МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Костромской государственный университет»

(КГУ)

ИАСТ

Кафедра автоматизированных систем и технологий

09.03.02

Направление подготовки/Специальность Информационные системы и технологии

Дисциплина Технологии компьютерного зрения

# Лабораторная №5.

# Локальные особенности.

Выполнили студенты Копосов Лев Владимирович

Копосов Владимир Владимирович

Группа 22-ИСбо-1б

Проверил \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Подпись преподавателя \_\_\_\_\_\_\_\_\_\_\_\_

Кострома

Вопросы:

1. Что такое дескриптор локальной особенности?

Дескриптор локальной особенности - это точка с характерной окрестностью. Она содержит информацию о ее характеристиках (углах, границах) и отличается от всех точек в этой окрестности.

1. Что такое инвариантность алгоритма поиска по отношению к чему-либо?

Инвариантность - требование, при котором дескрипторы устойчивы к определенным преобразованиям (к геометрическим и фотометрическим изображениям). Определение особой точки должно быть независимо к аффинным преобразованиям. Особая точка должна определяться, если передвигаем, поворачиваем или масштабируем изображение

Инвариантность к воздействию - свойство детектора всегда находить локальные особенности несмотря на это воздействие. Например: инвариантность к сдвигу яркости - детектор опирается на производные яркости, которые не меняются от сдвига.

1. Какие виды локальных особенностей вы знаете? Какие алгоритмы применяются для их поиска?

Мы знаем следующие виды локальных особенностей: углы, края, блобы, области (регионы).

Для поиска углов применяется детектор углов Харриса, Харриса-Лапласиана;

Для поиска краев применяется способ конволюции (свертки), градиентный метод;

Для поиска блобов применяется детектор Log, DoG;

Для поиска областей применяется детектор IBR, метод MSER;

Для поиска также используется универсальный дескриптор SIFT, модификация PCA-SIFT.

Вывод в консоль.

##### **Задание 1**

Создайте приложение на Python, которое выполняет следующие операции:

1. Загружает изображение из заданного файла
2. Использует детектор Ши-Томаси для обнаружения удачных точек (функция **cv2.goodFeaturesToTrack**)
3. Отображает загруженное изображение и найденные точки в окне.
4. По нажатию клавиш "вверх" и "вниз" должен изменяться параметр функции GFTT **qualityLevel** (диапазон 0.01 - 0.9).
5. По нажатию клавиш "влево" и "вправо" должен изменяться параметр функции GFTT **minDistance** (диапазон 5 - 100).
6. По нажатию Esc окно должно быть закрыто.



##### **Задание 2**

Загрузите изображение-сцену bookz.jpg, и один из образцов bookN.jpg по выбору пользователя.

Используя детектор **cv2.SIFT** и сопоставитель **cv2.BFMatcher** с параметрами по умолчанию, найдите список точек и их дескрипторов для обоих изображений методом **.detectAndCompute()**.

Примените критерий Лёве для отсева неоднозначных пар точек. Покажите сопоставления на экране, используя функцию **cv2.drawMatches()**.

Нажатие стрелок "вверх" и "вниз" должно менять порог отсева для критерия Лёве (диапазон 0.1...1.0).



##### **Задание 3**

Модифицируйте программу, составленную в задании 2, следующим образом.

Вместо использования функции **cv2.drawMatches()**, постройте по найденным соответствиям гомографическое преобразование методом RANSAC.

Используйте **cv2.perspectiveTransform()** для нахождения точек сцены, соответствующих углам объекта, и обведите его рамкой, вызвав функцию **cv2.polylines()**.



##### **Задание 4**

Модифицируйте программу, составленную в задании 3, следующим образом.

Вместо статического файла сцены bookz.jpg загрузите видео bookz.mp4, и обведите искомый объект на каждом кадре.



##### **Дополнительное задание**

Используя функцию time() из модуля time, отдельно оцените время, требуемое для выполнения операций:

* поиска точек в кадре;
* проведения сопоставления точек;
* отсева точек и поиска гомографии.

Оценку усредните по всему клипу bookz.mp4. Попробуйте произвести оценку для алгоритмов извлечения особенностей SIFT, ORB и KAZE. Для сопоставления точек используйте BFMatcher. Для SIFT и KAZE используйте метрику L1 (cv2.NORM\_L1), для ORB - расстояние Хэмминга (cv2.NORM\_HAMMING).



