МЕТОДЫ СИНТЕЗА КАРТОГРАФИЧЕСКОГО ФОНА, БЫСТРОГО ИЗМЕНЕНИЯ МАСШТАБА КАРТЫ И ЭШЕЛОНИРОВАННОГО СКРОЛЛИНГА

Бородин В.А., Васюхин М.И.
Институт кибернетики им. В.М.Глушкова НАН Украины, г.Киев
vas.geovideo@naverex.kiev.ua

Соболев О.Н.
ЦНИИ ВВС ВС Украины, г.Киев

Abstract

Borodin V.A., Vasjuhin M.I., Sobolev O.N. Methods of synthesis of cartographical background, fast change of map scale and echelon scrolling. *The method of creation and display of a map for real time geoinformational complexes (GC RT) on the basis of the microprogrammed controller is described. The method of a map preparation, which allows to carry out fast echelon scrolling and scale change of a map is offered.*

Задача быстрого изменения масштаба карты и эшелонированного скроллинга является характерной для геоинформационных комплексов реального времени (ГК РВ). Успешному её решению предшествует ряд операций синтеза определенных картографических срезов, которые составят картографический фон. Существенной особенностью ГК РВ является, кроме обычных мониторов, входящих в состав системы, наличие экрана (экранов) коллективного пользования. Такой экран может иметь значительные размеры, но главной особенностью его должна быть возможность реализации "электронного кино", т.е. умение воспроизводить динамические сцены в виде движущихся объектов на картографическом фоне [1-3].

Главной трудностью синтеза и отображения таких изображений на экране является резкое увеличение скорости движения таких объектов (крылатых ракет, самолетов и т.д.) и, как следствие, ужесточение требований к программно-техническим методам и средствам, осуществляющим преобразования данных изображений, параллельному переносу, операциям изменения масштаба и поворота. Чтобы успеть обеспечить отображение динамической сцены в виде быстро движущихся объектов, представленных сложными символами на фоне карты с одновременным перемещением картографического фона с помощью операций изменения масштаба и эшелонированного скроллинга, важно производить эти преобразования как можно быстрее, чтобы освободить процессор для решения других задач [3].

Предлагается метод создания и отображения карты, при котором сначала создается программно-техническая среда, удовлетворяющая целям построения ГК ОВ. В качестве такой среды используется система на основе отечественных программно-аппаратных средств, состоящая из микропрограммируемого контроллера - МПК [4], совместно с видеоконтроллером – А554-1 [5], осуществляющая генерацию статического цветного изображения (топографической карты) и вывод его на экран видеотерминала. Организация работы данной системы отличается тем, что после операции представления карты операция вывода изображений сложных символов увязывается с предшествующей как программно, так и аппаратно. В качестве языка программирования использован язык МИКРОКОД. Достоинством описанной среды является то, что МПК может выполнять функции как АЛУ, так и геометрического процессора.

Формат управляющих команд МПК приведен в табл.1. МПК выполняет 9 типов микрокоманд, в числе которых 4 типа операционных микрокоманд: ОП1, ОП2, ОП3, ОП4; засылки константы ЗКН; условного перехода – УПУ; трех типов команд ввода-вывода – ВВ1, ВВ2, ВВ3. Установлено, что с помощью МПК достаточно удобно адресовать данные в видеоконтроллер и формировать команды вывода видеоинформации на экран.

Видеоконтроллер, управляемый МПК (при коде выборки 1П), обеспечивает прием следующей информации в зависимости от типа управляющей команды:

С1 – адрес начала генерации. Это адрес первой выбираемой ячейки буферной памяти при выводе информации на экран (если регенерация видеоинформации на экран осуществляется с нулевого адреса буферной памяти, то код команды будет равен 100000);

С2 – масштаб по горизонтали (x) и вертикали (y) (масштаб мог изменяться от 1 до 16, если масштаб изображения не изменяется, то код команды будет 40000);

С3 – задание величины заголовка. В макете комплекса под заголовок отводится до 31 первых строк экрана. Если код заголовка задается командой 20000, то текстовая информация выдается в размере этих строк. Такой величины заголовка оказалось достаточно для вывода номенклатурного номера листа карты.

