УДК 677.024: 519.15

БЫСТРЫЕ АЛГОРИТМЫ ВЫДЕЛЕНИЯ РАППОРТОВ ТОЧЕЧНЫХ ИЗОБРАЖЕНИЙ

Г.И. БОРЗУНОВ

(Московский государственный текстильный университет им. А.Н. Косыгина)

Выполненный ранее теоретический анализ простейших алгоритмов выделения раппортов в текстильных рисунках показал, что эти алгоритмы могут быть использованы в учебных или демонстрационных целях для обработки изображений небольшой размерности. В данной работе впервые рассматриваются разработанные автором более быстрые алгоритмы.

Быстрое выделение раппортов изображений, не содержащих случайных искажений (изображения класса 1), выполняется по следующей схеме (Rpt3): 1. Для заданного изображения A[m][n] вычисляется размер раппорта по вертикали mr: попарно сравниваются горизонтальные полосы ((0, 0), (mr-1, n-1)), ((mr, 0), (2*mr-1, n -1)) , ((2*mr, 0), (3*mr-1, n -1)), …, при изменении ширины полос mr от 2 до m/2 и определяется такое значение mr, при котором все сравниваемые полосы совпадают. 2. Вычисляется размер раппорта по горизонтали nr: попарно сравниваются вертикальные полосы ((0, 0), (m-1, nr-1)), ((0, nr), (m-1, 2*nr-1)) , ((0, 2*nr), (m-1, 3*nr -1)),…, при изменении ширины вертикальных полос nr от 2 до n/2 и определяется такое значение nr, при котором все сравниваемые строки совпадают. 3. В качестве раппорта изображения A[m][n] принимается

R[mr][nr], левый верхний угол которого совпадает с любым элементом массива A[m][n]. Определим функции несовпадения W3(mr, A [m][n]) и W4(nr, A [m][n]): int W3(mr, A [m][n]) {int ir; for(int i=mr; i<m; i++) for(int j=0; j<n; j++) {ir=i%mr; if(A[i][j]≠A[ir][j]) { return 1;} } return 0;}. int W4(nr, A [m][n]) {int jr; for(int i=0; i<m; i++) for(int j=nr;j<n; j++) { jr=j%nr; if(A[i][j]≠A[i][jr]) { return 1;} } return 0;} Используя функции W3, W4, псевдокод алгоритма Rpt3 можно представить в виде функции: int mr=0, nr=0; void Rpt3(A [m][n]) {int ir=0, jr=0; /* Определение вертикального размера раппорта */ for(int imr=2; imr ≤m/2; imr ++) {if(W3(imr, A [m][n]))==0){mr= imr; break;} } if(mr==0) {return;} /* Определение горизонтального размера раппорта */ for(jnr=2; jnr ≤n/2; jnr++){if(W4(jr, A [m][n])==0){nr=jnr; break;}} return;} /* Раппорт не существует, если mr==0 или nr==0; иначе раппорт R[mr][nr] совпадает с любым подмассивом A[mr][nr]*/

Пусть выполняется K=m=n. В худшем случае для вычисления функции несовпадения W3(ir, A [m][n]) при каждом вызове потребуется следующее количество циклов: T6(K)= (K- ir) × K. При определении вертикального размера раппорта по алгоритму Rpt3 параметр ir меняется от 2 до K/2, поэтому T7(K) = (((K- 2) × K)+((K- 3) × K))+…+ ((K- K/2) × K)) =)= ((K- 2)+(K-3)+…+ (K- K/2)) × K= ((K/2-1) × K – ((1+2+…+K/2) -1)) × K= $K^3/2$ - K^2 – ((K/2× (K/2+1)/2-1)×K= $K^3/2$ - K^2 – $K^3/8$ – $K^2/4$ +K=3/8× K^3 - 5/4×K^2 + K ≈O(K^3). Временная сложность определения горизонтального размера раппорта по алгоритму Rpt3 с использованием функции W4(mr, A [m][n]) также равна T6(K) ≈O(K^3). Так как в целом работа алгоритма Rpt3 сводится к последовательному определению вертикального размера раппорта и определению горизонтального размера раппорта, то временная сложность всего алгоритма Rpt3 равна: T7(K) = 2×T6(K)≈ O(K^3).

Пусть изображение второго A2[m][n] получается из изображения первого класса A1[m][n] в результате случайного изменения некоторых точек. Пусть R1[mr][nr] является раппортом изображения A1[m][n], то есть функция несовпадений для R1[mr][nr] равна нулю (W(A1[m][n],V1[m][n])=0). Тогда размеры раппорта A2[m][n] по вертикали и горизонтали соответственно равны mr, nr, то есть существует R2[mr][nr], для которого W(A2[m][n],V2[m][n]) → min. Действительно попытка использовать в качестве раппорта R2[mr2][nr2] (подмассива A2[m][n]), размеры по вертикали и горизонтали отличаются от значений mr, nr, приводит почти всегда к увеличению значения функции несовпадений, ввиду того, что к несовпадениям из-за наложенного на A1[m][n] шума добавляются несовпадения точек R2[mr2][nr2] с точками A1[m][n] из-за различия размеров R2[mr2][nr2] и размеров R1[mr][nr]. После определения размеров mr, nr раппорта изображения A2[m][n] вычисление раппорта этого изображения завершается выбором любого R2[mr][nr], являющегося подмассивом A2[m][n], и устранением различий точек R2[mr][nr], вызванных наложенным на A1[m][n] шумом, одним из методов теории вероятностей. Ниже впервые приводится псевдокод Rpt4 описанного выше алгоритма быстрого определения раппортов изображений класса 2. В алгоритме Rpt4 для определения размеров раппорта изображения A[m][n] используются функции W5, W6, которые получены в результате переработки функций W3, W4: long int W5(mr, A [m][n]) {long int s=0; int ir; for(int i=mr;i<m;i++) for(int j=0;j<n;j++) {ir=i%mr; if(A[i][j]<A[ir][j]) s+=(A[ir][j] - A[i][j]); else s+=(A[i][j] - A[ir][j]);} return s;} long int W6(nr, A [m][n]) {long int s=0; int jr; { int jr; for(int i=0;i<m;i++) for(int j=nr; j<n; j++) {jr=j%nr; if(A[i][j]<A[i][jr])s+=(A[i][jr] - A[i][j]);else s+=(A[i][j] - A[i][jr]);} return s;}

