Санкт-Петербургский государственный университет

Кафедра системного программирования

Власова Анна Сергеевна

Использование глубины для сшивки изображений с параллаксом на мобильном телефоне

Курсовая работа

Научный руководитель:
ст. преп. Смирнов М. Н.

Консультант:
Сысоенко С. С.

Санкт-Петербург
2019

Оглавление

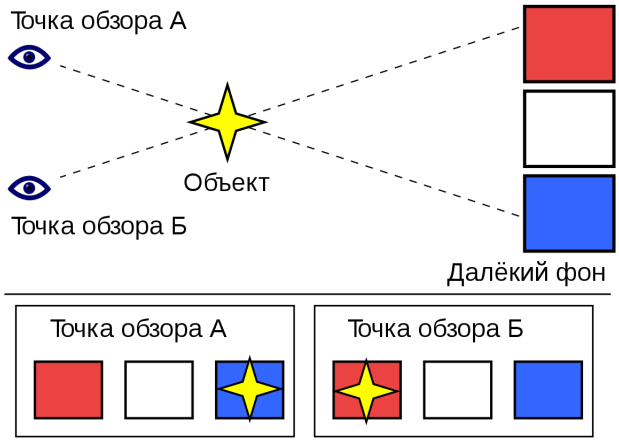
Введение

Сшивка (регистрация) изображений является распространенной задачей компьютерного зрения, заключающейся в бесшовном соединении нескольких изображений в одно. Полученное изображение охватывает большую часть пространства, поскольку обладает более широким углом обзора по сравнению с обычной фотографией. Применение подобные снимки находят, например, в создании виртуальных туров, предназначение которых подробно отображать трехмерное пространство на экране. Примером таких туров являются Google Карты – в них есть возможность "походить" по улицам (Google Street View), а также внутри некоторых заведений. Однако там эффект трехмерного пространства достигается благодаря созданию снимков с помощью специального оборудования (то есть владелец кафе сам по себе не сможет дать пользователям возможность виртуально посмотреть на свое заведение). У подавляющей части приложений для камер телефонов есть встроенная возможность снимать панорамы, но тем не менее в большинстве своем сшивка производится качественно и без артефактов только для удаленных сцен.

Задача построения панорамного снимка решается с помощью перспективного преобразования плоскости. Однако этот подход (далее он будет разобран подробнее) использует предположение о том, что сцена плоская, то есть все элементы сцены находятся примерно на одном расстоянии от камеры. При съемке в помещении эта предпосылка нарушается, поскольку появляется существенная разница в расстояниях от различных объектов сцены до камеры, и возникает эффект параллакса. Параллакс – это кажущееся изменение взаимной ориентации какого-либо объекта на фоне и на переднем плане; измеряется величиной угла между точками обзора (Рис. 1). Причиной появления данного эффекта служит смещение местоположения камеры, соответственно, при съемке с телефона без использования штатива в помещении параллакс является серьезным препятствием к созданию сшитого изображения.

В различных исследованиях задачи регистрации изображений в ос-

Рис. 1: Иллюстрация параллакса



нове решений проблемы параллакса зачастую используют такие понятия, как глубина и сегментация. Глубиной пикселя в данном случае называют расстояние от точки в пространстве, соответствующей этому пикселю, до камеры. Функция $D(x, y)$ задает карту глубин изображения. Существуют эффективные методы нахождения карты глубин, однако в данной работе информация о глубине считается данной (для этого берется датасет TUM[5], на котором глубины уже посчитаны). Сегментация – разделение изображения на множество областей по какому-либо признаку, к примеру, возможно сегментировать изображение по цвету. В основе некоторых подходов лежит такая идея: если разбить изображение на множество небольших кусочков, то можно получить набор приблизительно плоских областей, для которых задачу можно сводить к случаю плоской сцены. Например, метод As-Projective-As-Possible [3] сегментирует изображение с помощью прямоугольной сетки. При редкой сетке изображение будет получаться недостаточно сглаженным, при частой качество естественно повысится, однако сложность будет слишком большой, для того чтобы регистрацию было возможным в разумное время провести на мобильном телефоне. Для оптимизации этого подхода в данной работе используется сегментация изображения на основе информации о глубине.