При этом отметим, что при коде выборки 2П видеоконтроллер обеспечивает:

С9 – адресацию ячейки ОЗУ функциональной памяти (ФП) для записи кода полутона (если запись начинается с 0-адреса ОЗУ ФП, то код команды равен 100000);

С10 – задание кода полутона, который получается смешиванием основных цветов, представляемых сигналами R,G,B.

Формат управляющих команд **Таблица 1**

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
	--								Код выборки 1							
	1		Адрес начала регенерации													
С1	0	М	0						Мх				My			
С2	0	0	З						Заголовок (количество строк)							
С3	--								Код выборки, 0							
	1			Адрес ячейки буферной памяти (старшие разряды)												
С4	0	1	0	Г	С				Адрес ячейки буферной памяти (младшие разряды)							
С5																
С6	0	0	0	ДС					Дг							
С7	0	0	1		Число повторов Дс или Дг											
С8	0	1	1	Наклон				Обр.	Образец							
	--								Код выборки, 2							
С9	1	М1	У/о	Р2	Р1										Адрес пр.	
С10	0	М2	У/о			Дпр.				Дпр.					Дпр.	

Где ▮ - неиспользованные разряды;

С – тип управляющей команды; Адрес Пр.-- адрес ячейки ОЗУ ФП; Дг - данные записываемые в графическую память; Дпр. – данные записываемые ОЗУ ФП; Дс - данные, записываемые в символьную память; З - признак заголовка; М - признак масштаба; М1,М2 - признаки мерцания; Мх,Му - коды масштабирования по координатам Х и У соответственно; Обр. – признак образца; Г - признак выбираемой графической памяти; С - признак выбираемой символьной памяти; Р - признак регенерации; Р1,Р2 - признаки редактирования; У/о - признак У/о.

В командах С9 и С10 задаются также признаки мерцания как отдельных контуров элементов изображения, так и составляющих цвет полутонов.

Выдачу информации из ОЗУ ФП на шины ИУС видеоконтроллер обеспечивает при коде выборки 2И.

При коде выборки ОП видеоконтроллер обеспечивает запись видеоданных в символьную или графическую часть буферной памяти (БП).Запись информации в буферную память производится по адресу, заданному в управляющих командах С4 и С5. Этот адрес соответствует адресу точки экрана, начиная с которой вырисовывается топографическая карта. Поскольку в качестве инструментария среды был построен макет с емкостью экрана 10^5 адресуемых цветных точек, с условием, что каждой ячейке буферной памяти соответствует определенная точка экрана, то для адресации одной ячейки буферной памяти требуется 17-разрядный код, 13 разрядов которого описываются командой С4, а 4 разряда – командой С5. Описание точки экрана, например, с адресом 54430_{10}(x=30, y=170) или 152236_8 будет представлено этими командами (запись в графическую часть буферной памяти) следующим образом: С4 – 106511, С5 – 50016.

Видеоданные описываются командой С6 и представляют собой номер ячейки ОЗУ ФП, в которой описан код полутона, задаваемого при построении контура изображения на экране.

Запись информации в буферную память осуществляется с учетом следующих трех признаков: признака повторов, признака наклонов, признака образца.

Признак образца определяет однократную или многократную запись видеоданных в буферную память по адресу, указанному в командах С4 и С5. Количество повторов указывается в команде С7. После записи первой порции видеоданных, видеоинформация будет записываться в ячейки буферной памяти в соответствии с признаком наклона (команда С8). Признак образца (команда С8) представляют собой «маску» для записываемой видеоинформации. Наличие «1» в любом из восьми разрядов образца означает, что видеоданные записываются в буферную память, а «0» – что в текущую ячейку буферной памяти (в соответствии с признаком наклона) записывается «0».

Так строится "электронное" изображение карты в буферной памяти, которое по коду выборки ОИ с помощью видеоконтроллера выводится на экран.

Таким образом, суть метода построения топографической карты в буферной памяти состоит в следующем. При коде выборки 1П выполняются команды С1, С2, С3 (блок управления), при коде выборки 2П производится загрузка ОЗУ ФП (команды С9 и С10). При коде выборки ОП выполняются команды С4, С5, С7 и С8, которые описывают порядок формирования изображений в буферной памяти.

