Camera Based Product Counting of Belt Conveyors

M.Uğur PARLAK[1]

ABSTRACT

This work involves the development of a vision-based system for counting the numberof products that pass on a conveyor belt. The system has applications in automatic control and optimization of industrial processes that involve belt conveyors and their related packaging operations. Determining automatically the number of products to be packaged without the need for additional hardware setup yields an important reducement in the initial cost of the complete installation. In this work, by introducing a vision based approach, thus with involvement of image and video processing techniques, counting of products passing on a belt conveyor system just using a camera is investigated.

Keywords: *Matlab, Conveyor Belt, Image Processing, Simuling, Image Processing Toolbox, Camera Based Product Counting On Belt Conveyors*

1. INTRODUCTION

The basis of this project is to apply computer vision techniques to develop a program which should recognize and count products passing on a conveyor belt image frames, and enumerate their value. That is to have a computer "watch" the image and then tell the user the total number of the products which are on the image, therefore to be packaged in the next step.

2. CONVEYOR BELT SYSTEMS:

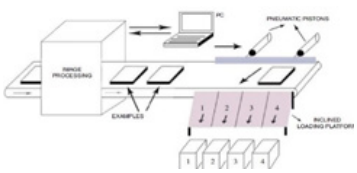

Figure 1.1 *A Generic Conveyor Belt system*

A conveyor belt is a mechanical system that is used to move products from one location to another. Conveyor belt is a continuous loop that is mostly used by almost all factories that make serial production.

As it is known, conveyor belt system is used for efficient transportation of products from one location to another. Therefore, transportation of product becomes faster, easier and safer. Therefore, it is important to decrease cost of production. Aim of conveyor belt that is used in the study is to transfer material from one location to another and also it is used to sort products according to height and shape products.

[1] *Mechatronics Engineering Department İstanbul Aydın University*

3. IMAGE PROCESSING PART

Image processing techniques or in other words computer vision refers to the application of human vision techniques to a computer, teaching the computer to see. The subject itself has been around since the 1960s, but it is only recently that it has been possible to build useful computer systems using ideas from computer vision. This subject is driven by three main areas: computational geometry, artificial intelligence, and image processing (Forsyth 2003).

Computational geometry now is widely used in every corner of science and engineering, from design and manufacturing to astrophysics, molecular biology and fluid dynamics. To build 3D computer models, lots of problems can be solved in these areas.

In artificial intelligence field, people use computer vision technique as a tool to involve both the acquisition and processing of visual information. For instance, recently some companies and research groups focus on face recognition technique, which is widely used in virtual reality, national ID, security trading terminal, CCTV control and etc (Zhao 2003).

A computational vison – image processing system executes the following processes in the specified order:

- Image capture and enhancement
- Segmentation
- Feature extraction
- Matching features to models
- Exploitation of constraints and image cues to recover information lost during image processing,
- Application of domain knowledge to recognize objects in the scene and their attributes.

Image processing is the base of the other two areas. It refers to processing digital images by means of a digital computer. We know that 80% of the information that we get from outside world are captured by the vision, so it is not surprising that images play the single most important role in human perception. However, humans are limited to the visual band of the electromagnetic spectrum, such as ultrasound, electron microscopy. Therefore, we need the imaging machines which cover almost the entire electromagnetic spectrum, ranging from gamma to radio waves. They can operate on images generated by sources that human are not accustomed to associating with image. Thus, digital image processing is applied a wide and varied fields (Gonzalez 2002, p.1-2). For example, in medical imaging, it can be used to enhance imagery, or identify important phenomena or events.

Once the image is acquired in digital form, it is processed using digital image processing routines with the final goal of image segmentation. The initial processing is usually a filtering operation to reduce the noise which is introduced by the image formation process. This process is noisy due to sampling, quantization and random disturbances in the capture hardware.

