

# Сопоставление разномодальных изображений (оптика и КТ-реконструкция)

Студент: Коновалов Валентин, Б05-112

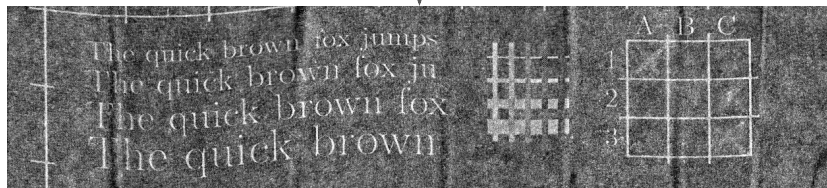
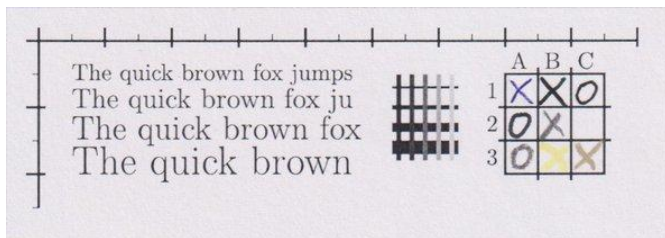
Руководитель: Полевой Дмитрий

Кафедра когнитивных технологий

# Область исследования

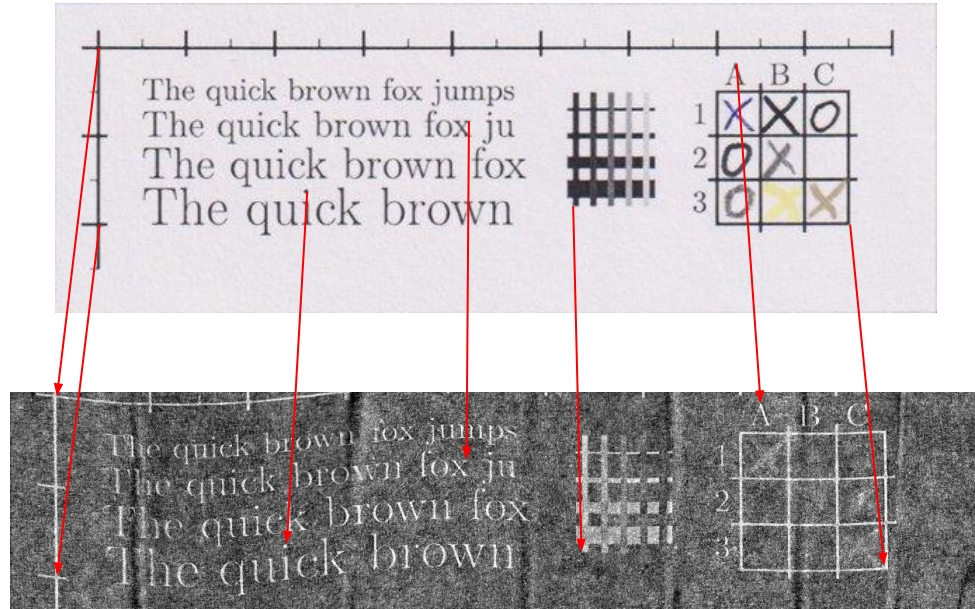
Для анализа алгоритмов разворачивания строится сопоставление между

- изображением из оптического диапазона (скан исходного образца, цветной)
- виртуальной развёрткой гнутого листа (черно-белое изображение)



# Область исследования

Ниже приведен пример сопоставления с малым количеством пар точек, сделанный вручную



# Область исследования

Строить сопоставления вручную долго и не технологично. Для оптики существуют автоматические методы. Мы попробуем на их основе сопоставить разномодальные изображения.

