## ABSTRACT

Edge detection is used in computer vision applications for contours extraction of objects. The usual method is to use convolution operation of the image with complex filters like Sobel or Prewitt.

### Sobel Filter

### Real Imaginary

### 1.0 0.0 -1.0 1.0 2.0 1.0

### 2.0 0.0 -2.0 0.0 0.0 0.0

### 1.0 0.0 -1.0 -1.0 -2.0 -1.0

### Prewitt Filter

### Real Imagianry

### 0.5 0.0 -0.5 0.5 0.5 0.5

### 0.5 0.0 -0.5 0.0 0.0 0.0

### 0.5 0.0 -0.5 -0.5 -0.5 -0.5

You may extract the edges for example with vec2D wrapper described in article [Vector Class Wrapper SSE Optimized for Math Operations](http://www.codeproject.com/useritems/SSE_optimized_2D_vector.asp).

However unless integer optimized, floating point operations might take quite a long time. With wavelet transform, you might achieve similar results with a few mathematical operations. For example, Haar transform of the image provides details of that image contained in the high frequency bands very similar in appearance if you used X and Y difference filters on the same image.

### X Difference Filter Y Difference Filter

### 0.0 0.0 0.0 0.0 0.5 0.0

### 0.5 0.0 -0.5 0.0 0.0 0.0

### 0.0 0.0 0.0 0.0 -0.5 0.0

If we keep the details of the image obtained with Haar transform, remove the coarse-grained low frequency component and perform image reconstruction, we obtain the edges of the objects present in the image.

# **2. INTRODUCTION**

**2.1 Edge**

An edge in an image is a contour across which the brightness of the image changes abruptly. In image processing, an edge is often interpreted as one class of singularities. Singularities can be characterized easily as discontinuities where the gradient approaches infinity. However, image data is discrete, so edges in an image often are defined as the local maxima of the gradient.

## 2.2 Edge detection

Edge detection is an important task in image processing. It is a main tool pattern recognition, image segmentation, and scene analysis. An edge detector is basically a high pass filter that can be applied to extract the edge points in image.

## 2.3 Image Processing

Image processing involves changing the nature of an image in order to either

a. Improves its pictorial information for human interpretation.

b. Render it more suitable for machine perception.

Image processing is any form of [signalprocessing](http://en.wikipedia.org/wiki/Signal_processing) for which the input is an image, such as a photographer [videoframe](http://en.wikipedia.org/wiki/Video_frame); the [output](http://en.wikipedia.org/wiki/Output) of image processing may be either an image or a set of characteristics or [parameters](http://en.wikipedia.org/wiki/Parameter) related to the image. Most image-processing techniques involve treating the image as a [two dimensional](http://en.wikipedia.org/wiki/Two-dimensional) [signal](http://en.wikipedia.org/wiki/Signal_(electrical_engineering)) and applying standard signal-processing techniques to it.

Image processing refers to processing of 2D picture by a computer. An image defined in the “real world” is considered to be a function of two real variables.

Modern digital technology has made it possible to manipulate multi-dimensional signals with system that range from simple digital circuits to advance parallel computers. The goal of this manipulation can be divided into three categories:

* Image Processing (image in -> image out)
* Image Analysis (image in -> measurements out)
* Image Understanding (image in -> high-level description out)

## 2.4 Three Phases of Image Processing

1. **Image Recognition:**

The identification of object in an image. This process would probably start with image processing techniques such as noise removal, followed by feature extraction to locate lines, regions and possibly area with certain textures.

1. **Image Analysis:**

Image analysis is the extraction of meaningful information from image from digital image by means of digital image processing techniques. Image analysis task can be as simple as reading bar codes tags or as sophisticated as identifying a person from their face.

.

1. **Image detection:**

Image detection is an important task in image processing. It is a main tool pattern recognition, image segmentation, and scene analysis. An edge detector is basically a high pass filter that can be applied to extract the edge points in image

**2.5 Filters**

Edge detect filters search for borders between different colors and so can detect contours of objects.

They are used to make selections and for many artistic purposes.

* In this project we are detecting the edges of an image using wavelet transformation which contain RGB transform, absolute values, contract stretch.
* The RGB transform contain many filters such as bior13, coif1, duad10, inter1 and many more.

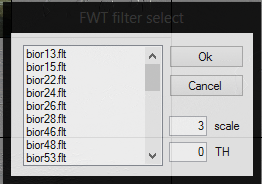


Fig.1 FWT filter

* Each filter have different wavelet transform.

3. LITERATURE SURVEY

**3.1 About Microsoft .Net**

Microsoft made the specifications for .net development platform freely available for the compiler vendors in the form of common language specification (CLS). The common language specifications provide the specifications for a language to compile into a common platform. The compiler vendors must design the compiler in such a way that the compiled code conforms these specifications. These compilers compile the programs written in the high level language into a format called intermediate language format.

Intermediate Language format

High Level Language

Compiler

Fig.2 Microsoft .NET

This IL code format is not the machine language code. So, in order to execute the program we need to compile it again into machine language. This is done by the Common Language Functions (CLR). The Just-in-time compiler (JIT compiler) of the CLR takes the IL code as input and Compiles it and executes it.

CLR

DLL in

IL Format (C.DLL)

.NET C# Compiler

Source Code in C#

Fig.3 A Sample view of .NET Framework

* **The .NET Framework has two main parts:**

1. The Common Language Runtime (CLR).

2. A hierarchical set of class libraries.

The CLR is described as the “execution engine” of .NET. It provides the environment within which programs run. The most important features are

* Conversion from a low-level assembler-style language, called Intermediate Language (IL), into code native to the platform being executed on.
* Memory management, notably including garbage collection.
* Checking and enforcing security restrictions on the running code.
* Loading and executing programs, with version control and other such features.
* The following features of the .NET framework are also worth description:
* **Managed Code**

The code that targets .NET, and which contains certain extra Information - “metadata” - to describe itself. Whilst both managed and unmanaged code can run in the runtime, only managed code contains the information that allows the CLR to guarantee, for instance, safe execution and interoperability.

* **Managed Data**

With Managed Code comes Managed Data. CLR provides memory allocation and Deal location facilities, and garbage collection. Some .NET languages use Managed Data by default, such as C#, Visual Basic.NET and JScript.NET, whereas others, namely C++, do not. Targeting CLR can, depending on the language you’re using, impose certain constraints on the features available. As with managed and unmanaged code, one can have both managed and unmanaged data in .NET applications - data that doesn’t get garbage collected but instead is looked after by unmanaged code.

* **Common Type System**

The CLR uses something called the Common Type System (CTS) to strictly enforce type-safety. This ensures that all classes are compatible with each other, by describing types in a common way. CTS define how types work within the runtime, which enables types in one language to interoperate with types in another language, including cross-language exception handling. As well as ensuring that types are only used in appropriate ways, the runtime also ensures that code doesn’t attempt to access memory that hasn’t been allocated to it.

* **Common Language Specification**

The CLR provides built-in support for language interoperability. To ensure that you can develop managed code that can be fully used by developers using any programming language, a set of language features and rules for using them called the Common Language Specification (CLS) has been defined. Components that follow these rules and expose only CLS features are considered CLS-compliant.

