РЕФЕРАТ

Отчет: 32 страницы, 11 рисунка, 2 таблицы, 10 источников, 1 приложение.

ОБРАБОТКА ИЗОБРАЖЕНИЙ, ДЕСКРИПТОРЫ ТОЧЕК, DAISY, PYTHON.

В данной работе рассмотрены алгоритмы построения дескрипторов в задачах обработки изображений.

Цель работы – разработка устойчивого к пространственным преобразованиям алгоритма построения дескрипторов.

Рассмотрены принципы построения дескрипторов изображений, создана программная реализация с применением алгоритма DAISY, проведено экспериментальное исследование работы алгоритма.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В ряде задач цифровой обработки изображений возникает необходимость однозначного численного описания представленных на изображении точек. В частности, на численном описании отдельных точек и областей изображения основаны решения задач идентификации объектов, совмещения изображений, построения карт глубины и восстановления трехмерной сцены методами фотограмметрии, и многих других.

Математические объекты, описывающие точку изображения, называют дескрипторами. Существует значительное число алгоритмов построения дескрипторов, отличающихся деталями своей реализации, скоростью и точностью работы, а также корректностю результатов для различных исходных данных.

Основными требованиями к дескриптору являются однозначность результата для одинаковых точек изображений, устойчивость к пространственным и яркостным преобразованиям исходных данных, а также вычислительная сложность алгоритмов построения.

Целью работы является разработка дескриптора изображений, устойчивого к пространственным преобразованиям.

В данной работе были рассмотрены некоторые дескрипторы и алгоритмы их построения, методы программной реализации. Разработана реализация построения дескрипторов на основе алгоритма0 DAISY.

Используемые алгоритмы реализованы на языке программирования общего назначения Python 3.7 с использованием open-source библиотек для обработки изображений OpenCV версии 3.4.2 и scikit-image версии 0.6.12. В качестве тестовых данных использовались наборы изображений из ряда открытых источников.

1 Исследование алгоритмов построения дескрипторов изображений

1.1 Постановка задачи

Дескриптором называют математический объект, сопоставленный с определенной точкой изображения, и представляющий достаточно однозначное ее описание, позволяющее с высокой степенью уверенности идентифицировать аналогичную точку или область на другом изображении.

Как правило, дескриптор представляет собой вектор значений, вычисляемый определенной функцией для точки изображения.

Стоит заметить, что на практике для однозначного описания точки изображения недостаточно исключительно информации о яркости отдельного пикселя. В связи с этим, в качестве исходных данных для построения дескриптора используется набор из нескольких точек изображения, находящихся в окрестности заданной.

Рассмотрим следующую постановку задачи. Обозначим дескриптор точки p как d_p. Пусть дано изображение:

$$I : S \to \mathbb{R}, \quad S \subset \mathbb{R}^k, \tag{1}$$

где k – размерность, для плоских изображений равная 2.

Построение дескриптора будет выглядеть следующим образом:

$$d_p = f(I(\hat{p})), \quad \hat{p} \in S. \tag{2}$$

где f – функция построения дескриптора,

$\hat{p}$ – набор точек изображения I.

Данная схема сохраняется независимо от конкретного алгоритма построения дескриптора. Конкретный вид и свойства полученного результа-

та будут зависеть как от вида функции f, так и от конфигурации набора исходных точек $\hat{p}$.

Основным требованием к дескриптору является однозначность результата, т.е. удовлетворение значения (2) следующему выражению для произвольного числа изображений N:

$$f(I_i(\hat{p_i})) = f(I_j(\hat{p_j})), \quad i = 1 \dots N, \quad j = 1 \dots N, \tag{3}$$

где p_i, p_j – пара соответствующих точек изображений I_i, I_j.

Следует заметить, что построенные дескрипторы, как правило, подвергаются нормализации для уменьшения влияния на точность идентификации точек яркостных характеристик изображения и пространственных преобразований. Конкретные методы нормализации будут рассмотрены в следующих разделах.

Таким образом, используя различные функции построения и варьируя набор исходных точек, можно получать дескрипторы, обладающие различными свойствами, достоинствами и недостатками для конкретных задач.

Основными факторами, принимаемыми в расчет при разработке алгоритма построения дескрипторов, являются:

1. Устойчивость к пространственным преобразованиям - сохранение значения дескриптора при повороте, сдвиге, масштабировании изображения;

2. Устойчивость к яркостным преобразованиями - сохранение значения дескриптора при изменении яркости и контрастности;

3. Вычислительная сложность построения;

4. Информативность - достаточность данных дескриптора для дальнейшей идентификации и сравнения точек.

Существует значительное число реализаций алгоритмов построения дескрипторов. Рассмотрим наиболее популярные подходы.

1.2 Существующие алгоритмы построения дескрипторов

Следует отметить разницу в требованиях к алгоритму построения дескрипторов в зависимости от предметной области применения. В случае использования дескрипторов для описания особых точек (features) необходима повышенная точность идентификации и инвариантность к преобразованиям, вычислительная сложность же не является самым важным критерием, т.к. число особых точек на изображении, как правило, на порядки меньше его размерности.

Напротив, в задачах плотного сопоставления изображений, возникающих в фотограмметрии и построении карт глубины, требуется вычисление дескриптора для каждого пикселя, что накладывает дополнительные ограничения на вычислительную сложность. При этом точностью построения и идентификации возможно в некоторой степени пренебречь, т.к. в данных задачах дескрипторы применяются для оценки общих тенденций преобразований между изображениями, что позволяет за счет их большого количества добиться хороших результатов усреднением и дополнительной валидацией.

