МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования
«Юго-Западный государственный университет»
(ЮЗГУ)

Кафедра механики, мехатроники и робототехники

УТВЕРЖДАЮ
проректор по учебной работе
_____ О.Г. Локтионова
«___» _____ 2015 г.

ИНФОРМАЦИОННЫЕ УСТРОЙСТВА И СИСТЕМЫ В МЕХАТРОНИКЕ И РОБОТОТЕХНИКЕ

Методические указания к выполнению практических работ по дисциплине «Информационные устройства и системы в мехатронике и робототехнике» для студентов направлений 15.03.06

Курск 2015

УДК 621

Составители: С.И. Савин

Рецензент
Кандидат технических наук, доцент *Е.Н. Политов*

Информационные устройства и системы в мехатронике и робототехнике: методические указания к выполнению практических работ по дисциплине «Информационные устройства и системы в мехатронике и робототехнике» / Юго-Зап. гос. ун-т; сост. С.И. Савин. Курск, 2015. 21 с.: ил. 15, табл. 2. Библиогр.: с.21.

Методические указания содержат сведения методе применения разложения в тригонометрический ряд Фурье для очистки зашумленного сигнала, методах пороговой обработки изображений.

Методические указания соответствуют требованиям программы, утверждённой учебно-методическим объединением (УМО).

Предназначены для студентов направлений подготовки 15.03.06 – Мехатроника и робототехника всех форм обучения.

Текст печатается в авторской редакции

Подписано в печать . Формат 60x84 1/16
Усл.печ.л. ___. Уч.-изд.л. ___ Тираж 100 экз. Заказ.
Бесплатно.
Юго-Западный государственный университет.
305040 Курск, ул. 50 лет Октября, 94

ПРИМЕНЕНИЕ РАЗЛОЖЕНИЯ В ТРИГОНОМЕТРИЧЕСКИЙ РЯД ФУРЬЕ ДЛЯ ОЧИСТКИ СИГНАЛА ОТ ШУМА

Цель работы: *изучение методов реализации разложения в тригонометрический ряд Фурье периодических и не периодических сигналов средствами математического пакета MathCAD, а также использования полученного разложения для очистки сигнала от шума.*

Объект исследования: *тригонометрические ряды Фурье.*

Аппаратные средства: *математический пакет MathCAD.*

1.1. Краткие теоретические сведения

Периодическая функция может быть представлена своим разложением в тригонометрический ряд Фурье:

$$f(t)=\frac{a_0}{2}+\sum_{n=1}^{\infty}\left[a_n\cos(n\cdot t)+b_n\sin(n\cdot t)\right], \qquad (1.1)$$

где a_0, a_n, b_n – коэффициенты ряда Фурье. Для их вычисления можно воспользоваться следующими формулами:

$$a_0=\frac{1}{\pi}\int_{-\pi}^{\pi}f(t)dt,\ a_n=\frac{1}{\pi}\int_{-\pi}^{\pi}f(t)\cos(n\cdot t)dt,\ b_n=\frac{1}{\pi}\int_{-\pi}^{\pi}f(t)\sin(n\cdot t)dt \qquad (1.2)$$

Более общая форма записи данного разложения имеет следующий вид:

$$f(t)=\frac{a_0}{2}+\sum_{n=1}^{\infty}\left[a_n\cos\left(\frac{n\cdot\pi\cdot t}{L}\right)+b_n\sin\left(\frac{n\cdot\pi\cdot t}{L}\right)\right], \qquad (1.3)$$

Для вычисления коэффициентов ряда следует воспользоваться следующими формулами:

$$a_0=\frac{1}{L}\int_{-L}^{L}f(t)dt,\ a_n=\frac{1}{L}\int_{-L}^{L}f(t)\cos\left(\frac{n\cdot\pi\cdot t}{L}\right)dt, \qquad (1.4)$$

$$b_n=\frac{1}{L}\int_{-L}^{L}f(t)\sin\left(\frac{n\cdot\pi\cdot t}{L}\right)dt \qquad (1.5)$$

2. Методика выполнения лабораторной работы

Зададимся некоторой полигармонической функцией $f(t)$:

$$f(t)=\sum_{i=1}^{n}\left[A_i\sin(\omega_i t)+B_i\cos(\omega_i t)\right] \qquad (1.6)$$

В Mathcad это реализуется, например, следующим кодом:

$$f(t) := 15\sin(2\cdot t) + 3\sin(40\cdot t) + 1\sin(20\cdot t) + 2\cos(45\cdot t) + 4\cos(30\cdot t) + 5\cos(55\cdot t)$$

Покажем график функции $f(t)$ (см. рисунок 1).

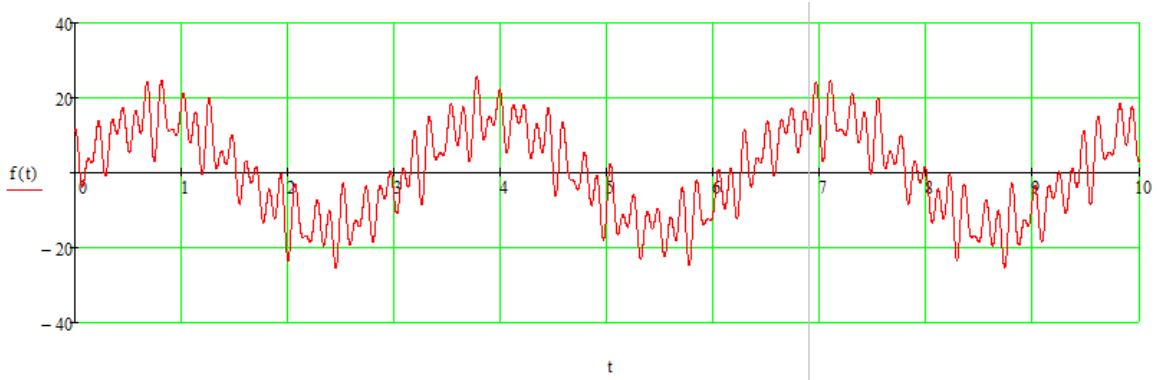

Рисунок 1.1 Зависимость $f(t)$

Для реализации разложения в ряд Фурье воспользуемся аналитической формой выражений для коэффициентов ряда:

$$a0 := \frac{1}{\pi}\cdot\int_{-\pi}^{\pi} f(t)\,dt$$

$$GetA(n) := \frac{1}{\pi}\cdot\int_{-\pi}^{\pi} f(t)\cdot\cos(n\cdot t)\,dt$$

$$GetB(n) := \frac{1}{\pi}\cdot\int_{-\pi}^{\pi} f(t)\cdot\sin(n\cdot t)\,dt$$

Если требуется получить большое количество членов тригонометрического ряда удобно использовать вызов записанных выше функций в цикле:

```
a := | for i ∈ 1..Number
     |   Out_i ← GetA(i)
     | Out

b := | for i ∈ 1..Number
     |   Out_i ← GetB(i)
     | Out
```