* **The Class Library**

.NET provides a single-rooted hierarchy of classes, containing over 7000 types. The root of the namespace is called System; this contains basic types like Byte, Double, Boolean, and String, as well as Object. All objects derive from System. Object. As well as objects, there are value types. Value types can be allocated on the stack, which can provide useful flexibility. There are also efficient means of converting value types to object types if and when necessary.

The set of classes is pretty comprehensive, providing collections, file, screen, and network I/O, threading, and so on, as well as XML and database connectivity.

The class library is subdivided into a number of sets (or namespaces), each providing distinct areas of functionality, with dependencies between the namespaces kept to a minimum.

* **Languages Supported By .Net**

The multi-language capability of the .NET Framework and Visual Studio .NET enables developers to use their existing programming skills to build all types of applications and XML Web services. The .NET framework supports new versions of Microsoft’s old favorites Visual Basic and C++ (as VB.NET and Managed C++), but there are also a number of new additions to the family.

Visual Basic .NET has been updated to include many new and improved language features that make it a powerful object-oriented programming language. These features include inheritance, interfaces, and overloading, among others. Visual Basic also now supports structured exception handling, custom attributes and also supports multi-threading.

Visual Basic .NET is also CLS compliant, which means that any CLS-compliant language can use the classes, objects, and components you create in Visual Basic .NET.

Managed Extensions for C++ and attributed programming are just some of the enhancements made to the C++ language. Managed Extensions simplify the task of migrating existing C++ applications to the new .NET Framework.

C# is Microsoft’s new language. It’s a C-style language that is essentially “C++ for Rapid Application Development”. Unlike other languages, its specification is just the grammar of the language. It has no standard library of its own, and instead has been designed with the intention of using the .NET libraries as its own.

Microsoft Visual J# .NET provides the easiest transition for Java-language developers into the world of XML Web Services and dramatically improves the interoperability of Java-language programs with existing software written in a variety of other programming languages.

Active State has created Visual Perl and Visual Python, which enable .NET-aware applications to be built in either Perl or Python. Both products can be integrated into the Visual Studio .NET environment. Visual Perl includes support for Active State’s Perl Dev Kit.

* **Other languages for which .NET compilers are available include**
* FORTRAN
* COBOL
* Eiffel

|  |  |
| --- | --- |
| ASP.NET  XML WEB SERVICES | Windows Forms |
| Base Class Libraries | |
| Common Language Runtime | |
| Operating System | |

Fig.4 **.**Net Framework

C#.NET is also compliant with CLS (Common Language Specification) and supports structured exception handling. CLS is set of rules and constructs that are supported by the CLR (Common Language Runtime). CLR is the runtime environment provided by the .NET Framework; it manages the execution of the code and also makes the development process easier by providing services.

C#.NET is a CLS-compliant language. Any objects, classes, or components that created in C#.NET can be used in any other CLS-compliant language. In addition, we can use objects, classes, and components created in other CLS-compliant languages in C#.NET .The use of CLS ensures complete interoperability among applications, regardless of the languages used to create the application.

* **Constructors And Destructors:**

Constructors are used to initialize objects, whereas destructors are used to destroy them. In other words, destructors are used to release the resources allocated to the object. In C#.NET the sub finalize procedure is available. The sub finalize procedure is used to complete the tasks that must be performed when an object is destroyed. The sub finalize procedure is called automatically when an object is destroyed. In addition, the sub finalize procedure can be called only from the class it belongs to or from derived classes.

* **Garbage Collection**

Garbage Collection is another new feature in C#.NET. The .NET Framework monitors allocated resources, such as objects and variables. In addition, the .NET Framework automatically releases memory for reuse by destroying objects that are no longer in use.

In C#.NET, the garbage collector checks for the objects that are not currently in use by applications. When the garbage collector comes across an object that is marked for garbage collection, it releases the memory occupied by the object.

* **Overloading**

Overloading is another feature in C#. Overloading enables us to define multiple procedures with the same name, where each procedure has a different set of arguments. Besides using overloading for procedures, we can use it for constructors and properties in a class.

* **Multithreading:**

C#.NET also supports multithreading. An application that supports multithreading can handle multiple tasks simultaneously, we can use multithreading to decrease the time taken by an application to respond to user interaction.

* **Structured Exception Handling**

C#.NET supports structured handling, which enables us to detect and remove errors at runtime. In C#.NET, we need to use Try…Catch…Finally statements to create exception handlers. Using Try…Catch…Finally statements, we can create robust and effective exception handlers to improve the performance of our application.

# **3.1.1 About C# .NET framework**

## Microsoft .NET

The Microsoft .NET software developers list can be downloaded from Microsoft official website. It contains the following:-

* Compiler for C#
* Common Language Runtime
* CLR Debugger
* .Net base classes
* Some utilities

# **C# Base Classes**

A significant part of the power of the .Net framework comes from the base classes supplied by Microsoft as part of the .NET framework. These classes are all callable from C# and provide the bind of basic functionality that is needed by many applications to perform, amongst other things, basic system, windows. The types of purposes you can use the base classes to do include

* String handling
* Arrays, lists, maps etc.,
* Accessing files and the file system
* Accessing the registry
* Security
* Windowing
* Windows messages
* Database access

Visual C# .NET 2003 is the modern, innovative programming language and tool for building .NET-connected software for Microsoft Windows, the Web, and a wide range of devices. With syntax that resembles C++, a flexible integrated development environment (IDE), and the capability to build solutions across a variety of platforms and devices, Visual C# .NET 2003 significantly eases the development of .NET-connected software.

Visual C# .NET builds on a strong C++ heritage. Immediately familiar to C++ and Java developers, C# is a modern and intuitive object-oriented programming language that offers significant improvements, including a unified type system, "unsafe" code for maximum developer control, and powerful new language constructs easily understood by most developers.

Developers can take advantage of an innovative component-oriented language with inherent support for properties, indexers, delegates, versioning, operator overloading, and custom attributes. With XML comments, C# developers can produce useful source code documentation. An advanced inheritance model enables developers to reuse their code from within any programming language that supports .NET.

C# developers can join the newest, fastest-growing developer community, in which they can exchange code and resources, leverage skills across multiple computing environments, and contribute to the standardization process that ensures vibrant and active community participation.

With a superior IDE, Visual C# .NET provides users with the ultimate developer environment, bringing together the development community and valuable online resources. The Start Page offers developers a one-click portal to updates, preferences, information on recently used projects, and the MSDN Online community. Improved IntelliSense, the Toolbox, and the Task List provide significant productivity enhancements, while Auto Hide windows and multiple-monitor support help programmers maximize screen real estate and customize their development environment. New custom build rules make developing robust and powerful software easier than ever.

Using the Web Forms Designer and XML Designer, developers can use IntelliSense features and tag completion or the WYSIWYG editor for drag-and-drop authoring to build interactive Web applications. With a few simple steps, programmers can design, develop, debug, and deploy powerful XML Web services that reduce development time by encapsulating business processes accessible from any platform.

