вычислительного эксперимента показывают, что ортогональность базиса позволяет однозначно декодировать переданную информацию, при этом сохраняется помехоустойчивость. Используемые огибающие, хранящиеся в памяти схемы, обладают основными свойствами, поэтому полученная сигнально-кодовая конструкция обладает всеми преимуществами по спектральной эффективности и помехоустойчивости.

Разработанный аппаратно-программный комплекс показывает возможность реализации сигнально-кодовых конструкций на реальном оборудовании. Использованная программированная логическая интегральная схема обладает всеми необходимыми модулями, но не пригодна для применения в малогабаритных системах связи. Дальнейшая работа будет заключаться в реализации метода кодирования и декодирования передаваемой информации на контролере STM32F107.

Литература

1. Жиляков Е.Г., Белов С.П., Урсол Д.В. Оптимальные канальные сигналы при цифровой передаче с частотным уплотнением. – "Научные ведомости БелГУ", сер. Информатика, 2009, № 7(62), вып. 10/1, с.166 – 172.

2. Жиляков Е.Г., Урсол Д.В., Красильников В.В. Плис как средство реализации оптимальной обработки канальных сигналов. – "Вопросы радиоэлектроники", сер. ЭВТ, , 2013, вып.. 1., с. 102-109

Работа выполнена при поддержке гранта РФФИ 12-07-00514-а.

Статья поступила 05.12.2013

Д.т.н., проф. Е.Г. Жиляков, к.т.н., доц. А.А. Черноморец, к.т.н. А.А. Барсук, А.Ю. Лихошерстный (НИУ БелГУ)

E.G. Zhilyakov, A.A. Chernomorets, A.A. Barsuk, A.U. Likhosherstniy

ОБ ИСПОЛЬЗОВАНИИ МНОГОЯДЕРНЫХ ПРОЦЕССОРОВ ДЛЯ ОБРАБОТКИ ИЗОБРАЖЕНИЙ

USING MULTI-CORE PROCESSORS IN IMAGE PROCESSING

В статье рассматривается подход к реализации параллельного алгоритма фильтрации высокочастотных шумов на изображениях на основе новых методов субполосного анализа и синтеза с использованием многоядерных процессоров.

The approach to implementation of the parallel algorithm of removing high frequency noise from image based on new methods of subband analysis and synthesis using multi-core processors is considered in the report.

Ключевые слова: обработка изображений, частотный интервал, фильтрация, параллельный алгоритм, многоядерный процессор, SIMD, MIMD, SMP, GPU, CUDA, CPU, OpenMP.

Key words: image processing, frequency interval, filtration, parallel algorithm, multicore-processor, SIMD, MIMD, SMP, GPU, CUDA, CPU, OpenMP.

Одним из основных видов представления информации об окружающей реальности являются изображения, полученные в результате регистрации электромагнитных излучений, отражаемых и испускаемых различными физическими объектами. Анализ таких изображений позволяет получать знания о запечатленных на них объектах.

Изображения, получаемые с устройств регистрации, как правило, лишь частично удовлетворяют заданным требованиям, поэтому после оценки качества исходных изображений формулируются требования к предварительной обработке [1]. В соответствии с этими требованиями и выбирается наиболее подходящий вид (способ, алгоритм, цепочка преобразований и т.д.) обработки исходного изображения.

Зачастую задачи обработки изображений связаны с выполнением большого количества вычислительных операций, что приводит к необходимости использования параллельных вычислений и соответствующих вычислительных устройств для их решения. Организация параллельности вычислений, когда в один и тот же момент выполняется одновременно несколько операций обработки данных, осуществляется, в основном, за счет введения избыточности функциональных устройств (многопроцессорности).

