Заметки к курсовой работе по теме «Автоматическое составление схемы для вышивки крестом по заданному изображению» студента группы ПВ-231, Чупахиной Софии (1-ая аттестация).

Для выполнения стандартных действий при работе с изображениями: изменения пропорций, смещения цветовой гаммы, пикселизации или размытия изображения — существуют готовые алгоритмы решения. Проблемы, с которыми приходится сталкиваться в реальной жизни, часто содержат их в себе в качестве подзадач. Однако бывает и так, что подзадача оказывается модифицированным вариантом одной из типовых задач такого плана, и применить стандартные алгоритмы не получится.

Рассмотрим пример из жизни. Среди любителей рукоделия популярна вышивка крестом — в немалой степени из-за простоты этого вида вышивки. Что имеется в виду под «простотой»? Изображение в вышивке крестом (по большей части) составляется из отдельных крестиков разных цветов, расположенных вдоль прямоугольной сетки с постоянным размером и квадратной формой ячейки. То есть этот способ составления изображения схож с растровым типом графики. Это сходство само по себе наводит на мысль, что растровое изображение можно легко перевести в схему для вышивки крестом и так же легко автоматизировать этот процесс составления схем.

Определим **цель работы**: написать программу, которая составляет схему для вышивки крестом по заданному изображению и дополнительным параметрам.

Решается ли поставленная цель какой-либо стандартной функцией? Какого плана обработка требуется растровому изображению, чтобы оно могло стать схемой для вышивки? Начать стоит с уменьшения количества пикселей. Изображение с самым низким (из стандартных) разрешений, 240*320 пикселей, при переводе 1 пикселя в 1 крестик образует схему для весьма внушительного панно — стоит ли говорить о разрешениях, где общее количество точек измеряется в мегапикселях?.. Стандартная процедура пикселизации вполне себе решает эту задачу, существенно уменьшая разрешение — объединяя соседние пиксели в квадратные области одинакового цвета вдоль сетки, укрупненной относительно изначальной.

Но после этого шага мы столкнемся с потребностью сократить количество цветов. Оставлять изображение в прежней палитре (а по умолчанию в форматах .png и .jpg доступно более 16000000 цветов), бессмысленно, потому что пользователь никогда не подберет нитки для воспроизведения этих цветов с такой точностью. Все это многообразие должно быть сведено к нескольким десяткам цветов. А пикселизация с фиксированным количеством цветов — уже не такая тривиальная задача.

Учитывая сказанное выше, подытожим: программа должна принимать на вход графический растровый файл и параметры схемы (требуемые разрешение схемы в крестиках и количество цветов), а возвращать растровое изображение, разбитое на пиксели заданного количества цветов в соответствии с указанными размерами; плюс измененное в целях читаемости схемы (что такое читаемость схемы, рассмотрим позднее).

Задача, поставленная нами, весьма конкретна, и можно составить примерное словесное описание алгоритма, создающего схему:

- Принимает на вход графический растровый файл и введённые пользователем данные:
  - Требуемая длина схемы в крестиках (вводится только длина или только ширина, а

оставшийся параметр рассчитывается из пропорций самого изображения, либо вводятся оба параметра, и изображение обрезается, чтобы его соотношение длины к ширине было равно соответствию введенных длины и ширины);

  - Количество цветов, из которых вышивка будет состоять;

- Выделяет в данном изображении n цветов, условно говоря, встречающихся наиболее часто. Всегда ли цветами, которые стоит выделить, будут именно самые частые цвета, рассмотрим ниже;

- Разбивает изображение на квадраты-пиксели в соответствии с заданной длиной/шириной и присваивает каждому квадрату один из выбранных шагом ранее основных цветов, в зависимости от цветов оригинального изображения, используемых в этом квадрате, и их близости к выбранным;

- Сохраняет как отдельные растровые файлы один из вариантов либо два варианта схемы:

  - Изображение, разбитое на различимые квадраты-пиксели (разделенные относительно тонкой сеткой) тех цветов, которые наиболее часто встречаются в исходном изображении и должны будут использоваться в вышивке;

  - Изображение, разбитое на различимые квадраты-пиксели, цвета которых могут быть изменены с целью "читаемости"(если в схеме используются два близких оттенка одного цвета, на втором варианте схемы один из них может быть смещен в какой-то другой цвет, чтобы пользователь в дальнейшем их не путал) и с наложенным поверх каждого квадрата значком, отдельным для каждого цвета (точка, меньший квадрат, треугольник, пара полос и т. д.) с той же целью;

- Сохраняет в отдельный файл или выводит на экран нужную для использования схемы информацию: соответствие реальных цветов с их обозначениями на второй схеме, названия/номера используемых цветов в одной или нескольких существующих палитрах ниток.

