ИТЕРАТИВНАЯ АППРОКСИМАЦИЯ ПОСЛЕДОВАТЕЛЬНОСТЕЙ ПО МАКСИМУМУ ПЕРИМЕТРА И С ИСПОЛЬЗОВАНИЕМ НЕРАВЕНСТВА ТРЕУГОЛЬНИКОВ

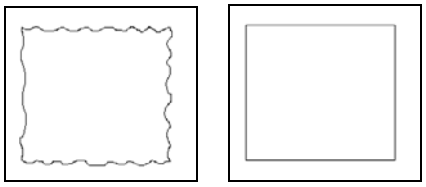
Р.В. Хмелев
Институт систем обработки изображений РАН,
Самарский государственный аэрокосмический университет

Аннотация

В статье описываются три авторских алгоритма итеративной аппроксимации, в первую очередь, итеративной полигональной аппроксимации контурных цепей, основанных на принципе максимума периметра аппроксимирующего многоугольника. Алгоритмы предназначены для анализа формы контуров. Также представлена модификация, позволяющая применять эти алгоритмы для итеративной аппроксимации одномерных последовательностей, могущая использоваться при анализе звука и растра изображений.

Введение

Большинство объектов реального мира можно отнести к каким-либо классам, причем объекты внутри класса различаются мелкими деталями структуры, а объекты разных классов – крупными. Таким образом, чтобы классифицировать объекты, необходимо найти их упрощенное представление, состоящее из одних крупных деталей (рис. 1).



а) *б)*

Рис. 1. а) Прямоугольник с изломанной границей, б) идеальный прямоугольник. Чтобы идентифицировать фигуру (а) как прямоугольник, нужно найти способ удалить мелкие несущественные детали.

Один из способов упрощенного представления плоского объекта со сложным контуром состоит в использовании особых точек контура и связей между этими точками, в частности, угловыми. На сегодняшний день известно несколько алгоритмов поиска углов в замкнутых контурах [1], к недостаткам этих алгоритмов можно отнести сложности настройки параметров, а также неискоренимые ошибки типа пропуска углов или обнаружения ложных углов. Вариация параметров увеличивает количество ошибок либо первого, либо второго рода.

В [1] говорится о том, что, несмотря на кажущуюся очевидность понятия угла, нет общепринятого математического определения угла для контуров растровых изображений. Кроме того, некоторые объекты реального мира вообще не содержат ярко выраженных углов. Поэтому при построении упрощенного представления контура предпочтительнее принципиально отказаться от поиска *углов*, и искать вместо них некие *узлы аппроксимации*, наиболее важные для описания формы контура, а будут эти узлы с точки зрения человека похожи на углы или нет – неважно. При соединении этих узлов отрезками получается аппроксимация контура многоугольником с числом сторон, меньшим, чем исходное число точек контура.

Александр Колесников в [2] приводит обзор алгоритмов полигональной аппроксимации контуров, разрабатывавшихся в течение последних 30 лет. Все эти алгоритмы решали одну из двух задач – минимизации ошибки аппроксимации при фиксированном числе узлов (min-ε задача) или минимизации числа узлов при заданной максимально допустимой ошибке (min-# задача).

Колесников насчитывает более сотни алгоритмов, посвященных этой теме, и он же отмечает, что такое их количество говорит скорее об их низком качестве и/или вычислительной эффективности, что оставляет возможности для улучшения. Представляется, что это связано главным образом с неудачным выбором критерия оптимизации, который, однако, был очевиден, и который можно было легко посчитать, а именно, минимизации отклонения.

Требовалось найти другой критерий, который бы более соответствовал субъективному человеческому восприятию. В данной работе в качестве такого критерия предлагается критерий максимально возможного периметра. На этом критерии основано три авторских алгоритма.

Работа, использующая этот критерий, как минимум, однажды уже публиковалась (Vallone, [3]), однако ее текст найти не удалось, из упоминания у Колесникова [2] известно, что в ней использовался метод оптимизации колонии муравьев (ACO – Ant Colony Optimization) [4, 5]. Первый из алгоритмов, представленных в данной работе, использует подход, похожий на упрощенную схему ACO.

Второй алгоритм (итеративного удаления), похож на ранее известный алгоритм слияния, который открывался, как минимум, девять раз [6, 7, 8, 9, 10, 11, 12, 13, 14], и который отличается от предлагаемого тем, что использует не идеологию максимизации периметра, а идеологию минимизации отклонения аппроксимации. Далее будет показано, чем такой подход принципиально хуже.

Третий предлагаемый алгоритм, который также использует итеративное удаление точек, отличается от всех известных тем, что на каждой итерации удаляется сразу много точек, что делает его самым быстрым.

Разработанные алгоритмы относятся к классу алгоритмов итеративной аппроксимации, поскольку предполагают последовательное добавление узлов

аппроксимации до получения удовлетворительного описания последовательности (рис. 2).

Однако притом, что алгоритмы создания последовательности добавления узлов были успешно разработаны, на вопрос о критериях удовлетворительности аппроксимации для описания последовательности однозначного ответа найти не удалось. В критериях удовлетворительности слишком много зависит от контекста задачи.

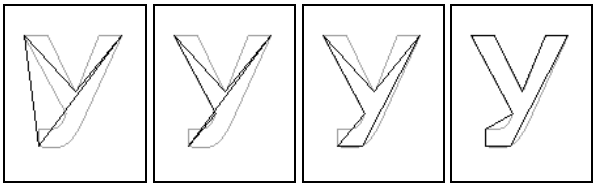


Рис. 2. Итеративная аппроксимация контура буквы «У». Слева направо: аппроксимация контура с помощью 4, 5, 6, и 9 узлов

Когда эти алгоритмы были разработаны и успешно применены для анализа контурных цепей, возникла идея разработать подобные алгоритмы для одномерных последовательностей типа звуковых волн или строк и столбцов растра изображений, чтобы получать упрощенные описания этих последовательностей меньшим числом отсчетов, но в то же время с минимальными потерями информации. С помощью небольшой модификации алгоритмы для одномерных последовательностей также были разработаны, и могут применяться при сжатии и анализе звука и изображений.

Главный недостаток предлагаемых алгоритмов – эмпиризм, то, как и почему они работают, проще проиллюстрировать с помощью примеров, чем строго доказать математически. Рассмотрим эти алгоритмы в порядке разработки (который совпадает с порядком уменьшения вычислительной сложности), чтобы проще было понять, откуда взялись те или иные идеи и решения. Сначала рассмотрим алгоритмы для контурных цепей, а затем их модификацию для одномерных последовательностей.

Далее «*N*» означает количество точек в исходном контуре или последовательности, а в заголовках алгоритмов указывается их вычислительная сложность в зависимости от *N*.

Принцип максимизации периметра аппроксимирующего многоугольника

Прежде чем описывать собственно алгоритмы, начнем с идеи, лежащей в их основе. Пусть есть исходный многоугольник *P*, состоящий из *N* вершин: $P = \{P_i\}_{i=0}^{N-1}$. Стороны этого многоугольника образуют контур, обозначим его C_p. Пусть также есть другой многоугольник *A* с таким же числом вершин $A = \{A_i\}_{i=0}^{N-1}$, которые, во–первых, лежат на контуре C_p, во–вторых, не фиксированы жестко, и могут свободно перемещаться по этому контуру, но с соблюдением следующих ограничений – они не должны совпадать друг с другом, не должны нарушать порядок следования друг за другом с точки зрения обхода контура C_p и никакие три точки не должны лежать на одной стороне (нет вырожденных углов) (рис. 3).

