Санкт-Петербургский государственный университет

СМИРНОВ Евгений Вячелавович

Выпускная квалификационная работа

Исследование алгоритмов автоматической регистрации данных лазерного сканирования

Уровень образования: магистратура

Направление: 05.04.03 «Картография и геоинформатика»

Основная образовательная программа: ВМ.5523 «Геоинформационное картографирование»

Научный руководитель:
Доцент кафедры картографии и геоинформатики,
канд. техн. наук Войнаровский А. Е.

Рецензент:
Корж Р.С.
Ведущий специалист,
ООО «Архитектурная фотограмметрия»

Санкт-Петербург
2022

Оглавление

Введение

В последние десятилетия все больше в различных сферах производственной деятельности используется технология наземного лазерного сканирования (НЛС). Достоинствами данной технологии, которые высоко ценятся при архитектурных, реставрационных, строительных и иных работах, стали высокая скорость сканирования, существенная степень автоматизации процесса, а также высокоточное и полное представление трехмерных данных. Такой вид представления пространственной информации существенно упрощает процесс создания и обновления планов, разрезов, что является неотъемлемой частью обмерной документации любых архитектурных объектов.

При этом, не смотря на в целом высокую степень автоматизации, важной особенностью большинства сканирующих систем является то, что материалы каждого сеанса сканирования получаются в собственной системе координат, что приводит к необходимости сведения всех результатом в единую систему отсчета. Классическим вариантом регистрации данных лазерного сканирования является сведение по общим точкам, в качестве таких обычно выступают марки заранее закрепленные на объекте, поиск и указание таких точек на сканах производится вручную специалистом и занимает много времени.

Поиск соответствующих точек может быть осуществлен на развертках скана и на ортофотопланах, которые являются цифровыми изображениями, исходя из чего, при подборе и поиске таких точек можно применить методы компьютерного зрения.

Актуальность исследования: использование методов компьютерного зрения позволяет значительно ускорить и упростить процесс регистрации данных лазерного сканирования.

Цель работы: изучение алгоритмов автоматизации регистрации данных лазерного сканирования, и их точности.

Для достижения поставленной цели необходимо было решить следующие задачи:

- Рассмотрение методов регистрации данных лазерного сканирования;
- Изучение методов поиска общих точек на ортофотопланах с использованием алгоритмов компьютерного зрения;
- Исследование методов отбора общих точек на ортофотопланах;
- Разработка алгоритма нахождения общих точек на ортофотопланах;
- Проверка работы алгоритма на реальных объектах;

– Анализ полученных результатов и формулирование выводов на основе исследования.

Объект исследования: регистрация данных лазерного сканирования;

Предмет исследования: методы автоматической регистрации данных лазерного сканирования.

Глава 1. Регистрация данных наземного лазерного сканирования. Теория.

1.1.Наземные лазерные сканеры. Принципы работы.

Многие современные практические и научные задачи могут быть решены посредством представления данных в трехмерном пространстве. Например, современная методика архитектурных обмеров не обходится без подобных данных. Также подобное представление находит применение в строительных работах, горнодобывающей промышленности и иной деятельности.

В настоящее время для решения задач, которые требуют представления данных в трехмерном пространстве помимо классических методов и инструментов, используется наземное лазерное сканирование. [1]

Система для наземного лазерного сканирования состоит из наземного лазерного сканера (НЛС) и полевого персонального компьютера со специализированным программным обеспечением. НЛС состоит из лазерного дальномера, адаптированного для работы с высокой частотой, и блока развертки лазерного луча. [2]

Схема сканирующей системы представлена на рисунке 1.

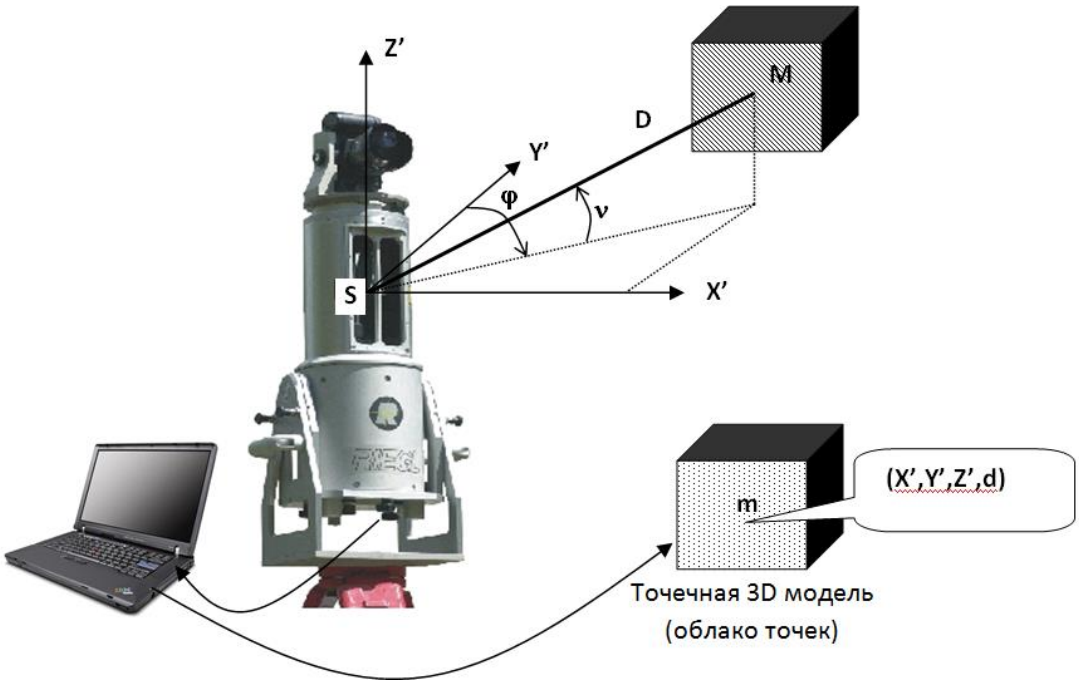

Рисунок 1 Схема работы наземного лазерного сканера [23]

Координаты каждой точки области сканирования представляют собой результат обработки трех измерений, совершаемых наземным лазерным сканером: горизонтальный (**φ**) и вертикальный (**θ**) углы и расстояние (**D**) от сканера до точки. Современные НЛС позволяют пользователю широкие возможности для настройки сеанса сканирования: задаются интервалы измерения вертикальных и горизонтальных углов, а также шаг, с которым будут измеряться углы. Основываясь на данных параметрах, сканер отклоняет лазерный луч на указанные углы и регистрирует значения **φ, θ, D** для каждой точки с определенной точностью.

Координаты точек вычисляются по формулам:

$$\left.\begin{aligned} X &= D\cos(\varphi)\sin(\theta) \\ Y &= D\sin(\varphi)\sin(\theta) \\ Z &= D\cos(\theta) \end{aligned}\right\} \quad (1)$$

В связи с конструктивными особенностями результат каждого сеанса сканирования представлен в пространственной системе координат с началом в центре сканера.

Наземные лазерные сканеры принято разделять в зависимости от метода определения расстояний на импульсные, фазовые и триангуляционные. Каждый вид наземных лазерных сканеров имеет свои преимущества и недостатки, в зависимости от которых, их применение может быть успешно при решении различных практических и научных задач.

Импульсные: в основе принципа действия импульсных сканеров лежит определение времени, за которое луч лазера проходит расстояние от сканера и обратно. Зная скорость распространения электромагнитных волн в воздухе, можно получить искомые расстояния. [18]

В данном случае расстояние определяется по формуле:

$$D = \frac{vt}{2} \quad (2)$$

v – скорость распространения электромагнитных волн в воздухе;

t – время, за которое луч проходит расстояние от сканера и обратно.

На рисунке 2 представлена схема работы импульсного дальномера наземного лазерного сканера.

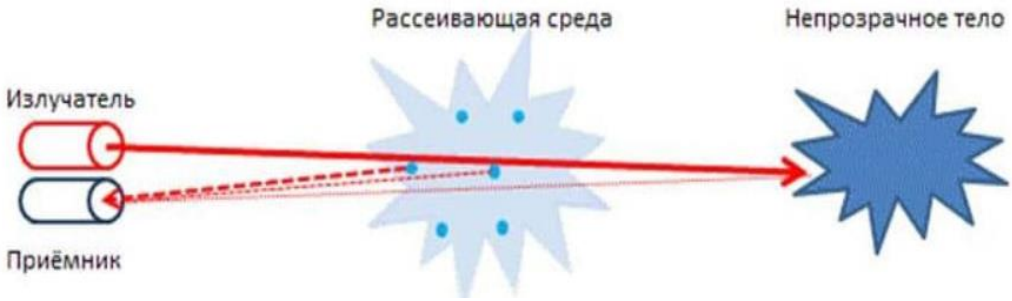

Рисунок 2 Импульсный дальномер [24]

Преимуществами данного типа определения дальности является больший диапазон расстояний, на которых возможно проведение работ. Соответственно такой тип сканеров находит большее применение на открытых обширных пространствах.

Фазовые: в основе фазовых лазерных сканеров лежит определение разности фаз уходящего и принимаемого сигнала. [3]

В таких сканерах расстояния рассчитываются по формуле:

$$D = \frac{\Delta\varphi v}{4\pi f} \quad (3)$$

v – скорость распространения электромагнитных волн

$\Delta\varphi$ – разность фаз уходящего и принимаемого сигнала

f – частота.

На рисунке 3 представлена схема работы фазового дальномера.

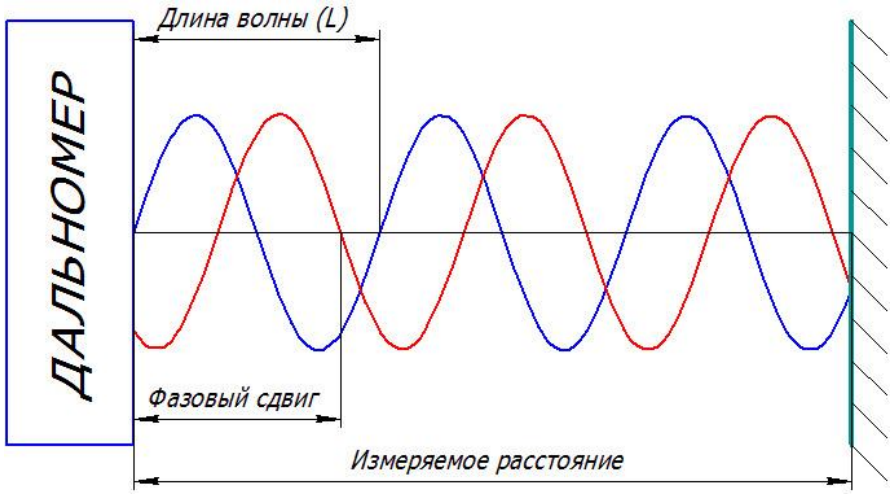

Рисунок 3 Фазовый дальномер [25]

Преимуществами данного типа дальномера являются более высокая скорость и точность работы на малых расстояниях, при этом применение его на обширных пространствах затруднительно.

Триангуляционные сканеры: в триангуляционных сканерах излучающий и принимающий элементы находятся на известном расстоянии друг от друга, называемом базисом. В данном случае известны: расстояние между датчиком и излучателем и угол между лазерным лучом и базисом, также возможно определить под каким углом луч возвращается на датчик. Следовательно, зная три элемента возможно решить, получаемый треугольник и определить координаты точки.

Триангуляционные лазерные сканеры позволяют выполнять измерения с высочайшей точностью до десятых и даже сотых долей миллиметра, но на очень небольшой дистанции. [4]

Поскольку дальность действия производимых на сегодня триангуляционных лазерных сканеров составляет от десятков сантиметров до 25 м, ошибки в измеряемых углах, вызванные влиянием атмосферы (рефракцией и затуханием электромагнитных колебаний), практически не оказывают влияния на результаты измерений. [5]

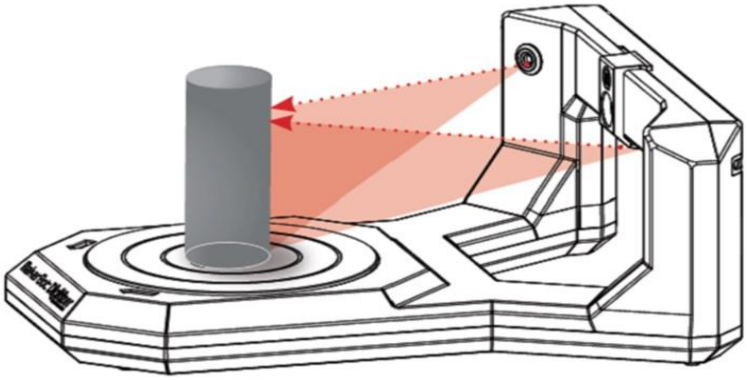

Рисунок 4 Триангуляционный сканер [26]

1.2. Алгоритмы регистрации данных лазерного сканирования.