SIFT, SURF, ORB. Все сосут, ибо тяжелые, хороши для фичей, плохи для dense. Отъебитесь от меня пожалуйста, спасибо.

2 Разработка алгоритма построения дескрипторов изображений

2.1 Алгоритм построения дескрипторов

В качестве основы для разрабатываемого метода используется алгоритм DAISY. В основе алгоритма лежит метод построения карт ориентации градиентов.

Картой ориентации градиента (Gradient Orientation Map) называют градиент изображения, построенный в определенном направлении. Результатом построения градиента является изображение с выделенными краями объектов, и в целом выраженными областями перепадов яркости.

Ориентация градиента в определенном направлении делает более выраженными перепады яркости изображения в этом направлении. Сказал как боженька.

Построенные карты градиентов подвергаются последовательному размытию с постепенным увеличением ядра гауссовского фильтра. Полученные карты называются свернутыми картами ориентации (Convolved Orientation Maps).

Таким образом, получается набор изображений, каждое из которых с увеличением уровня размытия представляет собой все более обобщенную информацию о градиенте яркости исходного изображения.

Описанные операции выполняются один раз при начале работы алгоритма, полученные COM далее используются для семплирования результатов.

Сэмплирование результатов для каждой точки производится по следующему принципу:

КАРТИНКА ТУТА

Для каждой указанной на схеме точки сэмплируются значения COM, начиная с наименьшей степени размытия и заканчивая максимальной.

Полученный вектор значений нормализуется.

Полный дескриптор получается путем конкатенации всех нормализованных векторов значений COM, начиная с центральной точки.

При необходимости может производиться нормализация полного дескриптора одним из описанных способов.

Each circle represents one histogram region, which is part of the descriptor vector. Each histogram represents the Gradient Orientations (GOs) within this region. The gradient is split into H discrete orientations, so each single histogram has H entries.

Algortihm uses Q rings around the central point, on which the COMs are sampled, each ring has T histograms.

Therefore, each descriptor has $D_s = (Q \cdot T + 1) \cdot H$ entries.

To compute GOs at a specified point (u, v), several oriented derivatives of image I are computed as following:

$$G_o^\sigma = G^\sigma * \left(\frac{dI}{do}\right)^+, \left(\frac{dI}{do}\right)^+ = max\left(\frac{dI}{do}, 0\right),$$

where G^σ is a Gaussian kernel with standard deviation σ and o is a derivative orientation.

The results G_o^σ are referred as Convolved Orientation Maps (COMs).

In the next step the vector $h_\sigma(u, v)$ is being built as following:

$$h_\sigma(u, v) = [G_1^\sigma(u, v), \ldots, G_H^\sigma(u, v)]^T$$

Vector $h(u, v)$ is then normalized to unit norm. It represents the values of all the GOs at a point (u, v) after convolution with a Gaussian kernel with a standard deviation σ.

The full DAISY descriptor for location (u_0, v_0) is defined as a simple concatenation of all the vectors $h_{\sigma i}(u, v)$ for all points beginning with the center:

$$D(u_0, v_0) = [h_{\sigma 1}(u_0, v_0), h_{\sigma 1}(I_1(u_0, v_0, R_1)), \ldots, h_{\sigma 1}(I_T(u_0, v_0, R_1),$$

$$h_{\sigma 2}(I_1(u_0, v_0, R_2)), \ldots, h_{\sigma 2}(I_T(u_0, v_0, R_2),$$

$$\ldots\ldots$$

$$h_{\sigma Q}(I_1(u_0, v_0, R_Q)), \ldots, h_{\sigma Q}(I_T(u_0, v_0, R_Q)],$$

where $I_j(u, v, R)$ is the location with distance R from (u, v) in the direction given by j when the directions are quantized into the T values.

2.2 Программная реализация

Алгоритм сшивки изображений для исследования был реализован на языке Python 3.7 с использованием библиотеки OpenCV 3.4.2.

Функция, осуществляющая процедуру совмещения, принимает на вход два изображения и тип используемого дескриптора. Используются дескрипторы SIFT, SURF, ORB, BRIEF. Заметим, что алгоритмы SIFT и SURF являются патентованными, и не присутствуют в стандартной библиотеке OpenCV из-за лицензионных ограничений. Данные алгоритмы доступны в пакете xfeatures2d при использовании версии библиотеки, включающей неофициальные алгоритмы, opencv-contrib. Использование данных алгоритмов разрешается только в некоммерческих целях [11].

Для вычисления дескрипторов производится преобразование исходных изображений в градации серого.

```
# convert the image to grayscale
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
```

Производится поиск наборов особых точек при помощи выбранного дескриптора.

```
# detect keypoints in the image
detector = cv2.FeatureDetector_create("SIFT")
kps = detector.detect(gray)

# extract features from the image
extractor = cv2.DescriptorExtractor_create("SIFT")
(kps, features) = extractor.compute(gray, kps)
```

Найденные наборы проверяются на совпадение дескрипторов методом knn. Полученные совпадения дополнпительно тестируются на предмет нахождения расстояния между точками в заданных пределах. Пары, прошедшие тест, используются для вычисления преобразования.

```
# compute the raw matches and initialize the list of actual
# matches
```

```
matcher = cv2.DescriptorMatcher_create("BruteForce")
rawMatches = matcher.knnMatch(featuresA, featuresB, 2)
matches = []

# loop over the raw matches
for m in rawMatches:
# ensure the distance is within a certain ratio of each
# other (i.e. Lowe's ratio test)
if len(m) == 2 and m[0].distance < m[1].distance * ratio:
matches.append((m[0].trainIdx, m[0].queryIdx))
```