1. Постановка задачи

Целью данной работы является разработка прототипа мобильного приложения (т.е. алгоритма не требующего слишком большой вычислительной сложности), позволяющего сшивать изображения на основе метода APAP с предварительной сегментацией по глубине. В зависимости от результатов сравнения полученного алгоритма с исходным требовалось проверить гипотезу о том, что APAP таким образом действительно оптимизируется. Входными данными помимо двух изображений служат карты глубин. Для достижения поставленной цели были определены следующие задачи:

- провести обзор существующих решений регистрации изображений;
- проанализировать различные способы сегментации изображения по глубине;
- реализовать сегментацию;
- встроить ее в APAP;
- провести апробацию, сравнить результаты с исходным алгоритмом.

2. Обзор

2.1. Гомография: базовый подход

Гомографией называется проективное преобразование плоскости, задается оно следующей матрицей:

$$H = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & 1 \end{pmatrix}$$

где коэффициенты g и h отвечают за изменение перспективы, а остальные параметры за масштабирование и сдвиги по осям.

Существуют разные алгоритмы восстановления этой матрицы [4]. По парам точек (на исходном изображении и на получившемся после преобразования гомографии) можно восстановить H с помощью алгоритма прямого линейного преобразования (DLT). Также для этого используют подход RANSAC (RAndom SAmple Consensus), поскольку он устойчив к выбросам. Его идея состоит в том, что на каждой итерации в цикле из множества данных соответствующих друг другу пар точек случайно выбирается некоторая часть. По ним строится матрица преобразования. Далее для всех точек проверяется, удовлетворяет ли построенная матрица уравнению на эту точку ($x' = Hx$ в однородных координатах, то есть в таких, в которых объекты не изменяются при умножении их координат на некоторый скаляр). Затем проверяется, какая матрица лучше, построенная на этом шагу или "текущая лучшая", шаг алгоритма повторяется.

Методы, в основе которых лежит использование глобальной гомографии, то есть матрица H одна для всех точек изображения, пользуются предпосылкой о том, что сцена плоская. К сожалению, в помещении это далеко не так и использование данного подхода не приводит к хорошим результатам. Однако стоит заметить, что большинство более продвинутых алгоритмов основано на гомографии.

2.2. 3D реконструкция и алгоритм имитации отжига

Алгоритм предложенный в статье [1], описывающей этот метод, имеет следующую структуру:

- на изображениях выбираются особые точки, затем они сопоставляются между собой на различных изображениях;
- по этим данным с помощью восьми-точечного алгоритма находят движение камеры между снимками (матрица поворота и сдвига);
- далее становится возможным восстановить 3D-структуру особых точек, то есть глубину в них;
- точки кластеризуются таким образом, чтобы в каждом кластере точки лежали приблизительно в одной плоскости;
- для каждого кластера находится матрица его преобразования.

Данный алгоритм в конечном итоге в работе никак не использовался и приведен исключительно с целью обзора разносторонних алгоритмов из области сшивки изображений.

Задачу кластеризации в данном случае формально можно сформулировать так:

$$\min_{\{\theta_k\}_{k=1}^{K}} \sum_{k=1}^{K} \sum_{y_i \in C_k} d(y_i, g_{\theta_k})$$

где K – количество кластеров, θ_i – вектор над $\mathbb{R}^3$, задающий плоскость (он состоит из величин обратных к пересечениям плоскости с Ox, Oy и Oz), а d – некоторая метрика, позволяющая оценить, насколько далека точка от плоскости (можно рассмотреть квадрат длины расстояния до проекции). Авторы подхода решают данную задачу методом имитации отжига.

Общая идея состоит в том, что некоторая “температура” постепенно уменьшается (на каждой итерации в цикле она умножается на заданный параметр $\alpha < 1$). Внутри цикла рассчитывается в зависимости от

температуры вероятность перехода i-той точки в k-ый кластер по формуле

$$P(y_i \in g_{\theta_k}) = \frac{exp(-D(y_i, g_{\theta_k})/T)}{\sum_{j=1}^{K} exp(-D(y_i, g_{\theta_j})/T)}$$

где D – функция с выбранным изначально параметром, показывающая ошибку между точкой y_i и плоскостью, которой она принадлежит. Далее рассчитываются параметры плоскостей, для этого используется метод градиентного спуска. Подобные итерации повторяются, пока температура не достигнет 0.

