Лабораторная работа № 1 «Векторная графика» в OpenOffice.org

Цель работы**:** изучение технологии создания векторных графических изображений

1. Краткие теоретические сведения

В настоящее время существует два класса программных средств, которые используются в области цифровой графики. Эти классы программ выделяются в зависимости от способа кодирования графической информации.

Действительно, цифровая графика бывает двух видов:

1) пиксельная графика (bitmapedmages, scannedimages, rasterimages)представляет собой совокупность дискретных элементов, которые различаются только цветом (тоном) и взаимным расположением.

2) векторная графика (vectorDrawing, vectorillustration) представляет собой линейно-контурное изображение, которое состоит из независимого описания границ векторных объектов и их заполнения ("заливок").

Для векторной графики характерно разбиение изображения на ряд графических примитивов – точки, прямые, ломаные, дуги, полигоны. Таким образом, появляется возможность хранить не все точки изображения, а координаты узлов примитивов и их свойства (цвет, связь с другими узлами и т. д.).

Рассмотрим изображение, приведенное нарис.1.

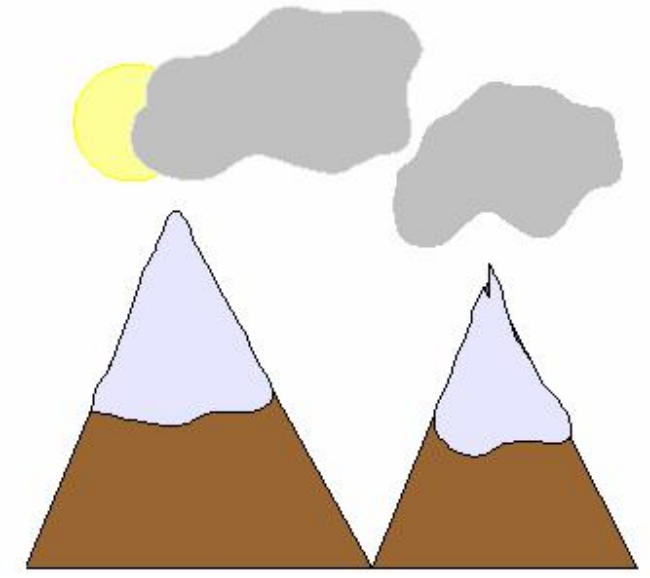


Рис. 1

На изображении легко можно выделить множество простых объектов - отрезки прямых, ломанные, эллипс, замкнутые кривые. Представим себе, что пространстворисунка существует в некоторой координатной системе.

Тогда можно описать это изображение, как совокупность простых объектов, вышеперечисленных типов, координаты узлов которых заданы вектором относительно точки начала координат (рис. 2).

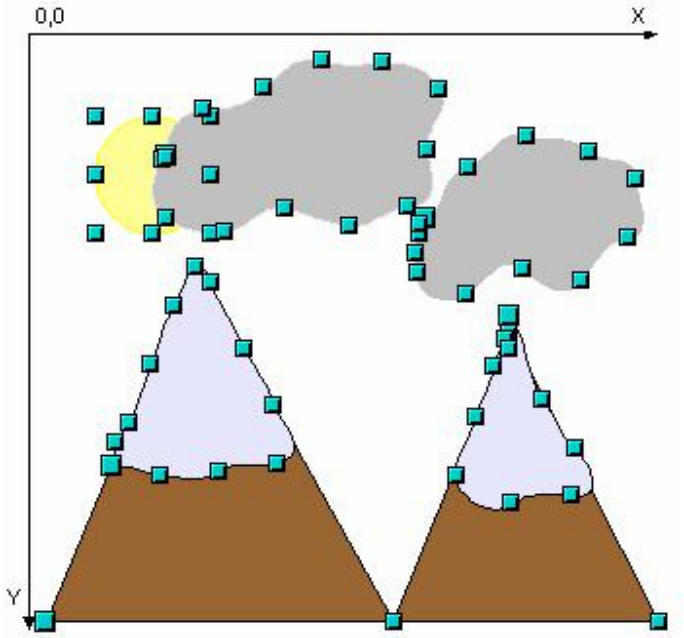


Рис.2

Например, чтобы компьютер нарисовал прямую, нужны координаты двух точек, которые связываются по кратчайшей прямой. Для дуги задается радиус и т. д. Таким образом, векторная иллюстрация – это набор геометрических примитивов.

Векторные изображения формируются из объектов *(*точка*,* линия*,* окружность*,* прямоугольник и пр*.),* которые хранятся в памяти компьютера в виде графических примитивов и описывающих их математических формул.

Поэтому векторную графику еще именуют как объектно-ориентированную (object-orientedgraphics).

Важной деталью является то, что объекты задаются независимо друг от друга и, следовательно, могут перекрываться между собой.

Таким образом, при использовании векторного представления изображение хранится в памяти как база данных описаний примитивов и параметров макета (размеры холста, единицы измерения и т. д.).

К основным графическим примитивам, используемым в векторных графических редакторах, относятся: точка, прямая, кривая Безье, эллипс (окружность), полигон (прямоугольник) и др.

Например, графический примитив точка задается своими координатами (X,Y), линия – координатами начала (X_1,Y_1) и конца (X_2,Y_2), окружность – координатами центра (X,Y) и

радиусом (R), прямоугольник – координатами левого верхнего угла (X_1,Y_1) и правого нижнего угла (X_2,Y_2) и так далее. Для каждого примитива задаетсятакже цвет.

Примитив строится вокруг его узлов (nodes) (рис. 3).

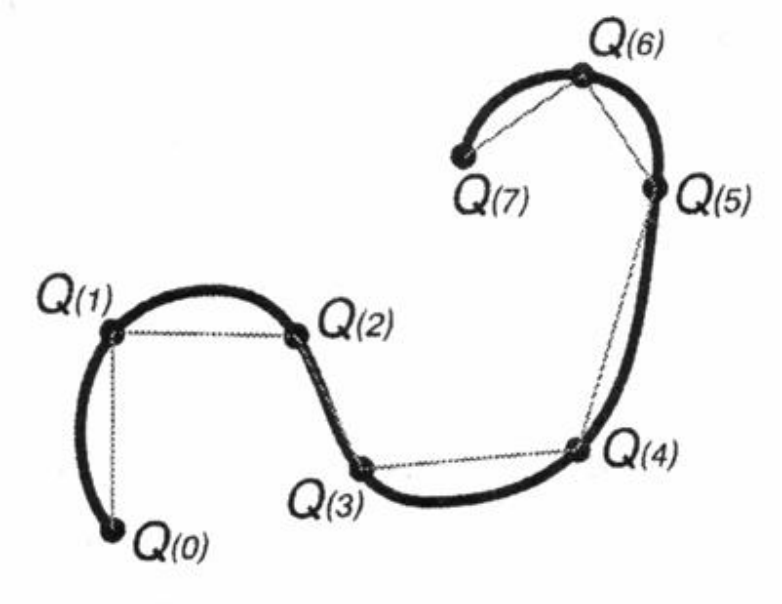


Рис. 3

Координаты узлов задаются относительно координатной системы макета. А изображение будет представлять из себя массив описаний типа:

отрезок (20,20-100,80);

окружность(50,40-30);

кривая_Безье(20,20-50,30-100,50).

Каждому узлу приписывается группа параметров в зависимости от типа примитива, которые задают его геометрию относительно узла.

Такой набор параметров, которые играют роль коэффициентов и других величин в уравнениях и аналитических соотношениях объекта данного типа, называют аналитической моделью примитива.

Отрисовать примитив – значит построить его геометрическую форму по его параметрам согласно его аналитической модели.

Векторное изображение может быть легко масштабировано без потери деталей, так как это требует пересчета сравнительно небольшого числа координат узлов.

Следует заметить, что векторным можно назвать только способ описания изображения, а само изображение для нашего глаза всегда растровое. Таким образом, задачами векторного графического редактора являются растровая прорисовкаграфических примитивов и предоставление пользователю сервиса по изменению параметров этих примитивов.

Отрисовать изображение – значит выполнить последовательно процедуры прорисовки всех его деталей.

Самой простой аналогией векторного изображения может служить аппликация. Все изображение состоит из отдельных кусочков различной формы и цвета (даже части растра), «склеенных» между собой. Понятно, что таким образом трудно получить фотореалистичное изображение, так как на нем сложно выделить конечное число примитивов. Поэтому векторные изображения выглядят искусственно, а их авторы весьма ограничены в средствах живописи.

Однако существенными достоинствами векторного способа представления изображения, по сравнению с растровым, являются:

1) векторное изображение может быть легко масштабировано без потери качества, так как это требует пересчета сравнительно небольшого числа координат узлов;

2) графические файлы, в которых хранятся векторные изображения, имеют существенно меньший, по сравнению с растровыми, объем (порядка нескольких килобайт);

3) рисовать в векторных графических редакторах быстро и просто, при этом возможно независимое редактирование частей рисунка;

4) средства векторных графических редакторов отличает высокие скорость выполнения операций и точность прорисовки (до 1 000 000 точек на дюйм).