После нахождения совпавших точек, по их наборам для каждого изображения вычисляется матрица проективного преобразования. Если совпавших опорных точек найдено меньше, чем четыре, проективное преобразование не может быть вычислено, и функция возвращает статус ошибки.

```
 # compute the homography between the two sets of points
(H, status) = cv2.findHomography(ptsA, ptsB, cv2.RANSAC,
reprojThresh)
```

Построенная матрица проективного преобразования применяется ко второму входному изображению, трансформируя его до совпадения положений опорных точек.

Исходное первое изображение и преобразованное второе записываются в одно результирующее. В зависимости от требуемых параметров, сложение может производиться с различными значениями альфа-канала, в том числе с выделением яркостью пересекающейся области либо одного из изображений в целях повышения наглядности.

Функция возвращает результирующее изображение и отчет о работе алгоритма, включающий время выполнения и построенную матрицу проективного преобразования.

Полный код функции представлен в приложении А.

3.2 Экспериментальное исследование работы алгоритма

Выбранные для исследования цифровые шумы реализованы в виде отдельных функций, принимающих на вход изображение, подлежащее зашумлению, и параметры шума. Исходный код функций зашумления представлен в приложении А.

В качестве оценочных параметров точности работы алгоритма были выбраны общее количество найденных особых точек, процент совпадений от общего числа и смещение преобразованного изображения относительно истинного положения в результате неточности построенной матрицы проективного преобразования. Количество найденных точек возвращается функцией сшивки после успешного завершения работы. Смещение относительно истинного положения вычисляется по координатам углов исходного и преобразованного изображений, оценочными параметрами смещения выступают медиана и максимум расстояний между соответствующими углами в пикселях.

Результаты экспериментов сохраняются в виде текстовых отчетов и (или) изображений, доступных для дальнейшего анализа и визуализации.

Для приведенных экспериментов вторым изображением, поступающим на вход алгоритма сшивки, выступала программно вырезанная прямоугольная часть центральной области первого изображения, равная 25 процентам его площади. Такой подход, в отличии от аналогичного практическому применению алгоритма использования частично пересекающихся фотографий, позволяет гарантировать нахождение максимального принципиально возможного количества совпадающих точек, а также упростить расчет смещения и визуализацию.

2.2.1 Сравнение работы алгоритмов при применении размытия

Размытие изображения, строго говоря, не является цифровым шумом, хотя и может быть достаточно распространенным дефектом фотографии. Тем не менее, размытие хорошо подходит для общего сравнения алгоритмов, так как является стационарным и легко воспроизводимым процессом понижения качества картинки.

В данном эксперименте входное изображение размывалось с итеративным увеличением размера ядра гауссовского фильтра m. Эксперимент прекращался, когда число найденных опорных точек становилось меньше четырех. Входные изображения приводились к размеру в 512 пикселей по горизонтали. На рисунках 1-2 представлены примеры результатов работы программы.

Рисунок 1 – Пример отображения программой совпадений особых точек на размытом изображении

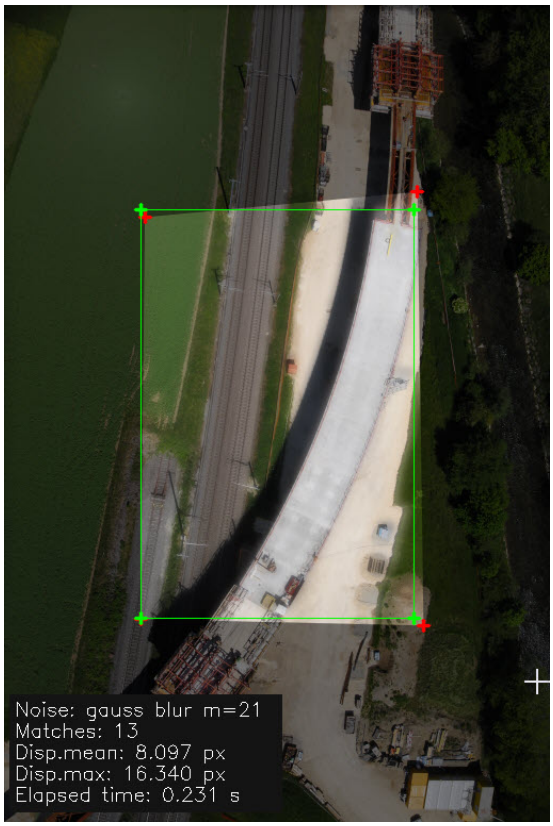

Рисунок 2 – Пример отображения программой результата совмещения

Таблица 1 – Сравнение числа найденных точек при увеличении уровня размытия

m	Алгоритмы		
	SIFT	SURF	ORB
-	496	996	1286
3	276	481	685
7	93	187	237
11	51	87	106
17	25	48	23
19	21	42	17
21	16	30	9
27	9	22	-
35	6	16	-
41	7	8	-
49	5	6	-
53	5	-	-

Прочерком обозначены значения, при которых алгоритм находит менее четырех совпадающих опорных точек.