With Visual C# .NET 2003, developers can take advantage of Microsoft .NET and incorporate next-generation technology for resource management, unified types, and remoting. With Microsoft .NET, developers gain superior memory management technology for seamless garbage collection and reduced program complexity. Developers can use the Microsoft .NET Framework Common Type System to leverage code written in any of more than 20 languages that support .NET, while making efficient remote procedure calls.

Developers can also use the tested and proven .NET Framework class library to gain powerful built-in functionality, including a rich set of collection classes, networking support, multithreading support, string and regular expression classes, and broad support for XML, XML schemas, XML namespaces, XSLT, XPath, and SOAP. And, with the Java Language Conversion Assistant (JLCA), programmers can begin migrating their Java-based projects to the Microsoft .NET environment.

Using Visual C# .NET 2003, developers can construct powerful Web services that encapsulate business processes and make them available to applications running on any platform. Developers can easily incorporate any number of Web services that are catalogued and available in many independent Universal Description, Discovery, and Integration (UDDI) directories, providing a strong foundation of services and business logic for their applications.

Visual C# .NET 2003 also enables developers to build the next generation of Windows-based applications. With visual inheritance, developers can greatly simplify the creation of Windows-based applications by centralizing in parent forms the common logic and user interface for their entire solution. Using control anchoring and docking, programmers can build resizable forms automatically, while the in-place menu editor enables developers to visually author menus directly from within the Forms Designer.

Visual C# .NET 2003 is a modern, innovative programming language and tool for building .NET-connected software for Microsoft Windows, the Web, and a wide range of devices. With familiar C++-like syntax, a flexible integrated development environment (IDE), and the capability to build solutions across a variety of platforms and devices, Visual C# .NET 2003 significantly eases the development of .NET-connected software.

Visual C# .NET provides users with a superior developer environment, bringing together the development community and valuable online resources. The Start Page offers developers a one-click portal to updates, preferences, information on recently used projects, and the MSDN Online community. Improved IntelliSense, the Toolbox, and the Task List provide significant productivity enhancements, while Auto Hide windows and multiple-monitor support help programmers maximize screen real estate and customize their development environment.

With Visual C# .NET 2003, developers can take advantage of Microsoft .NET and incorporate next-generation technology for resource management, unified types, and remoting. With Microsoft .NET, developers gain superior memory management technology for seamless garbage collection and reduced program complexity. Developers can use the Microsoft .NET Framework Common Type System to leverage code written in any of more than 20 languages that support .NET, while making efficient emote procedure calls.

Literature [survey](http://www.blurtit.com/q876299.html) is the most important step in software development process. Before developing the tool it is necessary to determine the time factor, economy n company strength. Once these things r satisfied, ten next steps is to determine which operating system and language can be used for developing the tool. Once the [programmers](http://www.blurtit.com/q876299.html) start building the tool the programmers need lot of external support. This support can be obtained from senior programmers, from [book](http://www.blurtit.com/q876299.html) or from websites. Before building the system the above consideration r taken into account for developing the proposed system.

3.1.2 Features of .Net

Microsoft .NET is a set of Microsoft software technologies for rapidly building and integrating XML Web services, Microsoft Windows-based applications, and Web solutions. The .NET Framework is a language-neutral platform for writing programs that can easily and securely interoperate. There’s no language barrier with .NET: there are numerous languages available to the developer including Managed C++, C#, Visual Basic and Java Script. The .NET framework provides the foundation for components to interact seamlessly, whether locally or remotely on different platforms. It standardizes common data types and communications protocols so that components created in different languages can easily interoperate.

“.NET” is also the collective name given to various software components built upon the .NET platform. These will be both products (Visual Studio.NET and Windows.NET Server, for instance) and services (like Passport, .NET My Services, and so on).

**3.1.3 Objectives Of .Net Framework**

1. To provide a consistent object-oriented programming environment whether object codes is stored and executed locally on Internet-distributed, or executed remotely.

2. To provide a code-execution environment to minimizes software deployment and guarantees safe execution of code.

3. Eliminates the performance problems.

There are different types of application, such as Windows-based applications and Web-based applications.

3.2 ABOUT Of SQL-Server

The OLAP Services feature available in SQL Server version 7.0 is now called SQL Server 2000 Analysis Services. The term OLAP Services has been replaced with the term Analysis Services. Analysis Services also includes a new data mining component. The Repository component available in SQL Server version 7.0 is now called Microsoft SQL Server 2000 Meta Data Services. References to the component now use the term Meta Data Services. The term repository is used only in reference to the repository engine within Meta Data Services

SQL-SERVER database consist of six type of objects,

They are,

1. TABLE

2. QUERY

3. FORM

4. REPORT

5. MACRO

**3.2.1 Table**

A database is a collection of data about a specific topic.

* **Views Of Table**

We can work with a table in two types,

1. Design View

2. Datasheet View

* **Design View**

To build or modify the structure of a table we work in the table design view. We can specify what kind of data will be hold.

#### Datasheet View

To add, edit or analyses the data itself we work in tables datasheet view mode.

**3.2.2 Query**

A query is a question that has to be asked the data. Access gathers data that answers the question from one or more table. The data that make up the answer is either dynasty (if you edit it) or a snapshot (it cannot be edited).Each time we run query, we get latest information in the dynasty. Access either displays the dynasty or snapshot for us to view or perform an action on it, such as deleting or updating.

**3.3 Image**

An image (from [Latin](http://en.wikipedia.org/wiki/Latin_language): *imago*) is an artifact that depicts or records visual perception, for example a [two-dimensional](http://en.wikipedia.org/wiki/Two-dimensional) picture, that has a similar appearance to some [subject](http://en.wikipedia.org/wiki/Subject_(philosophy)) – usually a physical object or a [person](http://en.wikipedia.org/wiki/Person), thus providing a [depiction](http://en.wikipedia.org/wiki/Depiction) of it.

Images may be two-[dimensional](http://en.wikipedia.org/wiki/Dimension), such as a [photograph](http://en.wikipedia.org/wiki/Photograph), screen display, and as well as a three-dimensional, such as a [statue](http://en.wikipedia.org/wiki/Statue) or [hologram](http://en.wikipedia.org/wiki/Hologram). They may be captured by [optical](http://en.wikipedia.org/wiki/Optics) devices – such as [cameras](http://en.wikipedia.org/wiki/Camera), [mirrors](http://en.wikipedia.org/wiki/Mirror), [lenses](http://en.wikipedia.org/wiki/Lens_(optics)), [telescopes](http://en.wikipedia.org/wiki/Telescope), [microscopes](http://en.wikipedia.org/wiki/Microscope), etc. and natural objects and phenomena, such as the [human eye](http://en.wikipedia.org/wiki/Human_eye) or water surfaces.

The word image is also used in the broader sense of any two-dimensional figure such as a [map](http://en.wikipedia.org/wiki/Map), a [graph](http://en.wikipedia.org/wiki/Graph_(data_structure)), a [pie chart](http://en.wikipedia.org/wiki/Pie_chart), or an [abstract painting](http://en.wikipedia.org/wiki/Abstract_art). In this wider sense, images can also be rendered manually, such as by [drawing](http://en.wikipedia.org/wiki/Drawing), [painting](http://en.wikipedia.org/wiki/Painting), [carving](http://en.wiktionary.org/wiki/carving), rendered automatically by [printing](http://en.wikipedia.org/wiki/Printing) or [computer graphics](http://en.wikipedia.org/wiki/Computer_graphics) technology, or [developed](http://en.wikipedia.org/wiki/Image_development_(visual_arts)) by a combination of methods, especially in a [pseudo-photograph](http://en.wikipedia.org/wiki/Pseudo-photograph).