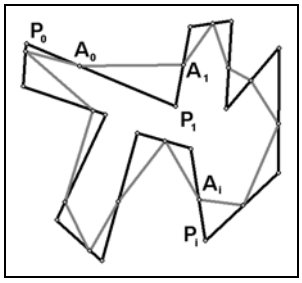


Рис. 3. Исходный многоугольник P и многоугольник A с таким же числом вершин. Вершины A лежат на сторонах многоугольника P и их положение может варьироваться

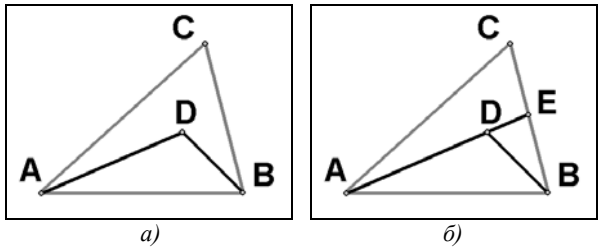
Задача состоит в том, чтобы путем вариации положения вершин A_i сделать периметр многоугольника A максимальным.

Лемма 1. Периметр многоугольника *A* будет максимально возможным только в том случае, если его вершины совпадают с вершинами исходного многоугольника *P*.

Чтобы доказать это утверждение, требуется сначала доказать несколько вспомогательных утверждений.

Лемма 2. Пусть есть произвольный треугольник *ABC* и точка *D*, принадлежащая области, ограниченной этим треугольником, положение которой может варьироваться (рис. 4а). Требуется найти такое положение точки *D*, при котором сумма расстояний *AD*+*BD* будет максимальной. Следует доказать, что при любом положении точки *D* внутри треугольника, не совпадающем с точкой *C*, выполняется следующее неравенство:

$$AD + BD < AC + BC\ .$$



а) *б)*

Рис. 4. Иллюстрации к лемме 2

Доказательство. Если точка *D* лежит на стороне *AB*, то доказательство тривиально. Предположим, что точка *D* является внутренней точкой треугольника (не лежит ни на одной из сторон). Продолжим отрезок AD до пересечения со стороной *BC*, а точку пересечения обозначим *E* (рис. 4б).

Очевидно, что *AD* < *AE*. В силу неравенства треугольников *BD* < *DE* + *BE*.

$$\underbrace{AE}_{AD+DE} + BE = AD + \underbrace{DE + BE}_{>BD} > AD + BD\ .$$

Таким образом, для любой внутренней точки *D* существует точка *E* на стороне треугольника *BC*, такая, что *AD* + *BD* < *AE* + *BE*. При этом точка *E* не мо-

жет совпадать с точкой *C*, это могло бы быть, только если бы точка *D* лежала на стороне *AC*, но по предположению, точка *D* не лежит ни на одной из сторон.

Очевидно, что $BE < BC$. В силу неравенства треугольников $AE < AC + CE$.

$$AC + \underbrace{BC}_{CE+BE} = \underbrace{AC + CE}_{> AE} + BE > AE + BE .$$

Таким образом, $AD + BD < AE + BE < AC + BC$, притом, что *D* – произвольная внутренняя точка треугольника, а *E* – точка на стороне треугольника, не равная *C*. Доказано, что для любой точки *X*, принадлежащей треугольнику и не совпадающей с *C*, выполняется:

$$AX + BX < AC + BC .$$

Лемма 3. Пусть есть две точки *A* и *B*, и отрезок *CD*, на котором лежит варьируемая точка *E* (рис. 5а). Требуется найти такое положение точки *E*, при котором достигается максимум выражения *AE+BE*. Следует доказать, что максимум этого выражения может достигаться только при *E = C* и/или *E = D*.

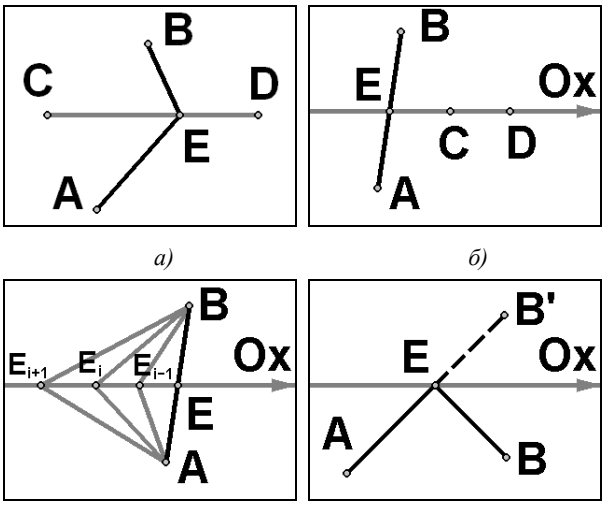


Рис. 5. Иллюстрации к доказательству леммы 3

Доказательство. В вырожденном случае, когда все четыре точки лежат на одной прямой, доказательство тривиально. Рассмотрим прямую, проходящую через отрезок *CD*, как ось координат *Ox*, и временно разрешим точке E варьироваться вдоль всей этой прямой, а не только в пределах отрезка *CD*. Достаточно будет доказать, при любом возможном взаимном расположении точек *A*, *B*, *C* и *D* величина *AE+BE* имеет лишь один локальный экстремум – минимум, соответственно, чтобы найти максимум при *E*, принадлежащем *CD*, достаточно проверить концы отрезка.

Рассмотрим случай, когда точки *A* и *B* лежат по разные стороны от оси *Ox* (рис. 5б).

Проведем через них отрезок, пересекающий ось *Ox*, а точку пересечения обозначим *E*. Очевидно, что в этой точке достигается глобальный минимум величины *AE+BE*. Докажем что при монотонном смещении точки E в одну сторону (влево или вправо) по оси Ox величина *AE+BE* также монотонно увеличивается, т.е., нет никаких других локальных минимумов.

Рассмотрим движение точки *E* влево (рис. 5в). При этом последовательно образуются треугольники ABE_i, каждый следующий треугольник ABE_{i+1} содержит в качестве внутренней точку предыдущего треугольника E_i. Согласно предыдущей лемме

$$AE_{i+1} + BE_i + 1 > AE_i + BE_i$$

Аналогично при движении точки *E* вправо. Таким образом, доказано что других локальных экстремумов нет.

Теперь рассмотрим случай, когда точки *A* и *B* лежат по одну сторону от оси *Ox* (рис. 5г). Построим для точки B точку *B'*, симметричную относительно оси *Ox*. Для любой точки *E* на оси *Ox* расстояние *BE* равно *B'E*. Доказательство единственности локального экстремума сводится к случаю, когда точки *A* и *B* лежат по разные стороны от оси *Ox*.

Таким образом, доказано, что максимум величины *AE+BE* при $E \in CD$ может достигаться лишь на концах отрезка *CD*.

Лемма 4. Пусть есть некоторая ломаная, определяемая N точками: $P_0, \ldots, P_{N-1}$ ($N \geq 3$) и точка *A*, положение которой может варьироваться вдоль всей ломаной (рис. 6). Следует доказать, что максимум суммы расстояний $P_0A + P_{N-1}A$ может достигаться только в вершинах ломаной $P_1, \ldots P_{N-2}$, но не в промежуточных точках отрезков, составляющих ломаную.