Аппроксимация тригонометрическим рядом $F(t)$ исходной функции имеет следующий вид:

$$F(t) := \frac{a0}{2} + \sum_{n=1}^{Number}\left(a_n\cdot\cos(n\cdot t) + b_n\cdot\sin(n\cdot t)\right)$$

Покажем на одном рисунке исходную функцию и её аппроксимацию:

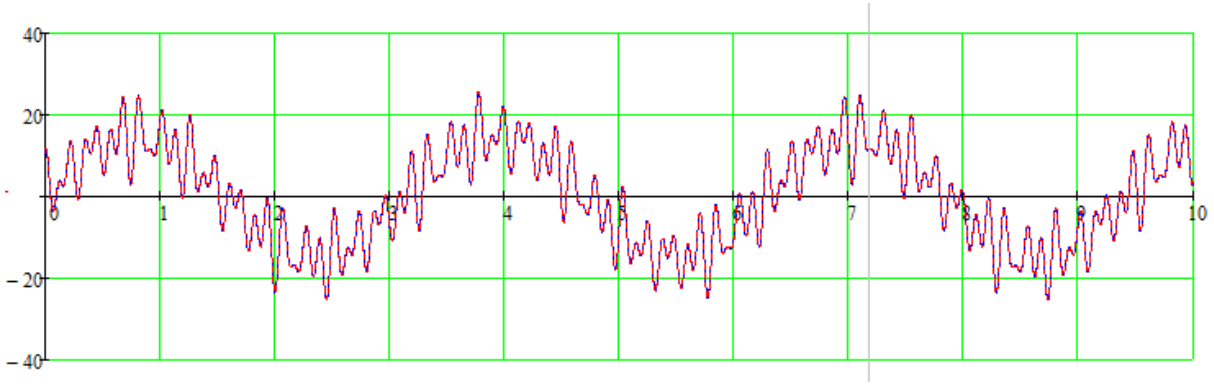

Рисунок 1.2 Зависимости $F(t)$ и $f(t)$

Можем видеть, что зависимости совпали. Набор коэффициентов тригонометрического ряда можно рассматривать как спектр функции. Обработаем этот спектр пороговой обработкой, реализованной следующим кодом:

```
threshold := 10

A := | for i ∈ 1..Number
     |   Out_i ← if(i < threshold, a_i, 0)
     | Out

B := | for i ∈ 1..Number
     |   Out_i ← if(i < threshold, b_i, 0)
     | Out
```

Построим функцию, используя полученные в результате обработки значения спектра:

$$Ffiltered(t) := \frac{a0}{2} + \sum_{n=1}^{Number} \left(A_n \cdot \cos(n \cdot t) + B_n \cdot \sin(n \cdot t)\right)$$

Исходная и полученная зависимость показана на рисунке 3

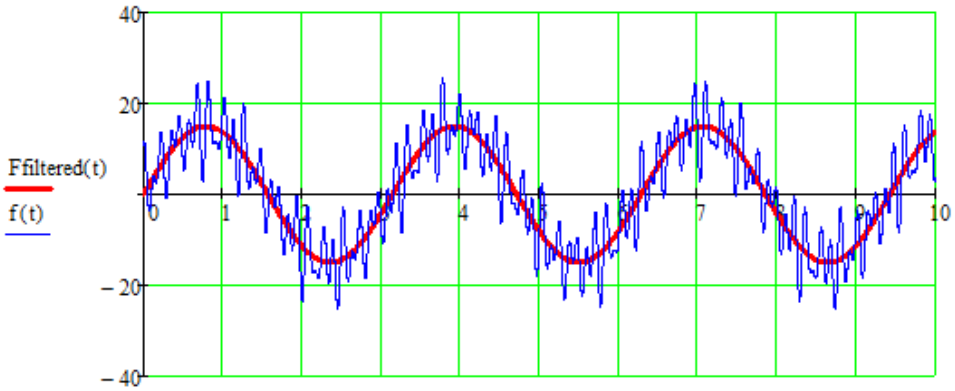

Рисунок 1.3 Зависимости $F_{filtered}(t)$ и $f(t)$

Рассмотрим случай, когда функция $f(t)$ непериодическая (см. рисунок 4):

$$f(t)=\sum_{i=1}^{n}\left[A_i \sin(\omega_i t)+B_i \cos(\omega_i t)\right]+\sum_{i=1}^{m}\left(a_i t^i\right) \qquad (1.7)$$

Функцию указанного вида можно получить, например, используя следующий код:

```
f(t) := 15 sin(2·t) + 3 sin(40·t) + 1 sin(20·t) + 2 cos(45·t) + 4 cos(30·t) + 5 cos(55·t) + 10t
```

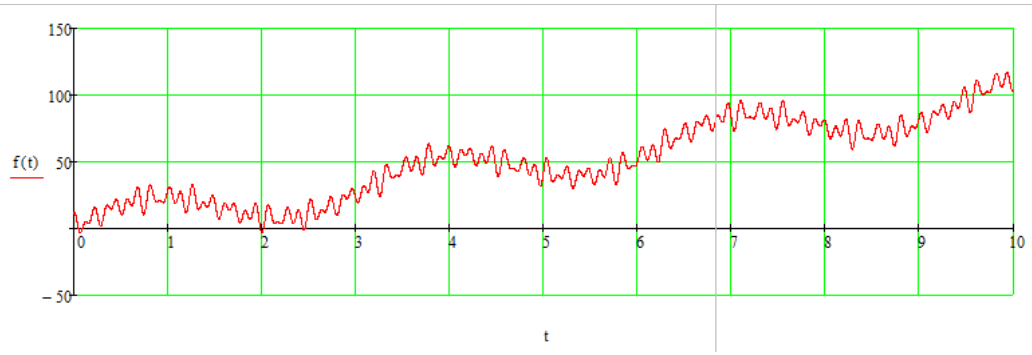

Рисунок 1.4 Непериодическая зависимость $f(t)$

Используя написанный ранее код произведем аппроксимацию заданной непериодической функции тригонометрическим рядом. Результат показан на рисунке 5.

