ЛАБОРАТОРНАЯ РАБОТА 3. СИНТЕЗ ПАНОРАМНЫХ ИЗОБРАЖЕНИЙ

3.1 Теоретическая часть

Формированием панорамных изображений стали заниматься практически сразу после появления технологии фотографии. Первые панорамные фотоаппараты были созданы уже в середине XIX века. С наступлением эры цифровой видеотехники, решение задачи получения панорамных снимков стало еще более востребованным.

Применительно к системам видеонаблюдения говорят о задаче формирования не фото, а видеопанорамы: единого видеоизображения, получаемого от нескольких источников (камер), или от одной поворотной камеры. В случае синтеза видеопанорамы с помощью нескольких камер, при достаточной частоте обновления изображения (25 кадров в секунду и выше) говорят о видеопанораме в реальном времени. При применении поворотной камеры панорама не будет отображать наблюдаемую сцену в реальном времени.

Видеопанорамное изображение в системах наблюдения целесообразно осуществлять, когда размер объекта, или площадь зоны наблюдения превышает поле зрения камеры. Представление видеоданных в виде отдельных фрагментов, полученных от несвязанных камер, затрудняет восприятие оператором происходящего, снижает быстроту принятия решений, требует от него большего внимания. Использование в данном случае широкоугольного объектива приведет к геометрическим искажениям в кадре при компенсации которых произойдет потеря разрешения.

Задача синтеза панорамы (предположим, из двух фрагментов) может быть решена путем корреляционного совмещения одного изображения с другим. Причем, совмещаемое изображение должно иметь степени свободы по масштабу и углам поворота. Последовательно трансформируя один из фрагментов (изменяя масштаб и углы поворота с некотором шагом) и совмещая его (изменяя смещение одного снимка относительно второго) с другим находят максимум отклика корреляционной функции в некоторой точке r, тем самым определяя оптимальные параметры для сшивки изображений.

$$r = \frac{\sum_{x,y}(L_1(x,y) \cdot L_2(x,y))}{\sqrt{\sum_{x,y}(L_1(x,y))^2 \cdot \sum_{x,y}(L_2(x,y))^2}}$$

где L_1 – опорное изображение, L_2 – совмещаемое изображение.

Данный метод, однако, мало применим на практике. Во-первых, он очень чувствителен к шумам и неодинаковой яркости фрагментов. Во-вторых, даже при фрагментах небольшого размера он требует огромного количества вычислительных ресурсов.

Общепринятый алгоритм для синтеза панорам состоит из следующих шагов.

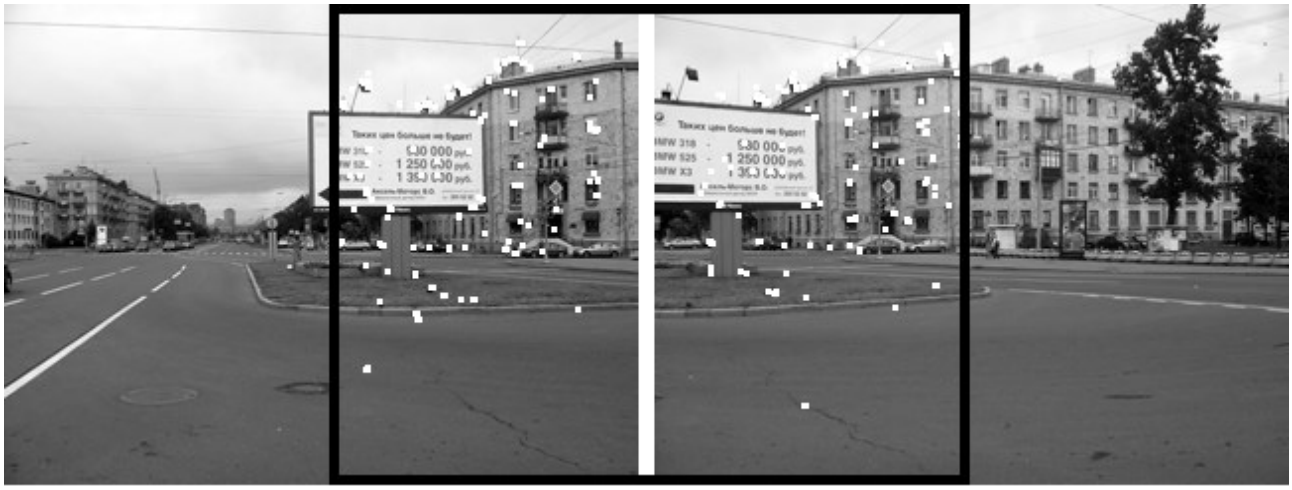

Рис. 3.1. Характерные точки сшиваемых изображений
(зона перекрытия показана рамкой)