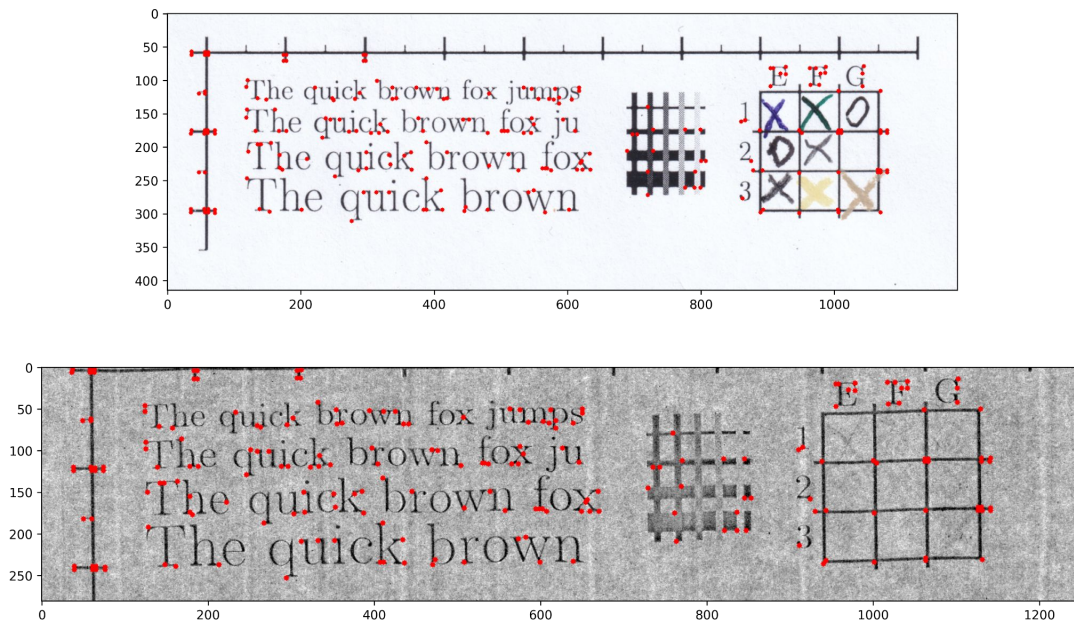
Тем не менее предполагается, что в будущем часть точек будет проставлена человеком и найденные алгоритмом соответствия также в какой-то степени будут проходить ручную верификацию.

# Постановка задачи

- разработать метод оценки качества сопоставления разномодальных изображений
- провести исследование существующих методов автоматического сопоставления, сравнить их качество

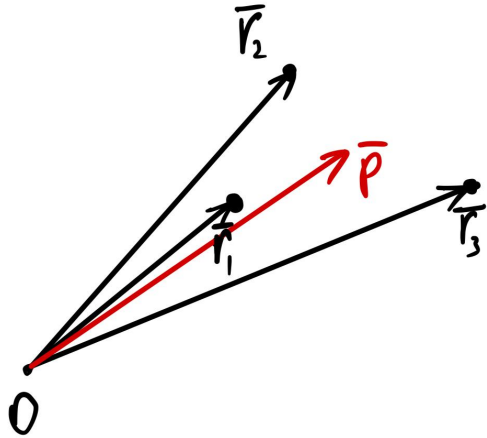
# Разработка метода оценки качества

Искажения исходного образца могут быть нелинейными, поэтому построить единую модель преобразования нельзя. Будем использовать данные с эталонной разметкой для того, чтобы хотя бы локально эту модель получить.



# Разработка метода оценки качества

Решение проблемы нелинейности: считаем преобразование локально аффинным и используем барицентрические координаты



$$\vec{p} = m_1 \vec{r}_1 + m_2 \vec{r}_2 + m_3 \vec{r}_3$$

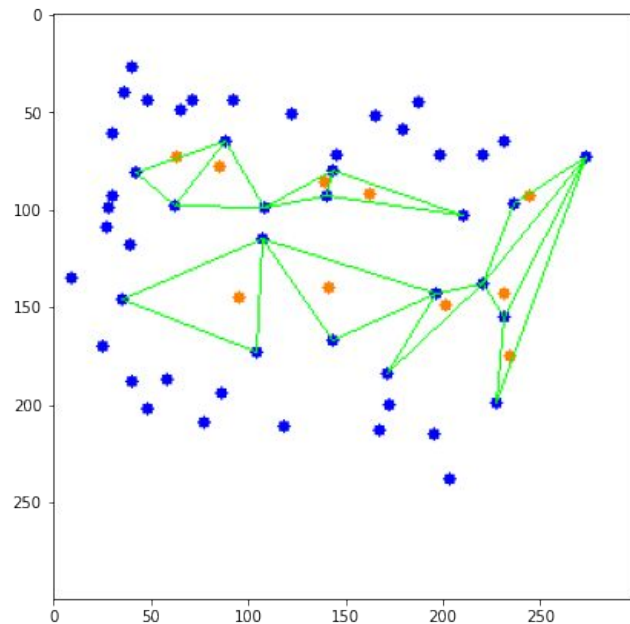
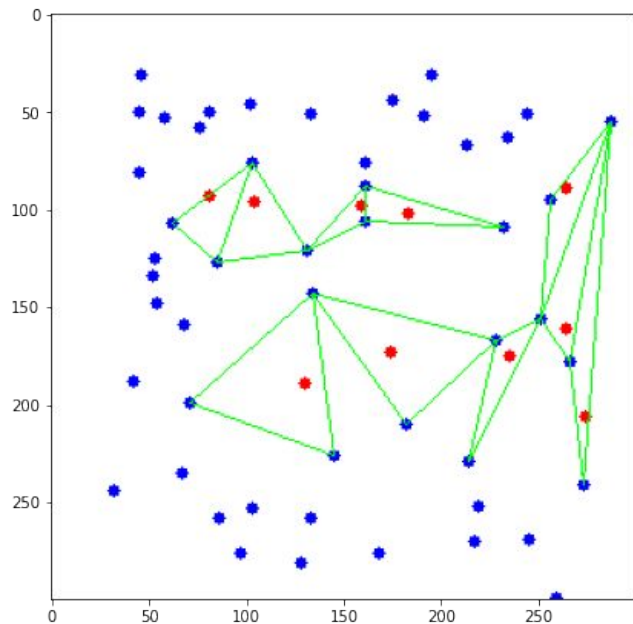
$$m_1 + m_2 + m_3 = 1$$