A volatile image is one that exists only for a short period of time. This may be a reflection of an object by a mirror, a projection of a [camera obscura](http://en.wikipedia.org/wiki/Camera_obscura), or a scene displayed on a [cathode ray tube](http://en.wikipedia.org/wiki/Cathode_ray_tube). A fixed image, also called a [hard copy](http://en.wikipedia.org/wiki/Hard_copy), is one that has been recorded on a material object, such as [paper](http://en.wikipedia.org/wiki/Paper) or [textile](http://en.wikipedia.org/wiki/Textile) by [photography](http://en.wikipedia.org/wiki/Photography) or any other digital process.

#### 3.4 Edge Detection

Edge detection is in which aim at identifying points in a [digital image](http://en.wikipedia.org/wiki/Digital_image) at which the [image brightness](http://en.wikipedia.org/wiki/Luminous_intensity) changes sharply or, more formally, has discontinuities. The points at which image brightness changes sharply are typically organized into a set of curved line segments termed edges. The same problem of finding discontinuities in 1D signal is known as [step detection](http://en.wikipedia.org/wiki/Step_detection) and the problem of finding signal discontinuities over time is known as [change detection](http://en.wikipedia.org/wiki/Change_detection). Edge detection is a fundamental tool in image processing, machine vision and computer vision, particularly in the areas of feature detection and feature extraction.

**3.4.1 Different Edge Detection Methods**

* **Sobel Operator**

The operator consists of a pair of 3×3 convolution kernels as shown in below figure. One kernel is simply the other rotated by 90°.



Masks used by Sobel Operator

These kernels are designed to respond maximally to edges running vertically and horizontally relative to the pixel grid, one kernel for each of the two perpendicular orientations. The kernels can be applied separately to the input image, to produce separate measurements of the gradient component in each orientation (call these Gx and Gy). These can then be combined together to find the absolute magnitude of the gradient at each point and the orientation of that gradient. The gradient magnitude is given by:



Typically, an approximate magnitude is computed using:



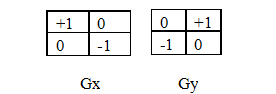
Which is much faster to compute.

The angle of orientation of the edge (relative to the pixel grid) giving rise to the spatial gradient is given by:



* **Robert’s cross operator:**

The Roberts Cross operator performs a simple, quick to compute, 2-D spatial gradient measurement on an image. Pixel values at each point in the output represent the estimated absolute magnitude of the spatial gradient of the input image at that point. The operator consists of a pair of 2×2 convolution kernels as shown in Figure 2. One kernel is simply the other rotated by 90°[4]. This is very similar to the Sobel operator.



Masks used for Robert operator.

These kernels are designed to respond maximally to edges running at 45° to the pixel grid, one kernel for each of the two perpendicular orientations. The kernels can be applied separately to the input image, to produce separate measurements of the gradient component in each orientation (call these Gx and Gy ). These can then be combined together to find the absolute magnitude of the gradient at each point and the orientation of that gradient. The gradient magnitude is given by:



Although typically, an approximate magnitude is computed using:



Which is much faster to compute.

The angle of orientation of the edge giving rise to the spatial gradient (relative to the pixel grid orientation) is given by:



* **Prewitt’s operator:**

Prewitt operator is similar to the Sobel operator and is used for detecting vertical and horizontal edges in images.



Masks for the Prewitt gradient edge detector

**3.5 Wavelet Transform**

In [mathematics](http://en.wikipedia.org/wiki/Mathematics), a wavelet series is a representation of a [square-integrable](http://en.wikipedia.org/wiki/Square-integrable) ([real](http://en.wikipedia.org/wiki/Real_number)- or [complex](http://en.wikipedia.org/wiki/Complex_number)-valued) [function](http://en.wikipedia.org/wiki/Function_%28mathematics%29) by a certain [orthonormal](http://en.wikipedia.org/wiki/Orthonormal)[series](http://en.wikipedia.org/wiki/Series_%28mathematics%29) generated by a [wavelet](http://en.wikipedia.org/wiki/Wavelet). Nowadays, wavelet transformation is one of the most popular candidates of the time-frequency-transformations. This article provides a formal, mathematical definition of an orthonormal wavelet and of the integral wavelet transform.

Wavelet transforms also decompose a signal into a set of "frequency bands" (referred to as scales) by projecting the signal onto an element of a set of basic functions. Although the scales do not live in the frequency domain, projection of the signal onto different scales is equivalent to band pass filtering with a bank of constant-Q filters. The basic functions are called wavelets. Wavelets in a basis are all similar to each other, varying only by dilation and translation.

##### 3.5.1 A Motivation for Wavelets

The short-time Fourier transform is frequently utilized for no stationary signal analysis. Although a powerful tool, it has some limitations in analyzing time-localized events. The wavelet transform has similarities with the short-time Fourier transform, but it also possesses a time-localization property that generally renders it superior for analyzing no stationary phenomena. We now review the Fourier and short-time Fourier trans- forms, discuss some often desirable properties that the short-time Fourier transform does not possess, and introduce the wavelet transform.

4. SYSTEM REQUIRMENT SPECIFICATIONS

**4.1 Hardware requirements**

SYSTEM : Pentium III 700 MHz

HARD DISK : 40 GB

MONITOR : 15 VGA colour monitor

RAM : 256MB

**4.2 Software requirements**

OPERATING SYSTEM : Windows XP Professional

FRONT END : Microsoft Visual Studio .Net 2010

BACK END : SQL SERVER 2008

CODING LANGUAGE : C# .Net

###### 5. BACKGROUND

#### Some techniques of Edge Detections

* **Laplacian**

It wishes to build a morphing algorithm which operates on features extracted from target images automatically. It can be a good beginning to find the edges in the target images.

* **Marr-Hildreth**

Marr-Hildreth uses the Gaussian smoothing operator to improve the response to noise, which is differentiated by the Laplacian of Gaussian is called the LoG operator.

* **Mean Shift Smoothing**

The difference mean shift smoothing makes in the edge detection task can be observed by performing the same edge detection process on an image that has been smoothed, and comparing it with the outcome of the edge detector on the untouched image. The discussion about whether mean shift smoothing improves edge detection in the general case remains open, since there is no clear rule describing the cases in which mean shift smoothing does not yield better results (as will be presented in the following section). Nevertheless, it is safe to say that in many cases this smoothing process can indeed improve edge detection.