Во время работ по сканированию объекта производится работа с нескольких станций, а так как при каждом сеансе работы сканы получаются в новой системе координат, возникает необходимость в приведении данных в единую систему координат проекта. Данный процесс называется регистрацией.

В настоящее время существует несколько методов пересчета координат сканов:

- Регистрация по маркам;
- Регистрация по геометрическим элементам;
- Регистрация по результатам инструментальных измерений;
- Регистрация методом ICP;

1.2.1. Регистрация по опорным точкам

Регистрация по опорным точкам является наиболее распространенным алгоритмом. Для ориентирования сканов относительно друг друга применяются специальные маркированные точки, положение которых в пространстве определено с высокой точностью, – марки. Для этого используется различное геодезическое оборудование (тахеометр, GPS-приемники и др.). При таком способе регистрации сканов значительно возрастает время, затрачиваемое на проведение полевых работ, однако данный способ является наиболее точным и надежным, благодаря чему и находит столь широкое применение. [2]

При проведении работ вокруг первой сканирующей станции устанавливаются шесть или более специальных марок, у четырех из которых (четвертая для контроля) геодезическим методом определяются пространственные координаты во внешней системе координат. Эти марки будут являться опорными. Со второй станции в поле зрения сканера должно попадать более трех марок, отобразившихся на первом скане, и не менее трех марок, которые будут видны с третьей станции, и т. д. На последней станции сканирования необходимо иметь минимум две опорные марки для уравнивания и оценки точности проложения сканерного хода. [2]

В результате создания проложения сканерного хода получаются три угловых и три линейных элемента для каждого скана, которые характеризуют его положение в пространстве. После регистрации данных наземного лазерного сканирования получается единое облако точек в необходимой системе координат. [6,7,8]

1.2.2. Регистрация по геометрическим элементам

При регистрации результатов сканирования в единую систему координат по геометрическим элементам, в отличие от предыдущего рассмотренного метода, в качестве опорных точек выступают точки пересечения плоскостей. Чаще всего сканирование проводится на архитектурных объектах, которые могут быть представлены совокупностью плоскостей. Для нахождения одной опорной точки необходимо

определить три взаимно непараллельные плоскости, при этом необязательно чтобы эти плоскости пересекались в физическом пространстве.

Для определения точек пересечения используются уравнения плоскостей следующего вида:

$$A \cdot x + B \cdot y + B \cdot z - 1 = 0 \quad (4)$$ уравнение плоскости в общем виде.

Так как для определения пространственных координат точки необходимы три пересекающиеся плоскости, которые дают систему, состоящую из трех уравнений, для нахождения координат точки необходимо решить данную систему.

$$\begin{cases} A_1 \cdot x + B_1 \cdot y + C_1 \cdot z - 1 = 0 \\ A_2 \cdot x + B_2 \cdot y + C_2 \cdot z - 1 = 0 \\ A_3 \cdot x + B_3 \cdot y + C_3 \cdot z - 1 = 0 \end{cases} \quad (5)$$ система уравнений плоскостей в общем виде.

где A1, B1, C1, A2, B2, C2, A3, B3, C3 – коэффициенты уравнений трёх различных плоскостей.

Такой метод может быть применен только при наличии в сканируемой области многих непараллельных плоскостей.

1.2.3 Регистрация по результатам инструментальных измерений

При регистрации по результатам инструментальных измерений используются встроенные в сканеры дополнительные устройства, которые позволяют сориентировать скан в необходимой системе координат.

Использование спутниковых геодезических технологий для определения линейных элементов внешнего ориентирования сканов в момент съемки сокращает время выполнения работ по сканированию объекта, а также позволяет отказаться от процесса создания рабочего съемочного обоснования. [2]

Существующие в настоящее время глобальные навигационные спутниковые системы определяют псевдодальности между спутником и принимающей аппаратурой, расположенной на земной поверхности. Для этого измеряется время прохождения сигнала от спутника к приемнику через атмосферу и космическое пространство, а так как скорость распространения электромагнитных волн и помехи накладываемые окружающей средой и

техническим устройством аппаратуры считаются известными, становится возможным определить расстояние между объектами.

Для определения пространственных координат в идеальных условиях достаточно трех спутников, результатом такого измерения будут координаты двух точек, из которых одна будет заведомо ложная, так как будет находится в космосе. Однако трех спутников достаточно если часы спутника и приемника абсолютно синхронизированы, но так как в реальности этого добиться невозможно, используется четвертый спутник, а поправка часов выступает в качестве еще одного неизвестного.

1.2.4. Алгоритм ICP

Наиболее популярным методом поиска тождественных точек является итеративный алгоритм ближайшей точки (ICP), разработанный Беслом и Маккеем . Было предложено несколько усовершенствований алгоритма ICP, таких как итеративная ближайшая совместимая точка (ICCP) и итеративная ближайая точка с использованием инвариантных характеристик (ICPIF). Алгоритм ICP требует хорошего первого приближения, чтобы получать корректные результаты. Алгоритм ICP также может сильно нагружать компьютер количеством вычислений при поиске сопряженных точек в перекрывающихся сканах. [9,10,11,12]

ICP (Iterative Closest Point) - это алгоритм компьютерного зрения, которые используется при регистрации данных лазерного сканирования для последовательного уменьшения до минимума расстояний между двумя облаками точек.

В ходе работы данный алгоритм приводит сканы друг к другу путем выполнения многократных аффинных преобразований (сдвиг и поворот) до момента, когда среднеквадратическая погрешность станет минимальной. [18]

При наличии двух облаков точек ai и bi и нормалей к ним ni, необходимо определить наиболее подходящие линейные и угловые элементы, с использованием которых облако точек ai будет приведено к облаку bi. Среднеквадратическая ошибка уравнивания сканов при работе алгоритма вычисляет по формуле 6.

$$\varepsilon = \sum_i [(Rai + t - bi) * ni]^2 \quad (6)$$

R – матрица поворота, t – матрица сдвига.

Матрица R представляет собой ортогональную матрицу 3x3 с определителем равным единице.

Матрица вращения для оси x:

$$Rx = \begin{matrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{matrix} \quad (7)$$

Матрица вращения для оси y:

$$Ry = \begin{matrix} \cos\alpha & 0 & \sin\alpha \\ 0 & 1 & 0 \\ -\sin\alpha & 0 & \cos\alpha \end{matrix} \quad (8)$$

Матрица вращения для оси z:

$$Rz = \begin{matrix} \cos\alpha & -\sin\alpha & 0 \\ \sin\alpha & \cos\alpha & 0 \\ 0 & 0 & 1 \end{matrix} \quad (9)$$

Матрица t определяет вектор смещения точек по осям x,y,z:

$t = [x,y,z]^T$

Для минимизации значения уравнения (1) необходимо вычислить функцию среднеквадратичной стоимости:

$$F(R,t) = \sum_{i=1}^{m}\sum_{j=1}^{N_p} \mu_{i.j}\,[(Rai + t - bi) * ni]^2 + \sum_{k=1}^{n}\sum_{l=1}^{N_q} \omega_{k.l}\,[(Rai + t - bi) * ni]^2 \quad (10)$$

где N_p и N_q – количество точек в множествах точек pi и qi, $\mu_{j,i}, \omega_{k.l}$- веса пар точек.

Значения весов определяются следующим образом:

$\mu_{j,i}, \omega_{k.l} = 1$, если pi является ближайшей точкой к qi, иначе $\mu_{j,i}, \omega_{k.l} = 0$ для всех i, j, k, l.

Наше выражение может быть представлено в другом виде:

$$F(R,t) = \frac{1}{\sum_{i=1}^{m}\sum_{j=1}^{N_p} \mu_{j,i}} \sum_{i=1}^{m}\sum_{j=1}^{N_q} \mu_{j,i}\,[(Rpi + t - qi) * ni]^2 \quad (11)$$

Этот вариант лишь немного влияет на точность оценки конечного результата. Это также немного замедляет сходимость, в смысле количества итераций, но ускоряет весь процесс. [13]

1.3. Алгоритмы поиска опорных точек для регистрации

Одним из вариантов регистрации данных лазерного сканирования является сведение сканов по опорным точкам, полученным в результате реализации поиска общих областей на изображениях, синтезированных из облаков точек, методами компьютерного зрения.

Наиболее подходящими для задачи обработки данных НЛС представляются методы SURF (Speeded Up Robust Features) и SIFT (Scale Invariant Feature Transform). Оба алгоритма реализуют нахождение инвариантных к масштабу и повороту точек изображений. Их основных различием являются методы нахождения особых точек, в алгоритме SURF используется матрица Гессе, а SIFT находит особы точки путем построения пирамиды разности гауссианов. [13,14,15]

В данной работе рассматривается возможность применения метода SIFT для цели регистрации данных наземного лазерного сканирования.

SIFT производит преобразование изначального изображения в набор локальных векторов объектов, которые являются инвариантными к перемещению, масштабированию, повороту. [14]

Последовательность действий алгоритма можно разделить на несколько этапов:

1. Нахождение локальных экстремумов точек на изображениях;
2. Определение градиентов точек и создание на их основе дескриптора;
3. Сопоставление дескрипторов на разных изображениях.

Первым этапом нахождения ключевых точек изображения является обнаружение локальных экстремумов, инвариантных к изменению масштаба изображения, этого можно достигнуть путем поиска стабильных объектов на различных масштабах примыкающих к изначальному. [16]

С целью нахождения стабильных элементов в различных масштабах строится пирамида разностей гауссианов. Гауссианом называется изображение, которое было размыто фильтром Гуса, его аналитический вид представлен в формуле 12.

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y) \quad (12)$$

L – значение гауссиана в точке с координатами(x,y)

σ - радиус размытия

G - гауссово ядро

I - значение исходного изображения

* – операция свёртки изображения.

Свертка изображения производится путем вычисления нового значения пикселя на основе его окружения.

$$G = \frac{1}{2\pi\sigma^2} \times exp^{\frac{-(x^2+y^2)}{\sigma^2}} \quad (13)$$

Рисунок 5 Уменьшение изображения для формирования октав [22]

При построении пирамиды разностей гауссианов исходное изображение в первую очередь уменьшается в два раза и с заданным размывают фильтром Гаусса. Данная процедура повторяется несколько раз до необходимого передела.

Так как для цифровых изображений под изменением масштаба можно понимать различие в пространственном охвате пикселя, таким образом моделируются различные варианты масштаба и достигается его инвариантость.

После получения разномасштабных изображений вычисляются разности гауссианов. Ими являются изображения, полученные путем попиксельного вычитания одного гауссиана с радиусом σ из соседнего гауссиана с радиусом размытия kσ. [16]

Функция разности гауссианов:

$$D(x, y, \sigma) = \big(G(x, y, k\sigma) - G(x, y, \sigma)\big) * I(x, y) \quad (14)$$

Составленная из разностей гауссианов пирамида является набором изображений, на которых возможно проводить определение локальных экстремумов яркостей точек. Это происходит путем сравнения пикселя с восемью окружающими его точками в данном масштабе и девятью точками для каждого смежного масштаба, если значение пикселя больше или меньше всех окружающих его, то он принимается за локальный экстремум. В ином случае окно проверки сдвигается на одно деление и происходит такой же процесс до тех пор пока не будет найден экстремум или алгоритм не дойдет до края изображения. [16]

На следующем этапе уточняется положение точки. Оно производится через разложение в ряд Тейлора функции разности гауссианов (14)

$$D(Z) = D + \frac{\partial D^T}{\partial Z} Z + \frac{1}{2} Z^T \frac{\partial^2 D}{\partial Z^2} Z \quad (15)$$

Z – вектор смещения относительно точки разложения функции

D – функция разности гауссианов.

Затем необходимо отсеять точки с небольшой контрастностью и точки, лежащие на границе объекта. В методе SIFT для проверки на контрастность используется следующий метод:

$$D(Z) = D + \frac{1}{2} \frac{\partial D^T}{\partial Z} Z \quad (16)$$

В случае недостаточной величины контраста точка исключается.

Точки, лежащие на границе объекта, можно исключить, использовав матрицу Гессе. Получается она путем взятия второй производной функции разности гауссинанов по двум переменным x и y. Матрица имеет размерность 2x2 и принимает следующий вид:

$$H = \begin{pmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{pmatrix} \quad (17)$$

Затем для данной матрицы необходимо вычислить определитель и след. После чего рассчитывается отношение квадрата следа к определителю:

$$\frac{Tr(H)^2}{\det(H)} = \frac{(D_{xx}+D_{yy})^2}{D_{xx}D_{yy}-D_{xy}{}^2} \quad (18)$$

Точки, определитель которых отрицательный, отбрасываются, а для остальных происходит сравнение предыдущего отношения с величиной:

$$\frac{(r+1)^2}{r} \quad (19)$$

$r = \frac{a}{b}$, a – большой изгиб; b – меньший изгиб.