Сферы применения векторной графики очень широки. В полиграфии – от создания красочных иллюстраций до работы со шрифтами. Все, что мы называем машинной графикой, 3D-графикой, графическими средствами компьютерного моделирования и САПР – все это сферы приоритета векторной графики, ибо эти ветви дерева компьютерных наук рассматривают изображение исключительно с позиции его математического представления.

Векторные графические изображения являются оптимальным средством хранения высокоточных графических объектов (чертежи, схемы и пр.), для которых имеет значение сохранение четких и ясных контуров.

В данной лабораторной работе мы рассмотрим возможности векторного графического редактора *OpenOffice.org Draw* в создании и редактировании различных блок-схем.

OpenOffice.org Draw позволяет создавать рисунки различной сложности иэкспортировать их с использованием нескольких общепринятых форматов изображений. Кроме того, можно вставлять в рисунки таблицы, диаграммы, формулыдругие элементы, созданные в программах OpenOffice.org.

Объекты векторной графики создаются в OpenOffice.org Draw с использованием линий и кривых панели Рисование, определенных с помощью математических векторов (рис. 4).

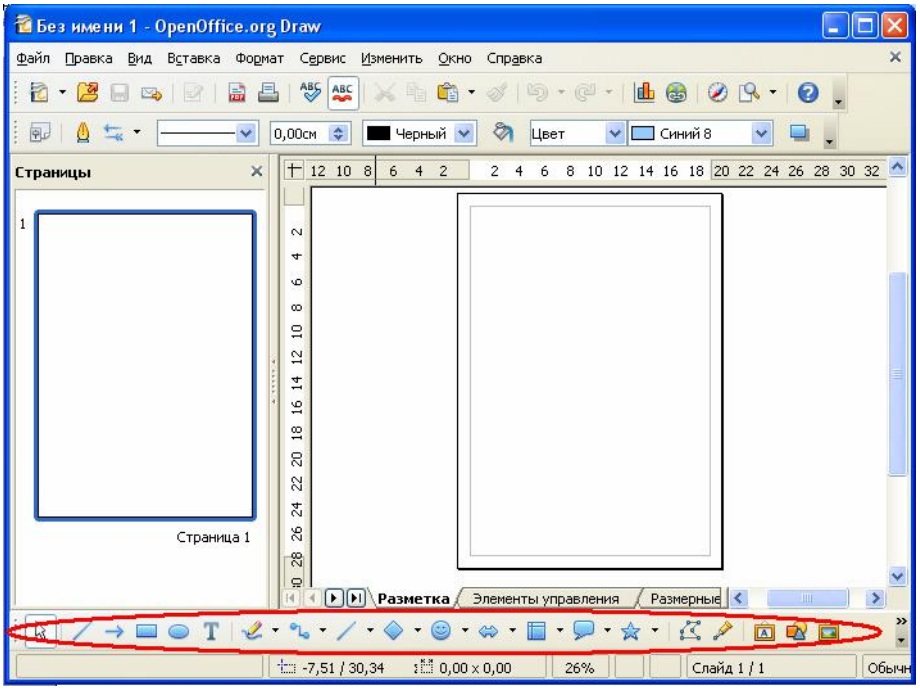


Рис. 4

Векторы описывают линии, эллипсы и многоугольники в соответствии с их геометрией (рис. 5).

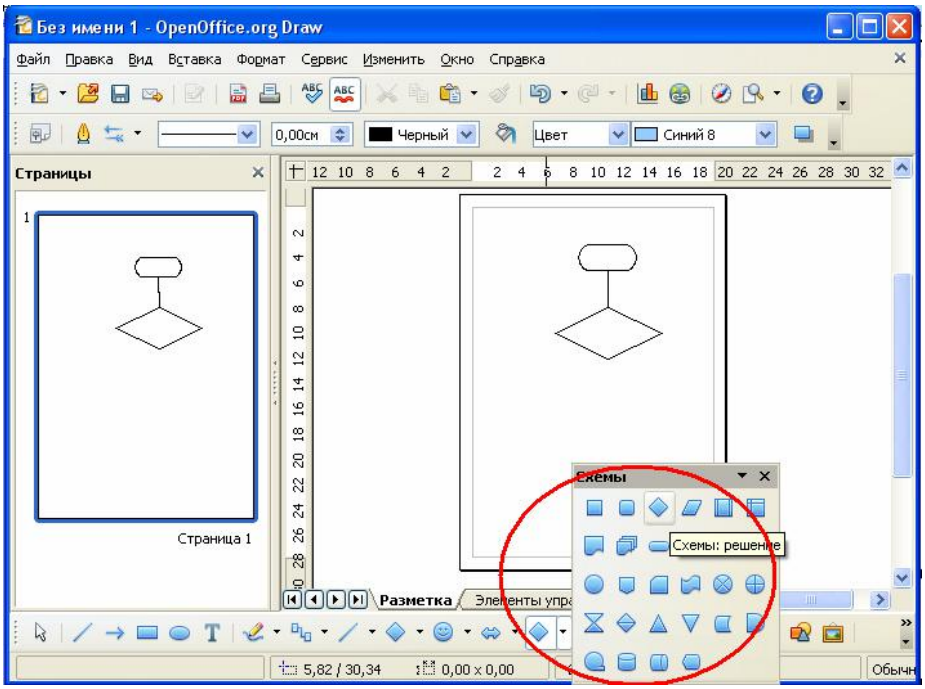


Рис.5

Объекты OpenOffice.org Draw могут быть связаны специальными соединительными линиями для отображения отношений между объектами. Эти линии прикрепляются к точкам соединения на рисованных объектах и перемещаются вместе с ними. Соединительные линии полезны при создании организационных и технических диаграмм.

Открыв панель инструментов Соединительные линии (рис. 6), можно добавлять в слайд соединительные линии объектов.

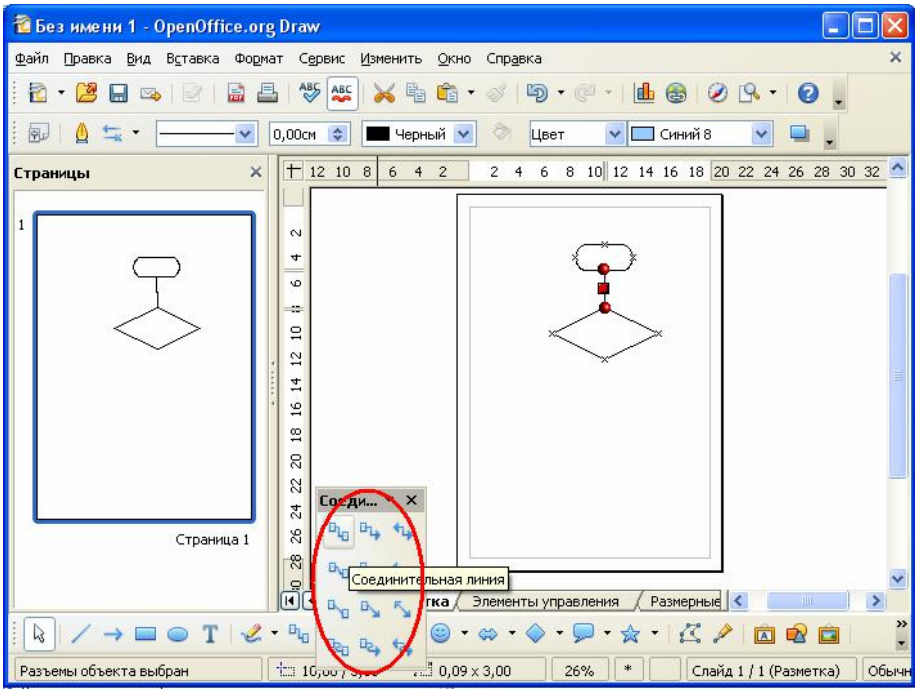


Рис. 6

Существуют четыре типа соединительных линий:

1) стандартная (с поворотами на 90 градусов);
2) подогнутая (из двух звеньев);
3) прямая;
4) сглаженная.

Если щелкнуть соединительную линию и переместить указатель в объект с заливкой или к краю объекта без заливки, появятся точки соединения.

Точка соединения - это фиксированная точка объекта, к которой можно подвести соединительную линию. При необходимости можно добавить в объект свою точку соединения или изменить ее свойства с помощью инструментов панели Точки соединений.

Программа OpenOffice.org Draw позволяет сохранять и экспортировать файлы в различных графических форматах, напримерODF, WMF, BMP, GIF, JPG, PNG и др.

2. Порядок выполнения работы

2.1. Задание

При выполнении работы необходимо:

1) создать в среде графического редактора OpenOffice.org Draw блок-схему алгоритма решения задачи в соответствии с вариантом;

2) сохранить в файл, содержащий блок-схему, в формате рисунка ODF (.odg);

3) экспортировать полученный файл рисунка в графический формат JointPhotographicExpertsGroup (.jpg);

4) подготовить отчет по результатам выполнения лабораторной работы и ответить на контрольные вопросы.

2.2. Технология работы

Чтобы создать блок-схему:

1) выберите нужный инструмент панели инструментов Блок-схемы на панели Рисование;
2) нарисуйте фигуру на слайде, перетаскивая курсор;
3) чтобы добавить другие фигуры, повторите предыдущие шаги;
4) откройте панель инструментов Соединительные линии на панели Рисование и выберите нужную соединительную линию;
5) наведите указатель на край фигуры, чтобы отобразились точки соединения;
6) щелкните точку соединения, перетащите указатель на другую фигуру и отпустите кнопку мыши;
7) чтобы добавить другие соединительные линии, повторите последние шаги.

