5 Use of Digital Signal Processing in the textile field

by
Nicolas-Alexander Tatlas

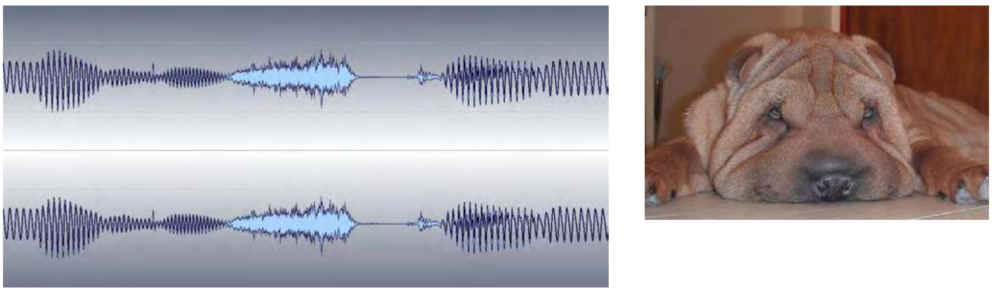
Department of Electronics Engineering, Technological Education Institute of Piraeus, Greece

5.1 Introduction

Digital Signal Processing (often called DSP) has been rapidly evolving during the past years, covering a growing list of application domains, from factory automations to special effects in movies. The DSP wide adoption and use is mainly due to the development of microelectronic systems with greater computational power, smaller size, restricted power requirements and decreasing cost. The ever-increasing capabilities of hardware allow complex algorithms to be executed in real time even on mobile smart-phone type devices. The aim of this chapter is to provide the reader with an insight on basic principles, algorithms and techniques of signals, digitization and digital processing as well as its use in the area of textile production.

5.2 Signals and Digitization

A signal is defined as any physical quantity that varies with time, space or any other independed variables. In other words, a signal is a sequence or function depending on one or more variable, containing information about the property. Examples of everyday signals are speech, as a function of sound pressure over time, the natural light deviation, as a function of sun illumination over time or the decrease in temperature when climbing a mountain as a function of temperature over height.



(a) (b)

Figure 5.1: (a) One-dimensional signal (sound) (b) Two-dimensional signal (image).

Signals can be divided in different categories depending on the number and type of their variables. Depending on the number of independent variables signals can be of one dimension (one variable), two dimensions (two variables) and so on. An example of a one dimensional signal is music, where the one independent variable is time and the depended variable is sound pressure. A black-and-white picture is a typical example of a two dimensional signal, where the depended variable is luminance and the independent variables are X and Y coordinates of the picture. Figure 5.1 shows an example of a one dimensional and a two dimensional signal.

If the independent variable is continuous the signal is called a continuous signal whilst if the independent variable is discrete the signal is called a discrete signal. Usually the independent variable is time, so the classification is between continuous-time or analogue signals that are defined for every time value and discrete-time signals that are defined only at certain time values. The process of converting a continuous-time signal to a discrete-time signal is *sampling.* Figure 5.2 shows an example of a continuous and a discrete time version of the same sine-wave signal, where T_s is the sampling period, inversely proportional to the sampling frequency f_s.

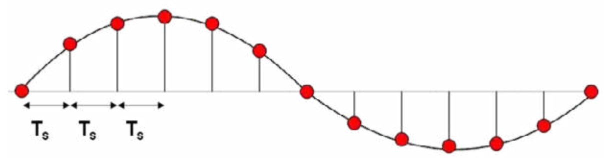


Figure 5.2: Continuous (black line) and discrete (red dots) time sine-wave signal.

According to Nyquist-Shannon sampling theorem, in order to accurately describe the original signal, the sampling frequency of the discrete signal must be at least twice the maximum frequency (f_{max}) contained in the continuous-time signal.

$$f_s \geq f_{max} \tag{Eq. 5.1}$$

Moreover, the actual values of a signal can be continuous, taking any possible values or discrete, taking values from a predetermined set. The conversion of a continuous to a discrete value signal is performed through quantization of the original values; this is done through some kind of approximation, such as truncation. Figure 5.3 shows a continuous (black line) and discrete (red line) value signal. The quantized signal in this example can have one of eight allowable values, while the continuous signal can have any permissible value. The distance between two consecutive discrete values is defined as the quantization step and is usually represented as Δ.