2.3. As-Projective-As-Possible

Для начала рассмотрим подробнее поиск глобальной матрицы гомографии. Как уже было замечено, нужно решить уравнение $x' = Hx$ в однородных координатах. Тогда векторное произведение векторов, стоящих слева и справа от знака равенства является нулевым вектором размерности 3. Это условие можно записать так:

$$0_{3\times1} = \begin{bmatrix} 0_{1\times3} & -x^T & y'x^T \\ x^T & 0_{1\times3} & -x'x^T \\ -y'x^T & x'x^T & 0_{1\times3} \end{bmatrix} h, \quad h = \begin{bmatrix} h1 \\ h2 \\ h3 \end{bmatrix}$$

где h_i – это i-ая строка искомой матрицы H. Всего в данной матрице 2 линейно-независимых строки (пусть они верхние). Назовем a_i матрицу из этих двух строк с подставленными значениями точки x_i и ее образа x'_i. Тогда задача формулируется как:

$$\hat{h} = \underset{h}{\operatorname{argmin}} \sum_{i=1}^{n} \|a_i h\|^2 \tag{1}$$

где норма h равна единице. Решением является один из сингулярных векторов матрицы составленный из a_i. В алгоритме APAP изображение разбивается сеткой на прямоугольники. Далее решается взвешенное уравнение (1). Чем ближе некоторая точка x_i к x_*, в локальной области которой ищется преобразование, тем больше ее вес. Итоговая

формула имеет следующий вид:

$$\hat{h} = \underset{h}{\operatorname{argmin}} \sum_{i=1}^{n} \|w_*^i a_i h\|^2$$

где $w_*^i = exp(-\|x_* - x_i\|/\sigma^2)$ (σ фиксируемый параметр). Решением данного уравнения также является один из сингулярных векторов, но на этот раз, сингулярных векторов произведения диагональной матрицы составленной из весов и матрицы составленной из векторов a_i. Первоначально изображение разбивается сеткой на множество прямоугольников и в качестве x_i выбираются их центры.

В статье, описывающей данный метод, приводится сравнение результатов его работы с многими другими алгоритмами регистрации изображений. Исходя из этого сравнения можно сказать, что APAP работает лучше многих альтернативных подходов.

3. Реализация

3.1. Инструменты

Поскольку задача поставлена так, что предполагается сделать лишь прототип мобильного приложения, особых ограничений на язык программирования не стояло. Однако в силу того, что были найдены открытые реализации алгоритма APAP своими же авторами на C++, а данная работа предполагает оптимизацию этого метода, в качестве языка разработки был выбран C++.

3.2. Алгоритм

В основе реализуемого алгоритма лежит подход APAP. Однако нетрудно заметить, что время его работы зависит от размерности сетки, на которую разбивается изображение. В случае слишком редкой сетки может получиться некачественно сшитая фотография – некоторые объекты могут двоиться (ghosting effects), кроме того могут быть видны швы. В случае мелкой сетки время работы оказывается слишком большим, и по причине этого вычислительных возможностей мобильного телефона может не хватать.

В рамках рассматриваемого алгоритма для начала требовалось сегментировать исходное изображение по глубине, выделить сегментированные объекты в прямоугольники. Тогда изображение разбивается неравномерной сеткой, при этом для каждого отдельного прямоугольника в этой сетке матрица гомографии получается приблизительно одной и той же, поскольку весь прямоугольник находится на одной глубине. Таким образом, подход As-Projective-As-Possible должен быть оптимизирован с использованием информации о глубине.

3.3. Сегментация

В классической задаче сегментации требуется выделить отдельные объекты на изображении. Поскольку для оптимизации APAP требуется

разбить все изображение на прямоугольники, классические алгоритмы сегментации не подходят. В данной работе предложено два варианта решения поставленной задачи сегментации на прямоугольники, один из которых и был использован в дальнейшем.

3.3.1. Деревья решений

Задача сегментации изображения на неравномерную сетку была рассмотрена, как задача машинного обучения: предположим, что имеется два фактора – координаты по оси Ox и по оси Oy, и по ним требуется предсказать глубину в точке с заданными координатами. Тогда дерево решений будет работать следующим образом: оно будет выбирать для первоначальной области (всего изображения) фактор (то есть одну из осей координат) и некую константу C, такую, что если разделить все точки на те, у которых координата по выбранной оси меньше либо равна C и те, у которых она больше C, то некоторая функция потерь для двух получившихся множеств будет оптимальной. Далее этот алгоритм рекурсивно запускается для каждой из полученных веток дерева. Получается, что на каждом шаге для каждого из обрабатываемых прямоугольников выбирается оптимальный разрез – вертикальная или горизонтальная прямая.