Filtering has its own associated degradations such as blurring the edges of objects. At this point the filtered image is then segmented into disjoint regions using a possible combination of edge detectors, thresholding techniques, morphological operations and other image processing transforms. This form of processing can be classified as low-level processing where the objects of interest are revealed. There is no object recognition or feature-based classification of regions.

4. MATLAB AND SIMULINK

Introduction

MATLAB is a high-performance language for technical computing created by The MathWorks in 1984. It integrates computation, visualization, and programming in an easyto-use environment where problems and solutions are expressed in familiar mathematical notation.

Typical uses include:

- Mathematics 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

MATLAB has evolved over a period of years with input from many users. In university environments, it is the standard instructional tool for introductory and advanced courses in mathematics, engineering, and science. In industry, MATLAB is the tool of choice for high-productivity research, development, and analysis. [1]

4.1. HISTORY OF MATLAB

MATLAB was invented in the late 1970s by Cleve Moler, then chairman of the computer science department at the University of New Mexico. He designed it to give his student's access to LINPACK and EISPACK without having to learn Fortran. It soon spread to other universities and found a strong audience within the applied mathematics community.

Jack Little, an engineer, was exposed to it during a visit Moler made to Stanford University in 1983. Recognizing its commercial potential, he joined with Moler and Steve Bangert. They rewrote MATLAB in C and founded The MathWorks in 1984 to continue its development. These rewritten libraries were known as JACKPAC.

MATLAB was first adopted by control design engineers, Little's specialty, but quickly spread to many other domains. It is now also used in education, in particular the teaching of linear algebra and numerical analysis, and is popular amongst scientists involved with image processing.

4.2. What Is MATLAB

MATLAB is a numerical computing environment and programming language. It allows easy matrix manipulation, plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs in other languages. Although it specializes in numerical computing, an optional toolbox interfaces with the Maple symbolic engine, allows it to be part of a full computer algebra system. Besides dealing with explicit matrices in linear algebra, it can handle differential equations, polynomials, signal processing, and other applications. Results can be made available both numerically and as excellent graphics.

MATLAB solves many technical computing problems, especially those with matrix and vector formulations, in a fraction of the time it would take to write a program in a scalar non-interactive language such as C or Fortran. The name MATLAB stands for Matrix Laboratory. MATLAB was originally written to provide easy access to matrix software developed by the LINPACK and EISPACK projects. Today, MATLAB engines incorporate the LAPACK and BLAS libraries, embedding the state of the art in software for matrix computation.

MATLAB features a family of add-on application-specific solutions called Toolboxes. Toolboxes allow learning and applying specialized technology. Toolboxes are comprehensive collections of MATLAB functions (M-files) that extend the MATLAB environment to solve particular classes of problems. Areas in which toolboxes are available include signal processing, control systems, neural networks, fuzzy logic, wavelets, simulation, and many others.

As of 2004, MATLAB was reported to be used by more than one million people in industry and academia.

4.3. WHAT IS SIMULINK

Simulink is a software bundled with MATLAB for modeling, simulating, and analyzing dynamic systems. It supports linear and nonlinear systems, modeled in continuous time, sampled time, or a hybrid of the two. Systems can also be multi-rate, i.e., have different parts that are sampled or updated at different rates.

Simulink enables users to pose a question about a system, model it, and see what happens. With Simulink, models can be built easily from scratch, existing models can be taken and be added to it. Thousands of engineers around the world use Simulink to model and solve real problems in a variety of industries.

5. IMAGE PROCESSING TOOLBOX

Image processing is any form of signal processing, where the input is an image, such as a photograph or video frame; the output may be either an image or a set of characteristics or parameters related to the image. Most image-processing techniques involve treating the image as a two-dimensional array and applying standard signal-processing techniques to it.