$$f(\vec{p}) = m_1 f(\vec{r}_1) + m_2 f(\vec{r}_2) + m_3 f(\vec{r}_3)$$

$f$  - аффинное преобразование

# Разработка метода оценки качества

Решение проблемы нелинейности: считаем преобразование локально афинным и используем барицентрические координаты



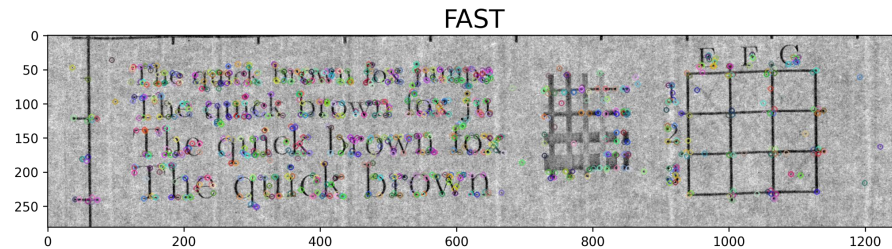
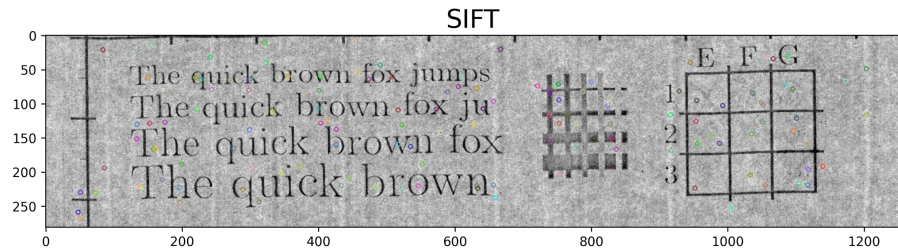
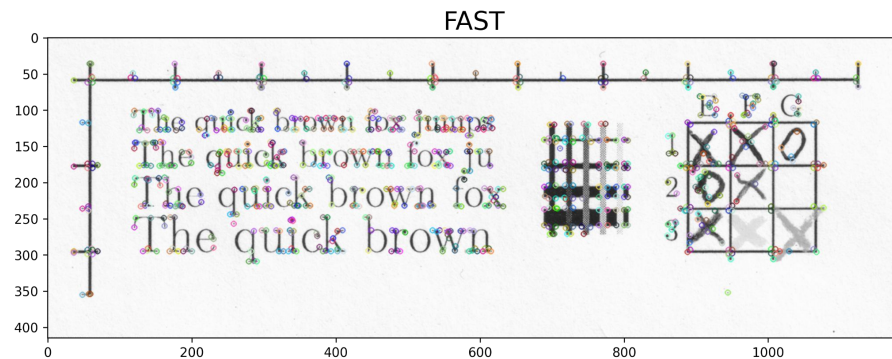
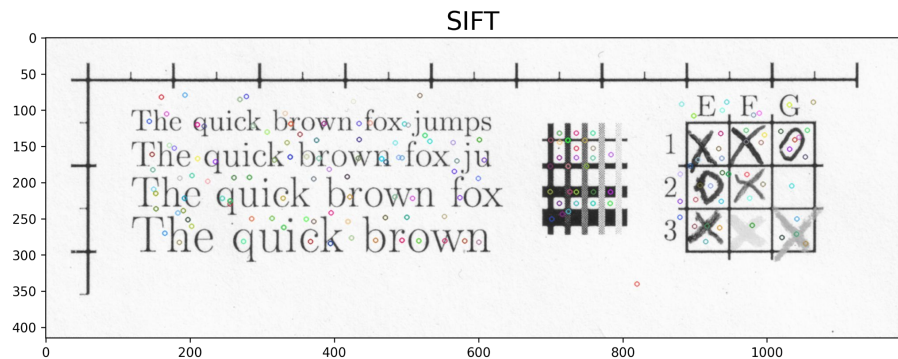