В случае, когда значение выражения 18 меньше значения выражения 19 точка принимается ключевой.

Следующий этап работы алгоритма определение ориентации опорной точки. Ориентация определяется по направлениям точек, соседних с особой. Вычисления направления ключевой очки происходит на изображении в пирамиде разности гауссианов, с масштабом наиболее близким к масштабу ключевой точки. [16]

Величина и направление градиента вычисляется по следующим формулам:

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x,y+1) - L(x,y-1))^2} \quad (20)$$

$$\theta(x,y) = \arctan\left(\frac{L(x,y+1)-L(x,y-1)}{L(x+1,y)-L(x-1,y)}\right) \quad (21)$$

m(x,y) - величина градиента точки

θ(x,y) - направление.

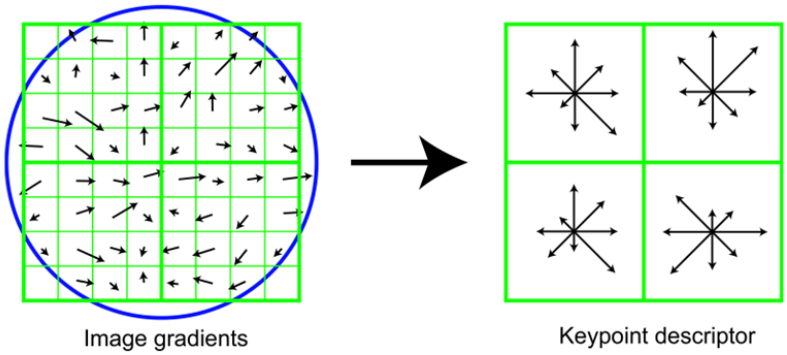

Рисунок 6 Дескриптор особой точки [16]

Градиенты рассматриваются в окрестности точки равной 3σ, так как на этом расстоянии значение гауссова ядра близко к единице.

После определения особых точек необходимо совместить одинаковые точки на разных изображениях. Для этого служит дескриптор. В данном методе дескриптором является вектор, вычисление которого происходит на гауссиане с ближайшим к ключевой точке масштабом. При расчёте используются значения градиентов в некоторой окрестности искомой точки. На данном этапе также достигается инвариантность относительно поворота, путем поворота окна ключевой точки на угол данной точки. [16]

Глава 2. Опыт предыдущих исследований

2.1. Применение метода SIFT

При проведении исследований по регистрации данных лазерного сканирования на втором и четвертом курсах проверялись также два подхода к использованию метода SIFT.

При написании выпускной квалификационной работы на степень бакалавра исследовался метод SIFT без внесения каких-либо изменений в алгоритм. Данный подход реализован и используется в программе SIFT.

При исследованиях, проводимых для написания курсовой работы на втором курсе, метод SIFT применялся на начальном этапе работы в урезанном виде. В данном случае производится построение пирамиды разностей гауссианов и определение локальных экстремумов, которые являются особыми точками изображения. На этом работа непосредственно с методом SIFT завершалась.

На следующем этапе для нахождения точек, которые можно считать одинаковыми на обоих снимках, использовалось то, что изначальные данные являются облаком точек и имеют трехмерные пространственные координаты, по которым возможно вычислить расстояние. Для фильтрации особых точек вычислялись расстояния от каждой N-ой особой точки до всех остальных N-1 точек по формуле нахождения расстояния в трехмерном пространстве по координатам.

В результате для каждого изображения получались квадратные матрицы с вычисленными расстояниями, которые затем вычитались одна из другой с целью вычисления погрешностей в расположении точек, которые принимаются одинаковыми. Значения ячеек полученной матрицы проверялись на принадлежность к интервалу от -0.5 до 0.5 сантиметров. Каждое значение ячейки, соответствующее данному требованию, записывалось в весовую матрицу со значением единицы, остальные записывались с нулевым весом. Затем точки с наибольшим весом использовались при расчете среднеквадратической погрешности (СКП). Для контроля правильности нахождения точек использовалась тройная величина допустимой разницы расстояний, в случае если СКП выходила за рамки допустимых значений из расчетов исключались точки с наименьшим весом.

2.2. Проверка полного алгоритма метода SIFT

При проверке работы в качестве тестовых объектов использовались различные типы архитектурных сооружений, как типовых фасадов домов, так и исторических зданий с различными особенностями постройки.

Данные лазерного сканирования, полученные на выбранных объектах переводились в растровые форматы данных, с которыми позволяет работать метод SIFT. В использованной в ходе работы программе ScanIMAGER есть возможность представления данных в виде разверток сканов и ортофотопланов на выбранную плоскость.

2.2.1. Развертки

Созданные в программе ScanIMAGER развертки импортировались в JPEG формат, с которым может работать исследуемая программа SIFT. При анализе изображений в программе метод SIFT находил наибольшее количество ключевых точек на участках разверток, которые имеют наименьшую разницу в ракурсах, с увеличением угла между двумя одинаковыми участками на развертках количество найденных точек резко падает. Также для найденных точек была характерна большая доля ошибочных результатов, которые было невозможно отфильтровать в актуальной версии программы. Данные результаты были характеры для всех типов сооружений, которые были выбраны тестовыми объектами в исследовании.

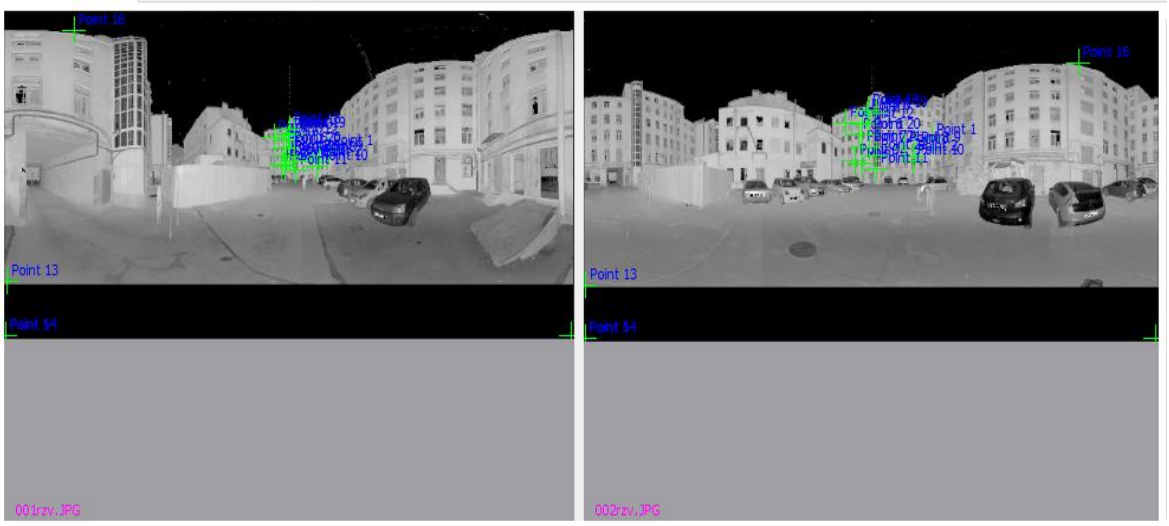

Рисунок 7 Результат работы метода на развертках

2.2.2. Ортофотопланы

При проверке метода SIFT на ортофотопланах создавались изображения с пространственным разрешением 25 миллиметров на пиксель, затем, полученные

ортофотопланы подвергались интерполяции с целью заполнения небольших пропусков, вызванных отсутствием точек в области, которая попадает на пиксель.

По результатам исследований работы метода SIFT для регистрации сканов фасадов зданий выявлено, что на ортофотопланах сканов с большими взаимными углами поворота горизонтальный масштаб изменяется на существенную величину, что ухудшает качество работы исследуемого метода. Это происходит в связи с тем, что при поиске особых точек изображения масштабируются, чем достигается инвариантность к масштабу, а когда масштаб изменяется только по одному направлению, масштабирование теряет большую часть эффективности. Для исключения влияния данного фактора на результаты работы, необходимо изменить горизонтальный масштаб изображения на близкий к ортофотоплану, с которым проводится сравнение.

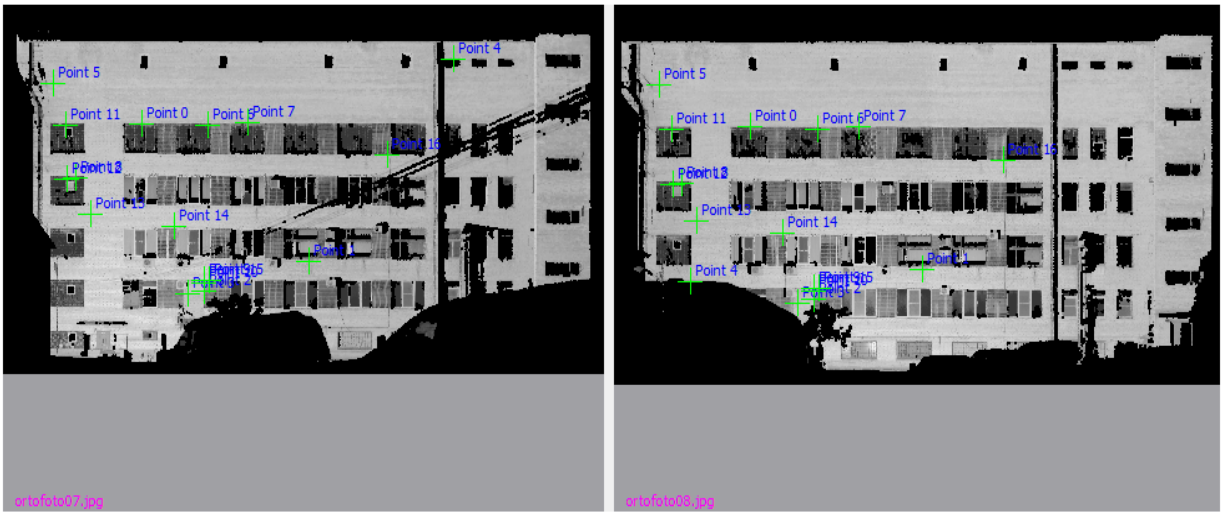

Рисунок 8 Результат работы метода на ортофотопланах

При работе со сканами помещений наиболее качественно точки были детектированы при сравнении стен, которые расположены на примерно равном удалении от сканеров. Это объясняется тем, что в таком случае получаемые изображения имеют схожие характеристики плотности точек и яркости пикселей, что облегчает работы алгоритма SIFT.

При поиске общих точек на потолке метод показал хороший результат в случае, если на потолке присутствуют различные элементы, которые сильно выделяются на общем фоне, такие как: балки, лепнина, рисунки, дефекты штукатурки и иные особенности.

При создании ортофотопланов на стены с большой разницей в расстоянии до сканера результаты были неудовлетворительными, это обусловлено сильно различающейся плотностью и яркостью точек на сканах.

Полученные результаты значительно превосходили, ранее полученные на развертках, также функционал программы ScanIMAGER позволял перевести пиксельные координаты цифровых изображений в пространственные координаты скана и рассчитать среднеквадратические погрешности, определенных ключевых точек. Результаты расчетов для ортофотопланов представлены в таблице 1. Здесь и далее в первой графе приводится адрес объекта, по сканам которого выполнялось исследование, во второй графе представлены названия сканов, в третьей графе количество найденных верных опорных точек, в четвертой графе максимальные среднеквадратические погрешности координат.

Таблица 1. Результаты расчета среднеквадратической погрешности для ортофотопланов

№	Адрес	Пара сканов	Количество общих точек	СКП, мм.
Фасады зданий				
1	Старо-Петергофский проспект, дом 44	FARO Scan: 007 –008	10	x = 7.4 y = 15.1 z = 9.7
2	Старо - Петергофский проспект, дом 44	FARO Scan: 007 – 009	17	x = 6.5 y = 9.8 z = 11.2
3	Старо - Петергофский проспект, дом 44	FARO Scan: 007 – 010	15	x = 7.1 y = 10.6 z = 8.4
4	8 линия Васильевского острова, дом 73	FARO Scan: 020 – 021	12	x = 6.4 y = 9.1 z = 11.9
5	Старо-Петергофский проспект, дом 54	FARO Scan: 001 – 002	14	x = 6.7 y = 7.4 z = 6.6
6	Старо-Петергофский проспект, дом 54	FARO Scan: 001 – 003	13	x = 5.7 y = 8.2 z = 9.1

7	Старо-Петергофский проспект, дом 54	FARO Scan: 001 – 004	13	x = 6.0 y = 11.5 z = 8.6
8	Старо-Петергофский проспект, дом 54	FARO Scan: 001 – 005	11	x = 7.4 y = 6.1 z = 8.3
Внутренние помещения				
9	Васильевский остров, Кадетская линия, дом 3	1et kadets: 065 – 066 потолок	4	x = 20.1 y = 19.2 z = 4.7
10	Васильевский остров, Кадетская линия, дом 3	1et kadets: 065 – 066 стена с одинаковым расстоянием до сканеров	4	x = 0.7 y = 9.2 z = 10.2
11	Васильевский остров, Кадетская линия, дом 3	1et kadets: 065 – 066 стена с разным расстоянием до сканеров	6	x = 8.3 y = 14.7 z = 17.6

Сопоставление точности регистрации разных пар сканов по ортофотопланам является некорректной поскольку они зависят от множества факторов: типа объектов, разрешения ортофотоплана, расстояния от сканера до объекта, структуры объекта, разрешающей способности сканера.

