

Санкт-Петербургский государственный университет

Кафедра системного программирования
Математическое обеспечение и администрирование информационных
систем

Александра Михайловна Елисеева

Исправление искажений в портретных снимках на широкоугольную камеру

Курсовая работа

Научный руководитель:
ст. преп. Пименов А. А.

Санкт-Петербург
2020

Оглавление

Введение	3
1. Постановка задачи	5
2. Обзор существующих решений	6
3. Реализация	8
3.1. Общая архитектура программы	8
3.2. Определение лиц на изображении	9
3.3. Ценовая функция	10
3.4. Преобразование изображения по сетке	12
3.5. Тестирование	13
Заключение	15
Список литературы	16

Введение

Камеры с широкоугольными объективами позволяют запечатлеть большее количество объектов на одной фотографии, однако в силу искажения перспективы визуальные пропорции на ней могут выглядеть неестественно. Это открывает простор для творчества, давая возможность выразенно акцентировать глубину и относительный размер, но вместе с тем обязывает аккуратнее подходить к выбору ракурса и положения объектов в кадре.

Немного остановимся на причине возникновения искажений перспективы. Форма объектов на изображении — результат прохода лучей от них через оптическую систему камеры и активации светочувствительной матрицы. В большинстве оптических систем используется перспективная (центральная) проекция. Она сохраняет прямые линии, но при большом поле зрения деформирует объекты по краям.

Рассмотрим эту проблему в контексте портретных фотографий на мобильные телефоны. Следует учитывать, что при использовании фронтальной камеры не так много возможностей выбрать удачный ракурс, особенно если на снимке присутствует несколько человек. Также в этом случае выше вероятность, что искажения нежелательны, а не используются сознательно для получения необычного результата. На Рис. 1 можно увидеть примеры искажения визуальных пропорций лиц.



(a) Фотография, снятая на камеру с полем зрения 103°



(b) Фотография, снятая на камеру с полем зрения 97°

Рис. 1

Эта проблема становится более актуальной в связи с популярностью мобильной фотографии и тем фактом, что многие смартфоны выходят с камерами с большим полем зрения (field of view). Например, одна из трех камер iPhone 11 Pro имеет FOV 120° , а фронтальная камера Google Pixel 4 - 90° .

В 2019 году YiChang Shih, Wei-Sheng Lai и Chia-Kai Liang разработали алгоритм коррекции искажений в селфи-снимках и описали его в статье [11]. Предлагается найти на фотографии регионы с лицами и локально применить к ним стереографическую проекцию. С помощью ценовой функции (energy function) на узлах сетки это преобразование осуществляется таким образом, что остальное изображение не деформируется, и на местах соединения двух различных проекций не возникает искажений.

1. Постановка задачи

Целью данной курсовой работы является реализация функциональности метода [11].

Для достижения этой цели необходимо решить следующие задачи:

1. Изучить предметную область, ознакомиться с имеющимися результатами, выбрать стек технологий
2. Разработать общую архитектуру приложения, соединяющего в себе этапы алгоритма из статьи [11]
3. Добавить функциональность обнаружения лиц и людей на изображении с использованием внешних библиотек и с использованием ввода от пользователя
4. Реализовать ценовую функцию на сетке
5. Оптимизировать ценовую функцию
6. Реализовать преобразование изображения по сетке
7. Протестировать полученные результаты и написать отчет

2. Обзор существующих решений

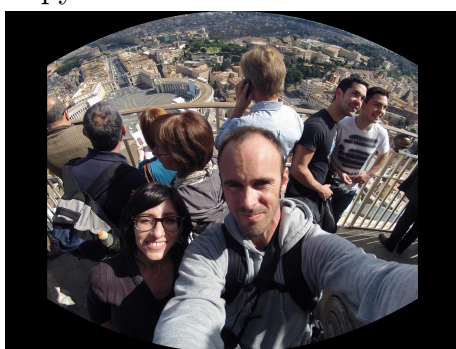
Один из способов исправления искажений перспективы — глобальные проективные преобразования. В случае с фотографиями на широкоугольную камеру хорошо себя показывают проекции, применяющиеся в картографии — среди них стереографическая проекция и проекция Меркатора [12]. Также существует проекция Панини, вдохновленная художниками эпохи Барокко (описана в работе [10]).