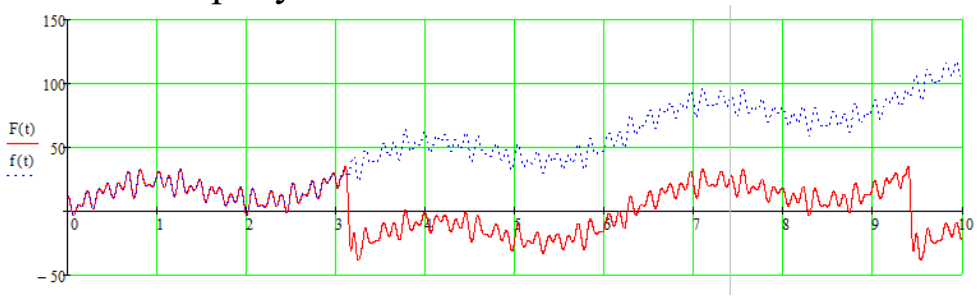

Рисунок 1.5 Зависимости $F(t)$ и $f(t)$

Обратим внимание, что аппроксимация достаточно точна на некотором интервале, и затем зависимость, заданная тригонометрическим рядом повторяется.

После очистки спектра и восстановления сигнала получим следующую зависимость (см. рисунок 1.6).

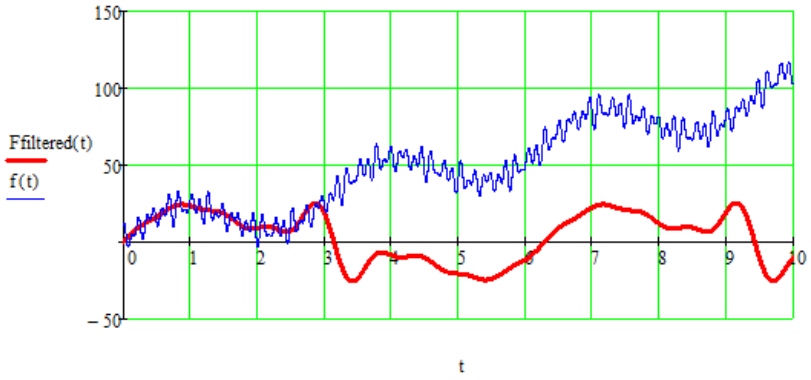

Рисунок 1.6 Зависимости $F_{filtered}(t)$ и $f(t)$

Линейная составляющая функции $f(t)$ привносит в спектр шум на низких частотах, что приводит к искажению сигнала, как это показано на рисунке 1.6.

Используем следующий код, позволяющий задавать размер период аппроксимации:

$$L := 2\pi$$

$$a0 := \frac{1}{L}\cdot\int_{-L}^{L} f(t)\,dt$$

$$GetA(n) := \frac{1}{L}\cdot\int_{-L}^{L} f(t)\cdot\cos\left(\frac{n\cdot\pi\cdot t}{L}\right)dt$$

$$GetB(n) := \frac{1}{L}\cdot\int_{-L}^{L} f(t)\cdot\sin\left(\frac{n\cdot\pi\cdot t}{L}\right)dt$$

$$Number := 300$$

$$a := \left|\begin{array}{l}\text{for } i \in 1..Number\\ \quad Out_i \leftarrow GetA(i)\\ Out\end{array}\right. \qquad b := \left|\begin{array}{l}\text{for } i \in 1..Number\\ \quad Out_i \leftarrow GetB(i)\\ Out\end{array}\right.$$

$$F(t) := \frac{a0}{2} + \sum_{n=1}^{Number}\left(a_n\cdot\cos\left(\frac{n\cdot\pi\cdot t}{L}\right) + b_n\cdot\sin\left(\frac{n\cdot\pi\cdot t}{L}\right)\right)$$

Период аппроксимации определяет отрезок времени, на котором исходная функция будет аппроксимирована точно. Он определяется переменной L.

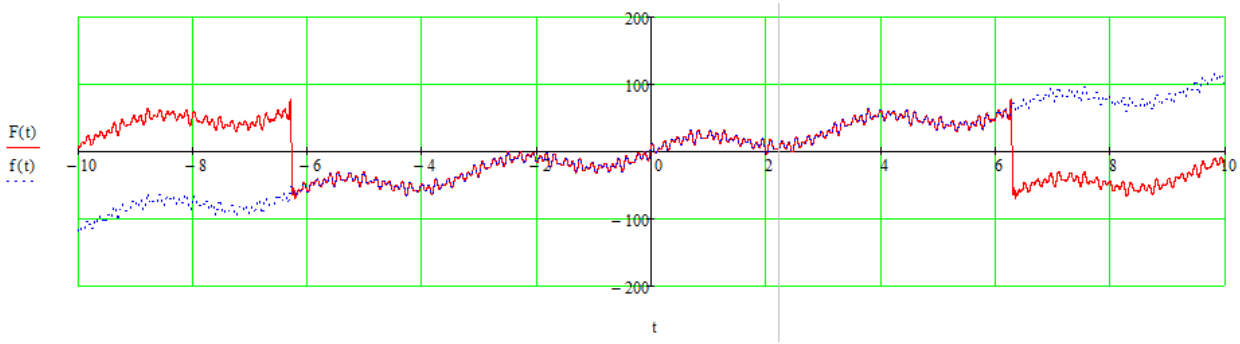

Рисунок 1.7 Зависимости $F(t)$ и $f(t)$

После очистки сигнал выглядит следующим образом:

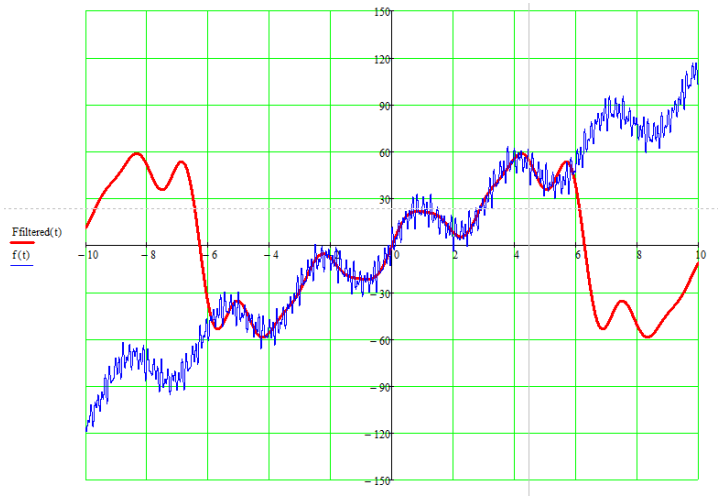

Рисунок 1.8 Зависимости $F_{filtered}(t)$ и $f(t)$

3. Задание на выполнение лабораторной работы

Задание на выполнение лабораторной работы состоит в разложении в ряд Фурье и последующей очистке полигармонических и непериодических сигналов, используя все методы, описанные в данной работе.

Исходные функции следует выбирать из таблицы 1.1 в соответствии с номером студента в списке группы.

Таблица 1.1 Задания на выполнение лабораторной работы

№	Задание
1	$f(t)=10\sin(2\cdot t)+2\sin(12\cdot t)+1.5\sin(10\cdot t)$ $f(t)=0.35\sin(10\cdot t)+0.15\sin(8\cdot t)+0.35\sin(6\cdot t)+3t$

2	$f(t)=0.2\sin(17\cdot t)+0.1\cos(16\cdot t)+15\cos(3\cdot t)$ $f(t)=10.45\sin(2\cdot t)+0.65\sin(10.5\cdot t)+0.05\sin(18\cdot t)+0.5\sin(19\cdot t)+4t$
3	$f(t)=0.35\sin(10\cdot t)+0.15\sin(8\cdot t)+0.35\sin(6\cdot t)$ $f(t)=12.85\sin(1.7\cdot t)+0.9\sin(41\cdot t)+0.1\sin(10\cdot t)+0.2\sin(12\cdot t)+10t^2$
4	$f(t)=10.15\sin(2\cdot t)+0.45\sin(8\cdot t)+0.85\sin(12\cdot t)+1.05\sin(10.5\cdot t)$ $f(t)=0.35\sin(10\cdot t)+0.15\sin(8\cdot t)+0.35\sin(6\cdot t)+10t+7t^2$
5	$f(t)=0.25\sin(13\cdot t)+0.35\sin(12\cdot t)+0.75\sin(5\cdot t)+11.05\sin(1.2\cdot t)$ $f(t)=10.15\sin(2\cdot t)+0.45\sin(8\cdot t)+0.85\sin(12\cdot t)+1.05\sin(10.5\cdot t)+5t^3$
6	$f(t)=10.45\sin(2\cdot t)+0.65\sin(10.5\cdot t)+0.05\sin(18\cdot t)+0.5\sin(19\cdot t)$ $f(t)=0.1\cos(14\cdot t)+0.7\sin(6\cdot t)+25.6\sin(t)+0.9\sin(8\cdot t)+10t+2t^2$
7	$f(t)=12.85\sin(1.7\cdot t)+0.9\sin(41\cdot t)+0.1\sin(10\cdot t)+0.2\sin(12\cdot t)$ $f(t)=10\sin(2\cdot t)+2\sin(12\cdot t)+1.5\sin(10\cdot t)+5t^3+2t^2+3t$
8	$f(t)=0.6\cos(14\cdot t)+10.95\sin(1.1\cdot t)+0.5\sin(9t)+0.7\sin(30\cdot t)+0.8\sin(12\cdot t)$ $f(t)=0.35\sin(10\cdot t)+0.15\sin(8\cdot t)+0.35\sin(6\cdot t)+30+t$
9	$f(t)=0.1\cos(14\cdot t)+0.7\sin(6\cdot t)+25.6\sin(t)+0.85\sin(12\cdot t)+0.9\sin(8\cdot t)$ $f(t)=10\sin(2\cdot t)+2\sin(12\cdot t)+1.5\sin(10\cdot t)+10t+3$
10	$f(t)=0.5\cos(12\cdot t)+0.55\sin(17\cdot t)+0.65\sin(30\cdot t)+0.3\sin(15\cdot t)+12.45\sin(t)$ $f(t)=12.85\sin(1.7\cdot t)+0.9\sin(41\cdot t)+0.1\sin(10\cdot t)+0.2\sin(12\cdot t)+3t^2+t+2$

ПРОГРАММНАЯ ОБРАБОТКА ИЗОБРАЖЕНИЙ ПОРОГОВЫМИ МЕТОДАМИ

Цель работы: *изучение приёмов разработки программ для обработки изображений пороговыми методами*

Объект исследования: *методы пороговой обработки изображений.*

Аппаратные средства: *программа «виртуальный лабораторный комплекс Vision Lab», среда программирования C++ Builder.*

2.1. Краткие теоретические сведения

2.1.1. Общие сведения об обработке изображений

Первые научные труды, посвященные цифровой обработке изображений появились в 60-х годах XX века. Сейчас существует множество работ, посвящённых данной проблеме, причём развитие данного направления науки ускоряется общим прогрессом в области создания вычислительной техники. Разработки в области обработки изображений нашли применение в системах технического зрения, применяемых в робототехнике и мехатронике.

В системах технического зрения изображения обрабатываются с целью повысить их качество, выделить определённые объекты на изображении, получить из изображения информацию о геометрических свойствах изображенных объектов и т.п. Особенностью подобной обработки является, как правило, высокий объём обрабатываемой информации и разнообразие задач обработки. Причем многие задачи, такие как распознавание объектов, требуют индивидуального подхода. Вместе с тем существует ряд методов, применяющихся при решении широкого круга задач обработки изображений.