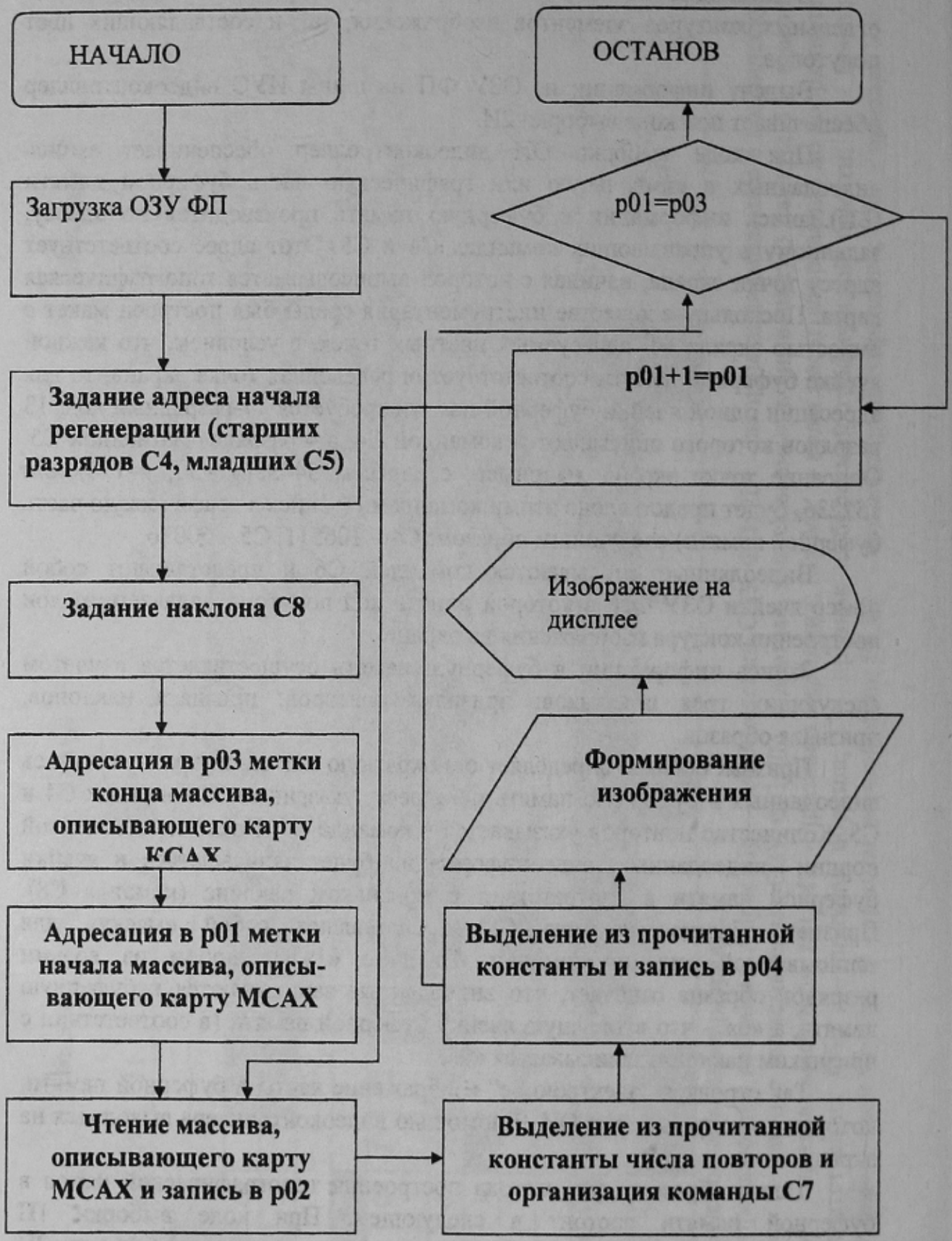

Рисунок 1- Алгоритм построения топографической карты в БП

Затем следует адресация модуля индикации как внешнего устройства-приемника (код выборки ОП), после чего выполняется команда С6 (блок формирования изображения).

Алгоритм построения топографической карты, рис.1, состоит в том, что сначала производится загрузка функциональной памяти (ФП), при этом описываются коды полутонов, на основе которых строится изображение карты (блок 1). Затем следует задание всех необходимых команд, предшествующих выдаче данных: адреса точки экрана (команды С4 и С5), с которой собственно и начинается описание карты (блок 2); наклона (С8), которые определяют порядок заполнения ячеек буферной памяти (блок 3). Адрес конца ранее созданного массива данных, описывающих карту (КСАХ) заносится в регистр р03, а адрес начала (МСАХ) – в р01 (блоки 4 и 5). Затем читается первая запись массива данных, и результат чтения заносится в регистр р02 (блок 6).

