**CONTENT­**

|  |  |  |
| --- | --- | --- |
| **S.NO** | **TITLE** | **PAGE .NO** |
| 1 | Abstract |  |
| 2 | Introduction |  |
| 3 | Existing system |  |
| 4 | Disadvantages |  |
| 5 | Proposed system |  |
| 6 | Advantages |  |
| 7 | System Architecture |  |
| 8 | Flow Diagram |  |
| 9 | Use case Diagram |  |
| 10 | Class Diagram |  |
| 11 | Sequence Diagram |  |
| 12 | Testing Of Product |  |
| 13 | Modules |  |
| 14 | Modules Description |  |
| 15 | Algorithm Description |  |
| 16 | Literature Survey |  |
| 17 | Hardware Requirements |  |
| 18 | Software Requirements |  |
| 19 | H/W&S/W Description |  |
| 20 | Screen Shots |  |
| 20 | Future Enhancement |  |
| 21 | Conclusion |  |
| 22 | References |  |

**Oil Spill SAR Image Segmentation via Probability Distribution Modelling**

**ABSTRACT:**

* In this work, we aim to develop an effective segmentation method which addresses, marine oil spill identification in SAR images by investigating the distribution of SAR images.
* To seek effective oil spill segmentation, we revisit the SAR imaging mechanism in order to attain the probability distribution of oil spill in SAR images, in which the characteristics of SAR images are properly modelled.
* We then exploit the distribution to formulate the segmentation energy functional, by which oil spill characteristics are incorporated to guide oil spill segmentation.
* There is a necessity for detection and classification, to identify such spill or non-spill, then performance of the process will evaluated with suitable metrics.

**Objective:**

* To check the oil spill or non-spill.
* To improve the efficiency of the process.

**INTRODUCTION**

Adulteration in food products means the addition of prohibited substance either partly or wholly for the state of financial gain or lack of hygienic conditions of processing and storing which leads to the consumer being cheated. Ignorance of this fact is not fair since this may endanger consumer health. For most of the vegetable oil, adulteration detection method is based on conventional chemical tests. Measurement of difference in fatty acid composition and triacylglycerol (TAG) have been utilized for adulteration detection in olive oil. In some cases, the cheap oils used for adulteration have similar composition of TAG and hence the chemical tests may prove unreliable.

The latest commercial iPhone application, Meal Snap, can assist users to record and recognize food images, whereas it still requires the user to manually recognize food types. Automatic food intake assessment that avoids the inaccuracy in manual recording and food estimation deserves more research effort for obesity management. Object recognition has been one of the fundamental areas in pattern recognition for decades, producing prosperous results in specific object recognition, such as faces and cars. Food recognition is challenging as compared to specific object recognition because it is essentially an intra-class recognition problem. Intra-class recognition is still unsolved, especially for objects with extreme variation, such as animals, furniture, flowers, and food. The appearance of food exhibits a higher degree of variance even for the same food type. General object recognition methods have been applied to food recognition. These techniques include colour histogram, texture and bag-of-feature classification. In our previous paper, we found these existing techniques could have acceptable results for regular-shaped food recognition and fast food recognition; however, the recognition performance on generally arbitrary food was not as promising. In fact, colour histogram-based classification is sensitive to lighting conditions, and the bag-of-feature method extracts the statistics of key image patches, which do not explicitly represent the necessary location information in food. Novel techniques requires to be invented for general arbitrary food recognition. Given a sample of food, one is able to recognize its ingredients from shapes, textures and colors. With the combination of the ingredients, one could recognize the food based on the distribution of ingredients. Sometimes, persons may also have difficulties to recognize food if they do not have sufficient knowledge to distinguish food only through their appearances. It is challenging to automate food recognition in a similar way that humans recognize food from ingredients. Different food types could have similar appearances, while the same food type could have different appearances. In addition, recipes, cooking methods, and chef’s personal preference affect the appearances of food ingredients. The second challenge is that even though ingredients could be detected correctly, food could also have unstructured ingredient distribution. For some types of food, the ingredients are distributed randomly across a plate. The third important challenge is the occlusion in food images. Food is usually placed in certain containers such that some key elements may be covered or occluded by other ingredients. Food recognition from images in different scales is another challenge. Some types of food could not be differentiated through its own sizes in images. For example, brown rice looks similar to a baked potato without considering their relative scale. This paper presents an ingredient-based food recognition method. Our first innovation is the improvement of the current part-based object recognition model toward texture-oriented and location-flexible to detect food ingredients. The state-of-the-art part-based detectors are not directly suitable for food ingredient detection concerning the aforementioned challenges. We modify the detector in three ways for the purpose of food ingredient detection. The ingredient detector tries to find food ingredients on a single scale in order to retain relative ingredient scales. After that, the scale invariance is achieved in a multi-scale support vector machine (SVM) during food classification. The part-based model uses the shape of objects as a key property. We enrich the model with the ability to verify detection with texture models, meaning that both the shape model and texture model are used to detect food ingredients. In the existing part-based based model, the geometry locations of the parts are modeled strictly. In our model, we employ a more flexible location mechanism for food ingredients to represent food with more deformation. Our second contribution is development of a multi-view multi-kernel SVM to classify various combinations of food ingredients under occlusion. We design the SVM with multiple kernels that include a hierarchy of element kernels. The top level is the viewpoint level where we adopt a multi-view scheme to address occlusion. On the viewpoint level, each view corresponds to a kernel function, and all the kernel functions from multi-view are combined together according to the geometry similarities between viewpoints. Under each viewpoint kernel, we design spatial pyramid kernels to achieve scale invariance. Then under each scale, there is a linear combination of linear, quasi-linear, and nonlinear element kernels to classify food ingredient features. By employing such a hierarchy of kernel functions, we accomplish a classifier that detects and classifies food of different scales and points of view. The experiments show the effectiveness of the proposed classifier.

Coconut oil is abundant in the southern parts of India and so is extensively used for edible purposes. Coconut oil is rich in Medium Chain Triglycerides (MCT) and Medium Chain Fatty Acids (MCFA) that are burnt immediately to produce energy and are not converted as triglycerides thereby accounting for increasing high-density lipoprotein (HDL) and lowering low-density lipoprotein (LDL). Coconut can be extracted through ‘dry process’ or ‘wet process’. The Virgin Coconut Oil (VCO) is extracted through wet processing. It involves no chemical treatment or heat treatment. The in vitro study by Nevin and Rajamohan showed that VCO was capable of reducing low density lipoproteins oxidation. In view of its nutritional values and demand, coconut oil is expensive and high possibility of adulterating with less expensive oils. Therefore, we have used VCO for this study to develop a cost effective, reliable method to detect adulteration, which can be extended for other edible oils with some modifications. When VCO is adulterated, its physical properties such as density, viscosity changes and has direct impact on ultrasonic velocity, reflection coefficient and attenuation when a wave propagate through it. Having this as the base, this paper proposes to develop handheld direct ultrasonic detection system which utilises the attenuation coefficient, reflection coefficient and velocity of propagation of ultrasound in the oil medium incorporating ANN based algorithm to reduce the percentage of error in the prediction.

**DOMAIN EXPLANATION**

**IMAGE PROCESSING**

* 1. **What is an image?**

An image is an array, or a matrix, of square pixels (picture elements) arranged in columns and rows.

**Figure 1: An image — an array or a matrix of pixels arranged in columns and rows.**

In a (8-bit) greyscale image each picture element has an assigned intensity that ranges from 0 to 255. A grey scale image is what people normally call a black and white image, but the name emphasizes that such an image will also include many shades of grey.

**Figure 2: Each pixel has a value from 0 (black) to 255 (white). The possible range of the pixel values depend on the colour depth of the image, here 8 bit = 256 tones or greyscales.**

A normal grayscale image has 8 bit colour depth = 256 grayscales. A “true colour” image has 24 bit colour depth = 8 x 8 x 8 bits = 256 x 256 x 256 colours = ~16 million colours.

**Figure 3: A true-colour image assembled from three grayscale images coloured red, green and blue. Such an image may contain up to 16 million different colours.**

Some grayscale images have more grayscales, for instance 16 bit = 65536 grayscales. In principle three grayscale images can be combined to form an image with 281,474,976,710,656 grayscales.

There are two general groups of ‘images’: vector graphics (or line art) and bitmaps (pixel-based or ‘images’). Some of the most common file formats are:

GIF — an 8-bit (256 colour), non-destructively compressed bitmap format. Mostly used for web. Has several sub-standards one of which is the animated GIF.

JPEG — a very efficient (i.e. much information per byte) destructively compressed 24 bit (16 million colours) bitmap format. Widely used, especially for web and Internet (bandwidth-limited).

TIFF — the standard 24 bit publication bitmap format. Compresses non-destructively with, for instance, Lempel-Ziv-Welch (LZW) compression.

PS — Postscript, a standard vector format. Has numerous sub-standards and can be difficult to transport across platforms and operating systems.

PSD – a dedicated Photoshop format that keeps all the information in an image including all the layers.

Pictures are the most common and convenient means of conveying or transmitting information. A picture is worth a thousand words. Pictures concisely convey information about positions, sizes and inter relationships between objects. They portray spatial information that we can recognize as objects. Human beings are good at deriving information from such images, because of our innate visual and mental abilities. About 75% of the information received by human is in pictorial form. An image is digitized to convert it to a form which can be stored in a computer's memory or on some form of storage media such as a hard disk or CD-ROM. This digitization procedure can be done by a scanner, or by a video camera connected to a frame grabber board in a computer. Once the image has been digitized, it can be operated upon by various image processing operations.

Image processing operations can be roughly divided into three major categories, Image Compression, Image Enhancement and Restoration, and Measurement Extraction. It involves reducing the amount of memory needed to store a digital image. Image defects which could be caused by the digitization process or by faults in the imaging set-up (for example, bad lighting) can be corrected using Image Enhancement techniques. Once the image is in good condition, the Measurement Extraction operations can be used to obtain useful information from the image.

Some examples of Image Enhancement and Measurement Extraction are given below. The examples shown all operate on 256 grey-scale images. This means that each pixel in the image is stored as a number between 0 to 255, where 0 represents a black pixel, 255 represents a white pixel and values in-between represent shades of grey. These operations can be extended to operate on colour images. The examples below represent only a few of the many techniques available for operating on images. Details about the inner workings of the operations have not been given, but some references to books containing this information are given at the end for the interested reader.

**Images and pictures**

As we mentioned in the preface, human beings are predominantly visual creatures: we rely heavily on our vision to make sense of the world around us. We not only look at things to identify and classify them, but we can scan for differences, and obtain an overall rough feeling for a scene with a quick glance. Humans have evolved very precise visual skills: we can identify a face in an instant; we can differentiate colors; we can process a large amount of visual information very quickly.

However, the world is in constant motion: stare at something for long enough and it will change in some way. Even a large solid structure, like a building or a mountain, will change its appearance depending on the time of day (day or night); amount of sunlight (clear or cloudy), or various shadows falling upon it. We are concerned with single images: snapshots, if you like, of a visual scene. Although image processing can deal with changing scenes, we shall not discuss it in any detail in this text. For our purposes, an image is a single picture which represents something. It may be a picture of a person, of people or animals, or of an outdoor scene, or a microphotograph of an electronic component, or the result of medical imaging. Even if the picture is not immediately recognizable, it will not be just a random blur.