Псевдокод функции Rpt4 можно представить в следующем виде:

int array Rpt4(A2 [m][n]) {int mrmin=0, nrmin=0; long int wmrmin=255×n×m; long int wnrmin=255×n×m; long int ww=0; for(int imr=2; imr≤m/2; imr++) {ww= W5(imr, A [m][n]) if(ww< wmrmin) {mrmin=imr, wmrmin=ww; }} for(nr=2; nr≤n/2; nr++) {ww= W6(nr, A [m][n]) if(ww< wnrmin)

{nrmin=nr, wnrmin=ww; } }

//Устранение различий точек, вызванных шумовым эффектом

Распределить память для long int NA[mr][nr]=0; double float SA[mr][nr]=0; int R[mr][nr]; int ir, jr; for(int i=0;i<m;i++) for(int j=0;j<n;j++) {ir=i%mr; jr=j%nr; SA[ir][jr] =(SA[ir][jr] + A[i][j]); NA[ir][jr]++;} for(int i=0;i<mr;i++) for(int j=0;j<nr;j++) R[i][j]=int(SA[i][j]/NA[i][j];)

//Расчет изображения, полученного после устранения шумового эффекта

Распределить память для int AA[m][n]; for(int i=0;i<m;i++) for(int j=0;j<n;j++) {ir=i%mr;jr=j%nr; AA[i][j]=R[ir][jr]; } Освободить память NA[mr][nr]; SA[mr][nr]; R[mr][nr]; return AA[m][n]; }.Подмассив A[mr][nr] является раппортом изображения AA[m][n]. Как видно из приведенного выше псевдокода Rpt4, в данном варианте алгоритма для значений точек, имеющих в пределах раппорта одни и те же координаты (ir, jr), в качестве одинаковых значений приняты средние значения всех таких точек в пределах исходного изображения A[m][n]. Для вычисление функции несовпадений W5(mr, A [m][n]) требуется (K-mr) ×K циклов как в лучшем, так и в худшем случае. Аналогично для вычисления функции несовпадений W6(nr, A [m][n]) требуется выполнение (K-nr) ×K циклов также в лучшем и в худшем случаях. Эти оценки количества циклов, необходимых для вычисления W5(mr, A [m][n]), W6(nr, A [m][n]), совпадают с оценками количества циклов, требуемых для вычисления W3(mr, A [m][n]) и W4(nr, A [m][n]). При определении размеров раппорта mrmin и mrmin по алгоритму Rpt4 параметры mr и nr меняются от 2 до K/2, как и изменение этих параметров при выполнении алгоритма Rpt3. Таким образом, временные сложности вычисления mrmin, mrmin равны и определяются выражением: $T8(K)=T9(K) = ((K/2-1)\times K-((1+2+\ldots+K/2)-1))\times K \approx O(K^3)$. Устранение различий точек, вызванных шумовым эффектом, содержит два последовательно выполняющихся циклических участка. Реализация первого из них требует выполнения в лучшем и худшем случаях K^2 циклов, второй выполняется в худшем случае за $(K^2/4)$ циклов. Тогда временная сложность устранения различий точек, вызванных шумовым эффектом, оказывается равной: $T10(K) = K^2 + (K^2/4) \approx O(K^2)$. Работа алгоритма Rpt4 завершается расчетом кода изображения, для которого требуется выполнить K^2 циклов. Временная сложность этого заключительного этапа равна: $T11(K) = K^2 \approx O(K^2)$. Рассмотренные выше этапы алгоритма Rpt4 выполняются последовательно, поэтому временная сложность алгоритма Rpt4 в целом равна: $T11(K)= T8(K) + T9(K) + T10(K) + T10(K) = O(K^3)+ O(K^3)+ O(K^2)+ O(K^2)\approx O(K^3)$.

В Ы В О Д Ы

1. Разработанные автором алгоритмы Rpt3, Rpt4 обладают значительно более низкой временной сложностью по сравнению с решающими аналогичные задачи простейшими алгоритмами Rpt1 и Rpt2. Алгоритм Rpt3 обеспечивает по сравнению с Rpt1 коэффициент ускорения O(K) при выделении раппортов изображений, не содержащих случайных искажений, а алгоритм Rpt4 – по сравнению с Rpt2 коэффициент ускорения $O(K^3)$, при выделении раппортов изображений, в которых имеют место искажения цвета некоторых точек.

2. Дальнейшее снижение временной сложности выделения раппортов может быть достигнуто с использованием средств параллельного программирования.

Рекомендована кафедрой информационных технологий и компьютерного дизайна. Поступила 16.04.09.