2.3. Проверка метода SIFT с фильтрацией по расстояниям

В ходе исследования были проведены работы по проверке алгоритма получения опорных точек сканов через фильтрацию по евклидовым расстояниям, полученных методом SIFT локальных экстремумов точек.

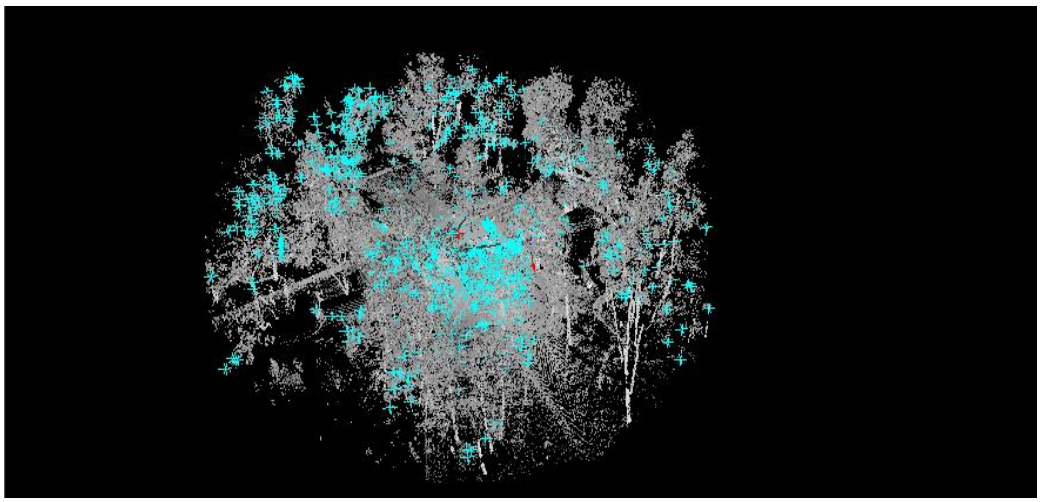

Рисунок 9 Скан с отобранными особыми точками

Результаты работы алгоритма проверялись через вычисление среднеквадратической погрешности координат точек, значение которой должно было быть в пределах 1,5 сантиметров. В случае если она оказывалась больше допустимого значения, отбрасываются точки с наименьшими весами. Оставшиеся после фильтрации точки пригодны для использования для регистрации.

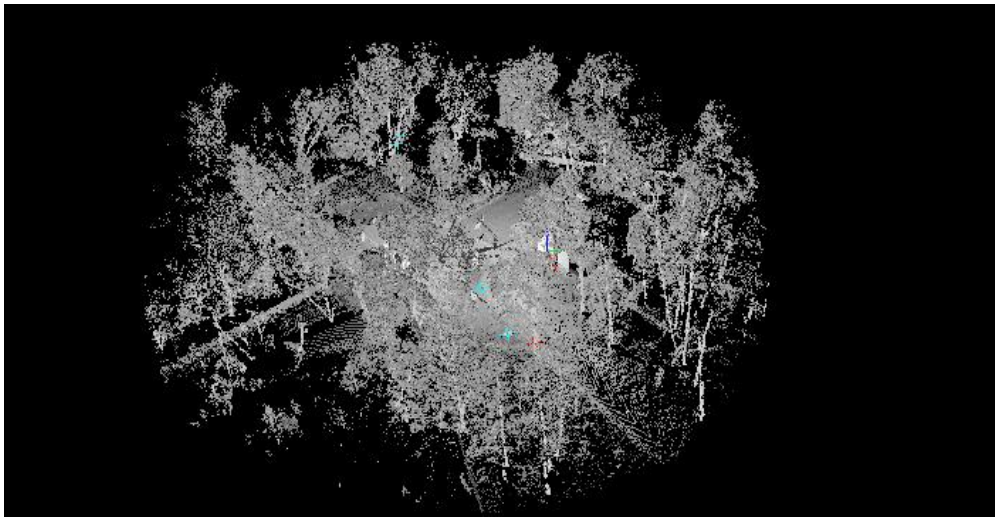

Рисунок 10 Скан с найденными общими точками

Таблица 2. Результаты проверки метода SIFT с фильтрацией по расстояниям

№	Адрес	Пара сканов	Количество общих точек	СКП, мм.
Внутренние помещения				
1	Васильевский остров, 8 линия	Fasad: 004 –005	53	x = 13.3 y = 12.5 z = 6.9
2	Васильевский остров, 8 линия	FARO Scan:	108	x = 2.4

		015 - 016		y = 2.6 z = 2.3
3	Выборгское шоссе	Scan: 011 - 012	52	x = 5.8 y = 4.6 z = 5.2
4	деревня Космозеро, дорожная улица, церковь Александра Свирского	K ozero: 012 - 014	8	x = 4 y = 3.9 z =8.1
Фасады зданий				
5	г. Петергоф, ул. Чайковского, Адмиральский домик	F002 – F015	7	x = 2.3 y = 10.2 z = 7.9
6	Васильевский остров, 8 линия	FARO Scan: 025 - 026	13	x = 2.7 y = 4.6 z = 3.1
7	деревня Космозеро, дорожная улица, церковь Александра Свирского	Kozero: 001 - 002	7	x = 4 y = 2.7 z = 9.8
9	деревня Космозеро, дорожная улица, церковь Александра Свирского	K ozero: 005 - 006	9	x = 7 y = 5.3 z = 6.7
9	деревня Космозеро, дорожная улица, церковь Александра Свирского	K ozero: 009 - 010	7	x = 6.2 y = 6.4 z = 4.5

Наибольшее влияние на получаемые результаты, как и в подразделе 2.1 данной главы имела разность ракурсов, получаемых при развертке на плоскость цилиндра, на который проецировались точки скана при создании развертки.

Глава 3. Исследование методов отбора общих точек на ортофотопланах.

3.1. Программные средства

Для проведения исследования с использованием материалов лазерного сканирования необходимо определиться с перечнем программных средств, в которых реализована возможность работы с данными материалами. Проведение данного исследования подразумевает получение ортофотопланов заданного пространственного разрешения из материалов сканирования и трехмерных координат с него, а также растровых изображений и координат цифрового изображения. Все реализованы в программе ScanIMAGER.

В данном программном обеспечении разработан большой выбор вариантов обработки сканов, их визуализации и моделирования. Программа состоит из нескольких модулей. Модуль «Управление данными» используется для включения и удаления сканов, используемых в проекте, а также сохранения результатов работы. В модуле «Сканы» возможно производить настройку отображения сканов, а также экспортировать в форматы, применяемые в других программных продуктах. Следующий модуль включенный программу «Регистрация». Для этого существует функция «Ввод марок», где можно обозначить точки на скане, которые являются марками, причём можно выбрать, метка какого именно типа (ZF-марка, фотограмметрическая марка, точка-пятно) будет обозначена на скане. После выбора точки она появляется в списке в нижней части рабочего окна со своими координатами. Также в программе предусмотрена возможность представления трехмерного облака точек, коим являются сканы, в виде ортофотопланов. Для работы с данным форматом присутствуют модуль «Ортофотопланы». [17]

Исследование, проводимое с использованием ортофотопланов, подразумевало применение различных алгоритмов компьютерного зрения, которые могут быть программно реализованы на языке Python. Язык Python был выбран по причине наибольшей распространенности в области работы с пространственными данными, следовательно, на нем должно быть достаточное количество библиотек, позволяющих производить обработку. Средой программирования применяемой в данном исследовании является Anaconda.

При реализации алгоритма использовались следующие библиотеки:

NumPy – это пакет для научных вычислений на языке Python. Это библиотека Python, которая предоставляет объект многомерного массива, различные производные

объекты (такие как маскированные массивы и матрицы) и набор подпрограмм для быстрых операций с массивами, включая математические, логические, манипуляции с фигурами, сортировку, выбор, ввод-вывод. , дискретные преобразования Фурье, основы линейной алгебры, основные статистические операции, случайное моделирование и многое другое. [19]

OpenCV (Open Source Computer Vision Library) - это библиотека программного обеспечения для компьютерного зрения и машинного обучения с открытым исходным кодом. OpenCV был создан, чтобы обеспечить общую инфраструктуру для приложений компьютерного зрения и ускорить использование машинного восприятия в коммерческих продуктах. Являясь свободным программным обеспечением библиотека позволяет легко использовать и изменять код.[20]

3.2. Описание функционала алгоритма на языке Python

Создаваемые в программе ScanIMAGER ортофотопланы из-за различной плотности точек в проецируемых на плоскость областях и проводимой после проецирования интерполяции, получаются со значительной разницей в яркостях и контрасте. Для выравнивания яркостей изображений используется функция CLAHE (Contrast Limited Adaptive Histogram Equalization).

Рисунок 11 Ортофотопланы до выравнивания гистограмм

Данная функция предназначена для адаптивного выравнивания гистограммы яркости изображения. В данном случае изображение разбивается на блоки, для каждого из которых затем растягивается гистограмма. Таким образом, в каждом фрагменте будет использована своя гистограмма. Также есть возможность ограничения контраста. Если какая-либо ячейка гистограммы превышает указанный предел контрастности, эти пиксели обрезаются и равномерно распределяются по другим ячейкам перед применением выравнивания гистограммы. После выравнивания для удаления артефактов на границах тайлов применяется билинейная интерполяция.[21]

Рисунок 12 Ортофотопланы после выравнивания гистограмм

3.2.1 Методы указания точек на ортофотопланах.

Наиболее простым вариантом указания точек на ортофотопланах, является ручной выбор, в программе написанной в среде Anaconda это происходит двойным нажатием на точку, которую необходимо указать. Данный метод указания точек позволяет выбрать те области, которые при визуальной оценке имеют наибольший контраст.

Рисунок 13 Результат ручного указания точек

Второй метод основан на алгоритме SIFT, что позволяет использовать математическую основу данного алгоритма, которая базируется на определении локальных экстремумов пирамиды разностей гауссианов.

Третьим вариантом набора точек, чья проверка проводилась в ходе исследования, было указание первых трех первых областей с их указанием на обоих ортофотопланах, с последующим вычислением среднего смещения точек общих точек на изображениях и и его использования при указании остальных областей только на первом ортофотоплане. Данный метод позвояет проводить поиск точек не на всем изображении а в небольшой области вокруг определенной по вычисленному смещению точки.

Рисунок 14 Результат указания точек методом SIFT

3.2.2 Метод определения общих точек на втором ортофотоплане.

Для определения точек, соответствующих выделенным на первом ортофотоплане, использовался метод расчета коэффициентов корреляции, в библиотеке OpenCV есть функция matchTemplate, которая позволяет сравнивать, созданный шаблон с другим изображением и через определение максимального коэффициента корреляции получать похожую область. Для нахождения соответствующей части на втором изображении шаблон последовательно перемещается по всему изображению. В каждом местоположении вычисляется величина совпадения изображения с шаблоном с использованием нормализованного коэффициента корреляции. [27]

$$R(x,y) = \frac{\sum_{x',y'} T'(x',y') * I'(x+x',y+y'))}{\sqrt{\sum_{x',y'} T'(x',y')^2 * \sum_{x',y'} I'(x+x',y+y')^2}} \quad (22)$$

На рисунке 15 представлен результат расчета коэффициентов корреляции для рисунка 13.

Рисунок 15 Результат расчета коэффициентов корреляции

Изначальные результаты поиска не удовлетворяют требованиям количества и точности определения точек, которые впоследствии должны использоваться для регистрации сканов между собой. Для того чтобы решить данные проблемы необходимо проводить фильтрацию получаемых результатов, а также реализовать поиск характерных точек с субпипксельной точностью.

