вывод сообщения пользователю «Частота считывания должна быть числовым и положительным значением»
Тип	Негативный

ID Теста	Б6
Объект тестирования	Module Server.py.
Цель тестирования	Проверка открытия сервера для передачи данных.
Входные данные	from Server import * server=Server()
Ожидаемый результат	В консоль выведется сообщение «Сервер запущен и ожидает подключений...»
Тип	Позитивный

ID Теста	Б7
Объект тестирования	Module Server.py.
Цель тестирования	Проверка случая, когда в ip сервера были переданы не верные данные.
Входные данные	server=Server('121')
Ожидаемый результат	Вывод сообщения пользователю «Введенный ip не может быть использован.»
Тип	Негативный

ID Теста	Б8
Объект тестирования	Module Server.py и Client.py, метод Data().
Цель тестирования	Проверка передачи данных через сокет на внешнее устройство.
Входные данные	server.Server(127.0.0.1) На клиенте запускаем файл Client.py и внутри него создаем массив [(2,3),(4,8)] points_arr=server.Data()
Ожидаемый результат	Переменная point_arr будет содержать эллементы [(2,3),(4,8)]
Тип	Позитивный

ID Теста	Б9
Объект тестирования	Module Client.py, метод LidarThread().
Цель тестирования	Проверка случая, когда не получилось подключиться к сокету.
Входные данные	server.Server(127.0.0.1) На клиенте запускаем файл Client.py в файле указан не правильны ip, а именно(127.0.0.2) для подключения
Ожидаемый результат	Вывод сообщения пользователю «Не получилось подключиться к роботу для считывания данных»
Тип	Негативный

ID Теста	Б10
Объект тестирования	Module dataHandler.py, метод GetCoordinates().
Цель тестирования	Проверка правильности рассчета координат для точек считанных с лидара.
Входные данные	Создаем экземпляр класса data=DataHandler(). coordinates=data.GetCoordinates(400, 28, 0, 0, 0)
Ожидаемый результат	Переменная coordinates будет иметь два значения 353.18 187.79 (координаты X и Y соответственно)
Тип	Позитивный

ID Теста	Б11
Объект тестирования	Module dataHandler.py, метод GetCoordinates().
Цель тестирования	Проверка случая, когда в метод были переданы не корректные данные.
Входные данные	Создаем экземпляр класса data=DataHandler(). coordinates=data.GetCoordinates('asdas', 28, 0,0,0)
Ожидаемый результат	Вывод сообщения пользователю «В метод GetCoordinates() были переданы аргумент/ы не числового типа.»
Тип	Негативный

ID Теста	Б12
Объект тестирования	Module obstacle.py, метод GetWidth()
Цель тестирования	Проверка правильности расчета расстояния между препятствиями.
Входные данные 1. Для превого случая:	obs=ObstacleClassifier() point1=Point(-50.15,151.94,160,108.27,0,0) point2=Point(-50.15,151.94,160,108.27,0,0) dist1=obs.GetWidth(point1,point2) 2. Для второго случая: point3=Point(-39.52,154.01,159,104.39,0,0) point2=Point(-50.15,151.94,160,108.27,0,0) dist2=obs.GetWidth(point3,point2)
Ожидаемый результат	Метод вернет расстояние между точкми: 1. dist1=0. 2. dist2=10.829672201872034
Тип	Позитивный

ID Теста	Б13
Объект тестирования	Module obstacle.py, метод GetWidhtEveryPoint()
Цель тестирования	Проверка правильности расчета расстояния между всеми точками препятствий.
Входные данные	obs=ObstacleClassifier() point1=Point(-50.15,151.94,160,108.27,0,0) point2=Point(-50.15,151.94,160,108.27,0,0) point3=Point(-39.52,154.01,159,104.39,0,0) obstacle=ObstacleData([point1,point2,point3],'red',0) dist1=GetWidhtEveryPoint(obstacle.Points)
Ожидаемый результат	dist1==10.829672201872034
Тип	Позитивный

ID Теста	Б14
Объект тестирования	Module obstacle.py, метод GetWidth()
Цель тестирования	Проверка случая, когда в метод был передан не объект класса Point
Входные данные	obs=ObstacleClassifier() dist1=obs.GetWidth(1,2)
Ожидаемый результат	Вывод сообщения пользователю «В метод GetWidth были переданны аргумента отличные от объекта класса Point»
Тип	Негативный

ID Теста	Б15
Объект тестирования	Module obstacle.py, метод GetWidhtEveryPoint()
Цель тестирования	Проверка случая, когда в метод был передан не объект класса ObstacleData
Входные данные	obs=ObstacleClassifier() dist1=obs.GetWidhtEveryPoint(1)
Ожидаемый результат	Вывод сообщения пользователю «В метод GetWidhtEveryPoint был передан аргумент отличный от list»
Тип	Негативный

ID Теста	Б16
Объект тестирования	Module obstacle.py, метод ICP().
Цель тестирования	Проверка правильности работы данного алгоритма(метода).
Входные данные	Datasets=[] Datasets.append([Массив содержит данные, которые находятся в файле по ссылке *тык*],0,0,0) Datasets.append([Массив содержит данные, которые находятся в файле по ссылке *тык*],0,0,0) datasets2,errorYaw,errorMoveX,errorMoveY= obs.ICP(Datasets) Datasets.insert(1,dataset) mass=Datasets[1].Points[0]
Ожидаемый результат	errorYaw<0.000001 errorMoveX<0.000001 errorMoveY<0.000001 Переменная mass должна содержать данные указанные в файле по ссылке *тык*
Тип	Позитивный

ID Теста	Б17
Объект тестирования	Module obstacle.py, метод ICP().
Цель тестирования	Проверка случая, когда в метод ICP() был передан не объект класса Dataset
Входные данные	obs=ObstacleClassifier() obs.ICP(1)
Ожидаемый результат	Вывод сообщения пользователю «В метод ICP были переданны аргумента отличные от объекта класса Dataset»
Тип	Негативный

ID Теста	Б18
Объект тестирования	Module obstacle.py, метод GetObstaclesPoints().
Цель тестирования	Проверка правильности работы данного алгоритма(метода GetObstaclesPoints()).
Входные данные	Создаем объект класса obs = ObstacleClassifier() obstacles=obs.GetObstaclesPoints([Point(-50.15,151.94,160,108.27,0,0), Point(-20.15,151.94,160,108.27,0,0), Point(200.15,151.94,160,108.27,0,0)])
Ожидаемый результат	len(obstacles)==2 and len(obstacles[0].Points)==2 and len(obstacles[1].Points)==1 and obstacles[0].Color!=obstacles[1].Color
Тип	Позитивный

ID Теста	Б19
Объект тестирования	Module obstacle.py, метод GetObstaclesPoints().
Цель тестирования	Проверка правильности отсеивания тех же точек(для эффективности работы алгоритма(метода GetObstaclesPoints())).
Входные данные	Создаем объект класса obs = ObstacleClassifier() obs=ObstacleClassifier() obstacles=obs.GetObstaclesPoints([Point(-50.15,151.94,160,108.27,0,0), Point(-50.15,151.94,160,108.27,0,0)]) sum1=len(obstacles) obstacles=obs.GetObstaclesPoints([Point(-150.15,151.94,160,108.27,0,0), Point(-50.15,151.94,160,108.27,0,0)]) sum2=len(obstacles)
Ожидаемый результат	sum1!=sum2. sum1==1. sum1==2.
Тип	Позитивный

ID Теста	Б20
Объект тестирования	Module obstacle.py, метод GetObstaclesPoints().
Цель тестирования	Проверка случая, когда в метод был передан не массив объектов класса Point
Входные данные	Создаем объект класса obs = ObstacleClassifier() obstacles=obs.GetObstaclesPoints(1,2)
Ожидаемый результат	Вывод сообщения пользователю «В метод GetObstaclesPoints() были переданны аргумент отличный от типа list с объектами класса Point»
Тип	Негативный

ID Теста	Б22
Объект тестирования	Module obstacle.py, метод GetMergeObstacles().
Цель тестирования	Проверка случая, когда в метод GetMergeObstacles был передан не list содержащий объекты класса ObstacleData()
Входные данные	obs = ObstacleClassifier() obstacles=obs.GetObstaclesPoints(1)
Ожидаемый результат	Вывод сообщения пользователю «В метод GetMergeObstacles() были переданны аргумент отличный от list содержащий объекты класса ObstacleData»
Тип	Негативный

ID Теста	Б21
Объект тестирования	Module obstacle.py, метод GetMergeObstacles().
Цель тестирования	Проверка правильности работы метода GetMergeObstacles объединение рядом стоящих препятствий.
Входные данные	obstacles=[] obs=ObstacleClassifier() obstacle1=ObstacleData([Point(- 50.15,151.94,160,108.27,0,0),Point(- 50.15,151.94,160,108.27,0,0)],"Black 0) obstacle2=ObstacleData([Point(- 50.15,151.94,160,108.27,0,0),Point(- 50.15,151.94,160,108.27,0,0)],"Black 0) obstacles.append(obstacle1) obstacles.append(obstacle2) sum1=len(obstacles) sumPoint1=len(obstacles[0].Points) obstacles=obs.GetMergeObstacles(obstacles) sum2=len(obstacles) sumPoint2=len(obstacles[0].Points)
Ожидаемый результат	sum1>sum2 sumPoint2>sumPoint1.
Тип	Позитивный

ID Теста	Б23
Объект тестирования	Module Map.py, метод FuncLoadKart().
Цель тестирования	Проверка правильности работы метода FuncLoadKart().
Входные данные	obstacles=[] namefile="B23.txt"(То что содержит файл *тык*) obstacles=FuncLoadKart(namefile)
Ожидаемый результат	len(obstacles) ==28.
Тип	Позитивный

ID Теста	Б24
Объект тестирования	Module Map.py, метод FuncLoadKart().
Цель тестирования	Проверка случая, когда указанный файл для загрузки пуст или содержит не подходящие данные.
Входные данные	namefile="B24.txt"Файл содержит следующие данные (S,S,S,A,ASDA) obstacles=FuncLoadKart(namefile)
Ожидаемый результат	Вывод сообщения пользователю «Файл пуст или содержит не корректные данные»
Тип	Негативный

ID Теста	Б25
Объект тестирования	Module Map.py, метод FuncLoadKart().
Цель тестирования	Проверка случая, когда не был указан файл.
Входные данные	namefile= obstacles=FuncLoadKart(namefile)
Ожидаемый результат	Вывод сообщения пользователю «Файл не был выбран»
Тип	Негативный

ID Теста	Б26
Объект тестирования	Module Map.py, метод FuncSaveKart().
Цель тестирования	Проверка правильности работы метода FuncSaveKart().
Входные данные	obstacles1=[] obstacle1=ObstacleData([Point(- 50.15,151.94,160,108.27,0,0),Point(- 50.15,151.94,160,108.27,0,0)],"Black 0) obstacles1.append(obstacle1) obstacles1.append(obstacle1) savefile="B26.txt " FuncSaveKart(savefile,obstacles1) namefile="B26.txt" obstacles2=FuncLoadKart(namefile))
Ожидаемый результат	len(obstacles1)==len(obstacles2) Количество препятствий в переменных одинаковое
Тип	Позитивный

ID Теста	Б27
Объект тестирования	Module Map.py, метод FuncSaveKart().
Цель тестирования	Проверка случая, когда при сохранении карты не был указан файл.
Входные данные	obstacles=[] savefile= FuncSaveKart(savefile,obstacles)
Ожидаемый результат	Вывод сообщения пользователю «Имя файла для сохранения пусто»
Тип	Негативный

ID Теста	Б28
Объект тестирования	Module Map.py, метод FuncSaveKart().
Цель тестирования	Проверка случая, когда был указан существующий файл.
Входные данные	obstacles=[] savefile="B28.txt"(файл B28.txt уже создан в директории) FuncSaveKart(savefile,obstacles)
Ожидаемый результат	Вывод сообщения пользователю «Указанное имя файла для сохранения уже существует»
Тип	Негативный

ID Теста	Б29
Объект тестирования	Module Map.py, метод IncreaseKoef().
Цель тестирования	Проверка правильности работы метода IncreaseKoef().
Входные данные	koef=5 koef=IncreaseKoef(koef)
Ожидаемый результат	Пременная koef==6.
Тип	Позитивный

ID Теста	Б30
Объект тестирования	Module Map.py, метод IncreaseKoef().
Цель тестирования	Проверка случая, когда был достигнут максимальный масштаб карты.
Входные данные	global koef=10 koef=IncreaseKoef(koef)
Ожидаемый результат	koef==10 (масштаб не изменится).
Тип	Негативный

ID Теста	Б31
Объект тестирования	Module Map.py, метод DecreaseKoef().
Цель тестирования	Проверка правильности работы метода DecreaseKoef().
Входные данные	global koef=5 koef=DecreaseKoef(koef)
Ожидаемый результат	koef==4.
Тип	Позитивный