(а) Фотография, снятая на камеру с FOV 112°



(б) Ее стереографическая проекция



(с) Ее проекция Меркатора



(д) Ее проекция Панини

Рис. 2

Одна из первых работ [3] на эту тему была представлена Денисом Зориным и Аланом Барром в 1995 году. Они доказали, что никакая глобальная проекция не может сохранить прямые линии прямыми, не меняя при этом форму объектов. На Рис. 2 для всех представленных проекций виден этот эффект: хотя элементы у края изображения выглядят более естественными, по центру прямые линии искажаются. Также Денис Зорин и Алан Барр описали простое семейство преобразований, подходящих для обработки широкоугольных фотографий. Однако, их метод тоже создает новые искажения: например, мониторы на Рис. 3 выглядят более выпукло, чем на исходном изображении.

Некоторые существующие методы требуют явного предварительно-



(a) Фотография, снятая на камеру с FOV 92°



(b) Результат ее обработки методом [3]

Рис. 3

го задания информации — это могут быть контрольные точки, как для опции Perspective Warp в программе Adobe Photoshop [7], или прямые линии, как в работе [1].

Метод, изложенный в статье [11], является автоматическим и выполняет локальные преобразования, позволяя тем самым исправлять искажения и не создавать новых. Например, на Рис. 4 видно, как изменяются визуальные пропорции лиц, а фон изображения при этом не затрагивается.



(а) Фотография, снятая на камеру с FOV 97°



(б) Результат ее обработки методом [11]

Рис. 4

3. Реализация

3.1. Общая архитектура программы

Для реализации программы был выбран язык Python, потому что для него существуют хорошие библиотеки, связанные с оптимизацией, вычислениями и работой с изображениями.

Архитектура программы естественным образом следует за шагами алгоритма, предложенного в [11]. На Рис. 5 перечислены основные компоненты, при этом синим цветом выделены уже реализованные компоненты, функциональность которых требуется в ходе работы приложения, а красным — компоненты, реализованные мной.