Таблица 2 – Сравнение медианы смещения при увеличении уровня размытия

m	Алгоритмы		
	SIFT	SURF	ORB
-	0,032	0,000	0,130
3	0,082	0,085	0,163
7	0,146	0,170	2,670
11	0,270	0,303	1,953
17	0,917	1,100	2,852
19	0,405	1,476	15,543
21	2,243	2,893	336,972
27	5,564	1,012	-
35	13,711	5,951	-
41	44,210	10,701	-
49	84,046	22,116	-
53	120,100	-	-

В таблице 2 цветом выделены значения медианы смещения в более чем 10 пикселей, как выбранный критерий хорошо различимого визуально показателя недостаточной точности. Прочерком обозначены значения, при которых алгоритм находит менее четырех совпадающих опорных точек.

Результаты эксперимента отражены на рисунке 3.

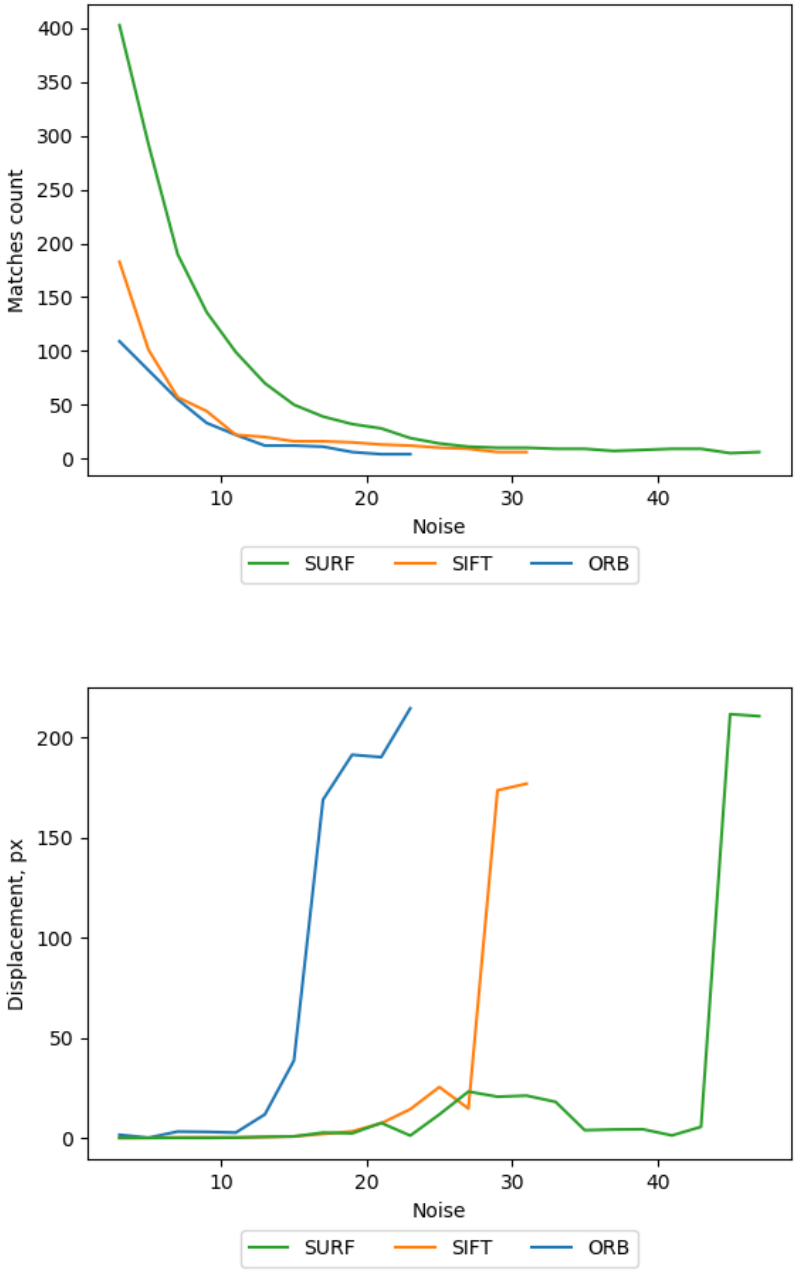

Рисунок 3 – Поведение числа совпадений точек и смещения при размытии изображения

2.2.2 Сравнение работы алгоритмов при применении случайных шумов

При использовании случайных шумов эксперимент производился несколько раз для каждого уровня зашумленности с усреднением результатов для устранения флуктуаций и получения более общей картины.

Для каждого уровня зашумленности значения усреднялись по 30 экспериментам. Варьируемыми параметрами шумов выступали стандартное отклонение σ для гауссовского шума и вероятность p для шума соль-и-перец. Эксперимент прекращался, когда число найденных опорных точек становилось меньше четырех. Входные изображения приводились к размеру в 512 пикселей по горизонтали.

Результаты экспериментов отражены на рисунках 4-5.