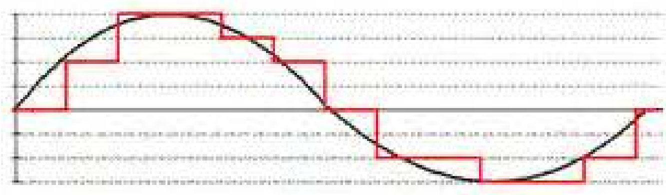


Figure 5.3: Continuous (black line) and discrete (red line) value sine-wave signal.

The error introduced by mapping continuous valued signals through a finite set of discrete values is called quantization error or quantization noise. The process of quantizing an analogue signal leads to information loss, because of the fuzziness introduced. Any sample that lies in an area Δ/2 around a permissible quantization level is represented by the same value; the quantization error is generally a characteristic of the number of quantization levels L_q.

The final step of the quantization process in order to obtain a digital signal in the sample coding, where a unique binary number is assigned to every quantization level. Given that a binary word of length N can describe 2^N numbers, the number of coding bits N must satisfy the following Eq.:

$$2^N \geq L \qquad \text{(Eq. 5.2)}$$

For example, the quantization process shown in Figure 5.3 can be coded using 3 bits describing 8 values. An approximation of the signal-to-noise ratio for a sine wave quantized signal depending on the sample bit depth is given by:

$$SNR = 6N + 1.76dB \qquad \text{(Eq. 5.3)}$$

Audio in music CDs is digitized with f_s=44,1KHz and N=16bits per sample. The sampling rate and bit depth selection is a trade-off between the signals accurate representation and the requirement for increased storage and processing, as explained in the following chapters.

5.3 Signal processing basics

Signal processing may refer to the analysis, or modification or other operation on a signal usually in order to extract information, such as a property. For example, the average luminance signal can be extracted from the images sensor of a digital camera and evaluated in order to decide if a flash light is necessary and determine its properties. Another example is the volume adjustment on any mobile phone: The incoming voice signal is either amplified or attenuated in order to meet the user requirements. Both the above examples have a common element, a device that is responsible for the processing: In DSP context, a *system* may be defined as the apparatus to perform any operation on a signal. The system may be mechanical, electrical, microelectronic or even software running on a computer. *Digital signal processing* refers to processing on digital signals, i.e. discrete time and discrete value signals performed by a digital system, realized in hardware or in software.

5.3.1 Digital vs Analogue Signal Processing

Electronic analogue signal processing was employed for decades, using circuit elements such as resistors, capacitors, diodes, transistors etc. The ability of analogue systems to provide functions such as derivatives, integrals as well as to solve differential equations was exploited in such systems. On the contrary, digital signal processing relies on numerical calculation. There are numerous of advantages in using DSP, such as:

(i) A digital processing system is extremely flexible. A software modification can radically change its operation. An analogue system would likely require to be redesigned.

(ii) The accuracy of a digital system is far greater. Even a slight tolerance on circuit elements induces an uncertainty towards the system precision, while digital systems accuracy is defined in a straightforward manner. Moreover, circuit element tolerance may be time-varying because of temperature fluctuations or aging.

(iii) Digital Signals can be stored and copied in magnetic, optical or solid state media without the possibility of degradation, offering the possibility of non-real time processing.

(iv) Complex processing algorithms requiring extremely high numerical precision may be realized in software.

(v) Usually the digital processing of a signal is cost-effective compared to the analogue processing, because of the decreasing prices of hardware and the restricted development effort required.

Of course, there are boundaries in digital signal processing, set by the restrictions in analogue-to-digital converter operating frequencies as well as the digital processors instruction speed capabilities. Thus, high bandwidth signals such as satellite television signals are usually pre-processed in the analogue domain.

5.3.2 Basic DSP system

Figure 5.4 shows a typical block diagram for a DSP system. As all natural signals are analogue, usually such a systems first part is an analogue to digital (A/D) converter, including a sampling and a quantization stage, as previously explained. The A/D output is a digital signal adequately describing the original signal attributes and appropriate as input to the Digital Processing stage. The processing can be performed using (a) Application specific hardware systems, or (b) Digital Signal Processor-based systems or (c) Generic computer systems, as well as systems that combine the above.