Каждая запись массива данных представляет собой константу, описывающую адрес кода полутона из ОЗУ ФП, а также количество точек экрана, которым присваивается цвет этого полутона. Например, КОН 140050 расшифровывается следующим образом: 14 – адрес кода полутона из ОЗУ ФП, а 0050 – количество точек экрана раскрашиваемых в цвет этого полутона. Следующим шагом (блок 7) является выделение из константы в регистре р02 количества повторов и формирование команды числа повторов (С7).

Далее из константы в регистре р02 (блок 8) выделяются данные (адрес кода полутона) и осуществляется переход к формированию изображения карты (блоки 9 и 10).

Для перехода к чтению следующей записи массива данных, проверяется равенство значений адресов начала (МСАХ) и конца (КСАХ) массива (блок 11). Если они равны, то считается, что карта построена и осуществляется переход к блоку ОСТАНОВ, а если нет, то наращивается адрес массива МСАХ на 1 (блок 12) и осуществляется переход к чтению следующей записи (блок 6).

Вместе с тем для значительного класса изображений, не сильно насыщенных утилитами, такой путь не является достаточно эффективным. Так простые схемы, графики и т.п., с успехом можно представить с помощью векторного метода.

Предложен алгоритм построения карты с помощью векторного метода, рис.2. Он начинается с операции загрузки ОЗУ ФП (блок 1), при этом необходимо привести экран в исходное состояние (в исходном состоянии весь экран – “черный”) – алгоритм, представленный подпрограммой очистки ОЧЭК [6]. Далее, рис.2, в регистр р02 засылается число выдаваемых векторов - счетчик векторов (блок 3), а в регистр р04 - число параметров, описывающих каждый вектор (блок 4).