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

1. Improve its pictorial information for human interpretation,

2. Render it more suitable for autonomous machine perception.

We shall be concerned with digital image processing, which involves using a computer to change the nature of a digital image. It is necessary to realize that these two aspects represent two separate but equally important aspects of image processing. A procedure which satisfies condition, a procedure which makes an image look better may be the very worst procedure for satisfying condition. Humans like their images to be sharp, clear and detailed; machines prefer their images to be simple and uncluttered.

**Images and digital images**

Suppose we take an image, a photo, say. For the moment, lets make things easy and suppose the photo is black and white (that is, lots of shades of grey), so no colour. We may consider this image as being a two dimensional function, where the function values give the brightness of the image at any given point. We may assume that in such an image brightness values can be any real numbers in the range (black) (white).

A digital image from a photo in that the values are all discrete. Usually they take on only integer values. The brightness values also ranging from 0 (black) to 255 (white). A digital image can be considered as a large array of discrete dots, each of which has a brightness associated with it. These dots are called picture elements, or more simply pixels. The pixels surrounding a given pixel constitute its neighborhood. A neighborhood can be characterized by its shape in the same way as a matrix: we can speak of a neighborhood,. Except in very special circumstances, neighborhoods have odd numbers of rows and columns; this ensures that the current pixel is in the centre of the neighborhood.

**Image Processing Fundamentals:**

**Pixel:**

In order for any digital computer processing to be carried out on an image, it must first be stored within the computer in a suitable form that can be manipulated by a computer program. The most practical way of doing this is to divide the image up into a collection of discrete (and usually small) cells, which are known as pixels. Most commonly, the image is divided up into a rectangular grid of pixels, so that each pixel is itself a small rectangle. Once this has been done, each pixel is given a pixel value that represents the color of that pixel.

It is assumed that the whole pixel is the same color, and so any color variation that did exist within the area of the pixel before the image was discretized is lost. However, if the area of each pixel is very small, then the discrete nature of the image is often not visible to the human eye.

Other pixel shapes and formations can be used, most notably the hexagonal grid, in which each pixel is a small hexagon. This has some advantages in image processing, including the fact that pixel connectivity is less ambiguously defined than with a square grid, but hexagonal grids are not widely used. Part of the reason is that many image capture systems (e.g. most CCD cameras and scanners) intrinsically discretize the captured image into a rectangular grid in the first instance.

**Pixel Connectivity**

The notation of pixel connectivity describes a relation between two or more pixels. For two pixels to be connected they have to fulfill certain conditions on the pixel brightness and spatial adjacency.

First, in order for two pixels to be considered connected, their pixel values must both be from the same set of values V. For a grayscale image, V might be any range of graylevels, e.g. V={22,23,...40}, for a binary image we simple have V={1}.

To formulate the adjacency criterion for connectivity, we first introduce the notation of neighborhood. For a pixel p with the coordinates (x,y) the set of pixels given by:

Eqn:eqnla1

is called its 4-neighbors. Its 8-neighbors are defined as

Eqn:eqnla2

From this we can infer the definition for 4- and 8-connectivity:

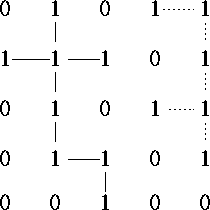
Two pixels p and q, both having values from a set V are 4-connected if q is from the set Eqn:eqnlan4and 8-connected if q is from Eqn:eqnlan8.

General connectivity can either be based on 4- or 8-connectivity; for the following discussion we use 4-connectivity.

A pixel p is connected to a pixel q if p is 4-connected to q or if p is 4-connected to a third pixel which itself is connected to q. Or, in other words, two pixels q and p are connected if there is a path from p and q on which each pixel is 4-connected to the next one.

A set of pixels in an image which are all connected to each other is called a connected component. Finding all connected components in an image and marking each of them with a distinctive label is called connected component labeling.

An example of a binary image with two connected components which are based on 4-connectivity can be seen in Figure 1. If the connectivity were based on 8-neighbors, the two connected components would merge into one.



**Figure 1** Two connected components based on 4-connectivity.

# Pixel Values

Each of the pixels that represents an image stored inside a computer has a pixel value which describes how bright that pixel is, and/or what color it should be. In the simplest case of binary images, the pixel value is a 1-bit number indicating either foreground or background. For a gray scale images, the pixel value is a single number that represents the brightness of the pixel. The most common motepixel format is the byte image, where this number is stored as an 8-bit integer giving a range of possible values from 0 to 255. Typically zero is taken to be black, and 255 is taken to be white. Values in between make up the different shades of gray.

To represent colour images, separate red, green and blue components must be specified for each pixel (assuming an RGB colour space), and so the pixel `value' is actually a vector of three numbers. Often the three different components are stored as three separate `grayscale' images known as color planes (one for each of red, green and blue), which have to be recombined when displaying or processing. Multispectral Images can contain even more than three components for each pixel, and by extension these are stored in the same kind of way, as a vector pixel value, or as separate color planes.

The actual grayscale or color component intensities for each pixel may not actually be stored explicitly. Often, all that is stored for each pixel is an index into a colour map in which the actual intensity or colors can be looked up. Although simple 8-bit integers or vectors of 8-bit integers are the most common sorts of pixel values used, some image formats support different types of value, for instance 32-bit signed integers or floating point values. Such values are extremely useful in image processing as they allow processing to be carried out on the image where the resulting pixel values are not necessarily 8-bit integers. If this approach is used then it is usually necessary to set up a colormap which relates particular ranges of pixel values to particular displayed colors.

**Pixels, with a neighborhood:**

****

**Color scale**

The two main color spaces are **RGB** and **CMYK.**

**RGB**

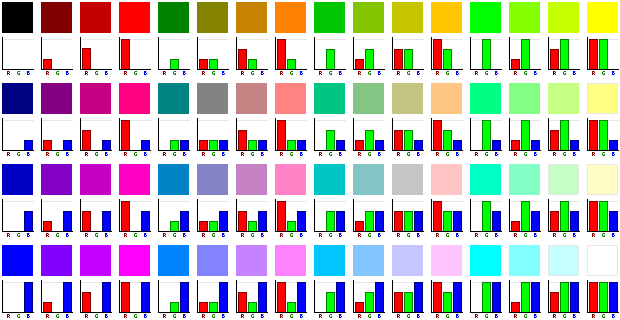
The **RGB color model** is an additive color model in which red, green, and blue light are added together in various ways to reproduce a broad array of colors. RGB uses additive color mixing and is the basic color model used in television or any other medium that projects color with light. It is the basic color model used in computers and for web graphics, but it cannot be used for print production.

The secondary colors of RGB – cyan, magenta, and yellow – are formed by mixing two of the primary colors (**red, green** or **blue**) and excluding the third color. Red and green combine to make yellow, green and blue to make cyan, and blue and red form magenta. The combination of red, green, and blue in full intensity makes white.



The additive model of RGB. Red, green, and blue are the primary stimuli for human color perception and are the primary additive colours.

To see how different RGB components combine together, here is a selected repertoire of colors and their respective relative intensities for each of the red, green, and blue components:

[](http://en.wikipedia.org/wiki/Image:RGB-combinations.png)

**\*Typical uses of MATLAB include:-**

- Math and computation.

-Algorithm development

-Data acquisition

-Modeling, simulation, and prototyping

-Data analysis, exploration, and visualization

-Scientific and engineering graphics

-Application development, including graphical user interface building

**Some applications:**

Image processing has an enormous range of applications; almost every area of science and technology can make use of image processing methods. Here is a short list just to give some indication of the range of image processing applications.

1. Medicine

* Inspection and interpretation of images obtained from X-rays, MRI or CAT scans,
* Analysis of cell images, of chromosome karyotypes.

2. Agriculture

* Satellite/aerial views of land, for example to determine how much land is being used for different purposes, or to investigate the suitability of different regions for different crops,
* Inspection of fruit and vegetables distinguishing good and fresh produce from old.

3. Industry

* Automatic inspection of items on a production line,
* Inspection of paper samples.

4. Law enforcement

* Fingerprint analysis,
* Sharpening or de-blurring of speed-camera images.

**Aspects of image processing:**

It is convenient to subdivide different image processing algorithms into broad subclasses. There are different algorithms for different tasks and problems, and often we would like to distinguish the nature of the task at hand.

* **Image enhancement**: This refers to processing an image so that the result is more suitable for a particular application.

Example include:

* sharpening or de-blurring an out of focus image,
* highlighting edges,
* improving image contrast, or brightening an image,
* Removing noise.
* **Image restoration**. This may be considered as reversing the damage done to an image by a known cause, for example:
* removing of blur caused by linear motion,
* removal of optical distortions,
* Removing periodic interference.
* **Image segmentation**. This involves subdividing an image into constituent parts, or isolating certain aspects of an image:
* circles, or particular shapes in an image,
* In an aerial photograph, identifying cars, trees, buildings, or roads.

These classes are not disjoint; a given algorithm may be used for both image enhancement or for image restoration. However, we should be able to decide what it is that we are trying to do with our image: simply make it look better (enhancement), or removing damage (restoration).

**An image processing task**

We will look in some detail at a particular real-world task, and see how the above classes may be used to describe the various stages in performing this task. The job is to obtain, by an automatic process, the postcodes from envelopes. Here is how this may be accomplished:

* **Acquiring the image**: First we need to produce a digital image from a paper envelope. This can be done using either a CCD camera, or a scanner.
* **Preprocessing:** This is the step taken before the major image processing task. The problem here is to perform some basic tasks in order to render the resulting image more suitable for the job to follow. In this case it may involve enhancing the contrast, removing noise, or identifying regions likely to contain the postcode.
* **Segmentation**: Here is where we actually get the postcode; in other words we extract from the image that part of it which contains just the postcode.
* **Representation and description** These terms refer to extracting the particular features which allow us to differentiate between objects. Here we will be looking for curves, holes and corners which allow us to distinguish the different digits which constitute a postcode.
* **Recognition and interpretation**: This means assigning labels to objects based on their descriptors (from the previous step), and assigning meanings to those labels. So we identify particular digits, and we interpret a string of four digits at the end of the address as the postcode.

**EXISTING SYSTEM**

Today, adulterations are more sophisticated. Therefore, it is necessary to use advanced and suitable methods to detect adulteration. Generally, using physical properties like refractive index, viscosity, melting point, saponification and iodine value are not anymore practical to detect adulteration. Hence in this process, an Application of computer vision system in food processing. Computer vision have proven very successful in the analysis of food on the basis of color, size, shape, texture etc. It is the science that developed algorithm basis by which useful information automatically extracted from observed image. A low-cost visual-based color classification system is presented. This system consists of a color webcam interfaced to the computer. The entire system operates on a Matlab driven program that analyzes the color of the objects being inspected. The main idea of this study is to understand the use of color in machine vision and to enhance the color processing in PC-based vision systems. The theory of operation for this system includes the process of dithering, calculation of the mean values and histograms, and the final result. Experimental results obtained from different fruits are analyzed and tabulated. One of the potential applications is to segregate the ripe and unripe fruits. This system is fundamentally capable of identifying four types of tropical fruits.

**Disadvantages:**

* Due to the absence of multiple linear regression, the accuracy performance of the process is low.
* Adulteration types was identified based on the thresholding, not the appropriate labels.

**PROPOSED METHOD**

* This process is going to implement based on the simulation in digital image processing in MATLAB.
* The efficient algorithm will be implemented in this process to identify the spill of the oil, hence the user can segment the region of spill by SAR image.
* Then the proposed method is sub-divided into two categories, which is identification and classification. During the identification, it shows the spill portion.
* Hence the classification, proposed to estimate the spill or non-spill image.