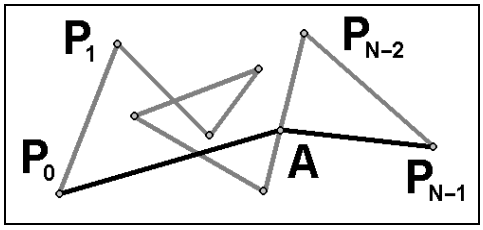
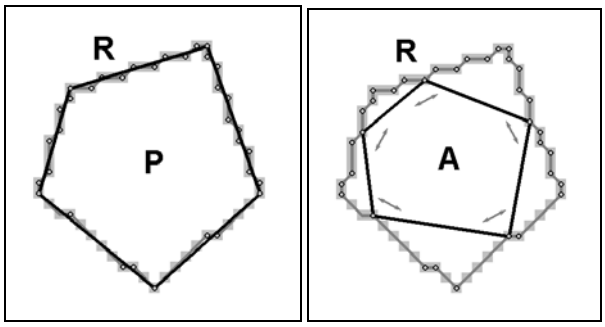


Рис. 6. Иллюстрации к доказательству леммы 4

Доказательство. Поиск максимума сводится к поиску максимума в каждом отрезке, составляющем ломаную, по предыдущей лемме в каждом отрезке максимум может достигаться только на его концах, что и следовало доказать.

Вернемся теперь к лемме (1). Пусть для многоугольника $P = \{P_i\}_{i=0}^{N-1}$ найден аппроксимирующий многоугольник с максимально возможным периметром $A = \{A_i\}_{i=0}^{N-1}$. Возьмем одну из его вершин A_i и рассмотрим сумму длин сторон $A_{i-1}A_i + A_iA_{i+1}$ (индексы вершин берутся циклическими, в данном случае это означает «предыдущая и следующая вершины»). Поскольку периметр многоугольника *A* является максимально возможным, то и эта сумма является максимально возможной при фиксированных вершинах A_{i-1} и A_{i+1} – если бы это было не так, то вариацией вершины A_i можно было бы добиться увеличения длины периметра многоугольника *A*, но он уже является максимальным. Следовательно, по предыдущей лемме, вершина A_i совпадает с одной из вершин многоугольника *P*. Поскольку A_i – произвольно взятая вершина, следовательно, все вершины *A* совпадают с вершинами *P* (конец доказательства).

Теперь, пусть существует многоугольник P, состоящий из M вершин, и многоугольник R, состоящий из N вершин и являющийся растровым представлением многоугольника P ($M<N$) (рис. 7а).



а) *б)*

Рис. 7. а) Многоугольник P и его растровое представление, многоугольник R. б) Поиск вершин исходного многоугольника P в его растровом представлении R с помощью вариации вершин аппроксимирующего многоугольника A, для которого следует добиться максимально возможного периметра

Пусть известно лишь растровое преставление и количество вершин исходного многоугольника P, но не их расположение. Попробуем найти эти вершины в многоугольнике R, используя M-угольник A с варьируемыми вершинами (рис. 7б). Найдем такой вариант расположения вершин A, при котором его периметр будет максимальным. По лемме 4, вершины A следует варьировать не непрерывно, а в пределах конечного множества вершин R, поэтому количество вариантов для перебора ограничено.

С помощью такой процедуры можно найти *правдоподобное* положение вершин исходного M-угольника P. Именно правдоподобное, а не точное, поскольку в лемме (1) P и A – M-угольники, а растровое представление R имеет не M, а N вершин.

На практике, для растрового представления контура, как правило, нет априорной информации об объекте, по которому это растровое представление получено, в частности, исходное количество углов и были ли углы вообще. Можно лишь попробовать различные варианты и выбрать тот, который более всего удовлетворит по каким-то критериям.

Наилучшей аппроксимацией из M узлов для N-угольника ($M<N$) будем считать M-угольник, обладающий наибольшим возможным периметром, при условии, что его вершины совпадают с вершинами N-угольника.

Чтобы решить, сколько узлов достаточно для удовлетворительной аппроксимации N-угольника, нужно построить все наилучшие его аппроксимации с числом вершин от 2 до N, поскольку может оказаться, что единственно удовлетворительной аппроксимацией N-угольника является сам N-угольник, а все многоугольники с меньшим числом вершин являются неудовлетворительными. Для построения всех возможных наилучших аппроксимаций был разработан следующий алгоритм.

Алгоритм максимального периметра ($O(N^4)$)

Самое простое, что можно использовать – полный перебор. Чтобы найти аппроксимацию из M узлов ($2\le M\le N$-1) с максимальным периметром, нужно перебрать все возможные сочетания из N по M:

$$C_N^M = \frac{N!}{M!(N-M)!},$$

при этом проверка каждого сочетания из M узлов состоит в нахождении суммы длин расстояний между узлами. Поскольку одни и те же расстояния используются по много раз, их можно посчитать сразу и занести в таблицу размером $N(N\text{-}1)/2$. Таким образом, число сложений для нахождения лучшей аппроксимации из M узлов будет порядка

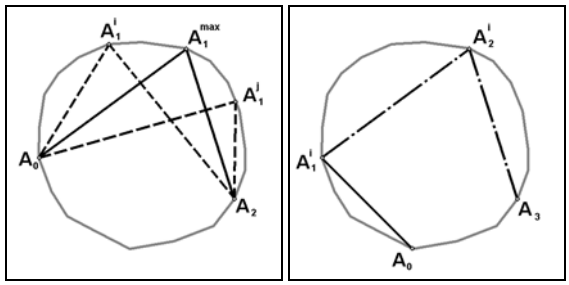
$$S_N^M = NC_N^M = \frac{N!}{(M\text{-}1)!(N-M)!},$$

а суммарное число сложений при поиске аппроксимаций со всеми возможными количествами узлов при этом будет

$$S_N = \sum_{M=2}^{N-1} S_N^M \ .$$

Можно уменьшить вычислительную сложность, если при увеличении количества используемых узлов аппроксимации использовать ранее полученные данные.

Для начала рассмотрим, как происходит поиск лучшей аппроксимации из трех узлов (аппроксимация из двух узлов ищется тривиально как две наиболее удаленные друг от друга точки контура). Требуется проверить все возможные сочетания из трех узлов. Каждое сочетание образуется следующим образом: фиксируем первый и последний узлы, а средний варьируем, чтобы найти максимум периметра треугольника (рис. 8а).



а) *б)*

Рис. 8. а) Поиск наилучшей аппроксимации треугольником, проверка сочетания начального узла A_0 и конечного узла A_2. При этом варьируется средний узел A_1, и при A_1^{max} находится максимально возможная длина ломаной $A_0A_1A_2$. Лучшие ломаные из двух звеньев находятся для всех возможных сочетаний начального и конечного узла.

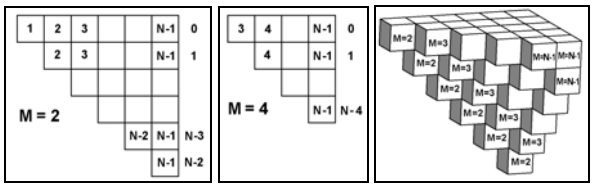
б) Поиск наилучшей аппроксимации четырехугольником, проверка сочетания начального узла A_0 и конечного узла A_3. При этом варьируется только второй узел A_1, поскольку для любого узла уже известна лучшая ломаная из двух звеньев (штрихпунктирная линия), соединяющую его с точкой A_3.

