УДК 681.3:004.272

Л.И. Тимченко [1], д-р. техн. наук,
А.А. Яровой [2], **Н.И. Кокряцкая** [1], кандидаты техн. наук,
[1] Государственный экономико-технологический
университет транспорта
(Украина, 03049, Киев, ул. Лукашевича, 19,
тел. (044) 5915135, e-mail: timchen@list.ru),
[2] Винницкий национальный технический университет
(Украина, 21021, Винница, Хмельницкое шоссе, 95,
тел. (0432) 598243, e-mail: axa@vinnitsa.com)

Теоретические и прикладные аспекты параллельно-иерархического многоуровневого преобразования цифровых сигналов *

Выполнен анализ структурно-функциональной организации сетевой архитектуры параллельно-иерархического (ПИ) преобразования цифровых сигналов и методов формирования масок при его реализации. Предложен метод оптимизированного формирования масок при ПИ преобразовании для кодирования информации и разработан программный комплекс, в котором повышено быстродействие прямого и обратного ПИ преобразования информации без ее потерь.

Виконано аналіз особливостей структурно-функціональної організації сітчастої архітектури паралельно-ієрархічного (ПІ) перетворення цифрових сигналів і методів формування масок при його реалізації. Запропоновано метод оптимізованого формування масок при ПІ перетворенні для кодування інформації та розроблено програмний комплекс, в якому підвищено швидкість прямого та зворотнього ПІ перетворення інформації без її втрат.

Ключевые слова: параллельные вычисления, параллельно-иерархическое преобразование, обработка изображений, кодирование информации.

Решение проблемы быстрого преобразования больших массивов информации — изображений для эффективного обеспечения ее записи, сохранения, обработки и считывания — связаны с созданием быстродействующих устройств кодирования и декодирования. Быстродействие процесса коди-

* Данные исследования выполнены в рамках НИР № GP/F44/051 «Методы и средства организации высокопроизводительных параллельно-иерархических вычислительных процессов в интеллектуальных системах», которая осуществляется за счет бюджетных средств МОНМС Украины, предоставленных как грант Президента Украины для поддержки научных исследований молодых ученых.

рования-декодирования массивов цифровых данных зависит, прежде всего, от реализуемого алгоритма цифровой обработки. Современный уровень развития схемотехники многоканальных цифровых устройств кодирования- декодирования больших массивов информации все еще характеризуется преимущественно последовательными алгоритмами цифровой обработки с существенными временными затратами, которые связаны с последовательным во времени процессом кодирования-декодирования [1—4].

Указанная проблема становится наиболее актуальной в области кодирования изображений, где целесообразно использовать параллельную обработку. В отличие от широко распространенных типов кодирования видеоинформации, например разностной, кодово-импульсной, дельта-модуляции, в основу которых положен принцип последовательного кодирования разностной информации, предлагается использовать параллельно-пирамидальный принцип обработки распределенных в пространственно-временной области результатов кодирования массива данных, что приводит к существенному повышению алгоритмического быстродействия, уплотнению массивов данных и обеспечению параллельно-иерархической (ПИ) формы описания сигналов (изображений) [5].

Постановка задачи. В работах [6—9] рассмотрены примеры математических моделей сетевого метода ПИ преобразования цифровых сигналов, а также некоторые вопросы их применения. Для решения поставленной задачи повышения быстродействия прямого и обратного ПИ преобразования сигналов и оптимизации избыточности представления масок определим абстрактную модель сетевой структуры, способы представления масок для ее реализации и выполним экспериментальные исследования многоуровневого преобразования маскированных сигналов. В качестве цифровых сигналов используем массивы числовой информации и отсчеты полутоновых изображений.

Для обоснования целесообразности применения в прикладных задачах кодирования, обработки и сравнения изображений в ходе экспериментальных исследований выполнено компьютерное моделирование, разработан набор демонстрационных приложений, которые дают возможность оценить достоверность работы предложенной сетевой модели.

Основные понятия, определения и алгоритмы сетевой модели ПИ многоуровневого преобразования маскированных сигналов. Используемые в теории информационных структур и теории графов линейные списки и деревья [10] являются структурами с упорядоченными связями. Поэтому информация о структуре при их описании определяется типом связей между элементами данных и является структурой с гибкой иерархией. Одним из путей реализации параллелизма при работе с многосвяз-

ными структурами является регуляризация, обеспечивающая описание их с помощью регулярной сетевой структуры преобразования. При этом информация о связях включается в сетевую структуру преобразования в явном виде, т.е. представляется в виде элементов данных.

Введем некоторые понятия, относящиеся к древовидной модели сетевой структуры ПИ преобразования с регулярными связями [5]. Пусть граф $G=(V,E)$ — структура обработки данных, состоящая из множества узлов V и множества ребер E. Граф является направленным, если ребра представлены в виде упорядоченных пар узлов. Дерево преобразования определяется направленным графом, обладающим следующими свойствами: только корневые узлы не имеют входящих в них дуг, в каждый последующий узел входит множество дуг, число которых определяется структурой обрабатываемых данных.

Структура обработки данных отождествляется с направленным графом, в котором узлам соответствуют элементы данных, а направленные дуги, связывающие узлы, описывают различные зависимости между элементами и маркируются соответствующим образом.

Структура обработки данных модели ПИ преобразования $D=\{K,\Psi\}$ определяется множеством K узлов и множеством $\Psi=\{f_1, f_2,...\}$ функций $f_i:K^{\rightarrow}1$ и $1^{\rightarrow}K$, которые отображают множество узлов в один узел и наоборот. Два узла, K и K', связаны дугой f_i, если $K'=f_i(K)$.

Структура обработки данных $D=\{K,\Psi\}$, при которой множество узлов отображается в один узел, т.е. $K^{\rightarrow}1$, а функция Ψ определяется F^*-критерием, образует конвергентную структуру поддерева.

Структура обработки данных $D^*=\{K,\Psi^*\}$, при которой один узел отображается в множество узлов K, т.е. $1^{\rightarrow}K$, а функция Ψ^* определяется Q^*-функцией преобразования, образует дивергентную структуру поддерева.

Из работы [5] следует, что использование ПИ преобразования для различных задач зависит от вида критерия, выбора общей части F^* и соответствующего ему типа Q^*-преобразования. В общем случае критерий выбора общей части может иметь дискриминантный (числовой), рассматриваемый здесь, а также структурный, физический и интеллектуальный смысл. Будем исследовать разновидности дискриминантного Q^*-преобразования, основы теории которого являются определяющими для структурного, физического и интеллектуального Q^*-преобразований. Из полученных массивов на основе F^*-критерия и Q^*-преобразования сформируем новый массив.

Полученный массив назовем массивом второго порядка и преобразуем по такому же алгоритму. Этот процесс преобразования определим как преобразование по вертикали. Если для этих массивов применить тот

же алгоритм, то этот процесс преобразования массивов можно назвать преобразованием по горизонтали. Поэтому все полученные таким образом массивы являются массивами первого порядка. Массивы первого порядка преобразуются до тех пор, пока последние не станут нулевыми, образуя при этом массивы второго порядка.