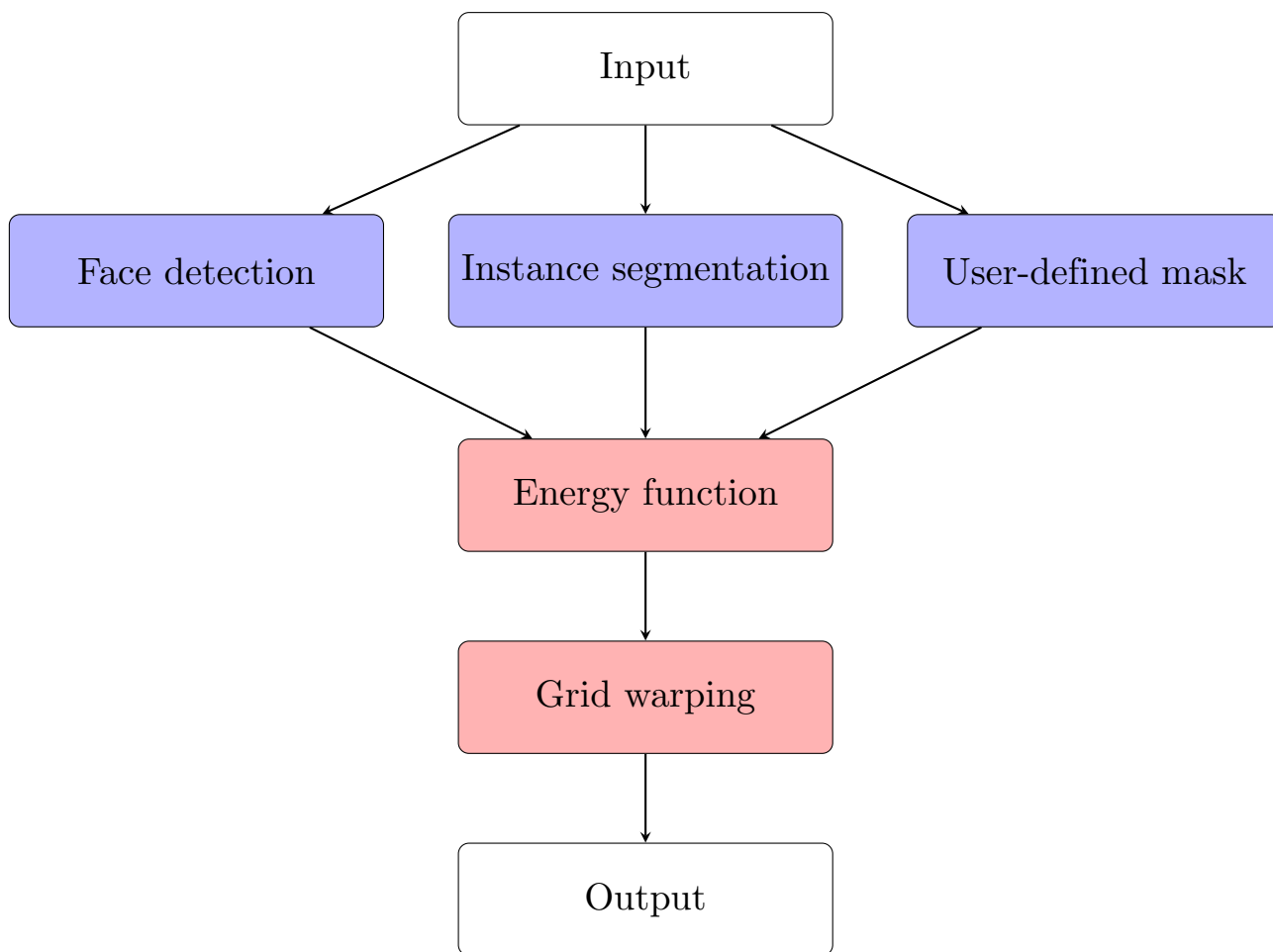


Рис. 5: Архитектура приложения

3.2. Определение лиц на изображении

В основе алгоритма лежит применение локальных преобразований к лицам людей на изображении, и для корректной работы программе необходима функциональность обнаружения лиц. Для этого было решено использовать три различных подхода, предоставляя пользователю возможность выбрать наиболее удобный для него.

1. Предлагается использовать готовую реализацию каскадного классификатора на основе признаков Хаара, предложенного в статье [13], из библиотеки OpenCV.

Данный классификатор возвращает прямоугольные области, ограничивающие найденные им лица. При использовании только таких областей возникает несколько проблем: во-первых, в них может попадать лишний участок фона рядом с лицом, во-вторых, в

них не всегда попадает вся голова. Обе из них могут привести к заметным искажениям.

2. Предлагается использоваться предварительно обученный на датасете COCO фреймворк Mask R-CNN, предложенный в статье [8]. С его помощью можно получить маску, задающую относящиеся к каждому человеку пиксели. Для достижения лучших результатов, как одно из направлений для дальнейших исследований, стоит подготовить датасет для обучения модели для определения масок лиц, а не всего тела человека.
3. Предлагается передать на вход маску, определенную пользователем вручную. Для этого необходимо изображение, где лицо каждого человека выделено своим цветом, а фон оставлен черным.

Итоговый вес, назначаемый v_i (i вершине прямоугольной сетки), считается по следующей формуле:

$$w_i = \begin{cases} 1, & \text{если } v_i \in \text{обнаруженной области} \\ 0, & \text{иначе} \end{cases}$$

3.3. Ценовая функция

Ценовая функция, описанная в статье [11], задается на вершинах сетки $\{v_i\}_{i=1}^n$ и состоит из нескольких слагаемых. Примем $\{p_i\}_{i=1}^n$ для обозначения вершин исходной прямоугольной сетки, а $\{u_i\}_{i=1}^n$ — для ее стереографической проекции.

1. Преобразование лиц

$$E_f = \sum_{k \in \text{Faces}} \sum_{i \in F_k} w_k \cdot m_i \cdot \|v_i - (S_k \cdot u_i + t_k)\|_2^2 + \lambda(S_k) \quad (1)$$

Данное слагаемое отвечает за приближение каждого лица к его стереографической проекции. Рассмотрим подробнее каждый из его элементов.