Побочным результатом этого является то, что для каждого сочетания начального и конечного узла находится наиболее длинная ломаная из двух отрезков, соединяющая эти начальный и конечный узлы. Будем называть эту ломаную *лучшей* ломаной из двух звеньев для данных начального и конечного узлов.

На следующей итерации, когда ищется наилучшая аппроксимация четырехугольником, можно использовать лучшие ломаные, найденные ранее при поиске наилучшей аппроксимации треугольником. При этом также фиксируется сочетание начального и конечного узлов и варьируется только второй узел, поскольку уже известна лучшая ломаная из двух звеньев, соединяющая этот второй узел с конечным (рис. 8б).

Таким образом, при увеличении количества узлов аппроксимации используются лучшие ломаные, полученные на предыдущей итерации. Для каждого сочетания начального и конечного узла поиск новой лучшей ломаной имеет линейную сложность, благодаря вариации только одного промежуточного узла.

При увеличении количества узлов, участвующих в аппроксимации, на единицу, количество возможных сочетаний начального и конечного узлов уменьшается также на единицу, поскольку при M узлов аппроксимации индекс конечного узла не может быть ближе M-1 позиций по отношению к начальному (рис. 9а, б).



а) *б)* *в)*

Рис. 9. а) Таблица лучших ломаных в вырожденном случае (M=2, ломаная превращается в отрезок), справа от строк указан начальный узел, в ячейках строки – возможные конечные узлы.
б) Таблица лучших ломаных при аппроксимации четырехугольником (M=4).
в) Совокупность таблиц лучших ломаных для каждого возможного количества узлов аппроксимации образует пирамидальную трехмерную таблицу.

Из-за того, что необходимо сохранять лучшие ломаные предыдущей итерации, образуется трехмерная таблица лучших ломаных, имеющая пирамидальную форму (рис. 9в), каждая ломаная уровня M рекурсивно ссылается на ломаную на уровне M-1. Количество элементов в таблице по формуле объема пирамиды можно оценить как $N^3/3$. Притом, что сложность нахождения каждой лучшей ломаной в среднем линейная, суммарная сложность заполнения всех ячеек таблицы оценивается как $O(N^4)$.

Как показали тесты, данный алгоритм дает *самую красивую* аппроксимацию заданным числом узлов, даже если явно выраженных углов в контуре нет, в частности, для растровой окружности получаются почти правильные многоугольники (насколько это возможно при использовании одних только точек растрового контура окружности) (рис. 10).

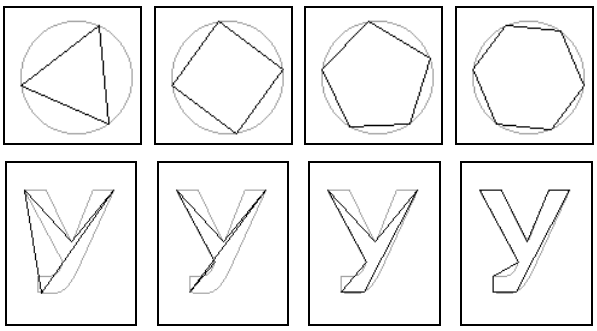


Рис. 10. Примеры работы алгоритма максимального периметра. Вверху – контур окружности, аппроксимируемой 3, 4, 5 и 6 узлами. Внизу – контур буквы «У», аппроксимируемый 4, 5, 6 и 9 узлами.

Однако из-за огромной вычислительной сложности и больших требований к памяти, этот алгоритм мало пригоден для работы в реальном времени, поэтому в дальнейшем были разработаны другие алгоритмы.

Алгоритм итеративного удаления (O(N log N))

Как говорит название, в данном алгоритме происходит последовательное удаление вершин исходного N-угольника в специальном порядке. Как и предыдущий, данный алгоритм использует идею максимально возможного периметра аппроксимирующего многоугольника, точнее, порожденную ей идею о *минимизации потерь* в длине периметра при удалении вершин исходного N-угольника.

Введем понятие: *важностью* узла будем называть величину уменьшения периметра многоугольника при замене этого узла и двух отрезков, соединяющих его с другими узлами, одним отрезком. Важность узла выражается через неравенство треугольника (рис. 11).

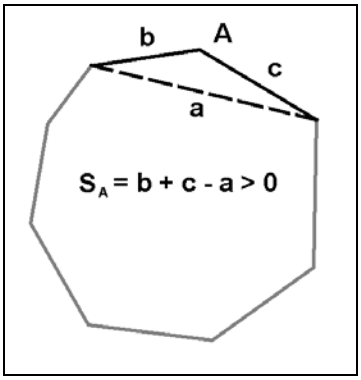


Рис. 11. Важность узла A (S_A) выражается через неравенство треугольника.

$$S_A = b + c - a > 0 . \quad (1)$$

Структурно алгоритм итеративного удаления довольно прост и состоит из трех пунктов.

1. В исходном многоугольнике для каждой вершины находится важность узла.

2. Среди всех узлов находится самый маловажный и удаляется, при этом нужно обновить значения важности в двух вершинах, соседних с удаленной.

3. Возвращаемся к пункту 2 и повторяем, пока не останутся две вершины (рис. 12).

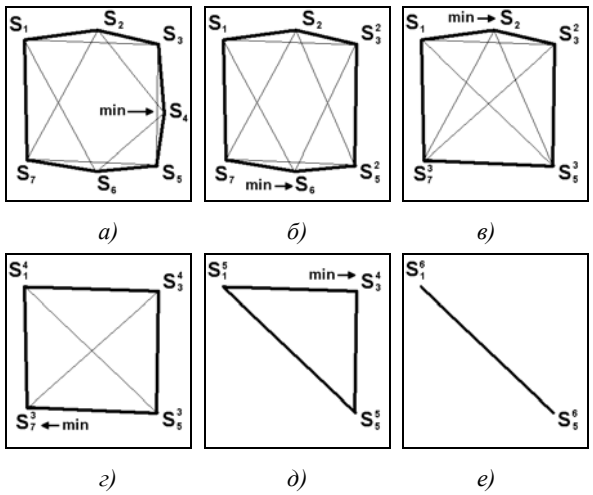


Рис. 12. Пример работы алгоритма итеративного удаления

Таким образом, строится обратная последовательность аппроксимирующих многоугольников с уменьшающимся числом вершин. На каждой итерации требуется находить максимально быстро самый маловажный узел, и, поскольку список узлов является динамическим, в алгоритме используются сбалансированные бинарные деревья. В бинарных деревьях операции с одним узлом имеют сложность в среднем $O(\log N)$, поэтому общая сложность алгоритма $O(N \log N)$.

Как показывают тесты, алгоритм хорошо подходит для построения аппроксимаций контуров с заметными человеческому глазу углами, даже если углы скруглены или срезаны. Связано это с тем, что в окрестностях даже скругленных углов локализовано много узлов, которые при начальном нахождении значения важности являются сравнительно важными и удаляются в последнюю очередь (рис. 13).