Массивы второго порядка также преобразуются по критерию общей части F^*. В результате обработки массивов второго порядка получается массив третьего порядка, который преобразуется по описанному выше алгоритму для массива первого порядка. Таким образом, массивы каждого порядка преобразуются также по горизонтали до получения нулевого массива. Этот процесс продолжается до тех пор, пока не образуются нулевые массивы по горизонтали и не сформируется исходная информация для построения новых массивов по вертикали. Массивы, находящиеся на разных уровнях, с течением времени сами преобразуются по горизонтали. Этот процесс назовем эволюцией массива, а преобразование Q^* — оператором эволюции массива.

Свойство 1. Последовательное во времени формирование конвергентных K и дивергентных D поддеревьев образует $K-D$ дерево.

Свойство 2. Соседние $K-D$ деревья одного уровня физически во времени сдвинуты относительно друг друга на одно K и одно D поддеревья. Нелинейная структура $K-D$ деревьев образует обобщенное дерево сети.

Сечениями обобщенного дерева являются одноименные узлы K и D поддеревьев, имеющих одинаковые пути к корневым узлам.

Хвостовыми (результирующими) узлами являются одиночные узлы K поддеревьев обобщенного дерева сетевого преобразования, в сечениях которого находится один узел K поддеревьев.

Сетевое дерево — это конечное множество $K-D$ деревьев, из которых соседние поддеревья одного уровня сдвинуты во времени одно относительно другого на $K-D$ дерево, а число сечений определяется числом хвостовых узлов, положение которых в последовательности сечений имеет вид $(2c+3)$, где c — номер сечения, $c=0,1,\ldots$.

Ветвью сетевого дерева является любое произвольно сформированное в соответствии с предыдущим определением $K-D$ дерево.

Свойство 3. Число уровней сетевого дерева определяется числом его хвостовых узлов, увеличенным на единицу.

Уровни определяются процессами эволюции массивов по горизонтали и переходом к эволюции массивов по вертикали.

Предлагаемый способ параллельного преобразования сигналов рассмотрим с помощью сетевого алгоритма [5, 7, 11], основные свойства которого — параллелизм и иерархия, синхронность и детерминирован-

ность. Сеть состоит из множества конечных множеств Ω, множества элементов A и условно разбита на ряд уровней.

Сеть ПИ преобразования включает совокупность следующих характеристик:

конечное число M множеств на u иерархических уровнях,

$$C \in (\Omega, A, Q^*, F^*) \in \{M_1^1(t_0),\ M_2^1(t_0), ..., M_h^1(t_0),\ M_1^2(t_1), ..., M_n^u(t_s)\},$$

где h — число исходных множеств M, $h \geq 2$; u — порядковый номер уровня, $u \geq 2$; n — порядковый номер u-го уровня, $n \geq 2$; t_s — такт или шаг, на котором сформировалось соответствующее множество на u-м уровне, $s \geq 1$; t_0 — первый или начальный такт, где формируются исходные множества первого уровня;

конечное множество элементов

$$A = \{a_1^1(t_1),\ a_2^1(t_1),\ ...,\ a_h^1(t_1),\ a_1^1(t_3),\ ...,\ a_{n'}^{u'}(t_s)\},$$

где $u' \geq 2$ — порядковый номер уровня; $n' \geq 1$ — порядковый номер множества, которому принадлежит элемент; t_s' — такт, на котором формируется соответствующий элемент.

Множества конечных множеств Ω и множества элементов A пересекаются:

$$\Omega \cap A = \varnothing,\ M_i^j(t_s) = \{a_1^{j-1}(t_s),\ a_2^{j-1}(t_s),\ ...,\ a_k^{j-1}(t_s)\},$$

где $M_i^j(t_s)$ — исходное множество для j-го уровня.

Обозначим F^* критерий выбора элемента из множества $a_i^j(t_s) = F^*[M_i^j(t_{s-1})]$ (переход от множества $M_i^j(t_{s-1})$ к элементу $a_i^j(t_s)$) и Q^* — функцию преобразования множества, $Q^*_{a_i^j(t_s)}[M_i^j(t_{s-1})] = M(t_{s+1})$ (переход от элемента $a_i^j(t_s)$ к множеству $M(t_{s+1})$).

Мощность исходных множеств $M_i^1(t_0)$ обозначим m, а их число — H:

$$M_1^1(t_0) = \{a_{11}, a_{12}, ..., a_{1m}\};$$

$$M_2^1(t_0) = \{a_{21}, a_{22}, ..., a_{2m}\}, ..., M_h^1(t_0) = \{a_{h1}, a_{h2}, ..., a_{hm}\}.$$

Каждое из этих множеств преобразуется по единому сетевому алгоритму, а все множества отрабатываются параллельно. Из множества M_i выбирается один элемент a_i и элемент $a_i^1(t_1) = a_i$ является элементом сети C.

Элемент a_i из множества M выбирается по F^*-критерию, т.е. $a_i = F^*(M)$, $a_i \in M$. С учетом этого выбранного элемента выполняется преобразование данного множества, в результате чего формируется новое множество той

же мощности, в котором все элементы, равные выбранному (если такие имеются), определенным образом отмечены, например обнулены. Если сформируется множество $a_{i1}, a_{i2}, 0, a_{i4}, 0, 0, a_{i7}, ..., 0, a_{im}$, то появится элемент $a_i^1(t_1) = a_{i3} = a_{i5} = a_{i6} = a_{ij} = a_{im-1}$. Такая операция называется Q^*-преобразованием. Q^*-преобразование множества $M = \{a_i\}$, с учетом выбранного элемента $a_i \in M$, определим как преобразование, в результате которого формируется новое множество той же мощности, все элементы которого, равные a_i, и сам этот элемент имеет вид $Q^*_{a_i}(M) = M'$.

Далее, из вновь полученного множества выбирается по F^*-критерию следующий элемент, a_{ij_1}, $a_j^1(t_3) = a_{ij_1}$ $(a_{ij_1} = a_{ij})$, и выполняется Q^*-преобразование. В результате формируется множество, в котором все элементы, равные a_{ij_1}, отмечены. Итерационное преобразование выполняется до тех пор, пока все элементы исходного множества не будут отмечены. Такое множество определяется как нулевое, и дальнейшее его преобразование не выполняется.

Нулевым множеством называется такое множество, все элементы которого в результате Q^*-преобразования отмечены.

Процесс преобразования в нулевое множество назовем процессом сходимости. Очевидно, чем меньше тактов выборки необходимо осуществить до формирования нулевого множества, тем лучше сходимость данного процесса. Такой вид преобразования назовем горизонтальным, или преобразованием ветви.

Последовательность шагов преобразования исходного множества с учетом промежуточных результатов до получения нулевого множества называется ветвью.

