

УДК 681.323; 621.372.54

Реализация ранжирующих и медианных фильтров на процессоре NM6403 (L1879BM1)

С.В. Мушкаев

Введение

Медианная фильтрация является достаточно изученным и широко применяемым аппаратом в области цифровой обработки сигналов. Характерной особенностью медианного фильтра является его нелинейность, что особо выделяет его среди множества других алгоритмов фильтрации. Во многих случаях применение медианного фильтра более предпочтительно, поскольку, например, процедуры линейной обработки являются оптимальными при гауссовском распределении сигналов и помех, что в реальных изображениях далеко не так. В случаях, когда перепады яркости велики по сравнению с дисперсией аддитивного белого шума медианный фильтр дает меньшее значение СКО по сравнению с оптимальным линейным фильтром. Особенно эффективным медианный фильтр оказывается в случае импульсного шума. Так же особо важным свойством медианного фильтра является сохранение контуров изображений.

Как известно, медианный фильтр представляет собой оконный фильтр, последовательно скользящий по полю изображения или сигнала, и возвращающий на каждом шаге один из элементов попавших в окно. Как правило, используются окна нечетного размера, и тогда возвращается центральный элемент ряда, полученного в результате сортировки всех элементов окна. Однако, в ряде задач цифровой обработки сигналов помимо среднего бывает необходимо выбирать произвольный ранжируемый элемент - от минимального до максимального. В данной статье рассматривается именно такой произвольно ранжирующий фильтр. Предлагаемый алгоритм является

Рассматривается принцип построения быстрых одномерных медианных фильтров произвольного порядка. Приводится его реализация на процессоре NM6403. Предложенная схема реализации исключает операции сравнения в явном виде и не требует команд условного перехода, что позволяет эффективно распараллеливать алгоритм обработки и использовать векторные процессоры. Важным достоинством приведенного алгоритма является его линейная сложность относительно порядка фильтра (размера окна фильтрации). Механизм вычислений таков, что помимо медианного значения фильтр может одновременно выполнять функцию выбора минимального, максимального или же значения любого другого ранга.

универсальным относительно степени ранжируемого элемента, так как настройка фильтра на поиск минимального, максимального, медианного значения или значения любого другого ранга задается параметрически. Кроме того, алгоритм достаточно легко расширяется, что позволяет создать эффективный совмещенный фильтр, возвращающий одновременно несколько значений разного ранга.

Ниже приведены примеры работы различных ранжирующих фильтров с размером окна, равным 3. (медианного – рис.1, минимального и максимального - рис.2)

Алгоритм построения одномерного медианного фильтра

Несмотря на то, что рассматриваемый ниже алгоритм имеет достаточно общий вид и является универсальным для любых платформ, предназначен он, в первую очередь, для процессоров с параллельной обработкой данных. В частности, данный алгоритм разрабатывался, исходя из следующих архитектурных особенностей процессора NM6403:

- Предполагается, что исходные данные являются целочисленными со знаком и могут иметь азрядность: от 2 до 64 бит. В целях достижения

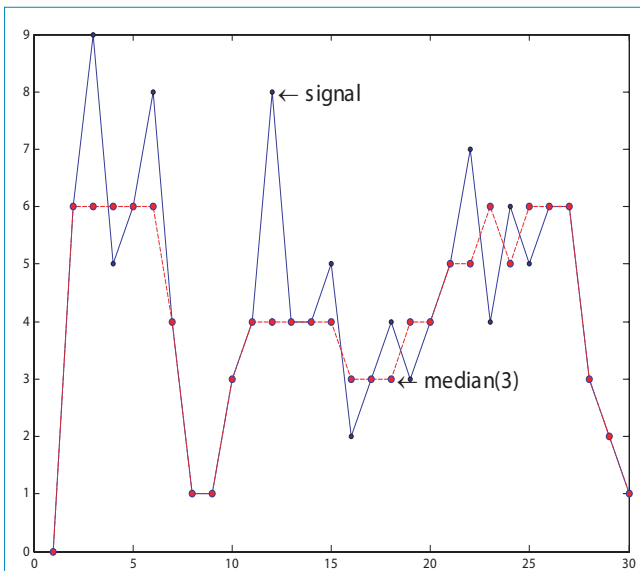


Рис. 1 Пример обработки медианным фильтром

максимальной производительности данные должны быть упакованы в64-разрядные слова и обрабатываться параллельно на векторном узле процессора NM6403.

- Основной внутренней операцией медианного фильтра является сравнение двух чисел. Но так как обработка данных ведется параллельно сразу над группой чисел, то при работе с этими данными алгоритм должен исключать операции условного перехода.