1. Нахождение характерных точек (в англоязычной литературе используют термин «ключевая» точка «key point») на исходных изображениях в зоне перекрытия. Под характерной точкой понимают некоторый малый фрагмент изображения, в котором как значение яркостного градиента, так и производная (скорость изменения) градиента по направлению высоки. Как правило, характерных точками являются различные углы на изображениях (рис. 3.1).

2. Нахождение в зоне перекрытия одних и тех же особенностей на различных снимках (рисунок 3.2). То есть определение соответствующих друг другу пар характерных точек

   Согласованные пары точек — это основа для синтеза панорамы. С помощью установленных пар на последующих этапах решают задачи калибровки, трансформации и объединения снимков. В случае синтеза панорамы из нескольких кадров (частей) при неизвестном заранее расположении каждого снимка, на данном этапе проводят процедуру регистрации – идентификации местоположения отдельных изображений на общей панораме и установление взаимных соответствий характерных точек.
3. В современных программных пакетах следующим шагом алгоритма сшивки является калибровка изображений. Эта процедура направлена на минимизацию искажений объектива, оптических дефектов, различий экспозиции. С помощью информации о согласованных парах характерных точек минимизируют влияние дисторсии (геометрических искажений) объектива на точность сшивки панорамы.
4. Ключевым этапом является процедура идентификации параметров уравнений трансформации изображений и последующее преобразование фрагментов с объединением в единую панораму. Данный этап требует задания вида трансформации, которое определяется типом создаваемой панорамы. Например, при отсутствии, или несущественности перспективных искажений у

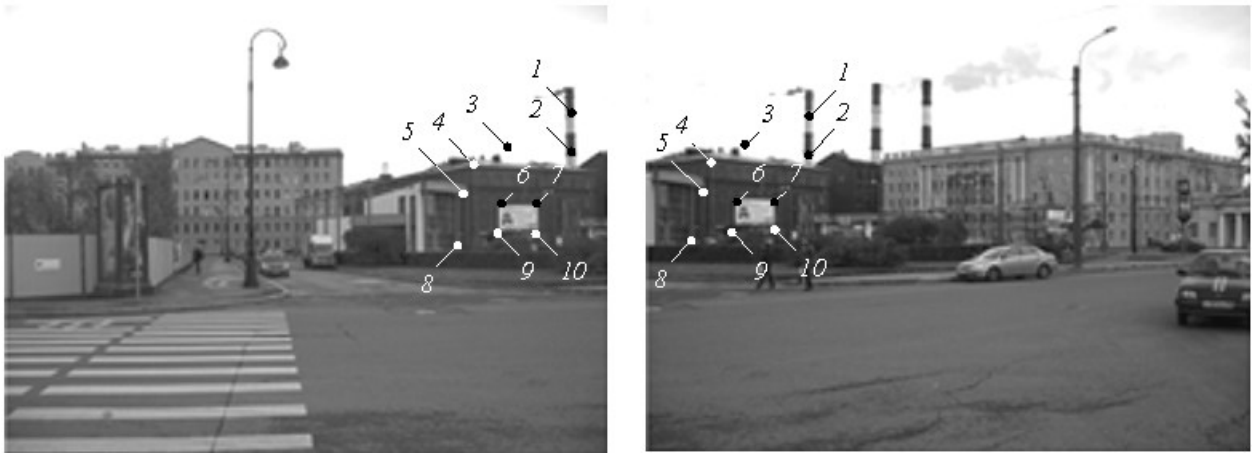

Рис. 3.2. Пары согласованных точек на сшиваемых изображениях

фрагментов имеет смысл использовать аффинное преобразование. Примером такого изображения может быть «сшивка» панорамы из отсканированных частей единого документа (географической карты, картины, и т.п.). В общем случае применяют перспективное преобразование, учитывающее все возможные искажения снимков.