**Advantages:**

* The classification model is used, to improve the accuracy performance of the process.
* Spill or non-spill types was identified based on the appropriate labels.
* It has more reliable, because the quality of the image will be optimum if the process is run for several times.

**TESTING OF PRODUCT**

### 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.

**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.

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 requirement. 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.

**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.

# 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.

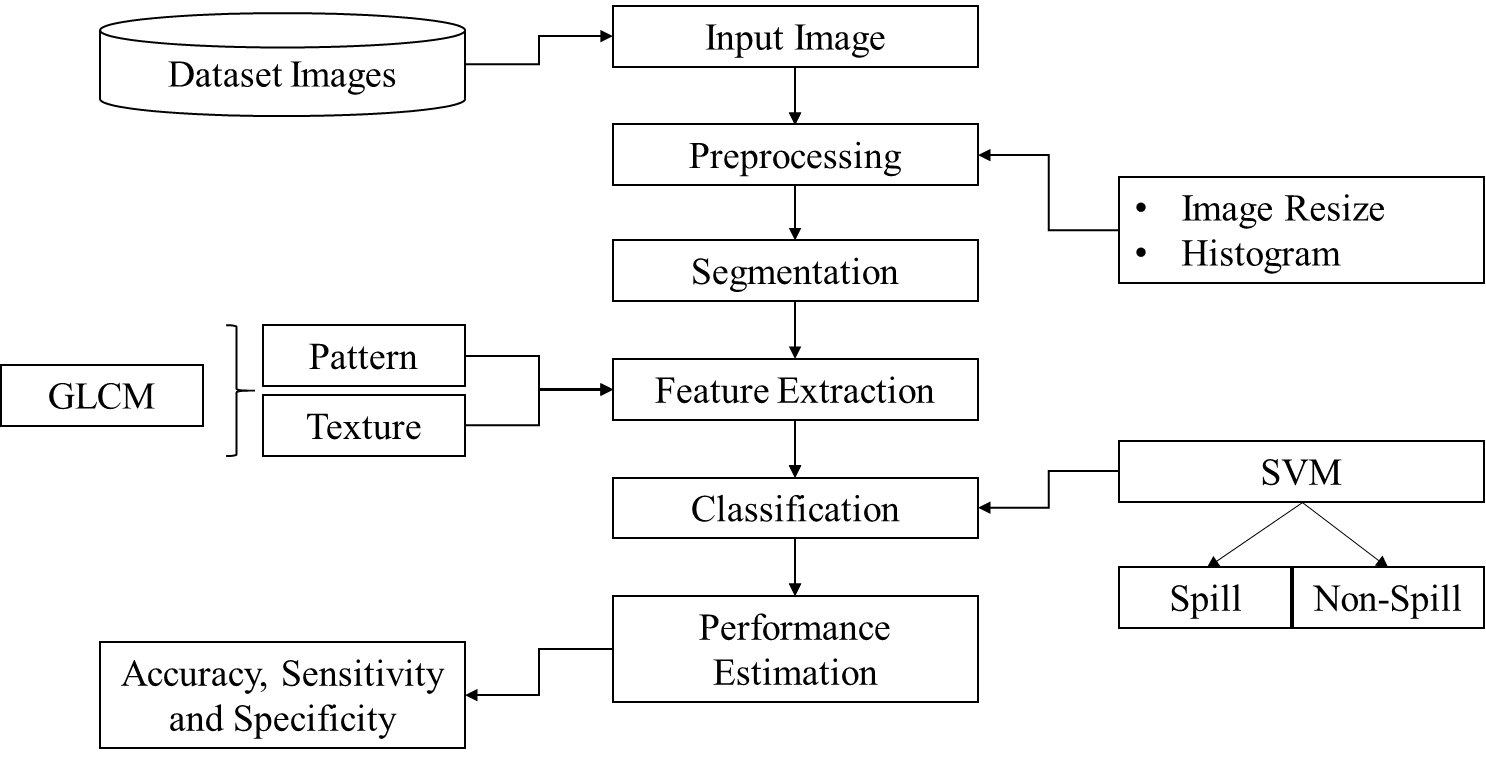
**Acceptance Testing**

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.

**SYSTEM ARCHITECTURE**

**FLOW DIAGRAM**

****

**MODULES**

* Input Image
* Preprocessing
* Feature Extraction
* Classification
* Performance Estimation

**MODULES DESCRIPTION**

**INPUTIMAGE:**

Read an image into the workspace, using the imread command. The example reads one of the sample images included with the toolbox, an image, and stores it in an array named I .The imread infers from the file that the graphics file format is Tagged Image File Format (TIFF).Display the image, using the imshow function. You can also view an image in the Image Viewer app. The imtool function opens the Image Viewer app which presents an integrated environment for displaying images and performing some common image processing tasks. The Image Viewer app provides all the image display capabilities of imshow but also provides access to several other tools for navigating and exploring images, such as scroll bars, the Pixel Region tool, Image Information tool, and the Contrast the adjustment tool.

**PREPROCESSING:**

***Image Resize:***

* In computer graphics and digital imaging, **image** **scaling** refers to the resizing of a digital image. In video technology, the magnification of digital material is known as upscaling or resolution enhancement.
* When scaling a vector graphic image, the graphic primitives that make up the image can be scaled using geometric transformations, with no loss of image quality. When scaling a raster graphics image, a new image with a higher or lower number of pixels must be generated.
* In the case of decreasing the pixel number (scaling down) this usually results in a visible quality loss.
* From the standpoint of digital signal processing, the scaling of raster graphics is a two-dimensional example of sample rate conversion, the conversion of a discrete signal from a sampling rate (in this case the local sampling rate) to another.

***Noise Filtering:***

* Image processing is basically the use of computer algorithms to perform image processing on digital images.
* Digital image processing is a part of digital signal processing.
* Digital image processing has many significant advantages over analog image processing.
* Image processing allows a much wider range of algorithms to be applied to the input data and can avoid problems such as the build-up of noise and signal distortion during processing of images.
* Wavelet transforms have become a very powerful tool for de-noising an image.

***Gaussian:***

In pre-processing we are applying Gaussian filtering to our input image. Gaussian filtering is often used to remove the noise from the image. Here we used wiener function to our input image. **Gaussian filter** is windowed filter of linear class, by its nature is weighted mean. Named after famous scientist Carl Gauss because weights in the filter calculated according to Gaussian distribution.

The Gaussian Smoothing Operator performs a weighted average of surrounding pixels based on the Gaussian distribution. It is used to remove Gaussian noise and is a realistic model of defocused lens. Sigma defines the amount of blurring. The radius slider is used to control how large the template is. Large values for sigma will only give large blurring for larger template sizes. Noise can be added using the sliders.

**Gaussian filter algorithm:**

1. Given window size 2*N*+1 calculate support points *xn*=3*n*/*N*, *n*=-*N*, -*N*+1, ... , *N*;
2. Calculate values *G"n*;
3. Calculate scale factor *k'*=∑*G"n*;
4. Calculate window weights *G'n*=*G"n*/*k'*;
5. For every signal element:
   1. Place window over it;
   2. Pick up elements;
   3. Multiply elements by corresponding window weights;
   4. Sum up products — this sum is new filtered value.

**Feature Extraction:**

***LBP:***

* Local binary patterns (LBP) is a type of visual descriptor used for classification in computer vision.
* LBP is the particular case of the Texture Spectrum model proposed in 1990.
* It has since been found to be a powerful feature for texture classification; it has further been determined that when LBP is combined with the Histogram of oriented gradients (HOG) descriptor, it improves the detection performance considerably on some datasets.

***The LBP feature vector, in its simplest form, is created in the following manner:***

* Divide the examined window into cells (e.g. 16x16 pixels for each cell).
* For each pixel in a cell, compare the pixel to each of its [8 neighbors](https://en.wikipedia.org/wiki/Pixel_connectivity" \o "Pixel connectivity) (on its left-top, left-middle, left-bottom, right-top, etc.). Follow the pixels along a circle, i.e. clockwise or counter-clockwise.
* Where the center pixel's value is greater than the neighbor's value, write "0". Otherwise, write "1". This gives an 8-digit binary number (which is usually converted to decimal for convenience).
* Compute the [histogram](https://en.wikipedia.org/wiki/Histogram" \o "Histogram), over the cell, of the frequency of each "number" occurring (i.e., each combination of which pixels are smaller and which are greater than the center). This histogram can be seen as a 256-dimensional [feature vector](https://en.wikipedia.org/wiki/Feature_vector" \o "Feature vector).
* Optionally normalize the histogram.
* Concatenate (normalized) histograms of all cells. This gives a feature vector for the entire window.

The feature vector can now be processed using the [Support vector machine](https://en.wikipedia.org/wiki/Support_vector_machine), [extreme learning machines](https://en.wikipedia.org/wiki/Extreme_learning_machine" \o "Extreme learning machine), or some other [machine learning](https://en.wikipedia.org/wiki/Machine_learning" \o "Machine learning) algorithm to classify images. Such classifiers can be used for [face recognition](https://en.wikipedia.org/wiki/Facial_recognition_system" \o "Facial recognition system) or texture analysis.

**KNN CLASSIFIERS:**

In [statistics](https://en.wikipedia.org/wiki/Statistics), the ***k*-nearest neighbors algorithm** (***k*-NN**) is a [non-parametric](https://en.wikipedia.org/wiki/Non-parametric_statistics" \o "Non-parametric statistics) [supervised learning](https://en.wikipedia.org/wiki/Supervised_learning" \o "Supervised learning) method first developed by [Evelyn Fix](https://en.wikipedia.org/wiki/Evelyn_Fix" \o "Evelyn Fix) and [Joseph Hodges](https://en.wikipedia.org/wiki/Joseph_Lawson_Hodges_Jr." \o "Joseph Lawson Hodges Jr.) in 1951,[[1]](https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm" \l "cite_note-1) and later expanded by [Thomas Cover](https://en.wikipedia.org/wiki/Thomas_M._Cover" \o "Thomas M. Cover).[[2]](https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm" \l "cite_note-2) It is used for [classification](https://en.wikipedia.org/wiki/Statistical_classification" \o "Statistical classification) and [regression](https://en.wikipedia.org/wiki/Regression_analysis" \o "Regression analysis). In both cases, the input consists of the *k* closest training examples in a [data set](https://en.wikipedia.org/wiki/Data_set" \o "Data set). The output depends on whether *k*-NN is used for classification or regression:

* In *k-NN classification*, the output is a class membership. An object is classified by a plurality vote of its neighbors, with the object being assigned to the class most common among its *k* nearest neighbors (*k* is a positive [integer](https://en.wikipedia.org/wiki/Integer" \o "Integer), typically small). If *k* = 1, then the object is simply assigned to the class of that single nearest neighbor.
* In *k-NN regression*, the output is the property value for the object. This value is the average of the values of *k* nearest neighbors.

*k*-NN is a type of [classification](https://en.wikipedia.org/wiki/Classification" \o "Classification) where the function is only approximated locally and all computation is deferred until function evaluation. Since this algorithm relies on distance for classification, if the features represent different physical units or come in vastly different scales then [normalizing](https://en.wikipedia.org/wiki/Normalization_(statistics)" \o "Normalization (statistics)) the training data can improve its accuracy dramatically.[[3]](https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm" \l "cite_note-:0-3)[[4]](https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm" \l "cite_note-4)

Both for classification and regression, a useful technique can be to assign weights to the contributions of the neighbors, so that the nearer neighbors contribute more to the average than the more distant ones. For example, a common weighting scheme consists in giving each neighbor a weight of 1/*d*, where *d* is the distance to the neighbor.[[5]](https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm" \l "cite_note-5)

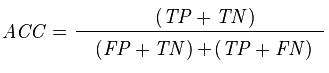
The neighbors are taken from a set of objects for which the class (for *k*-NN classification) or the object property value (for *k*-NN regression) is known. This can be thought of as the training set for the algorithm, though no explicit training step is required.