В настоящее время большинство современных процессоров являются многоядерными, т.е. по сути представляют собой

SMP-систему с несколькими процессорами. Аппаратные решения для рабочих станций обычно содержат 2 или 4 ядра, для серверных решений этот показатель может достигать 8. В качестве удобной системы для программирования параллельных вычислений для таких процессоров выступает библиотека OpenMP, позволяющая с минимальными затратами труда реализовывать требуемые параллельные вычислительные операции. В то же время использование центральных процессоров для параллельных вычислений наиболее хорошо подходит для решения не связанных друг с другом, или имеющих сложную логическую структуру задач, поскольку каждое ядро является независимым полноценным процессором и, в целом, в них реализуется концепция MIMD.

С другой стороны, для решения задач, характеризующихся высокой вычислительной сложностью, все более широко используются графические карты. Это связано с тем, что большинство процедур обработки трехмерной графики является вычислительно сложными и обладает высоким потенциальным параллелизмом. Этот факт обуславливает историческое развитие видеоускорителей как отдельных от центрального процессора вычислительных устройств, обладающих возможностью параллельной обработки данных и построенных на базе SIMD архитектуры. Это привело к появлению нового направления в компьютерной обработке данных, называемого GPGPU (General-purpose graphics processing units) - технология использования видеоускорителей в вычислениях общего назначения.

На сегодня основными производителями видеоускорителей являются две корпорации: AMD и Nvidia. Каждая из них разрабатывает собственную технологию программирования параллельных вычислений для своих продуктов - FireStream и CUDA соответственно. Так же существует открытая технология программирования GPGPU задач, не привязанная к конкретному производителю или архитектуре, называемая OpenCL (Open Computing Language). Следует отметить, что самой динамично развивающейся, поддерживаемой и удобной для использования является CUDA. Еще одним существенным преимуществом этой технологии перед другими является наличие библиотеки cuBlas (CUDA Basic Linear Algebra Subprograms) - библиотеки подпрограмм линейной алгебры.

Использование видеоускорителей в задачах общих вычислений целесообразно тогда, когда необходимо решать вычислительно сложные задачи, обладающие высоким потенциальным параллелизмом на уровне данных. Типичным представителем такого класса задач является, например, задача умножения матриц. Ускорение, получаемое при использовании графических процессоров для ее решения, может составлять десятки раз по сравнению с современными центральными процессорами. На рис. 1 представлен график сравнения скорости вычисления произведения матриц с использованием библиотеки cuBlas и MKL (Intel Math Kernel Library - математическая библиотека, оптимизированная для процессоров фирмы Intel). Вычисления производились на графическом ускорителе Tesla M2090 и центральном шестиядерном процессоре Intel Xenon x5680 [2].

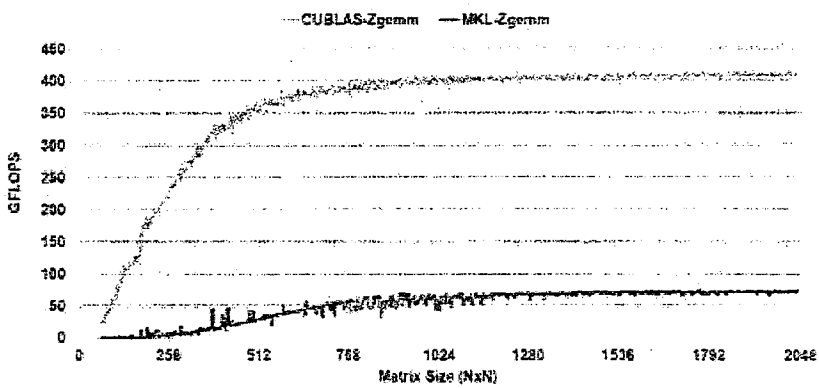

Рис. 1
График сравнения скорости вычисления произведения матриц с использованием библиотеки cuBlas и MKL

На графике видно, что в некоторых случаях ускорение на видеокарте по сравнению с центральным процессором достигает 15 раз. Следует отметить, что при использовании библиотеки MKL и CPU производства Intel, ресурсы центрального процессора задействованы наиболее эффективно - используются все доступные вычислительные ядра и векторизация.

Для исследования вопроса о применении многоядерных процессоров для обработки изображений рассмотрим алгоритм

фильтрации высокочастотных шумов на основе новых методов субполосного анализа и синтеза. Блок-схема алгоритма представлена на рис. 2 и 3.