Рассматривая все H исходных множеств первого уровня, при выборе элемента $a_j^1(t_j)$, $i = \{1, 2, ..., h\}$, $j = \{1, 3, 5, ...\}$, из каждого множества формируем новые множества, где i — порядковый номер исходного множества; t_j — такт, в котором выбран элемент. При первом выборе элементов из H входных множеств в такте t_1 формируется новое множество

$$M_1^2(t_1) = \{a_1^1(t_1), a_2^1(t_1), a_3^1(t_1), ..., a_h^1(t_1)\}.$$

На втором шаге преобразования t_3 из исходных множеств формируется еще одно множество из H элементов:

$$M_2^2(t_3) = \{a_1^1(t_3), a_2^1(t_3), ..., a_h^1(t_3)\}.$$

Такое преобразование выполняется до тех пор, пока все исходные множества не станут нулевыми.

Множества $M_1^2(t_1), M_2^2(t_3), ..., M_i^2(t_j)$ являются исходными множествами второго уровня. Их также будем преобразовывать по сетевому алго-

ритму до полной сходимости. Тогда преобразование, выполняемое на первом уровне H исходных множеств, элементами которых являются элементы первого уровня, будем считать вторым уровнем.

При преобразовании множеств второго уровня формируются элементы, создающие исходные множества для третьего уровня и так далее, до k-го уровня, на котором элементы уже не создают новое множество. Все преобразования происходят по тактам t_i, $i = 1, 2, 3, \ldots$. В каждом такте для любого уровня происходит выбор элементов из множеств по F^*-критерию или Q^*-преобразование множеств в соответствии с ранее выбранными элементами, что свидетельствует о свойстве синхронности данной сети.

Структура сетевого алгоритма, представляющая собой совокупность множеств $M_j^i(t_k)$ и элементов $a_j^i(t_{k+1})$, приведена на рис. 1. Ориентированные дуги соединяют множества и элементы, при этом некоторые дуги направлены от множеств $M_j^i(t_k)$ к элементам $a_j^i(t_{k+1})$, а другие — от элементов к множествам. Дуга, направленная от множества $M_j^i(t_k)$ к элементу $a_j^i(t_{k+1})$, определяет F^*-критерий выбора элемента, а дуга, направленная от элемента $a_j^i(t_{k+1})$ к множеству $M_j^i(t_{k+2})$, указывает на Q^*-преобразование множества. Дуги являются направленными, следовательно, это ориентированный граф с нулевым множеством $M_2^1(t_s)$.

Каждый элемент, обозначенный на рис.1 знаком $\otimes$, не входит ни в одно множество, так как в данном такте для своего уровня выбран единственным, не участвует в дальнейшей обработке массивов и является ее результатом. Такие элементы назовем хвостовыми, или элементами, формирующими результат.

При Q^*-преобразовании множеств, могут быть отмечены, например, равные элементы. Выделим информацию обо всех равных элементах и их расположении во множестве. Для этого каждому Q^*-преобразованию множества необходимо поставить в соответствие двоичный код, в котором единицы находятся в разрядах, соответствующих позициям равных элементов в множестве. Все остальные разряды кода, соответствующие другим элементам множества, заполняются нулями.

Под теневой маской (далее просто маска) Q^*-преобразования множества будем понимать двоичный код, разрядность которого равна мощности множества, а единицы находятся в разрядах кода, соответствующих местонахождению отмеченных на данном шаге элементов множества. Такие маски формируются для всех промежуточных и нулевых множеств во всех ветвях и на всех уровнях.

Блок-схемы реализуемых алгоритмов цифровой обработки сигналов, соответствующие прямому и обратному ПИ преобразованиям, представ-

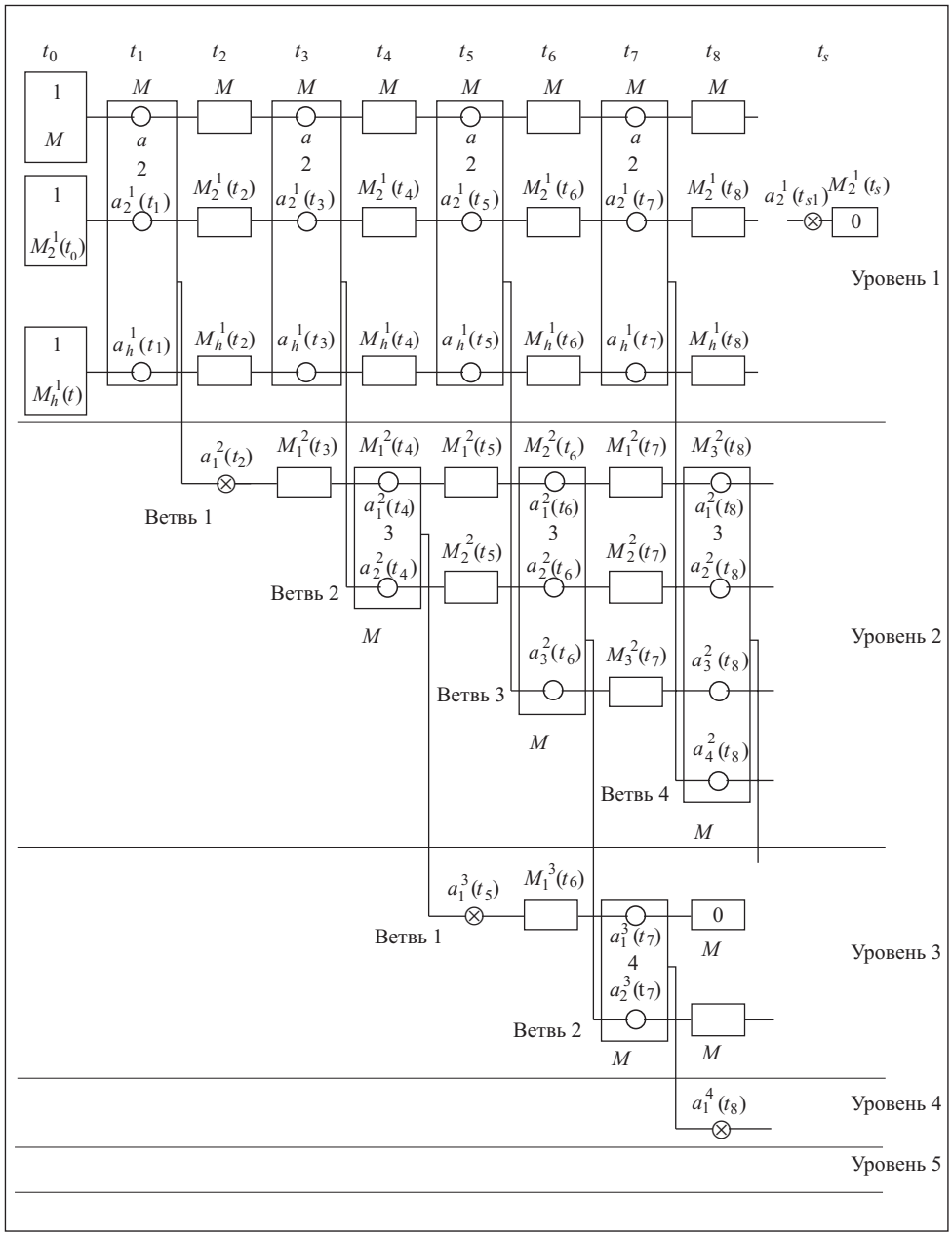

Рис. 1. Структурная схема организации ПИ сети: 1—4 в вертикальных прямоугольниках — исходные множества G-преобразования для формирования ветвей ПИ сети; горизонтальные прямоугольники — промежуточные множества (результат Q^*-преобразования); ○ — элементы; ⊗ — элементы $a_1^2(t_2), a_1^3(t_5), a_1^4(t_8)$; $\boxed{0}$ — нулевое множество