A peculiarity of the *k*-NN algorithm is that it is sensitive to the local structure of the data.

**PERFORMANCE ANALYSIS:**

The accuracy, sensitivity and specificity of the classifier is measured. The accuracy represents the efficiency of the process. The sensitivity shows how the algorithm gives correct classification. The specificity shows how the algorithm rejects the wrongly classification results. We designed a spatial consistency constraint in a graphical model to improve the detection performance. Our lesion characterization method is based on the multi-atlas approach. We have improved the appearance constraint for better structure estimation and lower method complexity without the additional structure delineation step.

The performance of the process is measured based on the calculation of Accuracy, Area under curve of the process.



The performance of the process is measured in terms of performance metrics like Precision, Recall, F-measure and false positives. The







1. TP is the total number of correctly classified foreground (true positives).
2. FN is the total number of false negatives, which accounts for the incorrect number of disease type pixels classified as dataset (false negatives).
3. FP is the total number of false positives, which means the pixels are incorrectly classified as images (false positives).
4. True negative = correctly rejected

Sensitivity and specificity are statistical measures of the performance of a binary classification test, also known in statistics as a classification function:

Sensitivity (also called the true positive rate, the recall, or probability of detection in some fields) measures the proportion of actual positives that are correctly identified as such (e.g., the percentage of sick people who are correctly identified as having the condition).

Specificity (also called the true negative rate) measures the proportion of actual negatives that are correctly identified as such (e.g., the percentage of healthy people who are correctly identified as not having the condition).

Equivalently, in medical tests sensitivity is the extent to which actual positives are not overlooked (so false negatives are few), and specificity is the extent to which actual negatives are classified as such (so false positives are few). Thus a highly sensitive test rarely overlooks an actual positive (for example, showing "nothing bad" despite something bad existing); a highly specific test rarely registers a positive classification for anything that is not the target of testing (for example, finding one bacterial species and mistaking it for another closely related one that is the true target); and a test that is highly sensitive and highly specific does both, so it "rarely overlooks a thing that it is looking for" and it "rarely mistakes anything else for that thing." Because most medical tests do not have sensitivity and specificity values above 99%, "rarely" does not equate to certainty. But for practical reasons, tests with sensitivity and specificity values above 90% have high credibility, albeit usually no certainty, in differential diagnosis.

Sensitivity therefore quantifies the avoiding of false negatives and specificity does the same for false positives. For any test, there is usually a trade-off between the measures – for instance, in airport security, since testing of passengers is for potential threats to safety, scanners may be set to trigger alarms on low-risk items like belt buckles and keys (low specificity) in order to increase the probability of identifying dangerous objects and minimize the risk of missing objects that do pose a threat (high sensitivity). This trade-off can be represented graphically using a receiver operating characteristic curve. A perfect predictor would be described as 100% sensitive, meaning all sick individuals are correctly identified as sick, and 100% specific, meaning no healthy individuals are incorrectly identified as sick. In reality, however, any non-deterministic predictor will possess a minimum error bound known as the Bayes error rate.

**LITERATURE SURVEY**

**1. Title: Personalized Classifier for Food Image Recognition**

**Author: Shota Horiguchi , Member, IEEE, Sosuke Amano, Makoto Ogawa, and Kiyoharu Aizawa , Fellow, IEEE**

Currently, food image recognition tasks are evaluated against fixed datasets. However, in real-world conditions, there are cases in which the number of samples in each class continues to increase and samples from novel classes appear. In particular, dynamic datasets in which each individual user creates samples and continues the updating process often has content that varies considerably between different users, and the number of samples per person is very limited. A single classifier common to all users cannot handle such dynamic data. Bridging the gap between the laboratory environment and the real world has not yet been accomplished on a large scale. Personalizing a classifier incrementally for each user is a promising way to do this. In this paper, we address the personalization problem, which involves adapting to the user’s domain incrementally using a very limited number of samples. We propose a simple yet effective personalization framework, which is a combination of the nearest class mean classifier and the 1-nearest neighbor classifier based on deep features. To conduct realistic experiments, we made use of a new dataset of daily food images collected by a food-logging application. Experimental results show that our proposed method significantly outperforms existing methods.

In this paper, we deal with such real-world problems by personalizing classifiers sequentially. Incremental learning, in which additional samples of existing classes or novel classes are learned without retraining, is necessary for obtaining visual knowledge from sequential data. Conventional methods learn metrics to classify existing classes using a large number of samples at first, and then learn novel classes incrementally. However, most of them retain constraint assumptions. One is that the number of samples of the initial classes does not increase. In actual situations, the domains of the initial classes may change. Another is that newly trained classes have nearly the same number of samples as initially prepared classes. In the real world, novel classes have a much smaller number of samples than initial classes. Therefore, existing incremental learning techniques are insufficient for personalization.

In addition to incremental learning, personalization of classifiers should also fulfill two other aspects. One is domain adaptation. For each person, the domain, that is, the data comprised in each class, is different. The class definitions are different; thus, it is important to assume that each personal data item is from a different domain. The other is one-shot learning. When focusing on a given person, the number of images is limited; thus, it is also an important factor in personalization that new classes can be learned by using only one or a few samples. Food image classification is one practical use case of image recognition technology. There are many studies in food image recognition. Almost all of the previous studies followed the general method of using fixed food image datasets. However, such a controlled scenario is not appropriate for a real-world purpose. We have been running a food-logging application for smartphones and have built a food image dataset named FLD consisting of over 1.5 million images. They are food images uploaded daily by the users, and as a result, the number of classes grew incrementally. Analysis of FLD showed that the appearance of food images differs considerably between people.

Our contributions are summarized as follows:

We propose a simple yet powerful personalization framework called the sequential personalized classifier (SPC).

This attains performance comparable to that of the CNN classifier at first and becomes more accurate after sequential personalization without any heavy retraining.

We evaluated SPC’s real-world performance on a new dataset of daily food images we built to reproduce real situations, rather than artificially designed simulations of real-world data.

We conducted exhaustive comparisons against previous methods and found that the proposed method significantly outperforms existing techniques.

**Advantages:**

The sequential personalized classifier (SPC) is used to achieve rotation invariance, and considers rotation and scale invariance.

**Disadvantages:**

It is, therefore, crucial to develop texture features that are not only discriminative across many classes, but also invariant to key transformations.

**2. Title: A Food Recognition System for Diabetic Patients Based on an Optimized Bag-of-Features Model**

**Author:** Marios M. Anthimopoulos, Member, IEEE, Lauro Gianola, Luca Scarnato, Peter Diem, and Stavroula G. Mougiakakou, Member, IEEE

Computer vision-based food recognition could be used to estimate a meal’s carbohydrate content for diabetic patients. This study proposes a methodology for automatic food recognition, based on the bag-of-features (BoF) model. An extensive technical investigation was conducted for the identification and optimization of the best performing components involved in the BoF architecture, as well as the estimation of the corresponding parameters. For the design and evaluation of the prototype system, a visual dataset with nearly 5000 food images was created and organized into 11 classes. The optimized system computes dense local features, using the scale-invariant feature transform on the HSV color space, builds a visual dictionary of 10000 visual words by using the hierarchical k-means clustering and finally classifies the food images with a linear support vector machine classifier. The system achieved classification accuracy of the order of 78%, thus proving the feasibility of the proposed approach in a very challenging image dataset.

A food recognition application was introduced by Shroff for the classification of fast-food images into four classes. For each segmented food item, a vector of color (normalized RGB values), size, texture (local entropy, standard deviation, range), shape, and context-based features is computed and fed to a feed-forward artificial neural network (ANN), resulting in recognition accuracy of the order of 95%, 80%, 90%, and 90% for hamburgers, fries, chicken nuggets, and apple pies, respectively. A set of color (pixel intensities and color components) and texture (Gabor filter responses) features was used, together with a support vector machine (SVM) classifier, for the recognition of 19 food classes, leading to a recognition rate of the order of 94% for food replicas and 58% for real food items. proposed the use of scale invariant feature transform (SIFT) features clustered into visual words and fed to a simple Bayesian probabilistic classifier that matches the food items to a food database containing images of fast-food, homemade food, and fruits. A recognition performance of 92% was reported given that the number of references per food class in the database is larger than 50 and the number of food items to be recognized is less than six. The proposed food recognition system consists of two stages: food image description and image classification. During food image description, a set of characteristics representing the visual content of the image is extracted and quantified. This set provides input to the second stage, where a classifier assigns to the image one class out of a predefined set of food classes. The design and development of both stages involves two phases: training and testing. During the training phase, the system learns from the acquired knowledge, while during the testing phase the system recognizes food types from new, unknown images.

The first experiment proved the superiority of dense key point extraction which was able to produce the required large number of patches with minimum overlap between them. The second experiment investigated the effect of the descriptor’s size on the final performance. The best results were obtained by the combination of descriptors with sizes 16, 24, and 32. By using descriptors with different sizes, the BoF system gained multi resolution properties that increased the final performance, since the food scale may vary among the images. Then, the hsv SIFT was chosen among 14 different color and texture descriptors as giving the best results. Hsv SIFT constitutes a differential descriptor that describes the local texture in all three different color channels of the HSV color space. This fact enables it to include color information, apart from texture, but also keep some invariance in intensity and color changes. The color capturing ability of hsv-SIFT was also proved by the descriptors’ fusion experiment that failed to increase the performance after combining it with the best color descriptors. As regards the learning of the visual dictionary, k-means was compared to its hierarchical version hk-means. The latter managed to produce almost equivalent results with k-means, for the optimal number of visual words, while being extremely computationally efficient. The optimal number of words was determined to be approximately 10000, since fewer words resulted in clearly worse results and more words did not improve the performance. For the final classification, two linear and four nonlinear machine-learning techniques were employed, with the linear giving the best results, especially for large number of features. This is probably caused by the high dimensionality of the feature space, as this makes the problem linearly separable, at least to some extent.

**Advantages:**

The surface representation and SIFT is used to reduce the dimensionality of the histogram space.

**Disadvantages:**

In the image texture on method the local feature is not a function of the imaging parameters.

**3. Title: The Design and Implementation of an Ingredient Based Food Calorie Estimation System Using Nutrition Knowledge and Fusion of Brightness and Heat Information**

**Author: SIRICHAI TURMCHOKKASAM AND KOSIN CHAMNONGTHAI**

To measure the calorie of food, which are varied depending on its ingredients and volume in each cooking time, it is required to calculate calories of food before consuming. Based on nutrition knowledge, ingredients that are components of food naturally have different calories. This paper proposes a method of ingredient-based food calorie estimation using nutrition knowledge and thermal information. In this method, an image of the food is first recognized as a type of food, and ingredients of the recognized food are retrieved from the database with their nutrition knowledge and pattern of brightness and thermal images. Simultaneously, the image is segmented into boundaries of ingredient candidates, and all boundaries are then classified into ingredients using fuzzy logic based on their heat pattern and intensities. The classified ingredients from all boundaries are finally calculated for total calories based on area ratio and nutrition knowledge. The performance of our proposed method shows acceptable results comparing with the calories set up by the conventional destructive method.