5. Заключительным этапом является блэндинг. Это комплексная процедура, направленная на повышение визуального качества панорамы, включающая выравнивание яркости и цветовой палитры фрагментов, маскирование «швов», удаление «призраков» (движущихся объектов). Кроме того, в блэндинг включают и процедуру проецирования панорамы на заданную поверхность – сферическую, цилиндрическую, эквидистантную и пр.

Простейшим примером, демонстрирующим процесс синтеза единого изображения, является построение панорамы из двух фрагментов. Для нахождения характерных точек существует множество алгоритмов – «детекторов». Самыми распространенным являются SIFT и SURF. Базовым детектором, лежащем в основе многих других, более совершенных является уголковый детектор Харриса. Угол на изображении интересен тем, что по нему легко локализовать соответствующую особенность на различных снимках. В угловой точке градиент яркости меняет свое направление в предельном случае на 90°.

Чтобы отличить угловую точку от наклонной линии, которая тоже имеет производные по обоим направлениям, в автоматическом экстракторе углов в каждой точке вычисляют собственные значения λ_1 и λ_2 матрицы гессиан

$$\mathbf{H} = \begin{bmatrix} \sum_{x,y}\left(\frac{\partial^2 L_{(x,y)}}{\partial x^2}\right) & \sum_{x,y}\left(\frac{\partial^2 L_{(x,y)}}{\partial x \partial y}\right) \\ \sum_{x,y}\left(\frac{\partial^2 L_{(x,y)}}{\partial x \partial y}\right) & \sum_{x,y}\left(\frac{\partial^2 L_{(x,y)}}{\partial y^2}\right) \end{bmatrix},$$

где суммы берут по заранее заданному размеру окна (например, блоку 5x5 пикселей).

Затем вычисляют коэффициент обусловленности $cond(\mathbf{H})=\lambda_{max}/\lambda_{min}$. Когда коэффициент обусловленности матрицы **H** близок к единице, причем оба собственных значения большие, фрагмент считают угловым. Если фон однородный - оба собственных значения близки к нулю. Для изображения наклонной линии одно из собственных значений будет нулевым.

Реализация этого алгоритма сопряжена с существенными вычислительными затратами. Чтобы сократить объем вычислений используют двухшаговый алгоритм извлечения угловых точек. На первом шаге применяют «функцию углового отклика» $CR=min|dL/dx,\ dL/dy|$, представляющую собой абсолютное значение меньшей составляющей вектора градиента. Матрицу **H** вычисляют только для потенциальных характерных точек, где *CR* превышает заранее заданный порог. Все предполагаемые угловые фрагменты, найденные на первом шаге, далее оценивают через вычисление собственных значений. Первый шаг играет главную роль в сокращении объема вычислений, требуемых на втором шаге. Степень сокращения зависит от содержания изображений и значения порога, заданного на первом шаге. Как правило, второй шаг применяют менее чем к пяти процентам площади изображения. В результате реализации этих процедур автоматически находят некоторое число характерных точек в области перекрытия сшиваемых изображений, достаточное для вычисления параметров преобразования.

После выделения характерных точек на двух соседних изображениях формируют пары согласованных характерных точек, представляющих на изображениях одни и те же области. Для этого производят корреляционное сопоставление блоков с центрами в найденных характерных точках. Каждый блок первого изображения сравнивают со всеми блоками второго изображения в зоне перекрытия. Мера подобия двух блоков с центрами в характерных точках $p_n(x_p,y_p)$ и $q_m(x_q,y_q)$

$$SM(p_n,q_m)=\frac{D_x(p_n,q_m)+D_y(p_n,q_m)}{2},$$

где *m, n* – число пикселей в сравниваемых блоках по вертикали и горизонтали, соответственно,

$$D_x(p_n,q_m)=1-\frac{\sum_{i=1}^{m}\sum_{j=1}^{n}\left|\frac{dL_p(x_{p,j},y_{p,i})}{dx}-\frac{dL_q(x_{q,j},y_{q,i})}{dx}\right|}{\sum_{i=1}^{m}\sum_{j=1}^{n}\frac{dL_p(x_{p,j},y_{p,i})}{dx}+\sum_{i=1}^{m}\sum_{j=1}^{n}\frac{dL_q(x_{q,j},y_{q,i})}{dx}},$$

$$D_y(p_n,q_m)=1-\frac{\sum_{i=1}^{m}\sum_{j=1}^{n}\left|\frac{dL_p(x_{p,j},y_{p,i})}{dy}-\frac{dL_q(x_{q,j},y_{q,i})}{dy}\right|}{\sum_{i=1}^{m}\sum_{j=1}^{n}\frac{dL_p(x_{p,j},y_{p,i})}{dy}+\sum_{i=1}^{m}\sum_{j=1}^{n}\frac{dL_q(x_{q,j},y_{q,i})}{dy}}.$$