ID Теста	Б32
Объект тестирования	Module Map.py, метод DecreaseKoef().
Цель тестирования	Проверка случая, когда был достигнут минимальный масштаб карты.
Входные данные	global koef=1 koef=DecreaseKoef(koef)
Ожидаемый результат	koef==1 (масштаб не изменится).
Тип	Негативный

ID Теста	Б33
Объект тестирования	Module Map.py, метод FuncOn().
Цель тестирования	Проверка правильности работы метода FuncOn().
Входные данные	global flag=0 flag=FuncOn(flag)
Ожидаемый результат:	flag==0 and th2.is_alive()==1
Тип	Позитивный

ID Теста	Б34
Объект тестирования	Module Map.py, метод FuncOff().
Цель тестирования	Проверка правильности работы метода FuncOff().
Входные данные	global flag=1 th2 = Thread(target = Draw_Points) flag=FuncOff(flag)
Ожидаемый результат	flag==1 and th2.is_alive()==0
Тип	Позитивный

ID Теста	Б35
Объект тестирования	Module Map.py, метод FuncMeasurementRateLidar()
Цель тестирования	Проверка возможности смены частоты опроса лидара
Входные данные	global MRL=1 MRL=FuncMeasurementRateLidar(10)
Ожидаемый результат	MRL=10 (частота опроса лидара в секунду увеличиться)
Тип	Позитивный

ID Теста	Б36
Объект тестирования	Module Map.py, метод FuncMeasurementRateLidar()
Цель тестирования	Проверка случая, когда была переданная отрицательная частота опроса лидара.
Входные данные	global MRL=1 MRL=FuncMeasurementRateLidar(-10)
Ожидаемый результат	Вывод сообщения пользователю «Частота опроса лидара не может быть меньше 1»
Тип	Негативный

ID Теста	Б37
Объект тестирования	Module Map.py, метод FuncMeasurementRateLidar()
Цель тестирования	Проверка случая, когда было переданно буквенное значения для частоты опроса лидара.
Входные данные	global MRL=1 MRL=FuncMeasurementRateLidar("ss")
Ожидаемый результат	Вывод сообщения пользователю «Частота опроса лидара не может иметь не числовое значение»
Тип	Негативный

ID Теста	Б38
Объект тестирования	Module Map.py, метод FuncMeasurementRateKart()
Цель тестирования	Проверка возможности смены частоты отрисовки карты
Входные данные	global MRK=1 MRK=FuncMeasurementRateKart(0.1)
Ожидаемый результат	MRK=0.1 (частота отрисовки карты уменьшиться)
Тип	Позитивный

ID Теста	Б39
Объект тестирования	Module Map.py, метод FuncMeasurementRateKart()
Цель тестирования	Проверка возможности смены частоты отрисовки карты
Входные данные	global MRK=1 MRK=FuncMeasurementRateKart(0)
Ожидаемый результат	Вывод сообщения пользователю «Частота опроса карты не может быть меньше 1»
Тип	Негативный

ID Теста	Б40
Объект тестирования	Module Map.py, метод FuncMeasurementRateKart()
Цель тестирования	Проверка возможности смены частоты отрисовки карты
Входные данные	global MRK=1 MRK=FuncMeasurementRateKart("sadea")
Ожидаемый результат	Вывод сообщения пользователю «Частота опроса карты не может иметь не числовое значение»
Тип	Негативный

3.2 Аттестационное тестирование

ID Теста	A1
Объект тестирования	ФТ:2,3,4
Цель тестирования	Успешное построение карты на основе полученных данных с лидара
Расположение робота	Расположение робота указанно на рис. 1
Входные данные	Лидар подключен к роботу Внешнее устройство подключено к роботу через сокет У оператора включенно приложение на компьютере Оператор нажимает кнопку «Запуск визуализации»
Ожидаемый результат	Отрисованная карта показанна на рис. 2
Тип	Позитивный

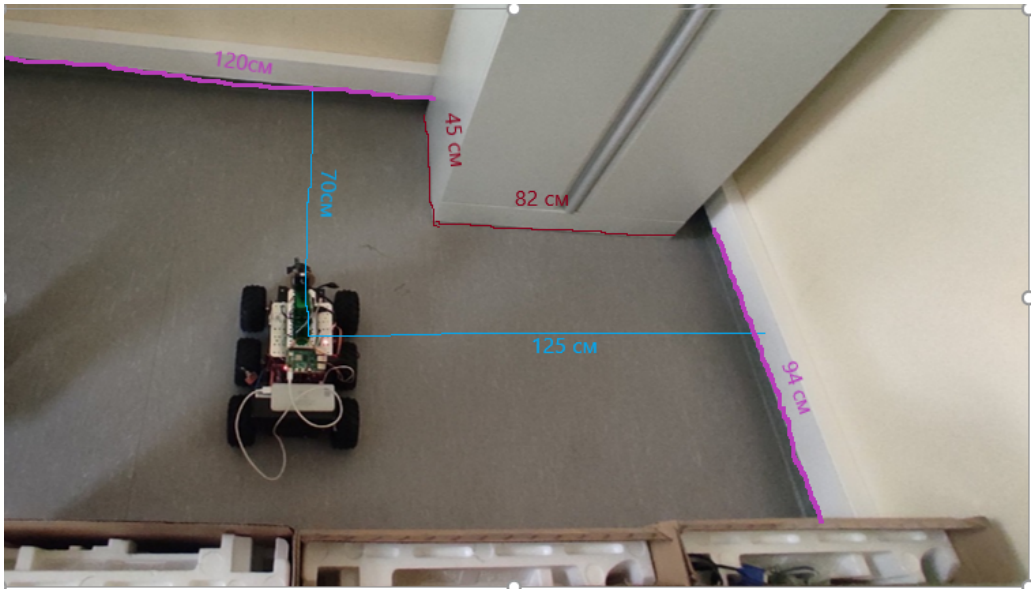


Рис. 1 – Положение робота для теста A1

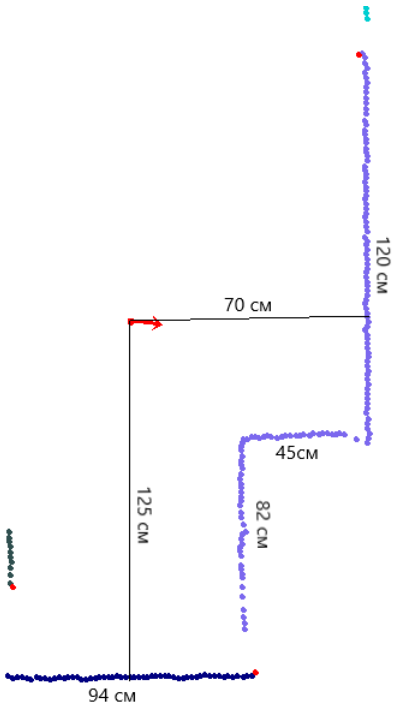


Рис. 2 – Ожидаемый результат для теста A1

ID Теста	A2
Объект тестирования	ФТ:2,3,4
Цель тестирования	Проверка подключение или исправности лидара.
Входные данные	Лидар не подключен к роботу Внешнее устройство подключено к роботу через сокет У оператора включенно приложение на компьютере Оператор нажимает кнопку «Запуск визуализации»
Ожидаемый результат	Вывод сообщения пользователю «Лидар не обнаружен системой»
Тип	Негативный

ID Теста	А3
Объект тестирования	ФТ:2,3,4
Цель тестирования	Проверка подключения робота по сокету к внешнему устройству.
Входные данные	Лидар подключен к роботу Внешнее устройство подключено к роботу через сокет У оператора включенно приложение на компьютере Оператор нажимает кнопку «Запуск визуализации» У робота отключается питание
Ожидаемый результат	Вывод сообщения пользователю «Робот пропал из сети»
Тип	Негативный

ID Теста	A4
Объект тестирования	ФТ:7
Цель тестирования	Проверка возможности просмотра кол-во нанесенных препятствий на карту. Возможность выбирать конкретное препятствие.
Расположение робота	Расположение робота указанно на рис. 1
Входные данные	Лидар подключен к роботу Внешнее устройство подключено к роботу через сокет У оператора включенно приложение на компьютере Оператор нажимает кнопку «Запуск визуализации» После отрисовки препятствий, у оператора в приложении в соответствующем поле отображается кол-во препятствий и он может выбрать одно из них во вложенном списке.
Ожидаемый результат	Вывод кол-во препятствий равное четырем в соответствующем поле
Тип	Позитивный

ID Теста	A5
Объект тестирования	ФТ:5
Цель тестирования	Проверка возможности сохранение текущей карты в файл.
Расположение робота	Расположение робота указанно на рис. 1
Входные данные	Лидар подключен к роботу Внешнее устройство подключено к роботу через сокет У оператора включенно приложение на компьютере Оператор нажимает кнопку «Запуск визуализации» После отрисовки первых препятствий, оператор вводит название файла в соответствующее поле и нажимает кнопку «Сохранить текущую карту в файл».
Ожидаемый результат	В файл сохранится построенная карта.
Тип	Позитивный

ID Теста	A6
Объект тестирования	ФТ:5
Цель тестирования	Проверка возможности сохранение текущей карты в файл.
Входные данные	Лидар подключен к роботу Внешнее устройство подключено к роботу через сокет У оператора включенно приложение на компьютере Оператор нажимает кнопку «Запуск визуализации» После отрисовки первых препятствий, нажимает кнопку «Сохранить текущую карту в файл».
Ожидаемый результат	Вывод сообщения пользователю «Укажите имя файла для сохранение карты»
Тип	Негативный

ID Теста	A7
Объект тестирования	ФТ:5
Цель тестирования	Проверка возможности сохранение текущей карты в файл.
Входные данные	Лидар подключен к роботу Внешнее устройство подключено к роботу через сокет У оператора включенно приложение на компьютере Оператор нажимает кнопку «Запуск визуализации» После отрисовки первых препятствий, оператор вводит название файла в соответствующее поле и нажимает кнопку «Сохранить текущую карту в файл».
Ожидаемый результат	Вывод сообщения пользователю «Указанный файл уже существует укажите другой»
Тип	Негативный

ID Теста	A8
Объект тестирования	ФТ:5
Цель тестирования	Проверка возможности загрузки карты из файла.
Входные данные	Лидар подключен к роботу Внешнее устройство подключено к роботу через сокет У оператора включенно приложение на компьютере Оператор вводит в соответствующее поел название файла A8(То что содержит файл *тык*) и нажимает кнопку «Загрузить карту из файла»
Ожидаемый результат	Загруженная карта отображается в приложении 
Тип	Позитивный

ID Теста	A9
Объект тестирования	ФТ:5
Цель тестирования	Проверка возможности загрузки карты из файла.
Входные данные	Лидар подключен к роботу Внешнее устройство подключено к роботу через сокет У оператора включенно приложение на компьютере Оператор нажимает кнопку «Загрузить карту из файла»
Ожидаемый результат	Вывод сообщения пользователю «Укажите имя файла для загрзуки карты»
Тип	Негативный

ID Теста	A10
Объект тестирования	ФТ:5
Цель тестирования	Проверка загрузки карты из файла.
Входные данные	Лидар подключен к роботу Внешнее устройство подключено к роботу через сокет У оператора включенно приложение на компьютере Оператор вводит название файла и нажимает кнопку «Загрузить карту из файла»
Ожидаемый результат	Вывод сообщения пользователю «Указанного файла не существует укажите другой»
Тип	Негативный

ID Теста	A11
Объект тестирования	ФТ:6
Цель тестирования	Проверка возможности изменение масштаба.
Входные данные	Лидар подключен к роботу Внешнее устройство подключено к роботу через сокет У оператора включенно приложение на компьютере Оператор нажимает кнопку «Запуск визуализации» Оператор нажимает кнопку «-» или «+»
Ожидаемый результат	Карта отдаляется и приближается соответственно нажатим клвишам.
Тип	Позитивный

ID Теста	A12
Объект тестирования	ФТ:6
Цель тестирования	Проверка возможности остановки визуализации карты.
Входные данные	Лидар подключен к роботу Внешнее устройство подключено к роботу через сокет У оператора включенно приложение на компьютере Оператор нажимает кнопку «Запуск визуализации» Оператор нажимает кнопку «Остановить визуализацию»
Ожидаемый результат	Визуализация карты останавливается и на экране отображается текущая уже построенная карта соответственно карта не достраивается.
Тип	Позитивный

ID Теста	A13
Объект тестирования	ФТ:6
Цель тестирования	Проверка возможности перемещение по карте.
Входные данные	Лидар подключен к роботу Внешнее устройство подключено к роботу через сокет У оператора включенно приложение на компьютере Оператор нажимает кнопку «Запуск визуализации» Оператор нажимает кнопки влево,вправо или вверх,вниз на соответствующих осях по бокам карты
Ожидаемый результат	Перемещение по карте в соответствии нажатим клавишам.
Тип	Позитивный