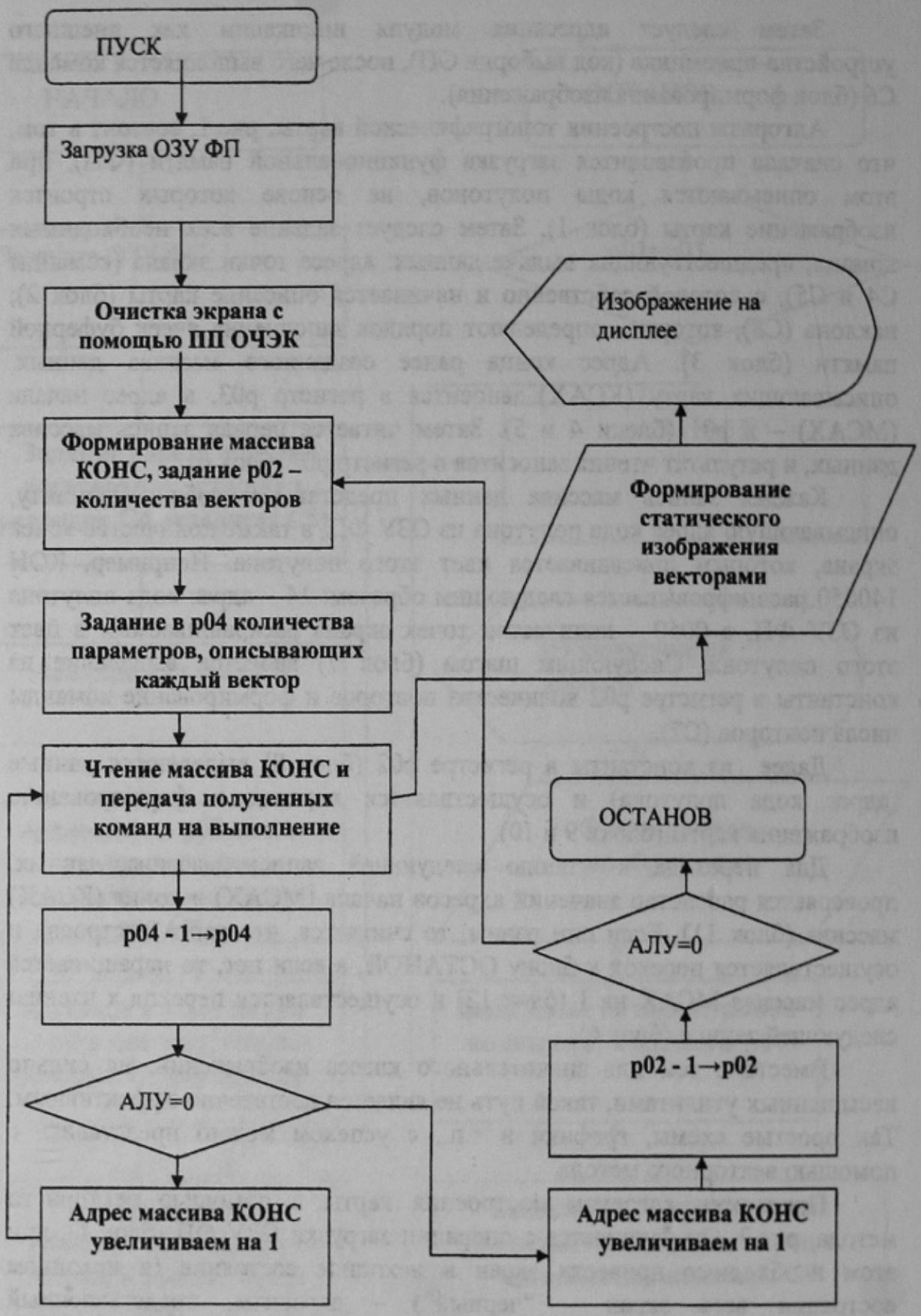

Рисунок 2 - Алгоритм построения карты векторным методом

Причем параметры всех векторов представляются в массиве КОНС так, что «X» обозначает номер выдаваемого вектора, а константы являются командами С4, С5, С7, С8 и С6 соответственно.

При считывании массива КОНС осуществляется формирование изображения (блоки 6 и 7). После того, как первая константа массива КОНС прочитана, из содержимого регистра р04 вычитается «I» (блок 8) и проверяется, все ли управляющие команды для построения одного вектора прочитаны (блок 9).

Если «нет», то адрес массива КОНС увеличивается на «1» (блок 10) и осуществляется его повторное чтение (блок 5). Если же все команды прочитаны и выполняется условие АЛУ=0 («да»), то адрес массива КОНС увеличивается на «1» (блок 11), но при этом содержимое счетчика векторов уменьшается на «1» (блок 12) и проверяется, все ли вектора построены (блок 13). Если «да», то осуществляется переход к команде ОСТАНОВ, а если «нет», - к блоку задания числа параметров следующего вектора (блок 4). Если «да», то осуществляется переход к команде «ОСТАНОВ», а если «нет», - к блоку задания числа параметров следующего вектора (блок 4).

Предложен метод быстрого изменения масштаба карты и эшелонированного скроллинга.

В качестве примера рассмотрим задачу, возникающую в тренажерах для пилотов самолетов, когда на экране тренажера отображается карта местности, над которой «летит» пилот, и нужно подвергать это изображение карты таким преобразованиям, которые обеспечили бы по ходу движения «самолета» необходимый просмотр местности, который может быть получен необходимым масштабированием и проведением скроллинга.

Для выполнения данной задачи необходимо также производить операцию изменения масштаба при моделировании эффекта увеличения высоты полета [1-3].

Так, когда карта дана в виде файла в некотором растровом формате, соответствующем изображению из $n \times m$ пикселей, то в процессе ее отображения необходимо преобразовать в изображение с площадью $[\alpha n] \times [\alpha n]$ пикселей. Возможно также растяжение по одной из координат на некоторый коэффициент $\frac{\beta}{\alpha}$, и тогда полученное изображение будет $[\alpha n] \times [\beta m]$ пикселей.

Для осуществления этой операции следует положить пиксель с координатами (i, j), $i, j \in N$ равным среднему арифметическому от цветов первоначального растрового изображения попавших в интервал $(\alpha i, \alpha(i+1)) \times (\beta j, \beta(j+1))$, и при этом пиксели, которые попали в этот

интервал не полностью, а с частью площади s суммируются с цветом, умноженным на коэффициент s [7].