Данный алгоритм не показал приемлемых результатов (Рис. 2). Эта часть была реализована на Python с использованием библиотеки Scikit-Learn.

3.3.2. Разбиения и склеивания

Идея предложенной сегментации состоит в следующем: сначала идет стадия разделения, на которой для первоначального прямоугольника проверяется выполнение некоторого предиката, и в случае его невыполнения прямоугольник делится на 4 равных (пополам по оси Ox и по оси Oy). Для каждого из полученных прямоугольников запускается эта же процедура. На стадии склеивания для каждого прямоугольника проверяется, не стоит ли склеить между собой "строчки" из прямоуголь-

Рис. 2: Сегментация с помощью деревьев решений

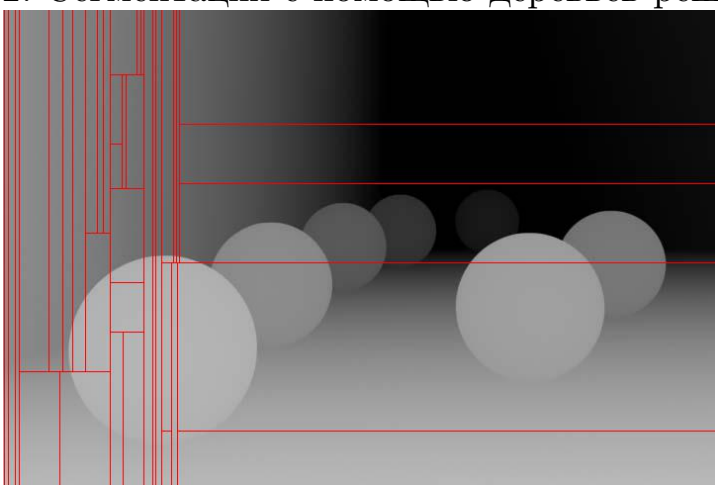
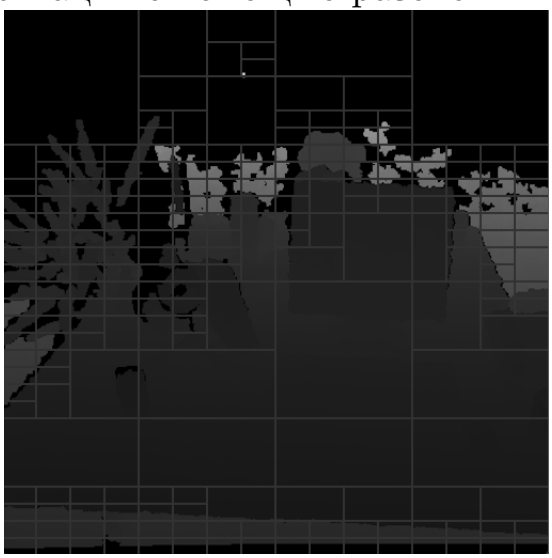


Рис. 3: Сегментация с помощью разбиений и склеиваний



ников внутри него, а затем ”столбцы”. В качестве предиката использовалось условие о том, что прямоугольник уже достаточно маленький, либо дисперсия в нем достаточно маленькая. Данная сегментация хороша своей простотой и тем, что время ее работы составляет порядка секунды. Пример ее результата показан на (Рис. 3).