ID Теста	A14
Объект тестирования	ФТ:8
Цель тестирования	Проверка возможность определения длины периметра препятствия.
Расположение робота	Расположение робота указанно на рис. 1
Входные данные	Лидар подключен к роботу Внешнее устройство подключено к роботу через сокет У оператора включенно приложение на компьютере Оператор нажимает кнопку «Запуск визуализации» После отрисовки препятствий, оператор выбирает препятствие с фиолетовым цветом во вложенном меню справа.
Ожидаемый результат	Вывод длины периметра равной 245-248 в соответствующее поле
Тип	Позитивный

ID Теста	A15
Объект тестирования	ФТ:9
Цель тестирования	Проверка возможности изменять частоту опроса лидара и частоту обновления карты.
Входные данные	Лидар подключен к роботу Внешнее устройство подключено к роботу через сокет У оператора включенно приложение на компьютере Оператор нажимает кнопку «Запуск визуализации» Оператор вводит положительное число больше 1 в соответствующее поле под кнопкой «Сохранить частоту обновления лидара в секундах» и нажимает ее. Оператор вводит положительное число больше 1 в соответствующее поле под кнопкой «Сохранить частоту обновления лидара в секундах» и нажимает ее.
Ожидаемый результат	Новые препятствия будут появляться на карте каждую секунду.
Тип	Позитивный

ID Теста	A16
Объект тестирования	ФТ:9
Цель тестирования	Проверка возможности изменять частоту опроса лидара и частоту обновления карты.
Входные данные	Лидар подключен к роботу Внешнее устройство подключено к роботу через сокет У оператора включенно приложение на компьютере Оператор нажимает кнопку «Запуск визуализации» Оператор вводит -10 в соответствующее поле под кнопкой «Сохранить частоту обновления лидара в секундах» и нажимает ее. Оператор вводит -10 в соответствующее поле под кнопкой «Сохранить частоту обновления лидара в секундах» и нажимает ее.
Ожидаемый результат	Вывод сообщения пользователю «Частота опрса должна быть положительной и больше 0.1».
Тип	Негативный

ID Теста	A17
Объект тестирования	ФТ:9
Цель тестирования	Проверка возможности изменять частоту опроса лидара и частоту обновления карты.
Входные данные	Лидар подключен к роботу Внешнее устройство подключено к роботу через сокет У оператора включенно приложение на компьютере Оператор нажимает кнопку «Запуск визуализации» Оператор нажимает кнопку «Сохранить частоту обновления лидара в секундах» . Оператор нажимает кнопку «Сохранить частоту обновления лидара в секундах» .
Ожидаемый результат	Вывод сообщения пользователю «Поле для частоты опроса не должно быть пустое».
Тип	Негативный

ID Теста	A18
Объект тестирования	ФТ:10
Цель тестирования	Проверка возможности просмотра расстояние до препятствия относительно текущего положения робота.
Расположение робота	Расположение робота указанно на рис. 1
Входные данные	Лидар подключен к роботу Внешнее устройство подключено к роботу через сокет У оператора включенно приложение на компьютере Оператор нажимает кнопку «Запуск визуализации» После отрисовки препятствий, оператор наводит мышью на точку, которая соединяется черной линей с роботом и имеет подпись в 70см на рис. 2 .
Ожидаемый результат	В соответствующем поле выводится расстояние 68-72мм.
Тип	Позитивный

3.3 Интеграционное тестирование

ID Теста	И1
Объект тестирования	Связь интеграции №1
Цель тестирования	Проверка передачи данных с лидара между модулем lidarClass.py и модулем Client.py
Описание тестирования	Тесты для данной интеграции проводят по след. сценарию: Шаг 1: В Rasspbery PI4 подключаем лидар по порту /dev/ttyACM0 и включаем робота Шаг 2: В файле Client.py импортируем Класс Lidar(), создаем объект класса и передаем в конструктор класса порт /dev/ttyACM0(соответствующий лидару) на устройстве Rasspbery PI4 Шаг 3: Вызываем функцию GetMeasurementData() и возвращаемые данные присваиваем к переменной mass
Входные данные	местоположение робота картинку
Ожидаемый результат	Переменная mass в файле Client.py заполнится более 600 точками(дистанция, угол) типа Float считанных с лидара
Тип	Позитивный

ID Теста	И2
Объект тестирования	Связь интеграции №1
Цель тестирования	Проверка передачи некорректных данных с лидара между модулем lidarClass.py и модулем Client.py
Описание тестирования	Тесты для данной интеграции проводят по след. сценарию: Шаг 1: В Rasspbery PI4 подключаем лидар по порту /dev/ttyACM0 и включаем робота Шаг 2: В файле Client.py импортируем Класс Lidar(), создаем объект класса и передаем в конструктор класса порт /dev/ttyACM0(соответствующий лидару) на устройстве Rasspbery PI4 Шаг 3: Вызываем функцию GetMeasurementData() и возвращаемые данные присваиваем к переменной s Шаг 4: s=1; mass=s
Входные данные	-
Ожидаемый результат	Система прекращает работу и выводим в log файл информациию об ошибке. Вывод ошибки «С лидара было полученно значение типа int, а не list»
Тип	Негативный

ID Теста	ИЗ
Объект тестирования	Связь интеграции №1
Цель тестирования	Проверка передачи некорректных данных с лидара между модулем lidarClass.py и модулем Client.py
Описание тестирования	Тесты для данной интеграции проводят по след. сценарию: Шаг 1: В Rasspbery PI4 подключаем лидар по порту /dev/ttyACM0 и включаем робота Шаг 2: В файле Client.py импортируем Класс Lidar(), создаем объект класса и передаем в конструктор класса порт /dev/ttyACM0(соответствующий лидару) на устройстве Rasspbery PI4 Шаг 3: Вызываем функцию GetMeasurementData() и возвращаемые данные присваиваем к переменной s Шаг 4: s=[1,2,3]; mass=s
Входные данные	-
Ожидаемый результат	Система прекращает работу и выводим в log файл информациию об ошибке. Вывод ошибки «С лидара был получен массив переменных int, а не tuple.»
Тип	Негативный

ID Теста	И4
Объект тестирования	Связь интеграции №1
Цель тестирования	Проверка передачи некорректных данных с лидара между модулем lidarClass.py и модулем Client.py
Описание тестирования	Тесты для данной интеграции проводят по след. сценарию: Шаг 1: В Rasspbery PI4 подключаем лидар по порту /dev/ttyACM0 и включаем робота Шаг 2: В файле Client.py импортируем Класс Lidar(), создаем объект класса и передаем в конструктор класса порт /dev/ttyACM0(соответствующий лидару) на устройстве Rasspbery PI4 Шаг 3: Вызываем функцию GetMeasurementData() и возвращаемые данные присваиваем к переменной s Шаг 4: s=[(0,1),(1,0.0)]; mass=s
Входные данные	-
Ожидаемый результат	Система прекращает работу и выводим в log файл информациию об ошибке. Вывод ошибки «С лидара был получен массив кортежей с переменными типа int, а не типа float»
Тип	Негативный

ID Теста	И5
Объект тестирования	Связь интеграции №2
Цель тестирования	Проверка передачи данных через сокеты между модулем Server.py и Client.py
Описание тестирования	Тесты для данной интеграции проводят по след. сценарию: Шаг 1: На внешнем устройстве в файле создаем объекта класса Server(127.0.0.1), при инициализации класса создается сервер с ip 127.0.0.2 Шаг 2: Запускаем файл Client.py на роботе в котором подключаемся к серверу client_socket.connect(('127.0.0.2', 12345)) и отправляем данные из массива mass (переменная содержит данные указанные в файле во входных данных) client_socket.send(str(mass).encode()) Шаг 3: На внешнем устройстве после подключения клиента в файле вызываем функцию Data() (points_arr=server.Data())
Входные данные	файл с точками *тык*
Ожидаемый результат	На внешнем устройстве переменная(массив) points_arr заполнится кортежами(точками(дистанция, угол))типа Float в количестве 666
Тип	Позитивный

ID Теста	И6
Объект тестирования	Связь интеграции №2
Цель тестирования	Проверка случая, когда не получилось подключиться к сокету между модулем Server.py и Client.py
Описание тестирования	Тесты для данной интеграции проводят по след. сценарию: Шаг 1: На внешнем устройстве в файле создаем объекта класса Server(127.0.0.1), при инициализации класса создается сервер с ip 127.0.0.2 Шаг 2: Запускаем файл Client.py на роботе в котором подключаемся к серверу client_socket.connect(('123.2.1.2', 12345)) и отправляем данные из массива mass (переменная содержит данные указанные в файле во входных данных) client_socket.send(str(mass).encode()) Шаг 3: На внешнем устройстве после подключения клиента в файле вызываем функцию Data() (points_arr=server.Data())
Входные данные	-
Ожидаемый результат	Вывод сообщения пользователю на роботе «Не получилось подключиться к серверу для отправки данных»
Тип	Негативный

ID Теста	И7
Объект тестирования	Связь интеграции №4
Цель тестирования	Проверка корректности рассчета координат и запись информации о каждой точки в отдельнай объект класса point()
Описание тестирования	Тесты для данной интеграции проводят по след. сценарию: Шаг 1: Создаем массив для записи полученных данных points_arr=[] Шаг 2: Пробегаем по элементу в массиве points_arr (переменная содержит данные указанные в файле во входных данных) for i in points_arr: Шаг 3: Внутри цикла вызываем функцию GetCoordinates() — x,y=data.GetCoordinates(float(i[1]),float(i[0]),0,0,0) Шаг 4: Также внутри цикла полученные данные записываем в объект класса Point() — point = Point(x, y, i[1], i[0],0,0) Шаг 5: Также внутри цикла заполняем созданный массив в начале объектами класса Point() — points.append(point)
Входные данные	файл с точками *тык*
Ожидаемый результат	Переменная points содержит в себе элементы указанные в файле с точками *тык*
Тип	Позитивный

ID Теста	И8
Объект тестирования	Связь интеграции №5
Цель тестирования	Проверка корректности кластеризации препятствий и запись информации о каждом препятствии в отдельнай объект класса ObstacleData()
Описание тестирования	Шаг 1: переменная points содержит в себе данные указанные в файле во входных данных Шаг 2: dataset=Dataset(points,0,0,0) Шаг 3: datasets.append(dataset) Шаг 4: datasets.append(dataset) Шаг 5: obs = ObstacleClassifier() Шаг 6: datasets,moveX,moveY,yawI=obs.ICP(datasets) Шаг 7: obstacles = obs.GetObstaclesPoints(datasets[-1].Points)
Входные данные	файл с точками *тык*
Ожидаемый результат	Переменная obstacles содержит в себе содержит в себе элементы указанные в файле с точками *тык*
Тип	Позитивный

3.4 Нагрузочное тестирование

ID Теста	N1
Объект тестирования	Система полностью
Цель тестирования	Проверка времени работы программы
Описание тестирования	Нажать кнопку «Визуализация данных».
Входные данные	файл с точками в количестве 900 шт.
Ожидаемый результат	Карта отобразится в приложении менее чем за 1 секунду.
Тип	Позитивный

ID Теста	N2
Объект тестирования	Система полностью
Цель тестирования	Проверка времени работы программы
Описание тестирования	Нажать кнопку «Визуализация данных».
Входные данные	файл с точками в количестве 2000 шт.
Ожидаемый результат	Карта отобразится в приложении менее чем за 1.5 секунды.
Тип	Позитивный

ID Теста	N3
Объект тестирования	Система полностью
Цель тестирования	Проверка времени работы программы
Описание тестирования	Нажать кнопку «Визуализация данных».
Входные данные	файл с точками в количестве 4000 шт.
Ожидаемый результат	Карта отобразится в приложении менее чем за 2 секунды.
Тип	Позитивный

4 Журнал тестирования

ID	Попытка	Описание	Ожидаемый результат	Фактический результат	Результат
Б1	1	Проверка возможности получения данных с лидара	Переменная mass заполнится более чем шестью-стами точками(дистанция, угол) считанных с лидара	Переменная mass имеет 682 точки считанных с лидара, что удовлетворяет ожидаемому результату	Пройден
Б2	1	Проверка случая, когда при считывание данных с лидара произошла ошибка.	Вывод сообщения пользователю «Неверный порт» и команда завершается	В консоли у пользователя отобрарилось сообщение «Неверный порт»	Пройден
Б3	1	Проверка возможности установки частоты считывание данных с лидара.	Переменная timer больше 10-ти секунд, но меньше 11	Переменная timer имеет значение 10.116515398025513 что удовлетворяет ожидаемому результату	Пройден
Б4	1	Проверка случая, когда методу были переданны не допустимые значения.	Вывод сообщения пользователю «Частота считывания должна быть числовым и положительным значением»	В консоли у пользователя отобрарилось сообщение «Частота считывания должна быть числовым и положительным значением»	Пройден
Б5	1	Проверка случая, когда методу были переданны не допустимые значения.	Вывод сообщения пользователю «Частота считывания должна быть числовым и положительным значением»	В консоли у пользователя отобрарилось сообщение «Частота считывания должна быть числовым и положительным значением»	Пройден