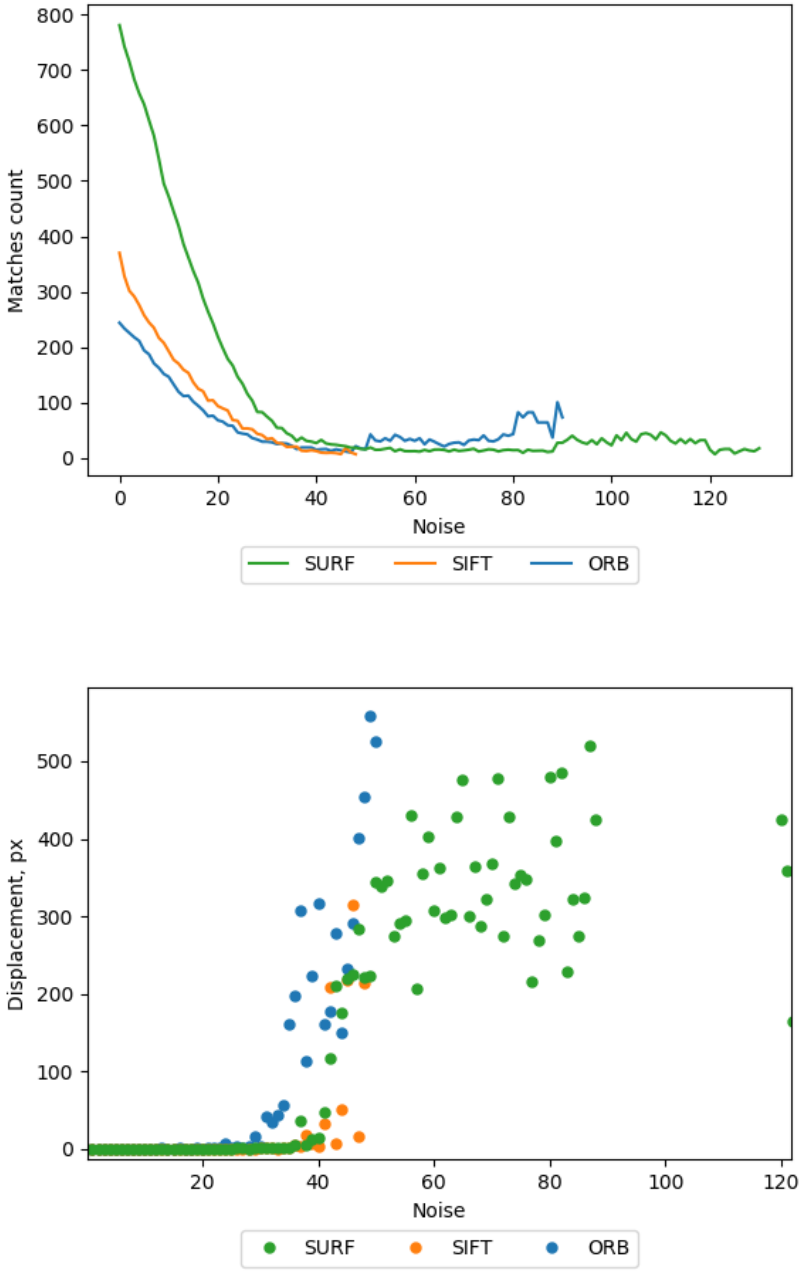

Рисунок 4 – Поведение числа совпадений точек и смещения при использовании гауссовского шума

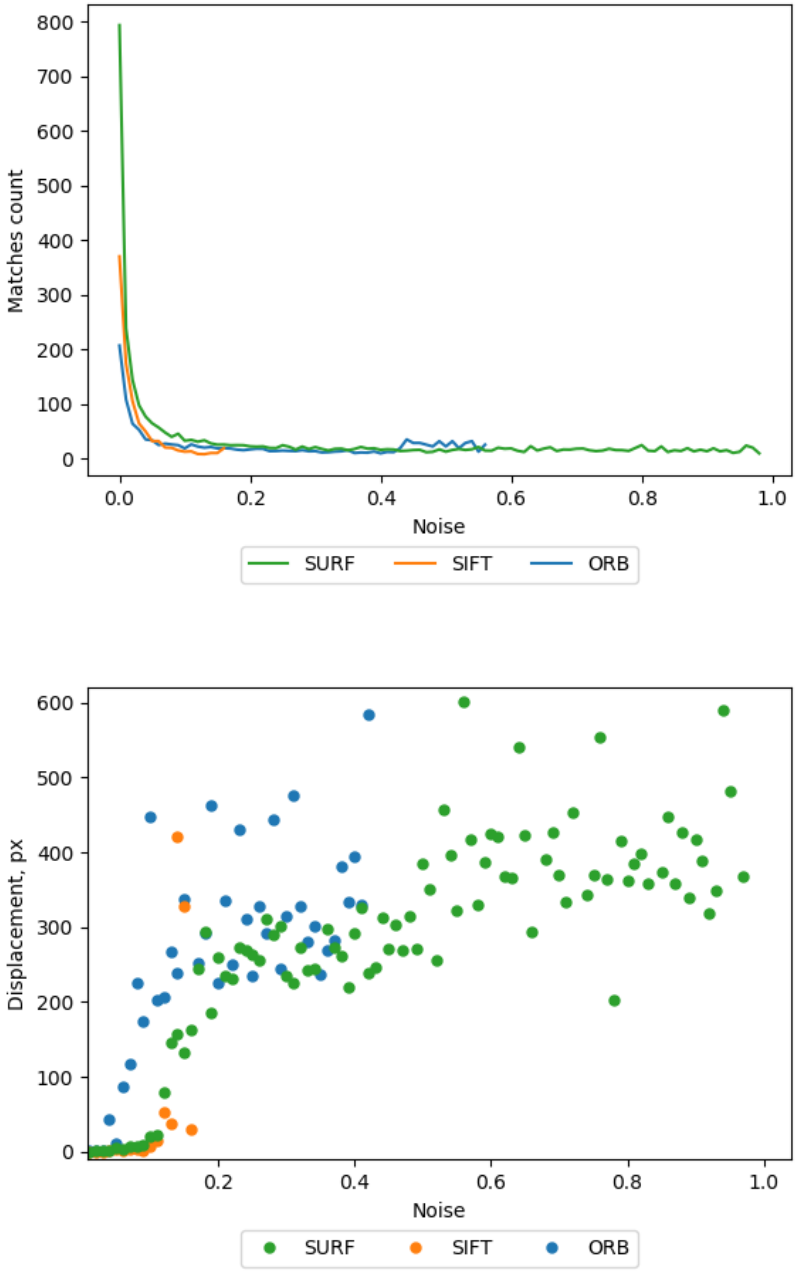

Рисунок 5 – Поведение числа совпадений точек и смещения при использовании шума соль-и-перец