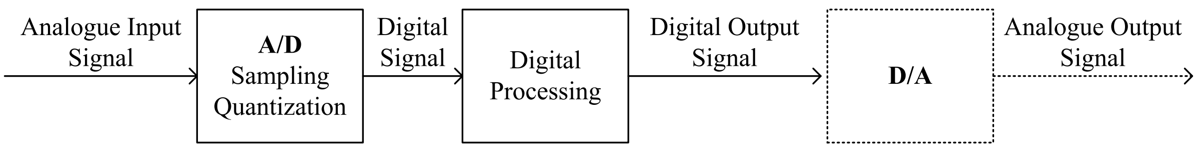


Figure 5.4: DSP system block diagram.

A hardwired implementation may be optimized resulting in lower costs and increased computational efficiency. On the other hand a software-based implementation usually requires less development time and is easily reconfigurable. Some applications require the output of the processing system to be given in an analogue form, such as audio signals, requiring an additional stage, that of a Digital to Analogue converter. However, in most cases this is not required, such as image processing where both static images and video are digitally captured.

5.4 Discrete Time Signals & Systems

It is obvious that DSP systems operate on discrete and quantized signals. In order to proceed, a short introduction on basic concepts regarding discrete signals and systems is presented in this paragraph. The following discrete signals are considered elementary:

a) Unit impulse sequence

$$\delta[n] = \begin{cases} 1, n = 0 \\ 0, n \neq 0 \end{cases} \quad \text{(Eq. 5.4)}$$

b) Unit step sequence

$$u[n] = \begin{cases} 1, n \geq 0 \\ 0, n < 0 \end{cases} \quad \text{(Eq. 5.5)}$$

c) Constant sequence

$$x[n] = A \quad \text{(Eq. 5.6)}$$

d) Linear sequence

$$x[n] = An \quad \text{(Eq. 5.7)}$$

e) Exponential sequence

$$x[n] = A^n \quad \text{(Eq. 5.8)}$$

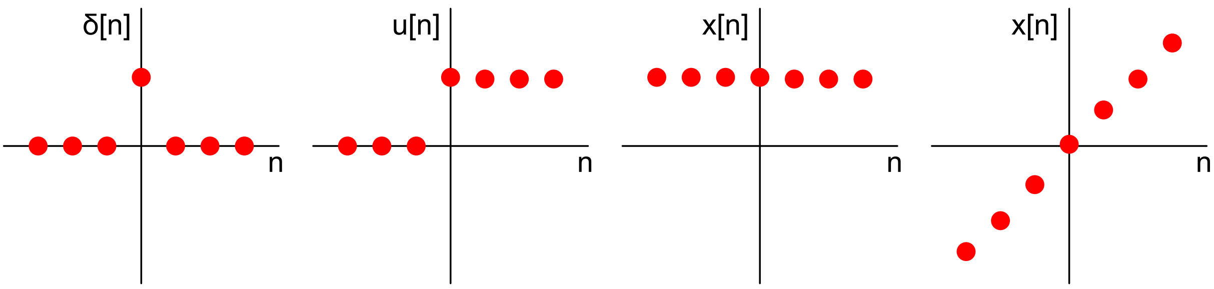


Figure 5.5: Elementary signals, (a) unit impulse, (b) unit step (c) constant and (d) linear sequences.

5.4.1 Basic Functions

Transformation of the independent variable

A signal s[n] may be time shifted by replacing the independent variable n with n-k. If k is positive a delay is introduced, while if negative the signal is advanced in time. Note that only a previously stored signal can be advanced in time. For example, a delayed unit step sequence would be:

$$u[n-k] = \begin{cases} 1, n \geq k \\ 0, n < k \end{cases} \quad \text{(Eq. 5.9)}$$

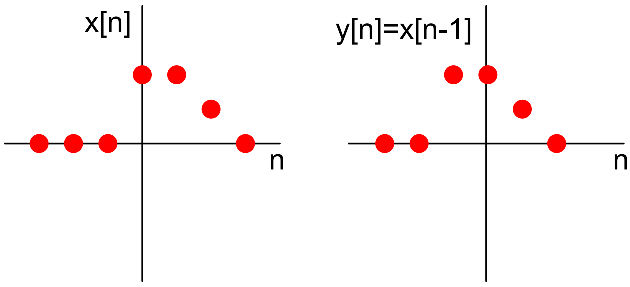


Figure 5.6: Example of signal delay.