ID	Попытка	Описание	Ожидаемый результат	Фактический результат	Результат
Б6	1	Проверка открытия сервера для передачи данных.	В консоль выведется сообщение «Сервер запущен и ожидает подключений...»	В консоли у пользователя отобразилось сообщение «Сервер запущен и ожидает подключений...»	Пройден
Б7	1	Проверка случая, когда в ip сервера были переданы не верные данные.	Вывод сообщения пользователю «Введенный ip не может быть использован.»	socket.gaierror: [Errno 11001]	Не пройден Ошибка E1
Б7	2	Проверка случая, когда в ip сервера были переданы не верные данные.	Вывод сообщения пользователю «Введенный ip не может быть использован.»	В консоли у пользователя отобразилось сообщение «Введенный ip не может быть использован.»	Пройден
Б8	1	Проверка передачи данных через сокет на внешнее устройство.	Переменная point_arr будет содержать элементы [(2,3),(4,8)]	Переменная point_arr содержит элементы [(2,3),(4,8)]	Пройден
Б9	1	Проверка случая, когда не получилось подключиться к сокету.	Вывод сообщения пользователю «Не получилось подключиться к роботу для считывания данных»	ConnectionRefusedError: [WinError 10061]	Не пройден Ошибка E2
Б9	2	Проверка случая, когда не получилось подключиться к сокету.	Вывод сообщения пользователю «Не получилось подключиться к роботу для считывания данных»	В консоли у пользователя отобразилось сообщение «Не получилось подключиться к роботу для считывания данных.»	Пройден
Б10	1	Проверка правильности расчета координат для точек считанных с лидара.	Переменная coordinates будет иметь два значения X==353.18 Y==187.79	Переменная coordinates имеет два значения (X==353.18, Y==187.79)	Пройден

ID	Попытка	Описание	Ожидаемый результат	Фактический результат	Результат
Б11	1	Проверка случая, когда в метод были переданы не корректные данные.	Вывод сообщения пользователю «В метод GetCoordinates() были переданы аргумент/ы не числового типа.»	TypeError: can't multiply sequence by non-int of type 'float'	Не пройден Ошибка Е3
Б11	2	Проверка случая, когда в метод были переданы не корректные данные.	Вывод сообщения пользователю «В метод GetCoordinates() были переданы аргумент/ы не числового типа.»	В консоли у пользователя отобрарилось сообщение «В метод GetCoordinates() были переданы аргумент/ы не числового типа.»	Пройден
Б12	1	Проверка правильности расчета расстояния между препятствиями.	Метод вернет расстояние между точкми: dist1=0 и dist2= 10.829672201872034	Переменная dist1==0 и dist2== 10.829672201872034	Пройден
Б13	1	Проверка правильности расчета расстояния между всеми точками препятствий.	obstacle.Width== 10.829672201872034	Поле obstacle.Width в объекте класса == 10.829672201872034	Пройден
Б14	1	Проверка случая, когда в метод был передан не объект класса Point	Вывод сообщения пользователю «В метод GetWidth были переданны аргумента отличные от объекта класса Point»	AttributeError: 'int' object has no attribute 'X'	Не пройден Ошибка Е4

ID	Попытка	Описание	Ожидаемый результат	Фактический результат	Результат
Б14	2	Проверка случая, когда в метод был передан не объект класса Point	Вывод сообщения пользователю «В метод GetWidth были переданны аргумента отличные от объекта класса Point»	В консоли у пользователя отобразилось сообщение «В метод GetWidth были переданны аргумента отличные от объекта класса Point»	Пройден
Б15	1	Проверка случая, когда в метод был передан не объект класса ObstacleData	Вывод сообщения пользователю «В метод GetWidhtEveryPoint были переданны аргумента отличные от list содержащие объекты класса Point»	TypeError: object of type 'int' has no len()	Не пройден Ошибка Е5
Б15	2	Проверка случая, когда в метод был передан не объект класса ObstacleData	Вывод сообщения пользователю «В метод GetWidhtEveryPoint были переданны аргумента отличные от list содержащие объекты класса Point»	В консоли у пользователя отобразилось сообщение «В метод GetWidhtEveryPoint были переданны аргумента отличные от list содержащие объекты класс Point»	Пройден
Б16	1	Проверка правильности работы данного алгоритма(метода)	errorYaw<0.000001 errorMoveX<0.000001 errorMoveY<0.000001 Переменная mass должна содержать данные указанные в файле по ссылке *тык*	errorYaw==6.765257078782533e-15 errorMoveX==9.094947017729282e-13 errorMoveY==1.5916157281026244e-12 Переменная mass содержит данные указанные в файле по ссылке *тык*. Все переменные соответствуют требованиям	Пройден

ID	Попытка	Описание	Ожидаемый результат	Фактический результат	Результат
Б17	1	Проверка случая, когда в метод ICP() был передан не объект класса Datasets	Вывод сообщения пользователю «В метод ICP были переданы аргументы отличные от объекта класса Datasets»	TypeError: 'int' object is not subscriptable	Не пройден Ошибка E6
Б17	2	Проверка случая, когда в метод ICP() был передан не объект класса Datasets	Вывод сообщения пользователю «В метод ICP были переданны аргумента отличные от объекта класса Datasets»	В консоли у пользователя отобразилось сообщение «В метод ICP были переданны аргументы отличные от объекта класса Datasets»	Пройден
Б18	1	Проверка правильности работы данного алгоритма(метода GetObstaclesPoints())	len(obstacles)==2 len(obstacles[0]. Points) ==2 len(obstacles[1]. Points) ==1 obstacles[0].Color!= obstacles[1].Color	len(obstacles)==2 len(obstacles[0]. Points) ==2 len(obstacles[1]. Points) ==1 obstacles[0].Color== obstacles[1].Color (цвет у двух препятствий одинаковый, что не соответствует требованию теста)	Не пройден Ошибка E7
Б18	2	Проверка правильности работы данного алгоритма(метода GetObstaclesPoints()).	len(obstacles)==2 len(obstacles[0]. Points) ==2 len(obstacles[1]. Points) ==1 obstacles[0].Color!= obstacles[1].Color	len(obstacles)==2 len(obstacles[0]. Points) ==2 len(obstacles[1]. Points) ==1 obstacles[0].Color!= obstacles[1].Color	Пройден

ID	Попытка	Описание	Ожидаемый результат	Фактический результат	Результат
Б19	1	Проверка правильности отсеивания тех же точек(для эффективности работы алгоритма(метода GetObstaclesPoints)).	sum1!=sum2. sum1==1. sum1==2.	sum1!=sum2. sum1==1. sum1==2.	Пройден
Б20	1	Проверка случая, когда в метод был передан не массив объектов класса Point.	Вывод сообщения пользователю «В метод GetObstaclesPoints были переданны аргумент отличный от типа list с объектами класса Point»	TypeError: '<' not supported between instances of 'NoneType' and 'int'	Не пройден Ошибка E8
Б20	2	Проверка случая, когда в метод был передан не массив объектов класса Point.	Вывод сообщения пользователю «В метод GetObstaclesPoints были переданны аргумент отличный от типа list с объектами класса Point»	В консоли у пользователя отобрарилось сообщение «В метод GetObstaclesPoints были переданны аргумент отличный от типа list с объектами класса Point»	Пройден
Б21	1	Проверка правильности работы метода GetMergeObstacles объединение рядом стоящих препятствий.	sum1>sum2, sumPoint2> sumPoint1	sum1==2, sum2==1, sumPoint2==4, sumPoint1==2 (все переменные соответствуют требованию теста)	Пройден

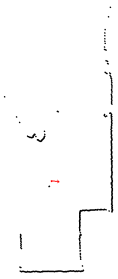
ID	Попытка	Описание	Ожидаемый результат	Фактический результат	Результат
Б22	1	Проверка случая, когда в метод GetMergeObstacles был передан не list содержащий объекты класса ObstacleData().	Вывод сообщения пользователю «В метод GetMergeObstacles были переданны аргумент отличный от list содержащий объекты класса ObstacleData»	AttributeError: 'int' object has no attribute 'Points'	Не пройден Ошибка Е9
Б22	2	Проверка случая, когда в метод GetMergeObstacles был передан не list содержащий объекты класса ObstacleData().	Вывод сообщения пользователю «В метод GetMergeObstacles были переданны аргумент отличный от list содержащий объекты класса ObstacleData»	В консоли у пользователя отобрарилось сообщение «В метод GetMergeObstacles были переданны аргумент отличный от list содержащий объекты класса ObstacleData»	Пройден
Б23	1	Проверка правильности работы метода FuncLoadKart().	len(obstacles) ==28	len(obstacles) ==28	Пройден
Б24	1	Проверка случая, когда указанный файл для загрузки пуст иили содержит не подходящие данные.	Вывод сообщения пользователю «Файл пуст или содержит не корректные данные»	ValueError: could not convert string to float: 'S'	Не пройден Ошибка Е10
Б25	1	Проверка случая, когда не был указан файл.	Вывод сообщения пользователю «Файл не был выбран»	No such file or directory: ''	Не пройден Ошибка Е11

ID	Попытка	Описание	Ожидаемый результат	Фактический результат	Результат
Б26	1	Проверка правильности работы метода FuncSaveKart().	len(obstacles1)==len(obstacles2) Количество препятсвтий в переменных одинаковое	len(obstacles1)==len(obstacles2) Количество препятсвтий в переменных было одинаковое и равное двум	Пройден
Б27	1	Проверка случая, когда при сохранении карты не был указан файл	Вывод сообщения пользователю «Имя файла для сохранения пусто»	был создан файл без названия	Не пройден Ошибка E12
Б28	1	Проверка случая, когда был указан существующий файл	Вывод сообщения пользователю «Указанное имя файла для сохранения уже существует»	был перезаписан существующий файл	Не пройден Ошибка E13
Б29	1	Проверка правильности работы метода IncreaseKoef()	Пременная koef==6	Пременная koef==6	Пройден
Б30	1	Проверка случая, когда был достигнут максимальный масштаб карты.	Пременная koef==10	Пременная koef==11	Не пройден Ошибка E14
Б30	2	Проверка случая, когда был достигнут максимальный масштаб карты.	Пременная koef==10	Пременная koef==10	Пройден
Б31	1	Проверка правильности работы метода DecreaseKoef().	Пременная koef==4	Пременная koef==4	Пройден

ID	Попытка	Описание	Ожидаемый результат	Фактический результат	Результат
Б32	1	Проверка случая, когда был достигнут минимальный масштаб карты.	Пременная koef==1	Пременная koef==1	Пройден
Б33	1	Проверка правильности работы метода FuncOn().	flag==0 and th2.is_alive()==1	flag==0 and th2.is_alive()==1	Пройден
Б34	1	Проверка правильности работы метода FuncOff().	flag==1 and th2.is_alive()==0	flag==1 and th2.is_alive()==0	Пройден
Б35	1	Проверка возможности смены частоты опроса лидара.	MRL=10 (частота опроса лидара в секунду увеличиться)	MRL=10 (частота опроса лидара в секунду увеличиться)	Пройден
Б36	1	Проверка случая, когда была переданная отрицательная частота опроса лидара	Вывод сообщения пользователю «Частота опроса лидара не может быть меньше 1»	MRL=-10	Не пройден Ошибка E15
Б37	1	Проверка случая, когда было переданно буквенное значения для частоты опроса лидара	Вывод сообщения пользователю «Частота опроса лидара не может иметь не числовое значение»	ValueError: invalid literal for int() with base 10: 'ss'	Не пройден Ошибка E16
Б38	1	Проверка возможности смены частоты отрисовки карты.	MRK=0.1 (частота отрисовки карты уменьшиться)	MRK=0.1 (частота отрисовки карты уменьшиться)	Пройден
Б39	1	Проверка возможности смены частоты отрисовки карты	Вывод сообщения пользователю «Частота опроса карты не может быть меньше 1»	MRL=0	Не пройден Ошибка E17

ID	Попытка	Описание	Ожидаемый результат	Фактический результат	Результат
Б40	1	Проверка возможности смены частоты отрисовки карты	Вывод сообщения пользователю «Частота опроса карты не может иметь не числовое значение»	ValueError: invalid literal for int() with base 10: 'ss'	Не пройден Ошибка E18

ID	Попытка	Описание	Ожидаемый результат	Фактический результат	Результат
A1	1	Успешное построение карты на основе полученных данных с лидара.			Пройден
A2	1	Проверка подключение или исправности лидара.	Вывод сообщения пользователю «Лидар не обнаружен системой»	Система вывело сообщение пользователю «Лидар не обнаружен системой»	Пройден
A3	1	Проверка подключения робота по сокету к внешнему устройству.	Вывод сообщения пользователю «Робот пропал из сети»	Система вывело сообщение пользователю «Робот пропал из сети»	Пройден
A4	1	Проверка возможности просмотра кол-во нанесенных препятствий на карту.	Вывод кол-во препятствий равное четырем в соответствующем поле.	Система вывела количество препятсвтий равное четырем.	Пройден
A5	1	Проверка возможности сохранение текущей карты в файл.	В файл сохранится построенная карта.	В файл сохранилась построенная карта.	Пройден
A6	1	Проверка возможности сохранение текущей карты в файл.	Вывод сообщения пользователю «Укажите имя файла для сохранение карты»	Вывод сообщения пользователю «Укажите имя файла для сохранение карты»	Пройден
A7	1	Проверка возможности сохранение текущей карты в файл.	Вывод сообщения пользователю «Указанный файл уже существует укажите другой»	Вывод сообщения пользователю «Указанный файл уже существует укажите другой»	Пройден