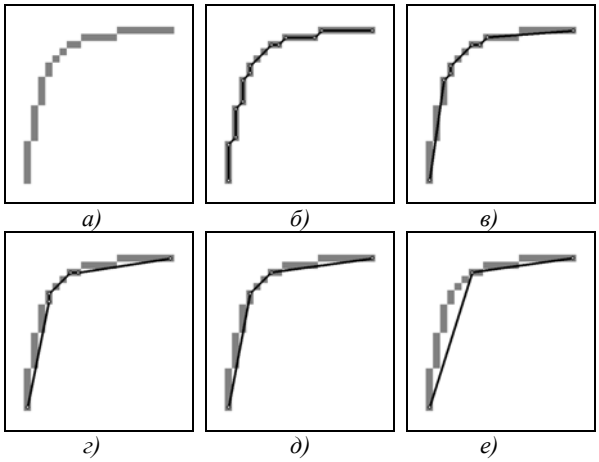


Рис. 13. Данная иллюстрация приведена в качестве наглядного объяснения того, как алгоритм выделяет скругленные углы. а) Исходный фрагмент контура. б)–е) Этапы удаления узлов (каждая следующая картинка соответствует нескольким итерациям удаления)

Традиционный алгоритм слияния [6, 7, 8, 9, 10, 11, 12, 13, 14], использующий в качестве критерия удаления минимум отклонения, в данном примере срезал бы угол в первую очередь, поскольку на первых итерациях при этом достигалось бы минимальное увеличение погрешности аппроксимации.

В качестве реального примера работы с контурами, содержащими выраженные углы можно сравнить рис. 2 (на самом деле он иллюстрирует работу алгоритма итеративного удаления) и рис. 10 (алгоритм максимального периметра) – для данного контура результаты работы двух алгоритмов идентичны при числе узлов 5, 6 и 9. Для четырех узлов результаты немного различаются. При тестировании алгоритмов выяснилось, что алгоритм итеративного удаления подчеркивает углы лучше, однако проигрывает при аппроксимации контуров, не содержащих выраженных углов, в частности, для контура окружности (рис. 14), однако аппроксимация окружности малым числом узлов не представляет практического интереса при распознавании. Интерес представляет то, что при итеративном удалении растровая окружность очень быстро превращается в выпуклый многоугольник.

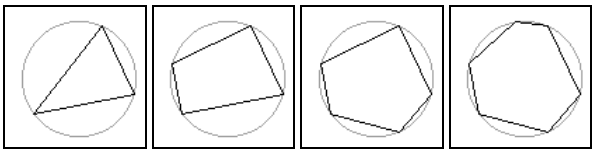


Рис. 14. Результаты работы алгоритма итеративного удаления для растровой окружности при числе узлов аппроксимации 3, 4, 5, 6

В принципе, алгоритм с вычислительной сложностью $O(N \log N)$ уже пригоден для обработки в реальном времени, однако работа со сбалансированными бинарными деревьями вычислительно тяжела – хотя операции с деревьями и имеют логарифмический порядок сложности, каждая итерация балансировки дерева довольно громоздка. Поэтому поиск более быстрого алгоритма был продолжен и увенчался успехом.

Алгоритм удаления локальных минимумов (O(N) – O(N log N))

В предыдущем алгоритме на каждой итерации удаляется узел, являющийся глобальным минимумом важности для данного контура. Затем удаляется следующий глобальный минимум, который может оказаться как соседним узлом, так и узлом в противоположной части контура и т.д. Следует обратить внимание на то, что до некоторой итерации фрагменты контура не влияют друг на друга (рис. 15).

Это свойство навело на мысль о том, что на каждой итерации можно удалять не один глобальный минимум, а сразу все локальные минимумы, при этом качество аппроксимации не должно сильно пострадать.

Далее следует описание алгоритма удаления локальных минимумов, при этом нужно иметь в виду, что, поскольку контур циклический, то выражение вида «узел с индексом ***i*+1**» означает «следующий по циклу» и т.п.

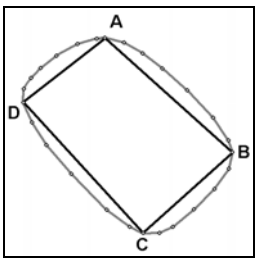


Рис. 15. До того, как исходный контур при последовательном удалении узлов превратится в четырехугольник ABCD, будут последовательно удалены узлы внутри фрагментов AB, BC, CD и AD, при этом внутренние узлы каждого фрагмента не влияют на внутренние узлы соседних фрагментов

1. Как и в алгоритме итеративного удаления, в первую очередь находится значение важности всех узлов в исходном *N*-угольнике.

2. Находятся все узлы, являющиеся локальными минимумами важности, при этом используется нестрогое неравенство:

$$S_{i-1} \geq S_i \geq S_{i+1}.$$

В случае равенства $S_i = S_{i+1}$ узел с индексом ***i*+1** не проверяется по этой формуле (чтобы не было подряд идущих минимумов).

3. Все локальные минимумы удаляются. Для каждого удаляемого узла ***i*** в двух соседних узлах **(*i*-1)** и **(*i*+1)** обновляются значения важности (могут как увеличиться, так и уменьшиться), поэтому после удаления узла ***i*** необходимо проверить на локальный минимум четыре соседних узла или меньше, если узлы с индексами ***i*-2** и/или ***i*+2** также являются локальными минимумами, удаляемыми на этой итерации (рис. 16).

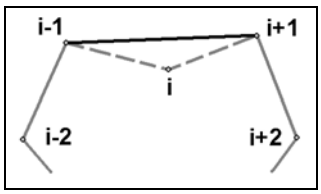


*Рис. 16. При удалении **i**-го узла меняются значения важности в узлах **i-1** и **i+1**. Перед следующей итерацией удаления следует проверить на локальный минимум не более четырех узлов: **i-2**, **i-1**, **i+1** и **i+2***

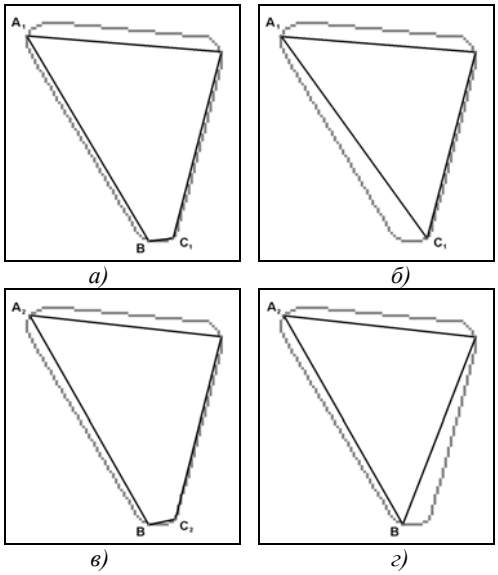
Все четверки узлов, соседних с удаленными минимумами, заносятся в список потенциальных новых локальных минимумов.

4. Возвращаемся к пункту 2 и продолжаем находить и удалять локальные минимумы до тех пор, прока не останутся два узла.

Поскольку в данном алгоритме на каждой итерации удаляется сразу много узлов, необходимо установить порядок добавления узлов в итеративную аппроксимацию. Поэтому для каждого узла исходного *N*-угольника находится максимум важности, достигавшийся в нем на какой-либо итерации, а затем узлы сортируются в порядке убывания максимума важности.