Значение $SM(p_n, q_m)$ изменяется в диапазоне от нуля до единицы, причем $SM(p_n, q_m)=1$ для идентичных блоков. В результате каждой точке из множества **P** сопоставляют точку из множества **Q** с мерой подобия $SM(p_n, q_m)$ и формируют множество согласованных пар **PQ**.

Шумы в сигнале и различие конфигураций блоков из-за недостаточного перекрытия изображений приводит к образованию ложно согласованных пар во множестве **PQ**.

Существуют различные подходы для фильтрации ошибочно согласованных пар. Одним из самых простых и эффективных является проверка на соотношение значений меры подобия. Для каждой ключевой точки p_k на одном изображении определяют меры подобия со всеми точками на втором сшиваемом фрагменте. Далее берут два максимальных значения $SM(p_k, q_{m1})$ и $SM(p_k, q_{m2})$ и получают их отношение. Предполагается, что если для некоторой ключевой точки p_k одного фрагмента найдено истинное соответствие на втором, то мера подобия для данной точки будет значительно большей, чем для второй по близости характерной точки. Напротив, если характерная точка не имеет аналога на сшиваемом фрагменте, то ее значения меры подобия с двумя наиболее похожими точками будут приблизительно равны. Таким образом, пара точек считается согласованной, если выполняется условие:

$$\frac{SM(p_k,q_{m1})}{SM(p_k,q_{m2})}>t$$

где значение t обычно выбирают равным 2-3.

При синтезе панорамы из двух изображений, полученных от соседних камер с общим оптическим центром, используют модель перспективного преобразования, учитывающая все возможные искажения растров (масштаб, поворот, смещение, перспективные искажения). Уравнение трансформации:

$$\begin{pmatrix} x_{ip} \\ y_{ip} \\ 1 \end{pmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \cdot \begin{pmatrix} x_{iq} \\ y_{iq} \\ 1 \end{pmatrix},$$

где x_{ip}, y_{ip} и x_{iq}, y_{iq} – координаты соответствующих пикселей в трансформируемом и в опорном изображениях, соответственно; $h_{11},\ldots, h_{33}$ – искомые параметры.

Для оценки параметров уравнения трансформации используют аппарат регрессионного анализа. На основании координат пар согласованных характерных точек ($\{(x_{ip}, y_{ip}), (x_{iq}, y_{iq})\}, i=1..N, N>5$) и выбранной модели уравнения составляют матрицу плана эксперимента

$$\mathbf{A} = \begin{bmatrix} x_{1,p} & y_{1,p} & 1 & 0 & 0 & 0 & -x_{1,p}\cdot x_{1,q} & -y_{1,p}\cdot x_{1,q} & -x_{1,q} \\ 0 & 0 & 0 & x_{1,p} & y_{1,p} & 1 & -x_{1,p}\cdot y_{1,q} & -y_{1,p}\cdot y_{1,q} & -y_{1,q} \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ x_{N,p} & y_{N,p} & 1 & 0 & 0 & 0 & -x_{N,p}\cdot x_{N,q} & -x_{N,p}\cdot x_{N,q} & -x_{N,q} \\ 0 & 0 & 0 & x_{N,p} & y_{N,p} & 1 & -x_{N,p}\cdot y_{N,q} & -y_{N,p}\cdot y_{N,q} & -y_{N,q} \end{bmatrix}.$$

Вектор неизвестных параметров $\mathbf{h}=(h_{11},\ldots,h_{33})^{\mathrm{T}}$ определяют методом наименьших квадратов путем решения системы уравнений:

$$\mathbf{A}\cdot\mathbf{h}=\mathbf{b},$$

где **b** - вектор откликов (значений целевой функции).