Any discrete signal can be written as a sum of gain-adjusted and time-shifted impulse signals:

$$y[n] = \sum_{k=-\infty}^{+\infty} x(k)\,\delta[n-k] \quad \text{(Eq. 5.10)}$$

Addition, multiplication and scaling

Amplitude scaling, meaning attenuation or amplification of signal by a constant is accomplished by multiplying every sample with that constant:

$$y[n] = Ax[n] \quad \text{(Eq. 5.11)}$$

The sum of two signals is given by the sums of their values sample-by-sample:

$$y[n] = x_1[n] + x_2[n] \quad \text{(Eq. 5.12)}$$

For example, mixing two different audio signals requires their addition according to the following equation. If the gain needs to be adjusted then each signal is first multiplied, then both are added.

Finally, the product of two signals is given by the product of their values sample-by-sample:

$$y[n] = x_1[n]x_2[n] \quad \text{(Eq. 5.13)}$$

5.4.2 Discrete-time systems

A discrete-time system is a system that accepts as input and produces according to a well defined set of rules as output discrete signals x[n] and y[n] respectively. The systems that are analyzed in this chapter have two fundamental features: they are Linear and Time-Invariant, and are thus named LTI systems. A *linear* system satisfies the superposition principle, meaning that the output of a system to a sum of signals is equal to the sum of the outputs for each individual signal, given that the signal is in a relaxed state. *Time-invariant* is a system whose behavior and properties remain constant over time.

Moreover, a system is *causal* if its output at any instance is a function only of the present and past inputs, thus does not depend on future inputs. Finally a system is *stable* if and only if every bounded input produces a bounded output.

If a LTI system in a relaxed state is stimulated with a unit impulse signal, its output characterizes the system; this output is the system *impulse response* h[n]. Given the impulse response of a LTI system, its output y[n] for any input x[n] is found through the convolution operation, symbolized as "*":

$$y[n] = h[n] * x[n] \quad \text{(Eq. 5.14)}$$

Given that a signal can be expressed as the sum of time shifted impulse signals, and that we are dealing with LTI systems, the previous equation can be also written as:

$$y[n] = \sum_{k=-\infty}^{+\infty} x(k)\, h[n-k] \quad \text{(Eq. 5.15)}$$

which is the description of linear convolution. Note that if K_1 is the length of one sequence and K_2 the length of a second sequence, the result of their convolution is of K_1+K_2-1 length.

5.4.3 Discrete-time system structures

The duration of the impulse response leads to two categories of LTI systems: an impulse that has a finite number of non-zero samples characterizes a Finite Impulse Response or FIR system; otherwise, the impulse belongs to an Infinite Impulse Response or IIR system. It is evident that the computation of the convolution of a signal with an infinite impulse requires an infinite number of operations. However, IIR systems can be practically implemented, using previous output values or recursive implementations.

5.4.4 Implementation of discrete-time systems

The implementation of discrete-time systems requires two block categories, (a) arithmetic units to perform basic calculations such as Eq. 5.11–Eq. 5.13, and (b) memory in order to store values, such as needed to implement Eq. 5.9. The usually employed blocks are four basic discrete-time systems shown in Figure 5.7. If the output of a system depends only on the input at the same instance, the system is called memoryless; otherwise it is called dynamic. However, the concept of memoryless systems should not be confused with the requirement for memory in an implementation, as memory is usually necessary to store intermediate values during any processing.

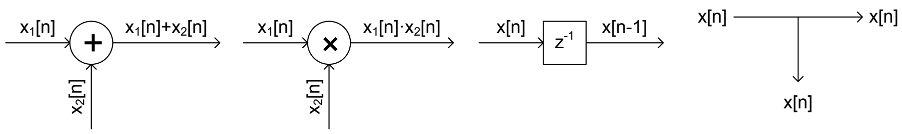


Figure 5.7: Basic implementation blocks (a) adder (b) multiplier (c) delay (d) splitter.