Сложность описанной выше фазы алгоритма – линейная ($O(N)$), кроме того, операции существенно менее сложны вычислительно, чем работа со сбалансированными бинарными деревьями. Сложность последующей сортировки узлов $O(N \log N)$, но вычислительная сложность сортировки при том же порядке также существенно ниже, чем у деревьев. Как показали тесты, программная реализация алгоритма удаления локальных минимумов работает примерно в 3-4 раза быстрее алгоритма итеративного удаления для последовательностей длиной 200-1000 узлов.

Что же касается результатов, то они несколько отличаются для двух алгоритмов, но обычно близки. Иногда при количестве узлов, недостаточном для адекватной аппроксимации, результаты могут заметно различаться. На рис. 17 показан пример фигуры, которая более-менее адекватно аппроксимируется четырехугольником и не вполне адекватно – треугольником.



а) *б)* *в)* *г)*

Рис. 17. Результаты аппроксимации фигуры двумя алгоритмами (четыре и три узла). Порядок удаления узлов различается в обоих алгоритмах, и, соответственно, различаются получаемые аппроксимации.
а,б) Алгоритм итеративного удаления.
в,г) Алгоритм удаления локальных минимумов

Из-за смещения точки *A* на два пиксела и точки *C* на один пиксел порядок добавления узлов *B* и *C* изменился (у них близкая максимальная важность). Поэтому по результатам работы двух алгоритмов аппроксимирующие четырехугольники выглядят очень похоже, а треугольники – заметно различаются.

Модификация алгоритмов для применения к одномерным последовательностям

Способность алгоритмов итеративной аппроксимации выделять точки, наиболее важные с точки зрения формы контура, натолкнула на мысль попробовать создать алгоритм, выделяющий самые важные точки одномерных последовательностей, например, оцифрованных звуковых волн, а также строк и столбцов растровых изображений. Для этого понадобилось лишь заменить геометрическую формулу важности узла.

Пусть *i, j, k* – индексы попарно связанных узлов одномерной последовательности *A*, *i*<*j*<*k*, нужно найти важность *j*-того, при условии, что известны значения A_i, A_j, A_k. Используем выражение, аналогичное ранее использовавшемуся неравенству треугольников, с той лишь разницей, что вместо длин отрезков в нем используются характеристики хорд, соединяющих узлы.

$$S_j = Hi_j + H_{jk} - H_{ik}, \qquad (2)$$

где $H_{mn} = \dfrac{(A_n - A_m)^2}{n-m}$, $m < n$.

Характеристика хорды *H* состоит из произведения разницы величин узлов на скорость изменения величины. Нетрудно доказать, что если три узла лежат на одной прямой (на графике последовательности), то важность среднего узла равна нулю, в противном случае она больше нуля, как и в геометрическом случае.

Одномерные последовательности, в отличие от контурных цепей, не являются циклическими, поэтому есть небольшая разница в программных реализациях алгоритмов, а именно, начальный и конечный узлы никогда не удаляются, формулу важности к ним применить невозможно, поэтому их важность считается бесконечно большой.

На практике для одномерных последовательностей были реализованы алгоритмы итеративного удаления и удаления локальных минимумов. Алгоритм, аналогичный алгоритму максимального периметра, реализовывать было признано нецелесообразным из-за больших размерностей последовательностей (в частности, звуковых) и огромных требований к памяти и вычислительной сложности алгоритма.

Более подробно об экспериментах с последовательностями, а именно, звуковыми и растровыми, будет сказано в следующем пункте.

Области применения алгоритмов итеративной аппроксимации

Далее приводится обзор возможных направлений, в которых, следует проводить исследования данных алгоритмов итеративной аппроксимации. Для некоторых задач приведены идеи, базовые схемы решения, которые еще не были глубоко исследованы. Тем не менее, поскольку нет уверенности, что у автора хватит времени и сил на все эти задачи, было решено опубликовать эти схемы в надежде, что ими заинтересуются другие исследователи.

Контурный анализ. Алгоритмы итеративной аппроксимации изначально предназначались в первую очередь для нахождения адекватных аппроксимаций контуров, и эта задача на данный момент не имеет удовлетворительного решения.

Аппроксимирующие многоугольники предполагалось использовать при сравнении контуров объектов, в этой задаче возможно даже необязательно находить лучшие аппроксимации контура и сравнивать несколько аппроксимаций с различным числом узлов.

Часто контуры лишь отдаленно напоминают многоугольники, и точнее их можно было бы охарактеризовать как фигуры с явно выраженными выпуклостями и вогнутостями (рис. 18).

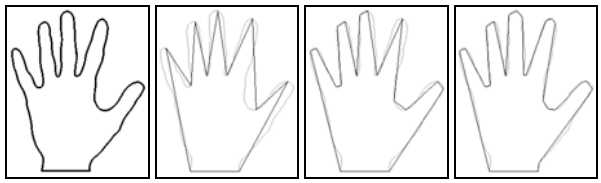


Рис. 18. Слева направо: контур руки и его аппроксимации из 24, 17 и 11 узлов.

Предположительно, для таких фигур удастся создать алгоритмы сравнения, анализирующие взаимное расположение выпуклых и вогнутых фрагментов в аппроксимирующих многоугольниках.

Выделение контуров на изображениях. Проблема выделения контуров является традиционной для цифровой обработки изображений, и методов было придумано множество.

С помощью итеративной аппроксимации можно предложить еще один подход. Поскольку итеративная аппроксимация применяется к одномерным последовательностям, а не к двумерным растрам, следует выделить на изображениях одномерные последовательности и найти в них самые характерные узлы. В качестве последовательностей проще всего использовать строки, столбцы и диагонали, но желательно также использовать несколько промежуточных направлений. С помощью такого подхода хорошо выделяются размытые контуры (рис. 19).

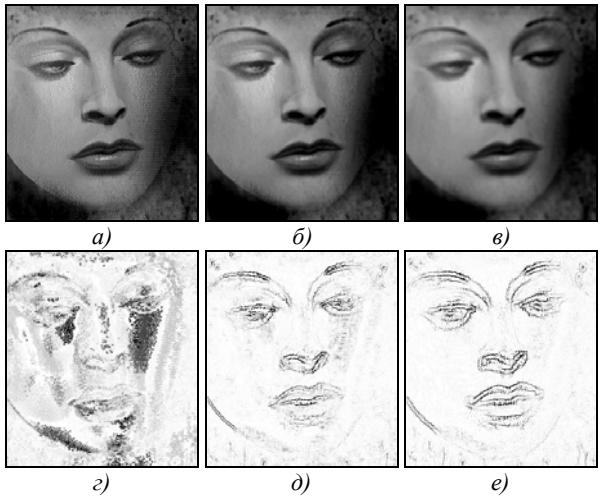


Рис. 19. Примеры применения итеративной аппроксимации для выделения контуров на изображении с различным сглаживанием. Сверху – исходные изображения, снизу – полутоновые контура, полученные с помощью итеративной аппроксимации по строкам, столбцам и двум диагоналям (в каждой точке максимум важности, выбранный из четырех вариантов аппроксимации).

Рис 19а, г - нет сглаживания. Исходное изображение очень зернисто, поэтому находится много контуров, на которые человек не обратил бы внимания.

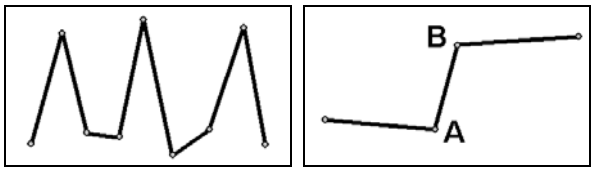
Рис.19 б, д - сглаживание квадратом 3x3. Зерна почти исчезли, как и соответствующие им контура. Следует обратить внимание на двойной дугооб-

разный контур в левой нижней части картинки, структура таких переходов показана на рис. 20б.