лены на рис. 2, *а* и *б*. Как видно из рис. 2, *б*, в блоках *2* и *3* преобразования происходят параллельно по всем уровням. Если во всех четных уровнях выполняется операция выбора по F^*-критерию, то во всех нечетных уровнях осуществляется Q^*-преобразование, и наоборот. В блоке *4* анализируется наличие одинаковых элементов, или элементов последующих уровней на всех уровнях. На одних уровнях на данном шаге алгоритма такие элементы могут быть, а на других уровнях — отсутствовать. Поэтому из блока *4* возможен выход по двум направлениям одновременно, что следует обязательно учитывать при использовании данного алгоритма.

Для алгоритмов цифровой обработки сигналов приведенная абстрактная ПИ модель является универсальной. Доказательством этого утверждения могут быть результаты экспериментальных исследований параллельной обработки для ПИ преобразования не только сигналов, но и изображений при реализации на различных наборах данных: в виде двумерной матрицы данных разной размерности, а также пятноподобных изображений профиля лазерного луча разной размерности [5—9].

Таким образом, суть ПИ метода заключается в одновременном использовании последовательности множеств массивов информации, образующих множества информационных полей на различных уровнях иерархии, и рекурсивном формировании новых последовательностей информационных потоков на различных уровнях иерархии, что позволяет реализовать стратегию многоуровневого взаимодействия от общего к частному.

Предложенный процесс обработки является пирамидальным. Действительно, в процессе обработки числовой информации с каждым шагом *G*-преобразования, реализуемого в каждой ветви [5], количество чисел уменьшается. Если множества, получаемые после каждого шага, поставить последовательно одно на другое, то образуемый ими трехмерный контур будет иметь форму пирамиды.

Рассмотрим пример пирамидального процесса обработки информации в каждой ветви на основе *G*-преобразования (табл. 1). Пусть входное множество есть M = {2, 3, 5, 7, 4, 5, 4, 4}. Сумма входной числовой информации $\sum_{i=1}^{8} a_i = 34$. На каждом шаге (ступени пирамиды) находится минимальное число. Вычитаем его из каждого числа входного можества чисел, определяя промежуточные результаты: $a_1 = a_{\min} \cdot N$, где N — число ненулевых элементов.

Пирамидальная вычислительная структура на основе ПИ преобразования образует сеть в виде ПИ пирамиды (рис. 3). Для каждой пирамиды используется свой процессорный элемент (ПЭ), а число ПЭ определяется суммарным числом ветвей ПИ сети.