Опираясь на этот план, можно выделить наиболее объемные и проблематичные подзадачи, хотя не факт, что такое деление сохранится при переходе к практической части.

✓Проблема выбора основных цветов.

Может показаться, что достаточно отсортировать цвета по количеству пикселей, приходящихся на них в изображении, и взять несколько первых позиций. Но всегда ли приведет ли к желаемому результату такой подход? Допустим, если изображение представляет собой относительно небольшие пятна разных групп цветов на однотонном фоне, то может оказаться, что наиболее часто встречающимися цветами будут близкие оттенки цвета фона, и ни один из цветов самих объектов учтен не будет. Существует ли готовое решение для этой проблемы? Возможно, стоит ввести какое-то минимальное расстояние между выбираемыми цветами. Возможно, стоит анализировать не только количество пикселей определенного цвета, но и то, является ли это количество максимальным в каком-то диапазоне близких к рассматриваемому цветов, и искать таким образом кластеры цветов. *Последний подход, если окажется верным, может оказаться базой для введения более сложных функций: например, поиска кластеров, областей цветов уже в самом изображении, а не в распределении цветов по количеству пикселей, и настройка пользователем «цельности» этих кластеров в формируемой схеме — либо*

же настройка чувствительности в обнаружении «кластеров по частоте» для получения схем с разными палитрами.

✓Проблема работы с большими массивами.

В ряде библиотек для работы с растровыми графическими файлами они преобразуются в массив, где каждый пиксель — числовое значение, кодирующее цвет этого пикселя. Тогда для фотографии разрешением, к примеру, 960*1280 количество элементов в таком массиве будет превышать миллион. Не станет ли такое количество элементов препятствием для быстрого выполнения программы, особенно при использовании более сложных схем выбора основных цветов? Если да, то существуют ли готовые алгоритмы, ускоряющие работу над массивами, представляющими изображение, в целом или выполнение функций, которые мы в общих чертах описывали выше?

✓Проблема создания понятного интерфейса

Напрямую эта подзадача не относится к основной; условно можно назвать ее проблемой «фронтенда», в то время как рассматриваемые выше пункты — проблемы «бэкенда». Программы и библиотеки, создаваемые нами в рамках предмета ОП на первом курсе, были больше похожи на черные ящики: единственным элементом интерфейса была консоль, отображающая стандартные потоки ввода-вывода; более сложные типы данных, чем отдельные числа и строки, приходилось вписывать непосредственно в файл программы, а если программа предусматривала создание и редактирование файлов, оно производилось в автоматическом режиме. Для результата курсовой работы хотелось бы иметь более удобный графический интерфейс: отдельное окно, в котором отображались бы исходный и результирующий файл, панель с настройкой основных параметров схемы с помощью ввода чисел или передвижения ползунков, меню с основными функциями программы (добавление нового файла, преобразование текущего, сохранение результата, возможно, открытие предыдущих файлов и так далее).

Можно ли назвать тему этой работы **актуальной**? С одной стороны, для составления схем вышивки крестом существует весьма большое количество готовых программ. С учетом этого тема может показаться исчерпанной, а решение, написанное в рамках данной курсовой — вторичным. Однако можно попытаться, помимо основного функционала, добавить более сложные настройки (примеры мы приводили, когда говорили о проблеме выбора цветов), посмотрев, таким образом, с новой стороны даже на относительно простую проблему. Ну и нельзя не отметить, что решение этой задачи даёт набор навыков и инструментов для решения других задач программной обработки изображения похожей сложности, открывает путь к использованию этих решений в составе более сложных проектов. Актуальность этих навыков не утратится со временем, что дает нам право назвать в каком-то роде актуальной и выбранную тему.