Рис 19 в, е) Сглаживание квадратом 5х5. Зерна еще менее заметны, характерные контуры лица еще более выделяются на общем фоне

Особенность восприятия человеком изображений состоит в том, что он может воспринимать их как очень детально, так и крупным планом, в частности, поле пикселов с мелким повторяющимся узором человек воспринимает как однотонное, а алгоритмы, как правило, находят там множество контуров (рис. 19а). Помочь в данном случае может поиск контуров на изображениях с разной степенью сглаживания (рис. 19б, в).

В программных реализациях алгоритмов итеративной аппроксимации запоминаются не только значения максимальной важности, которая когда-либо достигалась в узле последовательности, но и хорды, которые связывали данный узел с соседями в момент достижении этого максимума. Шумовые контуры обычно имеют близких соседей и с одной и с другой стороны (рис. 20а), в то время ,как контуры, интересные для анализа – близкого соседа с одной стороны и далекого – с другой (рис. 20б).



а) *б)*

Рис. 20. а) Типичная картина аппроксимаций шумовой последовательности. Каждый узел имеет близких соседей с обеих сторон.
б) В узлах А и В, соответствующих контурам объектов, как правило, одна хорда короткая, вторая – длинная

Повышение резкости изображений в наиболее важных узлах. Существующие алгоритмы повышения резкости изображений основаны, как правило, на высокочастотной фильтрации, применяемой ко всему изображению целиком, что приводит к повышению резкости как интересующих объектов, так и шумов. С помощью итеративной аппроксимаций предположительно можно создать алгоритм, повышающий резкость контуров избирательно, например, только в парах узлов, соответствующих рис. 20б. Повышение резкости будет осуществляться сдвигом таких узлов друг к другу (рис. 21).

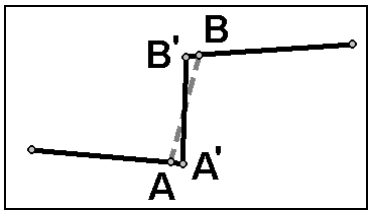


Рис. 21. Повышение резкости контуров путем сдвига контурных узлов друг к другу. А' и В' – новые положения узлов А и В

Сжатие изображений и звука с потерями и без потерь. Так же, как и для контуров, итеративная аппроксимация позволяет получать упрощенные представления одномерных последовательностей, в которых сохранены основные узловые точки (рис. 22). Самые важные узлы могут использоваться как в традиционном дифференциальном кодировании в качестве опорных узлов, так и в сжатии с потерями.

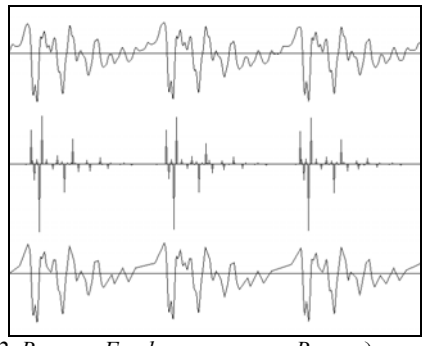


Рис. 22. Вверху: График звука «а». В середине: график важности узлов, положительные значения соответствуют узлам, выпуклым вверх, отрицательные – выпуклым вниз. Внизу: линейная интерполяция, построенная примерно по 20% самых важных узлов исходной последовательности

Приблизительная схема алгоритма дифференциального сжатия звуковой последовательности без потерь со сложностью $O(N \log N)$ может быть следующей:

Строится последовательность узлов каким-то из алгоритмов удаления, затем узлы сортируются в порядке убывания максимально достигавшейся в них важности.

Последовательность подвергается дифференциальному кодированию целиком с минимально возможным числом бит на отсчет.

Выбирается самый важный узел внутри последовательности, который разбивает последовательность на две части.

Для каждой части повторяются пункты 2, 3, 4, при этом на верхние уровни разбиения передается информация о минимальном достигнутом размере сжатых подпоследовательностей (порядка $\log N$ вложенных уровней), в результате на верхних уровнях, оценив степень сжатия в различных вариантах, можно решить, целесообразно ли деление данной подпоследовательности или экономичнее кодировать ее целиком.

Если научиться находить периоды в звуковых последовательностях (через сравнение сочетаний основных узлов), то можно увеличить степень сжатия, кодируя лишь изменения периодов типа сдвигов основных узлов и изменения их амплитуды.

Распознавание речи. При экспериментах с итеративной аппроксимацией звуковых последовательностей обнаружились интересные свойства одних и тех же гласных звуков, сказанных разными людьми, а именно, что их графики важности похожи по структуре, несмотря на то, что внешне кривые кажутся совсем непохожими. Рабочая гипотеза на данный момент состоит в том, что гласные звуки напоминают следующую модель:

$$f(t) = g(t)\,h(t),$$

где $g(t)$ – медленно меняющаяся периодическая функция модуляции с диапазоном значений [0,1];
$h(t)$ – быстро осциллирующая периодическая функция с периодом, укладывающимся в период $g(t)$ целое число раз.

На основе картины выраженности узлов удалось смоделировать синтетические звуки, похожие на гласные, произносимые человеком (рис. 23).

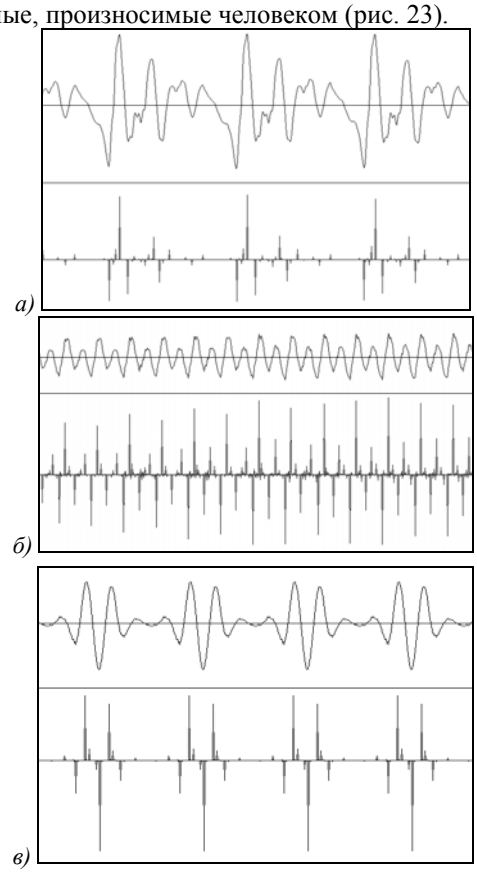


Рис. 23. На каждой картинке вверху – график звуковой последовательности, внизу – соответствующий ему график важности узлов. Все три последовательности звучат как «о».
а) Голос мужчины. б) Голос женщины. в) Синтетический

Есть надежда, что с помощью итеративной аппроксимации звуковых последовательностей удастся создать неспектральный алгоритм распознавания речи, не требующий настройки на голос человека.

Выделение характерных точек траектории движения объектов. Довольно часто человек распознает объекты окружающего мира не только по их форме, но и по характеру их движения, в частности, многих людей можно узнать по походке. Чтобы проанализировать характер движения, нужно найти некоторые особые точки объекта и изучить траекторию их движения.