Так как матрица **A** не квадратная, она не имеет обратной матрицы. Поэтому используют псевдообратную матрицу $\mathbf{A}^{+}=(\mathbf{A}^{\mathrm{T}}\cdot\mathbf{A})^{-1}\cdot\mathbf{A}^{\mathrm{T}}$. Тогда

$$\mathbf{h}=\left(\mathbf{A}^{\mathrm{T}}\cdot\mathbf{A}\right)^{-1}\cdot\mathbf{A}^{\mathrm{T}}\cdot\mathbf{b}=\mathbf{A}^{+}\cdot\mathbf{b}. \qquad (3.1)$$

Для задачи наименьших квадратов данное выражение называют нормальной системой относительно **h**.

Расчет псевдообратной матрицы - трудоемкая процедура, требующая значительного объема вычислений. Существует большое количество численных методов решения данной задачи. Популярны подходы, основанные

на ортогональных разложениях матрицы **A**. В задаче построения панорамных изображений часто используют сингулярное разложение.

С помощью какого-либо численного метода (например, *QR*-алгоритма, который реализован в большинстве математических пакетов) матрицу **A** представляют в виде произведения:

$$\mathbf{A} = \mathbf{U} \cdot \mathbf{\Sigma} \cdot \mathbf{V}^{\mathrm{T}},$$

где **Σ** - диагональная матрица.

Элементы на диагонали матрицы **Σ** являются сингулярными числами матрицы **A**, причем расположены по убыванию. **U**, $\mathbf{V}^{\mathrm{T}}$ - матрицы, столбцы которых содержат левые и правые сингулярные векторы для соответствующих им значений собственных чисел на диагонали матрицы **Σ**. Причем

$$\mathbf{V} \cdot \mathbf{V}^{\mathrm{T}} = \mathbf{I}, \ \mathbf{U} \cdot \mathbf{U}^{\mathrm{T}} = \mathbf{I},$$

где **I** - единичная матрица, так как **U**, **V –** ортогональные матрицы**.**

Столбцы матрицы **U** - это собственные векторы $\mathbf{A}^{\mathbf{T}} \cdot \mathbf{A}$, а столбцы матрицы $\mathbf{V}^{\mathrm{T}}$ – собственные векторы $\mathbf{A} \cdot \mathbf{A}^{\mathbf{T}}$.

Имея сингулярное разложение, можно представить решение задачи наименьших квадратов в явном виде, не прибегая к трудоемкому обращению матриц:

$$\mathbf{A}^{+} = \left(\mathbf{V} \cdot \mathbf{\Sigma} \cdot \mathbf{U}^{\mathrm{T}} \cdot \mathbf{U} \cdot \mathbf{\Sigma} \cdot \mathbf{V}^{\mathrm{T}}\right)^{-1} \cdot \mathbf{V} \cdot \mathbf{\Sigma} \cdot \mathbf{U}^{\mathrm{T}} =$$

$$\left(\mathbf{\Sigma} \cdot \mathbf{V}^{\mathrm{T}}\right)^{-1} \cdot \left(\mathbf{V} \cdot \mathbf{\Sigma}\right)^{-1} \cdot \left(\mathbf{V} \cdot \mathbf{\Sigma}\right) \cdot \mathbf{U}^{\mathrm{T}} = \mathbf{V} \cdot \mathbf{\Sigma}^{-1} \cdot \mathbf{U}^{\mathrm{T}}.$$

Результат обращения диагональной матрицы

$$\mathbf{\Sigma}^{-1} = \mathbf{diag}\left(\frac{1}{\sqrt{\lambda_1}}, \ldots, \frac{1}{\sqrt{\lambda_n}}\right),$$

где $\lambda_1, \ldots, \lambda_n$ - собственные числа. Таким образом, вектор решения МНК

$$\mathbf{x} = \mathbf{V} \cdot \mathbf{\Sigma}^{-1} \cdot \mathbf{U}^{\mathrm{T}} \cdot \mathbf{y}.$$

Определив матрицу правых собственных векторов $\mathbf{V}^{\mathrm{T}}$ и выбрав ее последний столбец, которому соответствует минимальное собственное значение **A**, расположенное на последней строке диагональной матрицы **Σ**, получают вектор параметров, дающий минимальную невязку для выражения (3.1).