В программе были реализованы следующие варианты фильтрации точек:

- Фильтрация по присутствующим черным пикселям;
- Фильтрация по местоположению;
- Фильтрация по расстояниям;

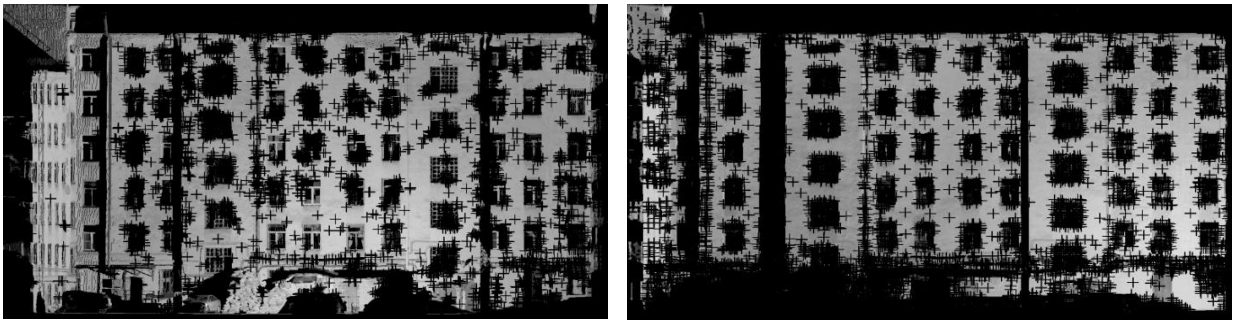

Рисунок 16 Результат расчета корреляции без фильтрации

Суть фильтрации по черным точкам заключается в том, что области на ортофотопланах, на которые не попадает ни одного сигнала, принятого сканером, являются черными, следовательно вносят сильное влияние при расчете коэффициента корреляции. Данный факт приводит к тому, что при расчете корреляции, в первую очередь при автоматическом наборе точек, большая часть результатов со значительным присутствием в них черных пикселей будет неверной и вносить большую ошибку в результат. Чтобы этого избежать в программу встроен фильтр, который удаляет точки, если процент содержащихся в них пикселей выше определенного порога, что позволяет отбросить большую часть заведемо

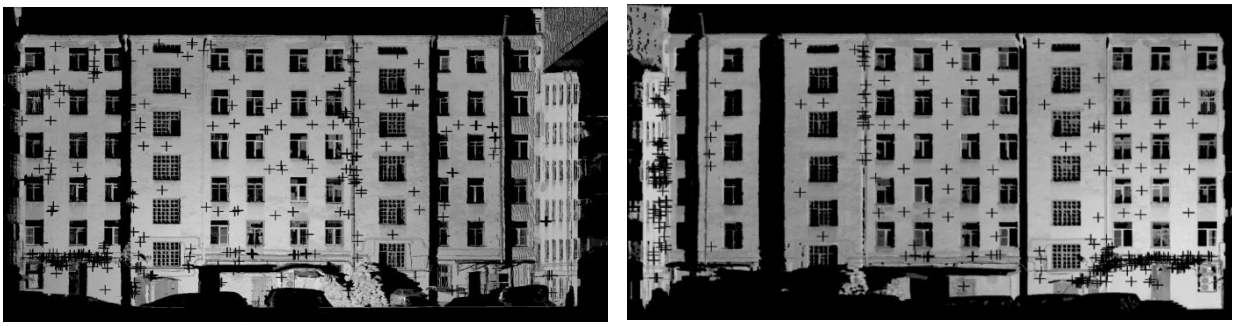

Рисунок 17 Результат расчета корреляции с фильтрацией черных пикселей

неверных результатов.

На рисунках 16 и 17 представлены результаты работы программы расчета коэффициентов корреляции, с набором точек через расчет локальных экстремумов

алгоритмом SIFT, без фильтрации черных пикселей и с фильтрацией. Улучшение результатов видно невооруженным взглядом, однако данных результатов недостаточно для использования точек при регистрации сканов.

Для дальнейшей фильтрации был разработан алгоритм разделения изображений на одинаковое количество блоков, при этом поиск тождественных точек осуществлялся по соответствующим блокам. Предположение о возможной работоспособности данного метода заключалось в том, что сканы, на основе которых создавались ортофотопланы, были предварительно ориентированы в пространстве и, не смотря на смещение по оси x и поворот вокруг вертикальной оси, на соответствующие части попадают одинаковые фрагменты.

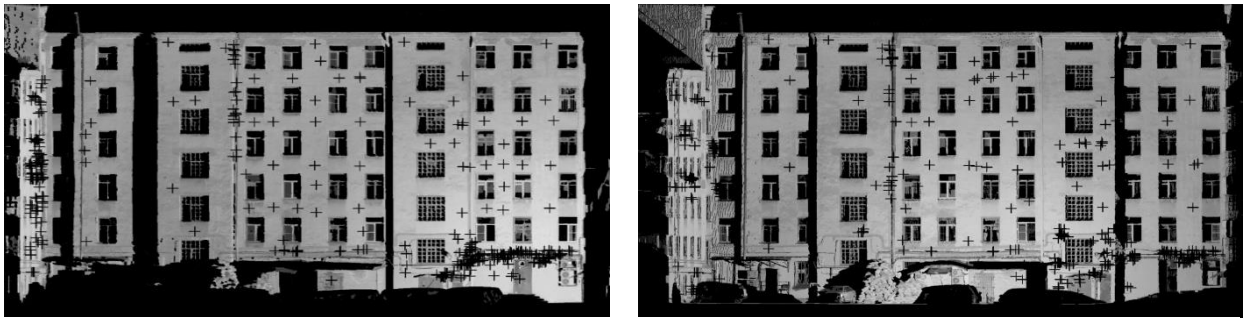

Рисунок 18 Результат работ метода с разбиением на блоки

Так как библиотека OpenCV позволяет работать с координатами изображения было принято решение провести фильтрацию точек с расчетом расстояний на цифровом изображении в пиксельном выражении. Суть идеи заключается о расчете взаимных расстояний всех точек на обоих ортофотопланах, с последующим вычитанием соответствующих с последующим вычитанием массива расстояний одного ортофотоплана из другого. Полученные при вычитании значения должны попадать в допуск, который устанавливается в зависимости от пространственного разрешения ортофотоплана.

На рисунке 19 видно, что по сравнению с рисунками 16. 17 и 18 результаты значительно улучшились.

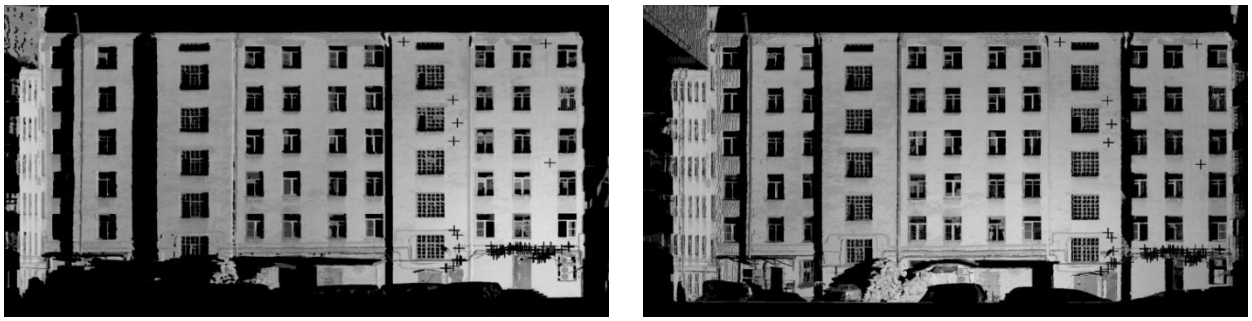

Рисунок 19 Результат фильтрации по расстояниям

Найденные на ортофотопланах точки можно сохранить в текстовые файлы, которые затем можно использовать для перевода из пиксельных координат изображения в пространственные координаты скана.

3.3 Проверка работы метода на объектах

Проверка работы метода набора и фильтрации опорных точек для регистрации данных лазерного сканирования проводилась для каждого из вариантов набора точек, представленных в предыдущем разделе главы. Также проводилась проверка влияния функции разбиения изображения на фрагменты представленной в подразделе 3.2.2., так как при предварительной оценке работы программы возникли сомнения в целесообразности использования данной функции.

3.3.1 Проверка субпиксельной точности

В программе была реализована возможность расчета координат с субпиксельной точностью. Вычисления координат проводились по формуле средневзвешенного, где весовыми элементами были коэффициенты корреляции для пикселей примыкающих к искомому. Проверка качества расчета координат проводилась путем сравнения результатов работы программы с измерениями выполненными тахеометром на марках, установленных перед сканированием.

В Таблице 3 представлены результаты проверки работы данной функции на ортофотопланах с пространственным разрешением 10 миллиметров на пиксель.

Таблица 3 результаты проверки субпиксельной точности

Номер марки	Разрешение скана	Координаты тахеометром	Координаты указанной точки	Координаты найденной точки
1	10 мм./пикс.	x = 2.883911 y = 69.581222 z = 1.35	x = 2.883428 y = 69.583831 z = 1.35251	x = 2.883750 y = 69.580629 z = 1.349628
2	10 мм./пикс.	x = 2.883911 y = 70.640642 z = 0.263	x = 2.881389 y = 70.638100 z = 0.267990	x = 2.882721 y = 70.633001 z = 0.267970
3	10 мм./пикс.	x = -13.566847 y = 71.829442 z = 0.448	x = -13.568361 y = 71.825725 z = 0.446323	x = -13.563376 y = 71.835720 z = 0.447365
4	10 мм./пикс.	x = -13.623853 y = 69.742886	x = -13.617830 y = 69.746138	x = -13.621674 y = 69.744262

		z = -0.015	z = -0.013642	z = -0.001534
5	10 мм./пикс.	x = -0.667643 y = 76.934208 z = 1.905	x = -0.665924 y = 76.931446 z = 1.906298	x = -0.663473 y = 76.934513 z = 1.907520

Основываясь на результатах таблицы 3 можно заключить, что функция позволяет определять координаты с точностью до 0.3-0.4 пикселя.

3.3.2 Проверка работы метода при ручном наборе точек

Для проверки метода использовались ортофотопланы, созданные на основе сканов различных типов фасадов и помещений, в работе использовались как типовые здания и интерьеры, так и нетиповые, характерные для различных архитектурных объектов разных веков.

В таблице 4 представлены результаты проверки метода на вышеперечисленных объектах. Здесь и далее в первой графе названия пар использованных сканов, во второй общее количество найденных алгоритмом точек, в третьей графе количество точек, использованных при уравнивании, во всех последующих графах представлены среднеквадратические погрешности по осям x, y и z соответственно.

Таблица 4 Результаты проверки метода с ручным набором точек

Сканы	Количество найденных точек	Количество точек используемых в уравнивании	СКП x	СКП y	СКП z
1\001-002_25	24	9	9.1мм	8.1 мм	1.1 мм
1\002-003_25	3	3	9.5 мм	5.2 мм	9.1 мм
2\007-008_25	15	10	4.4 мм	9.7 мм	5 мм
2\008-009_25	8	5	1.5 мм	9.2 мм	4.5 мм
2\009-010_25	15	6	6.7 мм	9.7 мм	1.9 мм
Kadet_korp\065-066_5	8	5	6 мм	8.6 мм	2.6 мм
Mohovaya 34\030-031_5	17	16	6.7 мм	7.8 мм	3.5 мм
Moskovskyi vok\003-008_15	7	4	8.3 мм	9.7 мм	9 мм

Solovki\086-087_5	7	7	8 .4мм	7.1 мм	6 мм
Solovki\085-086_5	10	9	4.8 мм	2.8 мм	3.5 мм
Solovki\084-085_5	13	13	4.3 мм	5.2 мм	3.9 мм
Solovki\079-080_15	23	11	5.2 мм	7 мм	3.6 мм
Solovki\078-079_15	20	9	7.0 мм	9 мм	6 мм
Solovki\077-078_15	24	19	3.7 мм	4.5 мм	4.8 мм
Solovki\075-077_15	18	15	7.4 мм	9.9 мм	9.6 мм
Solovki\038-040-15	17	16	2.4 мм	6.2 мм	3 мм
Solovki\033-034_5	9	7	2.3 мм	8.3 мм	2.1 мм
Solovki\032-033_5	10	10	3.5 мм	4.1 мм	3.7 мм
Solovki\001-003_5	12	12	5.1 мм	2.8 мм	4.4 мм
Solovki\000-001_5	12	12	4.4 мм	2.2 мм	3.2 мм
universitetskaya 3\ 005-006_10	8	5	8.9 мм	1.8 мм	2.9 мм
universitetskaya 3\006-007_10	4	3	5.4 мм	5.1 мм	7.3 мм

Из таблицы видно, что при наборе точек на одном ортофотоплане с дальнейшей фильтрацией, описанной во 2 главе, удается получать наборы опорных точек для большинства пар сканов в необходимом количестве и с результатом в пределах 1 сантиметра, что является приемлемым результатом для дальнейшей регистрации сканов по данным точкам.

3.3.3 Проверка работы метода при наборе точек методом SIFT

При проверке работы данного метода использовался тот же набор сканов, что и при ручном наборе для большей объективности получаемых результатов.

В таблице 5 представлены результаты проверки метода.