Следовательно, количество операций, необходимых для вывода карты в измененном масштабе, а, значит, и создание нового растрового изображения с количеством пикселей $[\alpha n] \times [\beta m]$, будет превышать количество пикселей в первоначальном растровом файле карты - nm.

Для экономии используемой памяти компьютера при хранении карты и времени на обработку и изменения масштаба достаточно приемлемым является способ сжатия изображений на основе квадратомических деревьев [8,9], однако эффективность его применения имеет место только при изменении масштаба на величину вида 2^n и не дают существенного эффекта при сжатии только по одной из координат.

Поэтому нашей целью явилось нахождение такого метода, который бы позволял совершать операцию изменения масштаба и растяжения изображение за количество операций, соизмеримое с количеством пикселей, составляющих изображение карты в новом масштабе, т.е. $[\alpha n] \times [\beta m]$, а также более экономно выполнял для нее операцию скроллинга или параллельного переноса. Для этого предлагается подобрать такой метод сжатия при хранении карты, который, с одной стороны, позволял бы сберегать ее описание так, чтобы карту всегда можно было бы отображать корректно (т.е. хранить первоначальное растровое отображение), а с другой стороны, позволял бы быстро получать изображение карты в новом масштабе $[\alpha n] \times [\beta m]$ для произвольных $\alpha, \beta \in [0,1]$.

Заметим, что при уменьшении масштаба мелкие детали, то есть те области одного цвета, площадь которых в новом масштабе становится меньше единичного пикселя нового масштаба, становится невидимыми. Следовательно, для уменьшения масштаба достаточно сохранять информацию о тех областях, площади которых сопоставимы с единичным пикселем в новом масштабе.

Возможен, однако, случай, когда в единичном прямоугольнике нового масштаба содержится несколько более мелких участков одного цвета.

В этом случае, с точки зрения оператора, не важно, какой цвет присвоится новому пикселю – можно, например, положить его равным цвету наибольшему из этих участков, т.к. именно такое решение удовлетворяет практическим требованиям для данной задачи [2].

Суть предлагаемого метода изменения масштаба карты состоит в следующем. Пусть карта представлена в виде матрицы $S[i,j]_{i=1,n;j=1,m}$. Для каждого массива $a[j] = S[k,j]_{j=1,m}$ (для всех k) выделим отрезки одного

цвета и зафиксируем их края A_{kl}, т.е. номера элементов $a[j]$, где эти участки заканчиваются.

Рассмотрим сжатие изображения $S[i,j]_{i=1,n;\,j=1,m}$ на коэффициент α по оси ординат. Заметим, что при изменении масштаба по оси ординат участки однородности для каждого из отрезков $a[j]$ сохраняются. Тогда концы отрезков этих участков однородности для каждого k будут αA_{kl}. Заметим, что те участки однородности, чья длина l такова, что $\alpha l < 1$ не попадают на отображения в новом масштабе.

Таким образом, получаем новые массивы чисел A'_{kl}, которые соответствуют новым цветам для карты в новом масштабе. Этот метод работает при сжатии в α раз вдоль оси координат Oy. Аналогичный метод можно осуществить и для сжатия в β раз вдоль оси координат Ox. Для этого создаются массивы чисел B_{pl}, которые хранят координаты концов однородных участков вдоль линий $b[j] = s[i,p]_{i=1,m}$. Операция изменения масштаба производится аналогично предыдущему случаю.

Предлагаемый метод хранения картографических данных, эффективный для выполнения операций изменения масштаба, использует известный принцип векторизации [1-3]. Однако в описанных в литературе системах этот принцип не используется для ускорения работы преобразований масштаба.

Предложенный метод хранения графической информации можно описать следующим образом. Для каждого цвета Ц (номер этого цвета положим k) выделим те участки, которые этим цветом окрашены, т.е. те односвязные фрагменты изображения, которые имеют цвет Ц. Выделяем границы этих областей - координаты $x_1[i], x_2[i]$ и $y_1[j], y_2[j]$, где $x_1[i]$ - это левая x-координата границы одноцветной области цвета Ц для i-го столбца, в котором содержится цвет Ц, а $x_2[i]$ - соответственно правая x-координата. Аналогично $y_1[j]$ - это нижняя y-координата границы одноцветной области цвета Ц для j-й строки, в которой содержится цвет Ц, а $y_2[j]$ - соответственно верхняя y-координата. Запишем, соответственно, в файл, описывающий изображение карты, номер цвета Ц и координаты $(x_2[i], y_2[i])$ для всех i, которые нумеруют эти граничные точки.