Таким образом, параметры уравнения трансформации определяют из сингулярного разложения матрицы **A**, как значения последнего (девятого) столбца матрицы $\mathbf{V}^{\mathrm{T}}$. Для оценки параметров уравнения трансформации используют алгоритм устойчивой оценки *RANSAC*:

1. Случайным образом из множества согласованных точек выбираются пять пар. По ним определяют параметры уравнения трансформации (создают текущую гипотезу уравнения). Для каждой пары множества согласованных точек вычисляют ошибку трансформации

$$E_i = \begin{pmatrix} x_{ip} \\ y_{ip} \\ 1 \end{pmatrix} - \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \cdot \begin{pmatrix} x_{iq} \\ y_{iq} \\ 1 \end{pmatrix}.$$

2. Подсчитывают число пар характерных точек (так называемых «не выбросов», в англоязычной литературе «*inliers*»), для которых ошибка трансформации E при текущей гипотезе не превышает некоторого порога (обычно 2-3 пикселя), остальные пары отбрасывают.

3. Шаги 1, 2 повторяют заранее определенное число итераций. На каждой итерации получают и проверяют новую гипотезу с помощью очередных случайно выбранных пяти пар согласованных точек. В результате запоминают параметры гипотезы, для которой получилось больше всего «не выбросов».

4. По пяти парам согласованных точек, с помощью которых была получена гипотеза, запомненная на шаге 3, вновь определяют параметры уравнения трансформации.

5. Исключают выбросы (то есть пары точек, для которых ошибка превышает порог) и пересчитывают параметры уравнения трансформации, но уже с учетом всех пар «не-выбросов».

По найденным параметрам уравнения выполняют трансформации изображений и «сшивку» панорамы. Пример снимков-фрагментов и созданного единого изображения приведен на рисунке 3.3

Рис. 3.3. Пример созданной панорамы (нижний ряд) из двух фрагментов (верхний ряд)

3.2. Задание на моделирование

Разработать программу, реализующую синтез панорамы из двух фрагментов с помощью библиотеки OpenCV.

Необходимо подобрать два фрагмента (зона перекрытия изображений должна занимать не менее 25% их площади, желателен поворот или перспективное преобразование одного изображения относительно другого) осуществить синтез панорамы.

При работе программы должны быть визуализированы основные этапы:

- изображения с найденными ключевыми точками;
- результат процедуры поиска согласованных пар;
- результат трансформации изображения (один из снимков является опорным и остается без изменений, трансформируют только второй фрагмент);
- результат построения панорамы.

3.3. Практическая часть

Для сшивки панорам в OpenCV существует специальный модуль. В последних версиях этот процесс полностью автоматизирован и вынесен на высокий уровень – в класс *cv::Stitcher*. Тем не менее, для понимания процесса создания панорамного изображения полезно будет обратиться к примеру из ранних версий OpenCV. Нижеприведенный пример не содержит таких процедур, как калибровка снимков и блэндинг, что сказывается на качестве итогового изображения, но зато позволяет наглядно продемонстрировать основные этапы алгоритма.

В начале с помощью детектора характерных точек (в примере использован ORB, но в OpenCV существует множество других вариантов) найдем *keypoints*, вычислим их дескрипторы и покажем результат (рис. 3.4)

```
#include "stdafx.h"
#include "opencv2/core/core.hpp"
#include "opencv2/features2d/features2d.hpp"
#include "opencv2/highgui/highgui.hpp"
#include "opencv2/calib3d/calib3d.hpp"
#include "opencv2/imgproc/imgproc.hpp"
#include <iostream>

using namespace cv;

int main()
{
//Загрузка изображений с диска
 Mat img1 = imread("C:/../mnt1.jpg");
 Mat img2 = imread("C:/../mnt2.jpg");

//Инициализация детектора, подготовка контейнеров для характерных
точек и их дескрипторов
 Ptr<ORB> detector = ORB::create();
 std::vector<KeyPoint> keypoints1, keypoints2;
 Mat descriptor1, descriptor2;

//Поиск характерных точек на сшиваемых изображениях и вычисление
их дескрипторов
 detector->detect(img1, keypoints1);
 detector->detect(img2, keypoints2);
 detector->compute(img1, keypoints1, descriptor1);
```

```
    detector->compute(img2, keypoints2, descriptor2);

   //Отрисовка найденных характерных точек
    Mat keypoints1draw, keypoints2draw;
    drawKeypoints(img1, keypoints1, keypoints1draw);
    drawKeypoints(img2, keypoints2, keypoints2draw);

    imshow("keypoints1draw", keypoints1draw);
    imshow("keypoints2draw", keypoints2draw);

    waitKey();
    return 0;
   }
```