### Canny edge detection

[John Canny](http://en.wikipedia.org/wiki/John_Canny) considered the mathematical problem of deriving an optimal smoothing filter given the criteria of detection, localization and minimizing multiple responses to a single edge. He showed that the optimal filter given these assumptions is a sum of four exponential terms. He also showed that this filter can be well approximated by first-order derivatives of Gaussians. Canny also introduced the notion of non-maximum suppression, which means that given the presmoothing filters, edge points are defined as points where the gradient magnitude assumes a local maximum in the gradient direction. Looking for the zero crossing of the 2nd derivative along the gradient direction was first proposed by [Haralick](http://en.wikipedia.org/wiki/Haralick" \o "Haralick). It took less than two decades to find a modern geometric variation meaning for that operator that links it to the [Marr–Hildreth](http://en.wikipedia.org/wiki/Marr-Hildreth_algorithm) (zero crossing of the Laplacian) edge detector. That observation was presented by [Ron Kimmel](http://en.wikipedia.org/wiki/Ron_Kimmel) and [Alfred Bruckstein](http://en.wikipedia.org/w/index.php?title=Alfred_Bruckstein&action=edit&redlink=1).

Although his work was done in the early days of computer vision, the [Canny edge detector](http://en.wikipedia.org/wiki/Canny_edge_detector) (including its variations) is still a state-of-the-art edge detector. Unless the preconditions are particularly suitable, it is hard to find an edge detector that performs significantly better than the Canny edge detector.

The Canny-Deriche detector was derived from similar mathematical criteria as the Canny edge detector, although starting from a discrete viewpoint and then leading to a set of recursive filters for image smoothing instead of [exponential filters](http://en.wikipedia.org/w/index.php?title=Exponential_filter&action=edit&redlink=1) or Gaussian filters.

The [differential edge detector](http://en.wikipedia.org/wiki/Edge_detection#Differential_edge_detection) described below can be seen as a reformulation of Canny's method from the viewpoint of differential invariants computed from a [scale space representation](http://en.wikipedia.org/wiki/Scale_space_representation) leading to a number of advantages in terms of both theoretical analysis and sub-pixel implementation.

6. PROPOSED SYSTEM

**6.1 Advantages of Edge Detection Using Wavelet**

* Allow the components of a non-stationary signal to be analyzed.
* Allow filters to be constructed for stationary and non-stationary signals Wells ,Strang.
* Have been applied in many other areas including non-linear regression and compression.
* Well localized in both time and frequency domain
* Give a better signal representation using Multiresolution analysis Walnut

**6.2 Disadvantage of Other Detection Methods**

* The result is binary.
* The amount of parameters leads to infinitely tweaking for getting just that little better result
* You still need connect the resulting edge to extract the complete edges that seem so obvious for the human eye mind.
* It generates responses that do not correspond to edges, so-called "false edges", and the localization error may be severe at curved edges
* If they are used mechanically without the knowledge of the actual theory involved, they sometimes yield erroneous results.

**7. SYSTEM DESIGN**

**7.1 INPUT DESIGN**

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

* What data should be given as input?
* How the data should be arranged or coded?
* The dialog to guide the operating personnel in providing input.
* Methods for preparing input validations and steps to follow when error occur.

**7.1.2 OBJECTIVES**

1. Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.

2. It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.

3. When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the userwill not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow

**7.2 OUTPUT DESIGN**

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system’s relationship to help user decision-making.

1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.

2. Select methods for presenting information.

3. Create document, report, or other formats that contain information produced by the system.

The output form of an information system should accomplish one or more of the following objectives.

* Convey information about past activities, current status or projections of the
* Future.
* Signal important events, opportunities, problems, or warnings.
* Trigger an action.
* Confirm an action.

8. IMPLEMENTATION

## 8.1 SYSTEM STUDY

**8.1.1 Feasibility Study**

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

* Economical Feasibility
* Technical Feasibility
* Social Feasibility
* **Economical Feasibility**

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

### Technical Feasibility

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

* **Social Feasibility**

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

### 8.2 SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the

Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

TYPES OF TESTS

* Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

* Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

* Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

* Valid Input : identified classes of valid input must be accepted.
* Invalid Input : identified classes of invalid input must be rejected.
* Functions : identified functions must be exercised.
* Output : identified classes of application outputs must be exercised.
* Systems/Procedures : interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

* System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

* White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

* Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

* **Unit Testing:**

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

**Test objectives**

* All field entries must work properly.
* Pages must be activated from the identified link.
* The entry screen, messages and responses must not be delayed.

**Features to be tested**

* Verify that the entries are of the correct format
* No duplicate entries should be allowed
* All links should take the user to the correct page.

# **Integration Testing**

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

**Test Results:**

All the test cases mentioned above passed successfully. No defects encountered.

* Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

**Test Results:**

All the test cases mentioned above passed successfully. No defects encountered.

**8.3 Edge detection of image**

Open the image and transform it to 1, 2 or 3 scales. You might add the threshold to remove the noise. Below, the daub1 filter is selected with 1 scale transform without denoising.

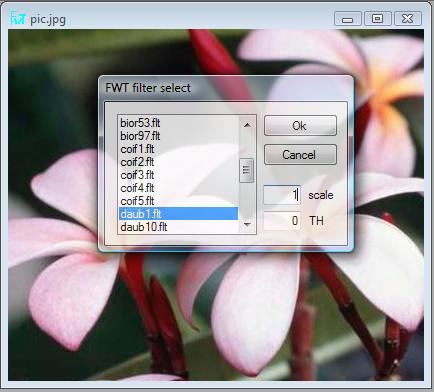


Fig.5 Opening the flower image and applying filter

You will get this FWT spectrum:

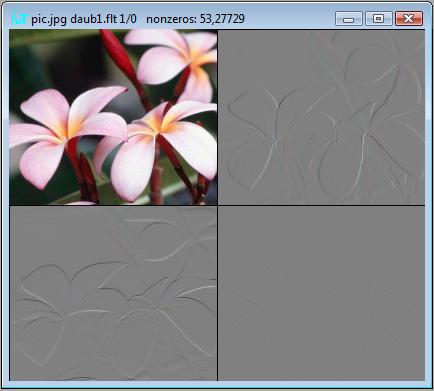


Fig.6 After applying the daub1.flt to the flower image

Now click Transform->Denoise menu item to remove low frequency component:

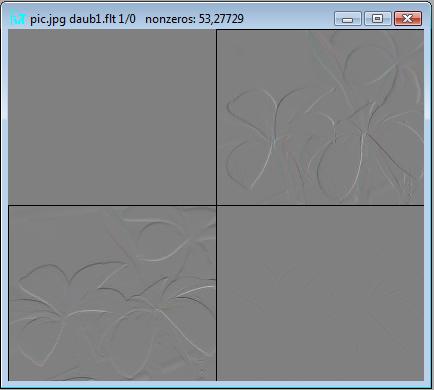


Fig.7 Removed the noise from the flower image

You might find the corresponding function in the BaseFWT2D class:

void BaseFWT2D::remove\_LLband()

{

if(m\_status<=0)

return;

unsignedint width=m\_width / (unsigned int) (pow(2.0f,(float)getJ()));

unsignedint height= m\_height / (unsigned int) (pow(2.0f,(float)getJ()));

for(unsigned int y=0; y<height;y++)

for(unsigned int x=0; x<width; x++)

spec2d[y][x]=0;

}

###### 9. RESULT



Now click Transform->FWT2D RGB synth to reconstruct the image:

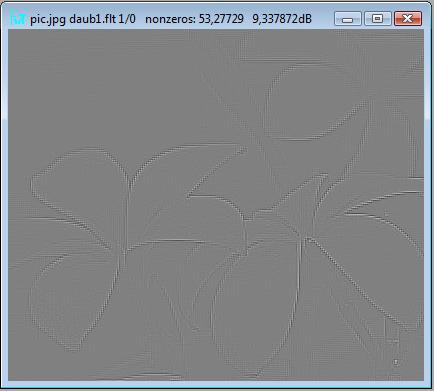


Fig.8 After giving the FWT2D RGB synth to the flower image

It does not seem like the edges yet. You need just subtract 128 from the image and compute the absolute value with

Transform->Abs values in menu item:

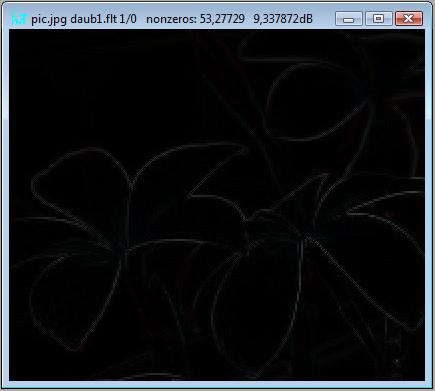


Fig.9 After giving the abs value

But the edges are rather vague. First I proceeded with contrast stretching, that is normalizing the image to 0 ... 255 range. But there might be several pixels at the upper limit of the range and it does not really improve the situation. The better choice would be non-linear normalization like logarithmic scale but I just multiply the pixel data by some value and obtain the more prominent edges. For 1 scale transform, the multiplication by 7 works well and does not overflow the 255 limit for the majority of pixels, but for 2 or 3 scales you might diminish the multiplication number.

Now click Transform->Contrast stretch to amplify your edges:

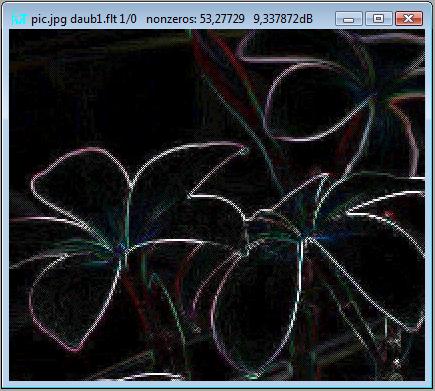


Fig.10 After giving the Contrast Stretch

You may compare the same picture results I obtained with the Sobel filter. It looks smoother, but then you might proceed to morphological operations like erosion and dilation and get a thin skeleton of the contour so in the end.

The results will be very close.

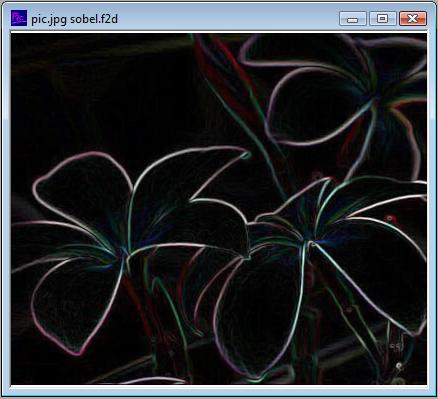


Fig.11 The Edge detected image of the flower image

I've developed Haar transform MMX optimization and in future, plan to provide an update to the code as EdgeDetector class or something similar and compare the performance.

**10. CONCLUSION**

Wavelet-based image compression method using in JPEG2000 is the new standard for still imagecompression. It provides a new frameworkand an integrated toolbox to better address increasingneeds for compression. It also provides a wide rangeof functionalities for still image applications.Lossless and lossy coding, embedded lossy to lossless,progressive by resolution and quality, highcompression efficiency, error resilience and losslesscolor transformations are some of its characteristics.Comparative results have shown that JPEG2000 isindeed superior to existing still image compressionstandards. Work is still needed in optimizing itsimplementation performance.

**11. APPENDIXES**

**Assemblyinfo.h**

#include "stdafx.h"

using namespace System;

using namespace System::Reflection;

using namespace System::Runtime::CompilerServices;

using namespace System::Runtime::InteropServices;

using namespace System::Security::Permissions;

//

// General Information about an assembly is controlled through the following

// set of attributes. Change these attribute values to modify the information

// associated with an assembly.

//

[assembly:AssemblyTitleAttribute("Img")];

[assembly:AssemblyDescriptionAttribute("")];

[assembly:AssemblyConfigurationAttribute("")];

[assembly:AssemblyCompanyAttribute("cam")];

[assembly:AssemblyProductAttribute("Img")];

[assembly:AssemblyCopyrightAttribute("Copyright (c) cam 2006")];

[assembly:AssemblyTrademarkAttribute("")];

[assembly:AssemblyCultureAttribute("")];

//

// Version information for an assembly consists of the following four values:

//

// Major Version

// Minor Version

// Build Number

// Revision

//

// You can specify all the value or you can default the Revision and Build Numbers

// by using the '\*' as shown below:

[assembly:AssemblyVersionAttribute("1.0.\*")];

[assembly:ComVisible(false)];

[assembly:CLSCompliantAttribute(true)];

[assembly:SecurityPermission(SecurityAction::RequestMinimum, UnmanagedCode = true)];

**Form1.h**

#pragma once

#include "fchildpic.h"

namespace Img

{

using namespace System;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Drawing;

/// <summary>

/// Summary for Form1

///

/// WARNING: If you change the name of this class, you will need to change the

/// 'Resource File Name' property for the managed resource compiler tool

/// associated with all .resx files this class depends on. Otherwise,

/// the designers will not be able to interact properly with localized

/// resources associated with this form.

/// </summary>

public ref class Form1 : public System::Windows::Forms::Form

{

public:

Form1(void)

{

InitializeComponent();

dir = IO::Directory::GetCurrentDirectory();

}

String^ dir; //current directory

protected:

/// <summary>

/// Clean up any resources being used.

/// </summary>

~Form1()

{

if (components)

{

delete components;

}

}

private:

System::Windows::Forms::MenuStrip^ mainMenu;

protected:

private:

System::Windows::Forms::ToolStripMenuItem^ fileToolStripMenuItem;

private:

System::Windows::Forms::ToolStripMenuItem^ openToolStripMenuItem;

private:

System::Windows::Forms::ToolStripSeparator^ toolStripMenuItem1;

private:

System::Windows::Forms::ToolStripMenuItem^ exitToolStripMenuItem;

private:

System::Windows::Forms::OpenFileDialog^ openPicDialog;

private:

/// <summary>

/// Required designer variable.

/// </summary>

System::ComponentModel::Container^ components;

#pragma region Windows Form Designer generated code

/// <summary>

/// Required method for Designer support - do not modify

/// the contents of this method with the code editor.