Block diagrams are commonly used for the implementation representation of discrete-time systems, used both for hardware and software implementation.

5.4.5 Frequency analysis

Analysis in the frequency domain of discreet time signals and systems provides an alternative regarding their design and implementation. In order to map a time series to a frequency domain series, a transition from the time to the frequency domain is needed. Periodical continuous signals can be analyzed in a series of sinusoidal signals (Fourier Series) or using other terms in an infinite number of discrete frequency spectrum components. If the signal is non -periodical, the expansions concept in the frequency domain is generalized and the Fourier Series is substituted by the Fourier Transform resulting in a continuous frequency spectrum. The Fourier Transform and the Inverse Fourier Transform are powerful tools, allowing the transition between the time domain and the frequency domain, without loss of the signal's information content. Especially when the signal is periodical and discrete the frequency spectrum is also periodical and discrete. In that case the transition is obtained by the Discrete Fourier Transform (DFT). The DFT for a finite sequence x[n] is given by:

$$X[k] = \sum_{n=0}^{N-1} x\,[n] e^{\frac{-j2\pi kn}{N}}, k = 0,1 \ldots, N-1 \qquad \text{(Eq. 5.16)}$$

The inverse operation, or IDFT is given by:

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X\,[k] e^{\frac{j2\pi kn}{N}}, n = 0,1 \ldots, N-1 \qquad \text{(Eq. 5.17)}$$

Direct computation of the N-point DFT requires computational cost proportional to N^2. The most important class of efficient DFT algorithms, known as the Fast Fourier Transform (FFT) algorithms, provide all DFT coefficients with a computational cost proportional to $N\log_2 N$.

5.5 Digital Image Processing

As mentioned in the introduction of this chapter, a digital image is two-dimensional signal. A black and white image can be expressed as a function of the luminosity for each x,y coordinate.

$$Luminosity = f(x,y) \qquad \text{(Eq. 5.18)}$$

What is commonly named a "black and white" image is actually a grayscale image, having a number of different luminosity values. Larger values usually correspond to whiter areas whereas smaller values indicate darker parts of the image. Figure 5.8 shows the luminosity signal for the image in Figure 5.1(b). For example, the white spot from the flash reflection on the left part of the image is manifested as a peak in luminosity values.

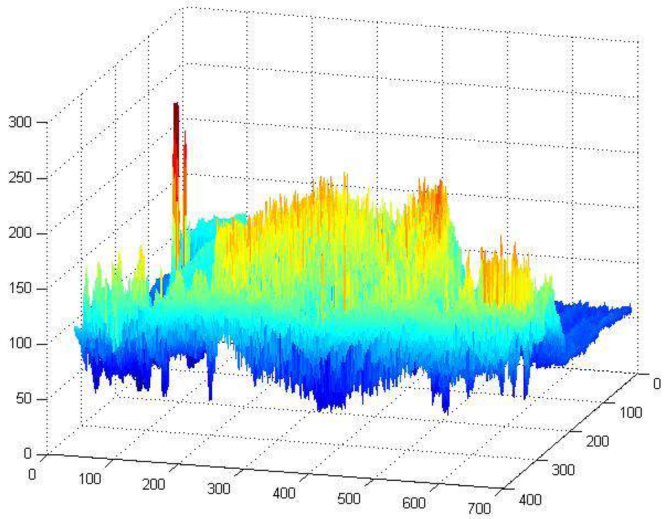


Figure 5.8: Luminosity representation.

The image digitization process does not differ from what was described for one dimensional signals. Equally distanced samples in the x,y space are obtained with the maximum distance of two sequential samples needs to be less than half the period of the faster changes in f(x,y). The samples are then quantized and coded, usually with N=8bits. The digital image samples are called picture elements or "pixels". Color images are represented in the same way; however three values are coded for each pixel, corresponding to the "amount" of each basic color, Red, Green and Blue.