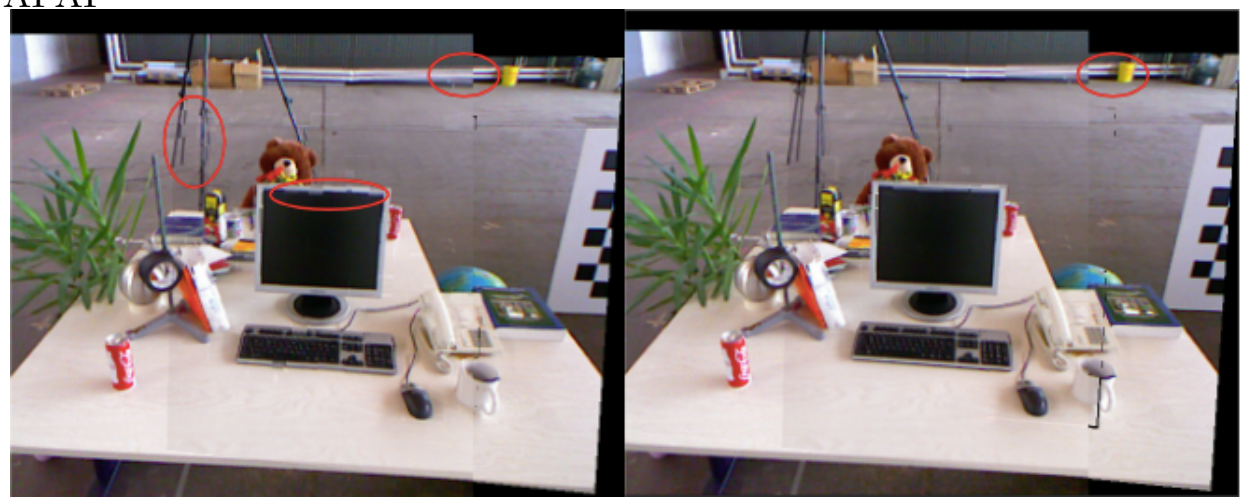
4. Апробация

Исходный APAP, так же как и APAP с использованием сегментации, были протестированы на данных из TUM датасета[5] с предподсчитанной глубиной. Брались изображения, снятые в помещении, чтобы проблема параллакса была особенно актуальной. Пример сравнения результатов показан на (Рис. 4). Было замечено, что реализованный подход показывает результаты лучше исходного по количеству артефактов. Кроме того, время его работы существенно отличается в меньшую сторону. Благодаря тому, что каждый крупный объект, находящийся на одной глубине, не разбивался мелкой сеткой и для него вычислялась лишь одна матрица гомографии, было сэкономлено порядка 70% времени работы исходного алгоритма. Так, например, левое изображения на (Рис. 4) было получено оригинальным As-Projective-As-Possible с разбиением 50 на 50 клеток за 185 секунд, тогда как время работы алгоритма с использованием сегментации составило лишь 60 секунд (и при этом дало лучший результат).

Поскольку для улучшения производительности использовалась дополнительная информация, а именно карта глубины одного из изображений, хотелось бы отметить, что ее действительно возможно получить. Для получения глубины на телефоне можно выделить следующие способы:

- классические алгоритмы восстановления информации о глубине принимают на вход два изображения, при этом для их получения должна использоваться стерео-пара. В случае телефонов с двойной камерой можно использовать такую камеру в качестве стерео-пары (начиная с версии Android P одновременный доступ к двум камерам стал возможен);
- на некоторых телефонах установлены Time-of-Flight камеры, с помощью них также возможно восстановить глубину;
- даже в случае отсутствия перечисленных камер, существует алгоритм, который с использованием глубинного изучения позволяет

Рис. 4: Результаты APAP (слева) и результаты оптимизированного APAP



по одному изображению восстанавливать карту глубины [2].

Заключение

Все поставленные цели были достигнуты.

- проведен обзор некоторых существующих решений регистрации изображений;
- предложено два варианта сегментации изображения, один из них с хорошим результатом;
- реализованы предложенные алгоритмы сегментации;
- лучший из них встроен в APAP;
- произведено сравнение результатов полученного алгоритма с исходным.

Список литературы

[1] B. Han C. Paulson D. Wu. Depth-based image registration via three-dimensional geometric segmentation // IET Computer Vision. — 2012.

[2] David Eigen Christian Puhrsch Rob Fergus. Depth Map Prediction from a Single Image using a Multi-Scale Deep Network. — 2014.

[3] Julio Zaragoza Tat-Jun Chin Michael S. Brown David Suter. As-Projective-As-Possible Image Stitching with Moving DLT.

[4] Szeliski Richard. Image Alignment and Stitching: A Tutorial. — 2006. — URL: http://www.cs.toronto.edu/~kyros/courses/2530/papers/Lecture-14/Szeliski2006.pdf.

[5] TUM dataset. — URL: vision.in.tum.de/data/datasets/rgbd-dataset/download.