- w_k — веса, определяющие принадлежность вершины регионам с лицами; нужны, чтобы не применять преобразований к другим участкам изображения

- m_i — значение сигмoиды $(1 + \exp(-(r - r_a)/r_b))^{-1}$, где $r = \|p_i\|_2$, а r_a и r_b подбираются в зависимости от размера изображения и входного FOV так, чтобы в центре значение было 0.01, а при угле обзора 100 — 1; нужно, чтобы сильнее корректировать искажения по краям изображения
- $S_k = \begin{pmatrix} a_k & b_k \\ -b_k & a_k \end{pmatrix}$, $t_k = \begin{pmatrix} t_{k1} \\ t_{k2} \end{pmatrix}$ — скрытые переменные, которые отбрасываются после работы оптимизатора; нужны, чтобы более аккуратно выполнять преобразования для каждого лица
- $\lambda(S_k) = w_s \|a_k - s_t\|_2^2$ - коэффициент регуляризации для a_k , поощряет оптимизатор держать его значение близким к s_t
В данной реализации выбраны $w_s = 2000$, $s_t = 1$

2. Сохранение прямых

$$E_b = \sum_i \sum_{j \in N(i)} \left\| (v_i - v_j) \times \frac{p_i - p_j}{\|p_i - p_j\|_2} \right\|_2^2 \quad (2)$$

Данное слагаемое штрафует сетку за отклонение от прямых линий, чтобы избежать искажений на стыке двух различных проекций между лицом и фоном.

Здесь $N(i)$ — множество соседних с v_i вершин сверху, снизу, слева и справа.

3. Регуляризация

$$E_r = \sum_i \sum_{j \in N(i)} \|v_i - v_j\|_2^2 \quad (3)$$

Данное слагаемое штрафует сетку за отклонение соседних вершин друг от друга, чтобы избежать искажений на стыке двух различных проекций между лицом и фоном.

4. Сохранение исходного размера изображения

$$E_a = E_l + E_r + E_t + E_b \quad (4)$$

$$\begin{cases} E_l = \mathbb{I}(v_{i,x} > 0) \cdot \|v_{i,x}\|_2^2 & \forall v_{i,x} \in \delta_{left} \\ E_r = \mathbb{I}(v_{i,x} < W) \cdot \|v_{i,x} - W\|_2^2 & \forall v_{i,x} \in \delta_{right} \\ E_t = \mathbb{I}(v_{i,y} > 0) \cdot \|v_{i,y}\|_2^2 & \forall v_{i,y} \in \delta_{top} \\ E_b = \mathbb{I}(v_{i,y} < H) \cdot \|v_{i,y} - H\|_2^2 & \forall v_{i,y} \in \delta_{bottom} \end{cases} \quad (5)$$

Данное слагаемое поощряет вершины, близкие к границам изображения, не сильно удаляться от них.

- \mathbb{I} — индикаторная функция
- $v_{i,x}$ и $v_{i,y}$ — первая и вторая координаты вершины v_i соответственно
- W — ширина изображения
- H — высота изображения
- δ_{side} — соответствующая граница изображения

Таким образом, ценовая функция имеет следующий вид:

$$E = \lambda_f \cdot E_f + \lambda_b \cdot E_b + \lambda_r \cdot E_r + \lambda_a \cdot E_a \quad (6)$$

В качестве весов выбраны значения $\lambda_f = 4$, $\lambda_b = 2$, $\lambda_r = 0.5$, $\lambda_a = 4$.

Реализация ценовой функции с целью повышения производительности написана с использованием библиотеки NumPy, ее оптимизация производится методом доверительной области [2], реализованным в библиотеке SciPy [9].

3.4. Преобразование изображения по сетке

После работы оптимизатор возвращает новую сетку, в соответствии с которой необходимо преобразовать изображение. Для реализации такой функциональности существует несколько методов, самыми распространенными из них являются интерполяция сплайн-поверхностями, интерполяция радиально-базисными функциями и поточечные преобразования. Их подробную сравнительную характеристику которых можно посмотреть, например, в работах [6] и [5]. В данной работе используется поточечное аффинное преобразование, потому что оно позволяет достичь хороших результатов, и при этом не очень сложно в реализации. Стоит заметить, что такое преобразование не является гладким, поэтому в случаях, когда гладность необходимо, стоит использовать

другие методы. Триангулируя исходную сетку и полученную сетку, по каждой паре соответствующих треугольников можно получить задаваемые ими аффинные преобразования. Применяя их к каждому треугольнику исходного изображения, можно получить новое. Для реализации использовались методы библиотеки OpenCV, в частности, интерполяция фильтром Ланцоша [4].

3.5. Тестирование

Для оценивания результата использовался небольшой тестовый набор из фотографий с разным FOV и положением людей.

После работы приложения в случае с FOV 100° и более на лицах видны заметные изменения. Например, можно рассмотреть Рис. 6 — лицо после работы программы выглядит менее растянуто, чем до этого.