# Запуск готовых алгоритмов

Все реализации взяты из библиотеки OpenCV. Для получения результата необходимы детектор, дескриптор и мэтчер особых точек. Мэтчер везде брался один и тот же

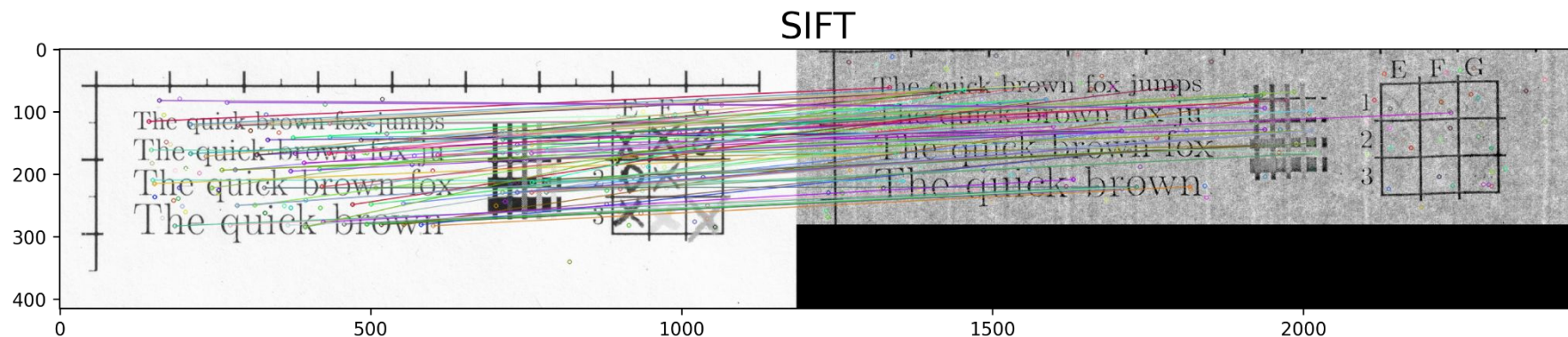
# Запуск готовых алгоритмов

Запуск различных детекторов особых точек:



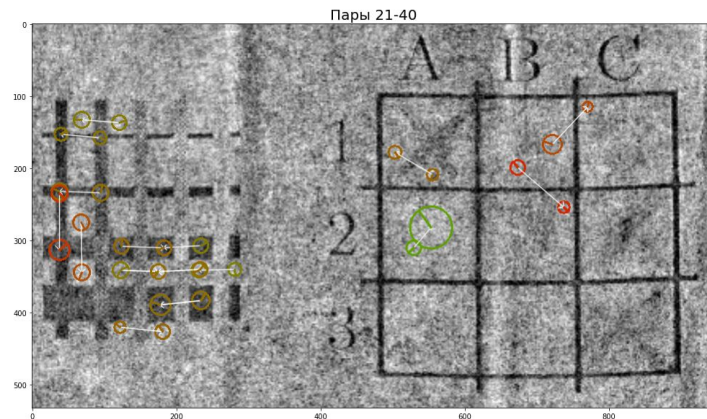
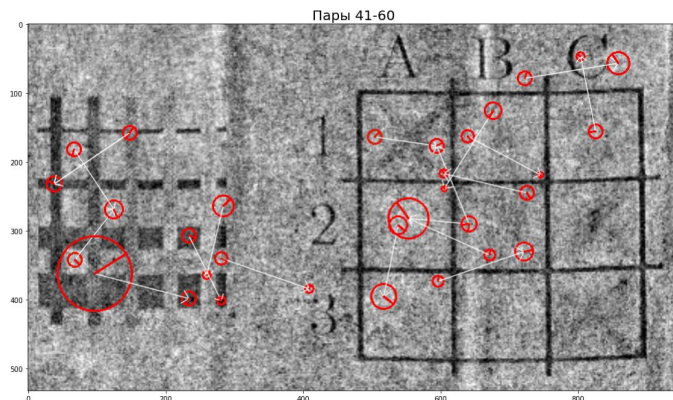
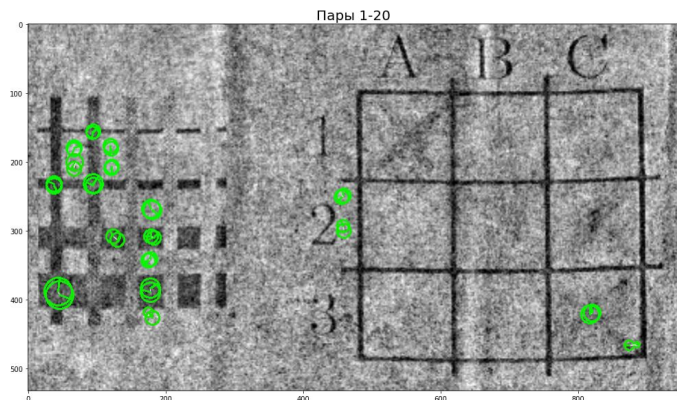
# Запуск готовых алгоритмов