Чтобы добавить текст в фигуры блок-схемы выполните одно из следующих действий:

1) дважды щелкните фигуру и введите или вставьте текст;
2) щелкните значок Текст на панели Рисование и перетащите текстовый объект на фигуру. Введите или вставьте текст в текстовый объект.

Чтобы добавить цвет заливки для фигуры:

1) выделите фигуру, а затем последовательно выберите команды Формат – Область;
2) выберите команду Цвет, а затем щелкните цвет в списке.

3. Контрольные вопросы

1) Чем отличается векторная графика от пиксельной (растровой)?
2) Приведите определение графического примитива.
3) В каком виде хранится векторное изображение в памяти компьютера?
4) Что называют аналитической моделью примитива? Приведите примеры таких моделей.
5) Что значит «отрисовать примитив» и «отрисовать изображение»?
6) Какие основные задачи решает векторный графический редактор?
7) Опишите достоинства и недостатки векторного способа представления изображения.
8) Приведите характеристику сферы применения векторной графики.
9) Какие основные объекты векторной графики используются в OpenOffice.orgDraw?
10) Какие типы соединительных линий определены в OpenOffice.org Draw?
11) Для чего предназначены точки соединения в рисунках OpenOffice.org Draw?

Задание на лабораторную работу

№ Варианта	Алгоритм
1	Сортировка вставками
2	Сортировка слияниями
3	Пузырьковая сортировка
4	Сортировка перемешиванием
5	Сортировка выбором
6	Линейный поиск
7	Поиск с барьером
8	Бинарный поиск
9	Интерполяционный поиск
10	Сортировка Шелла

ЛАБОРАТОРНАЯ РАБОТА №2

ВЕКТОРНАЯ ГРАФИКА. ПОНЯТИЕ ПРИМИТИВОВ ВЕКТОРНОЙ ГРАФИКИ. ЛИНИИ БЕЗЬЕ. ВЕКТОРНЫЕ ОПЕРАЦИИ НАД ПРИМИТИВАМИ В CORELDRAW

1.1. Цель работы

Изучение примитивов, аффинных преобразований и преобразований над замкнутыми объектами векторной графики на примере построения чертежа конструкции с помощью прикладного пакета программ CorelDRAW.

1.2. Теоретические сведения

Векторная графика – это использование геометрических примитивов, таких, как точки, линии, сплайны и многоугольники, для представления изображений в компьютерной графике.

Векторная графика описывает изображение с помощью математических формул. Основное преимущество векторной графики состоит в том, что при изменении масштаба изображения оно не теряет своего качества, что показано на рис. 1.1. Отсюда следует и ещё одно преимущество – при изменении размеров изображения не изменяется размер файла.

а) б) в)

Рис.1.1. Отличие растровой от векторной графики при изменении размеров объектов:

а – сходное векторное изображение; б – иллюстрация, увеличенная в 8 раз как векторное изображение; в – иллюстрация, увеличенная в 8 раз какрастровое изображение.

Основное отличие векторной графики от растровой – в принципе хранения изображения. Векторная графика описывает изображение с помощью математических формул. По своей сути любое изображение можно разложить на множество простых объектов, таких, как контуры, графические примитивы и т.д. Любой такой простой объект состоит из контура (на рис. 1.2 изображен черным цветом) и заливки (серый цвет).

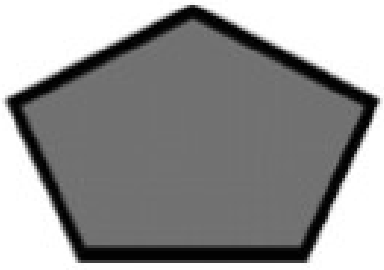


Рис. 1.2. Простой объект векторной графики

Основное преимущество векторной графики состоит в том, что при изменении масштаба изображения последнее не теряет своего качества. Отсюда следует и другой вывод – при изменении размеров изображения не изменяется размер файла. Ведь формулы, описывающие изображение, остаются те же, меняется только коэффициент пропорциональности. С другой стороны, такой способ хранения информации имеет и свои недостатки. Например, если делать очень сложную геометрическую фигуру (особенно если их много), то размер «векторного» файла может быть гораздо больше, чем его «растровый» аналогиз-за сложности формул, описывающих такое изображение.

В основе векторной графики лежат математические представления о свойствах геометрических фигур. Как было сказано выше, простейшим объектом (примитивом) векторной графики является линия. Поэтому в основе векторной графики лежит математическое представление линии.

Типичными примитивными объектами векторной графики являются:

– точки;

– прямые и ломаные линии;

– многоугольники;

– окружности и эллипсы;

– кривые Безье;

– текст.

Этот список неполон. Есть разные типы кривых (Catmull–Rom сплайны,NURBS и т.д), которые используются в различных приложениях. Рассмотримнекоторые из этих примитивов.

Точка

Точка на плоскости задается двумя числами (x, y), определяющими ее положение относительно начала координат.

Прямая линия

Известно, что для задания прямой линии достаточно двух параметров. Обычно график прям й линии описывается уравнением $y = kx + b$. Зная параметры k и b, всегда можно нарисовать бесконечную прямую линию в известной системе координат.

Отрезок прямой, ломаная линия и многоугольник

Для задания отрезка прямой надо знать еще несколько параметров, например координаты *x1* и *x2* начала и конца отрезка, поэтому для описания отрезка прямой линии необходимы четыре параметра. Ломаную линию можно представить в виде отрезков прямых. Многоугольник представляет собой замкнутую ломаную линию.

Окружность и эллипс

Окружности, эллипсы, параболы, гиперболы и другие линии, уравнения которых не содержат степеней выше второй, относятся к кривым второго порядка. Прямые линии – это частный случай кривых второго порядка.

Кривые Безье

Благодаря простоте задания и возможности манипулировать формой кривые Безье нашли широкое применение в компьютерной графике длятого, аффинные преобразования кривой (перенос, масштабирование, вращение) также легко могут быть осуществлены путём применения трансформаций к опорным точкам Наличие выпуклой оболочки значительно облегчает задачу о точках пересечен я кривых Безье: если не пересекаются выпуклые оболочки, то и не пересекаются сами кривые.

Наибольшее значение имеют кубические кривые Безье. Кривые высших порядков при о работке требуют большего объёма вычислений и для практических целей используются реже. Для построения сложных по форме линий отдельные кривые Безье могут быть последовательно соединены друг с другом в сплайн Безье. Для того чтобы обеспечить гладкость линии в месте соединения двух кривых, смежные опорные точки обеих кривых должны лежать на одной линии.

Текст

В компьютерных шрифтах, таких как TrueType, TimesNewRoman и другие, каждая буква создаётся из кривых Безье.

Векторные операции, применяемые к примитивам

Векторные графические редакторы типично позволяют вращать, перемещать, отражать, растягивать, скашивать, выполнять основные аффинные преобразования над объектами, комбинировать и группировать примитивы в более сложные объекты.

Более сложные преобразования включают булевы операции на замкнутых фигурах: объединение, дополнение, пересечение и т. д. Векторная графика идеальна для простых или составных рисунков, которые должны быть аппаратно-независимыми или не нуждаться в фотореализме. Например, PostScriptи PDF используют модель векторной графики.

Операция «Группировка объектов»

Операция «Группировка объектов» состоит в объединении двух или более объектов (контуров) в одну группу. С полученным таким образом, сгруппированным объектом, можно обращаться как с единым объектом. Его можно перемещать, поворачивать, растягивать и выполнять многие другие операции без искажения взаимного расположения и пропорций, входящих в него объектов.

Для реализации данной операции в CorelDRAW предусмотрена команда Arrange-Group (Расположение - Сгруппировать). Предварительно перед ее применением следует выделить все объекты,которые необходимо объединить вгруппу, с помощью инструмента Pick (Указатель). Это

осуществляют путемпротягивания с помощью мыши прямоугольной рамки выделения, захватывающей все группируемые объекты.

Каждый элементарный объект, сгруппированный в результате применениязтой операции, сохраняет свои свойства. Поэтому при необходимости всегда можно выполнить обратную операцию – разгруппировать группу объектов командой Arrange-Ungroup (Расположение - Разгруппировать) либо с помощью «горячих клавиш» Ctrl+U иработать с каждым простым объектом индивидуально.

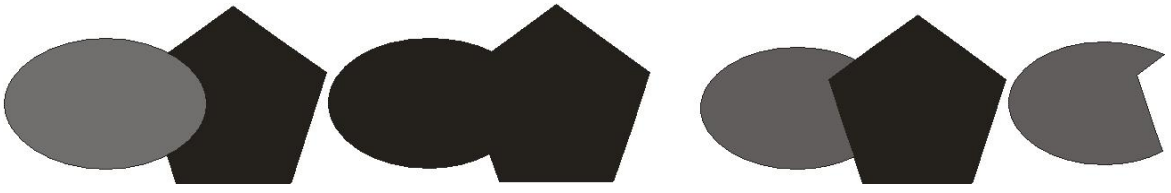
При реализации операции группировки можно использовать несколько уровней группировки. В этом случае разгруппировка объектов происходит в обратном порядке с сохранением иерархии группировки.