An image that is of length N_1 and width N_2 contains $N_1 \cdot N_2$ pixels. The total space S (or memory) needed to store a color image is given by the following Eq:

$$S = 3NN_1N_2(bits) \quad \text{(Eq. 5.19)}$$

It is obvious that increasing the resolution or bits per pixel allocation in an image (a) requires more space and (b) requires greater computational effort to process.

A number of stages are generally included in order to process a digital image, depending on the format and condition of the original picture as well as the required output. Figure 5.9 shows a general block diagram for an image dsp system. The digital image is captured or digitized from its analogue form in the acquisition stage. Depending on the image state, noise reduction or other algorithms to invert possible distortions are employed. The optimized picture is analyzed depending on the systems goal, such as to extract specific features from an image, such as boundary position, or to recognize objects, such as automobiles. Image compression in order to save storage space or increase transmission speed may be employed as a final stage.

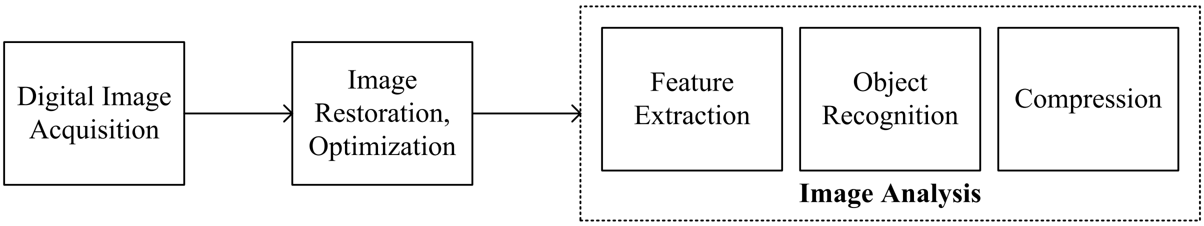


Figure 5.9: Image DSP general block diagram.

5.5.1 Image processing functions

An image processing stage may have as input or output either an image or a mathematical description. For example, any image compression software receives an image and outputs another image, while a Computer Aided Design (CAD) package is given coordinates and directives and outputs an image. The following analysis on image processing applies to algorithms that have both as input and output images. Functions in digital image processing provide values for output pixels based on the input pixels. Depending on the algorithm employed, these functions may be (a) Local, (b) Global and (c) Geometrical.

Local functions calculate the value of each output pixel with coordinates [n1,n2] using the value from the input image with the same coordinates as well as neighboring values. A neighborhood in an image is an area, usually a square of fixed size, centered on a pixel. A common local function is averaging where each pixel is results from an average of the input neighborhood pixel values. If the neighborhood of each pixel in the original image is not taken into account then the function is called a *point operation*. For example, point operations are Contrast adjustment, performed by multiplying each pixel value with a constant and Luminosity adjustment, performed by adding (or subtracting) each pixel value with a constant. Figure 5.10 shows an example for averaging, using a 9×9 neighborhood, contrast stretching using a constant of 3 and luminosity increase using a constant of 70.

(a) (b) (c)

Figure 5.10: (a) Averaging (b) Contrast and (c) Luminosity adjustment.

Global functions utilize values from the whole original image to calculate each pixel value. These are usually transformation functions or adjustments, requiring for example knowledge of the maximum or minimum value encountered in the original image.

Geometrical functions apply to coordinate modifications, such as relocation, rotation and mirroring. These functions usually provide images of the same area with their original pixel values not altered, but moved to other positions.

5.6 DSP in textile quality control

A widespread use of DSP systems in textile production is that of quality control of woven fabric. The aim is to detect defects during the process. Generally, there are two types of defects in textiles, (a) structural, usually caused from faults during the weaving process and (b) tonal defects based on the presence of pollutants or a temporary error in the coloring process. However, a number of other defects can present, adding complexity to the detection procedure.

The general architecture for an automatic textile quality control setup in shown in Figure 5.11, including a digital camera and a motor/position sensor for the fabric conveyer connected through appropriate interfaces to a computer, running software for image acquisition and processing.