Тогда для воспроизведения этого изображения, нужно пройти цикл по всем одноцветным участкам и закрасить их все необходимым цветом.

Оценим эффективность применения предложенного метода.

При хранении изображения в формате BMP (BitMapPointer) используется $3nm$ единиц памяти, поскольку в нем для каждого элемента

растрового массива изображения хранятся координаты (x,y,Col), где x - x-координата точки, а y - y-координата точки, Col - цвет данной точки.

Для хранения изображения предложенным методом требуется $3p$ единиц памяти, где p - это количество точек изображения, которые являются граничными слева и снизу (т.е. какая-то из соседних точек $(x+1,y)$ или $(x,y+1)$ имеет цвет, отличный от цвета (x,y)). Таких точек p, очевидно, не больше nm. Проведенные нами исследования показывают [9], для большинства карт такое число p существенно меньше nm, а в среднем равно $nm/2$.

Для выполнения операции сжатия или растяжения по какой-либо из координат на коэффициент α требуется $\frac{p}{2}$ арифметических операций умножения, поскольку достаточно умножить на α соответствующие координаты граничных точек, что даст новые граничные точки для изображения с измененным масштабом.

Следовательно, для изменения масштаба по данному методу требуется количество операций, меньшее, чем количество пикселей в растровом изображении карты nm.

Если выполнять операцию скроллинга как операцию переноса для каждой точки, то потребуются $2S$ арифметических операций при условии, что S - количество пикселей экрана.

Возможность экономии вычислительных ресурсов при выполнении скроллинга для карты, упакованной предложенным методом, заключается в том, что при передвижении карты не все точки экрана изменили свой цвет.

Так, переместим изображение произвольной карты, представленной в виде матрицы (например, изображение карты, представленной на рис 3.) $S[i,j]$ (растровая форма) на один пиксель вправо. Получим матрицу $S'[i,j]$.

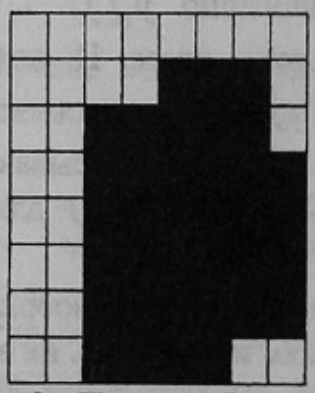

Рисунок 3 - Черная область на карте 8×8

Отметим, что не все элементы матрицы $S'[i,j]$ отличаются от соответствующих элементов $S[i,j]$. Действительно, пусть $x[i],y[i]$

массив координат точек, которые лежат на границе некоторой односвязной и одноцветной области на карте. Тогда если переместить только их на 1 пиксель вправо, т.е. выполнить преобразование $x[i]+1, y[i]$, получится матрица $S'[i,j]$. Заметим, что достаточно переместить только те элементы $x[i], y[i]$, для которых точка $x[i]+1, y[i]$ не будет лежать в той же одноцветной области, что и точка $x[i], y[i]$.

Аналогично происходит перемещение на 1 пиксель и в любом другом направлении. Таким образом, алгоритм параллельного переноса можно записать и для произвольного переноса на $k \in Z$ пикселей вправо и $l \in Z$ пикселей вверх следующим образом:

Для всей карты, которую нужно отобразить на экране выделяется массив тех точек, которые перемещаются при изменении на 1 пиксель вправо - $x_1(i), y_1(i)$, влево $x_2(i), y_2(i)$, вверх $x_3(i), y_3(i)$, вниз $x_4(i), y_4(i)$. Эти массивы длиной соответственно m_1, m_2, m_3 и m_4.

