Очищение текстур фасадов зданий с использованием их структуры

Владимир Кононов, Вадим Конушин, Антон Якубенко, Антон Конушин
Факультет Вычислительной Математики и Кибернетики
Московский Государственный Университет им. М.В. Ломоносова, Москва, Россия
{vladimir.kononov, vadim, toh, ktosh}@graphics.cs.msu.ru

Аннотация

В настоящее время активно развивается область трехмерной реконструкции городов по изображениям. Ее важной составной частью является создание реалистичных текстур фасадов городских зданий по фотографиям. Но на реальных снимках фасад зачастую загорожен различными объектами переднего плана, такими как деревья, провода, дорожные знаки и т.д. Поэтому для качественной реконструкции необходимо очищение текстуры фасадов от таких объектов. Современные методы очистки текстур работают очень медленно и не подходят для реального применения. В этой работе предлагается интерактивный алгоритм для нахождения и удаления объектов переднего плана на текстурах фасадов, основанный на использовании информации об их структурах.

Ключевые слова: *восстановление текстур, обработка изображений, 3D-реконструкция зданий, анализ структуры фасадов.*

1. ВВЕДЕНИЕ

Одним из этапов построения трехмерных моделей зданий как составной части трехмерной реконструкции городов является очищение текстур, полученных из фотографий, от различных объектов переднего плана, загораживающих фасад. Актуальность данной задачи вытекает из невозможности во многих случаях сфотографировать здание без посторонних объектов перед ним.

В этой статье предлагается алгоритм очистки текстуры зданий от объектов переднего плана с использованием информации о структуре фасада. На вход алгоритму подается ректифицированная текстура фасада, а также координаты и типы окон. Ректифицированная текстура может быть, например, получена указанием четырех углов стены на исходном изображении с последующим расчетом гомографии. На выходе алгоритма – текстура фасада с восстановленными областями под найденными объектами переднего плана. Ключевым требованием к алгоритму является скорость работы, достаточная для интерактивного взаимодействия.

Положение и типы окон могут быть получены с помощью автоматических или полуавтоматических алгоритмов, как [2]; или же размечены вручную. В предложенном алгоритме структура фасада задается в виде сетки из прямоугольных ячеек, каждая из которых содержит одно окно. Внешний вид всех ячеек одного типа считается одинаковым. Количество типов ячеек определяется количеством заданных типов окон, поэтому если в двух разных по внешнему виду ячейках находятся одинаковые окна, они должны быть отмечены как два разных типа окна (Рис. 1). Это может быть сделано вручную, или с использованием алгоритмов, умеющих различать типы ячеек, а не окон. В любом случае возможные усилия, затрачиваемые пользователем на этом этапе значительно меньше усилий, затрачиваемых на очищение текстуры существующими методами.

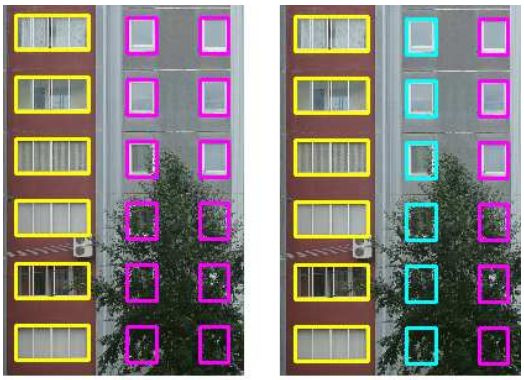


Рис. 1: Пример входных данных алгоритма: a) неправильно – типы окон не соответствуют типам ячеек; b) правильно – типы окон соответствуют типам ячеек.

2. СУЩЕСТВУЮЩИЕ МЕТОДЫ

К настоящему времени сформировалась целая исследовательская область по задаче восстановления текстуры (Image Completion). Эта задача состоит в заполнении некоторого участка изображения, называемого неизвестной областью, с использованием оставшейся части изображения, причем результат визуально не должен выглядеть неестественно. За последние десять лет в области восстановления текстуры появилось множество различных алгоритмов, которые можно грубо разделить на два больших класса.

Методы первого класса основаны на составлении и решении уравнений в частных производных [1]. Алгоритмы этого класса хорошо подходят только для восстановления узких и небольших областей, таких как царапины на фотографиях. В противном случае результат получается очень размытым.

Методы второго класса основаны на копировании информации с остальной части изображения на неизвестную область попиксельно или небольшими фрагментами. Некоторые алгоритмы жадно заполняют неизвестную область, часто с использованием функции приоритета [3]. Другие же формулирует специальную функцию энергии, и находят такое копирование фрагментов, которое минимизирует эту функцию [4].