(a) Фрагмент фотографии, снятой на камеру с FOV 120°



(b) Результат ее обработки моим приложением

Рис. 6

В случае со средним FOV ($70^\circ - 80^\circ$) результаты работы приложения не всегда очевидны. Например, требуется некоторая внимательность, чтобы заметить на Рис. 7 изменения с лицом девушки справа.



(a) Фотографии, снятая на камеру с FOV 70°



(b) Результат ее обработки моим приложением

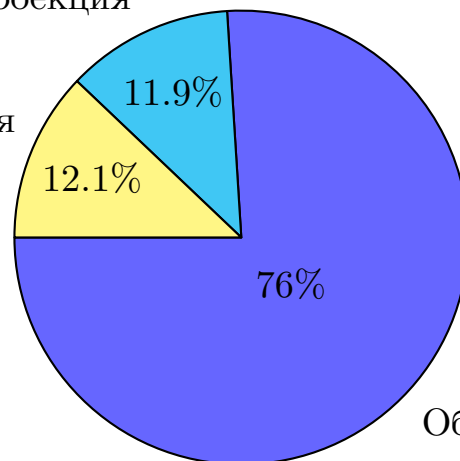
Рис. 7

В силу специфики проблемы для оценки результатов работы приложения дополнительно был проведен небольшой опрос. Его участникам было представлено несколько вопросов, каждый из которых включал в себя исходную фотографию, ее стереографическую проекцию и результат работы приложения. Среди них нужно было выбрать ту, которая показалась им наиболее естественной и близкой к реальности.

Результаты можно видеть на Рис. 8.

Стереографическая проекция

Исходная фотография



Обработка приложения

Рис. 8: Результаты опроса

Заключение

Из поставленных задач реализованы следующие:

- Разработана архитектура приложения
- В приложение встроена функциональность instance segmentation с использованием фреймворка Mask R-CNN, функциональность определения лиц с использованием готового Haar Cascade Classifier и функциональность получения маски, задаваемой пользователем
- С использованием OpenCV реализовано поточечное аффинное преобразование изображения по сетке
- Реализована оптимизационная задача: ценовая функция, необходимые ограничения
- Полученное приложение протестировано на фотографиях с разным FOV и положением людей

Разработка ведется в данном GitHub репозитории.

Список литературы

- [1] Carroll Robert, Agrawal Maneesh, Agarwala Aseem. Optimizing Content-Preserving Projections for Wide-Angle Images // ACM Trans. Graph. — 2009. — Jul. — Vol. 28, no. 3. — Access mode: <https://doi.org/10.1145/1531326.1531349>.
- [2] Conn Andrew R., Gould Nicholas I. M., Toint Philippe L. Trust-Region Methods. — USA : Society for Industrial and Applied Mathematics, 2000. — ISBN: 0898714605.
- [3] D. Zorin, A.H. Barr. Correction of geometric perceptual distortions in pictures // In SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques, pp. 257–264. — 1995.
- [4] Duchon Claude. Lanczos Filtering in One and Two Dimensions // Journal of Applied Meteorology - J APPL METEOROL. — 1979. — 08. — Vol. 18. — P. 1016–1022.
- [5] Glasbey Chris A., Mardia Kanti V. A review of image-warping methods. — 1998.
- [6] Goshtasby A. Transformation functions for image registration. — 2003.
- [7] Intwala Chintan, Agarwala Aseem. Perspective warp // US Patent 9,117,253. — 2015.
- [8] Mask R-CNN / Kaiming He, Georgia Gkioxari, Piotr Dollár, Ross Girshick. — 2017. — 03.
- [9] SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python / Pauli Virtanen, Ralf Gommers, Travis E. Oliphant et al. // Nature Methods. — 2020. — Vol. 17. — P. 261–272.
- [10] Sharpless Thomas K., Postle Bruno, German Daniel M. Pannini: A New Projection for Rendering Wide Angle Perspective Images // In Proc. 6th international conference on Computational Aesthetics in Graphics, Visualization and Imaging. 9–16. — 2010.
- [11] Shih YiChang, Lai Wei-Sheng, Liang Chia-Kai. Distortion-Free Wide-Angle Portraits on Camera Phones. — 2019. — Access mode: <https://doi.org/10.1145/3306346.3322948>.

- [12] Snyder John P. Map Projections: A Working Manual. — 1987.
- [13] Viola Paul, Jones Michael. Rapid Object Detection using a Boosted Cascade of Simple Features. — 2001. — 02.