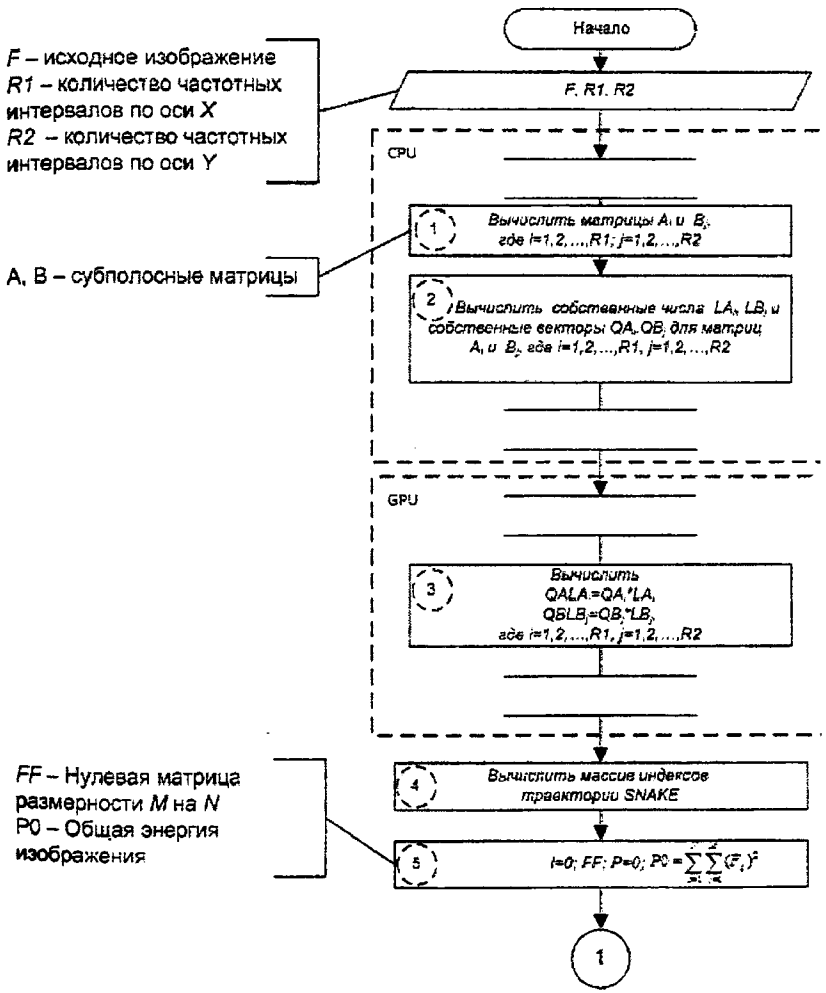

Рис. 2
Алгоритм удаления высокочастотных шумов на основе новых методов субполосного анализа и синтеза (первая часть)