Таблица 5 Результаты проверки метода с автоматизированным набором точек

Сканы	Количество найденных точек	Количество точек используемых в уравнивании	СКП x	СКП y	СКП z
1\001-002_25	16	6	3.7 мм	4 мм	1 мм
2\009-010_25	39	29	6.7 мм	9.8 мм	1 мм
Kadet_korp\065-066_5	39	26	6.7 мм	9.1 мм	4.3 мм
Mohovaya 34\030-031_5	41	41	5.7 мм	7.6 мм	9 мм
Moskovskyi vok\003-008_15	57	26	7.2 мм	5.6 мм	6.9 мм
Solovki\084-085_5	128	46	5.2 мм	2.5 мм	6 мм
Solovki\078-079_15	175	82	2.1 мм	8.3 мм	1.8 мм
Solovki\077-078_15	147	76	3 мм	2.8 мм	6.3 мм
Solovki\075-077_15	124	72	2.9 мм	1.4 мм	1.4 мм
Solovki\038-040-15	153	97	1.6 мм	1.9 мм	1 мм
Solovki\001-003_5	108	105	6 мм	6 мм	5.7 мм
Solovki\000-	148	147	7.4 мм	6.8 мм	5.7 мм

001_5					
universitetskaya 3\ 005-006_10	49	35	6.5 мм	7 мм	9.6 мм
universitetskaya 3\006-007_10	110	82	8.9 мм	6.6 мм	9.5 мм
yaroslavl\59-67_potolok	21	16	1 мм	7.1 мм	6.4 мм
yaroslavl\67-68_potolok	222	127	2.7 мм	1.5 мм	6.7 мм
yaroslavl\68-70_potolok	104	47	9 мм	3.1 мм	6.2 мм
yaroslavl\68-70_stena	81	38	2.5 мм	5.3 мм	2.3 мм

По таблице видно, что как и в случае с таблицей 3, среднеквадратическая погрешность определения опорных точек получается в пределах 1 сантиметра, что позволяет сделать вывод о возможности применения данных точек для дальнейшей регистрации сканов.

Также данный метод позволяет набирать гораздо большее количество точек, в сравнении с ручным набором, что позволяет иметь существенный запас точек для выбора оптимального набора данных, используемых при регистрации.

Недостатком данного метода является значительное время расчета коэффициентов корреляции из-за большого количества выделяемых точек.

3.3.4 Проверка эффективности фильтрации точек с указанием местоположения

В предыдущих подразделах 3 главы были представлены результаты расчета среднеквадратической погрешности на парах сканов с набором точек вручную и автоматизировано, в них данные были получены без проведения фильтрации по местоположению. Для сравнения был проведен поиск опорных точек с фильтрацией по местоположению и составлены таблицы 6 и 7 для данных результатов.

Таблица 6 Результаты проверки метода с ручным набором точек с фильтрацией по местоположению

Сканы	Количество найденных точек	Количество точек используемых в уравнивании	СКП x	СКП y	СКП z
1\001-002_25	19	4	8 мм	3мм	3 мм
1\002-003_25	5	4	9.7 мм	13.3 мм	1 мм
1\003-005_25	3	3	2.7 мм	2.1 мм	15.8 мм
2\007-008_25	5	3	1 мм	2.3 мм	1.5 мм
2\008-009_25	11	5	1.8 мм	9.9 мм	3.3 мм
2\009-010_25	5	3	2.2 мм	5 мм	1 мм
Kadet_korp\065-066_5	10	7	1 мм	6 мм	1.9 мм
Mohovaya 34\030-031_5	9	7	5.9 мм	7.3 мм	2 мм
Moskovskyi vok\003-008_15	6	3	12.4 мм	17.1 мм	-1 мм
Solovki\086-087_5	9	8	1.6 мм	8.1 мм	2 мм
Solovki\085-086_5	7	6	1 мм	4.5 мм	2.5 мм
Solovki\084-085_5	7	7	1.3 мм	1 мм	1 мм
Solovki\079-080_15	11	5	4.9 мм	9.2 мм	3.3 мм
Solovki\078-079_15	11	4	5.6 мм	3.1 мм	4.1 мм
Solovki\077-078_15	21	12	1 мм	8.2 мм	8 мм
Solovki\075-077_15	23	12	3.2 мм	3.8 мм	9.5 мм
Solovki\038-	11	9	8.7 мм	3.9 мм	1.1 мм

040-15					
Solovki\033-034_5	6	6	2.9 мм	5.1 мм	1.2 мм
Solovki\032-033_5	16	14	5.2 мм	7.2 мм	3.1 мм
Solovki\001-003_5	12	12	6.6 мм	3.1 мм	6.8 мм
Solovki\000-001_5	16	16	9.1 мм	2.7 мм	6 мм
universitetskaya 3\ 005-006_10	10	9	9.2 мм	2.1 мм	7.7 мм
universitetskaya 3\006-007_10	6	3	1.1 мм	1 мм	8 мм

Таблица 7 Результаты проверки метода с автоматизированным набором точек с фильтрацией по местоположению

Сканы	Количество найденных точек	Количество точек используемых в уравнивании	СКП x	СКП y	СКП z
1\001-002_25	92	10	9.4 мм	8.2 мм	2.5 мм
1\003-005_25	61	6	5.2 мм	8.7 мм	1 мм
2\009-010_25	53	32	6.7 мм	8.6 мм	1 мм
Kadet_korp\065-066_5	79	74	5.8 мм	8.5 мм	7.5 мм
Mohovaya 34\030-031_5	158	37	2.4 мм	4.7 мм	4.7 мм
Moskovskyi vok\003-008_15	97	29	7.6 мм	3.5 мм	2.2 мм
Solovki\084-085_5	176	152	3.3 мм	3.2 мм	2 мм
Solovki\078-079_15	139	58	5 мм	1.5 мм	2.6 мм
Solovki\075-	145	73	3 мм	2.5 мм	1.6 мм

077_15					
Solovki\038-040-15	239	168	1.8 мм	1.3 мм	2.2 мм
Solovki\001-003_5	199	141	9 мм	5.4 мм	4.8 мм
Solovki\000-001_5	212	209	7.9 мм	5.7 мм	8.6 мм
universitetskaya 3\006-007_10	50	44	9.5 мм	7.2 мм	5.4 мм
yaroslavl\59-67_potolok	55	21	1 мм	8.5 мм	6.4 мм
yaroslavl\67-68_potolok	200	52	8.7 мм	1 мм	6.5 мм
yaroslavl\68-70_potolok	171	27	1 мм	1 мм	3.7 мм
yaroslavl\68-70_stena	22	17	4 мм	8.1 мм	9 мм

Если сравнить таблицы 4 и 5, с результатами работы метода без фильтрации по местоположению и таблицы 6 и 7 с фильтрацией видно, что данная функция не оказывает существенного положительного влияния на получаемые результаты, напротив в некоторых случаях результаты расчета среднеквадратической погрешности для пар сканов с ручным указанием точек значительно ухудшились до 1,5 – 2 сантиметров. Основываясь на сравнении данных таблиц можно заключить, что использование фильтрация точек с использованием их местоположения на ортофотоплане нецелесообразна и может ухудшить получаемые результаты.

3.3.5 Проверка работы метода при указании точек на двух ортофотопланах

Основной проблемой при расчете коэффициентов корреляции точек на ортофотопланах является то, что алгоритм производит поиск точек на всем изображении, что увеличивает вероятность ошибочного исхода. Чтобы этого избежать необходимо указывать область, в которой будет производится поиск, указание точек на двух ортофотопланах с последующим вычислением смещения для автоматического нанесения центра области поиска на второй ортофотоплан, позволяет решить данную проблему. В

таблице 8 представлены результаты расчета среднеквадратической погрешности для пар ортофотопланов, на которых точки были указаны подобным образом.

Таблица 8 Результаты проверки метода с указанием точек на двух ортофотопланах

Сканы	Количество найденных точек	Количество точек используемых в уравнивании	СКП x	СКП y	СКП z
1\001-002_25	5	3	1.7 мм	2.6 мм	2.7 мм
2\008-009_25	5	3	8.4 мм	2.4 мм	9.9 мм
2\009-010_25	3	3	4.6 мм	1.5 мм	1.2 мм
Kadet_korp\065-066_5	6	3	1 мм	6.5 мм	1.7 мм
Mohovaya 34\030-031_5	8	8	3.1 мм	1.8 мм	1.7 мм
Moskovskyi vok\003-008_15	3	3	1.6 мм	4.9 мм	3.1 мм
Solovki\084-085_5	10	10	2.5 мм	2.2 мм	2.6 мм
Solovki\078-079_15	11	5	2 мм	1 мм	1 мм
Solovki\075-077_15	13	12	1 мм	4.6 мм	1 мм
Solovki\038-040-15	11	6	1 мм	1.2 мм	1 мм
Solovki\001-003_5	6	5	2.8 мм	1 мм	1.5 мм
Solovki\000-001_5	5	5	5 мм	1 мм	1.8 мм
universitetskaya 3\006-007_10	7	4	2.5 мм	4.7 мм	8 мм
yaroslavl\67-68_potolok	9	4	1 мм	1 мм	1 мм

Данную таблицу необходимо сравнить с таблицей 4, так как обе они были созданы на основе точек, которые набраны вручную. Проанализировав таблицу можно сделать вывод, что данный метод не улучшает полученные результаты, в таблице 8 результаты в среднем хуже на 1-2 миллиметра, что все равно является хорошим результатом, однако главным достоинством данного метода может стать использование его для указания областей поиска для точек набранных по методу SIFT, что позволит значительно ускорить получение результатов.

3.3.6 Проверка работы метода на повернутых сканах

Все предыдущие рассмотренные функции возможно применять только для предварительно ориентированных сканов, для которых есть возможность создать не повернутые друг относительно друга по вертикальной оси ортофотопланы. Однако, в значительном количестве случаев, получаемые в ходе съемок лазерным сканером материалы получаются произвольно ориентированными в пространстве.

Для устранения данной проблемы необходимо определить оптимальный угол вращения, на который будет поворачиваться скан, чтобы найти приемлемое для проведения поиска опорных точек положение за минимальное количество итераций при автоматизированном процессе регистрации.

В ходе исследования были проверены сканы, которые были повернуты на углы 11.25, 22.5 и 45 градусов. В результате при повороте на угол 45 градусов метод нахождения общих точек путем расчета коэффициентов корреляции показывает неудовлетворительные результаты и не находит общих точек, а в случае с поворотом на 11.25 и 22.5 градуса находит общие точки. Поэтому в таблице 9 представлены результаты определения качества нахождения опорных точек для пар сканов повернутых на данные углы.

Таблица 9 Результат проверки сканов с различным углом поворота

Сканы	Количество найденных точек	Количество точек используемых в уравнивании	СКП x	СКП y	СКП z
2\007-008_11.5	11	9	4.6 мм	3.3 мм	5.2 мм
2\007-008_22.5	11	9	7.2 мм	9.2 мм	9.1 мм
2\007-010_11.5	11	8	1.3 мм	1.8 мм	1.8 мм

2\007-010_22.5	4	3	3 мм	5.1 мм	4 мм
Kadet_korp\065-066_11.5	14	12	1.4 мм	3.5 мм	4.5 мм
Kadet_korp\065-066_22.5	10	7	1.7 мм	1.9 мм	1 мм
Moskovskyi vok\001-008_11.5	8	4	1 мм	1 мм	1 мм
Moskovskyi vok\001-008_22.5	4	3	5 мм	1 мм	3.7 мм
Solovki\084-085_11.5	11	11	3 мм	6.8 мм	2.9 мм
Solovki\084-085_22.5	12	12	1.7 мм	7.3 мм	2.6 мм
Solovki\074-075_11.5	12	8	1.3 мм	2.7 мм	3.3 мм
Solovki\074-075_22.5	15	12	4.4 мм	9.4 мм	3.9 мм
Solovki\000-001_11.5	11	11	2.4 мм	3.9 мм	4.1 мм
Solovki\000-001_22.5	9	9	4.1 мм	6.3 мм	6.1 мм

На основе данной таблицы можно заключить что оптимальным углом для поворота скана если при автоматизированной регистрации является угол 22.5 градуса, на данный угол больше 11.5 следовательно понадобится теоретически понадобится меньшее количество итераций для нахождения общих точек, а получаемая на основе этих измерений среднеквадратическая погрешность существенно не отличается.

Заключение

В ходе реализации поставленной цели были достигнуты следующие задачи:

- Рассмотрены и изучены методы регистрации данных лазерного сканирования, методы поиска общих точек на ортофотопланах с использованием алгоритмов компьютерного зрения;
- Исследованы и реализованы на языке Python методы отбора общих точек на ортофотопланах;
- Разработан алгоритм поиска общих точек на ортофотопланах;
- Проведены исследования по проверке работоспособности метода на реальных объектах.