/// </summary>

void InitializeComponent(void) {

System::ComponentModel::ComponentResourceManager^ resources = (gcnew System::ComponentModel::ComponentResourceManager(Form1::typeid));

this->mainMenu = (gcnew System::Windows::Forms::MenuStrip());

this->fileToolStripMenuItem = (gcnewSystem::Windows::Forms::ToolStripMenuItem());

this->openToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->toolStripMenuItem1 = (gcnew System::Windows::Forms::ToolStripSeparator());

this->exitToolStripMenuItem = (gcnew System::Windows::Forms::ToolStripMenuItem());

this->openPicDialog = (gcnew System::Windows::Forms::OpenFileDialog());

this->mainMenu->SuspendLayout();

this->SuspendLayout();

//

// mainMenu

//

this->mainMenu->Items->AddRange(gcnew cli::array < System::Windows::Forms::ToolStripItem^ > (1)

{

this->fileToolStripMenuItem

}

);

this->mainMenu->Location = System::Drawing::Point(0, 0);

this->mainMenu->Name = L"mainMenu";

this->mainMenu->Size = System::Drawing::Size(442, 24);

this->mainMenu->TabIndex = 1;

this->mainMenu->Text = L"mainMenu";

//

// fileToolStripMenuItem

//

this->fileToolStripMenuItem->DropDownItems->AddRange(gcnew cli::array < System::Windows::Forms::ToolStripItem^ > (3)

{

this->openToolStripMenuItem,

this->toolStripMenuItem1, this->exitToolStripMenuItem

}

);

this->fileToolStripMenuItem->Name = L"fileToolStripMenuItem";

this->fileToolStripMenuItem->Size = System::Drawing::Size(35, 20);

this->fileToolStripMenuItem->Text = L"&File";

//

// openToolStripMenuItem

//

this->openToolStripMenuItem->Name = L"openToolStripMenuItem";

this->openToolStripMenuItem->ShortcutKeys = static\_cast<System::Windows::Forms::Keys>((System::Windows::Forms::Keys::Control | System::Windows::Forms::Keys::O));

this->openToolStripMenuItem->Size = System::Drawing::Size(166, 22);

this->openToolStripMenuItem->Text = L"&Open ...";

this->openToolStripMenuItem->Click += gcnew System::EventHandler(this, &Form1::openToolStripMenuItem\_Click);

//

// toolStripMenuItem1

//

this->toolStripMenuItem1->Name = L"toolStripMenuItem1";

this->toolStripMenuItem1->Size = System::Drawing::Size(163, 6);

//

// exitToolStripMenuItem

//

this->exitToolStripMenuItem->Name = L"exitToolStripMenuItem";

this->exitToolStripMenuItem->ShortcutKeys = static\_cast<System::Windows::Forms::Keys>((System::Windows::Forms::Keys::Control | System::Windows::Forms::Keys::X));

this->exitToolStripMenuItem->Size = System::Drawing::Size(166, 22);

this->exitToolStripMenuItem->Text = L"E&xit";

//

// openPicDialog

//

this->openPicDialog->FileName = L"openPicDialog";

this->openPicDialog->Filter = L"Image files|\*.bmp;\*.jpg;\*.jpeg;\*.png;\*.tif;\*.tiff";

//

// Form1

//

this->AutoScaleDimensions = System::Drawing::SizeF(6, 13);

this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;

this->ClientSize = System::Drawing::Size(442, 336);

this->Controls->Add(this->mainMenu);

this->Icon = (cli::safe\_cast < System::Drawing::Icon^ > (resources->GetObject(L"$this.Icon")));

this->IsMdiContainer = true;

this->MainMenuStrip = this->mainMenu;

this->Name = L"Form1";

this->Text = L"FWT2D";

this->WindowState = System::Windows::Forms::FormWindowState::Maximized;

this->mainMenu->ResumeLayout(false);

this->mainMenu->PerformLayout();

this->ResumeLayout(false);

this->PerformLayout();

}

#pragma endregion

private:

System::Void openToolStripMenuItem\_Click(System::Object^ sender, System::EventArgs^ e)

{

if (openPicDialog->ShowDialog() == System::Windows::Forms::DialogResult::OK)

{

FChildPic^ mchld = gcnew FChildPic(dir, openPicDialog->FileName);

mchld->MdiParent = this;

mchld->Show();

}

}

};

}

**DlgFltSelect.h**

#pragma once

using namespace System;

using namespace System::ComponentModel;

using namespace System::Collections;

using namespace System::Windows::Forms;

using namespace System::Data;

using namespace System::Drawing;

using namespace System::IO;

namespace Img

{

/// <summary>

/// Summary for DlgFltSelect

///

/// WARNING: If you change the name of this class, you will need to change the

/// 'Resource File Name' property for the managed resource compiler tool

/// associated with all .resx files this class depends on. Otherwise,

/// the designers will not be able to interact properly with localized

/// resources associated with this form.

/// </summary>

public ref class DlgFltSelect : public System::Windows::Forms::Form

{

String^ m\_dir;

public:

DlgFltSelect(void)

{

InitializeComponent();

m\_filter = new wchar\_t[\_MAX\_PATH];

wcscpy(m\_filter, L"filters\\daub1.flt");

}

DlgFltSelect(String^ dir): m\_dir(dir) {

InitializeComponent();

m\_filter = new wchar\_t[\_MAX\_PATH];

swprintf(m\_filter, L"%s\\%s", PtrToStringChars(m\_dir), L"daub1.flt");

}

String^ fltname;

wchar\_t \*m\_filter;

int m\_scales;

int m\_th;

private:

System::Windows::Forms::Button^ okButton;

private:

System::Windows::Forms::Button^ cancelButton;

private:

System::Windows::Forms::TextBox^ scaletextBox;

private:

System::Windows::Forms::TextBox^ thtextBox;

private:

System::Windows::Forms::Label^ label1;

private:

System::Windows::Forms::Label^ label2;

private:

System::Windows::Forms::ListBox^ fltlistBox;

protected:

/// <summary>

/// Clean up any resources being used.

/// </summary>

~DlgFltSelect()

{

if (components)

{

delete components;

}

delete[] m\_filter;

}

private:

/// <summary>

/// Required designer variable.

/// </summary>

System::ComponentModel::Container^ components;

#pragma region Windows Form Designer generated code

/// <summary>

/// Required method for Designer support - do not modify

/// the contents of this method with the code editor.

/// </summary>

void InitializeComponent(void)

{

this->okButton = (gcnew System::Windows::Forms::Button());

this->cancelButton = (gcnew System::Windows::Forms::Button());

this->fltlistBox = (gcnew System::Windows::Forms::ListBox());

this->scaletextBox = (gcnew System::Windows::Forms::TextBox());

this->thtextBox = (gcnew System::Windows::Forms::TextBox());

this->label1 = (gcnew System::Windows::Forms::Label());

this->label2 = (gcnew System::Windows::Forms::Label());

this->SuspendLayout();

//

// okButton

//

this->okButton->DialogResult = System::Windows::Forms::DialogResult::OK;

this->okButton->Location = System::Drawing::Point(184, 12);

this->okButton->Name = L"okButton";

this->okButton->Size = System::Drawing::Size(75, 23);

this->okButton->TabIndex = 3;

this->okButton->Text = L"Ok";

this->okButton->UseVisualStyleBackColor = true;

this->okButton->Click += gcnew System::EventHandler(this, &DlgFltSelect::okButton\_Click);