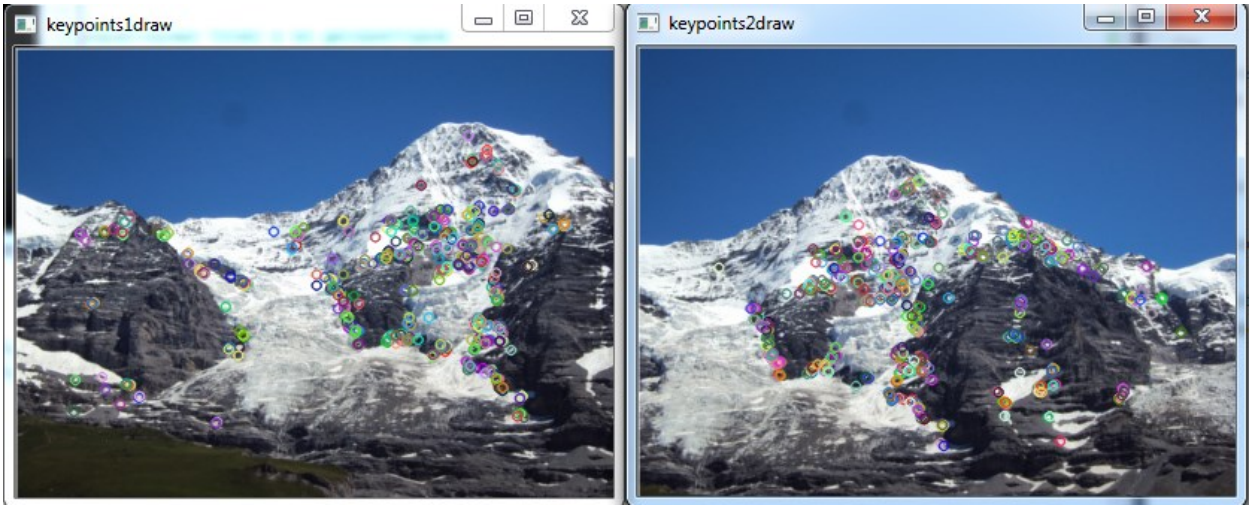

Рис. 3.4. Найденные характерные точки на изображениях

Дополним программу предыдущего примера сопоставлением дескрипторов. Для полного понимания приведенного кода, желательно самостоятельно разобраться с помощью документации, что представляют собой классы *DMatch* и *KeyPoint*

```
    //Сопоставление дескрипторов точек с помощью BFMatcher
    BFMatcher matcher(NORM_HAMMING);
    std::vector<std::vector<DMatch>> matches;
    matcher.knnMatch(descriptor1, descriptor2, matches, 2);

    std::vector<KeyPoint> matched1, matched2;
    std::vector<DMatch> good_matches;

    float match_ratio = 0.5f;

    for (size_t i = 0; i < matches.size(); i++) {
         DMatch first = matches[i][0];
```

```
            float dist1 = matches[i][0].distance;
            float dist2 = matches[i][1].distance;

            if (dist1 < match_ratio * dist2) {
                  int new_i = static_cast<int>(matched1.size());
                  matched1.push_back(keypoints1[first.queryIdx]);
                  matched2.push_back(keypoints2[first.trainIdx]);
                  good_matches.push_back(DMatch(new_i, new_i, 0));
            }
      }

      Mat dMatches;
      drawMatches(img1,   matched1,   img2,   matched2,   good_matches,
dMatches);

      imshow("Matches", dMatches);
```

Данный код реализует сопоставление с помощью алгоритма Brute Force Matching, суть которого состоит в выяснении дистанций (евклидово расстояние между векторами-дескрипторами) от каждой характерной точки одного изображения до каждой характерной точки другого.

Затем проводится фильтрация, в результате которой остаются только те пары точек, для которых расстояние до второго ближайшего дескриптора вдвое больше, чем до первого. Фильтрация основана на предположении, что расстояние между реально соответствующими друг другу дескрипторами много меньше, чем до ошибочно определенных. На рисунке 3.5 представлен результат.