Originally, food is input into the human body for the objective of energy, growth, organ tuning and maintenance, immunity, etc. While the human society has developed in civilization, human food tends to become worse due to favorite tastes, delivery convenience, long-term preservation, and so on. Health as the original purpose of food seems to be gradually ignored, and thus, people in civilized societies increasingly become obese and die. One of the important factors in human food is calories, which a man/woman should strictly consume in daily consumption. To measure food calories in order to calculate daily consumption of calories, food is baked, and burnt as the destructive way in five-hours and eight-minutes time, respectively, and the calories are subsequently measured by the decreased weight. Currently, the calories of food evaluated by the aforementioned destructive way are popularly used as a reliable standard for consumers to count and accumulate their daily-consumed calories. The calorie standard is statistically averaged data, which is exactly not guaranteed to match to all meals, even the same food types. To obtain accurate estimation, it is necessary to develop an automatic system that practically measures calories in each meal in real time.

Some researchers have worked toward this research problem for the final goal of food calorie measurement and estimation in each meal as follows. Determination of food portion size by image processing. This method mainly measures the area of food and calculates total calories by referring to statistical calorie data. Thermal imaging of pancakes (raw materials of the pancake) that are produced for the application in quality control. Thermal images of pancakes in the course of the baking process and thermal signal reconstruction of flour with various water content are introduced using an image processing technique. Presented a method of measuring calories and nutrition from food images. This uses a food image captured by the built in camera of mobile devices, segments a dish into food components, and calculates total calories using nutritional fact tables. Here presented a method of measuring calories and nutrition from food images. This is to identify food items in an image using image processing and segmentation. The K-nearest neighbor (KNN) method is used to recognize the food, and the calorie values are measured with the help of a nutrition table. A method using distance estimation and deep learning to simplify calibration in food calorie measurement and food calorie measurement using a deep learning network. This method assumes the application on a mobile device, which measures food area and converts into food weight as calibration for calorie measurements. The MT-Diet: automated smartphone based diet assessment with infrared images. This assumes the application on a smartphone and utilizes infrared to sense food components inside the dish. Hence proposed a semi-automated system for predicting calories in photographs of meals. The system uses a semi-automated approach to allow users to manually draw around the food portion using a polygonal tool. The performance is acceptable. Here presented a system called Im2Calories: towards an automated mobile vision for food diary. The system recognizes the contents of the meal from a single image and predicts its nutritional contents such as calories using CNN, which is trained by many images in advance. The system works well with many types of foods. Here proposed a method of a food calorie measurement system for obesity management.

**Advantages:**

The Absolute Gray Level Difference (AGLD) between a pixel and its neighbours to generate text ons, and used the histogram of them to represent the image.

**Disadvantages:**

The performance of CLBP is not better than that of VZ\_MR8 and VZ\_Joint on this database.

**4. Title: On the Significance of Real-World Conditions for Material Classification.-2004.**

**Author: Eric Hayman, Barbara Caputo, Mario Fritz and Jan-Olof Eklundh.**

Classifying materials from their appearance is a challenging problem, especially if illumination and pose conditions are permitted to change: highlights and shadows caused by 3D structure can radically alter a sample’s visual texture. Despite these difficulties, researchers have demonstrated impressive results on the CUReT database which contains many images of 61 materials under different conditions. A first contribution of this paper is to further advance the state-of-threat by applying Support Vector Machines to this problem. To our knowledge, we record the best results to date on the CUReT database.

In our work we additionally investigate the effect of scale since robustness to viewing distance and zoom settings is crucial in many real-world situations. Indeed, a material’s appearance can vary considerably as fine-level detail becomes visible or disappears as the camera moves towards or away from the subject. We handle scale-variations using a pure-learning approach, incorporating samples imaged at different distances into the training set. An empirical investigation is conducted to show how the classification accuracy decreases as less scale information is made available during training.

Since the CUReT database contains little scale variation, we introduce a new database which images ten CUReT materials at different distances, while also maintaining some change in pose and illumination. The first aim of the database is thus to provide scale variations, but a second and equally important objective is to attempt to recognise different samples of the CUReT materials. For instance, does training on the CUReT database enable recognition of another piece of sandpaper? The results clearly demonstrate that it is not possible to do so with any acceptable degree of accuracy. Thus we conclude that impressive results even on a well-designed database such as CUReT, does not imply that material classification is close to being a solved problem under real-world conditions.

The recognition of materials from their visual texture has many applications, for instance it facilitates image retrieval and object recognition. As a step towards the use of such techniques in the real world, recent developments have concentrated on being able to recognize materials from a variety of poses and with different illumination conditions. This is a particularly challenging task when the material has considerable 3-dimensional structure. With such 3D textures, cast shadows and highlights can cause the appearance to change radically with different viewing angles and illumination conditions. An example from the CUReT database.

The overall goal of our work is to bring material recognition algorithms closer still to the stage where they will be useful in real-world applications. Thus a major objective is providing robustness to variations in scale. Experiments will show that failure in this regard rapidly leads to a deterioration in classification accuracy. Our solution is a pure-learning approach which accommodates variations in scale in the training samples, similar to how differing illumination and pose are modeled. A further contribution concerns demonstrating the suitability of Support Vector Machines as classifiers in this recognition problem. Experiments show that the SVM classifier systematically outperforms the nearest-neighbor classification scheme adopted by Varma and Zisserman with which we compare our results, and we also demonstrate that we achieve an improvement on their Markov Random Field (MRF) approach which, to our knowledge, previously yielded the best overall classification rate on the CUReT database. As already alluded to, experiments are conducted on the CUReT image database which captures variations in illumination and pose for 61 different materials, many of which contain significant 3D structure. This database does not, however, contain many scaling effects. Some indication of the performance under varying scale can be achieved by artificially scaling the images by modifying the scales of the filters in the filter bank. However, we also investigate classification results on pictures of materials present in the CUReT database, imaged in our laboratory. The objectives of these experiments are two-fold. First, it permits a systematic study of scale effects while still providing some variations in pose and illumination. Second, we investigate whether it is possible to recognize materials in this new database given models trained on the CUReT database. This indeed proves a stern test, since both the sample of material, the camera and lighting conditions are different to those used during training. Thus the final contribution of this paper is the construction of a new database, designed to complement the CUReT database with scale variations. This database, called KTH-TIPS (Textures under varying Illumination Pose and Scale) is freely available to other researchers via the web.

**Advantages:**

This indeed proves a stern test, since both the sample of material, the camera and lighting conditions are different to those used during training.

**Disadvantages**:

This database does not, however, contain many scaling effects.

**5. Title: A Deep Learning based Food Recognition System for Lifelog Images**

**Author:** Binh T. Nguyen, Duc-Tien Dang-Nguyen, Tien X. Dang1, Thai Phat1, Cathal Gurrin

In this paper, we propose a deep learning based system for food recognition from personal life archive images. The system first identifies the eating moments based on multi-modal information, then tries to focus and enhance the food images available in these moments, and finally, exploits GoogleNet as the core of the learning process to recognise the food category of the images. Preliminary results, experimenting on the food recognition module of the proposed system, show that the proposed system achieves 95.97% classification accuracy on the food images taken from the personal life archive from several lifeloggers, which potentially can be extended and applied in broader scenarios and for different types of food categories.

Lifelogging is the process whereby individuals gather personal data about different aspects of their normal life activities for different purposes (Gurrin et al., 2014b), such as capturing photos of important daily events, logging sleep patterns, recording workouts, keeping records of food consumption or even mood changes. This collection of personal data about the individual’s life is typically called a lifelog or personal life archive. As digital storage is becoming cheaper and sensing technology is improving continuously, continuous and passive photo capture devices are now more popular and affordable. These devices can help us to maintain digital records about our daily life much efficiently. In the domain of health care, life-logging can play an important role in providing historical information about the person. It can be seen as a tool that monitors and tracks the quality of our life; our life-log records have the potential to tell us if we are sleeping well or not based on our sleeping patterns, if we have a proper diet based on our eating behavior or how well we are benefiting from our workouts via the analysis of calories burned and heart rate. In addition to automatically keeping a valuable diary of important moments and events of our life through the use of photos via wearable life-logging cameras and/or traditional cameras, visual life-logs can contain detailed information about our daily activities. A wearable camera, such as a Sense-Cam, can capture around one million images per year (Dang-Nguyen et al., 2017a), recording a huge amount of visual information about the wearer’s life including food consumption details. Such food photos can be leveraged to track nutritional intake of the individual on a personal level, which can provide important insights into the individual’s dietary habits and can also lead to many interesting applications such as automatic calculator of food consumption or personalized food recommendation systems. Monitoring the nutrition habits of a person is an established mechanism typically used in the health domain for several medical conditions such as obesity, hypertension and diabetes (A., 1992). Utilizing the visual food information available in one’s life-log can effectively replace traditional methods of food consumption analysis that currently depend mainly on subjective questionnaires, manual surveys and interviews (Liu et al., 2012). This can both make the process easier to the user, as well as providing objective results with fewer errors when compared to conventional manual food monitoring methods.

In the case of feature based approaches such as filtering with Gabor wavelets or other basis functions, rotation invariance is realized by computing rotation invariant features from the filtered images or by converting rotation variant features to rotation invariant features. Using a circular neighbor set, presented rotation invariant generalizations for all three mainstream paradigms.

**Advantages:**

In comparison to much larger Gabor filters that are often used in texture analysis.

**Disadvantages:**

They also serve as a welcome reminder that the addition of inferior operator does not necessarily enhance the performance.

**6. Title: Food Recognition using Ingredient-Level Features**

**Author: Jay Baxter Massachusetts Institute of Technology**

Food recognition is a difficult problem, because unlike objects like cars, faces, or pedestrians, food is deformable and exhibits high intra-class variation. This paper considers the approach of analyzing a food item at the pixel level by classifying each pixel as a certain ingredient, and then using statistics and spatial relationships between those pixel ingredient labels as features in an SVM classifier. We experimented with multiple variations on past methods, and found that using pixel ingredient labels to identify food greatly increases classification accuracy, but at the expense of higher computational cost.

Food recognition has only become a fairly popular topic in the last few years, which is largely a result of the quickly growing number of people who routinely take pictures of their food with cell phone cameras before eating it. The main application of food recognition is to create a nutritional information phone application that is able to analyze these pictures of food and deduce the nutritional content of the food eaten. The most critical and difficult step of this process is recognizing the type of food in the picture: once the type of food is determined, estimating quantity and nutritional information is much easier. Quantity can be determined by asking participants to include a thumb in their picture or have the food a fixed distance away from the camera. Nutritional information can be looked up in official databases. Therefore, this paper addresses the problem of food recognition: determining what type of food is in the picture, given that we know that the input picture is of food, and that the food is the main focus of the picture. We assume that the background is rather plain, like a plain tabletop. Even with such a restricted problem domain, food recognition is a very hard problem. Unlike other types of objects where object recognition has been more successful, such as faces, cars, and pedestrians, food is very deformable and has high intra-class variation, as is shown in Figure 1. Pasta is deformable, meaning that it is amorphous. The definition of pasta has nothing to do with shape, and only has to do with its ingredients and method of preparation. Ingredients and method of preparation manifest themselves in the visible features shape, color, and texture. However, even in a type of food we often think of as having a rigid shape and structure, a Big Mac, there is high intra-class variation: sometimes the meat is hidden underneath the bun, sometimes the cheese isn’t visible, sometimes the lettuce is hidden, etc. In this paper, we address these problems by trying to identify the ingredients that make up a food item. Humans describe food items and their differences in terms of the ingredients they contain and the way the ingredients are arranged, so it makes intuitive sense that we may be able to glean extra information by extracting features at an intermediate ingredient level instead of just as the original pixel level. After labeling the ingredients in the image, we extract features that describe the relatives quantities of each ingredient (which is useful for distinguishing a salad from a hamburger, because a salad has much more lettuce) as well as the spatial relationships between the ingredients (which is useful for determining a Big Mac from a normal hamburger, since we expect to see bread-meat-bread-meat-bread instead of bread-meat-bread).