Операции «Объединение форм объектов»

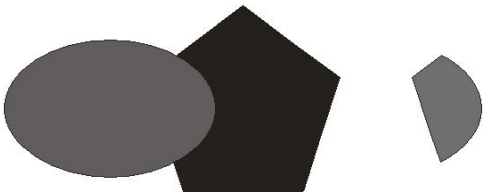
Под Объединением форм объектов (Shaping) подразумеваются операции по созданию новых объектов в результате определенного взаимодействия исходных. В современных векторных редакторах предусмотрены различные варианты объединения объектов. Наиболее распространенными из них являются три процедуры, принцип действия которых основан на использовании базовых логических операций ИЛИ, И, И–НЕ. Рассмотрим особенности их реализации на примере редактора CorelDRAW. В CorelDRAW предусмотрены три типа операций по объединению объектов: слияние (weld) (рис. 1.3, а), исключение (trim) (рис. 1.3, б) и пересечение (intersection) (рис. 1.3, в).

Перечисленные команды расположены в подменю, для отображения которого необходимо выбрать команду Arrange-Shaping (Расположение - Формирование):

– слияние (weld) – это операция по формированию из нескольких объектов одного, область которого будет совпадать с областью расположения исходных объектов. После ее выполнения в результирующий контур входят все области нижнего и верхнего контура (логическая операция ИЛИ). Причем под областью нового контура (объекта) понимается часть плоскости, ограниченная результирующим контуром и расположенная внутри этого контура;



а) б)



в)

Рис. 1.3. Примеры операций:

а– слияние (weld); б – исключение (trim); в – пересечение (intersection)

– исключение(trim) – здесь результирующий контур состоит из части нижнего объекта, не пересекающейся с верхним (операция НЕ). Действие команды Trim во многом подобно работе формы для выкраивания печенья израскатанного теста. Команда удаляет те части объекта, которые перекрываются с другими;

– пересечение (intersection) – в результирующий контур входят толькопересекающиеся (общие) области объектов. Это аналог логической операции И.

Полученный объект (контур) создается путем использования одной или нескольких операций по обработке форм объектов. В результате такой операции из нескольких объектов получается новый объект, обладающий свойствами самого верхнего из исходных объектов, участвующих в операции.Поэтому в отличие от рассмотренной ранее операции группировки здесь свойства составляющих объектов теряются.

Операция «Комбинирование объектов»

Комбинированием называется операция по созданию одного объекта из нескольких, который обладает следующими тремя свойствами:

– из объекта удаляются те области, в которых четное количествоисходных объектов накладывалось друг на друга;

– объект допускает редактирование своей формы с помощью контуровисходных объектов, которые при этом сохраняются;

– цвет объекта будет совпадать с цветом самого нижнего исходногообъекта.

При этом скомбинированные объекты имеют различные контуры, но одинаковые свойства (цвет контура, заливки, толщина контура).

Данная операция выполняется с помощью Команды Arrange - Combine(Расположить - Скомбинировать).

Достоинства векторной графики:

- существует возможность неограниченного масштабирования изображения без потери качества и практически без увеличения размеров исходного файла;
- изображение векторной графике значительно легче редактировать, поскольку готовое изображение без потери качества и практически без увеличения размеров исходного файла;
- векторным программа свойственна высокая точность рисования

Однако у векторной графики есть недостатки:

– практически невозможно осуществить экспорт изображения израстрового формата в векторный;

– векторный принцип описания изображения не позволяетавтоматизировать ввод графической информации, как это делает сканер для растровой графики;

– в векторной графике невозможно применение обширной библиотекиэффектов (фильтров), используемых при работе с растровыми изображениями.

1.3. Порядок выполнения работы

1. Изучить теоретическую часть работы.
2. Нарисуйте силуэт животного с помощью кривых Безье в соответствии с вариантом.
3. Исследовать операции «Группировка объектов», «Обработка форм объектов», «Комбинирование объектов» векторной графики.
4. Результаты работы в виде графических изображений вставить в отчет.
5. Оформить отчет и защитить работу.