//

// cancelButton

//

this->cancelButton->DialogResult = System::Windows::Forms::DialogResult::Cancel;

this->cancelButton->Location = System::Drawing::Point(184, 41);

this->cancelButton->Name = L"cancelButton";

this->cancelButton->Size = System::Drawing::Size(75, 23);

this->cancelButton->TabIndex = 4;

this->cancelButton->Text = L"Cancel";

this->cancelButton->UseVisualStyleBackColor = true;

this->cancelButton->Click += gcnew System::EventHandler(this, &DlgFltSelect::cancelButton\_Click);

//

// fltlistBox

//

this->fltlistBox->FormattingEnabled = true;

this->fltlistBox->Location = System::Drawing::Point(12, 12);

this->fltlistBox->Name = L"fltlistBox";

this->fltlistBox->Size = System::Drawing::Size(152, 160);

this->fltlistBox->Sorted = true;

this->fltlistBox->TabIndex = 0;

this->fltlistBox->SelectedIndexChanged += gcnew System::EventHandler(this, &DlgFltSelect::fltlistBox\_SelectedIndexChanged);

//

// scaletextBox

//

this->scaletextBox->Location = System::Drawing::Point(183, 83);

this->scaletextBox->Name = L"scaletextBox";

this->scaletextBox->Size = System::Drawing::Size(38, 20);

this->scaletextBox->TabIndex = 1;

this->scaletextBox->Text = L"3";

this->scaletextBox->TextAlign = System::Windows::Forms::HorizontalAlignment::Right;

//

// thtextBox

//

this->thtextBox->Location = System::Drawing::Point(183, 109);

this->thtextBox->Name = L"thtextBox";

this->thtextBox->Size = System::Drawing::Size(38, 20);

this->thtextBox->TabIndex = 2;

this->thtextBox->Text = L"20";

this->thtextBox->TextAlign = System::Windows::Forms::HorizontalAlignment::Right;

//

// label1

//

this->label1->AutoSize = true;

this->label1->Location = System::Drawing::Point(227, 86);

this->label1->Name = L"label1";

this->label1->Size = System::Drawing::Size(32, 13);

this->label1->TabIndex = 5;

this->label1->Text = L"scale";

//

// label2

//

this->label2->AutoSize = true;

this->label2->Location = System::Drawing::Point(227, 112);

this->label2->Name = L"label2";

this->label2->Size = System::Drawing::Size(22, 13);

this->label2->TabIndex = 6;

this->label2->Text = L"TH";

//

// DlgFltSelect

//

this->AutoScaleDimensions = System::Drawing::SizeF(6, 13);

this->AutoScaleMode = System::Windows::Forms::AutoScaleMode::Font;

this->ClientSize = System::Drawing::Size(278, 194);

this->ControlBox = false;

this->Controls->Add(this->label2);

this->Controls->Add(this->label1);

this->Controls->Add(this->thtextBox);

this->Controls->Add(this->scaletextBox);

this->Controls->Add(this->fltlistBox);

this->Controls->Add(this->cancelButton);

this->Controls->Add(this->okButton);

this->FormBorderStyle= System::Windows::Forms::FormBorderStyle::FixedDialog;

this->MaximizeBox = false;

this->MinimizeBox = false;

this->Name = L"DlgFltSelect";

this->Opacity = 0.85;

this->Text = L"FWT filter select";

this->Shown += gcnew System::EventHandler(this, &DlgFltSelect::DlgFltSelect\_Shown);

this->ResumeLayout(false);

this->PerformLayout();

}

#pragma endregion

////////////////////////////////////////////////////

////////////////////////////////////////////////////

private:

System::Void fltlistBox\_SelectedIndexChanged(System::Object^ sender, System::EventArgs^ e)

{

fltname = fltlistBox->SelectedItem->ToString();

swprintf(m\_filter, L"%s\\%s", PtrToStringChars(m\_dir), PtrToStringChars(fltlistBox->SelectedItem->ToString()));

}

private:

System::Void DlgFltSelect\_Shown(System::Object^ sender, System::EventArgs^ e)

{

DirectoryInfo^ di = gcnew DirectoryInfo(m\_dir);

array < FileInfo^ >^ fi = di->GetFiles("\*.flt");

for (int i = 0; i < fi->Length; i++)

{

FileInfo^ pfi = safe\_cast < FileInfo^ > (fi->GetValue(i));

fltlistBox->Items->Add(pfi->Name);

}

scaletextBox->Text = m\_scales.ToString();

thtextBox->Text = m\_th.ToString();

}

private:

System::Void okButton\_Click(System::Object^ sender, System::EventArgs^ e)

{

m\_scales = Convert::ToInt32(scaletextBox->Text);

m\_th = Convert::ToInt32(thtextBox->Text);

}

private: System::Void cancelButton\_Click(System::Object^ sender, System::EventArgs^ e)

{

}

};

}

###### 12. BIBLIOGRAPHY

**REFERENCE**

1. Canny, J., “A Computational Approach to EdgeDetection”, IEEE Trans.Pattern Analysis and MachineIntelligence, 8:679-714, November 1986
2. Maar, D., Hildreth E., “Theory of edge detection”,Proceedings Royal Soc. London, vol. 207, 187-217, 1980
3. Advanced Edge Detection Technique: Techniques in Computational Vision:http://www.cpsc.ucalgary.ca/Research/vision/501/edgedetect.pdf
4. AN333: Developing Peripherals for SOPC Builder: <http://www.altera.com/literature/an/an333.pdf>
5. Marimont, D.,\ Rubner Y., “A Probabilistic Frameword for Edge Detection and Scale Selection”,Proceedings of the IEEE International Conference on Computer Vision, 207-214, January 1998
6. Lindeberg T., “Edge Detection and Ridge Detectionwith Automatic Scale Selection”, International Journal of Computer Vision, vol. 30, number 2, 117-154, 1998Wufan Chen, Wavelet Analysis and Its Application for Signal Processing, Science Press, Beijing, pp. 175-185, 2002.
7. S Mallat, S Zhong, Characterization of Signals from Multiscale Edges, IEEE Trans Pattern Anal Mach Intell, Vol. 7, No. 14, pp. 710-732, 1992.
8. S Mallat, WL Hwang, Singularity Detection and Processing with the Wavelets, IEEE Trans Inform Theory, Vol. 2, No. 38, pp. 617-643, 1992.
9. FuhuiZou, Zhongke Li, Examination and Analysis of Image Edge Detection Algorithm, Computer Application, Vol. 1, No. 28, pp. 215-219, 2008.
10. Hanjal, Ic A, Shot-boundary Detection: Unraveled and Resolved, IEEE Transactions on Circuits and System for Video Technology, Vol. 2, No. 12, pp. 90-105, 2002.
11. Meer P, GeorGescu B, Edge Detection with Embedded Confidence, IEEE Transactions on Patterm Analysis and Machine Intelligence, Vol. 12, No. 23, pp. 1351-1365, 2001.
12. Li Wan, Ang Yi, Ming Fu, An Improved Edge-Detection Method Based on Canny Algorithm, Computing Technology and Automation, Vol. 22, No. 2, pp. 24-26, Aug. 2003.
13. Ye Yu, Jianhua Lu, JunliZheng, Hermite technique for edge detection in color images, Journal of Tsinghua University, Vol. 48, No. 4, pp. 553-557, 2008.