ID	Попытка	Описание	Ожидаемый результат	Фактический результат	Результат
A8	1	Проверка возможности загрузки карты из файла.	Загрузится карта указанная ниже 	Загрузилась карта указанная ниже 	Пройден
A9	1	Проверка возможности загрузки карты из файла.	Вывод сообщения пользователю «Укажите имя файла для загрзуки карты»	Вывод сообщения пользователю «Укажите имя файла для загрзуки карты»	Пройден
A10	1	Проверка загрузки карты из файла.	Вывод сообщения пользователю «Указанного файла не существует укажите другой»	Вывод сообщения пользователю «Указанного файла не существует укажите другой»	Пройден
A11	1	Проверка возможности изменение масштаба.	Карта отдаляется и приближается соответственно нажатим клвишам.	На кнопку «+» карту приближается, на кнопку «-» отдаляется, что соответствует требованиям	Пройден
A12	1	Проверка возможности остановки визуализации карты.	Визуализация карты остонавливается и на экране отображается текущая уже построенная карта.	На кнопку «Остановить визуализация» - остонавливается, на кнопку «Возобновить визуализацию» - возобновляется , что соответствует требованиям	Пройден

ID	Попытка	Описание	Ожидаемый результат	Фактический результат	Результат
A13	1	Проверка возможности перемещение по карте.	Перемещение по карте.	Пользователь нажимает стрелки «Вверх», «Вниз» карта перемещается по оси Y и нажимает стрелки «Влево»,«Вправо» карта перемещается по оси X, что соответствует требованиям	Пройден
A14	1	Проверка возможность определения длины периметра препятствия.	Вывод длины периметра равной 245-248 в соответствующее поле	Система вывела длину периметра равной 247, что соответствует тербованию	Пройден
A15	1	Проверка возможности изменять частоту опроса лидара и частоту обновления карты.	Новые препятствия будут появляться на карте каждую секунду.	Обновление карты и отрисовка новых препятствий при движении тележки обновляется раз в секунду	Пройден
A16	1	Проверка возможности изменять частоту опроса лидара и частоту обновления карты.	Вывод сообщения пользователю «Частота опрса должна быть положительной и больше 0.1».	Вывод сообщения пользователю «Частота опрса должна быть положительной и больше 0.1»	Пройден
A17	1	Проверка возможности изменять частоту опроса лидара и частоту обновления карты.	Вывод сообщения пользователю «Поле для частоты опроса не должно быть пустое».	Вывод сообщения пользователю «Поле для частоты опроса не должно быть пустое»	Пройден

ID	Попытка	Описание	Ожидаемый результат	Фактический результат	Результат
А18	1	Проверка возможности просмотра расстояние до препятствия относительно текущего положения робота.	В соответствующем поле выводится расстояние 68-72мм.	В соответствующем поле отобразилось расстояние 69мм до препятствия, что соответствует требованиям	Пройден
N1	1	Проверка времени работы программы.	Карта отобразится в приложении менее чем за 1 секунду.	Карта отобразилась в приложении за 0.95 секунды	Пройден
N2	1	Проверка времени работы программы.	Карта отобразится в приложении менее чем за 1.5 секунды.	Карта отобразилась в приложении за 1.47 секунды	Пройден
N3	1	Проверка времени работы программы.	Карта отобразится в приложении менее чем за 2 секунды.	Карта отобразилась в приложении за 8.34 секунды.	Не пройден Ошибка E19
И1	1	Проверка передачи данных с лидара между модулем lidarClass.py и модулем Client.py.	Переменная mass в файле Client.py заполнится более 600 точками(дистанция, угол) типа Float считанных с лидара.	Переменная mass в файле Client.py заполнилась 666 точками(дистанция, угол) типа Float считанных с лидара.	Пройден
И2	1	Проверка передачи некорректных данных с лидара между модулем lidarClass.py и модулем Client.py.	Система прекращает работу и выводи в log файл информациию об ошибке. Вывод ошибки «С лидара было полученно значение типа int, а не list».	Система прекращает работу и выводи в log файл информациию об ошибке. Вывод ошибки «С лидара было полученно значение типа int, а не list».	Пройден

И3	1	Проверка передачи некорректных данных с лидара между модулем lidarClass.py и модулем Client.py.	Система прекращает работу и выводи в log файл информациию о ошибке. Вывод ошибки «С лидара был получен массив переменных int, а не tuple.».	Система прекращает работу и выводи в log файл информациию об ошибке. Вывод ошибки «С лидара был получен массив переменных int, а не tuple.».	Пройден
И4	1	Проверка передачи некорректных данных с лидара между модулем lidarClass.py и модулем Client.py.	Система прекращает работу и выводим в log файл информациию об ошибке. Вывод ошибки «С лидара был получен массив кортежей с переменными типа int, а не типа float».	Система прекращает работу и выводим в log файл информациию об ошибке. Вывод ошибки «С лидара был получен массив кортежей с переменными типа int, а не типа float».	Пройден
И5	1	Проверка передачи данных через сокеты между модулем Server.py и Client.py.	На внешнем устройстве переменная(массив) points_arr заполнится кортежами(точками (дистанция, угол))типа Float в количестве 666.	Переменная points_arr заполнилась кортежами(точками (дистанция, угол))типа Float в количестве 666.	Пройден

ID	Попытка	Описание	Ожидаемый результат	Фактический результат	Результат
И6	1	Проверка случая, когда не получилось подключиться к сокету между модулем Server.py и Client.py.	Вывод сообщения пользователю на роботе «Не получилось подключиться к серверу для отправки данных»	TimeoutError: [WinError 10060].	Не пройден Ошибка E20
И7	1	Проверка корректности рассчета координат и запись информации о каждой точки в отдельнай объект класса point().	Переменная points содержит в себе элементы указанные в файле с точками *тык*	Переменная points соответствует ожидаемому результату	Пройден
И8	1	Проверка корректности кластеризации препятствий и запись информации о каждом препятствии в отдельнай объект класса ObstacleData().	Переменная obstacles содержит в себе содержит в себе элементы указанные в файле с точками *тык*	Переменная obstacles соответствует ожидаемому результату	Пройден

5 Журнал ошибок

Ошибка E1

1. Тест: Б7
2. Цель теста: Проверка случая, когда в ip сервера были переданы не верные данные.
3. Входные данные: server=Server('121')
4. Ожидаемый результат: Вывод сообщения пользователю «Введенный ip не может быть использован.»
5. Фактический результат: socket.gaierror: [Errno 11001].
6. Исправление: Добавить проверку, если в класс Server() передается ip, который не может быть использован.

Ошибка E2

1. Тест: Б9
2. Цель теста: Проверка случая, когда не получилось подключиться к сокету.
3. Входные данные:
   (a) server.Server(127.0.0.1)
   (b) На клиенте запускаем файл Client.py в файле указан не правильны ip, а именно(127.0.0.2) для подключения
4. Ожидаемый результат: Вывод сообщения пользователю «Не получилось подключиться к роботу для считывания данных»
5. Фактический результат: ConnectionRefusedError: [WinError 10061].
6. Исправление: Добавить проверку, если на клиенте было указан ошибочный ip адрес.

Ошибка E3

1. Тест: Б11
2. Цель теста: Проверка случая, когда в метод были переданы не корректные данные.
3. Входные данные:

(a) Создаем экземпляр класса data=DataHandler().

(b) coordinates=data.GetCoordinates('asdas', 28, 0,0,0)

4. Ожидаемый результат: Вывод сообщения пользователю «В метод GetCoordinates() были переданы аргумент/ы не числового типа.»

5. Фактический результат: TypeError: can't multiply sequence by non-int of type 'float'.

6. Исправление: Добавить проверку, если в метод передан не допустимый тип переменной.

Ошибка Е4

1. Тест: Б14

2. Цель теста: Проверка случая, когда в метод был передан не объект класса Point.

3. Входные данные:

(a) Создаем экземпляр класса obs=ObstacleClassifier()

(b) dist1=obs.GetWidth(1,2)

4. Ожидаемый результат: Вывод сообщения пользователю «В метод GetWidth были переданны аргумента отличные от объекта класса Point»

5. Фактический результат: AttributeError: 'int' object has no attribute 'X'.

6. Исправление: Добавить проверку, если в метод передан не допустимый тип переменной.

Ошибка Е5

1. Тест: Б15

2. Цель теста: Проверка случая, когда в метод был передан не объект класса ObstacleData.

3. Входные данные:

(a) Создаем экземпляр класса obs=ObstacleClassifier()

(b) dist1=obs.GetWidhtEveryPoint(1)

4. Ожидаемый результат: Вывод сообщения пользователю «В метод GetWidhtEveryPoint были переданны аргумента отличные от list содержащие объекты класса Point»

5. Фактический результат: TypeError: object of type 'int' has no len()

6. Исправление: Добавить проверку, если в метод передан не допустимый тип переменной.

Ошибка E6

1. Тест: Б17

2. Цель теста: Проверка случая, когда в метод ICP() был передан не объект класса Datasets.

3. Входные данные:

   (a) obs=ObstacleClassifier()

   (b) obs.ICP(1)

4. Ожидаемый результат: Вывод сообщения пользователю «В метод ICP были переданны аргументы отличные от объекта класса Datasets»

5. Фактический результат: TypeError: 'int' object is not subscriptable

6. Исправление: Добавить проверку, если в метод передан не допустимый тип переменной.

Ошибка E7

1. Тест: Б18

2. Цель теста: Проверка правильности работы данного алгоритма(метода GetObstaclesPoints(

3. Входные данные:

   (a) obs=ObstacleClassifier()

   (b) obstacles=obs.GetObstaclesPoints([Point(-50.15,151.94,160,108.27,0,0), Point(-20.15,151.
   Point(200.15,151.94,160,108.27,0,0)]))

4. Ожидаемый результат: len(obstacles)==2 len(obstacles[0]. Points) ==2 len(obstacles[1]. Points) ==1 obstacles[0].Color!= obstacles[1].Color

5. Фактический результат: len(obstacles)==2 len(obstacles[0]. Points) ==2 len(obstacles[1]. Points) ==1 obstacles[0].Color== obstacles[1].Color (цвет у двух препятствий одинаковый, что не соответствует требованию теста)

6. Исправление: Переработать присвоение цвета к препятствию.

Ошибка E8

1. Тест: Б20
2. Цель теста: Проверка случая, когда в метод был передан не массив объектов класса Point.
3. Входные данные:
   (a) obs=ObstacleClassifier()
   (b) obstacles=obs.GetObstaclesPoints(1,2)
4. Ожидаемый результат: Вывод сообщения пользователю «В метод GetObstaclesPoints были переданны аргумент отличный от типа list с объектами класса Point»
5. Фактический результат: TypeError: '<' not supported between instances of 'NoneType' and 'int'
6. Исправление: Добавить проверку, если в метод передан не допустимый тип переменной.

Ошибка E9

1. Тест: Б22
2. Цель теста: Проверка случая, когда в метод GetMergeObstacles был передан не list содержащий объекты класса ObstacleData().
3. Входные данные:
   (a) obs=ObstacleClassifier()
   (b) obstacles=obs.GetObstaclesPoints(1,2)
4. Ожидаемый результат: Вывод сообщения пользователю «В метод GetMergeObstacles были переданны аргумент отличный от list содержащий объекты класса ObstacleData»
5. Фактический результат: AttributeError: 'int' object has no attribute 'Points'
6. Исправление: Добавить проверку, если в метод передан не допустимый тип переменной.

Ошибка E10

1. Тест: Б24
2. Цель теста: Проверка случая, когда указанный файл для загрузки пуст или содержит не подходящие данные.
3. Входные данные:
   (a) namefile="B24.txt"Файл содержит следующие данные (S,S,S,A,ASDA)
   (b) obstacles=FuncLoadKart(namefile)
4. Ожидаемый результат: Вывод сообщения пользователю «Файл пуст или содержит не корректные данные»
5. Фактический результат: ValueError: could not convert string to float: 'S'
6. Исправление: Добавить проверку, если мы пытаемся загрузить данные из пустого файла или содержащий не корректные данные.

Ошибка E11

1. Тест: Б25
2. Цель теста: Проверка случая, когда не был указан файл.
3. Входные данные:
   (a) namefile=
   (b) obstacles=FuncLoadKart(namefile)
4. Ожидаемый результат: Вывод сообщения пользователю «Файл не был выбран»
5. Фактический результат: No such file or directory: ''
6. Исправление: Добавить проверку, если пользователь не указал название файла или файл не существует.

Ошибка E12

1. Тест: Б27
2. Цель теста: Проверка случая, когда при сохранении карты не был указан файл.