Application areas behind the interest in digital image processing methods are so varied and can be listed into two main categories (namely, improvement of pictorial information for human interpretation; and processing of image data for storage, transmission, and representation for machine vision.). Space applications, medical imaging, remote earth resources observations, and astronomy are examples of those applications, where improvement of pictorial information for human interpretation is apparent. Studying pollution patterns from aerial and satellite imagery, image enhancement and restoration procedures to process degraded images of unrecoverable objects are two further applications that can basically serve geographers and archeologists respectively. In machine vision applications, interest focuses on procedures for extracting information from an image in a form suitable for computer processing.

The digital image processing involves a number of fundamental steps (e.g. image acquisition, image enhancement and preprocessing, edge detection and segmentation, representation and description, and matching and recognition). The output of these steps is either an image or an image attribute.

An important characteristic in the design of systems is the level of testing and experimentation that is normally required before arriving at an acceptable solution and hence obtaining a feasible system implementation. This characteristic is applied to the field of digital image processing, where extensive experimental work involving software simulation and testing with large

sets of sample images is generally required to offer solutions to problems.

6. FUNDAMENTAL STEPS IN IMAGE PROCESSING

The field of digital image processing is referring to processing digital images using a digital computer. Whereas, a digital image is composed of a finite number of elements, each of which has a particular location and value. These elements are referred to as picture elements, image elements, or pixels. Before going to processing an image, it is converted into a digital form. Among the many and variant possible processing and analysis steps performed on a digital image, we, in this paper, are more interested in image edge detection and segmentation, and image matching and recognition processes.

Image Enhancement: Processing an image so that the result is more suitable for a particular application. (sharpening or deblurring 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. (removing of blur caused by linear motion, removal of optical distortions)

Image Segmentation: This involves subdividing an image into constituent parts, or isolating certain aspects of an image. (finding lines, circles, or particular shapes in an image, in an aerial photograph, identifying cars, trees, buildings, or roads.

Edge Detection: Edge detection is a type of image segmentation techniques, whose main task is to determine the presence of an edge or a line in an image and outlines them in an appropriate way. Prewitt, Sobel, and Canny have developed other edge detector operators, whose mask operators for both x, and y axes are tabulated in Table I. The mask operators are [3x3] arrays (matrices).

Name of the operator	Mask for x-axis			Mask for y-axis		
Prewitt	-1	0	1	1	1	1
	-1	0	1	0	0	0
	-1	0	1	-1	-1	-1
Sobel	-1	0	1	-1	-2	-1
	-2	0	2	0	0	0
	-1	0	2	1	2	1
Canny	1	2	1	-1	0	1
	0	0	0	-2	0	2
	-1	-2	-1	-1	0	1

Mask operators for three edge detecting operators

Image Matching: The matching process is refereeing to finding a correspondence between various data sets. The data sets can represent images, photographs, maps or any other form of object model. Matching has always been a challenging problem in the area of image research and development. Some factors, which can pose problems in image matching includes, but not limited to; changes in the image content, plane rotation, change in scale, change in illumination, and differences caused by electronic noise.

The basic principle of matching is to search through all the pixels for the right area which is identical to a given template image. However, because images are normally having huge amount of pixels, it's not realistic from the performance point of view to apply a complete search in the image space. For this specific reason, proposed image matching algorithms are improved to reduce searching time and having an optimized performance. Another fact that need to be considered as

well is existence of various number of image matching algorithms tailored to suit specific applications such that no general algorithm is available that is optimized for all variety of uses. Image matching algorithms are categorized as; area based, feature based, transformation model, direct methods, spatial domain methods, frequency domain methods, and image nature based methods. For the experimental and simulation purposes, in this paper, a relatively simple matching algorithm is proposed and adopted.

A condition is that both images are of the same dimensions. If all pixels are found identical then input images are considered matched. The algorithm however, may accept some differences between corresponding pixels and still consider a matching decision for input images, if the user fixes a threshold value for such acceptable differences

7. TYPES OF DIGITAL IMAGES

- **Binary:** Each pixel is just black or white. Since there are only two possible values for each pixel (0,1), we only need **one bit** per pixel.