3 Анализ полученных результатов

Проанализируем полученные результаты. Эксперименты показывают несколько интересных факторов в работе алгоритмов.

При размытии изображения дескриптор ORB раньше других рассмотренных начинает терять точность, несмотря на значительно большее общее количество найденных совпадений. Серьезное искажение результата достигается при размерах ядра Гауссова фильтра более 17 для тестового изображения размером 512 на 512 пикселей, с дальнейшим увеличением размера ядра ORB первым перестает находить достаточное число точек. Наилучшим образом при работе с размытым изображением показывает себя дескриптор SURF.

У всех рассмотренных алгоритмов наблюдается резкое увеличение искажения результата при переходе определенного порогового значения размытия.

На графике 6 можно наблюдать резкое падение качества при числе точек, меньшем чем 20. Это объясняется тем, что размытие может сильно ухудшать точность определения координат особых точек, даже при сохранении общей яркостной картины достаточном для построения дескриптора. Соответственно, при малых количествах точек, ошибки в отдельных координатах сильно влияют на построенную матрицу преобразования.

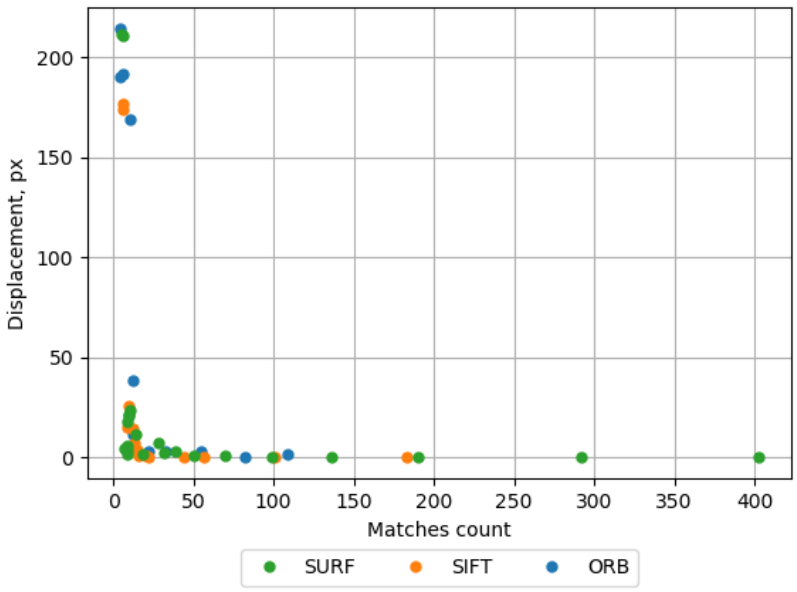

Рисунок 6 – Изменение смещения с ростом числа точек при использовании размытия

При применении случайных шумов можно наблюдать более интересную картину. При использовании дескриптора SIFT точность падает равномерно, алгоритм перестает работать на тестовых изображениях при значении σ гауссовского шума более 50 и при вероятности шума соль-перец более 0.15. При этом, остальные дескрипторы продолжают работать при значительно большем зашумлении, но стремительно теряют в точности, приводя к абсолютно непригодным для практического применения результатам. На рисунке 7 приведен пример результата, выданного алгоритмом SURF на значениях, при которых SIFT перестает работать.

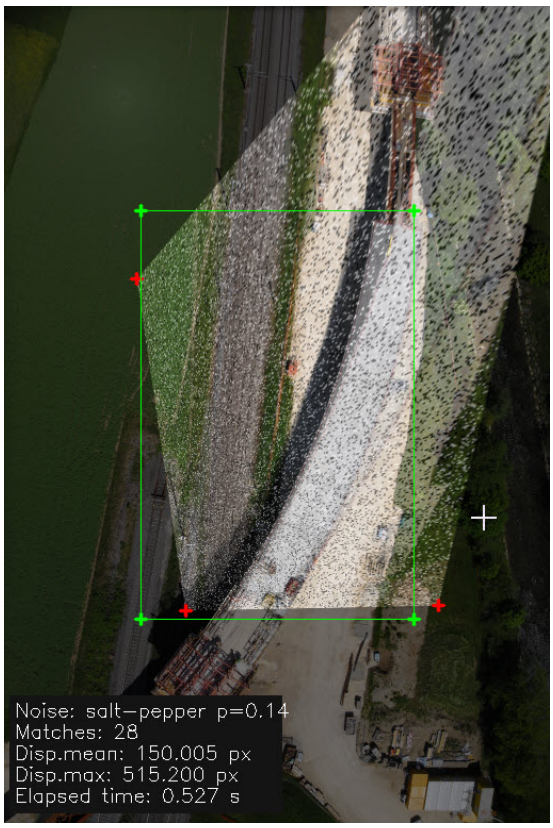

Рисунок 7 – Пример искажения при использовании алгоритма SURF

Принимая во внимание практическую непригодность изображений с подобными уровнями шумов и искажений, можно отбросить диапазоны значений за пределами стабильной работы алгоритма SIFT. На графиках 8-9 можно наблюдать этот диапазон.