Среди методов цифровой обработки изображений можно выделить несколько групп. Это методы поэлементной обработки изображений, методы, направленные на выделение определённых участков изображения, методы распознавания объектов, получения дополнительной информации об объекте.

Методы поэлементной обработки направлены на повышение качества изображения. Причем понятие качества связано с задачей обработки, то есть *качество* – есть характеристика самого изображения, а не мера близости к эталону. Например, если задача – выделить некоторый объект на изображении, то качество будет определяться уровнем зашумлённости изображения, четкостью границ, между объектом и фоном, и др. Методы поэлементной обработки можно разделить на две группы: позволяющие повысить зрительные характеристики и методы «препарирования изображений». К методам улучшения зрительных характеристик относятся повышение контраста, четкости, выравнивание яркости и другие. Эти методы применяются для улучшения визуального восприятия изображения, т. е. направлены на преобразование изображения с учетом специфики человеческого зрения (и способов воспроизведения изображения).

Препарирование изображений – это обработка изображения, с целью подчёркивания на нём некоторых существенных деталей и особенностей, и подавления несущественных. Такая обработка производится для улучшения восприятия изображения, упрощения дальнейшего анализа. Следует отметить, что после подобной обработки изображение значительно искажается. К методам препарирования изображений можно отнести пороговую обработку, методы подчёркивания контуров и другие. Особенностью функций поэлементного преобразования для препарирования изображения является управляемость или возможность настройки под конкретную задачу. В системах технического зрения методы препарирования изображений играют важную роль, тогда как методы улучшения зрительных характеристик используются только в случаях, когда необходимо передать изображение оператору.

2.1.2. Методы обработки изображений

Поэлементная обработка

Все методы поэлементной обработки изображений имеют схожую алгоритмическую структуру. Для изображения, представленного двухмерным массивом значений цвета в общем случае алгоритм поэлементной обработки представлен на рис. 1. На схеме переменные i и j – индексы элементов изображения, X –

исходное изображение, Y – обработанное изображение, f – функция обработки. В дальнейшем, при рассмотрении алгоритмов поэлементной обработки, приведённая часть будет опускаться, и рассматриваться будет только функция обработки $Y_{i,j} = f(X_{i,j})$.

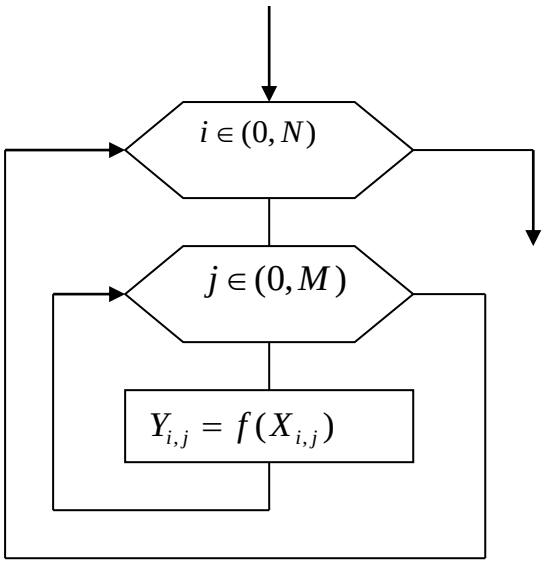

Рис. 2.1. Поэлементная обработка

Пороговая обработка

Пороговая обработка (иногда называется «бинаризацией»), это операция преобразования изображения в бинарное, двухградационное. Такое преобразование осуществляется для того, чтобы сократить информационную избыточность изображения, оставив ту информацию, которая нужна для решения конкретной задачи, и исключив несущественные особенности.

Суть метода состоит в том, что каждый элемент изображения сравнивается с неким пороговым значением. Причем сравниваться может яркость элемента или некоторая цветовая составляющая. Те элементы, значения яркости или цветности которых оказываются больше порогового считаются «единицами», а остальные «нулями» (если на выходе преобразователя должно быть изображение, то в соответствие единице и нулю ставятся некоторые цвета, например белый и черный). Алгоритм пороговой обработки представлен на рис.2. На схеме T – значение порога.

На рис. 3 представлено изображение до и после пороговой обработки.

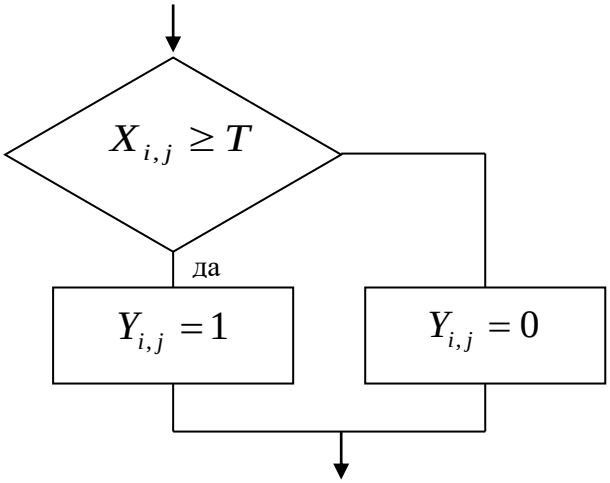

Рис. 2.2. Пороговая обработка

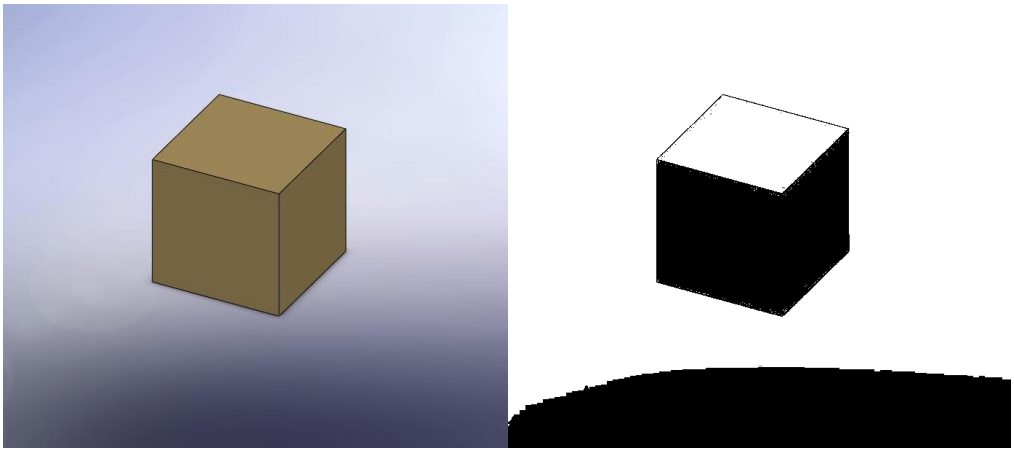

Рис. 2.3. Изображение до и после пороговой обработки; обработанное изображение показано справа