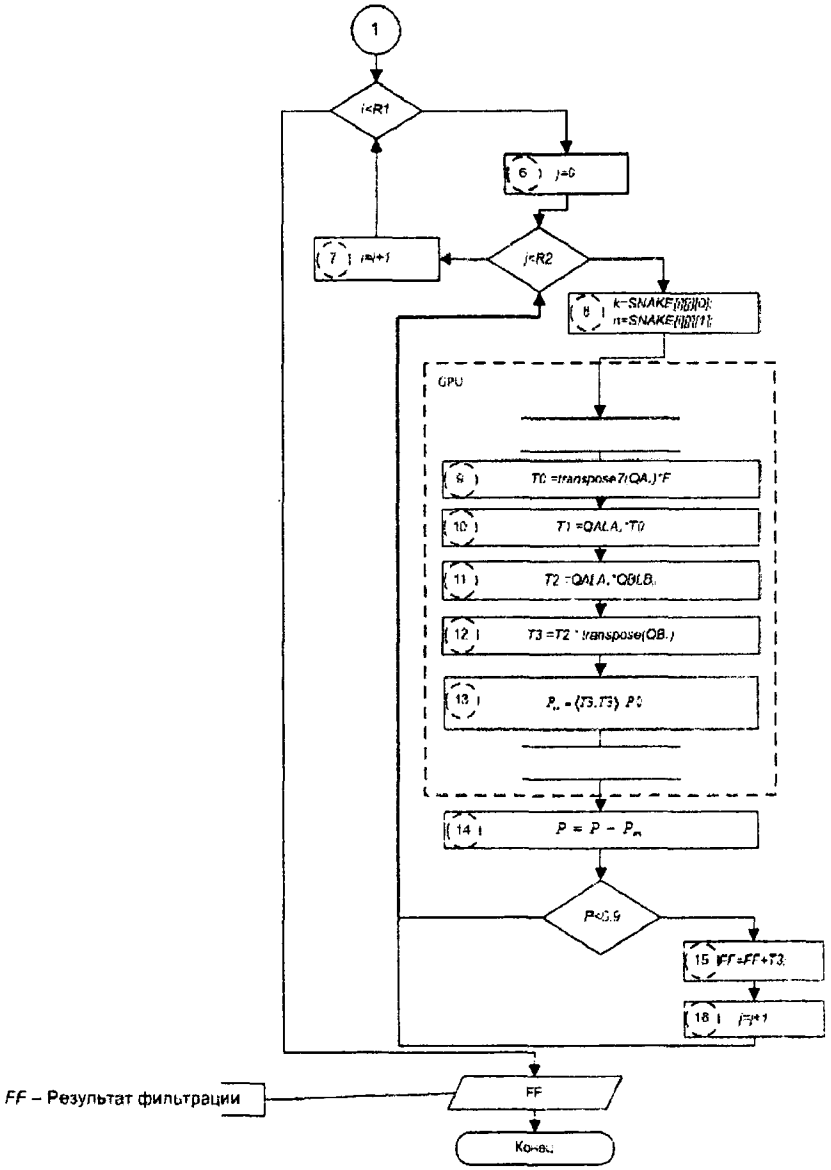

Рис. 3

Алгоритм удаления высокочастотных шумов на основе новых методов субполосного анализа и синтеза (вторая часть)

На первом шаге алгоритма производится расчет субполосных матриц A_i и B_j [3]. Пара таких матриц должна быть рассчитана $R1+R2$ раз, где каждый элемент вычисляется в соответствии с соотношением

$$a_{nk}^{i} = \begin{cases} \dfrac{\sin(\sigma_{i+1}\cdot(n-k)) - \sin(\sigma_i\cdot(n-k))}{\dfrac{n-k}{\pi}}, n \neq k, \\ \dfrac{\sigma_{i+1}-\sigma_i}{\pi}, n = k, \end{cases}$$

$$b_{nk}^{j} = \begin{cases} \dfrac{\sin(\gamma_{j+1}\cdot(n-k)) - \sin(\gamma_j\cdot(n-k))}{\dfrac{n-k}{\pi}}, n \neq k, \\ \dfrac{\gamma_{j+1}-\gamma_j}{\pi}, n = k, \end{cases}$$

$$i = 1,2,\ldots,R1;\ j = 1,2,\ldots,R2;$$

$$\sigma_1 = 0, \sigma_2 = \frac{2\pi}{M}, \sigma_i = \sigma_{i-1} + \frac{4\pi}{M}$$

$$\gamma_1 = 0, \gamma_2 = \frac{2\pi}{N}, \gamma_i = \gamma_{i-1} + \frac{4\pi}{N},$$

где: M,N - размеры изображений,

$R1$, $R2$ - количества частотных интервалов.

Такие расчеты предполагают множественную проверку условий, поэтому плохо подходят для реализации на GPU, что связано с особенностями работы последнего. Поэтому вычисления шага 1 алгоритма фильтрации были реализованы на CPU в многопоточном режиме.