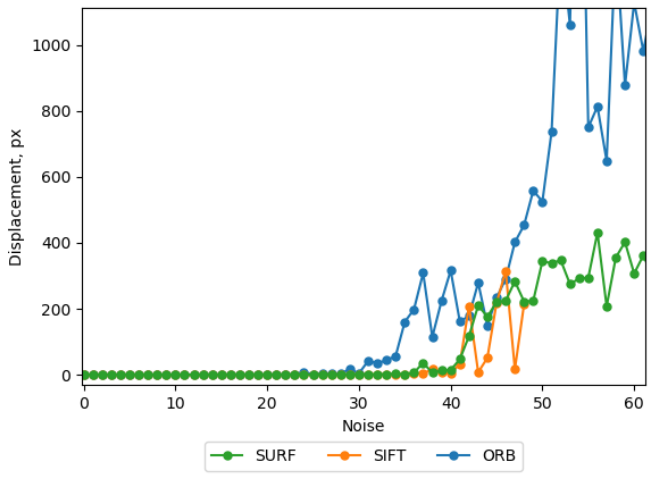

Рисунок 8 – Увеличенный фрагмент графика смещения при использовании гауссовского шума

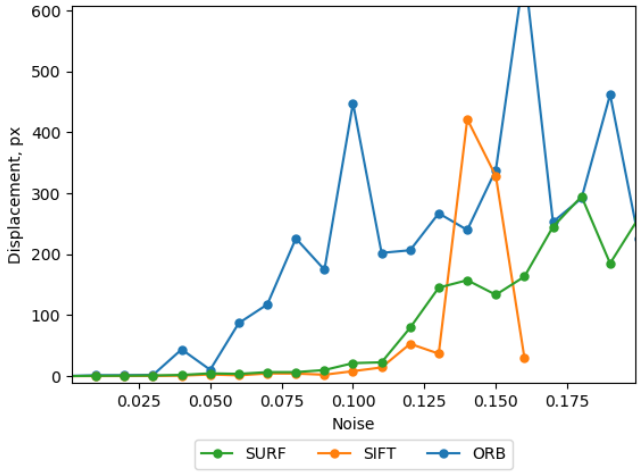

Рисунок 9 – Увеличенный фрагмент графика смещения при использовании шума соль-перец

Как можно видеть, все три алгоритма сохраняют достаточную точность при практически достижимых уровнях гауссовского шума. Однако, ORB наименее устойчив к шуму соль-и-перец, что можно видеть по резкому пику на графике при вероятности 0.03, сохраняющемуся даже при усреднении по набору экспериментов.

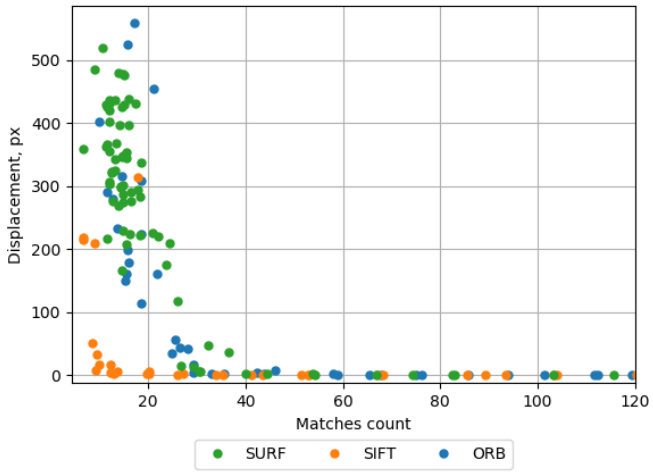

Рисунок 10 – Изменение смещения с ростом числа точек при использовании гауссовского шума

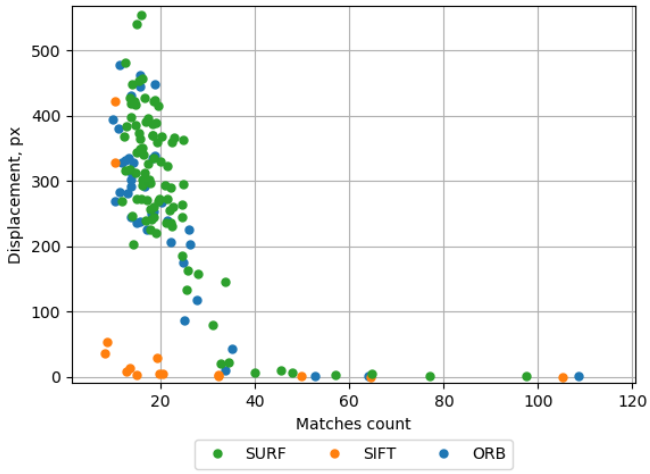

Рисунок 11 – Изменение смещения с ростом числа точек при использовании шума соль-и-перец

Зависимость смещения от количества найденных точек, как видно на графиках 10-11, выражена не так ярко, как при размытии, и можно наблюдать более плавный переход.