Математическое представление преобразования:

$$y(i,j) = \begin{cases} 1 & при \quad X(i,j) \geq T \\ 0 & при \quad X(i,j) < T \end{cases} \qquad (2.1).$$

2.2. Описание виртуального лабораторного комплекса Vision Lab

Практическое ознакомление с особенностями различных методов обработки изображений возможно с использованием виртуального лабораторного комплекса Vision Lab. Лабораторный комплекс представляет собой программу для ЭВМ, реализующую обработку изображений множеством методов, являющихся базовыми в обработке изображений в системах технического зрения.

Основные характеристики лабораторного комплекса:

- работая в лабораторном комплексе, можно применять реализованные методы как по отдельности, так и последовательно;
- в лабораторном комплексе реализованы поэлементные методы обработки изображений, в том числе пороговые методы, методы подчеркивания контуров, работа с цветовыми каналами;
- в лабораторном комплексе реализованы методы выделения участков изображения, в том числе поиск линий, контуров, крупных объектов на фоне шума;
- в лабораторном комплексе реализованы методы распознавания и получения дополнительной информации об объектах на изображении, в том числе метод поиска направления коридора и метод поиска пакетов в коридоре.

Чтобы применить реализованные в виртуальном лабораторном комплексе Vision Lab пороговые методы обработки, необходимо выбрать вкладку «Пороговая обработка». Чтобы произвести пороговую обработку по яркости следует нажать кнопку «Бинаризация по яркости» и задать пороговое значение яркости. Чтобы произвести пороговую обработку по цветовой составляющей следует нажать кнопку «Бинаризация по цветовым составляющим», выбрать цвет и задать пороговые значения яркости и цвета.

2.3. Методика выполнения лабораторной работы

Первая часть лабораторной работы заключается в обработке изображения с использованием виртуального лабораторного комплекса Vision Lab.

Вторая часть лабораторной работы заключается в реализации алгоритмов пороговой обработки на языке C++ в среде программирования C++ Builder.

Для того, чтобы организовать обработку изображения необходимо создать пользовательский интерфейс, позволяющий загружать изображения. Один из вариантов подобного интерфейса показан на рис. 4.

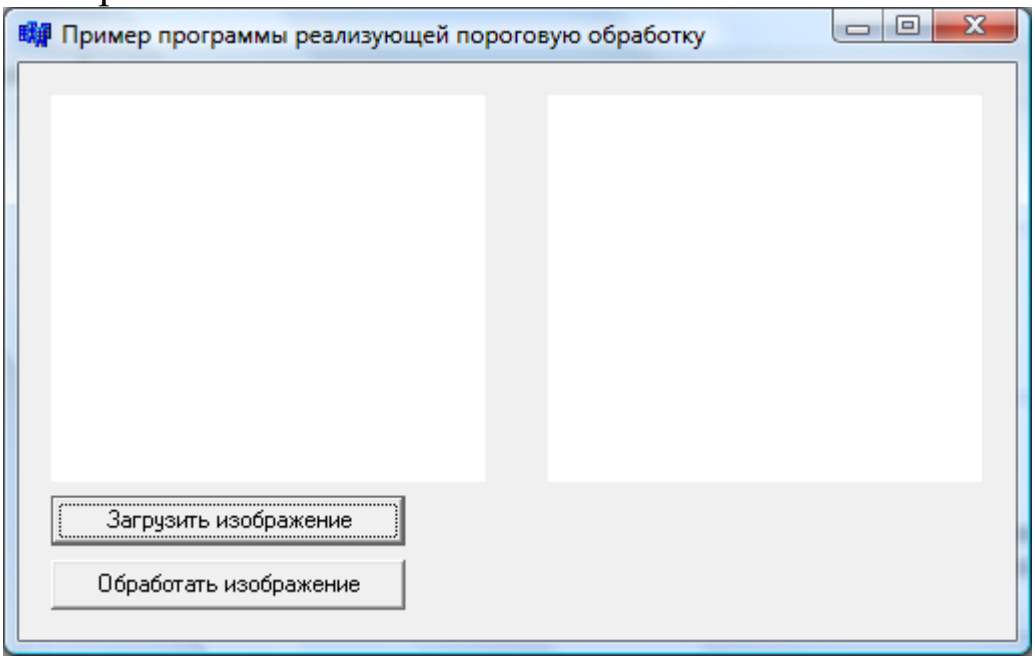

Рис. 2.4 Пример пользовательского интерфейса

Предложенный пример пользовательского интерфейса представляет собой четыре компонента из стандартной библиотеки C++ Builder – два компонента Image и два компонента Button. Компонент Image расположен на вкладке Additional панели компонентов (см. рис. 2.5).

Компонент Button расположен на вкладке Standard панели компонентов (см. рис. 2.5).

После размещения компонентов на форме следует настроить их свойства. Это может быть произведено путём изменения их свойств в окне Object Inspector. Окно Object Inspector отображающее свойства компонента Button показано на рис. 2.6.