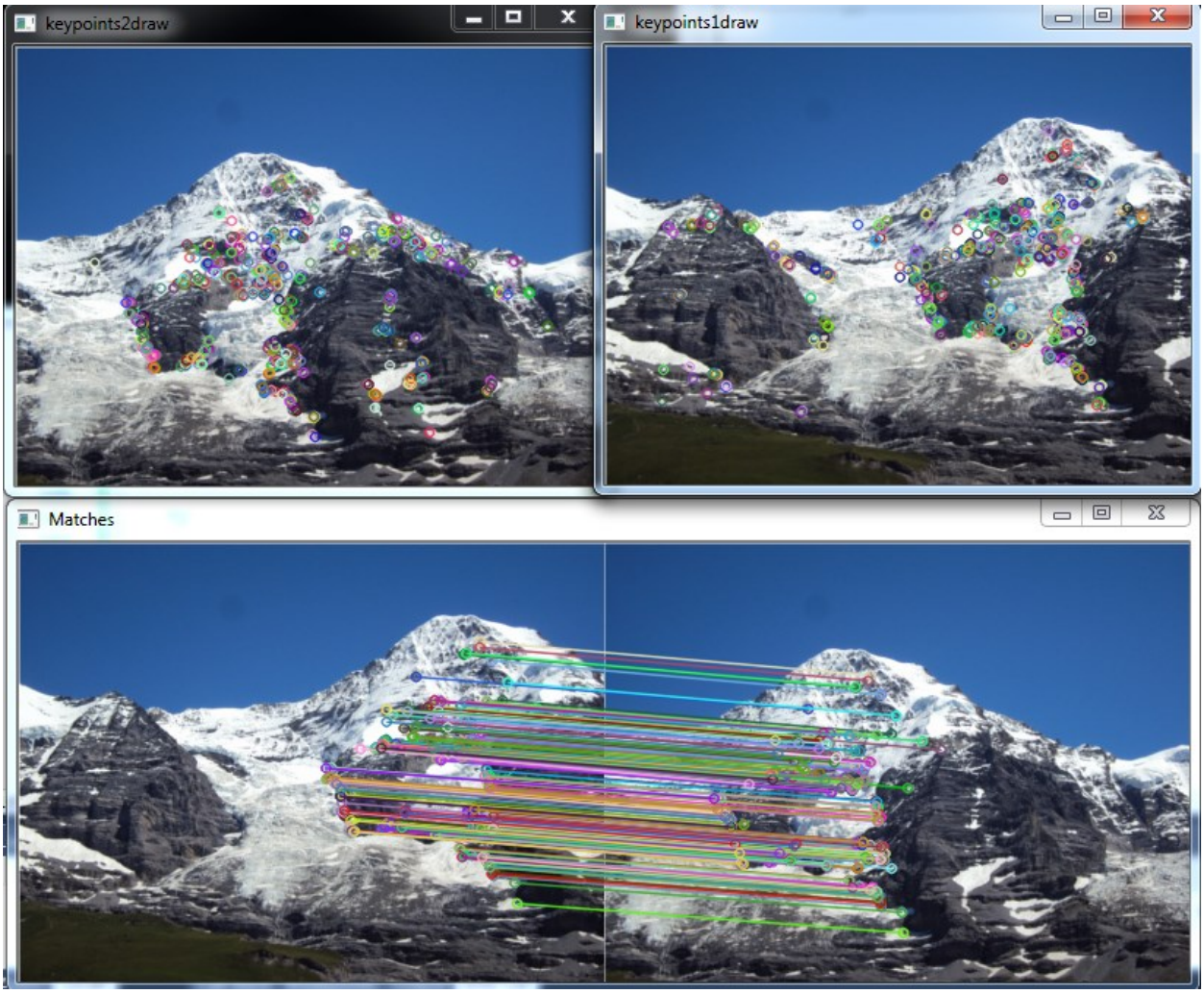

Рис. 3.5. Результат сопоставления точек.

В заключительной части на основании сопоставленных точек оценивается матрица гомографии, необходимая для трансформации второго фрагмента соответственно первому. Студентам предлагается выполнить этот этап самостоятельно (допускается использование функционала библиотеки OpenCV).

3.4. Контрольные вопросы

1. Перечислите основные этапы создания панорамного изображения.
2. Какие точки на изображении являются характерными, или ключевыми?
3. Объясните принцип работы уголкового детектора Харриса?
4. Дайте определение перспективному преобразованию изображения?
5. Для чего служит и в чем заключается алгоритм RANSAC?