**Advantages:**

Initial constraints on descriptor similarity are used to create a short list of potential matches for each region in the other image.

**Disadvantages:**

Combining them with Textron’s-only features once again does not bring any improvement.

**7. Title: A comparative analysis of binary patterns with discrete cosine transform for gender classification.-2011**

**Author: RODRIGUES, Marcos, KORMANN, Mariza and TOMEK, Peter.**

This paper presents a comparative analysis of binary patters for gender classification with a novel method of feature transformation for improved accuracy rates. The main requirements of our application are speed and accuracy. We investigate a combination of local binary patterns (LBP), Census Transform (CT) and Modified Census Transform (MCT) applied over the full, top and bottom halves of the face. Gender classification is performed using support vector machines (SVM). A main focus of the investigation is to determine whether or not a 1D discrete cosine transform (DCT) applied directly to the grey level histograms would improve accuracy. We used a public database of faces and run face and eye detection algorithms allowing automatic cropping and normalization of the images. A set of 120 tests over the entire database demonstrate that the proposed 1D discrete cosine transform improves accuracy in all test cases with small standard deviations. It is shown that using basic versions of the algorithms, LBP is marginally superior to both CT and MCT and agrees with results in the literature for higher accuracy on male subjects. However, a significant result of our investigation is that, by applying a 1D-DCT this bias is removed and an equivalent error rate is achieved for both genders. Furthermore, it is demonstrated that DCT improves overall accuracy and renders CT a superior performance compared to LBP in all cases considered.

Real-time gender classification is a requirement for marketing applications where legal and ethical constraints do not allow the saving of images either locally or remotely for later processing. The ADMOS project is funded by the European Union and aims to develop a real time gender classification and age estimation to be used in private spaces of public use, such as shopping malls, fairs and outdoor events. The main computing operations on an image within the time frame of live capture include face detection, gender classification and age estimation. We are investigating a number of methods that have the potential to be fast, accurate and robust. In this paper we report on a combination of techniques involving LBP–Local Binary Patterns, CT–Census Transform, MCT–Modified Census Transform, DCT–Discrete Cosine Transform and SVM–Support Vector Machines. It is shown that DCT can remove LBP’s bias towards higher accuracy for male subjects and that it renders CT a superior technique when compared to LBP.

LBP is a non-parametric method used to summarize local structures of an image and have been extensively exploited in face analysis for gender, age, and face recognition Normally, LBP are employed in local and holistic approaches and a number of extensions have been demonstrated in the literature in connection with linear discriminant analysis and support vector machines. The Census Transform is similar to LBP; the main difference lies on how bits are concatenated together. Although this is a seemingly small difference, it has significant bearings on the final grey level scale histograms and thus, on the texture descriptors in various regions of an image.

The Census Transform has not been extensively exploited in face analysis as LBP; some previous work include. LBP and DCT have been used together in connection with face recognition and gender classification. However, it is important to note that when DCT is used, it is invariably in connection with a 2DDCT. Normally a DCT is performed over the entire input image using different block sizes. In a similar fashion, LBP is normally applied over regions and over the entire image and such histograms are concatenated into a combined one. Here we explore these techniques aiming at fast processing for real time applications. We only use a single pass, non-optimised LBP, CT or MCT over the input image, followed by a 1D-DCT applied to the resulting histograms. The purpose is to investigate whether or not the 1D-DCT would improve gender classification. The approach is demonstrated by using a public database from which the various regions of interest are automatically selected by face and eye detection algorithms. It is shown that 1D-DCT improves gender classification in all cases considered.

**Advantages**:

Furthermore, whether or not the discrete cosine transform can be effectively used for gender classification in connection with such feature extraction techniques.

**Disadvantages**:

The Census Transform has not been extensively exploited in face analysis as LBP.

**8. Title: Enhanced Local Texture Feature Sets for Face Recognition Under Difficult Lighting Conditions.-2010**

**Author: Xiaoyang Tan and Bill Triggs.**

Recognition in uncontrolled situations is one of the most important bottlenecks for practical face recognition systems. We address this by combining the strengths of robust illumination normalization, local texture based face representations and distance transform based matching metrics. Specifically, we make three main contributions: (i) we present a simple and efficient preprocessing chain that eliminates most of the effects of changing illumination while still preserving the essential appearance details that are needed for recognition; (ii) we introduce Local Ternary Patterns (LTP), a generalization of the Local Binary Pattern (LBP) local texture descriptor that is more discriminant and less sensitive to noise in uniform regions; and (iii) we show that replacing local histogram Ming with a local distance transform based similarity metric further improves the performance of LBP/LTP based face recognition. The resulting method gives state-of-the-art performance on three popular datasets chosen to test recognition under difficult illumination conditions.

One of the key challenges of face recognition is finding efficient and discriminative facial appearance descriptors that can counteract large variations in illumination, pose, facial expression, ageing, partial occlusions and other changes. There are two main approaches: geometric feature-based descriptors and appearance-based descriptors. Geometric descriptors can be hard to extract reliably under variations in facial appearance, while appearance-based ones such as Eigen faces tend to blur out small details owing to residual spatial registration errors. Recently, representations based on local pooling of local appearance descriptors have drawn increasing attention because they can capture small appearance details in the descriptors while remaining resistant to registration errors owing to local pooling. Another motivation is the observation that human visual perception is well-adapted to extracting and pooling local structural information (‘micro-patterns’) from images. Methods in this category include Gabor wavelets, local autocorrelation filters, and Local Binary Patterns.

In this paper we focus on Local Binary Patterns (LBP) and their generalizations. LBP’s are a computationally efficient nonparametric local image texture descriptor. They have been used with considerable success in a number of visual recognition tasks including face recognition. LBP features are invariant to monotonic gray-level changes by design and thus are usually considered to require no image preprocessing before use In fact, LBP itself is sometimes used as a lighting normalization stage for other methods. However, in practice the reliability of LBP decreases significantly under large illumination variations (c.f. table 3). Lighting effects involve complex local interactions and the resulting images often violate LBP’s basic assumption that graylevel changes monotonically. We have addressed this problem by developing a simple and efficient image preprocessing chain that greatly reduces the influence of illumination variations, local shadowing and highlights while preserving the elements of visual appearance that are needed for recognition. Another limitation of LBP is its sensitivity to random and quantization noise in uniform and near-uniform image regions such as the forehead and cheeks. To counter this we extend LBP to Local Ternary Patterns (LTP), a 3-valued coding that includes a threshold around zero for improved resistance to noise. LTP inherits most of the other key advantages of LBP such as computational efficiency. Current LBP based face recognition methods partition the face image into a grid of fixed-size cells for the local pooling of texture descriptors (LBP histograms). This coarse (and typically abrupt) spatial quantization is somewhat arbitrary and not necessarily well adapted to local facial morphology. It inevitably causes some loss of discriminative power. To counter this we use distance transform techniques to create local texture comparison metrics that have more controlled spatial gradings.

The second approach seeks conventional image processing transformations that reduce the image to a more “canonical” form in which the variations are suppressed. This has the merit of easy application to real images and the lack of a need for comprehensive training data. Given that complete illumination invariants do not exist, one must content oneself with finding representations that are resistant to the most common classes of natural illumination variations. Most methods exploit the fact that these are typically characterized by relatively low spatial frequencies.

**Advantages**:

A distance transform based similarity metric that captures the local structure and geometric variations of LBP/LTP face images better than the simple grids of histograms that are currently used.

**Disadvantages**:

Gamma correction does not remove the influence of overall intensity gradients such as shading effects.

**9. Title: Dominant Local Binary Patterns for Texture Classification.-2009**

**Author: S. Liao, Max W. K. Law, and Albert C. S. Chung.**

This paper proposes a novel approach to extract image features for texture classification. The proposed features are robust to image rotation, less sensitive to histogram equalization and noise. It comprises of two sets of features: dominant local binary patterns (DLBP) in a texture image and the supplementary features extracted by using the circularly symmetric Gabor filter responses. The dominant local binary pattern method makes use of the most frequently occurred patterns to capture descriptive textural information, while the Gabor-based features aim at supplying additional global textural information to the DLBP features. Through experiments, the proposed approach has been intensively evaluated by applying a large number of classification tests to histogram-equalized, randomly rotated and noise corrupted images in , and CUReT texture image databases. Our method has also been compared with six published texture features in the experiments. It is experimentally demonstrated that the proposed method achieves the highest classification accuracy in various texture databases and image conditions.

TEXTURE classification plays an important role in computer vision and image processing applications. The applications include medical image analysis and understanding, remote sensing, object-based image coding, and image retrieval. As the demand of such applications increases, texture classification has received considerable attention over the last several decades and numerous novel methods have been proposed. For example, used the Gaussian Markov random fields (GMRF) to model texture patterns based on statistical relationship between adjacent pixel intensity the Gabor filters to an image and then computed the average filter responses as features. Introduced the multiresolution wavelet decomposition method, which generates coefficients in the HL, LH, and LL channels for subsequent classification tasks . applied the co-occurrence matrix to extract the mean intensity, contrast, and correlation information from the texture images.

However, the above techniques only encode the absolute texture orientation information, which is inadequate for rotation invariant texture classification. Therefore, to achieve rotation invariance in texture classification, researchers have attempted to either discard all orientation information or capture relative orientation information. To discard orientation information, removed the HH wavelet channels and combined the LH and HL wavelet channels to obtain rotation invariant wavelet features. Calculated isotropic rotation invariant features from Gabor filters. Constructed an isotropic circular Gaussian Markov random field (ICGMRF). On the other hand, some approaches capture the relative directional features rather than the absolute orientation information. into anisotropic circular GMRF model (ACGMRF) to capture rotation invariant relative orientation features. Along the same research line, utilized similar circular neighborhoods with 1-D DFT transformation.

Although the aforementioned methods are proofed to be rotation invariant, they are sensitive to the change of illumination condition which often exists in texture images because of the limitation of the imaging devices or the change of lighting condition. In real world applications, histogram equalization is often performed to mitigate the adverse effect of varying illumination condition. However, as we will show in the Experiments and Results Section (Section IV), the performances of the above methods drop significantly after the histogram equalization has been applied because, for these methods, intensity mean and image contrast are two important pieces of textural information for texture classification. proposed rotation and histogram equalization invariant features by observing the statistical distributions of the uniform local binary pattern the LBP method by calculating the derivative-based LBPs in the application of face alignment. However, the uniform LBPs are not the dominating patterns (i.e., patterns of the largest proportions in an image) in some textures with irregular edges and shapes.

In this paper, we are motivated to propose a new feature extraction method that is robust to histogram equalization and rotation. First, the conventional LBP approach is extended to the dominant local binary pattern (DLBP) approach in order to effectively capture the dominating patterns in texture images. Unlike the conventional LBP approach, which only exploits the uniform LBP, given a texture image, the DLBP approach computes the occurrence frequencies of all rotation invariant patterns defined in the LBP groups. These patterns are then sorted in descending order.