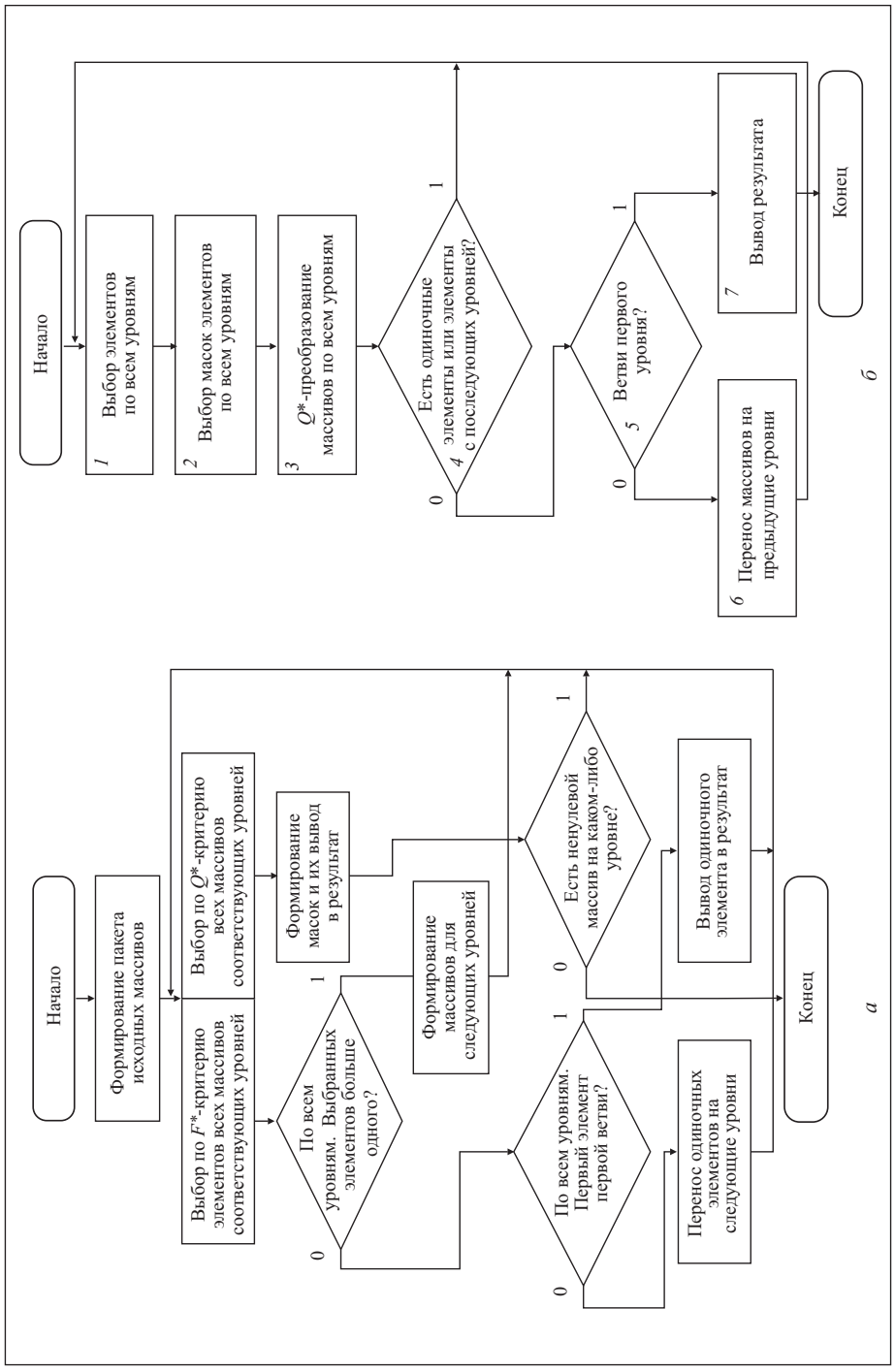

Рис. 2. Схемы алгоритмов прямого (*а*) и обратного (*б*) ПИ преобразований

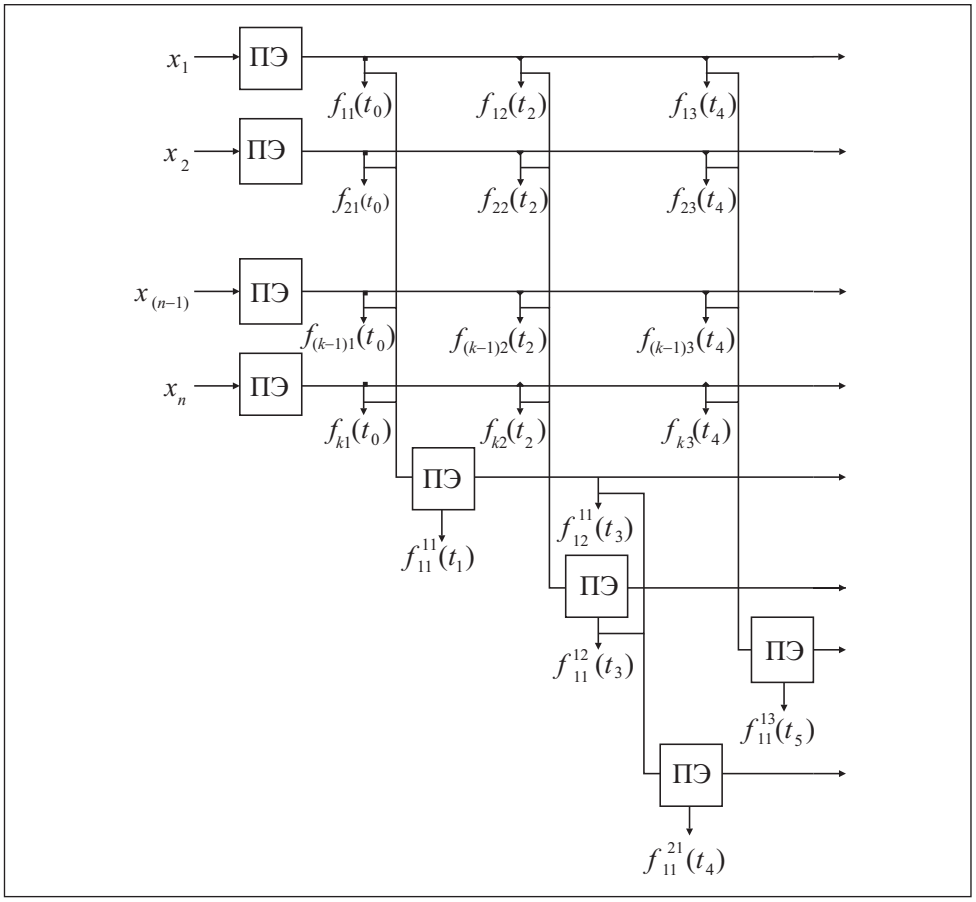

Рис. 3. Структурная схема взаимодействия информационных потоков в сети

Таблица 1

Числовая информация								Промежуточный результат
2	3	5	7	4	5	4	4	$2 \cdot 8 = 16$
0	1	3	5	2	3	2	2	$1 \cdot 7 = 7$
x	0	2	4	1	2	1	1	$1 \cdot 6 = 6$
x	*x*	1	3	0	1	0	0	$1 \cdot 3 = 3$
x	*x*	0	2	*x*	0	*x*	*x*	$2 \cdot 1 = 2$
x	*x*	*x*	0	*x*	*x*	*x*	*x*	$16 + 7 + 6 + 3 + 2 = 34$

Модель сетевого метода можно представить в формализованном виде исходя из следующих положений. Пусть имеется множество потоков входных данных. Возникает следующий вопрос. Как в реальном времени организовать параллельный вычислительный процесс, чтобы получить строго распределенную во времени и иерархии вычислительную сеть?

Ответом на этот вопрос может быть принцип построения следующей сетевой модели. Будем обрабатывать или преобразовывать множество входных потоков данных на различных k иерархических уровнях. Каждый уровень представляет собой совокупность ПЭ, функционирующих в строго фиксированные моменты времени t_j.

Пусть заданы n_1 функций $f_1(t), f_2(t), ..., f_{n_1}(t)$. Данные функции опишем на различных уровнях иерархии их представления от 1-го до j-го ($j=2l,\ l=1,2,...$ и $j=2l+3,\ l=0,1,2,...$):

$$\sum_{j=1}^{n_1}\sum_{i=1}^{n_{j1}} f_{j1}(t-i\tau)=\sum_{j=1}^{n_2}\sum_{i=1}^{n_{j2}} f_{j2}(t-2ij\tau)=\sum_{i=1}^{n_{13}} f_{13}(t-(2i+3)\tau)+$$

$$+\sum_{i=2}^{n_{23}} f_{23}(t-(2i+3)\tau)+\sum_{i=3}^{n_{33}} f_{33}(t-(2i+3)\tau)+...+\sum_{i=n}^{n_{n3}} f_{n3}(t-(2i+3)\tau)=$$

$$=\sum_{i=1}^{n_{14}} f_{14}(t-(2i+6)\tau)+\sum_{i=2}^{n_{24}} f_{24}(t-(2i+6)\tau)+...+\sum_{i=n-1}^{n_{(n-1)4}} f_{(n-1)4}(t-(2i+6)\tau)+$$

$$+\sum_{i=n}^{n_{n4}} f_{n4}(t-(2i+6)\tau)=...=\sum_{i=1}^{n_{12l}} f_{12l}(t-(2i+6l)\tau)+\sum_{i=2}^{n_{22l}} f_{22l}(t-(2i+6l)\tau)+...+$$

$$+\sum_{i=n-1}^{n_{(n-1)2l}} f_{(n-1)2l}(t-(2i+6l)\tau)+\sum_{i=n}^{n_{n2l}} f_{n2l}(t-(2i+6l)\tau)=$$

$$=\sum_{i=1}^{n_{1(2l+3)}} f_{1(2l+3)}(t-(2i+6l+3)\tau)+\sum_{i=n}^{n_{n(2l+3)}} f_{n(2l+3)}(t-(2i+6l+3)\tau), \qquad (1)$$

где τ — задержка формирования последующей функции относительно предыдущей; n_{jk} — число функций j-го разложения k-го функционального уровня.

Анализируя модель сетевого метода (1), можно сделать вывод о том, что в процессе образования каждого уровня в его ветвях формируется временной сдвиг τ, наличие которого приводит к получению хвостовых функций.