3. Входные данные:

   (a) obstacles=[]

   (b) savefile=

   (c) FuncSaveKart(savefile,obstacles)

4. Ожидаемый результат: Вывод сообщения пользователю «Имя файла для сохранения пусто»

5. Фактический результат: был создан файл без названия

6. Исправление: Добавить проверку, если пользователь не указал название файла при сохранении карты.

Ошибка E13

1. Тест: Б28

2. Цель теста: Проверка случая, когда был указан существующий файл.

3. Входные данные:

   (a) obstacles=[]

   (b) savefile="B28.txt"(файл B28.txt уже создан в директории)

   (c) FuncSaveKart(savefile,obstacles)

4. Ожидаемый результат: Вывод сообщения пользователю «Указанное имя файла для сохранения уже существует»

5. Фактический результат: был перезаписан существующий файл

6. Исправление: Добавить проверку, если пользователь указал существующий файл при сохранении карты.

Ошибка E14

1. Тест: Б30

2. Цель теста: Проверка случая, когда был достигнут максимальный масштаб карты.

3. Входные данные:

(a) global koef=10

(b) koef=IncreaseKoef(koef)

4. Ожидаемый результат: koef==10 (масштаб не изменится)

5. Фактический результат: koef==11

6. Исправление: Добавить проверку, если пользователь достиг максимального масштаба карты.

Ошибка E15

1. Тест: Б36

2. Цель теста: Проверка случая, когда была переданная отрицательная частота опроса лидара.

3. Входные данные:

   (a) global MRL=1

   (b) MRL=FuncMeasurementRateLidar(-10)

4. Ожидаемый результат: Вывод сообщения пользователю «Частота опроса не может быть меньше 1»

5. Фактический результат: MRL=-10

6. Исправление: Добавить проверку, если пользователь ввел значение меньше 1.

Ошибка E16

1. Тест: Б37

2. Цель теста: Проверка случая, когда было переданно буквенное значения для частоты опроса лидара.

3. Входные данные:

   (a) global MRL=1

   (b) MRL=FuncMeasurementRateLidar("ss")

4. Ожидаемый результат: Вывод сообщения пользователю «Частота опроса не может иметь не числовое значение»

5. Фактический результат: ValueError: invalid literal for int() with base 10: 'ss'

6. Исправление: Добавить проверку, если пользователь ввел буквенное значение (не числовое).

Ошибка E17

1. Тест: Б39

2. Цель теста: Проверка возможности смены частоты отрисовки карты.

3. Входные данные:

   (a) global MRK=1

   (b) MRK=FuncMeasurementRateKart(0)

4. Ожидаемый результат: Вывод сообщения пользователю «Частота опроса карты не может быть меньше 1»

5. Фактический результат: MRL=0

6. Исправление: Добавить проверку, если пользователь ввел значение меньше 1.

Ошибка E18

1. Тест: Б40

2. Цель теста: Проверка возможности смены частоты отрисовки карты.

3. Входные данные:

   (a) global MRK=1

   (b) MRK=FuncMeasurementRateKart("sadea")

4. Ожидаемый результат: Вывод сообщения пользователю «Частота опроса карты не может иметь не числовое значение»

5. Фактический результат: ValueError: invalid literal for int() with base 10: 'sadea'

6. Исправление: Добавить проверку, если пользователь ввел буквенное значение (не числовое).

Ошибка E19

1. Тест: N3
2. Цель теста: Проверка времени работы программы.
3. Входные данные:
   (a) файл с точками в количестве 4000 шт.
4. Ожидаемый результат: Карта отобразится в приложении менее чем за 2 секунды.
5. Фактический результат: Карта отобразилась в приложении за 8.34 секунды
6. Исправление: Необходимо ускорить работу программы для 3000 и более точек.

Ошибка E20

1. Тест: И6
2. Цель теста: Проверка случая, когда не получилось подключиться к сокету между модулем Server.py и Client.py.
3. Входные данные: -
4. Ожидаемый результат: Вывод сообщения пользователю на роботе «Не получилось подключиться к серверу для отправки данных»
5. Фактический результат: TimeoutError: [WinError 10060] Попытка установить соединение была безуспешной, т.к. от другого компьютера за требуемое время не получен нужный отклик, или было разорвано уже установленное соединение из-за неверного отклика уже подключенного компьютера
6. Исправление: Необходимо добавить проверку, если попытка подключиться к серверу(внешнему устройству) была безуспешной.

6 Примеры тестов

```
def testB21():
    obstacles=[]
    obs=ObstacleClassifier()
    obstacle1=ObstacleData([Point(-50.15,151.94,160,108.27,0,0),Point(-50.15,151.94,160,108.27,0,0)],"Black",0)
    obstacle2=ObstacleData([Point(-50.15,151.94,160,108.27,0,0),Point(-50.15,151.94,160,108.27,0,0)],"Black",0)
    obstacles.append(obstacle1)
    obstacles.append(obstacle2)
    sum1=len(obstacles)
    sumPoint1=len(obstacles[0].Points)
    obstacles=obs.GetMergeObstacles(obstacles)
    sum2=len(obstacles)
    sumPoint2=len(obstacles[0].Points)
    assert sum1>sum2
    assert sumPoint2>sumPoint1
```

Рис. 3 – Блочный тест №21, позитивный.

```
def testB2(capsys):
    with pytest.raises(SystemExit) as pytest_wrapped_e:
        lidar = Lidar('COM3', 19200)
    out = capsys.readouterr()
    assert pytest_wrapped_e.type == SystemExit
    assert out.out=="Неверный порт!\n"
```