Так как процесс медианной фильтрации состоит в последовательном перемещении окна фильтра по массиву и сравнении данных внутри окна, то очевидно, что оптимальным алгоритм будет в случае, когда на очередном шаге фильтрации будут повторно использоваться результаты сравнения, полученные на предыдущих шагах. Данный подход возможен и реализуется без операций условного перехода. Это позволяет, с одной стороны, распараллелить обработку на процессоре, а с другой стороны, создать быстрый алгоритм фильтрации с линейно возрастающей сложностью в зависимости от размера окна.

Рассмотрим данный подход на примере короткого массива из 6 чисел $X=\{X_0, X_1, \dots, X_5\}$ с размером окна медианного фильтра, равного 5.

1. Сначала находим медиану в первом окне из 5 чисел $\{X_0, X_1, \dots, X_4\}$. Для этого производим всевозможные попарные вычитания: $X_i - X_j, i, j=0, \dots, 4$, и вычисляем их знак с помощью пороговой функции насыщения

$$F_{ij}=f(x_i-x_j), \text{ где } f(X) = \begin{cases} 0, & \text{если } X \geq 0 \\ -1, & \text{если } X < 0 \end{cases}$$

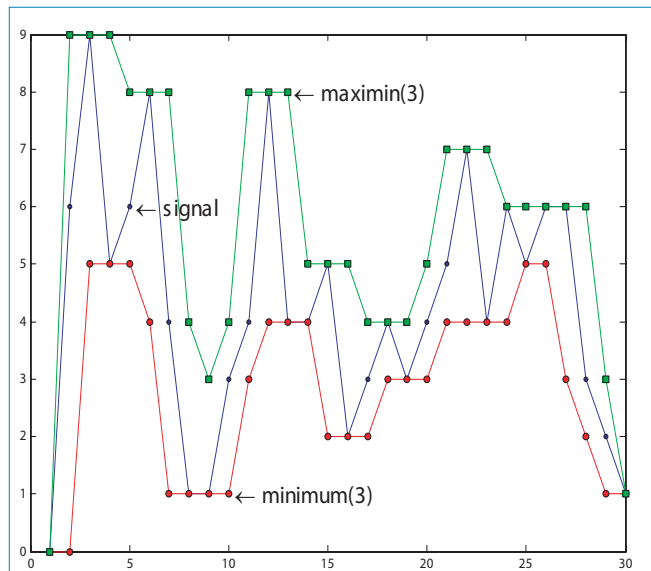


Рис. 2 Пример обработки фильтром выбора минимального и максимального значения

В результате чего составляется матрица $[F]$:

$f(x_0-x_0)$	$f(x_1-x_0)$	$f(x_2-x_0)$	$f(x_3-x_0)$	$f(x_4-x_0)$
$f(x_0-x_1)$	$f(x_1-x_1)$	$f(x_2-x_1)$	$f(x_3-x_1)$	$f(x_4-x_1)$
$f(x_0-x_2)$	$f(x_1-x_2)$	$f(x_2-x_2)$	$f(x_3-x_2)$	$f(x_4-x_2)$
$f(x_0-x_3)$	$f(x_1-x_3)$	$f(x_2-x_3)$	$f(x_3-x_3)$	$f(x_4-x_3)$
$f(x_0-x_4)$	$f(x_1-x_4)$	$f(x_2-x_4)$	$f(x_3-x_4)$	$f(x_4-x_4)$

2. В полученной матрице $[F]$, суммируем значения F_{ij} по столбцам и получаем массив $[N]$:

$$n_i = 4 + \sum_{j=0}^4 F_{ij} = 4 + \sum_{j=0}^4 f(x_i - x_j), i = 0..4$$

3. Полученные значения n_i – соответствуют порядковым номерам чисел X_i в отсортированном ряду (X_i соответствует минимуму при $n_i = 0$, медиане - при $n_i = 2$, и максимуму - при $n_i = 4$). Выделение элемента необходимого ранга по имеющемуся массиву $[N]$ можно осуществить как на скалярном ядре так и на векторном процессоре с помощью логическо-арифметических операций.

Представленный подход лежит в основе построения алгоритма быстрой медианной фильтрации по всему массиву, так как на каждом шаге он позволяет использовать накопленную ранее информацию, а именно массивы $[N]$ и $[F]$, для ускорения поиска медианы в следующем окне.

Покажем это на примере следующего окна, сдвинутого на 1 позицию: $\{X_1, X_2, \dots, X_5\}$.