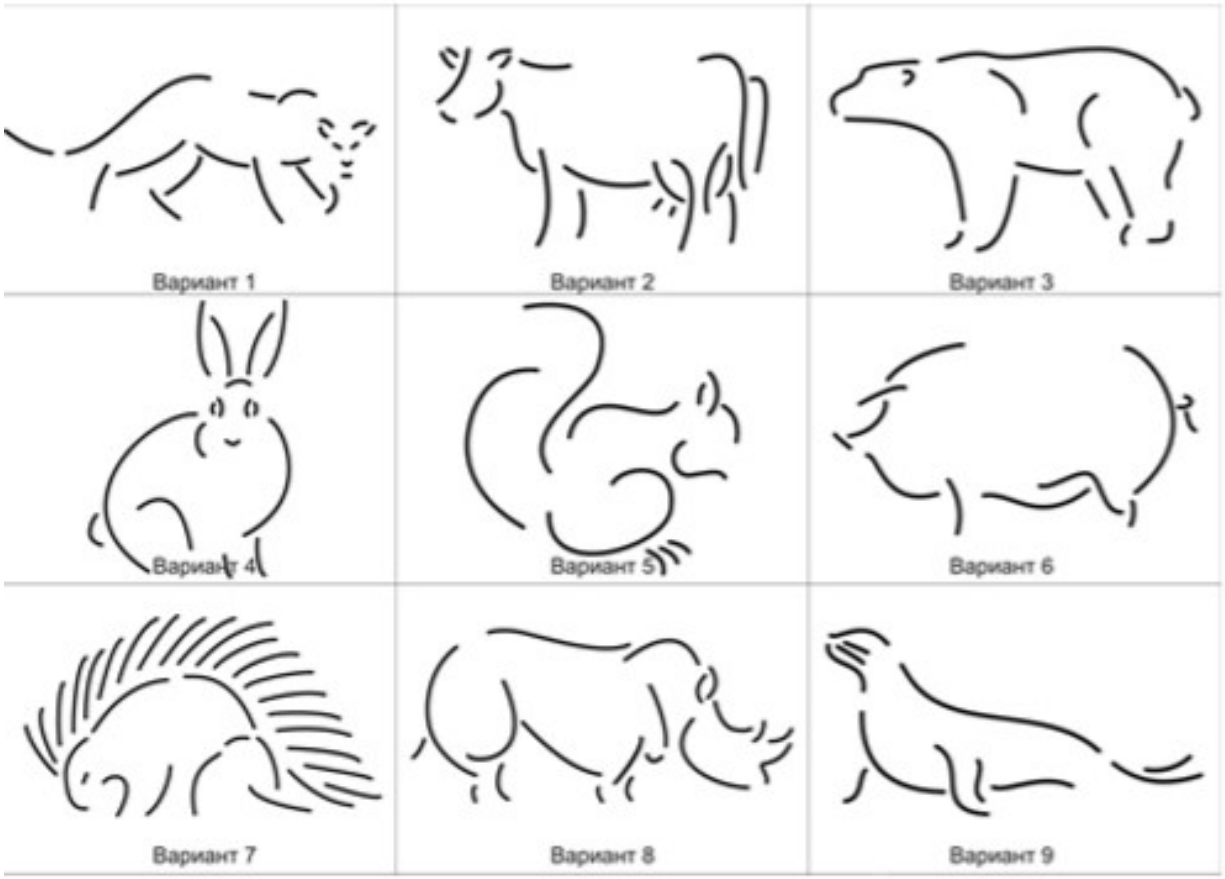
1.4. Сод ржание отчета

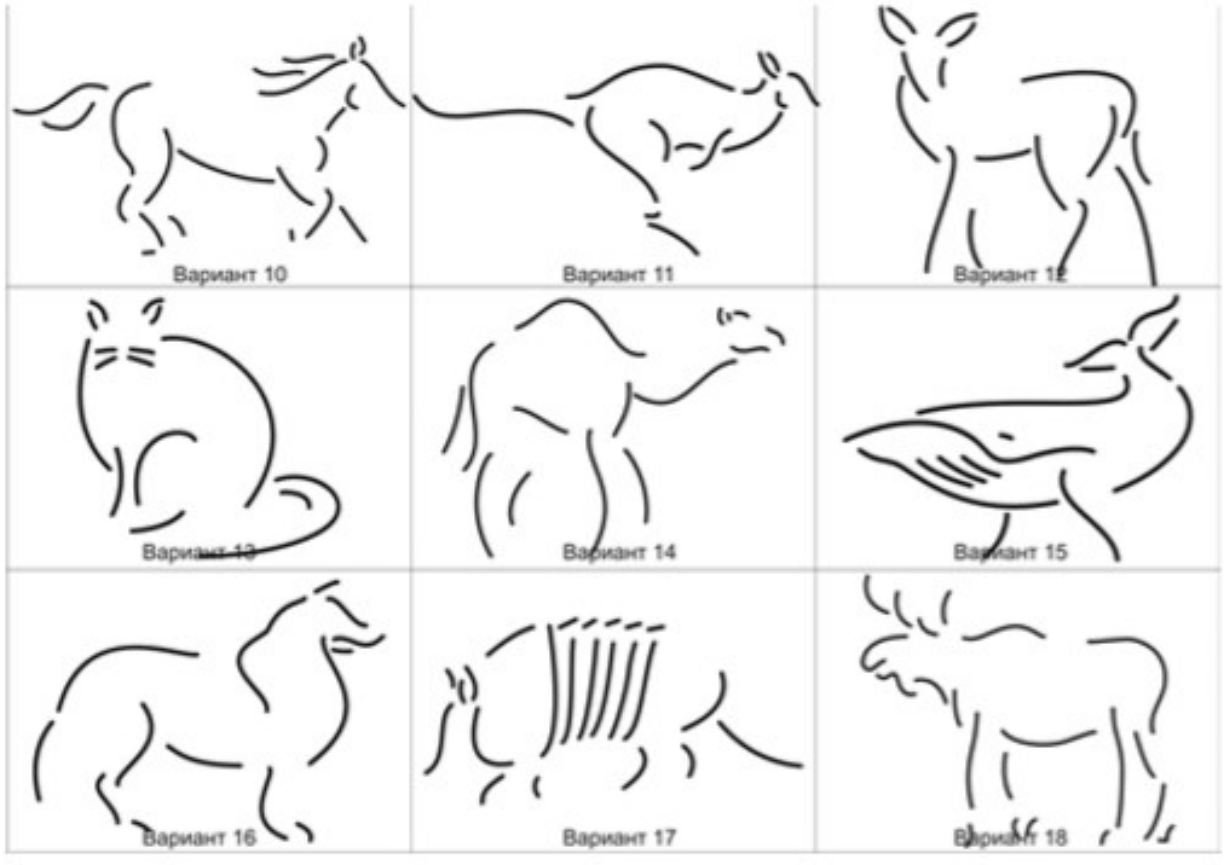
1. Цель работы.
2. Описание порядка выполнения работы с приведением заключительных из соображений, полученных в результате выполнения лабораторной работы.
3. Выводы по работе.

1.5. Контрольные вопросы

1. Что представляет собой векторная графика?
2. Как е преимущества векторной графики перед растровой?
3. Как е примитивы векторной графики вы знаете?
4. Как описываются примитивы в векторной графике?
5. Что такое кривая Безье?
6. Какое применение кривые Безье находят в векторной графике?
7. Опишите алгоритмы построения кривых Безье различных порядков.
8. Какие операции «Обработки форм объектов» вы знаете?
9. Назовите основные различия операций «Группировка объектов»,«Обработка форм объектов» и «Комбинирование объектов».

Задание на лабораторную работу





Лабораторная работа №3

ВЕКТОРНАЯ ГРАФИКА. ИЗУЧЕНИЕ ВЕКТОРНЫХ И РАСТРОВЫХ ЭФФЕКТОВ В ВЕКТОРНОЙ ГРАФИКЕ В CorelDRAW

2.1. Цель работы

Изучение основных векторных и растровых эффектов на объектах векторной графики на примере моделирования изделия РЭА с помощью прикладного пакета программ CorelDRAW.

2.2. Теоретические сведения

В программах векторной графики к объекту можно применить два вида эффектов: векторные и растровые.

Векторные эффекты

Программа CorelDRAW представляет пользователю широкие возможностипо созданию художественных эффектов в объектах векторной графики. Всеготаких эффектов десять:

– переход (blend);

– контур (contour);

– искажение (distortion);

– оболочка (envelope);

– выдавливание (extrude);

– тень (dropshadow);про

– интерактивная прозрачность (interactivetransparancy);

– линза (lens);

– перспектива (perspective);

– фигурная обрезка (powercIip).

Все они, кроме эффекта «Интерактивная прозрачность», формируются с помощью соответствующих команд меню Effects (Эффекты), а первые семь – также и интерактивными рабочими инструментами, расположенными в десятой ячейке блокаинструментов программы.

Следует отметить, что в отличие от растровых эффектов векторные эффекты применяются к объекту векторной графики непосредственно без каких-либо предварительных преобразований объекта. На панели инструментов (рис. 1.1) представлены интерактивные эффекты, т.е. такие эффекты, свойствакоторых можно изменять в реальном времени с помощью специальных ползунков и сеток.

Рис. 1.1. Меню «Интерактивные эффекты» на панели инструментов

Меню «Эффекты» содержит следующие команды:

– adjust (настройка цвета) – содержит набор команд для корректировкиокраски выделенного объекта. Сюда входят такие команды, как улучшение контраста, тоновые кривые, автоматическая настройка цвета и т.д.;

– transform (преобразование). Под трансформацией векторного объектапонимается изменение его формы или ориентации путем выполнения двух групп операций:

– масштабирование (изменение размеров), поворот, наклон и зеркальныеразвороты (по горизонтали и вертикали) объекта;

– различные операции обработки контура объекта;

– correction (корректирование) – данная настройка содержит однукоманду, позволяющую со сканированногоизображения удалять следы пыли, царапины, грубые помарки;

– artisticmedia (художественные средства) – позволяет рисоватьфигурную линию, толщина которой переменна и определяется ее стандартным профилем, выбранным на панели свойств (рис. 1.2). Под этим именем объединены сразу пять инструмен$_т$ов, которые позволяют создавать необычные графические обводки. Выбрать тип инструмента можно при нажатии соответствующего рисунка на панели свойств в ее левой части;

Рис. 1.2. Графическая обводка (инструмент – «Художественные средства»)

– blend (переход). Эффект перехода состоит в следующем. Из двухисходных векторных объектов, расположенных на некотором расстоянии друг от друга, формируется комбинированный объект (объект перехода), состоящий из двух базовых элементов и группы промежуточных. В качестве базовых элементов используются исходные объекты, а промежуточные элементы, формируемые программой автоматически, обладают следующими свойствами:их геометрические и цветовые параметры плавно изменяются при переходе от одного базового элемента к другому, что и показано на рис. 1.3.

Параметры эффекта регулируются с помощью управляющей конструкции инструмента, а также элементов управления панели свойств;

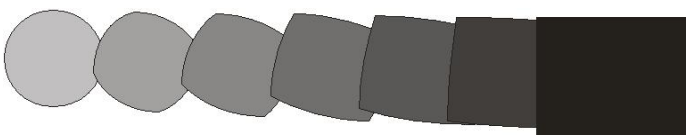


Рис. 1.3. Пошаговое перетекание из круга в прямоугольник (количество шагов перетекания 6)

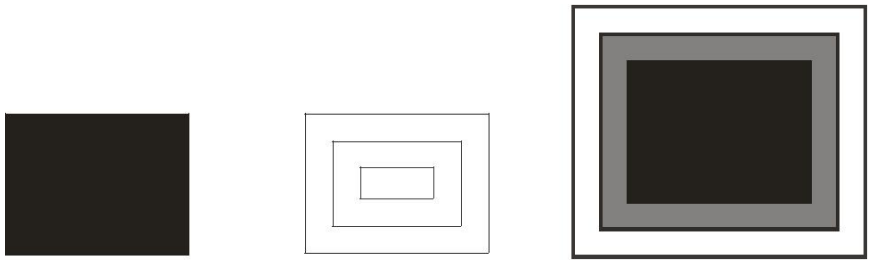
– contour (контур) – эффект, напоминающий пошаговый переход. Онзаключается в том, что вокруг выделенного объекта (рис. 1.4, а) на определенном расстоянии создаются подобные ему концентричные объекты.

Допускается создание трех разновидностей эффекта контура:

– *tocenter* (к центру) – витки равномерно заполняют всю внутреннююобласть объекта;

– *inside* (вовнутрь) – заданное количество витков заполняет частьвнутренней области объекта, примыкающую к его контуру (рис. 1.4, б);

– *outside* (наружу) – заданное количество витков располагается снаружиот объекта, примыкая к его контуру (рис. 1.4, в);



а б в

Рис. 1.4. Векторный эффект «контур»:

а – исходный объект; б – заданное количество витков заполняет частьвнутренней области объекта, примыкающую к его контуру; в –заданное количество витков располагается снаружи от объекта, примыкая к его контуру

Параметры эффекта регулируются с помощью управляющей конструкции инструмента, а также элементов управления панели свойств;

– envelope (оболочка) – с помощью данного эффекта можно задаватьпроизвольное искажение векторных объектов. Суть искажения состоит в том, что объект вписывается во внешнюю оболочку. Создать оболочку можно тремя способами: вручную (рис 1.5 а), используя заготовки (рис. 1.5, б) и преобразовав в оболочку любую из «посторонних фигур». Параметры эффекта регулируются с помощью управляющих маркеров (узелков), расположенных вдоль контура оболочки и элементов управления панели свойств. Альтернативным средством настройки параметров эффекта оболочки являетсядокер Envelope(Оболочка), открываемый одноименной командой подменю Dockers (Докеры) менюWindow (Окно);

а б

Рис. 1.5. Создание оболочки:

а– вручную; б – используя заготовки

– extrude (выдавливание) – имитируется вид в перспективе объемногообъекта, полученного в результате добавления некоторой толщины кисходному векторному объекту, имеющему плоскую форму, иначе –псевдообъём. Эффект экструзии позволяет построить на рисунке проекцию обобщенного цилиндра – тела, образующегося при перемещении плоской фигуры в пространстве в направлении, перпендикулярном ее плоскости.