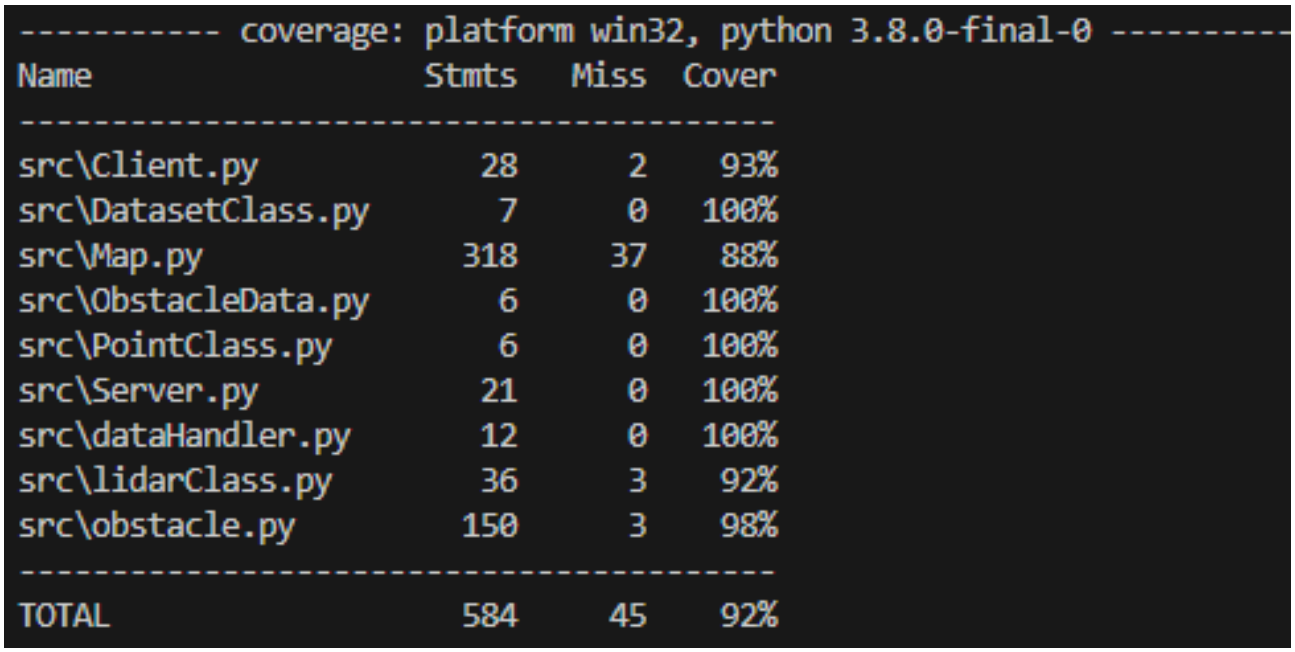
Рис. 4 – Блочный тест №2, негативный.

```
def testI8():
    points_arr=readfile("17_1")
    rows=[]
    datasets=[]
    for i in points_arr:
        x,y=data.GetCoordinates(float(i[1]),float(i[0]),yaw,centerX,centerY)
        point = Point(x, y, i[1], i[0],0,0)
        rows.append(point)
    dataset=Dataset(rows,0,0,0)
    datasets.append(dataset)
    datasets.append(dataset)
    datasets,movex,movey,yawl=obs.ICP(datasets)
    obstacles = obs.GetObstaclesPoints(datasets[-1].Points)
    obstacles2 = [ObstacleData([Point(999.75,2370.5,0,0,0,0),Point(999.76,2395.79,0,0,0,0),Point(996.59,2414.61,0,0,0,0),Point(1002.68,2430.43,0,0,0,0),Point(1004.02,2456.0,0,0,0,0),Point(1000.35,2477.73,0,0,0,0),Point(1006.63,2495.44,0,0,0,0),Point(1001.36,2521.82,0,0,0,0),
    Point(1002.41,2552.28,0,0,0,0),Point(1001.64,2588.32,0,0,0,0),Point(1011.41,2602.59,0,0,0,0),],'DarkSlateGray',0),ObstacleData([Point(1463.28,1588.7,0,0,0,0),],'Black',0),ObstacleData([Point(987.0,2907.42,0,0,0,0),Point(1004.34,2913.62,0,0,0,0),],'SlateGrey',0),
    ObstacleData([Point(993.2,2974.33,0,0,0,0),Point(1011.36,2980.44,0,0,0,0),Point(1032.61,2976.8,0,0,0,0),Point(1051.66,2979.56,0,0,0,0),Point(1070.16,2984.0,0,0,0,0),Point(1088.43,2989.17,0,0,0,0),Point(1110.58,2979.61,0,0,0,0),Point(1129.76,2980.41,0,0,0,0),Point(1147.99,
    2984.84,0,0,0,0),Point(1166.48,2988.08,0,0,0,0),Point(1187.27,2980.33,0,0,0,0),Point(1206.46,2979.16,0,0,0,0),Point(1225.03,2980.68,0,0,0,0),Point(1244.44,2977.05,0,0,0,0),Point(1262.77,2979.1,0,0,0,0),Point(1281.4,2978.93,0,0,0,0),Point(1298.52,2989.43,0,0,0,0),Point
    (1317.58,2985.84,0,0,0,0),Point(1336.62,2981.02,0,0,0,0),Point(1355.03,2980.92,0,0,0,0),Point(1373.83,2975.62,0,0,0,0),Point(1392.12,2975.06,0,0,0,0),Point(1410.26,2976.27,0,0,0,0),Point(1428.41,2977.27,0,0,0,0),Point(1446.82,2971.04,0,0,0,0),Point(1464.74,2978.58,0,0,0,
    0),Point(1482.91,2978.9,0,0,0,0),Point(1501.09,2979.0,0,0,0,0),Point(1519.28,2979.87,0,0,0,0),Point(1537.47,2979.53,0,0,0,0),Point(1555.68,2979.95,0,0,0,0),Point(1573.87,2979.16,0,0,0,0),Point(1591.79,2974.14,0,0,0,0),Point(1609.98,2973.9,0,0,0,0),Point(1628.08,2972.44,0,
    0,0,0),Point(1646.17,2970.75,0,0,0,0),Point(1664.79,2971.82,0,0,0,0),Point(1683.51,2976.04,0,0,0,0),Point(1702.59,2981.21,0,0,0,0),Point(1721.22,2981.58,0,0,0,0),Point(1738.91,2975.78,0,0,0,0),Point(1758.25,2979.63,0,0,0,0),Point(1776.42,2976.35,0,0,0,0),Point(1794.35,
    2971.86,0,0,0,0),Point(1812.62,2969.11,0,0,0,0),Point(1832.41,2972.96,0,0,0,0),Point(1850.95,2970.71,0,0,0,0),Point(1870.22,2971.13,0,0,0,0),Point(1890.83,2976.14,0,0,0,0),Point(1909.75,2974.11,0,0,0,0),Point(1929.79,2975.69,0,0,0,0),Point(1948.48,2972.21,0,0,0,0),Point
    (1970.78,2979.92,0,0,0,0),Point(1989.25,2974.98,0,0,0,0),Point(2008.32,2971.69,0,0,0,0),Point(2022.99,2955.92,0,0,0,0),],'NavyBlue',0),ObstacleData([Point(2008.2,2887.88,0,0,0,0),],'CornflowerBlue',0),ObstacleData([Point(1976.86,2777.93,0,0,0,0),Point(1975.47,2750.67,0,0,
    0,0),Point(1974.48,2725.34,0,0,0,0),Point(1972.09,2697.29,0,0,0,0),Point(1967.48,2644.18,0,0,0,0),Point(1965.54,2600.59,0,0,0,0),Point(1965.4,2563.64,0,0,0,0),Point(1966.23,2547.97,0,0,0,0),Point(1964.84,2527.77,0,0,0,0),Point(1964.88,2511.27,0,0,0,0),Point(1961.4,2472.05,
    0,0,0,0),Point(1959.12,2452.08,0,0,0,0),Point(1958.36,2435.77,0,0,0,0),Point(1965.77,2422.04,0,0,0,0),Point(1962.75,2402.25,0,0,0,0),Point(1967.27,2384.12,0,0,0,0),Point(1980.58,2370.1,0,0,0,0),Point(1971.26,2340.96,0,0,0,0),Point(1961.09,2311.1,0,0,0,0),Point(1964.46,
    2294.15,0,0,0,0),Point(1959.76,2264.39,0,0,0,0),Point(1960.18,2244.22,0,0,0,0),Point(1962.32,2227.54,0,0,0,0),Point(1961.93,2207.56,0,0,0,0),Point(1963.24,2189.38,0,0,0,0),Point(1963.87,2171.76,0,0,0,0),Point(1959.99,2158.86,0,0,0,0),Point(1964.81,2132.6,0,0,0,0),Point
    (1965.47,2109.61,0,0,0,0),Point(1963.06,2091.25,0,0,0,0),Point(1962.77,2076.16,0,0,0,0),Point(1965.63,2058.32,0,0,0,0),Point(1962.75,2034.52,0,0,0,0),Point(1964.51,2016.97,0,0,0,0),Point(1967.14,2001.6,0,0,0,0),Point(1970.04,1985.84,0,0,0,0),Point(1989.54,1987.68,0,0,0,0),
    Point(2005.73,1979.63,0,0,0,0),Point(2020.39,1975.62,0,0,0,0),Point(2039.63,1981.17,0,0,0,0),Point(2056.76,1978.31,0,0,0,0),Point(2070.87,1972.43,0,0,0,0),Point(2086.93,1979.67,0,0,0,0),Point(2112.17,1981.75,0,0,0,0),Point(2130.51,1977.63,0,0,0,0),Point(2145.68,1970.64,0,
    0,0,0),Point(2167.72,1974.25,0,0,0,0),Point(2190.39,1971.45,0,0,0,0),Point(2207.78,1970.68,0,0,0,0),Point(2224.43,1969.01,0,0,0,0),Point(2248.75,1971.8,0,0,0,0),Point(2263.31,1961.54,0,0,0,0),Point(2285.89,1968.62,0,0,0,0),Point(2301.97,1964.95,0,0,0,0),Point(2324.18,1964.
    38,0,0,0,0),Point(2341.0,1967.08,0,0,0,0),Point(2356.14,1968.62,0,0,0,0),Point(2373.31,1964.17,0,0,0,0),Point(2392.98,1967.61,0,0,0,0),Point(2442.94,1986.41,0,0,0,0),Point(2485.13,2000.53,0,0,0,0),Point(2494.19,1982.29,0,0,0,0),Point(2496.56,1960.94,0,0,0,0),Point(2487.55,
    1942.12,0,0,0,0),Point(2487.25,1920.32,0,0,0,0),Point(2491.12,1900.55,0,0,0,0),Point(2482.46,1876.13,0,0,0,0),Point(2489.23,1857.96,0,0,0,0),Point(2488.09,1837.88,0,0,0,0),Point(2486.51,1816.36,0,0,0,0),Point(2487.39,1796.68,0,0,0,0),Point(2488.84,1777.35,0,0,0,0),Point
    (2491.85,1758.57,0,0,0,0),Point(2487.7,1738.13,0,0,0,0),Point(2488.02,1719.02,0,0,0,0),Point(2491.86,1700.77,0,0,0,0),Point(2486.56,1680.83,0,0,0,0),Point(2487.76,1662.29,0,0,0,0),Point(2493.55,1644.48,0,0,0,0),Point(2492.08,1625.64,0,0,0,0),Point(2488.26,1606.69,0,0,0,0),
    Point(2487.07,1588.18,0,0,0,0),Point(2488.53,1569.97,0,0,0,0),Point(2486.65,1551.58,0,0,0,0),Point(2477.44,1533.05,0,0,0,0),Point(2488.88,1515.19,0,0,0,0),Point(2493.0,1496.95,0,0,0,0),Point(2489.77,1478.71,0,0,0,0),Point(2489.21,1460.47,0,0,0,0),Point(2480.33,1442.71,0,0,
    0,0),Point(2479.1,1424.64,0,0,0,0),Point(2484.52,1405.94,0,0,0,0),Point(2487.58,1387.24,0,0,0,0),Point(2487.32,1368.79,0,0,0,0),Point(2489.68,1349.86,0,0,0,0),Point(2486.74,1331.65,0,0,0,0),Point(2483.47,1313.49,0,0,0,0),Point(2482.8,1294.77,0,0,0,0),Point(2485.68,1275.14,
    0,0,0,0),Point(2485.25,1256.04,0,0,0,0),Point(2487.34,1236.11,0,0,0,0),Point(2482.31,1217.96,0,0,0,0),Point(2487.46,1196.67,0,0,0,0),Point(2481.7,1178.52,0,0,0,0),Point(2488.09,1162.92,0,0,0,0),Point(2479.82,1145.45,0,0,0,0),Point(2486.2,1122.44,0,0,0,0),Point(2484.63,
    1102.08,0,0,0,0),Point(2482.65,1081.64,0,0,0,0),Point(2477.51,1062.37,0,0,0,0),Point(2480.23,1046.62,0,0,0,0),Point(2484.51,1029.86,0,0,0,0),Point(2482.83,1008.21,0,0,0,0),Point(2476.25,988.79,0,0,0,0),Point(2482.22,970.18,0,0,0,0),Point(2481.75,954.75,0,0,0,0),Point(2484.
    53,937.27,0,0,0,0),Point(2480.0,915.63,0,0,0,0),Point(2483.82,896.84,0,0,0,0),Point(2481.41,881.59,0,0,0,0),Point(2477.94,866.86,0,0,0,0),Point(2476.74,850.46,0,0,0,0),Point(2485.2,827.23,0,0,0,0),Point(2487.51,807.71,0,0,0,0),Point(2481.36,793.87,0,0,0,0),Point(2480.65,
    775.92,0,0,0,0),Point(2484.43,754.26,0,0,0,0),Point(2482.1,736.7,0,0,0,0),Point(2479.1,719.7,0,0,0,0),Point(2475.63,702.66,0,0,0,0),Point(2484.15,675.32,0,0,0,0),Point(2476.98,660.68,0,0,0,0),Point(2483.85,633.54,0,0,0,0),Point(2482.78,612.8,0,0,0,0),Point(2484.28,589.24,
    0,0,0,0),Point(2484.62,566.2,0,0,0,0),Point(2483.11,544.42,0,0,0,0),Point(2481.21,522.52,0,0,0,0),Point(2488.56,503.02,0,0,0,0),Point(2484.58,482.32,0,0,0,0),Point(2479.55,462.3,0,0,0,0),Point(2486.11,442.4,0,0,0,0),Point(2479.09,423.71,0,0,0,0),Point(2479.73,396.05,0,0,0,
    0),Point(2466.09,384.09,0,0,0,0),Point(2450.11,388.82,0,0,0,0),],'MediumSlateBlue',0),ObstacleData([Point(2401.33,432.66,0,0,0,0),Point(2383.23,440.97,0,0,0,0),Point(2370.14,430.19,0,0,0,0),Point(2365.01,409.39,0,0,0,0),],'Blue',0),ObstacleData([Point(2489.62,236.41,0,0,0,
    0),Point(2481.3,214.8,0,0,0,0),Point(2484.82,193.62,0,0,0,0),Point(2488.72,171.55,0,0,0,0),Point(2482.89,144.93,0,0,0,0),Point(2487.93,120.22,0,0,0,0),Point(2482.34,91.78,0,0,0,0),Point(2479.91,76.71,0,0,0,0),Point(2485.19,58.03,0,0,0,0),Point(2485.73,29.89,0,0,0,0),Point
    (2484.4,12.18,0,0,0,0),Point(2482.9,-5.56,0,0,0,0),Point(2488.25,-34.27,0,0,0,0),Point(2487.38,-53.82,0,0,0,0),Point(2485.25,-71.72,0,0,0,0),Point(2483.99,-91.35,0,0,0,0),Point(2486.17,-117.01,0,0,0,0),Point(2484.58,-159.97,0,0,0,0),Point(2484.97,-184.11,0,0,0,0),Point
    (2474.88,-213.48,0,0,0,0),Point(2458.75,-235.82,0,0,0,0),Point(2463.36,-269.79,0,0,0,0),],'DarkTurquoise',0),ObstacleData([Point(2485.88,-266.24,0,0,0,0),Point(2387.23,-283.51,0,0,0,0),],'DarkGreen',0),ObstacleData([Point(2428.66,-425.78,0,0,0,0),Point(2418.84,-465.14,0,0,
    0,0),Point(2393.82,-476.27,0,0,0,0),],'DarkOliveGreen',0),ObstacleData([Point(2394.88,-652.38,0,0,0,0),Point(2368.36,-663.22,0,0,0,0),Point(2367.52,-700.14,0,0,0,0),],'MediumSeaGreen',0),ObstacleData([Point(2376.37,-763.25,0,0,0,0),Point(2374.18,-799.44,0,0,0,0),Point
    (2372.96,-839.43,0,0,0,0),Point(2374.73,-888.89,0,0,0,0),Point(2373.9,-932.8,0,0,0,0),Point(2376.21,-987.17,0,0,0,0),],'LightSeaGreen',0),ObstacleData([Point(2431.1,-1195.73,0,0,0,0),],'LawnGreen',0),ObstacleData([Point(2464.86,-1350.11,0,0,0,0),Point(2463.39,-1404.39,0,0,
    0,0),],'Gold',0),ObstacleData([Point(2463.75,-1466.37,0,0,0,0),],'DarkGoldenrod',0),ObstacleData([Point(2464.3,-1531.32,0,0,0,0),Point(2460.81,-1585.88,0,0,0,0),],'IndianRed',0),ObstacleData([Point(2464.86,-1667.3,0,0,0,0),],'SaddleBrown',0),ObstacleData([Point(2462.22,
    -1729.71,0,0,0,0),Point(2442.36,-1735.56,0,0,0,0),Point(2422.45,-1741.29,0,0,0,0),Point(2389.66,-1700.65,0,0,0,0),Point(2368.15,-1699.3,0,0,0,0),Point(2347.7,-1701.68,0,0,0,0),Point(2326.01,-1699.08,0,0,0,0),Point(2305.61,-1701.19,0,0,0,0),Point(2283.06,-1694.42,0,0,0,0),
    Point(2263.18,-1698.2,0,0,0,0),Point(2241.48,-1694.86,0,0,0,0),Point(2221.4,-1696.61,0,0,0,0),Point(2200.46,-1695.12,0,0,0,0),Point(2180.4,-1697.41,0,0,0,0),Point(2159.12,-1693.69,0,0,0,0),Point(2139.09,-1695.72,0,0,0,0),Point(2119.25,-1698.61,0,0,0,0),Point(2099.21,-1700.
    39,0,0,0,0),Point(2079.0,-1701.06,0,0,0,0),Point(2057.77,-1695.69,0,0,0,0),Point(2037.96,-1698.07,0,0,0,0),Point(2018.46,-1702.3,0,0,0,0),Point(1999.08,-1707.4,0,0,0,0),Point(1979.36,-1710.41,0,0,0,0),],'Chocolate',0),ObstacleData([Point(2069.64,-2482.47,0,0,0,0),Point
    (2045.16,-2485.89,0,0,0,0),Point(2020.65,-2489.17,0,0,0,0),Point(1996.13,-2492.29,0,0,0,0),Point(1972.29,-2501.22,0,0,0,0),Point(1946.69,-2495.1,0,0,0,0),Point(1919.51,-2472.91,0,0,0,0),],'Salmon',0),ObstacleData([Point(1613.24,360.61,0,0,0,0),],'Orange',0),ObstacleData
    ([Point(1741.75,-2809.22,0,0,0,0),Point(1714.77,-2800.64,0,0,0,0),Point(1688.29,-2800.88,0,0,0,0),Point(1661.9,-2802.96,0,0,0,0),Point(1635.45,-2803.87,0,0,0,0),Point(1609.01,-2804.62,0,0,0,0),Point(1582.53,-2804.21,0,0,0,0),Point(1556.08,-2804.63,0,0,0,0),Point(1529.59,
    -2798.9,0,0,0,0),Point(1503.17,-2799.0,0,0,0,0),Point(1476.75,-2798.94,0,0,0,0),Point(1450.34,-2798.71,0,0,0,0),Point(1423.92,-2798.33,0,0,0,0),Point(1397.48,-2798.78,0,0,0,0),Point(1371.07,-2798.07,0,0,0,0),Point(1344.55,-2800.19,0,0,0,0),Point(1318.0,-2802.15,0,0,0,0),
    Point(1291.57,-2800.95,0,0,0,0),Point(1265.15,-2799.59,0,0,0,0),Point(1238.43,-2803.86,0,0,0,0),Point(1211.92,-2802.37,0,0,0,0),Point(1185.49,-2800.51,0,0,0,0),Point(1158.91,-2800.49,0,0,0,0),Point(1132.49,-2798.32,0,0,0,0),Point(1106.27,-2797.99,0,0,0,0),Point(1079.89,
    -2791.45,0,0,0,0),Point(1053.32,-2790.81,0,0,0,0),Point(1027.18,-2786.0,0,0,0,0),Point(1000.85,-2783.01,0,0,0,0),Point(975.64,-2778.93,0,0,0,0),Point(950.55,-2758.7,0,0,0,0),],'DarkOrange',0),ObstacleData([Point(1389.14,680.46,0,0,0,0),Point(1391.53,733.64,0,0,0,0),Point
    (1390.92,762.02,0,0,0,0),Point(1384.24,778.22,0,0,0,0),Point(1378.61,798.42,0,0,0,0),Point(1363.94,814.37,0,0,0,0),Point(1348.88,825.73,0,0,0,0),Point(1335.5,842.26,0,0,0,0),Point(1319.12,845.54,0,0,0,0),Point(1303.96,838.43,0,0,0,0),Point(1291.41,826.56,0,0,0,0),Point
    (1275.9,821.03,0,0,0,0),],'Red',0),ObstacleData([Point(1248.71,768.98,0,0,0,0),Point(1230.33,730.91,0,0,0,0),Point(1215.85,735.07,0,0,0,0),Point(1211.87,752.62,0,0,0,0),Point(1204.17,772.88,0,0,0,0),Point(1187.18,782.2,0,0,0,0),Point(1165.84,781.95,0,0,0,0),Point(1149.15,
    781.04,0,0,0,0),Point(1134.5,773.79,0,0,0,0),Point(1118.69,765.02,0,0,0,0),],'DeepPink',0),ObstacleData([Point(984.3,535.17,0,0,0,0),],'MediumVioletRed',0),ObstacleData([Point(930.22,449.58,0,0,0,0),],'Magenta',0),ObstacleData([Point(739.1,117.57,0,0,0,0),],'BlueViolet',
    0),ObstacleData([Point(-838.86,-2688.19,0,0,0,0),Point(-866.53,-2677.22,0,0,0,0),],'DarkSlateGray',0),ObstacleData([Point(-966.89,-2792.65,0,0,0,0),],'Black',0),]
    for i in range(len(obstacles)):
        assert obstacles[i].Color==obstacles2[i].Color
        for j in range(len(obstacles[i].Points)):
            assert obstacles[i].Points[j].X==obstacles2[i].Points[j].X
            assert obstacles[i].Points[j].Y==obstacles2[i].Points[j].Y
```