Далее для найденных точек строятся дескрипторы и происходит мэтчинг



# Запуск готовых алгоритмов

Примеры хороших и плохих мэтчей для SIFT

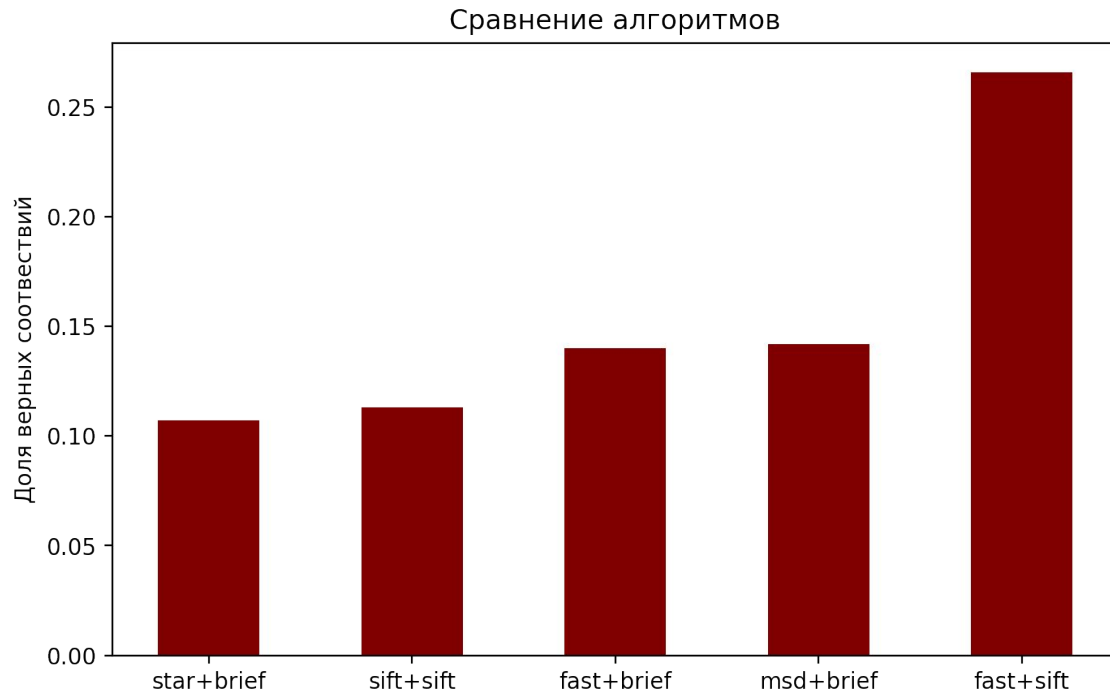


# Запуск готовых алгоритмов

Ошибка - расстояние между истинным образом точки и тем, что сопоставил ей алгоритм. Был произведён подбор гиперпараметров детекторов и дескрипторов для её минимизации

# Сравнение алгоритмов

Верными соответствиями полагаются пары с ошибкой не более 3 пикселей



# Выводы

- наибольшую долю верных соответствий дал детектор FAST в паре с дескриптором SIFT. Это можно объяснить тем, что FAST - детектор углов, коих на наших образцах довольно много
- детекторы SIFT и STAR выдают точки, которые плохо подходят для ручной верификации и точность их дальнейшего сопоставления невелика, поэтому использовать эти детекторы в нашем случае не стоит

# Дальнейшие исследования

- запустить алгоритмы не из OpenCV
- использовать Adaptive Non-maximal Suppression - метод получения особых точек с более равномерным распределением по картинке
- реализовать итеративный метод сопоставления с ручной верификацией



# Литература

- [1] Lin Chen, Franz Rottensteiner and Christian Heipke "Feature detection and description for image matching: from hand-crafted design to deep learning"
- [2] David G. Lowe "Distinctive Image Features from Scale-Invariant Keypoints"