Основные свойства данного эффекта можно изменить в дополнительном докере или панели свойств;

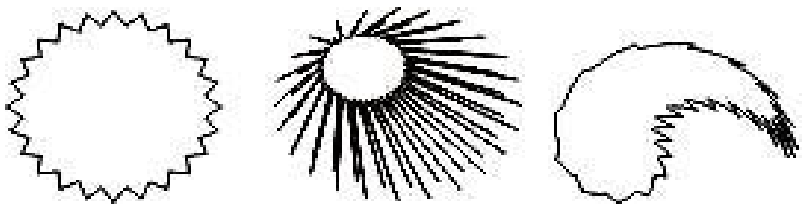
– lens (линзы). Объект с примененным эффектом линзы становитсяполупрозрачным или изменяет свой цвет по определенным правилам, в зависимости от выбранного типалинзы. Каждый из режимов линзы имеет свои настройки, однако некоторые из них повторяются;

– addperspective(добавить перспективу) – средство для автоматическогопостроения эффектов, имитирующих объем и глубину сцены;

– rellover (интерактивная кнопка) – эффект для создания элементавеб-дизайна, который изменяет свой внешний вид в зависимости от состояния указателя мыши;

– copyeffect (копировать эффект) – копирует на выделенный объектэффект, примененный к другому объекту;

– interactivedistortion (интерактивное искажение) – происходитискажение формы исходного векторного объекта (рис. 1.6, а) по закону, определяемому выбранным типом искажения, и с параметрами, заданными пользователем. Предусмотрены три разновидности эффекта искажения (типов искажения):*pushandpull*(выпуклость и вогнутость) (рис. 1.6, б),*zipper*(зигзаг) и twister (скручивание) (рис. 1.6, в). Параметры эффекта регулируются спомощью управляющей конструкции инструмента и элементов управления панели свойств;



а б в

Рис. 2.7. Эффект искажения:

а – исходный объект; б – выпуклость и вогнутость; в – скручивание

– interactiveshadowt(интерактивная тень). В CorelDRAW тенью называется монохром$_{ное}$ пиксельное изображение, автоматически формирующееся в составе с единенного объекта класса «падающая тень». В качестве управляющего в таком соединенном объекте выступает объект (или группа объектов), отбрасывающий тень. Поэтому все изменения, вносимые в управляющий объект, влияют на форму тени. Предусмотрено пять разновидностей эффекта тени (типов тени), которые определяют положение плоскости, на которую объект отбрасывает тень: *flat* (сзади), *bottom* (снизу), *mop* (сверху), *left* (с) и *right* (справа). Параметры эффекта регулируются с помощью управляющей конструкции инструмента и элементов управления панели свойств.

Растровые эффекты

Растровые эффекты применяются к объекту векторной графики только после конвертирования данного объекта в растровое (битовое) изображение. Данное действие реализовано через команду меню «Битовые изображения» - «Конвертировать в битовое изображение». Таким образом, существуетвозможность преобразовывать векторное изображение в растровое с применением различных цветовых форматов.

Существует множество растровых эффектов, которые можно применить к битовому изображению. Все они представлены в меню Bitmap (Битовые изображения) и перечислены ниже:

– 3D-эффекты – искажают плоское пиксельное изображение такимобразом, что оно начинает казаться трехмерным. Наиболее часто в этой категории используются эффекты:

– *emboss* (рельеф), преобразующий изображение в совокупность выступовили углублений при боковом освещении;

– *pagecurl* (отогнутый уголок), имитирующий загиб угла листа бумаги спиксельным изображением;

– artstrokes (художественные мазки) – сведены эффекты, имитирующиетрадиционные художественные приемы рисования – от рисунка итальянским карандашом или углем до акварели и пуантилизма;

– blur (размывание). Путем размывания смягчают резкость пиксельныхизображений, имитируют тени, туман, движение. Наиболее часто используемые эффекты – *gaussianblur*(размывание по Гауссу) и *motionblur*(движение);

– camera (камера) – имеется всего один эффект, имитирующий диффузноерассеивание света в оптике фотокамеры;

– colortransform (цветовые преобразования) – сведены четыре эффекта,которые настолько радикально изменяют цветовую гамму пиксельного изображения, что назвать это цветовой коррекцией не представляется возможным. С их помощью можно свести цветовую гамму к

базовым цветам, заменить цвета яркими, броскими оттн , сделать изображение похожим на фотоснимок при съемке против света;

– contour (контуры) – позволяет акцентировать в пиксельном изображениикромки – линии, по которым соприкасаются контрастно окрашенные области. Есть возможность задавать критерии поиска кромки, выбирать способ ее акцентирования и задавать цвет кромки;

– distort (искажение) – искажают пиксельное изображение без образованияиллюзиитрёхмерной. В эту категорию входят, например, такие эффекты, как *tile*(плитка),уменьшающий изображение и многократно повторяющий его ввиде кафельных плиток с росписью,*swirl*(завиток) и *wetpaint*(мокрая краска);

– noise (шум) – эффекты, позволяющие работать с визуальным шумом – случайным образом распределенными по площади рисунка пикселами. Эффекты позволяют добавлять шум к изображению, управлять его характеристиками и устранять нежелательный шум. Особенно полезными в практической работе являются эффекты:

– *diffuse* (диффузное рассеяние), позволяющий устранятьмикроскопические просветы в пиксельных изображениях;

– *dustandscratches*(пыль и царапины) –позволяющий устранятьпыль и царапины со сканированных фото;

– *removenoise* (устранить шум) и removemoire (устранить муар) – устраняют нежелательные узоры, возникающие на пиксельных изображениях при повторном растрировании с разрешением, отличным от исходного;

– sharpen (резкость) – пять эффектов, изменяющих характеристики ребер – линий, ограничивающих однородные фрагменты пиксельного изображения.

После выбора любой из команд, соответствующих растровому эффекту, на экране раскрывается диалоговое окно, элементы управления которого обеспечивают возможность настройки управляющих параметров и режимов, присущих выбранному эффекту.

2.3. Порядок выполнения работы

1. Изучить теоретическую часть работы.
2. Получить индивидуальное задание
3. Нарисуйте аналогичный пейзаж в соответствии с вариантом.
4. Исследовать векторные эффекты программы CorelDRAW.
5. Исследовать процесс преобразования векторного изображения в растровое и применить для полученного изображения основные растровые эффекты программы CorelDRAW.
6. Результаты работы в виде графических изображений вставить в отчет.
7. Оформить отчет и защитить работу.

2.4. Содержание отчета

1. Цель работы.

2. Описание порядка выполнения работы с приведением заключительных изображен й, полученных в результате выполнения лабораторной работы.

3. Выводы по работе.

2.5. Контрольные вопросы

Какие векторные эффекты в CorelDRAW вы знаете?

1. Чем отличается интерактивный эффект от эффекта, применяемого через «Главное меню»?

2. Какие виды искажения объекта можно получить в CorelDRAW?

4. Что представляют собой растровые эффекты в CorelDRAW и для какого изображения они применяются?

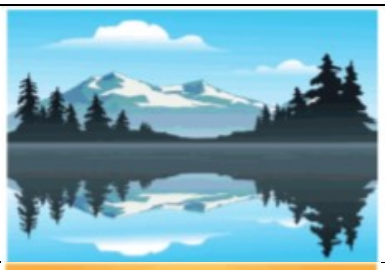
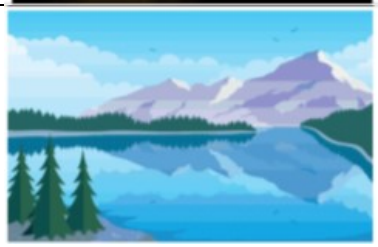
5. Какие растровые эффекты в CorelDRAW вы знаете?

6. Загиб листа бумаги – векторный или растровый эффект?

7. С помощью какого эффекта (векторного или растрового) создается тень объекта?

8. С помощью какого эффекта (векторного или растрового) можно удалить пыль и царапины со сканированного изображения?

Задание на лабораторную работу

№ Варианта	Пейзаж	№ Варианта	Пейзаж
1		7	
2		8	
3		9	
4		10	
5		11	
6		12	

ЛАБОРАТОРНАЯ РАБОТА №4. ВЕКТОРНАЯ ГРАФИКА В C#

Цели лабораторной работы

1. Познакомиться с таким понятием, как векторная графика, основными ее достоинствами и недостатками.

2. Изучить возможности технологии GDI+ для создания статических и динамических векторных изображений.

3. Разработать программу на языке C#.

Теоретический материал

Векторная графика – это способ представления изображений, основанный на использовании элементарных геометрических объектов: точек, линий, сплайнов и многоугольников.

Эту графику часто называют объектно-ориентированной, так как ее основе лежат примитивные объекты, такие как окружности, линии, сферы, кубы и тому подобное, и на их основе создаются более сложные объекты, образуя своего рода иерархическое дерево.

Любая векторная иллюстрация, то есть сама картинка, – это верхний уровень иерархии.