Рис. 5 – Интеграционный тест №8, позитивный.

7 Покрытие тестами

Покрытие тестами проводилось в среде Micrasoft Visual Code с помощью фреймворка pytest. Для запуска проверки покрытие тестами использовался ключ «cov». Для покрытие тестами учитывались блочные и интеграционные тесты.



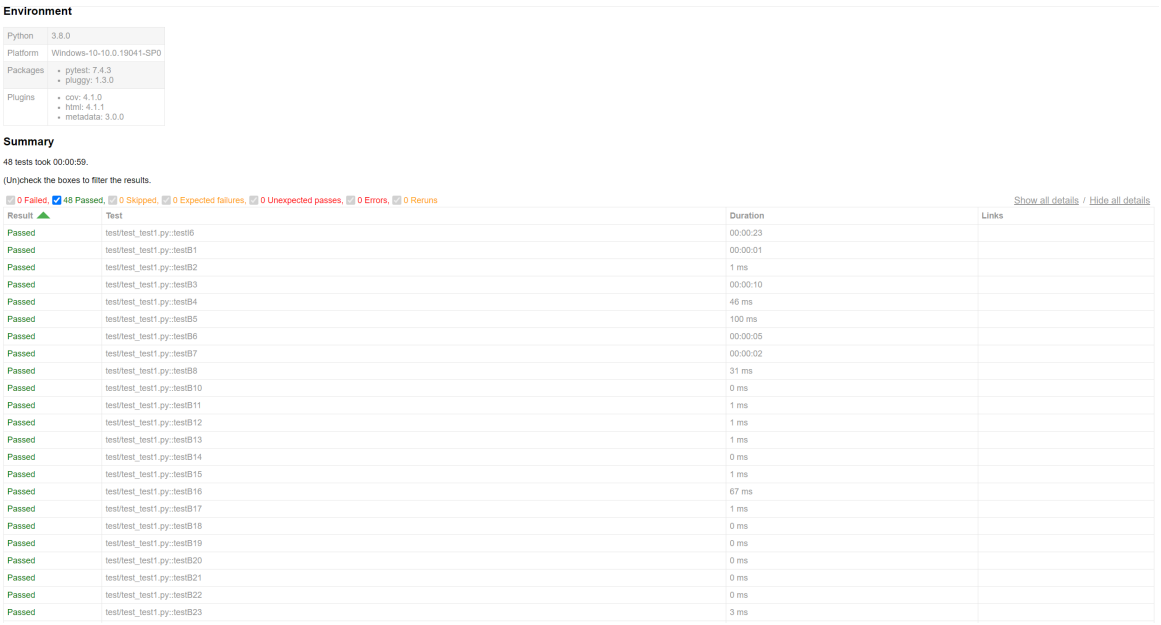
```
---------- coverage: platform win32, python 3.8.0-final-0 ----------
Name                    Stmts   Miss  Cover
-------------------------------------------
src\Client.py              28      2    93%
src\DatasetClass.py         7      0   100%
src\Map.py                318     37    88%
src\ObstacleData.py         6      0   100%
src\PointClass.py           6      0   100%
src\Server.py              21      0   100%
src\dataHandler.py         12      0   100%
src\lidarClass.py          36      3    92%
src\obstacle.py           150      3    98%
-------------------------------------------
TOTAL                     584     45    92%
```

Рис. 6 – Покрытие тестами

В итоге из 629 строк лишь 45 не было протестированно, в итоге покрытие тестами кода составило 92%.

8 Общее описание тестов

Кодом с помощью pytest было разработанно 48 тестов из них 40 блочных и 8 интеграционных тестов. В блочном тестирование было разработан 21 негативный тест и 19 позитивных, а также обнаружено 18 ошибок. В интеграционном 4 негативных и 4 позитивных тестах, а также обнаружена одна ошибка. Отчет об финальном прохождении тестов **с уже исправленными обнаруженными ошибками** можно посмотреть на рис 7 и 8



Environment

Python	3.8.0
Platform	Windows-10-10.0.19041-SP0
Packages	• pytest: 7.4.3 • pluggy: 1.3.0
Plugins	• cov: 4.1.0 • html: 4.1.1 • metadata: 3.0.0

Summary

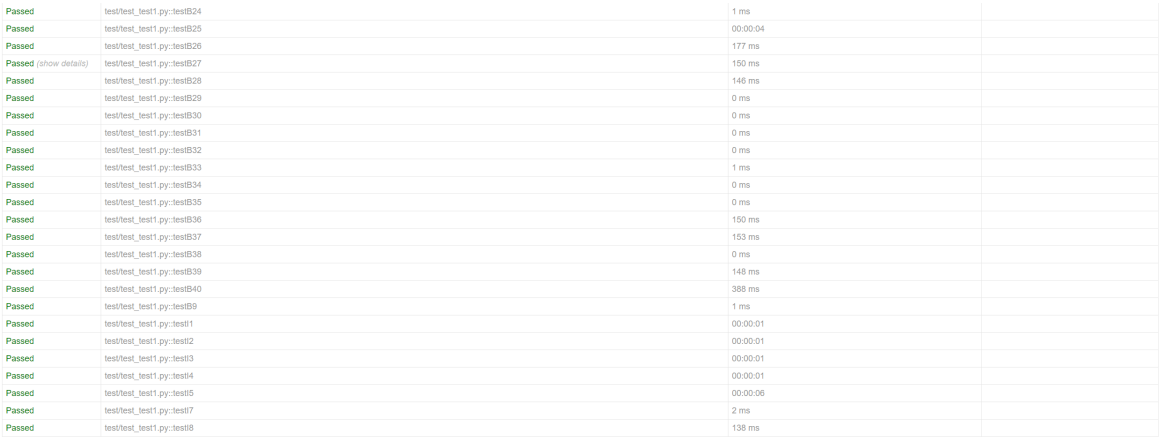
48 tests took 00:00:59.

(Un)check the boxes to filter the results.

0 Failed, 48 Passed, 0 Skipped, 0 Expected failures, 0 Unexpected passes, 0 Errors, 0 Reruns — Show all details / Hide all details

Result	Test	Duration	Links
Passed	test/test_test1.py::testI6	00:00:23	
Passed	test/test_test1.py::testB1	00:00:01	
Passed	test/test_test1.py::testB2	1 ms	
Passed	test/test_test1.py::testB3	00:00:10	
Passed	test/test_test1.py::testB4	46 ms	
Passed	test/test_test1.py::testB5	100 ms	
Passed	test/test_test1.py::testB6	00:00:05	
Passed	test/test_test1.py::testB7	00:00:02	
Passed	test/test_test1.py::testB8	31 ms	
Passed	test/test_test1.py::testB10	0 ms	
Passed	test/test_test1.py::testB11	1 ms	
Passed	test/test_test1.py::testB12	1 ms	
Passed	test/test_test1.py::testB13	1 ms	
Passed	test/test_test1.py::testB14	0 ms	
Passed	test/test_test1.py::testB15	1 ms	
Passed	test/test_test1.py::testB16	67 ms	
Passed	test/test_test1.py::testB17	1 ms	
Passed	test/test_test1.py::testB18	0 ms	
Passed	test/test_test1.py::testB19	0 ms	
Passed	test/test_test1.py::testB20	0 ms	
Passed	test/test_test1.py::testB21	0 ms	
Passed	test/test_test1.py::testB22	0 ms	
Passed	test/test_test1.py::testB23	3 ms	

Рис. 7 – Отчет о прохождении тестов



Passed	test/test_test1.py::testB24	1 ms	
Passed	test/test_test1.py::testB25	00:00:04	
Passed	test/test_test1.py::testB26	177 ms	
Passed *(show details)*	test/test_test1.py::testB27	150 ms	
Passed	test/test_test1.py::testB28	146 ms	
Passed	test/test_test1.py::testB29	0 ms	
Passed	test/test_test1.py::testB30	0 ms	
Passed	test/test_test1.py::testB31	0 ms	
Passed	test/test_test1.py::testB32	0 ms	
Passed	test/test_test1.py::testB33	1 ms	
Passed	test/test_test1.py::testB34	0 ms	
Passed	test/test_test1.py::testB35	0 ms	
Passed	test/test_test1.py::testB36	150 ms	
Passed	test/test_test1.py::testB37	153 ms	
Passed	test/test_test1.py::testB38	0 ms	
Passed	test/test_test1.py::testB39	148 ms	
Passed	test/test_test1.py::testB40	388 ms	
Passed	test/test_test1.py::testB9	1 ms	
Passed	test/test_test1.py::testI1	00:00:01	
Passed	test/test_test1.py::testI2	00:00:01	
Passed	test/test_test1.py::testI3	00:00:01	
Passed	test/test_test1.py::testI4	00:00:01	
Passed	test/test_test1.py::testI5	00:00:06	
Passed	test/test_test1.py::testI7	2 ms	
Passed	test/test_test1.py::testI8	138 ms	

Рис. 8 – Отчет о прохождении тестов

Также было проведено аттестационное тестирование в ходе тестирования было разработанно 18 аттестационных тестов. С помощью тестов были проверены все требования системы и все тесты завершились успешно. Таким образом, все требования были соблюдены.

Последним этапом тестировани это было проведение нагрзочного тестировани, на основе которого была выявлена ошибка, а именно экспоненциальное замедление работы программы, если построенная карта состоит из 4000 или более точек.

ID	Описание	Результат
N1	Проверка времени работы программы, если система за все время работы построила карту из 900 точек. Построение карты с таким количествой точек не должно превысить 1 секунды	Пройден
N2	Проверка времени работы программы , если система за все время работы построила карту из 2000 точек. Построение карты с таким количествой точек не должно превысить 1.5 секунды	Пройден
N3	Проверка времени работы программы, если система за все время работы построила карту из 4000 точек. Построение карты с таким количествой точек не должно превысить 2 секунды	Не пройден Ошибка E19

9 Результаты

За счет тестирование программы было выявленно ряд ошибок, которые были исправленные и программа стала более роботоспособной. Одной из основных обнаруженных ошибок это экспоненциальное замедление работы программы, если построенная карта состоит из 4000 или более точек. Данную ошибку предстоит решить для корректности и возможности работы данной системы.