Для вычисления нового значения медианы матрицу $[F]$ достаточно пополнить всего девятью разностями (на рисунке они обозначены жирным шрифтом), после чего медиана окажется в правой нижней подматрице 5×5 , как было описано выше:

$f(x_0-x_0)$	$f(x_1-x_0)$	$f(x_2-x_0)$	$f(x_3-x_0)$	$f(x_4-x_0)$	
$f(x_0-x_1)$	$f(x_1-x_1)$	$f(x_2-x_1)$	$f(x_3-x_1)$	$f(x_4-x_1)$	$f(x_5-x_1)$
$f(x_0-x_2)$	$f(x_1-x_2)$	$f(x_2-x_2)$	$f(x_3-x_2)$	$f(x_4-x_2)$	$f(x_5-x_2)$
$f(x_0-x_3)$	$f(x_1-x_3)$	$f(x_2-x_3)$	$f(x_3-x_3)$	$f(x_4-x_3)$	$f(x_5-x_3)$
$f(x_0-x_4)$	$f(x_1-x_4)$	$f(x_2-x_4)$	$f(x_3-x_4)$	$f(x_4-x_4)$	$f(x_5-x_4)$
	$f(x_1-x_5)$	$f(x_2-x_5)$	$f(x_3-x_5)$	$f(x_4-x_5)$	$f(x_5-x_5)$

При этом количество вычислений может быть значительно снижено за счет дополнительной оптимизации:

1. Разности x_i-x_j не несут полезной информации и поэтому могут быть исключены.

2. Для каждой разности $f(x_i-x_j)$ в таблице присутствует ее инверсия $f(x_j-x_i)$, учитывая,

что $f(x_j-x_i) = \text{not } f(x_i-x_j)$ время вычисления таких разности можно сократить.

3. При переходе на новое окно нет необходимости заново суммировать все элементы в столбцах новой подматрицы матрицы $[F]$, где $F_{ij}=f(x_i-x_j)$ $i,j=1,\dots,5$, а достаточно скорректировать накопленные суммы $[N]$ от предыдущего окна путем добавления новой разности $f(x_i-x_5)$ и вычитанием выпавшей разности $f(x_i-x_0)$.

Например, для 2-ой колонки матрицы $[F]$ справедливо:

$$\begin{aligned} \sum_{i=1}^5 f(x_1-x_i) &= f(x_1-x_5) + \sum_{i=0}^4 f(x_1-x_i) - f(x_1-x_0) = \\ &= f(x_1-x_5) + n_1 - f(x_1-x_0) \end{aligned}$$

При больших размерах окна такое суммирование особенно эффективно и приближает сложность алгоритма к линейной зависимости от размера окна.

Выводы

Приведенный алгоритм обладает следующими преимуществами:

- Алгоритм позволяет реализовывать универсальные функции с параметрическим указанием как размера окна, так и размера фильтруемого массива. Ограничения по выбору размера окна определяются лишь необходимостью выделения дополнительного объема памяти под промежуточные вычисления.

Литература

1. "Цифровая обработка изображений в информационных системах" / Грузман И.С., Киричук В.С., Косых В.П., Перетягин Г.И., Спектор А.А. Учебное пособие.- Новосибирск: Изд-во НГТУ, 2000. - 168.
2. "Быстрые алгоритмы в цифровой обработке

- Данный алгоритм инвариантен к выбору номера ранжируемого элемента и может быть выбран произвольно. При этом относительно несложно модифицировать процедуру фильтрации так, чтобы на выходе иметь одновременно несколько отфильтрованных массивов с различными значениями номеров ранжируемых элементов.

- Алгоритм имеет линейную структуру, исключая ветвления. Это позволяет эффективно производить распараллеливание вычислений и организовать поточную обработку данных.

- Алгоритм медианной фильтрации имеет линейную сложность и соответствует величине $O(W*N)$, где W -размер скользящего окна, N -длина фильтруемого массива. Этот результат достигается за счет того, что при сдвиге скользящего окна не производится заново сортировка элементов окна, а используются результаты сравнений, полученные на предыдущих шагах. В случае, если процессор позволяет производить арифметические и логические операции одновременно над группой упакованных элементов (iPentium(MMX),NM6403), время вычислений сокращается в соответствующее число раз.

Результаты

Приведенный выше алгоритм был реализован на процессоре NM6403(L1879BM1).

При размере исходного массива из 2048 16-разрядных элементов и размере скользящего окна, равного 16, выполнение процедуры фильтрации потребовало 97000 тактов, что при частоте процессора 40Мгц, составляет ~ 2.4 мс. Каждое выходное значение формируется в среднем за 47 тактов.

Эмпирическая формула полного числа тактов в зависимости от длины массива, размера скользящего окна и размерности данных выглядит следующим образом: $Clocks=10.38*N*W/K$, где N -длина массива, W -размер скользящего окна, K -число элементов упакованных в одном векторе.

изображений" /Под ред. Т.С. Хуанга. - М.: Радио и связь, 1984

3. Прэтт У. Цифровая обработка изображений: В 2 т. М.: Мир, 1982. Быстрые алгоритмы в цифровой обработке изображений /Под ред. Хуанга Т. С., М.: Радио и связь, 1984.