Ниже идут объекты, представляющие собой разнообразные векторные формы.

Каждый из этих объектов, в свою очередь, состоит из открытых или замкнутых контуров. Контуром в данном случае называется кривая, представляющая собой очертания некоторого графического объекта. Если начальная и конечная точки кривой совпадают, то такой контур называется замкнутым. Если же контур имеет четко обозначенные концевые точки, то он называется открытым.

Каждый контур состоит из сегментов, занимающих в контуре определенное положение, фиксируемое опорными точками или узлами – началом и концом сегмента. Замкнутые контуры могут иметь свойство заливки – цвета или узора, выводимого в замкнутой области, ограниченной кривой.

Самый нижний уровень иерархии образуют основные элементы векторных изображений: узлы и отрезки линий, их соединяющие.

Таким образом, упрощенную иерархию векторного изображения можно представить следующим образом (рис. 1).

Рис. 1. Иерархия векторного изображения

Информация о любом объекте в векторной графике хранится в описательной форме. Например, окружность будет храниться следующим образом:

- число, соответствующее координате x центра окружности;
- число, соответствующее координате y центра окружности;
- величина радиуса окружности;
- цвет и толщина контура;
- цвет и характер заливки (при ее наличии).

Векторная графика широко используется в рекламных и дизайнерских агентствах, редакциях и издательствах, так как выполняемые ими оформительские работы основаны на применении шрифтов и простейших геометрических элементов. Применение именно векторной графики позволяет более эффективно решать задачи подобного рода.

GDI (GraphicsDeviceInterface, GraphicalDeviceInterface) является интерфейсом Windows, предназначенным для представления графических объектов и вывод их на монитор или принтер. Он отвечает за отрисовку линий, кривых, отображение шрифтов и обработку палитры и обеспечивает унификацию работы с различными устройствами вместо прямого доступа к оборудованию, что позволяет одними и теми же функциями рисовать на различных устройствах, получая при этом практически одинаковые изображения.

Данная технология предоставляет богатые возможности для работы с векторной графикой. Основной ее элемент (линия) имеет такие атрибуты, как толщина, рисунок пунктира, цвет, которые собраны в одном объекте. Кроме того, существуют различные способы заливки областей,

также собранные в одном объекте, и поддержка матрицы поворотов и растяжений изображений векторной графики.

На базе технологии GDI была разработана GDI+. Это улучшенная среда для 2D-графики, расширенная возможностями сглаживания линий, использования координат с плавающей точкой, градиентной заливки, использованием ARGB-цветов и т. п. Рассмотрим основные возможности для работы с этой технологией, предоставляемые C#.

В рамках данной лабораторной работы мы освоим технику рисования картинок в оперативной памяти с последующим выводом их на форму. Для рисования мы будем использовать объект класса Bitmap, а для вывода на форму – элемент управления PictureBox.

Рассмотрим написание простейшего приложения, когда по нажатию на кнопку на форме будет отображаться красный кружок. Для этого создаем новый проект типа WindowsFormsApplication и размещаем на форме PictureBox и Button. Переходим к методу обработки нажатия на кнопку. В нем нам нужно, во-первых, создать объект класса Bitmap, с размерами, совпадающими с нашим PictureBox. В данном случае Bitmap выполняет роль холста, а PictureBox – рамки, в которую этот холст повесят. Поэтому размеры должны совпадать. В итоге получаем код:

```
Bitmapbmp = newBitmap (pictureBox1.Width, pictureBox1.Height);
```

Теперь мы должны создать объект класса Graphics – основного класса, предоставляющего доступ к возможностям GDI+. Для данного класса не определено ни одного конструктора. Его объект создается в ходе выполнения ряда методов применительно к конкретным объектам, у которых есть поверхность для рисования. Одним из таких объектов и является Bitmap. Поэтому создаем объект класса Graphics следующим образом:

```
Graphics gr = Graphics.FromImage(bmp);
```

Теперь все вызовы методов отображения фигур будут отрабатывать на нашей битовой карте. Класс Graphics содержит множество методов рисования вида Fill* или Draw*, отвечающих за отображение закрашенных или не закрашенных фигур. Первая группа методов в качестве одного из параметров принимает объект типа Brush (кисть), а вторая – объект типа Pen (карандаш). Исключение – метод DrawString, который отображает текст. Этот метод в качестве одного из параметров принимает объект Brush. Для того чтобы отобразить красный круг, как мы задумали ранее, необходимо написать следующий код:

```
gr.FillEllipse(Brushes.Red,10,10,40,40);
```

Обратите внимание, что координаты 10,10 – это не координаты центра эллипса , а координаты левого верхнего угла прямоугольника, в который будет вписан эллипс. Brushes.Red – стандартная красная кисточка. 40,40 – высота и ширина эллипса. Если вы сейчас запустите приложение и нажмете на кнопку, то ничего не произойдет. Вернее, эллипс будет нарисован, но

мы его не увидим, так как сейчас он находится в оперативной памяти. Для того чтобы отобразить нарисованную картинку на форме, необходимо добавить следующую строчку кода:

pictureBox1.Image = bmp;

В итоге на экране вы должны увидеть примерно следующее(рис. 2):

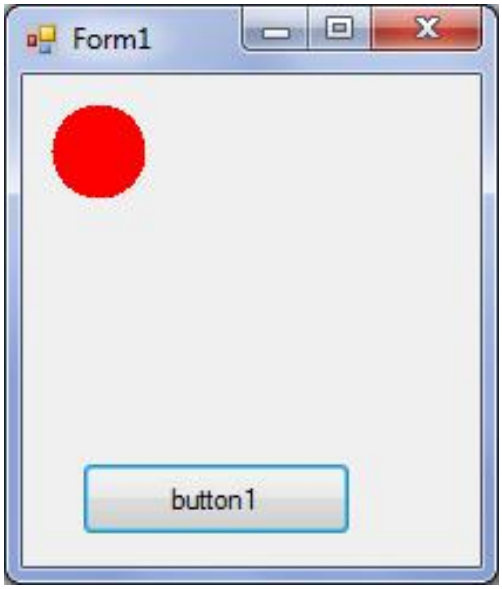


Рис. 2. Красный круг

Рисование в технологии GDI+ осуществляется слоями, которые перекрывают друг друга, поэтому если мы напишем вот такой код:

gr.FillEllipse(Brushes.Red,10,10,40,40); gr.FillEllipse(Brushes.Green,10,10,40,40); ,то на форме увидим только зеленый круг.

Теперь рассмотрим способы создания различных кисточек. Во-первых, создадим кисточку случайного цвета. Для этого напишем следующий код:

Random r=new Random();

SolidBrushsb=new SolidBrush(Color.FromArgb(r.Next(0,255), r.Next(0,255),

r.Next(0,255), r.Next(0,255)));

Метод FromArgb принимает в качестве параметров 4 числа, кодирующих цвет в системе ARGB: A (альфа) – прозрачность; R (red) – красный цвет; G (green) – зеленый цвет; B (blue) – синий цвет.Для создания кисточек с более интересными характеристиками необходимо подключить namespace System.Drawing.Drawing2D. Рассмотрим создание градиентной кисти. Напишемследующийкод:

LinearGradientBrushgradientBrush = new LinearGradientBrush(new Point(0, 0),new Point(40, 40), Color.Green, Color.Blue);

Как мы знаем, линейная градиентная заливка – это вид заливки,которой необходимо задать цвет и координаты двух ключевых точек, расположенных на одной прямой, а цвет точек между

ними рассчитывается относительно них по определенным математическим алгоритмам. В итоге получают плавные переходы из одного цвета в другой.

Следующая кисть, которую мы рассмотрим, – штриховая. Создается она следующим образом:

```
HatchBrushhatchBrush = new HatchBrush(HatchStyle.Cross,
Color.Red, Color.Yellow);
```

Первый параметр конструктора задает вид штриховки. Это системное перечисление с множеством вариантов. Второй параметр отвечает за цвет штрихов, а третий – цвет фона.

При использовании этой кисти в рисовании нашего круга, мы получим следующий результат (рис. 3):

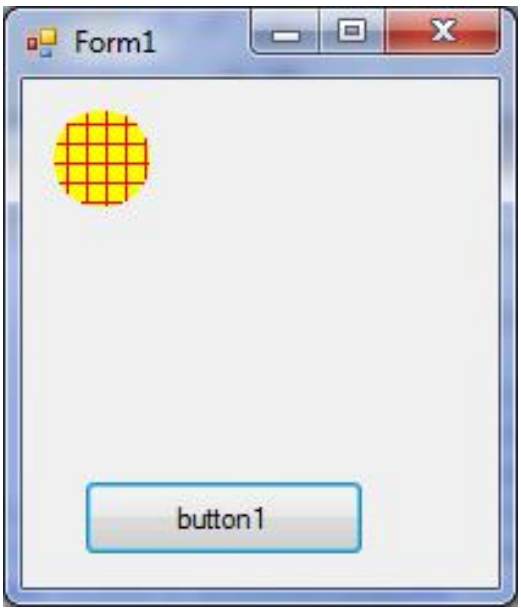


Рис. 3. Штрихованный круг

Теперь рассмотрим создание карандашей. Их можно создавать на основе цвета или на основе ранее созданной кисти. Можно указывать толщину линии, а также форму ее конечной точки (только для незамкнутых контуров) и прочее. Ниже приведен код для создания различных карандашей:

```
Pen p1=newPen(Color.Red); // Обычное красное перо
Pen p2=newPen(Color.Green, 4); // Ширина 4 пикселя
p2.EndCap = LineCap.ArrowAnchor; //Стрелочка на конце
Pen p3=new Pen(gradientBrush, 6); // Градиент
Pen p4=new Pen(hatchBrush, 6); // Штриховка
```

Для вывода текста необходимо использовать метод DrawString, принимающий в качестве параметров строку, которую нужно вывести, объект класса Font, кисть и координаты вывода. Конструктор класса Font, в свою очередь, принимает название и размер шрифта. Приведемкодвыводанадписи «Hello, world!»:

gr.DrawString(“Hello, world!”, new Fonr(“Arial”,15),hatchBrush,20,20);

Мы рассмотрели методы вывода различных векторных примитивов. Теперь рассмотрим методы их преобразования:

- перенос;
- поворот;
- масштабирование;
- сдвиг.