В любом случае всем алгоритмам восстановления текстур необходима размеченная вручную или другими алгоритмами неизвестная область. Также они работают непозволительно долго для использования их в интерактивных приложениях (от нескольких десятков минут до нескольких часов на изображениях в 1 – 3 мегапикселя). Поэтому в чистом виде они не могут применяться для решения поставленной задачи.

В статье [9] авторы также имеют дело с фасадами. Однако их алгоритм использует размеченную вручную неизвестную область, а также ограничивается работой с фасадами, высоты всех этажей которых одинаковые.

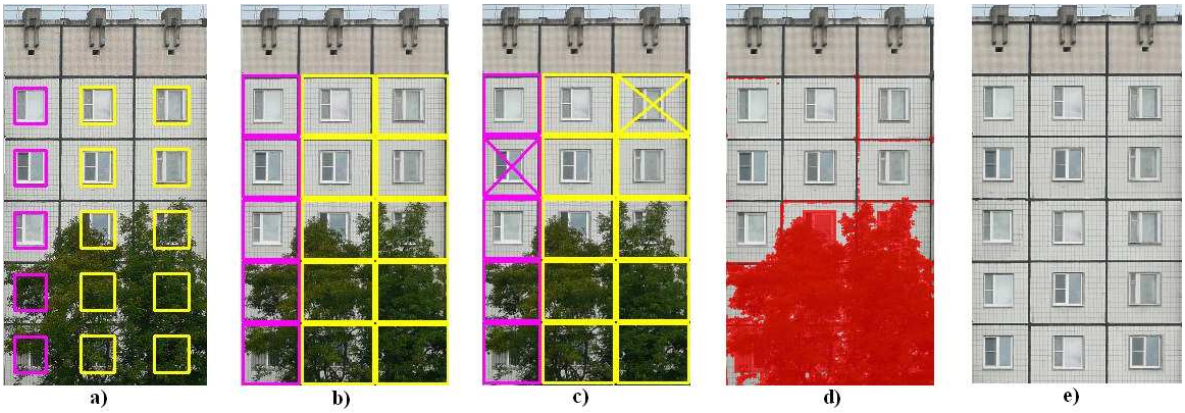


Рис. 2: Схема работы предложенного алгоритма: a) входная текстура, координаты и типы окон, b) построение структуры, c) нахождение чистой ячейки каждого типа, d) сегментация, e) восстановление текстуры.

Кора и Расмуссен в [6] недавно предложили алгоритм для автоматического восстановления структуры фасада с последующим нахождением и удалением объектов переднего плана, загораживающих здание. Схема этого метода похожа на схему предлагаемого алгоритма. Однако этот метод работает только для простых случаев загораживания. Например, он не справляется со случаями, когда объекты переднего плана загораживают более половины или близкое к этому число структурных ячеек. Также все примеры результатов работы показаны только для простых фасадов.

В данной статье предложен алгоритм, единственным ограничениям которого является наличие хотя бы одной полностью чистой ячейки каждого типа. Он не нуждается в ручной разметке неизвестной области, и может работать как полностью автоматически, так и интерактивно с минимальным пользовательским взаимодействием.

3. ОПИСАНИЕ ПРЕДЛОЖЕННОГО МЕТОДА

Предложенный алгоритм состоит из четырех шагов (Рис. 2):

1. Построение структуры фасада по окнам.
2. Нахождение чистой ячейки каждого типа.
3. Сегментация объектов переднего плана в остальных ячейках.
4. Восстановление текстуры в областях, отсегментировавшихся как объекты переднего плана.

3.1 Построение структуры фасада по окнам

На первом шаге алгоритм строит сетку фасада, то есть разбивает его на ячейки, используя информацию о координатах и типах окон. Изначально ячейки фасада считаются равными окнам. Затем они постепенно расширяются во все стороны до касания соседними элементами друг друга или границ фасада, при этом сохраняются размеры всех ячеек одного типа (Рис. 2b). Предложенный алгоритм подразумевает, что все ячейки одного типа имеют с точностью до освещения одинаковую текстуру, не считая возможных объектов переднего плана. Однако если для стен такое предположение естественно и выполняется для большинства зданий, то окна в одинаковых ячейках могут заметно отличаться по текстуре из-за разных занавесок, штор и возможности быть открытыми или закрытыми. Поэтому дальнейшая работа ведется с ячейками без окон, а сами окна обрабатываются отдельно.