Параллелизм ПИ преобразования реализован в ветвях каждого уровня в соответствии с временной координатой по горизонтали, а иерархия — по временной координате со сдвигом на один такт τ по вертикали.

На первом уровне параллельным способом в независимых ветвях выполняется исходное (заданное для конкретной задачи) преобразование и тем самым формируются информационные потоки данных для преобразования на последующих уровнях. На каждом последующем уровне выполняется формирование хвостового элемента.

Это преобразование включает совместное развитие идей параллелизма, иерархии и смешивания при обработке информационных потоков данных, которые затем распространяются по горизонтали на остальные элементы, а по вертикали или в общем случае по произвольному маршруту — на элементы других иерархических уровней. Обеспечение максимальной корреляции элементов ПИ преобразования достигается в результате организации сетевой структуры и многократного смешивания информационных потоков данных на различных уровнях иерархии.

Способы представления масок для реализации многоуровневого преобразования. Рассмотрим несколько способов представления масок и их свойства при реализации ПИ преобразования, влияющие на его характеристики [12].

Для восстановления исходной информации, преобразованной в соответствии с частной методикой ПИ преобразования, в процессе обработки массивов необходимо на каждом шаге преобразования t_i запоминать, на каких позициях в массиве $A_j^{\nu}(t_i)$ (где j — номер массива, ν — номер уровня) находятся элементы, равные элементу $a_j^{\nu}(t_{i-1})$.

Сформируем для этого двоичное слово, разрядность которого равна размерности массива $A_j^{\nu}(t_{i-2})$, а единицы стоят в тех позициях кода, в каких позициях массива находится элемент, равный выбранному. Все остальные позиции двоичного кода заполняются нулями. Этот двоичный код, формируемый на каждом шаге Q^*-преобразования массива, назовем маской: $F_j^{\nu}(t_i)(F_j^{\nu}(t_i))$ — маска массива $A_j^{\nu}(t_i)$ по элементу $a_j^{\nu}(t_{i-1})$. Маски формируются в процессе всего преобразования массива до его полной сходимости на всех уровнях и для всех ветвей. Маски необходимы для процесса декодирования и содержат информацию о том, на какой позиции (позициях) в массиве должен находиться выбранный элемент.

Первый способ представления масок — двоичные слова, размерность которых равна размерности массива. Этот способ формирования масок можно использовать в любых алгоритмах ПИ преобразования. Недостатком такого способа является громоздкость представления масок, но алгоритм их формирования при этом очень прост.

Второй способ представления масок — стековый. Он заключается в том, что масками являются начальные адреса (номера позиций) подмножеств с одинаковыми элементами или непосредственно адрес элемента,

для которого формируется маска. Этот способ позволяет сократить объем представления масок, но требует дополнительных преобразований при кодировании и декодировании массива.

Стековым этот способ называется потому, что при кодировании массива формируемые адреса выбранных элементов записываются по стековому принципу, широко распространенному в устройствах памяти [13, 14]. Декодирование массива со стековыми масками предполагает поэтапное преобразование массива по следующему правилу: из стека выбирается верхний адрес и значения этого адреса, массив заполняется элементами, равными элементу с данным адресом. Заполнение информации осуществляется до той позиции, адрес которой выбран из стека на предыдущих шагах декодирования. Если такой адрес отсутствует, то заполнение выполняется до последнего элемента массива. После того как из стека выбран последний адрес, процесс декодирования заканчивается.

Процессом заполнения массива элементами можно управлять, анализируя значения следующего и заменяемого элементов. Если эти значения равны, то заполняется следующая позиция. В противном случае, когда ранее заполненные элементы в массиве единичные или составляют одиночную группу, следующая позиция не заполняется .

Третий способ представления масок основан на оптимизации явно избыточного первого способа. Избыточность представления масок, разрядность которых равна размерности массива, состоит в том, что те разряды масок, в которых содержится единица, во всех следующих масках заполняются нулями. Такие разряды из следующих масок можно исключить, что и является основой этого способа. При кодировании массива в этом случае каждая последующая маска имеет размерность меньше предыдущей на число единиц в предыдущей маске.

Рассмотрим пример кодирования массива, состоящего из восьми элементов (табл. 2). В данном примере для хранения масок необходима память объемом 18 бит, в то время как для хранения полных масок требуется $5 \cdot 8 = 40$ бит. При этом последнюю маску можно не хранить, так как она состоит только из единиц. При декодировании в предпоследней маске на месте единиц необходимо устанавливать предпоследний выбранный элемент, а на месте нулей — последние выбранные элементы.

Четвертый способ — представление масок логико-временным кодом (ЛВК) [5, 15]. В этом случае каждому элементу соответствует определенный отрезок времени (квант времени) — ЛВК. Если при кодировании массива встречается элемент, равный выбранному, то соответствующий квант времени заполняется импульсом, в противном случае импульс отсутствует. Процесс кодирования масок выполняется параллельно для всех различных элементов массива. Маски ЛВК для массива из табл. 2 представлены на рис. 4.

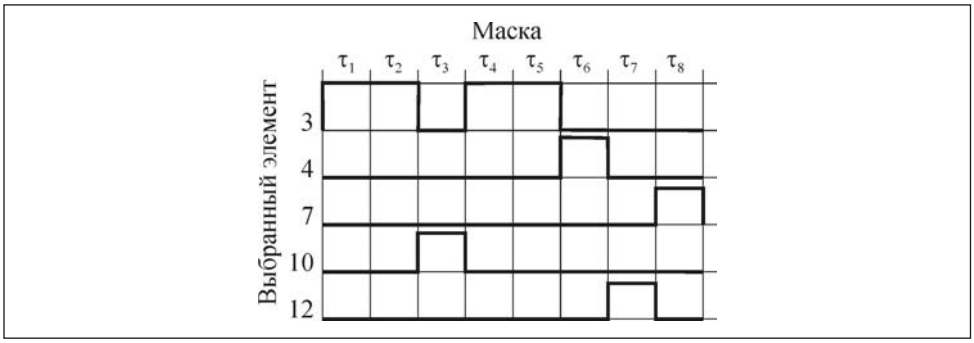

Рис. 4. Представление масок в виде ЛВК

Кроме четырех основных способов представления масок, возможны частные случаи, которые определяются алгоритмами ПИ преобразования. Так, при выборе элементов из массива по порядковому номеру в случае использования третьего способа представления масок в первом разряде всех масок всегда будет стоять единица. Результат разложения массива по такому алгоритму представлен в левой части табл. 3. В качестве исходного выбран массив, представленный в табл. 2.

Из табл. 3 видно, что первые разряды масок содержат единицы. Если выбранный элемент — единственный в массиве, то во всех остальных разрядах маски будут нули, что малоинформативно, и такую информацию при кодировании можно не использовать. В правой части табл. 3 представлен результат разложения массива с учетом изложенного выше. Как видим, маски такого массива сокращаются до одной восьмиразрядной маски.