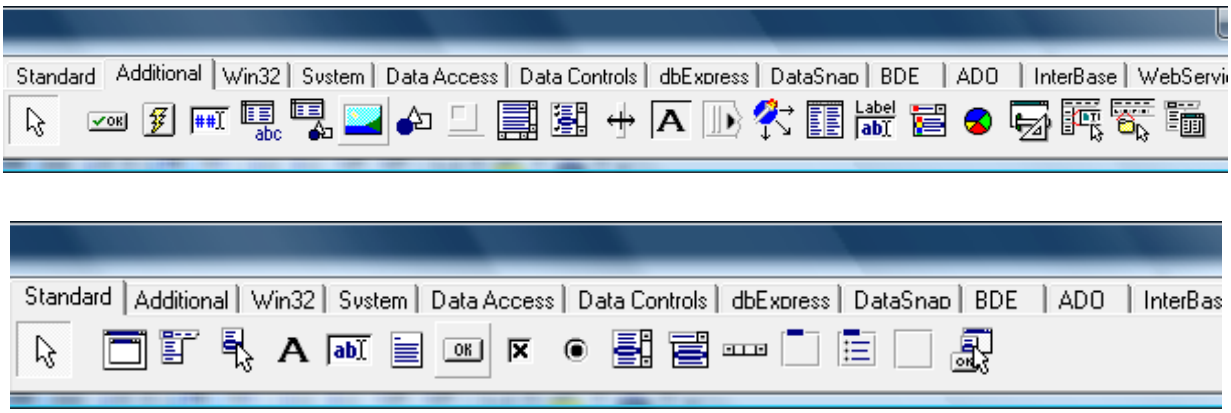

Рис. 2.5 Вкладка Additional и вкладка Standard панели компонентов C++ Builder

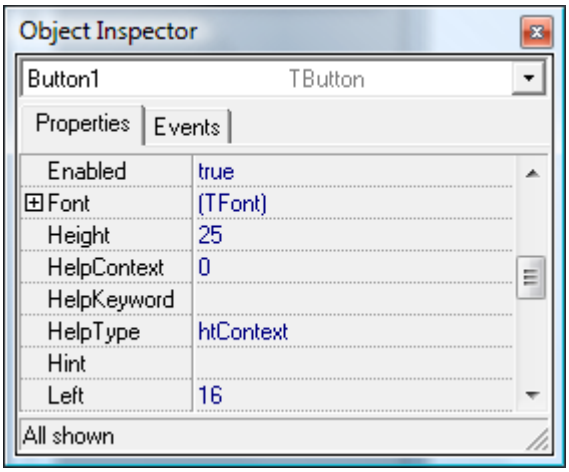

Рис. 2.6 Окно Object Inspector, отображающее свойства компонента Button

В предложенном примере были изменены свойства Name для компонентов Button, свойства Proportional и Stretch компонентов Image и свойства, определяющие положение компонентов на форме. Также, для того, чтобы сделать компоненты Image видимыми, пользователю необходимо загрузить в них некоторые изображения. Для этой цели можно использовать изображение, представляющее собой массив точек белого цвета. Можно загрузить такое изображение, используя окно Object Inspector для вызова диалогового окна Picture Editor. Диалоговое окно Picture Editor показано на рис. 2.7.

Также для загрузки изображений, которые будут отображаться по умолчанию, можно использовать специальный код (см. листинг 2.1).

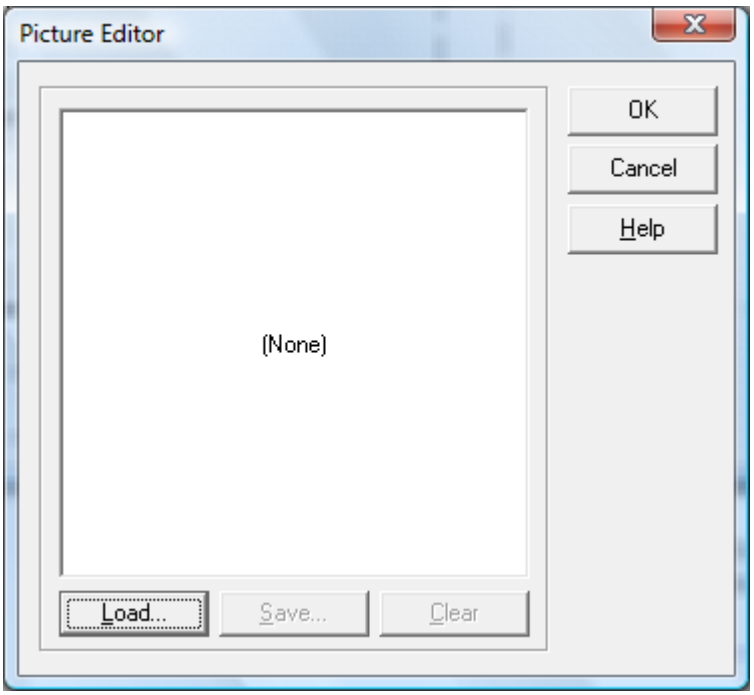

Рис. 2.7 Диалоговое окно Picture Editor

Листинг 2.1 Пример программного кода, реализующего загрузку изображения по умолчанию

```
void __fastcall TForm1::FormCreate(TObject *Sender)
{
Image1->Picture->LoadFromFile(ExtractFilePath( Application->ExeName ) + "\Начальное изображение.bmp");
Image2->Picture->LoadFromFile(ExtractFilePath( Application->ExeName ) + "\Начальное изображение.bmp");
}
```

Следует отметить, что функцию FormCreate среда C++ Builder может сгенерировать автоматически при двойном щелчке по форме.

После окончания работы над пользовательским интерфейсом следует разработать функции, которые будут выполняться при нажатии на кнопки (компоненты Button). В приведённом примере одна из кнопок выполняет загрузку изображения в один из компонентов Image, вторая выполняет обработку изображения.

Для организации загрузки изображения имеет смысл воспользоваться стандартным компонентом OpenDialog. Компонент OpenDialog расположен на вкладке Dialogs панели компонентов (см. рис. 2.8).

Рис. 2.8 Вкладка Dialogs панели компонентов C++ Builder

Загрузку изображений с использованием компонента OpenDialog реализует код показанный на листинге 2.2.

Листинг 2.2 Пример программного кода, реализующего загрузку изображения с использованием компонента OpenDialog

```
void __fastcall TForm1::Button1Click(TObject *Sender)
{
if (OpenDialog1->Execute())
Image1->Picture->LoadFromFile(OpenDialog1->FileName);
}
```

Для организации алгоритма пороговой обработки изображения, показанного на рис. 2.2, нужно преобразовать формат входных данных. Загруженное изображение представляет собой массив данных типа **TColor**. Нам необходимо преобразовать его, получив на выходе массив данных типа **int**, содержащий данные о яркости точек изображения. Для преобразования переменной типа **TColor** в переменную типа **int**, содержащую значение яркости, можно использовать код, показанный на листинге 2.3.