3.2 Нахождение чистой ячейки каждого типа

После построения сетки фасада алгоритм работает отдельно с каждым типом ячеек. Следующим шагом является поиск полностью чистой ячейки. Он основан на использовании метрики сходства ячеек одного типа. Особенностью задачи сравнения является то, что ячейки могут отличаться друг от друга не только из-за загораживающих объектов переднего плана. Неточность разметки и неточность самого фасада часто приводит к сдвигу в несколько пикселей между текстурами ячеек одного типа. Также отличия могут быть вызваны разностью в освещении. Корректная метрика сравнения должна быть как можно более инварианта к этим особенностям.

Было рассмотрено несколько различных метрик, в том числе метрика на основе взаимной информации и сумма квадратов попиксельных разностей. На практике лучшие результаты показала метрика, сравнивающая близости краев. Для вычисления самих краев используется алгоритм Canny, а карты близости строятся с помощью алгоритма Distance Transform. Затем получившиеся карты сравниваются попиксельно (Рис. 3). Нормализованная сумма квадратов разностей по всем пикселям карт близости является значением метрики.

Рис. 3: Схема работы метрики сравнения ячеек. От краев к центру: исходные ячейки, результат алгоритма Canny, результат Distance Transform, попиксельное сравнение.

Далее решается проблема нахождения абсолютно чистой ячейки среди всех элементов одного типа. Один из вариантов решения состоит в том, чтобы создать синтетическую ячейку, каждый пиксель которой является медианой соответствующих точек всех элементов данного типа. Затем, используя метрики, можно посчитать расстояние от всех ячеек до синтетической, и наиболее близкую назвать чистой. Проблема данного метода в том, что полученная синтетическая ячейка часто сильно размыта, из-за чего чистая ячейка может найтись неправильно. Кроме того, такой метод работает только в случае, когда более половины ячеек являются чистыми, что само по себе очень жесткое ограничение. Из-за этих же недостатков нельзя сразу решать задачу целиком простой медианной заменой (Рис. 6a).

Поэтому был предложен другой метод, не создающий в процессе работы синтетических ячеек, а использующий только

расстояния между реально существующими элементами. Чистой ячейкой объявляется та, сумма расстояний от которой до других по метрике минимальна. Этот подход использует тот факт, что чистые ячейки похожи друг на друга, а содержащие объекты переднего плана – нет. Например, ячейка, содержащая дерево, не похожа на ячейку, содержащую фонарный столб. Для этого метода ограничение на входные данные формулируется следующим образом: на фасаде должна быть хотя бы одна полностью чистая ячейка каждого типа.

Другим плюсом нахождения полностью чистой ячейки является возможность клонирования ее на все остальные элементы. В ряде случаев этого бывает достаточно, но часто результат бывает заметно синтетическим и неестественным (Рис. 6b). Поэтому необходимы дальнейшие шаги по поиску объектов переднего плана и восстановлению текстуры под ними.

3.3 Сегментация ячеек с нахождением объектов переднего плана

На этом этапе пиксели ячеек сегментируются на два класса: фасад и объекты переднего плана. Предложенный алгоритм формулирует задачу сегментации в виде задачи минимизации энергии специального вида, состоящей из унарных и бинарных слагаемых.

$$E = \sum_{p} E_u(p, l) + \sum_{p,q} E_b(p, q, l_p, l_q)$$

Здесь p, q – различные пиксели, а l – метка, которая может принимать значение "фасад" или "объект переднего плана".

Чтобы полностью включить информацию о системе, надо учитывать в минимизируемой энергии пиксели со всех ячеек одного типа. То есть считать бинарные слагаемые не только для соседних пикселей внутри ячейки, но и для всех пикселей на одинаковых позициях в разных ячейках. Однако на практике этот метод оказывается неприменим из-за большой вычислительной сложности минимизации такой энергии.

Поэтому предложенный метод сегментации работает с каждой ячейкой поодиночке. Для формулирования энергии остается воспользоваться найденной абсолютно чистой ячейкой. К минусам такой модели относятся неполное использование знаний о системе, а также зависимость от правильного нахождения полностью чистой ячейки. К плюсам – высокая скорость работы и приемлемые на практике результаты.