**Advantages**:

The performance of the proposed method is compared with six widely used image features.

**Disadvantages**:

The bilinear interpolation method is applied to retrieve the intensity at the positions that are not on the image grid.

**10. Title: ROTATION-INVARIANT TEXTURE RECOGNITION BY ROTATION COMPENSATION AND WAVELET ANALYSIS.-2010**

**Author: Huiguang Yang and Narendra Ahuja.**

New rotation-invariant wavelet-based texture recognition scheme is proposed. In the previous rotation-invariant approaches, the focus is on adapting the wavelet transform or filter to rotated texture. In our approach, instead, we estimate the rotation of the texture with respect to some reference orientation, and then rotate the texture image back to the reference orientation before applying the wavelet analysis to extract features. With such rotation compensation, even very simple features (such as 1-level DWT and the sub band energy) can be effective in achieving high classification accuracy as we demonstrate through our experiment.

Texture recognition and classification is an important and challenging task in image processing. A number of early methods use the statistical property of the image to analyze textures, such as the co-occurrence matrix method. Later on Gabor filter and wavelet-based methods become the classical methods for texture recognition. Other common methods include Gaussian Markov random field , local binary pattern (LBP) histogram autoregressive model and hidden Markov models. Among all those methods, Gabor filter, and particularly, wavelet-based methods are perhaps the most popular ones. The reason for the popularity of the wavelet-based methods is that it provides a natural partition of the image spectrum into multiscale and oriented sub bands via efficient transforms. There has been a rich variety of the wavelet-based texture analysis methods such as wavelet transform, wavelet packets, complex wavelet transform, rotated wavelet filter and so forth. Most of the wavelet methods make use of the energy distribution among the sub bands in frequency domain to identify texture. The sub band energy may be used directly, or features may be extracted from the wavelet coefficients using techniques such as generalized Gaussian density model.

Wavelet-based texture recognition is among the most effective and popular methods for texture analysis, which is due to the natural property of wavelet analysis that it can partition the image spectrum into multiscale and oriented sub bands. However, original wavelet-based texture analysis is not rotation-invariant, and has difficulties in recognizing the same texture with different orientations. Therefore, it is important to develop a texture recognition scheme that is rotation-invariant. In this study, we have proposed a new texture recognition scheme based on the approach called rotation compensation. In essence, we first estimate the rotation of the texture image with respect to some reference orientation, and then rotate the texture image back before performing wavelet-based analysis.

**Advantages:**

The classification performance might be further improved if more sophisticated features are used.

**Disadvantages:**

Original wavelet-based texture analysis is not rotation-invariant, and has difficulties in recognizing the same texture with different orientations.

**SYSTEM REQUIREMENTS**

**5.1 HARDWARE REQUIREMENTS:**

HARDWARE REQUIREMENTS:

* Processor : Intel Pentium.
* RAM : 4 GB

**5.2 SOFTWARE REQUIREMENTS:**

SOFTWARE REQUIREMENTS:

* OS : Windows 7
* Software : PYTHON Anaconda 3.7

**5.3 SOFTWARE DESCRIPTION:**

**5.3.1 Python**

Python is one of those rare languages which can claim to be both *simple* and powerful. You will find yourself pleasantly surprised to see how easy it is to concentrate on the solution to the problem rather than the syntax and structure of the language you are programming in. The official introduction to Python is Python is an easy to learn, powerful programming language. It has efficient high-level data structures and a simple but effective approach to object-oriented programming. Python's elegant syntax and dynamic typing, together with its interpreted nature, make it an ideal language for scripting and rapid application development in many areas on most platforms. I will discuss most of these features in more detail in the next section.

## **5.3.2 Features of Python**

### Simple

Python is a simple and minimalistic language. Reading a good Python program feels almost like reading English, although very strict English! This pseudo-code nature of Python is one of its greatest strengths. It allows you to concentrate on the solution to the problem rather than the language itself.

### Easy to Learn

As you will see, Python is extremely easy to get started with. Python has an extraordinarily simple syntax, as already mentioned.

### Free and Open Source

Python is an example of a FLOSS (Free/Libré and Open Source Software). In simple terms, you can freely distribute copies of this software, read its source code, make changes to it, and use pieces of it in new free programs. FLOSS is based on the concept of a community which shares knowledge. This is one of the reasons why Python is so good - it has been created and is constantly improved by a community who just want to see a better Python.

### High-level Language

When you write programs in Python, you never need to bother about the low-level details such as managing the memory used by your program, etc.

### Portable

Due to its open-source nature, Python has been ported to (i.e. changed to make it work on) many platforms. All your Python programs can work on any of these platforms without requiring any changes at all if you are careful enough to avoid any system-dependent features.

You can use Python on GNU/Linux, Windows, FreeBSD, Macintosh, Solaris, OS/2, Amiga, AROS, AS/400, BeOS, OS/390, z/OS, Palm OS, QNX, VMS, Psion, Acorn RISC OS, VxWorks, PlayStation, Sharp Zaurus, Windows CE and PocketPC!

You can even use a platform like [Kivy](http://kivy.org" \t "_blank) to create games for your computer and for iPhone, iPad, and Android.

### Interpreted

This requires a bit of explanation.

A program written in a compiled language like C or C++ is converted from the source language i.e. C or C++ into a language that is spoken by your computer (binary code i.e. 0s and 1s) using a compiler with various flags and options. When you run the program, the linker/loader software copies the program from hard disk to memory and starts running it.

Python, on the other hand, does not need compilation to binary. You just run the program directly from the source code. Internally, Python converts the source code into an intermediate form called bytecodes and then translates this into the native language of your computer and then runs it. All this, actually, makes using Python much easier since you don't have to worry about compiling the program, making sure that the proper libraries are linked and loaded, etc. This also makes your Python programs much more portable, since you can just copy your Python program onto another computer and it just works!

### Object Oriented

Python supports procedure-oriented programming as well as object-oriented programming. In procedure-oriented languages, the program is built around procedures or functions which are nothing but reusable pieces of programs. In object-oriented languages, the program is built around objects which combine data and functionality. Python has a very powerful but simplistic way of doing OOP, especially when compared to big languages like C++ or Java.

### Extensible

If you need a critical piece of code to run very fast or want to have some piece of algorithm not to be open, you can code that part of your program in C or C++ and then use it from your Python program.

### Embeddable

You can embed Python within your C/C++ programs to give scripting capabilities for your program's users.

### Extensive Libraries

The Python Standard Library is huge indeed. It can help you do various things involving regular expressions, documentation generation, unit testing, threading, databases, web browsers, CGI, FTP, email, XML, XML-RPC, HTML, WAV files, cryptography, GUI (graphical user interfaces), and other system-dependent stuff. Remember, all this is always available wherever Python is installed. This is called the Batteries Included philosophy of Python.

Besides the standard library, there are various other high-quality libraries which you can find at the Python Package Index.

**5.4 TESTING PRODUCTS:**

System testing is the stage of implementation, which aimed at ensuring that system works accurately and efficiently before the live operation commence. Testing is the process of executing a program with the intent of finding an error. A good test case is one that has a high probability of finding an error. A successful test is one that answers a yet undiscovered error.

Testing is vital to the success of the system. System testing makes a logical assumption that if all parts of the system are correct, the goal will be successfully achieved. . A series of tests are performed before the system is ready for the user acceptance testing. Any engineered product can be tested in one of the following ways. Knowing the specified function that a product has been designed to from, test can be conducted to demonstrate each function is fully operational. Knowing the internal working of a product, tests can be conducted to ensure that “al gears mesh”, that is the internal operation of the product performs according to the specification and all internal components have been adequately exercised.

**PYTHON and images**

* The help in PYTHON is very good, use it!
* An image in PYTHON is treated as a matrix
* Every pixel is a matrix element
* All the operators in PYTHON defined on Matrices can be used on images: +, -, \*, /, ^, sqrt, sin, cos etc.
* **PYTHON can import/export several image formats**
  + BMP (Microsoft Windows Bitmap)
  + GIF (Graphics Interchange Files)
  + HDF (Hierarchical Data Format)
  + JPEG (Joint Photographic Experts Group)
  + PCX (Paintbrush)
  + PNG (Portable Network Graphics)
  + TIFF (Tagged Image File Format)
  + XWD (X Window Dump)
  + PYTHON can also load raw-data or other types of image data
* **Data types in PYTHON**
  + Double (64-bit double-precision floating point)
  + Single (32-bit single-precision floating point)
  + Int32 (32-bit signed integer)
  + Int16 (16-bit signed integer)
  + Int8 (8-bit signed integer)
  + Uint32 (32-bit unsigned integer)
  + Uint16 (16-bit unsigned integer)
  + Uint8 (8-bit unsigned integer)

**Images in PYTHON**

Binary images: {0, 1}

• Intensity images: [0, 1] or uint8, double etc.

• RGB images: m-by-n-by-3

• Indexed images: m-by-3 color map

• Multidimensional images m-by-n-by-p (p is the number of layers)

**IMAGE TYPES IN PYTHON**

Outside PYTHON images may be of three types i.e. black & white, grey scale and colored. In PYTHON, however, there are four types of images. Black & White images are called binary images, containing 1 for white and 0 for black. Grey scale images are called intensity images, containing numbers in the range of 0 to 255 or 0 to 1. Colored images may be represented as RGB Image or Indexed Image.

In RGB Images there exist three indexed images. First image contains all the red portion of the image, second green and third contains the blue portion. So for a 640 x 480 sized image the matrix will be 640 x 480 x 3. An alternate method of colored image representation is Indexed Image. It actually exist of two matrices namely image matrix and map matrix. Each color in the image is given an index number and in image matrix each color is represented as an index number. Map matrix contains the database of which index number belongs to which color.

**IMAGE TYPE CONVERSION**

* RGB Image to Intensity Image (rgb2gray)
* RGB Image to Indexed Image (rgb2ind)
* RGB Image to Binary Image (im2bw)
* Indexed Image to RGB Image (ind2rgb)
* Indexed Image to Intensity Image (ind2gray)
* Indexed Image to Binary Image (im2bw)
* Intensity Image to Indexed Image (gray2ind)
* Intensity Image to Binary Image (im2bw)
* Intensity Image to RGB Image (gray2ind, ind2rgb)

**Key Features**

* High-level language for technical computing
* Development environment for managing code, files, and data
* Interactive tools for iterative exploration, design, and problem solving
* Mathematical functions for linear algebra, statistics, Fourier analysis, filtering, optimization, and numerical integration
* 2-D and 3-D graphics functions for visualizing data
* Tools for building custom graphical user interfaces

Functions for integrating PYTHON based algorithms with external applications and languages, such as C, C++, FORTRAN, Java, COM, and Microsoft Excel.

**FEASIBILITY STUDY**

The feasibility study is carried out to test whether the proposed system is worth being implemented. The proposed system will be selected if it is best enough in meeting the performance requirements.

The feasibility carried out mainly in three sections namely.

• Economic Feasibility

• Technical Feasibility

• Behavioral Feasibility

**Economic Feasibility**

Economic analysis is the most frequently used method for evaluating effectiveness of the proposed system. More commonly known as cost benefit analysis. This procedure determines the benefits and saving that are expected from the system of the proposed system. The hardware in system department if sufficient for system development.

**Technical Feasibility**

This study center around the system’s department hardware, software and to what extend it can support the proposed system department is having the required hardware and software there is no question of increasing the cost of implementing the proposed system. The criteria, the proposed system is technically feasible and the proposed system can be developed with the existing facility.