Листинг 2.3 Пример программного кода, реализующего преобразование данных типа TColor в данные типа int содержащие информацию о яркости.

```
int PixelBright(TColor Color)
{
int Blue, Green, Red;
int Result;

Blue = Color / (256*256);
Green = ( Color - (Blue * (256*256)) ) / 256;
Red = ( Color - (Blue * (256*256)) - (Green * 256) );

Result = Blue + Green + Red;
```

```
return Result;
}
```

Используя данную функцию представляется возможным реализовать алгоритм пороговой обработки изображения. Код, реализующий пороговую обработку показан на листинге 4.

Листинг 2.4 Код, реализующий пороговую обработку изображения.

```
void SizeKorrection(TImage *Input, TImage *Output)
{
if ((Input->Picture->Bitmap->Width != Output->Picture->Bitmap->Width) ||
   (Input->Picture->Bitmap->Height != Output->Picture->Bitmap->Height))
Output->Picture = Input->Picture;
}

void Binarisation(TImage *Input, TImage *Output, int Porog)
{
int CountX, CountY;
int i, j;
TColor Color;
int Lavel;

SizeKorrection(Input, Output);

CountX = Input->Picture->Bitmap->Width;
CountY = Input->Picture->Bitmap->Height;

for (i = 0; i < CountY; i++)
for (j = 0; j < CountX; j++)
{
Color = Input->Canvas->Pixels[j][i];

Lavel = PixelBright(Color);

 if (Lavel > Porog)
 Output->Canvas->Pixels[j][i] = clWhite;
```

```
 else
 Output->Canvas->Pixels[j][i] = clBlack;
}
}
```

Приведённый выше программный код реализует пороговую обработку всего изображения и загрузку результатов в второй компонент типа TImage. Вызов функции Binarisation производится из функции обработки нажатия кнопки на форме. Код, реализующий вызов функции Binarisation, показан на листинге 5. Приведённый код обрабатывает изображение с порогом 100.

Листинг 2.5 Пример программного кода, реализующего вызов функции Binarisation.

```
void __fastcall TForm1::Button2Click(TObject *Sender)
{
Binarisation(Image1, Image2, 100);
}
```

2.4. Задание на выполнение лабораторной работы и порядок её выполнения

Задание на выполнение лабораторной работы следует выбирать из таблицы 2.1 в соответствии с номером студента в списке группы.

Таблица 2.1 Задания на выполнение лабораторной работы

Вариант	p_1	p_2	p_3	p_4	p_5	Комментарий
1	100	150	200	5	720	Правая верхняя четверть
2	120	170	250	10	700	Левая верхняя четверть
3	50	100	200	1	680	Правая нижняя четверть
4	30	130	230	0	710	Левая нижняя четверть
5	10	200	300	20	705	Верхняя половина
6	110	180	280	25	690	Нижняя

						половина
7	70	170	270	15	695	Правая верхняя четверть
8	150	250	350	5	650	Левая верхняя четверть
9	200	300	400	10	640	Правая нижняя четверть
10	135	195	255	30	630	Левая нижняя четверть

Первая часть лабораторной работы заключается в обработке изображения с использованием виртуального лабораторного комплекса Vision Lab. Для выполнения обработки следует выполнить следующую последовательность действий:

1. Запустить приложение «виртуальный лабораторный комплекс Vision Lab»
2. Загрузить изображение, которое будет подвергнуто обработке
3. Выбрать вкладку «Пороговая обработка»
4. Нажать кнопку «Бинаризация по яркости» и задать пороговое значение яркости.
5. Произвести вышеуказанное действие для пороговых значений p_1, p_2, p_3 своего варианта
6. Произвести пороговую обработку по цветовой составляющей, нажав кнопку «Бинаризация по цветовым составляющим», выбрав цвет и задав пороговые значения яркости и цвета (p_4, p_5).

В отчёте о выполнении лабораторной работы должен быть показан процесс работы с программой и результаты обработки изображения.

Вторая часть лабораторной работы заключается в реализации алгоритмов пороговой обработки на языке C++ в среде программирования C++ Builder (см. п. 2.3).

Отчет о выполнении лабораторной работы должен содержать разработанный программный код и изображение программы во время работы. Также отчет должен содержать изображение до и после обработки.

В рамках выполнения лабораторной работы должна быть произведена пороговая обработка изображения с порогами p_1, p_2, p_3

Оформление отчета о выполнении лабораторной работы

Требования к отчету:

- отчет содержит титульный лист, описание выполняемого задания, описание проделанной работы, анализ полученных результатов, выводы, список использованной литературы;
- отчет выполняется на листах формата А4, 14 кегль, одинарный межстрочный интервал;
- список литературы оформляется согласно ГОСТ 7.1-2003.

Рекомендуемая литература

1. Кельберт М. Я., Что такое преобразование Фурье?, Матем. просв., сер. 3, 4, МЦНМО, М., с 188–202, 2000 г.
2. Привалов И.И. Ряды Фурье / Либроком, 168 стр., 2011 г.
3. Вернер Вольфганг Рогозинский, Годфри Гарольд Харди Ряды Фурье / Либроком 152 стр., 2009 г.
4. Власова Е.А. Ряды / МГТУ им. Н. Э. Баумана, Математика в техническом университете, 2006 г., 616 стр.
5. Колмогоров А. Н., Фомин С. В. Элементы теории функций и функционального анализа / Физматлит, 2012, 576 стр.
6. Воротников С.А. Информационные устройства робототехнических систем / Издательство МГТУ им. Н. Э. Баумана, 384 стр., 2005 г.
7. Яцун, С.Ф. Информационные устройства и системы в мехатронике [Текст]: учебное пособие / С.Ф. Яцун, П.А. Безмен // Курск: Юго-Зап. гос. ун-т. – 2013. – 240 с.
8. Яцун, С.Ф. Датчики и обработка сигналов в мехатронике [Текст]: учебное пособие / С.Ф. Яцун, П.А. Безмен // Курск: Юго-Зап. гос. ун-т. – 2014. – 238 с.