Возможна ситуация, когда весь массив состоит из одинаковых элементов. Это означает, что маска такого массива только одна и содержит

Таблица 2

Номер элемента	Массив	Маска				
1	3	1	0	0	1	1
2	3	1	1	0	0	
3	10	0	0	1		
4	3	1	0			
5	3	1				
6	4	0				
7	12	0				
8	7	0				
Выбранные элементы		3	4	7	10	12

единицы во всех разрядах. Такую маску можно не запоминать, а запомнить лишь размерность массива и элемент, из которого состоит данный массив. При декодировании массива необходимо учитывать, что если маска отсутствует, то весь массив состоит из одинаковых элементов.

Результаты экспериментальных исследований метода многоуровневого преобразования маскированных сигналов. В экспериментальных исследованиях использован разработанный программный комплекс для реализации прямого и обратного ПИ преобразования, который содержит два программных продукта [16, 17].

1. Программу реализации прямого ПИ преобразования цифровых сигналов с оптимизацией процедуры формирования масок. Применение оптимизированного алгоритма формирования масок, который позволяет уменьшить объем памяти, необходимый для их сохранения, по сравнению с неоптимизированным алгоритмом. При этом одним из этапов ПИ преобразования является умножение минимального элемента на мощность в операторе преобразования G (рис. 5, *а*).

2. Программу реализации обратного ПИ преобразования на основе оптимизированного метода формирования масок для восстановления преобразованных методом прямого ПИ преобразования с оптимизацией процедуры формирования масок информационных сред, представленных в виде двумерной матрицы данных или изображения. Особенностью реализованного алгоритма является модификация процесса декодирования на основе оптимизированного алгоритма работы с масками, которая повышает быстродействие декодирования (рис. 5, *б*).

Таблица 3

Номер элемента	Массив	Маска	Номер элемента	Массив	Маска
1	3	1 1 1 1 1	1	3	1
2	3	1 0 0 0	2	3	1
3	10	0 0 0	3	10	0
4	3	1 0	4	3	1
5	3	1	5	3	1
6	4	0	6	4	0
7	12	0	7	12	0
8	7	0	8	7	0
Выбранные элементы		3 4 7 10 12	Выбранные элементы		3 10 4 12 7

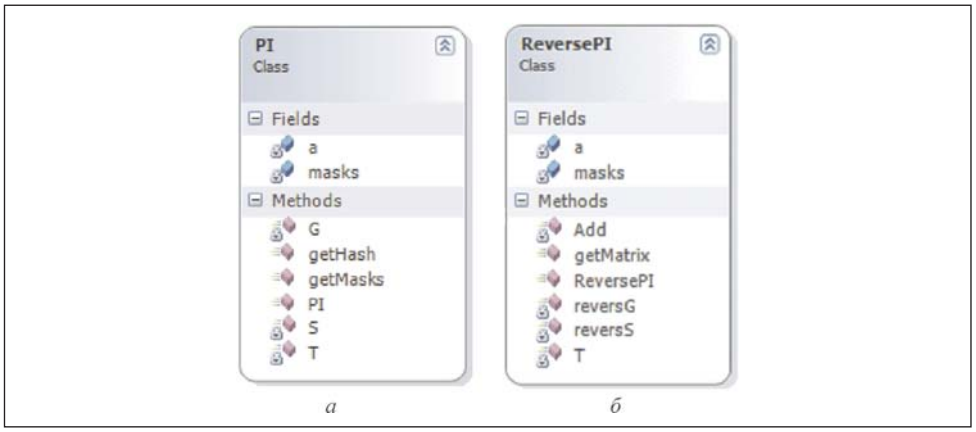

Рис. 5. Фрагменты диаграммы классов программы для реализации прямого (*а*) и обратного (*б*) ПИ преобразования на основе оптимизированного метода формирования масок

Язык реализации программного комплекса — C++. Функции программной библиотеки, после перекомпиляции, корректно работают с различными операционными системами: MS Windows, GNU/Linux, Mac OS.

Основные этапы реализации прямого ПИ преобразования с оптимизацией процедуры формирования масок следующие.

1. Загрузка информационного массива (в виде изображения или двумерной матрицы данных, заданной пользователем размерности). Формирование множеств $\{M_1^1(t_0), M_2^1(t_0), ..., M_h^1(t_0)\}$ (см. рис. 1).

2. Выполнение прямого ПИ преобразования с оптимизацией процедуры формирования масок над данными (изображением). Последовательное применение трех операторов $\Phi(M) = T[S(G(M))]$ (см. рис. 5, *а*).

2.1. Транспонирование (см. рис. 5, *а*, метод T). Переход от эволюции массивов по вертикали к эволюции массивов по горизонтали (см. рис. 1).

2.2. G-преобразование (см. рис. 5, *а*, метод *G*). Эволюция массивов по горизонтали (см. рис. 1).

2.3. Сдвиг (рис. 5, *а*, метод *S*); запоминание одномерной матрицы хвостовых элементов преобразованного информационного массива (см. рис. 5, *а*, метод getHash). В пределах одного уровня ветви сдвинуты во времени одна относительно другой на один такт (см. рис. 1).

3. Формирование одномерной матрицы оптимизированных масок (см. рис. 5, *а*, метод getMasks).

4. Ведение файла-протокола.

Основные этапы реализации обратного ПИ преобразования на основе оптимизированного масочного метода следующие.

1. Загрузка массива хвостовых элементов (в виде одномерного набора данных, полученных при кодировании методом прямого ПИ преобразования с оптимизацией процедуры формирования масок).

2. Загрузка массива оптимизированных масок (в виде одномерного набора данных (состоящего из 0 и 1), полученных при кодировании методом прямого ПИ преобразования с оптимизацией процедуры формирования масок).

3. Выполнение обратного ПИ преобразования на основе оптимизированного метода масок над заданными данными (см. рис. 5, *б*, метод ReversePI).

4. Восстановление начального массива данных (в виде двумерной матрицы данных или изображения (см. рис. 5, *б*, метод getMatrix)).

Разработанные программы позволяют загрузить изображение, или задать размеры матрицы для прямого и обратного ПИ преобразования, и заполнить заданную матрицу собственноручно или с помощью генератора псевдослучайных чисел. После этого для заданного массива данных выполняется ПИ преобразование, выводятся значения хвостовых элементов, их сумма и сумма входной матрицы (указанные суммы, согласно теории ПИ преобразования, должны совпадать), а также время обработки и обобщенный размер масок.

Разработанный программный комплекс для реализации прямого и обратного ПИ преобразования сигналов протестирован на различных наборах данных: в виде двумерной матрицы данных разной размерности, а также пятноподобных изображений профиля лазерного луча разной размерности. В тестовом примере осуществлена реализация прямого ПИ преобразования с оптимизацией формирования масок над цветным пятноподобным изображением профиля лазерного луча (в формате RGB размерностью 128 × 128 пикселей) и выполнено обратное ПИ преобразование на основе оптимизированного метода масок для его восстановления.