Данные методы широко применяются в ситуациях, когда вам необходимо изменить размер, положение или форму фрагмента рисунка, состоящего из большого числа объектов. Перечисленные выше методы применяют преобразования сразу ко всем отображаемым объектам, что экономит вам время.

Перенос – это линейное перемещение объекта, при котором размер и ориентация не меняются. Напишем код отрисовки красного обведенного контуром прямоугольника, затем вызовем метод переноса и повторим код:

```
gr.FillRectangle(Brushes.Red, 10, 10, 20, 40);
gr.DrawRectangle(p2, 10, 10, 20, 40);
gr.TranslateTransform(20, 20);
gr.FillRectangle(Brushes.Red, 10, 10, 20, 40);
gr.DrawRectangle(p2, 10, 10, 20, 40);
```

В итоге на экране вы должны увидеть примерно следующее(рис. 4):

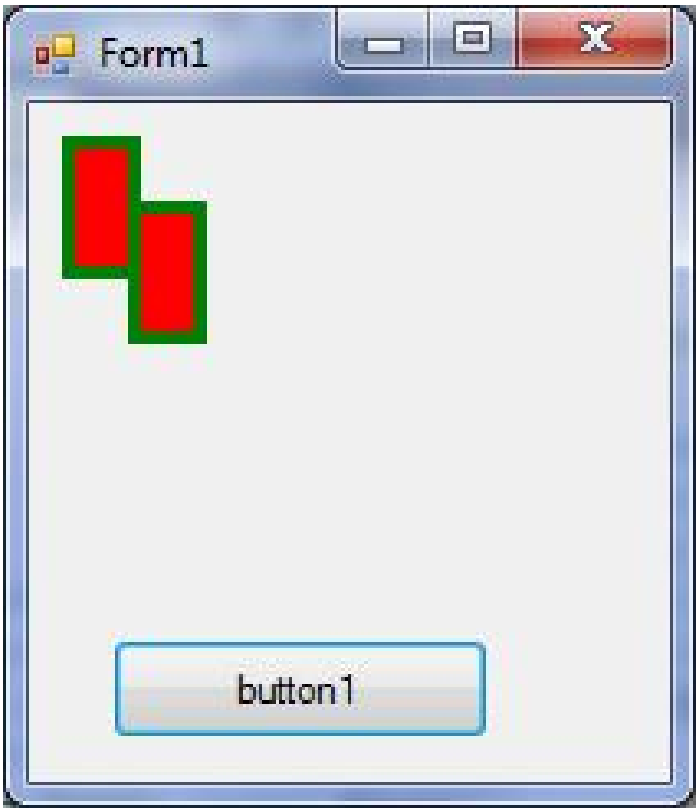


Рис. 4. Трансформация «Перенос»

Метод TranslateTransform принимает в качестве параметров величины переноса по обеим осям. По сути он сдвигает начало координат для отрисовки объекта.

Поворот – это изменение угла отрисовки объекта относительно начала координат. Величина поворота задается в радианах. Код данной трансформации следующий:

```
gr.FillRectangle(Brushes.Red,50, 50, 20, 40);
gr.DrawRectangle(p2, 50, 50, 20, 40);
gr.RotateTransform(20);
gr.FillRectangle(Brushes.Red, 50, 50, 20, 40);
gr.DrawRectangle(p2, 50, 50, 20, 40);
```

Обратите внимание, что поворот выполняется не относительно оси объекта, а относительно начала координат. В итоге у вас на форме должно появиться следующее изображение (рис. 5):

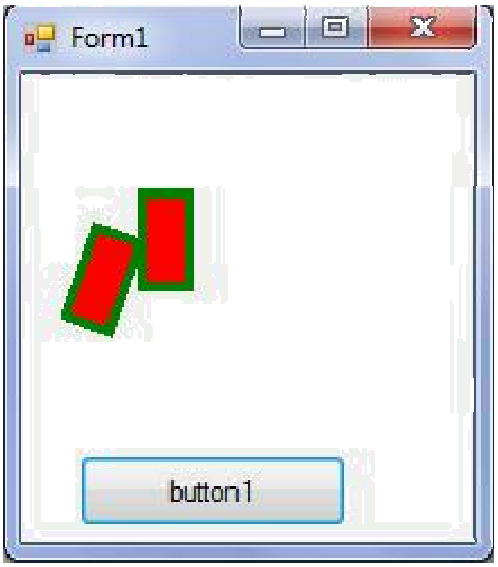


Рис. 5. Трансформация «Поворот»

Масштабирование – это изменение размеров объекта по обеим осям. Увеличим наш прямоугольник в два раза в высоту и ширину:

```
gr.FillRectangle(Brushes.Red,10, 10, 20, 40);
gr.DrawRectangle(p2, 10, 10, 20, 40);
gr.ScaleTransform(2,2);
gr.FillRectangle(Brushes.Red, 10, 10, 20, 40);
gr.DrawRectangle(p2, 10, 10, 20, 40);
```

В итоге у вас на форме должно появиться следующее (рис. 6):

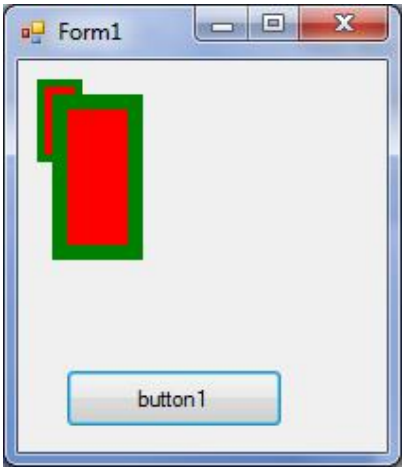


Рис. 6. Трансформация «Масштабирование»

Сдвиг – практически произвольное искажение объекта. Для него не существует какого -то отдельного метода. В данном случае необходимо напрямую задавать матрицу трансформации:

```
gr.FillRectangle(Brushes.Red,10, 10, 20, 40);
gr.DrawRectangle(p2, 10, 10, 20, 40);
Matrix matrix = new Matrix();
matrix.Shear(0.5f, 0.25f);
gr.Transform = matrix;
gr.FillRectangle(Brushes.Red, 10, 10, 20, 40);
gr.DrawRectangle(p2, 10, 10, 20, 40);
```

В итоге на экране должна получиться следующая картинка (рис. 7):

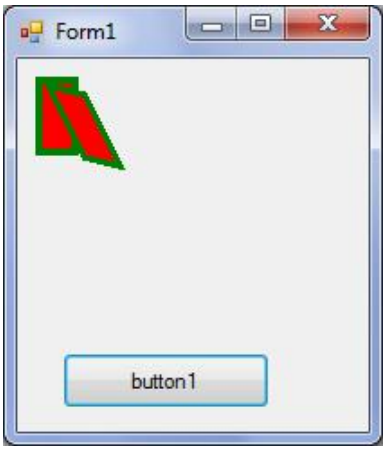


Рис. 7. Трансформация «Сдвиг»

Нужно заметить, что все рассмотренные трансформации основываются на преобразовании матрицы координат. И при вызове нескольких методов подряд их эффекты будут складываться. Для того чтобы вернуть матрицу координат в исходное состояние, необходимо вызвать метод ResetTransform.

Задание на лабораторную работу

№ варианта	Объект для первого режима	Направление движения для второго режима	Первый фоновый объект для второго режима	Второй фоновый объект для второго режима
1.	Рыбка	Слева направо	Куст водорослей	Груда камней
2.	Паровоз	Справа налево	Дом	Дерево
3.	Грузовик	По диагонали слева направо	Дерево	Знак «Главная дорога»
4.	Пароход	Слева направо	Айсберг	Остров с деревом
5.	Легковой автомобиль	Слева направо	Лиственное дерево	Ель
6.	Парашютист	Сверху вниз	Солнце	Облако
7.	Ракета	Снизу вверх	Воздушный шар	Облако
8.	Парусник	Слева направо	Облако	Айсберг
9.	Самолет	По диагонали снизу вверх	Дом	Облако
10.	Летающая тарелка	Сверху вниз	Луна	Дом