При анализе характера этих траекторий может использоваться итеративная аппроксимация для поиска основных узловых точек траектории. При этом величина хорды, соединяющей два узла, должна определяться по схеме, похожей на ту, что использовалась при итеративной аппроксимации звука или растра изображений, а именно, как квадрат расстояния между узлами, деленный на время, за которое точка попадает из одного узла в другой.

Эту технологию можно также применять для сжатия данных о захвате движения в трехмерной анимации.

Заключение

Алгоритмы итеративной аппроксимации, рассмотренные в данной работе, являются сравнительно новыми и малоизученными (первые два датируются августом 2003, последний – мартом 2005), однако очевидно, что они имеют преимущество перед ранее известными алгоритмами, и нет сомнений, что они будут играть важную роль в обработке сигналов.

Благодарности

Работа выполнена при поддержке российско-американской программы "Фундаментальные исследования и высшее образование" (BRHE), грантов Президента России № НШ-1007.2003.01 и Российского фонда фундаментальных исследований № 04-07-96500.

Литература

1. Chetverikov D., Szabó Z. Detection of High Curvature Points in Planar Curves // http://visual.ipan.sztaki.hu/corner/
2. Kolesnikov A. Polygonal approximation, 2003
3. Vallone U., Bidemensional shapes polygonalization by ACO, Proc. 3rd Int. Workshop on Ants Algorithms, Brussels, Belgium, 296-297, September 2002.
4. Dorigo N., Optimization, Learning, and Natural Algorithms, PhD Thesis, Dip. Elettronica e Informazione, Politecnico di Milano, Italia, 1992.
5. Dorigo M., G. Di Caro & L. M. Gambardella. Ant Algorithms for Discrete Optimization. Artificial Life, 1999. Vol. 5(2). P.137-172.
6. Leu J.G., Chen L., Polygonal approximation of 2d shapes through boundary merging, Pattern Recognition Letters, 1988. Vol.7(4). P.231-238.
7. Latecki L.J., Lakamper R., Convexity rule for shape decomposition based on discrete contour evolution, Computer Vision and Image Understanding, 1999. Vol. 73. P.441-454,.
8. Ku K.M., Chiu K.M., Polygonal approximation of digital curve by graduate iterative merging, Electronics Letters, 1995. Vol. 31. P. 444-446,.
9. Fahn C.-S., Wang J.-F., Lee J.-Y., An adaptive reduction procedure for the piecewise linear approximation of digitized curves, IEEE Trans. Pattern Analysis and Machine Intelligence, 1989. Vol. 11. P. 967-973.
10. Wu J.-S., Leou J.-J., New polygonal approximation schemes for object shape representation, Pattern Recognition, 1993. Vol. 26. P. 471-484.
11. Boxer L., Chang C.-S., Miller R., Rau-Chaplin A., Polygonal approximation by boundary reduction, Pattern Recognition Letters, 1993. Vol.14. P.111-119.
12. Visvalingam M., Whyatt J., Line generalization by repeated elimination of points, Cartographic Journal, 1993. Vol. 30(1). P. 46-51.
13. Pikaz A., Dinstein I., An algorithm for polygonal approximation based on iterative point elimination, Pattern Recognition Letters, 1995. Vol. 16(6). P. 557-563.
14. Horst J.A., Beichl I., A simple algorithm for efficient piecewise linear approximation of space curves, Proc. Int. Conf. Image Processing-ICIP'97, 1997. Vol. 2. P. 744-747.

Iterative approximation of sequences along the maximum of the perimeter and using the triangle inequality

R.V. Khmelev[1,2]
[1]*Image Processing Systems Institute of RAS*
[2]*Samara State Aerospace University (SSAU)*

Abstract:

The paper describes three proprietary algorithms of iterative approximation, primarily, iterative polygonal approximation of contour chains based on the principle of the maximum perimeter of an approximating polygon. The algorithms are designed to analyze the shape of the contours. A modification is also presented that allows to use these algorithms for iterative approximation of one-dimensional sequences, and that can be used in the analysis of sound and image raster.

Keywords: iterative approximation, triangle inequality, maximum perimeter of an approximating polygon, image raster, sound.

Acknowledgments: This work was supported by the Russian-American program "Basic Research and Higher Education" (BRHE), grants of the President of Russia No. NSh-1007.2003.01 and the Russian Foundation for Basic Research No. 04-07-96500.

Citation: Khmelev RV. Iterative approximation of sequences along the maximum of the perimeter and using the triangle inequality. Computer Optics 2005; 27: 155-164.

References:

[1] Chetverikov D, Szabó Z. Detection of high curvature points in planar curves. Source: http://visual.ipan.sztaki.hu/corner/.
[2] Kolesnikov A. Efficient algorithms for vectorization and polygonal approximation. Ch 3. PhD Thesis 2003.
[3] Vallone U. Bidemensional shapes polygonalization by ACO. Proc 3rd Int Workshop on Ants Algorithms 2002: 296-297. DOI: 10.1007/3-540-45724-0_33.
[4] Dorigo N. Optimization, learning, and natural algorithms. PhD Thesis. Milano, Italia: Politecnico di Milano; 1992.
[5] Dorigo M, Di Caro G, Gambardella LM. Ant algorithms for discrete optimization. Artif Life 1999; 5(2): 137-172. DOI: 10.1162/106454699568728.
[6] Leu JG, Chen L. Polygonal approximation of 2-D shapes through boundary merging. Pattern Recognit Lett 1988; 7(4): 231-238. DOI: 10.1016/0167-8655(88)90107-9.
[7] Latecki LJ, Lakamper R. Convexity rule for shape decomposition based on discrete contour evolution. Comput Vis Image Underst 1999; 73(3): 441-454. DOI: 10.1006/cviu.1998.0738.
[8] Ku KM, Chiu KM. Polygonal approximation of digital curve by graduate iterative merging. Electron Lett 1995; 31(6): 444-446. DOI: 10.1049/el:19950319.
[9] Fahn C-S, Wang J-F, Lee J-Y. An adaptive reduction procedure for the piecewise linear approximation of digitized curves. IEEE Trans Pattern Anal Mach Intell 1989; 11(9): 967-973. DOI: 10.1109/34.35499.
[10] Wu J-S, Leou J-J. New polygonal approximation schemes for object shape representation. Pattern Recogn 1993; 26(4): 471-484. DOI: 10.1016/0031-3203(93)90103-4.
[11] Boxer L, Chang C-S, Miller R, Rau-Chaplin A. Polygonal approximation by boundary reduction. Pattern Recogn Lett 1993; 14(2): 111-119. DOI: 10.1016/0167-8655(93)90084-Q.
[12] Visvalingam M, Whyatt J. Line generalization by repeated elimination of points. Cartogr J 1993; 30(1): 46-51. DOI: 10.1179/000870493786962263.
[13] Pikaz A, Dinstein I. An algorithm for polygonal approximation based on iterative point elimination. Pattern Recogn Lett 1995; 16(6): 557-563. DOI: 10.1016/0167-8655(95)80001-A.
[14] Horst JA, Beichel I. A simple algorithm for efficient piecewise linear approximation of space curves. Proc Int Conf on Image Proces (ICIP'97) 1997; 2: 744-747. DOI: 10.1109/ICIP.1997.638603.