Из результатов анализа исследований следует вывод, что рассмотренные методы выбора, нахождения и фильтрации опорных точек на ортофотопланах могут успешно применяться для цели регистрации данных лазерного сканирования. Расчеты показывают, что среднеквадратическая погрешность нахождения точек рассмотренными методами может находиться в пределах 1 сантиметра, что позволяет использовать их для приведения сканов друг к другу.

Результаты, полученные в данном исследовании, можно использовать как основу для создания и развития автоматической системы регистрации данных лазерного сканирования.

Литература

Литературные источники:

1. Комиссаров А.В. «Теория и технология лазерного сканирования для пространственного моделирования территорий», 2016, 279 с.

2. Середович В.А., Комиссаров А.В., Комиссаров Д.В., Широкова Т.А. «Наземное лазерное сканирование», Новосибирск: СГГА, 2009., 261 с.

3. Шамарина А. А., Мезенина К. О. «Методика наземного лазерного производства и обработка данных при хранении объектов историко-культурного наследия», Вестник ПНИПУ. Прикладная экология. Урбанистика. 2016. № 2. С. 45.

4. Хорошилова Ж. А. «О возможности построения математических моделей инженерных объектов по данным лазерного сканирования», Геодезия, геоинформатика, картография, маркшейдерия: сборник материалов в 3 томах Международной научной конференции. - Новосибирск : СГГА, 2013., Т. 3., С. 116 - 119.

5. Шамарина А.А. «Подбор уникальных моделей наземного лазерного сканера для анализа городской среды», AMIT . 2015. №2 (31)

6. Geoff Jacobs. «3d laser scanning // Ultra-fast, High-Definition, Reflector less Topographic Survey», Professional Surveyor magazine №5., 2004

7. Geoff Jacobs. «Uses in Building and Architectural Surveys», Professional Surveyor magazine №6. - 2005

8. Geoff Jacobs. «Uses of Scanning in Construction and Fabrication», Professional Surveyor magazine N 2., 2006

9. Besl P.J., McKay N.D. «A method for registration of 3-D shapes», IEEE Trans. Pattern Anal. Mach. Intell 1992, *14*(2), 239–256.

10. Godin G., Boulanger, P. «Range image registration through viewpoint invariant computation of curvature», International Archives of Photogrammetry and Remote Sensing, Zurich, Switzerland, 1995; 30, pp. 170–175.

11. Sharp G.C., Lee S.W., Wehe D.K. «ICP registration using invariant features», IEEE Trans. Pattern Anal. Mach. Intell 2002, *24*(1), 90–102.

12. Sequeira V., Ng K., Wolfart E., Goncalves J.G.M., Hogg, D. «Automated reconstruction of 3D models from real environments», ISPRS J. Photogramm. Remote Sens 1999, *54*(1), 1–22.

13. Zhengyou Zhang, «Iterative Point Matching for Registration of Free-Form Curves and Surfaces» INRIA Sophia-Antipolis, route des Lucioles, BP 93, F-06902 Sophia-Antipolis Cedex-FRANCE, 2004

14. David G. Lowe. «Object recognition from local scale-invariant features», International Conference on Computer Vision, Corfu, Greece, 1999

15. Herbert Bay, Andreas Ess, Tinne Tuytelaars, Luc Van Gool, «Speeded Up Robust Features», ETH Zurich, Katholieke Universiteit Leuven, 2006

16. David G. Lowe. «Distinctive Image Features from Scale-Invariant Keypoints», Computer Science Department University of British Columbia Vancouver, B.C., Canada, 2004

17. Тихонов С.Г., Войнаровский А.Е. «Руководство по ScanIMAGER», Санкт-Петербург: ООО «НПП Фотограмметрия», 2014

18. Szymon Rusinkiewicz, Marc Levoy, «Efficient Variants of the ICP Algorithm», International Conference on 3D Digital Imaging and Modeling (3DIM), 2001.

Электронные ресурсы:

19. What is NumPy?// Официальный сайт NumPy. URL: https://numpy.org/doc/stable/user/whatisnumpy.html(дата обращения: 07.02.2022)

20. AboutOpenCV // Официальный сайт OpenCV. URL: https://opencv.org/about/(дата обращения: 07.02.2022)

21. 16. Histograms - 2: Histogram Equalization // Онлайн учебник по OpenCV. URL: https://docs.opencv.org/4.x/d5/daf/tutorial_py_histogram_equalization.html (дата обращения 14.03.2022)

22. Image Blending Using Laplacian Pyramids // Becoming Human: Artificial Intelligence Magazine. URL: https://becominghuman.ai/image-blending-using-laplacian-pyramids-2f8e9982077f (дата обращения 22.05.2022)

23. Трехмерное лазерное сканирование // Инфопедия. URL: https://infopedia.su/8x20b3.html (дата обращения 26.04.2022)

24. Лидар или световой радар. Лазерная система управления ветротурбинами // Наука и техника. URL: https://naukatehnika.com/lazernaya-sistema-upravleniya-vetroturbinami.html (дата обращения 26.04.2022)

25. Принцип работы фазового дальномера. // Лазерный портал.URL: https://laserportal.ru/content_1268 (дата обращения 26.04.2022)

26. Современные типы 3d сканеров и их особенности. // MyGadgetShop. URL: https://mygs.ru/blog/sovremennye-tipy-3d-skanerov-i-ih-osobennosti (дата обращения 26.04.2022)

27. Template Matching // учебник по OpenCV. URL: https://docs.opencv.org/3.4/de/da9/tutorial_template_matching.html (дата обращения 26.04.2022)