На втором шаге алгоритма необходимо вычислить собственные числа и собственные векторы для полученных ранее матриц. Для решения этой задачи была использована открытая библиотека Eigen. Вычисления проводились в многопоточном режиме на центральном процессоре.

Третий шаг предполагает расчет значений матриц $QALA_i$ и $QBLB_j$.Эти значения будут использованы на следующих шагах.

Как было показано выше, использование видеоускорителей хорошо подходит для реализации вычислений с множественным потоком данных и одним потоком команд. Операция перемножения матриц как раз является одним из представителей этого класса. Поэтому вычисления третьего шага производились на GPU.

Шаги 4,5,6,7,8,14,15,16 не предполагают большого количества вычислительных операций, поэтому они производились на центральном процессоре в главном потоке.

Шаги 9,10,11,12 содержат операции умножения и транспонирования матриц.

Шаг 13 включает операцию скалярного произведения векторов (матрица T3 расположена в памяти как одномерный массив, поэтому в результате вычисления скалярного произведения будет получена сумма квадратов ее элементов).

При использовании 64 частотных интервалов по двум измерениям такие вычисления необходимо произвести 64^2, т.е. 4096, раз. Итого для решения поставленной задачи нужно выполнить 16384 операции матричного произведения.

При обработке изображений размерностью 1024x1024 пикселя примерное количество вычислительных операций, необходимое для перемножения двух матриц такой размерности составляет $\sim 10^9$ операций с плавающей точкой. Таким образом, общее количество вычислительных операций будет составлять $\sim 10^{13}$.

Данные вычисления целесообразно выполнять на графическом процессоре, т.к. это позволит получить существенное ускорение по сравнению с реализацией на центральном процессоре.

В целом, использование видеоускорителей для реализации субполосных методов и алгоритмов [4] позволит получить существенный прирост производительности, что связано со структурой предлагаемых алгоритмов.

В качестве аппаратной и программной баз наилучшим представляется использование графических ускорителей фирмы Nvidia и технологии CUDA, т.к. они максимально адаптированы для решения вычислительных задач общего назначения. При этом использование вычислительных возможностей центрального процессора целесообразно для решения задач не соответствующих SIMD концепции.

Литература

1. Гонсалес Р., Вудс Р. Цифровая обработка изображений. М., Техносфера, 2006. 1072 с.

2. NVIDIA Developer Zone [Электронный ресурс]. Режим доступа: https://developer.nvidia.com/cuBLAS.

3. Жиляков Е.Г., Черноморец А.А. Оптимальная фильтрация изображений на основе частотных представлений. – "Вопросы радиоэлектроники", 2008, сер. ЭВТ, вып. 1, с. 118-131.

4. Жиляков Е.Г., Черноморец А.А. О частотном анализе изображений. – "Вопросы радиоэлектроники". сер. ЭВТ, 2010, вып. 1, с. 94-103.

Работа выполнена при поддержке гранта РФФИ 12-07-00257-а.

Статья поступила 05.12.2013

Д.т.н., проф. Н.И. Корсунов, А.А. Начетов, д.т.н., проф. К.И. Логачев (НИУ "БелГУ)

N.I. Korsunov, A.A. Nachetov, K.I. Logachev

КОМПЕНСАЦИЯ ПОГРЕШНОСТИ В ДАННЫХ, ФОРМИРУЕМЫХ С ИСПОЛЬЗОВАНИЕМ ПРЕОБРАЗУЮЩЕЙ ФУНКЦИИ УМНОЖЕНИЯ

COMPENSATION OF ERRORS IN THE DATA RENDERED USING MULTIPLICATION FUNCTION CONVERTS

В статье предлагается подход к компенсации погрешности в данных, формируемых с использованием преобразующей функции умножения, позволяющий получить скорректированное значение результата умножения и значение компенсирующей поправки.

The paper proposes an approach to compensate for errors in the data generated using the transformative multiplication function that allows you to obtain the corrected value of the multiplier and the value of the compensation amendment.

Ключевые слова: компенсация погрешности, коррекция умножения, первичный преобразователь.

Key words: compensation of error, correction multiplier, transducer.