Значения бинарных слагаемых для соседних точек должны отражать тот факт, что близкие по значению пиксели скорее всего относятся к одному классу: или оба "объекты переднего плана" \ или оба "фасад". Другими словами, объекты переднего плана выделяются на фоне фасада. Поэтому в случае различных меток точек энергия должна зависеть от цвета соседних пикселей и быть обратно пропорциональной их похожести. В случае одинаковых меток энергия просто равна нулю.

$$E_b(p, q, l_p, l_q) = \begin{cases} \exp(-\|I(p) - I(q)\|), & l_p \neq l_q \\ 0, & l_p = l_q \end{cases}$$

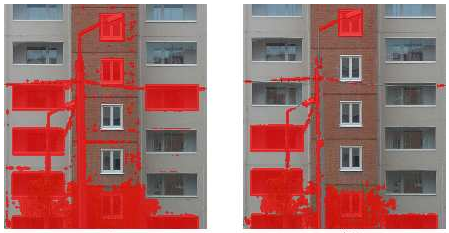
Унарные слагаемые должны отражать вероятности пикселей быть частью объекта переднего плана или частью фасада. Известна только одна чистая ячейка, соответственно можно считать, что чем ближе пиксель по цвету к соответствующему пикселю чистой ячейки, тем больше у него шансов быть частью фасада. Этот факт можно записать как значение унарного слагаемого при метке "фасад". При метке "объект переднего плана" значение унарного слагаемого является просто некоторой константой.

$$E_u(p, l) = \begin{cases} \exp(-\lambda * \|I(p) - I_{best}(p)\|), & l = facade \\ \exp(-\lambda * C), & l = foreground \end{cases}$$

I_{best} – абсолютно чистая ячейка, и C – экспериментально подобранная константа. λ остается единственным параметром системы, который показывает важность бинарных слагаемых по сравнению с унарными (Рис. 4). Изменяя только его, можно добиться приемлемого результата. Единого значения λ для всех случаев не существует. Это обусловлено различиями в освещенности ячеек и характере текстуры.

Для минимизации используется алгоритм разреза графов [7][8]. Используемая функция энергии удовлетворяет условиям регулярности, и, следовательно, может быть корректно минимизирована этим алгоритмом.

Окна на этом этапе сегментируются отдельно. Для этого применяется следующая эвристика. Окно считается либо целиком чистым, либо целиком загороженным в зависимости от количества пикселей по периметру, отсегментировавшихся как объекты переднего плана. Такой подход применяется из-за того, что окна могут сильно отличаться на разных ячейках, даже без наличия объектов переднего плана. Поэтому как сегментацию, так и восстановление текстуры, необходимо производить для всего окна целиком.



a) b)

Рис. 4: Результаты сегментации: a) $\lambda = 5$, b) $\lambda = 10$.

3.4 Восстановление текстуры под найденными объектами переднего плана

На этом шаге алгоритм восстанавливает текстуру фасада под найденными объектами переднего плана. Для этого используется найденная чистая ячейка, а также другие ячейки, оказавшиеся полностью чистыми после сегментации. Существует несколько подходов к решению задачи данного этапа.

Самый простой вариант – это заменить пиксели, отсегментированные как объекты переднего плана, на соответствующие им пиксели из чистой ячейки. Такой метод очень прост и быстро работает. Однако получающиеся результаты часто плохи. Это обусловлено тем, что чистая и целевая ячейки могут различаться по освещенности и точности разметки. Сходные проблемы возникают у всех методов, непосредственно копирующих пиксели из чистых ячеек.

Существует метод визуально естественной вставки части одного изображения в другое, который называется Poisson Image Editing [5]. Он основан на построении и решении уравнения Пуассона и учитывает перепады цвета во вставляемой части и краевые условия из целевого изображения. Этот метод можно непосредственно применять для вставки пикселей из чистой ячейки на места пикселей объектов переднего плана в других ячейках. Минусом данного метода является чувствительность к точности сегментации. Даже если один пиксель отсегментировался неправильно, он может войти в краевые условия, и Poisson Image Editing сработает не совсем кор-

ректно. Отчасти эту проблему можно решить сглаживанием краевых условий. Плюсом данного метода являются лучшие результаты (Рис. 5).

Рис. 5: Сравнение восстановления текстуры методом простой вставки и методом Poisson Image Editing.

4. РЕЗУЛЬТАТЫ

Данный метод был протестирован на различных зданиях г. Москвы и г. Кириши и показал неплохие результаты (Рис. 6c). Другим его достоинством является интерактивное время работы, а также предоставление пользователю возможности пошагово следить за ходом работы и поправлять результаты каждого шага по необходимости. Время работы предложенного метода составило в среднем 20 – 30 секунд в полностью автоматическом режиме для одного изображения 1400x1200 при реализации на C++ на компьютере Intel Celeron 1.5 GHz, 512 Mb RAM. С пользовательской коррекцией общее время обработки одного изображения возрастает до одной минуты. Для сравнения, реализованный на Matlab алгоритм [3] работает несколько часов, не считая ручного выделения объектов переднего плана. Метод из статьи [6] схож с предложенным в целом, но подходит для более узкого класса текстур. Сравнительное тестирование невозможно ввиду отсутствия доступной реализации метода, а также отсутствия данных по скорости работы в самой статье.