Приложение

```
import cv2
import numpy

defvidelenie(event,x,y,flags,param):
global mouseX,mouseY
if event == cv2.EVENT_LBUTTONDBLCLK:
            cv2.line(img_1_copy, (x-15, y), (x+15, y), (0, 0, 0), 2)
            cv2.line(img_1_copy, (x, y-15), (x, y+15), (0, 0, 0), 2)
mouseX, mouseY = x, y
elifevent==cv2.EVENT_MOUSEMOVE:
        k =10
font = cv2.FONT_HERSHEY_SIMPLEX
        org = (x, y)
roi = img 1 copy[y - k:y + k, x - k:x + k]
disp = numpy.around(numpy.var(roi),3)
        img 1 copy2 = img 1 copy.copy()
        cv2.putText(img_1_copy2, '{}'.format(disp), org, font, 1, (145, 45, 153), 2, cv2.LINE_8)
        cv2.namedWindow('image', cv2.WINDOW_KEEPRATIO)
        cv2.imshow('image', img 1 copy2)
defvidelenie 2(event,x,y,flags,param):
global mouseX r,mouseY r
if event == cv2.EVENT_LBUTTONDBLCLK:
            cv2.line(img 2 copy, (x-15, y), (x+15, y), (0, 0, 0), 2)
            cv2.line(img_2_copy, (x, y-15), (x, y+15), (0, 0, 0), 2)
mouseX r, mouseY r = x, y
defsub(corr):
min_val,  max_val,  min_loc,  max_loc  =  cv2.minMaxLoc(corr)
    d  =  numpy.mgrid[0:3,0:3]  +  0.5
d_x  =  d[1]
d y  =  d[0]
corr_n  =  corr[max_loc[1]  -  1:max_loc[1]  +  2,  max_loc[0]  -  1:max_loc[0]  +  2]
    print(numpy.shape(d x),numpy.shape(d y),numpy.shape(corr n))
    x  =  numpy.sum(numpy.multiply(d x,  corr n))  /  (numpy.sum(corr n))
    y  =  numpy.sum (numpy.multiply (d_y,  corr_n))  /  (numpy.sum (corr_n))
return x, y
deftile(img, dir_out):
global W,H
    height, width = img.shape
    W = 6
H = 3
i = 1
for ihin range(H):
for iwin range(W):
            x = int(width/W * iw)
            y = int(height/H * ih)
            h = int(height / H)
            w = int(width / W)
imgcrop = img[y:y+h, x:x+w]
            NAME = str(i)
            cv2.imwrite(dir_out + NAME + ".jpg", imgcrop)
            i = i+1
deffiltr(tchk1,tchk2):
    tchk1 = numpy.asarray(tchk1, dtype=numpy.float32)
    tchk2 = numpy.asarray(tchk2, dtype=numpy.float32)
    rasst1 = numpy.zeros((tchk1.shape[0],tchk1.shape[0]))
    tchk1 nai = []
    tchk2_nai = []
for i in range(tchk1.shape[0]):
for kin range(tchk1.shape[0]):
rasst_schet = numpy.sqrt((tchk1[i,0]-tchk1[k,0])**2+(tchk1[i,1]-tchk1[k,1])**2)
            rasst1[i,k] = rasst schet
    rasst2 = numpy.zeros((tchk2.shape[0],tchk2.shape[0]))
for i in range(tchk2.shape[0]):
for kin range(tchk2.shape[0]):
rasst_schet = numpy.sqrt((tchk2[i,0]-tchk2[k,0])**2+(tchk2[i,1]-tchk2[k,1])**2)
            rasst2[i,k] = rasst schet
rasst_minus = rasst1 - rasst2
for i in range(rasst minus.shape[0]):
        p = rasst_minus[i,:]
        k = numpy.where(numpy.abs(p)<1,1,0)
        r = numpy.count nonzero(k)
        print("r=",r)
if r>(0.1*rasst_minus.shape[0]):
```

```
            tchk1_nai.append([tchk1[i,0],tchk1[i,1]])
            tchk2_nai.append([tchk2[i,0],tchk2[i,1]])
    print(rasst_minus)
return tchk1_nai,tchk2_nai
defcorr(img_l,img_r):
global pts, pts_r1
i_max = pts.shape[0]
        k = 30
img_r_sub = img_r.copy()
img_l_sub = img_l.copy()
pts_r = []
pts_l = []
pts = numpy.asarray(pts, dtype=numpy.int64)
dispercia = []
korrel = []
        pts_2 = []
        pts_r_2 = []
        m = 0
for i in range(i_max):
if (pts[i][1]<50) or (pts[i][1]>img_l.shape[0]-50) or (pts[i][0]<50) or
(pts[i][0]>img_l.shape[1]-50):
continue
roi = img_l[pts[i][1] - k:pts[i][1] + k, pts[i][0] - k:pts[i][0] + k]
if cv2.countNonZero(roi)<(0.99*2*k*2*k):
continue
disp = numpy.var(roi)
roi_r = img_r[pts_r1[i][1] - k*2:pts_r1[i][1] + k*2, pts_r1[i][0] - k*2:pts_r1[i][0] + k*2]
            res = cv2.matchTemplate(roi_r, roi, method=cv2.TM_CCOEFF_NORMED)
min_val, max_val, min_loc, max_loc = cv2.minMaxLoc(res)
dispercia.append(disp)
korrel.append(max_val)
            x, y = sub(res)
            t  =  [x-1.5+max_loc[0]+pts_r1[i][0] - k,  y-1.5+max_loc[1]+pts_r1[i][1] - k]
res_l = cv2.matchTemplate(img_l, roi, method=cv2.TM_CCOEFF_NORMED)
x_l,y_l = sub(res_l)
t_l  =  [x_l-1.5+max_loc[0]+pts[i][0] - k,  y_l-1.5+max_loc[1]+pts[i][1] - k]
#pts_l.append(t_l)
pts_r.append(t)
#roi_2 = img_l[pts_r[m][1] - k:pts_r[m][1] + k, pts_r[m][0] - k:pts_r[m][0] + k]
            #if cv2.countNonZero(roi_2)<(0.99*2*k*2*k):
                #continue
            #t = [pts_l[i][0],pts_l[i][1]]
pts_2.append(t_l)
            print(pts_2)
            t = [pts_r[m][0],pts_r[m][1]]
            pts_r_2.append(t)
            m = m+1
pts_2,pts_r_2 = filtr(pts_2,pts_r_2)
for i in range(numpy.asarray(pts_2).shape[0]):
x,y=round(pts_r_2[i][0]),round(pts_r_2[i][1])
            cv2.line(img_r_sub, (x - 15, y), (x + 15, y), (0, 0, 0), 2)
            cv2.line(img_r_sub, (x, y - 15), (x, y + 15), (0, 0, 0), 2)
x,y=round(pts_2[i][0]),round(pts_2[i][1])
            cv2.line(img_l_sub, (x - 15, y), (x + 15, y), (0, 0, 0), 2)
            cv2.line(img_l_sub, (x, y - 15), (x, y + 15), (0, 0, 0), 2)
            print(x,y)
dispercia = numpy.asarray(dispercia, dtype=numpy.float64)
pts_r = numpy.asarray(pts_r, dtype=numpy.int64)
korrel = numpy.asarray(korrel, dtype=numpy.float64)
while(1):
            cv2.destroyAllWindows()
            cv2.namedWindow('img2', cv2.WINDOW_KEEPRATIO)
            cv2.imshow('img2',img_r_sub)
            cv2.namedWindow('img1', cv2.WINDOW_KEEPRATIO)
            cv2.imshow('img1',img_l_sub)
            k = cv2.waitKey(0) &0xFF
if k == ord('s'):
numpy.savetxt('D:/Kursovie/NIR/prov_korrel/Result_2/result_ukazannoe.txt', pts_2, fmt= '%-4d')
numpy.savetxt('D:/Kursovie/NIR/prov_korrel/Result_2/result_naidennoe.txt', pts_r_2, fmt= '%-4d')
                cv2.destroyAllWindows()
break
defcorr_2(img_l,img_r):
global pts,W,H
i_max = pts.shape[0]
        k = 30
img_r_sub = img_r.copy()
img_l_sub = img_l.copy()
pts_r = []
pts_l = []
pts = numpy.asarray(pts, dtype=numpy.int64)
```

```
dispercia = []
korrel = []
        pts_2 = []
        pts_r_2 = []
        m = 0
for i in range(i_max):
if (pts[i][1]<50) or (pts[i][1]>img_l.shape[0]-50) or (pts[i][0]<50) or
(pts[i][0]>img_l.shape[1]-50):
continue
roi = img_l[pts[i][1] - k:pts[i][1] + k, pts[i][0] - k:pts[i][0] + k]
if cv2.countNonZero(roi)<(0.99*2*k*2*k):
continue
g = pts[i][0]//(img_l.shape[1]/W)
            p = pts[i][1]//(img_l.shape[0]/H)
ch = round((g+1)+(p*6))
            print(W,H,g,p)
            print(ch)
            print(img_l.shape[0])
img_bl_bl = cv2.imread('D:/Kursovie/NIR/prov_korrel/sdvinutye/TILE/img_2/'+ str(ch)+'.jpg')
img_bl = cv2.cvtColor(img_bl_bl, cv2.COLOR_BGR2GRAY)
disp = numpy.var(roi)
            res = cv2.matchTemplate(img_bl, roi, method=cv2.TM_CCOEFF_NORMED)
min_val, max_val, min_loc, max_loc = cv2.minMaxLoc(res)
dispercia.append(disp)
korrel.append(max_val)
#x, y = sub(res)
            #t  =  [x-1.5+max_loc[0]+k,  y-1.5+max_loc[1]+k]
res_l = cv2.matchTemplate(img_l, roi, method=cv2.TM_CCOEFF_NORMED)
            x, y = max_loc[0] + k, max_loc[1] + k
            t = [x,y]
res_l = cv2.matchTemplate(img_l, roi, method=cv2.TM_CCOEFF_NORMED)
min_val, max_val, min_loc, max_loc = cv2.minMaxLoc(res_l)
#x_l,y_l = sub(res_l)
            #t_l  =  [x_l-1.5+max_loc[0]+k,  y_l-1.5+max_loc[1]+k]
            #pts_l.append(t_l)
x_l, y_l = max_loc[0] + k, max_loc[1] + k
t_l = [x_l,y_l]
pts_r.append(t)
#roi_2 = img_l[pts_r[m][1] - k:pts_r[m][1] + k, pts_r[m][0] - k:pts_r[m][0] + k]
            #if cv2.countNonZero(roi_2)<(0.99*2*k*2*k):
                #continue
            #t = [pts_l[i][0],pts_l[i][1]]
pts_2.append(t_l)
            t = [pts_r[m][0],pts_r[m][1]]
            pts_r_2.append(t)
            m = m+1
#pts_2,pts_r_2 = filtr(pts_2,pts_r_2)
for i in range(numpy.asarray(pts_2).shape[0]):
x,y=pts_r_2[i][0],pts_r_2[i][1]
            cv2.line(img_r_sub, (x - 15, y), (x + 15, y), (0, 0, 0), 2)
            cv2.line(img_r_sub, (x, y - 15), (x, y + 15), (0, 0, 0), 2)
x,y=pts_2[i][0],pts_2[i][1]
            cv2.line(img_l_sub, (x - 15, y), (x + 15, y), (0, 0, 0), 2)
            cv2.line(img_l_sub, (x, y - 15), (x, y + 15), (0, 0, 0), 2)
dispercia = numpy.asarray(dispercia, dtype=numpy.float64)
pts_r = numpy.asarray(pts_r, dtype=numpy.int64)
korrel = numpy.asarray(korrel, dtype=numpy.float64)
while(1):
            cv2.destroyAllWindows()
            cv2.namedWindow('img2', cv2.WINDOW_KEEPRATIO)
            cv2.imshow('img2',img_r_sub)
            cv2.namedWindow('img1', cv2.WINDOW_KEEPRATIO)
            cv2.imshow('img1',img_l_sub)
            k = cv2.waitKey(0) &0xFF
if k == ord('s'):
numpy.savetxt('D:/Kursovie/NIR/prov_korrel/Result_2/result_ukazannoe.txt', pts_2, fmt= '%-4d')
numpy.savetxt('D:/Kursovie/NIR/prov_korrel/Result_2/result_naidennoe.txt', pts_r_2, fmt= '%-4d')
                cv2.destroyAllWindows()
break
defcorr_3(img_l,img_r):
global pts, pts_r1
        k = 30
img_r_sub = img_r.copy()
img_l_sub = img_l.copy()
pts_r = []
pts_l = []
pts = numpy.asarray(pts, dtype=numpy.int64)
        pts_r1 = numpy.asarray(pts_r1, dtype=numpy.int64)
i_max = pts.shape[0]
z_max = pts_r1.shape[0]
```

```
        pts_2 = []
        pts_r_2 = []
        m = 0
for i in range(i_max):
if (pts[i][1]<50) or (pts[i][1]>img_l.shape[0]-50) or (pts[i][0]<50) or
(pts[i][0]>img_l.shape[1]-50):
continue
roi = img_l[pts[i][1] - k:pts[i][1] + k, pts[i][0] - k:pts[i][0] + k]
if cv2.countNonZero(roi)<(0.99*2*k*2*k):
continue
maxx = -1
for z in range(z_max):
if (pts_r1[z][1]<50) or (pts_r1[z][1]>img_r.shape[0]-50) or (pts_r1[z][0]<50) or
(pts_r1[z][0]>img_r.shape[1]-50):
continue
roi_r = img_r[pts_r1[z][1] - k:pts_r1[z][1] + k, pts_r1[z][0] - k:pts_r1[z][0] + k]
                res = cv2.matchTemplate(roi_r, roi, method=cv2.TM_CCOEFF_NORMED)
min_val, max_val, min_loc, max_loc = cv2.minMaxLoc(res)
if max_val>maxx:
maxx = max_val
z_fin = z
roi_r = img_r[pts_r1[z_fin][1] - k*2:pts_r1[z_fin][1] + k*2, pts_r1[z_fin][0] -
k*2:pts_r1[z_fin][0] + k*2]
            res = cv2.matchTemplate(roi_r, roi, method=cv2.TM_CCOEFF_NORMED)
min_val, max_val, min_loc, max_loc = cv2.minMaxLoc(res)
            x, y = sub(res)
            t  =  [x-1.5+max_loc[0]+pts_r1[i][0] - k,  y-1.5+max_loc[1]+pts_r1[i][1] - k]
res_l = cv2.matchTemplate(img_l, roi, method=cv2.TM_CCOEFF_NORMED)
x_l,y_l = sub(res_l)
t_l  =  [x_l-1.5+max_loc[0]+pts[i][0] - k,  y_l-1.5+max_loc[1]+pts[i][1] - k]
pts_r.append(t)
            pts_2.append(t_l)
        pts_2,pts_r_2 = filtr(pts_2,pts_r)
for i in range(numpy.asarray(pts_2).shape[0]):
x,y=round(pts_r_2[i][0]),round(pts_r_2[i][1])
            cv2.line(img_r_sub, (x - 15, y), (x + 15, y), (0, 0, 0), 2)
            cv2.line(img_r_sub, (x, y - 15), (x, y + 15), (0, 0, 0), 2)
x,y=round(pts_2[i][0]),round(pts_2[i][1])
            cv2.line(img_l_sub, (x - 15, y), (x + 15, y), (0, 0, 0), 2)
            cv2.line(img_l_sub, (x, y - 15), (x, y + 15), (0, 0, 0), 2)
            print(x,y)
pts_r = numpy.asarray(pts_r, dtype=numpy.float64)
while(1):
            cv2.destroyAllWindows()
            cv2.namedWindow('img2', cv2.WINDOW_KEEPRATIO)
            cv2.imshow('img2',img_r_sub)
            cv2.namedWindow('img1', cv2.WINDOW_KEEPRATIO)
            cv2.imshow('img1',img_l_sub)
            k = cv2.waitKey(0) &0xFF
if k == ord('s'):
numpy.savetxt('D:/Kursovie/NIR/prov_korrel/Result_2/Tile/1/auto/SIFT/result_ukazannoe.txt',
pts_2, fmt='%1.3f')
numpy.savetxt('D:/Kursovie/NIR/prov_korrel/Result_2/Tile/1/auto/SIFT/result_naidennoe.txt',
pts_r_2, fmt='%1.3f')
                cv2.destroyAllWindows()
break
img_1_1 = cv2.imread("D:/Kursovie/NIR/prov_korrel/Sdvinutye/1/001_25.jpg")
#img_1_1 = cv2.imread("D:/Kursovie/NIR/prov_korrel/povoroty/0.jpg")
img_1 = cv2.cvtColor(img_1_1, cv2.COLOR_BGR2GRAY)
#clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(35,35))
#img_1 = clahe.apply(img_1)
img_2_2 = cv2.imread("D:/Kursovie/NIR/prov_korrel/Sdvinutye/1/002_25.jpg")
#img_2_2 = cv2.imread("D:/Kursovie/NIR/prov_korrel/povoroty/40.jpg")
img_2 = cv2.cvtColor(img_2_2, cv2.COLOR_BGR2GRAY)
#clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(35,35))
#img_2 = clahe.apply(img_2)
pts = []
pts_r1 = []
img_1_copy = img_1.copy()
img_1_copy2 = img_1_copy.copy()
img_2_copy = img_2.copy()
tile(img_1,'D:/Kursovie/NIR/prov_korrel/sdvinutye/TILE/img_1/')
tile(img_2,'D:/Kursovie/NIR/prov_korrel/sdvinutye/TILE/img_2/')
cv2.namedWindow('image', cv2.WINDOW_KEEPRATIO)
cv2.setMouseCallback('image', videlenie)
cv2.namedWindow('image_r', cv2.WINDOW_KEEPRATIO)
cv2.setMouseCallback('image_r', videlenie_2)
p = 0
sumx = 0
sumy = 0
```

```
while(1):
    cv2.imshow('image',img 1 copy)
    cv2.imshow('image_r',img_2_copy)
    k = cv2.waitKey(0) &0xFF
if k == ord('q'):
break
elifk == ord('a'):
        t = [mouseX,mouseY]
pts.append(t)
if p <3:
t r = [mouseX r,mouseY r]
            pts r1.append(t r)
pts minusx = pts r1[p][0] - pts[p][0]
pts_minusy = pts_r1[p][1] - pts[p][1]
sumx = sumx + pts_minusx
sumy = sumy + pts_minusy
            print(pts minusx)
            print(pts minusy)
elifp >= 3:
t r = [mouseX + round(sumx/3),mouseY + round(sumy/3)]
            pts_r1.append(t_r)
            x = mouseX + round(sumx/3)
            y = mouseY + round(sumy/3)
            cv2.line(img 2 copy, (x - 15, y), (x + 15, y), (0, 0, 0), 2)
            cv2.line(img 2 copy, (x, y - 15), (x, y + 15), (0, 0, 0), 2)
        p = p+1
print(pts)
        print(pts_r1)
elifk == ord('c'):
        sift = cv2.SIFT create()
kp = sift.detect(img_1,None)
        img 1 copy = cv2.drawKeypoints(img 1,kp,img 1 copy)
pts = cv2.KeyPoint_convert(kp)
        sift = cv2.SIFT create()
kp = sift.detect(img 2,None)
        img_2_copy = cv2.drawKeypoints(img_2,kp,img_2_copy)
        pts_r1 = cv2.KeyPoint_convert(kp)
elifk == ord('m'):
mser = cv2.MSER create(min area = 1, max area = 1)
        regions, boxes = mser.detectRegions(img_1)
for box in boxes:
            x, y, w, h = box
            cv2.rectangle(img_1_copy2, (x,y),(x+w, y+h), (255, 0, 0), 2)
            t = [x,y]
pts.append(t)
        cv2.namedWindow('mser', cv2.WINDOW KEEPRATIO)
        cv2.imshow('mser', img_1_copy2)
pts = numpy.asarray(pts, dtype=numpy.int64)
corr 3(img 1,img 2)
cv2.waitKey(0)
cv2.destroyAllWindows()
```