Когда карта переместилась на $k \in Z$ пикселей вправо и $l \in Z$ пикселей вверх, то достаточно k раз переместить точки из $x_1(i), y_1(i)$ если $k>0$ или $-k$ раз точки из $x_2(i), y_2(i)$ если $k<0$, а также переместить l раз точки из $x_3(i), y_3(i)$ если $l>0$ или $-l$ раз точки из $x_4(i), y_4(i)$ если $l<0$, т.е. осуществить следующий цикл:

For j=1 to |k| do begin

For all i

$$S'[x_{(3-\operatorname{sign} k)/2}(i)+\operatorname{sign} k, y_{(3-\operatorname{sign} k)/2}(i)] = S[x_{(3-\operatorname{sign} k)/2}(i), y_{(3-\operatorname{sign} k)/2}(i)]$$

end;

For j=1 to |l| do begin

For all i

$$S'[x_{(7-\operatorname{sign} l)/2}(i), y_{(7-\operatorname{sign} l)/2}(i)+\operatorname{sign} l] = S[x_{(7-\operatorname{sign} l)/2}(i), y_{(7-\operatorname{sign} l)/2}(i)]$$

end;

Таким образом, выполняется $|k| m_{(3-\operatorname{sign} k)/2} + |l| m_{(7-\operatorname{sign} l)/2}$ операций присвоения. Отметим, что если осуществлять скроллинг карты с помощью параллельного переноса, то нужно совершить S операций присвоения (здесь S - количество пикселей экрана). Следовательно, при $|k| m_i + |l| m_j < S$ для параллельного переноса на k пикселей вправо и l пикселей вверх необходимо меньшее количество операций, чем при осуществлении параллельного переноса.

Легко заметить, что количество операций в предложенном алгоритме скроллинга будет $p \cdot (|k|+|l|)/2$. Следовательно, поскольку p можно в среднем положить $S/2$ [9], а скроллинг осуществляется на не более 1 пиксель по горизонтали и 1 пиксель по вертикали при описанном методе

количество операций при скроллинге будет вдвое меньше, чем при алгоритмах параллельного переноса.

Таким образом, предлагаемые методы позволяют существенно сэкономить вычислительные ресурсы при построении топографической карты растровым и векторным методами, что позволило осуществлять подготовку картографических данных для проведения быстрого изменения масштаба карты и эшелонированного скроллинга в реальном времени. Время преобразований карты, обеспечивающий эшелонированный скроллинг значительно сокращается за счет того, что предлагаемый метод переноса осуществляется за 1 операцию для каждого пикселя, а не за 2 операции, какие используются в известных методах параллельного переноса. Кроме того, предлагаемый метод хранения картографической информации обеспечивает экономию памяти в среднем на 50%, при этом сокращается на 75% количество арифметических операций при изменении масштаба, что позволяет выполнять эшелонированный скроллинг за вдвое меньшее количество операций, чем при выполнении его обычными методами.

Литература

1. Кошкарев А.В., Каракин В.П. Региональные геоинформационные системы.- М.: Наука, 1987.- 126 с.
2. Беляев А.Г. Метод визуализации аэронавигационной информации в системе ARGON// Математические машины и системы. - Киев, 2001.- №1-2..- С.106-113.
3. Васюхин М.И., Смолий В.В. Система визуализации тренажеров диспетчеров воздушного и морского пространства // Вестник ХГТУ.- Херсон.- 1999.- № 1(5).- С.126-127.
4. Контроллер микропрограммируемый А135-1. Руководство по эксплуатации. Часть 1. Техническое описание и инструкция по эксплуатации. 3.057.172 РЭ, 1979 .- 121 с.
5. Контроллер графический телевизионного индикатора А554-1. Руководство по эксплуатации. 3.044.005 РЭ, 1982.- 105 с.
6. Васюхин М.И. Алгоритмические и программно-аппаратные методы и средства построения интерактивных геоинформационных комплексов оперативного взаимодействия Дис… д-ра техн. наук: 05.13.13/Институт кибернетики НАН Украины – К., 2002.- 414с.
7. Бутаков Е.А., Островский В.И., Фадеев И.Л. Обработка изображений на ЭВМ. – М.: Радио и связь, 1987. – 240 с.
8. Hanan Samet, The quadtree and related hierarchical data structures, Computing Surveys, Vol.16, No 2, June 1984, p. 187-260.
9. Васюхін М.І., Головко Б.Б., Бородін В.А. Оцінка ефективності застосування принципу квадротомічного дерева при формуванні БД ГІС для растрових зображень // Вісник геодезії та картографії.- Київ, 2001.- № 1.- С.37-39.

Дата надходження до редколегії: 20.12.2003 р.