Итак, наивысшая точность при применении случайных шумов достигается при использовании алгоритма SIFT. Наихудший результат показывает ORB. Стоит заметить, что и при использовании импульсного шума соль-перец, и при использовании аддитивного гауссовского шума наибольшие количества особых точек находит алгоритм SURF.

Обобщая полученные результаты, можно сказать, что все рассмотренные алгоритмы проявили себя достаточно хорошо при тех уровнях зашумления, которые потенциально могут встретиться на практике. Значения, при которых наблюдается серьезное смещение относительно требуемого, в любом случае сделали бы на практике такую фотографию непригодной к использованию.

При размытии изображения наилучшим образом ведет себя алгоритм SURF, что позволяет рекомендовать его к использованию, если исходные фотографии сделаны с заметным нарушением фокусировки. При случайных шумах, как аддитивном, так и импульсном, наилучшим образом рабо-

тает SIFT, ценой большего времени выполнения и вычислительной сложности. Хуже всего в проведенных экспериментах отработал алгоритм ORB, в особенности, при использовании шума соль-перец.

ЗАКЛЮЧЕНИЕ

В ходе данной работы была рассмотрена задача сшивки изображения и методы ее решения. Проведен обзор математической постановки задачи. Рассмотрен алгоритм нахождения пространственного преобразования по опорным точкам. Был дан обзор нескольких методов поиска опорных точек, реализованных в открытой библиотеке OpenCV.

Рассмотрены цифровые шумы в задачах обработки изображений и их особенности, разобраны популярные математические модели искусственно генерируемых шумов, используемых для тестирования алгоритмов и оборудования.

Реализована компьютерная программа, осуществляющая совмещение изображений по особым точкам с возможностью выбора типа дескриптора. Также, реализована программная среда для экспериментов, позволяющая использовать изображения с генерацией различных типов и уровней зашумления, формировать отчеты о работе алгоритмов и анализировать графическое представление результатов.

В ходе серии экспериментов было выяснено, что представленные алгоритмы являются достаточно устойчивыми к зашумлению, и работают с достаточной точностью при уровнях шума, допускающих сохранение практической ценности таких изображений. Однако, можно выделить алгоритмы, лучше или хуже ведущие себя при определенных типах шумов. Результаты экспериментов показывают, что наилучшим образом все представленные алгоритмы справились с размытием, наихудшим - с импульсным шумом. Были сделаны выводы о применимости отдельных алгоритмов при конкретных практически возможных дефектах фотографий.

Дальнейшим развитием работы видится оценка при использовании на зашумленных изображениях восстанавливающих фильтров, либо восстановления шума при помощи моделей глубокого обучения.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 Bres, S. Detection of interest points for image indexation [Текст] / S. Bres, J. M. Jolion // International Conference on Advances in Visual Information Systems. – Springer, Berlin, Heidelberg, 1999. – P. 427-435.

2 Lowe, D. G. Distinctive image features from scale-invariant keypoints [Текст] / D. G. Lowe // International journal of computer vision. – 2004. – Vol. 60. – I. 2. – P. 91-110.

3 Rublee, E. ORB: an efficient alternative to SIFT or SURF [Текст] / E. Rublee, D. Brando, J. Joestar // ICCV '11 Proceedings of the 2011 International Conference on Computer Vision. – IEEE Computer Society Washington, DC, USA, 2011. – P. 2564-2571.

4 Проективное преобразование [Электронный ресурс] // Википедия : свободная энцикл. – Электрон. дан. – 2019. – URL: https://ru.wikipedia.org/?oldid=99107559 (дата обращения: 15.04.2019).

5 Аффинное преобразование [Электронный ресурс] // Википедия : свободная энцикл. – Электрон. дан. – 2019. – URL: https://ru.wikipedia.org/?oldid=93707864 (дата обращения: 15.04.2019).

6 Karami, E. Image Matching Using SIFT, SURF, BRIEF and ORB: Performance Comparison for Distorted Images [Электронный ресурс] / E. Karami, S. Prasad, M. Shehata // arXiv preprint. – 2015. – Электрон. дан. – URL: https://arxiv.org/ftp/arxiv/papers/1710/1710.02726.pdf (дата обращения: 25.05.2019).

7 Kong, H. A generalized Laplacian of Gaussian filter for blob detection and its applications [Текст] / H. Kong, H. C. Akakin, S. E. Sarma // IEEE

transactions on cybernetics. – IEEE Computer Society Washington, DC, USA, 2013. – V. 43. – I. 6. – P. 1719-1733.

8 Moeslund, T. B. BLOB Analysis: An introduction to video and image processing [Текст] / T. B. Moeslund. – Springer, London, 2012. – 227 p. – p. 5-20.

9 Цифровой шум изображения [Электронный ресурс] // Википедия : свободная энцикл. – Электрон. дан. – 2019. – URL: https://ru.wikipedia.org/?oldid=95235149 (дата обращения: 15.04.2019).

10 Deng, G. An adaptive Gaussian filter for noise reduction and edge detection [Текст] / G. Deng, L. W. Cahill // IEEE Conference Record. – IEEE Computer Society Washington, DC, USA, 1993. – P. 1615-1619. b

11 OpenCV Documentation [Электронный ресурс] : официальная документация библиотеки OpenCV. / Intel Corporation, Willow Garage Inc., Itseez Ltd. – Электрон. дан. – 2019. – URL: https://docs.opencv.org (дата обращения: 25.04.2019).

ПРИЛОЖЕНИЕ А

Код программы