Полученные результаты экспериментальных исследований (и тестового примера) свидетельствуют о повышении быстродействия ПИ преобразования по следующим критериям:

быстродействие обработки прямым ПИ преобразованием неоптимизированным методом — 984 мс, оптимизированным методом —782 мс;

быстродействие обработки обратным ПИ преобразованием неоптимизированным методом — 1250 мс, оптимизированным методом — 422 мс;

уменьшение избыточности объемов памяти при формировании массивов масок неоптимизированным методом — 25 057 800 бит, оптимизированным методом — 10 187 072 бита.

Выводы

Предложенный метод позволяет повысить быстродействие прямого и обратного ПИ преобразования, а также оптимизировать избыточность представления масок, разрядность которых равняется размерности обрабатываемых цифровых сигналов.

Выбранный в результате сравнительного анализа метод представления масок для оптимизации обратного ПИ преобразования изображений приводит к существенному уменьшению объемов памяти, необходимой для сохранения масок, а также имеет свойства выявления ошибок кодирования.

Результаты экспериментальных исследований подтвердили эффективность предложенных алгоритмов прямого и обратного ПИ преобразования цифровых сигналов и возможность применения их в прикладных задачах кодирования, обработки и сравнения изображений. Полученные результаты могут найти применение в дальнейших исследованиях по разработке высокопроизводительных многоуровневых параллельно-иерархических сетей на основе оптоэлектронной и оптической элементной базы.

An analysis of structural-functional organization of the network architecture of parallel-hierarhical transformation and mask generation methods has been performed. The method of optimized forming of masks by encoding information in parallel-hierarchical transformation is proposed. The software package with high speed performance of lossless direct and reversal parallel-hierarhical transformation of information has been developed.

СПИСОК ЛИТЕРАТУРЫ

1. *Прэтт У.* Цифровая обработка изображений. В 2-х томах. — М. : Мир, 1982. — Т. 1. — 310 с., Т. 2. — 790 с.
2. *Гренандер У.* Лекции по теории образов. Анализ образов. — М. : Мир, 1981. — 448 с.
3. *Скляр Б.* Цифровая связь. Теоретические основы и практическое применение. Изд. 2-е испр. : Пер. с англ. — М. : ИД «Вильямс», 2003. — 1104 с.
4. *Сергиенко А.Б.* Цифровая обработка сигналов: Учеб. пособие. Второе изд. — СПб. : Питер, 2006. — 752 с.
5. *Кожем'яко В.П., Кутаєв Ю.Ф., Свєчніков С.В. та ін.* Паралельно-ієрархічне перетворення як системна модель оптико-електронних засобів штучного інтелекту. — Вінниця : УНІВЕРСУМ-Вінниця, 2003. — 324 с.
6. *Yarovyy A.A.* Applied Realization of Neural Network and Neurolike Parallel-Hierarchical System Based on GPGPU. Development and application systems// Proc. of the 10th Intern. Conf. on DAS-2010. May 27—29, 2010, Suceava.— Romania: Suceava, Universitatea Stefan cel Mare Suceava, 2010. — P. 351—356.
7. *Kozhemyako V., Timchenko L., Yarovyy A.* Methodological Principles of Pyramidal and Parallel-Hierarchical Image Processing on the Base of Neural-Like Network Systems // Advances in Electrical and Computer Engineering. — 2008. — Vol. 8 (15), № 2 (30). — P. 54—60.
8. *Kozhemyako V.P., Timchenko L.I., Yarovyy A.A.* Software Support of Accurately Measurement and Prediction of Laser Beam Profile Characteristics// Proc. of the 10th Intern. Conf. «Swiatiowody i ich zastosowania», October 4—7, 2006, Krasnobryd, Poland. Tom 2 — Lublin, Wydawnictwo-Drukarnia Liber Duo s.c., 2006. — P. 675—684.

9. *Timchenko L., Kutaev Yu., Kozhemyako V. et al.* Method for Processing of Extended Laser Paths Images // Advances in Electrical and Computer Engineering. — 2003. — Vol. 3 (10), № 2 (20). — P. 66—78.
10. *Кнут Д.* Искусство программирования для ЭВМ. В 3-х т. — М. : Мир, 1978.
11. *Timchenko L., Kutaev Yu., Kozhemyako V. et al.* Method for Training of a Parallel-Hierarchical Network, Based on Population Coding for Processing of Extended Laser Paths Images // Proc. of SPIE. — 2002. —Vol. 4790. — P. 465—479.
12. *Яровий А.А.* Метод оптимізованого формування масок при кодуванні інформації в паралельно-ієрархічному перетворенні // Вісник ВПІ. — 2011. — № 6 (99). — С. 216—223.
13. *Брюхович Е.А.* Автоматический контроль и производительность ЭВМ. // УСИМ. — 1979. — № 4. — С. 83—86.
14. *Метлицкий Е.А., Каверзнев В.В.* Системы параллельной памяти: теория, проектирование, применение. Под ред. В.И. Тимохина. — Л. : Изд. Лениградского университета, 1989. — 240 с.
15. *Кожемяко В.П., Тимченко Л.И., Яровой А.А.* Модели параллельно-иерархической обработки информации на основе аппарата логико-временных функций// Тезисы докл. Междунар. науч. конф. «Информационные и компьютерные технологии, моделирование, управление». Грузия, Тбилиси: 1—4 ноября 2010 г. — Тбилиси : Изд-во ГТУ, 2010. — С. 231—232.
16. *Свідоцтво* про реєстрацію авторського права на твір № 39490. Комп'ютерна програма прямого паралельно-ієрархічного перетворення з оптимізацією формування масок (із множенням мінімального елемента на потужність в операторі перетворення G). / А.А Яровий, І.М. Сугак — Зареєстровано ДДІВ України 04.08.2011.
17. *Свідоцтво* про реєстрацію авторського права на твір № 39489. Комп'ютерна програма зворотного паралельно-ієрархічного перетворення на основі оптимізованого масковового методу (із множенням мінімального елемента на потужність в операторі перетворення G). / А.А.Яровий, І.М. Сугак — Зареєстровано ДДІВ України 04.08.2011.

Поступила 21.06.12;
после доработки 12.11.12

ТИМЧЕНКО Леонид Иванович, д-р техн. наук, профессор, зав. кафедрой телекоммуникационных технологий и автоматики Государственного экономико-технологического университета транспорта. В 1979 г. окончил Винницкий политехнический ин-т. Область научных исследований — системы искусственного интеллекта.

ЯРОВОЙ Андрей Анатольевич, канд. техн. наук, докторант, доцент кафедры компьютерных наук Винницкого национального технического университета, который окончил в 2001 г. Область научных исследований — параллельные вычисления, нейроподобные интеллектуальные системы, распознавание образов и обработка изображений.

КОКРЯЦКАЯ Наталья Ивановна, канд. техн. наук, доцент кафедры телекоммуникационных технологий и автоматики Государственного экономико-технологического университета транспорта. В 1973 г. окончила Винницкий педагогический ин-т. Область научных исследований — математическое моделирование параллельных процессов.