**Behavioral Feasibility**

People are inherently resistant to change and need sufficient amount of training, which would result in lot of expenditure for the organization. The proposed system can generate reports with day-to-day information immediately at the user’s request, instead of getting a report, which doesn’t contain much detail.

**System Implementation**

Implementation of software refers to the final installation of the package in its real environment, to the satisfaction of the intended users and the operation of the system. The people are not sure that the software is meant to make their job easier.

• The active user must be aware of the benefits of using the system

• Their confidence in the software built up

• Proper guidance is impaired to the user so that he is comfortable in using the application

Before going ahead and viewing the system, the user must know that for viewing the result, the server program should be running in the server. If the server object is not running on the server, the actual processes will not take place.

**User Training**

To achieve the objectives and benefits expected from the proposed system it is essential for the people who will be involved to be confident of their role in the new system. As system becomes more complex, the need for education and training is more and more important. Education is complementary to training. It brings life to formal training by explaining the background to the resources for them. Education involves creating the right atmosphere and motivating user staff. Education information can make training more interesting and more understandable.

**Training on the Application Software**

After providing the necessary basic training on the computer awareness, the users will have to be trained on the new application software. This will give the underlying philosophy of the use of the new system such as the screen flow, screen design, type of help on the screen, type of errors while entering the data, the corresponding validation check at each entry and the ways to correct the data entered. This training may be different across different user groups and across different levels of hierarchy.

**Operational Documentation**

Once the implementation plan is decided, it is essential that the user of the system is made familiar and comfortable with the environment. A documentation providing the whole operations of the system is being developed. Useful tips and guidance is given inside the application itself to the user. The system is developed user friendly so that the user can work the system from the tips given in the application itself.

**System Maintenance**

The maintenance phase of the software cycle is the time in which software performs useful work. After a system is successfully implemented, it should be maintained in a proper manner. System maintenance is an important aspect in the software development life cycle. The need for system maintenance is to make adaptable to the changes in the system environment. There may be social, technical and other environmental changes, which affect a system which is being implemented. Software product enhancements may involve providing new functional capabilities, improving user displays and mode of interaction, upgrading the performance characteristics of the system. So only thru proper system maintenance procedures, the system can be adapted to cope up with these changes. Software maintenance is of course, far more than “finding mistakes”.

**Corrective Maintenance**

The first maintenance activity occurs because it is unreasonable to assume that software testing will uncover all latent errors in a large software system. During the use of any large program, errors will occur and be reported to the developer. The process that includes the diagnosis and correction of one or more errors is called Corrective Maintenance.

**Adaptive Maintenance**

The second activity that contributes to a definition of maintenance occurs because of the rapid change that is encountered in every aspect of computing. Therefore Adaptive maintenance termed as an activity that modifies software to properly interfere with a changing environment is both necessary and commonplace.

**Perceptive Maintenance**

The third activity that may be applied to a definition of maintenance occurs when a software package is successful. As the software is used, recommendations for new capabilities, modifications to existing functions, and general enhancement are received from users. To satisfy requests in this category, Perceptive maintenance is performed. This activity accounts for the majority of all efforts expended on software maintenance.

**Preventive Maintenance**

The fourth maintenance activity occurs when software is changed to improve future maintainability or reliability, or to provide a better basis for future enhancements. Often called preventive maintenance, this activity is characterized by reverse engineering and re-engineering techniques

**CONCLUSION**

* The probability distribution was modeled with the intrinsic oil spill characteristics, which distinguish the oil spills from the background.
* We then exploited the probability distribution representation to construct the segmentation energy functional to implement oil spill segmentation.
* The incorporated probability distribution representation enhances the representational power of the segmentation energy functional.
* Particularly, in our proposed segmentation method, benefiting from the incorporation of the oil spill SAR image formation, the intrinsic characteristics of oil spills are exploited in favour of guiding the segmentation to operate intentionally toward the oil spill areas to perform accurate oil spill segmentation.
* Experimental results have shown the effectiveness of our proposed segmentation method, compared against several state-of-the-art segmentation methodologies.

**REFERENCES**

[1] H. Goncalves, L. Corte-Real, and J. Goncalves, “Automatic image registration through image segmentation and SIFT,” IEEE Trans. Geosci. Remote Sens., vol. 49, no. 7, pp. 2589–2600, Jul. 2011.

[2] A. Sedaghat, M. Mokhtarzade, and H. Ebadi, “Uniform robust scaleinvariant feature matching for optical remote sensing images,” IEEE Trans. Geosci. Remote Sens., vol. 49, no. 11, pp. 4516–4527, Nov. 2011.

[3] X. Jianbin, H. Wen, and W. Yirong, “An efficient rotation-invariance remote image matching algorithm based on feature points matching,” in Proc. I EEE Int. Geosci. Remote Sens. S ymp., 2005, vol. 1, pp. 647–649.

[4] J. Dai, W. Song, L. Pei, and J. Zhang, “Remote sensing image matching via Harris detector and SIFT discriptor,” in Proc. Int. Congr. Image Signal Process., 2010, vol. 5, pp. 2221–2224.

[5] A. Mukherjee, M. Velez-Reyes, and B. Roysam, “Interest points for hyperspectral image data,” IEEE Trans. Geosci. Remote Sens., vol. 47, no. 3, pp. 748–760, Mar. 2009.

[6] L. Dorado-Munoz, M. Velez-Reyes, A. Mukherjee, and B. Roysam, “A vector SIFT operator for interest point detection in hyperspectral imagery,” in Proc. Workshop Hyperspectr. Image Signal Process.—Evolution Remote Sensing, 2010, pp. 1–4.

[7] Z. Xiong and Y. Zhang, “A novel interest-point-matching algorithm for high-resolution satellite images,” IEEE Trans. Geosci. Remote Sens., vol. 47, no. 12, pp. 4189–4200, Dec. 2009.

[8] C. Huo, Z. Zhou, Q. Liu, J. Cheng, H. Lu, and K. Chen, “Urban change detection based on local features and multiscale fusion,” in Proc. IEEE Int. Geosci. Remote Sens. Symp., 2008, vol. 3, pp. 1236–1239.

[9] F. Tang and V. Prinet, “Computing invariants for structural change detection in urban areas,” in Proc. Urban Remote Sens. Joint Event, 2007, pp. 1–6.

[10] B. Sirmacek and C. Unsalan, “Urban-area and building detection using SIFT keypoints and graph theory,” IEEE Trans. Geosci. Remote Sens., vol. 47, no. 4, pp. 1156–1167, Apr. 2009.

[11] B. Sirmacek and C. Unsalan, “Urban area detection using local feature points and spatial voting,” IEEE Geosci. Remote Sens. Lett., vol. 7, no. 1, pp. 146–150, Jan. 2010.

[12] B. Sirmacek and C. Unsalan, “A probabilistic framework to detect buildings in aerial and satellite images,” IEEE Trans. Geosci. Remote Sens., vol. 49, no. 1, pp. 211–221, Jan. 2011.

[13] S. Xu, T. Fang, D. Li, and S. Wang, “Object classification of aerial images with bag-of-visual words,” IEEE Geosci. Remote Sens. Lett., vol. 7, no. 2, pp. 366–370, Apr. 2010.

[14] L. Chen, W. Yang, K. Xu, and T. Xu, “Evaluation of local features for scene classification using VHR satellite images,” in Proc. Urban Remote Sens. Joint Event, 2011, pp. 385–388.

[15] A. Skurikhin, “Visual attention based detection of signs of anthropogenic activities in satellite imagery,” in Proc. IEEE Appl. Imag. Pattern Recog. Workshop, 2010, pp. 1–8.

[16] S. Gleason, R. Ferrell, A. Cheriyadat, R. Vatsavai, and S. De, “Semantic information extraction from multispectral geospatial imagery via a flexible framework,” in Proc. IEEE Int. Geosci. Remote Sens. Symp., 2010, pp. 166–169.

[17] R. R. Vatsavai, A. Cheriyadat, and S. Gleason, “Unsupervised semantic labeling framework for identification of complex facilities in highresolution remote sensing images,” in Proc. Int. Conf. Data Mining Workshops, 2010, pp. 273–280.

[18] B. Ozdemir and S. Aksoy, “Image classification using subgraph histogram representation,” in Proc. Int. Conf. Pattern Recog., 2010, pp. 1112–1115.

[19] J. Bordes and V. Prinet, “Mixture distributions for weakly supervised classification in remote sensing images,” in Proc. Brit. Mach. Vis. Conf., 2008.

[20] L. Fei-Fei, R. Fergus, and P. Perona, “Learning generative visual models from few training examples: An incremental Bayesian approach tested on 101 object categories,” in Proc. IEEE Int. Conf. Comput. Vis. Pattern Recog, 2004, p. 178.

[21] G. Griffin, A. Holub, and P. Perona, “Caltech-256 object category dataset,” California Inst. Technol., Pasadena, CA, Tech. Rep. 7694, 2007.

[22] M. Everingham, A. Zisserman, C. Williams, L. Van Gool, M. Allan, C. Bishop, O. Chapelle, N. Dalal, T. Deselaers, G. Dorkó, S. Duffner, J. Eichhorn, J. Farquhar, M. Fritz, C. Garcia, T. Griffiths, F. Jurie, D. Keysers, M. Koskela, J. Laaksonen, D. Larlus, B. Leibe, H. Meng, H. Ney, B. Schiele, C. Schmid, E. Seemann, J. Shawe-Taylor, A. Storkey, S. Szedmak, B. Tr iggs, I . Ulusoy, V. Viitaniemi, and J. Zhang, “The 2005 PASCAL visual object classes challenge,” in Proc. Pascal Challenges Workshop, vol. 3944, Lecture Notes in Computer Science, 2006, pp. 117–176.

[23] J. Ashley, M. Flickner, J. Hafner, D. Lee, W. Niblack, and D. Petkovic, “The query by image content (QBIC) system,” in Proc. ACM SIGMOD Int. Conf. Manag. Data, 1995, p. 475.

[24] Q. Bao and P. Guo, “Comparative studies on similarity measures for remote sensing image retrieval,” in Proc. IEEE Int. Conf. Syst., Man Cybern., 2004, pp. 1112–1116.

[25] T. Bretschneider, R. Cavet, and O. Kao, “Retrieval of remotely sensed imagery using spectral information content,” in Proc. IEEE Int. Geosci. Remote Sens. Symp., 2002, pp. 2253–2255.

[26] T. Bretschneider and O. Kao, “A retrieval system for remotely sensed imagery,” in Proc. Int. Conf. Imag. Sci., Syst., Technol., 2002, vol. 2, pp. 439–445.

[27] G. Scott, M. Klaric, C. Davis, and C.-R. Shyu, “Entropy-balanced bitmap tree for shape-based object retrieval from large-scale satellite imagery databases,” IEEE Trans. Geosci. Remote Sens., vol. 49, no. 5, pp. 1603– 1616, May 2011.

[28] A. Ma and I. K. Sethi, “Local shape association based retrieval of infrared satellite images,” in Proc. IEEE Int. Symp. Multimedia, 2005, pp. 551–557.

[29] M. Ferecatu and N. Boujemaa, “Interactive remote-sensing image retrieval using active relevance feedback,” IEEE Trans. Geosci. Remote Sens., vol. 45, no. 4, pp. 818–826, Apr. 2007.

[30] B. Raghunathan and S. Acton, “Content based retrieval for remotely sensed imagery,” in Proc. IEEE Southwest Symp. Image Anal. Interpretation, 2000, pp. 161–165.