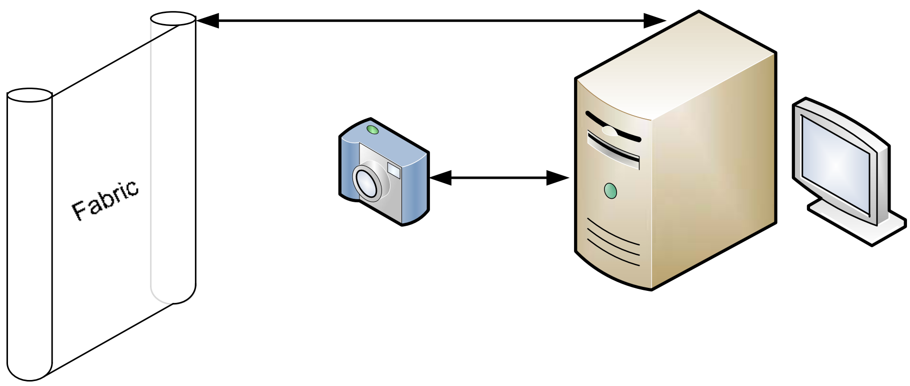


Figure 5.11: Automatic textile quality control system.

The fabric under examination is positioned between two axles adequately spaced to appropriately position part of the material for the digital camera to capture its image. The position of the fabric is controlled through the PC in order to synchronize each image obtained; a series of images is then automatically merged providing a digitized segment of the fabric for further processing. Upon the detection of a fault, an error message is created usually including a snapshot of the defect as well as the corresponding position of the axles.

Generally, there are two stages for operating such a processing system, (a) the training phase and (b) the recognition phase. During the training phase, the system is operated using a flawless fabric, in order to use the obtained images as reference. These images are first pre-processed with DSP algorithms that ensure distortion suppression, such as uneven brightness or contrast, following the extraction of specific textile features such as the color distribution, fiber unity etc. The same pre-processing and feature extraction stages are applied in normal operation; the system detects errors by comparing these features to the flawless set from the training phase. The challenge while operating such a system is (a) to select appropriate texture analysis features correlated to the detection of specific flaws (b) to fine-tune the process by introducing thresholds in order to minimize false positive events and maximize the detection capability.

The leading approaches to texture analysis are statistical and structural methods based on spatial frequencies. Statistical approaches are based on the analysis and characterization of patterns observed using statistical features, providing an overall features. The placement of textural sub-patterns, according to rules form the texture is the basis for structural methods, giving a detailed description of the surface under test. Moreover, the detection of flaws in texture images can be performed in two fundamental ways: The first is block-based processing method, where each block is compared to an error-free sample, as mentioned earlier. The second is to employ metrics based on whole images, requiring though a reference for every possible fault.

There are numerous specific techniques and algorithms that have been tested or are being employed in such systems, such as the Spatial Grey Level Co-occurrence (SGLC) and Cross correlation. The SGLC method utilizes matrices based on the second order probability for two pixels of different gray levels of given displacement to appear in an image. The method implementation requires the computation of the optimal image displacement vectors in terms of flaw detection capability, deeming it susceptible to errors from alignment mismatch of dimensional changes. Cross-correlation as a metric of similarity provides a clear way of detecting differences between the reference and inspected image, indicating a flaw. While computational intensive, in theory cross-correlation may detect any possible fabric defect as long as the test conditions are constant and/or pre-processing equalization stages are implemented. The reader is referred to (Bennamoun & Bodnarova, 2003) and (Kosek. 2005) for more information on detection methods as well as their efficacy and robustness.

5.7 References

J. Proakis & D. Manolakis, 1996, “Digital Signal Processing Principles, Algorithms and Applications”, 3^{d} edition Prentice Hall International, INC.

A.N. Skodras & B. Anastasopoulos, 2000, “Introduction to Digital Signal and Image Processing”, University of Patras (in Greek).

D. Manolakis & V. Ingle, 2011, "Applied Digital Signal Processing", Cambridge University Press.

M. Bennamoun & A. Bodnarova, 2003, "Digital image processing techniques for automatic textile quality control", Systems Analysis Modelling Simulation, 43:11, 1581–1614.

Miloslav Kosek. 2005, "Efficient application of signal processing methods in textile science". Proceedings of the 4th WSEAS international conference on Applications of electrical engineering (AEE'05), p. 155–159.