5. ЗАКЛЮЧЕНИЕ

В данной статье предложен алгоритм очищения текстур фасадов зданий от загораживающих их объектов переднего плана. Он может работать как в полностью автоматическом режиме, так и в интерактивном с минимальным пользовательским взаимодействием. Алгоритм показал хорошие результаты по качеству итоговой текстуры и по времени работы. В будущем планируется развить предложенный метод, расширив его применение на случай отсутствия абсолютно чистых ячеек. Также планируется провести корректное сравнение метода с [6].

6. БЛАГОДАРНОСТИ

Работа была выполнена по заказу компании "Медиа Софт Интегро" для Платформы CITY-3D и при частичной поддержке грантов РФФИ 08-01-00883-а, 09-01-92470-МНКС_а, и гранта Фонда "Научный потенциал" (Договор №184 от 14 января 2009 года).

7. ЛИТЕРАТУРА

[1] M. Bertalmio, G. Sapiro, V. Caselles, and C. Ballester, *Image Inpainting*, International Conference on Computer Graphics and Interactive Techniques, pp. 417–424, 2000

[2] M. Park, R. Collins, Y. Liu, *Deformed Lattice Discovery Via Efficient Mean-Shift Belief Propagation*, Proc. of the 10th European Conference on Computer Vision (ECCV), Part II, pp. 474–485, 2009

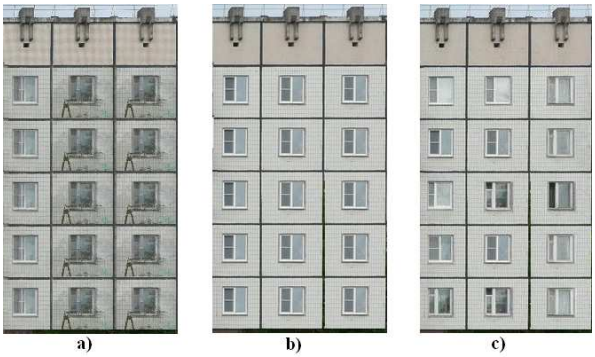


Рис. 6: a) Результат замены всех ячеек медианой; b) Результат замены всех ячеек абсолютно чистой; c) Результат работы предложенного алгоритма

[3] A. Criminisi, P. Perez, K. Toyama, *Object removal by exemplar-based inpainting*, In Conf. Computer Vision and Pattern Recog (CVPR) 2003, vol. 2, pp. 721–728, 2003

[4] Y. Wexler, E. Shechtman, M. Irani. *Space-Time Completion of Video*, IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), vol. 2, pp. 463–476, 2007

[5] P. Perez, M. Gangnet, A. Blake. *Poisson Image Editing*, ACM Transactions on Graphics (TOG), vol. 22, n. 3, 2003

[6] T. Korah, C. Rasmussen, *Analysis of Building Textures for Reconstructing Partially Occluded Facades*, Proc. of the 10th European Conference on Computer Vision (ECCV), Part I, pp. 359–372, 2008

[7] Y.Boykov, O. Veksler, R. Zabih, *Fast approximate energy minimization via graph cuts*, IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI), 23 (11): 1222-1239, 2001

[8] Y. Boykov, V. Kolmogorov, *An Experimental Comparison of Min-Cut/Max-Flow Algorithms for Energy Minimization in Computer Vision*, In Third International Workshop on Energy Minimization Methods in Computer Vision and Pattern Recognition, 2001

[9] V. Konushin, V. Vezhnevets, *Automatic building texture completion*, Proc. of Graphicon'2007, pp. 174-177, 2007

ОБ АВТОРАХ

Владимир Кононов является студентом Лаборатории компьютерной графики и мультимедиа Факультета вычислительной математики и кибернетики Московского государственного университета им. М. В. Ломоносова. Вадим Конушин и Антон Якубенко являются аспирантами той же лаборатории. Антон Конушин является сотрудником той же лаборатории.

ABSTRACT

Recently, 3D reconstruction of urban scenes has received much attention. One of its important problems is the creation of realistic facade textures from photos. Unfortunately, buildings facades are often partially occluded by various foreground objects such as trees, road signs, wires, etc. Therefore image completion is required. Modern image completion methods are too slow and not suitable for real applications. In this paper, we propose an interactive algorithm for foreground objects segmentation and occluded texture reconstruction, based on facade structure.