- **Grayscale:** Each pixel is a shade of gray, normally from **0** (black) to **255** (white). This range means that each pixel can be represented by **eight bits**, or exactly **one byte**. Other greyscale ranges are used, but generally they are a power of **2**.

- **True Color**, or **RGB**: Each pixel has a particular color; that color is described by the amount of **red, green** and **blue** in it. If each of these components has a range 0–255, this gives a total of **2563** different possible colors. Such an image is a "stack" of **three matrices**; representing the **red, green** and **blue** values for each pixel. This means that for every pixel there correspond 3 values.

8. MATLAB AND DIGITAL IMAGE PROCESSING

MATLAB is a high-performance language for technical computing. It integrates computation, visualization, and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation. The basic data element in MATLAB, which is an interactive system, is a matrix. This allows finding solutions to many technical computing problems, especially those involving matrix representations, in a relatively short time compared to the time required to write a program in a scalar non-interactive language such as C.

MATLAB is complemented by a family of application-specific solutions called toolboxes. The Image Processing Toolbox is a collection of MATLAB functions (called M-functions or M-files) that extend the capability of the MATLAB environment for the solution of digital image processing problems. The toolbox supports four types of images: (Grayscale, Binary, Indexed, and RGB images). The toolbox provides specific functions that perform converting images from one class to another.

Edges of an image are considered a type of crucial information that can be extracted by applying detectors with different techniques. The Image Processing Toolbox includes a group of edge detectors through its edge function. This functionality allows one to specify any of the derivative (gradient) filters discussed in the preceding section. The edge function accepts an intensity image and

returns a MATLAB binary image, where pixel values of 1 indicate where the detector located an edge and 0 otherwise.

General Commands:

- **imread:** Read an image
- **figure:** creates a figure on the screen.
- **imshow(g):** which displays the matrix g as an image.
- **pixval on:** turns on the pixel values in our figure.
- **impixel(i,j):** the command returns the value of the pixel (i,j)
- **iminfo:** Information about the image.

Arithmetic Operations:

These operations act by applying a simple function y=f(x) to each gray value in the image.

- Simple functions include **adding** or **subtract** a constant value to each pixel: y = x±C
- (imadd, imsubtract)
- **Multiplying** each pixel by a constant: y = C•x (immultiply, imdivide)
- **Complement:** For a grayscale image is its photographic negative.

Histograms:

- Given a grayscale image, its histogram consists of the histogram of its gray levels; that is, a graph indicating the number of times each gray level occurs in the image.
- We can infer a great deal about the appearance of an image from its histogram.
- In a **dark** image, the gray levels would be clustered at the lower end
- In a **uniformly bright** image, the gray levels would be clustered at the upper end.
- In a **well contrasted** image, the gray levels would be well spread out over much of the range.
- **Problem:** Given a poorly contrasted image, we would like to enhance its contrast, by spreading out its histogram. There are two ways of doing this.

Thresholding:

- **Single thresholding:** A grayscale image is turned into a binary image by first choosing a gray level **T** in the original image, and then turning every pixel black or white according to whether its gray value is greater than or less than **T.**
- A pixel becomes white if its gray level is > **T**
- A pixel becomes black if its gray level is <= **T**
- Double thresholding: Here we choose two values T1 and T2 and apply a thresholding operation as:
- A pixel becomes white if its gray level between **T1** and **T2**
- A pixel becomes black if its gray level is otherwise

Color Images:

- A **color model** is a method for specifying colors in some standard way. It generally consists of a 3D coordinate system and a subspace of that system in which each color is represented by a single point.
- **RGB:** In this model, each color is represented as 3 values **R**, **G** and **B**, indicating the amounts of red, green and blue which make up the color.
- **HSV:**
- Hue: The "true color" attribute (red, green, blue, orange, yellow, and so on).
- Saturation: The amount by which the color as been diluted with white. The more white in the color, the lower the saturation.

Value: The degree of brightness: a well lit color has high intensity; a dark color has low intensity.

9. CONCLUSION

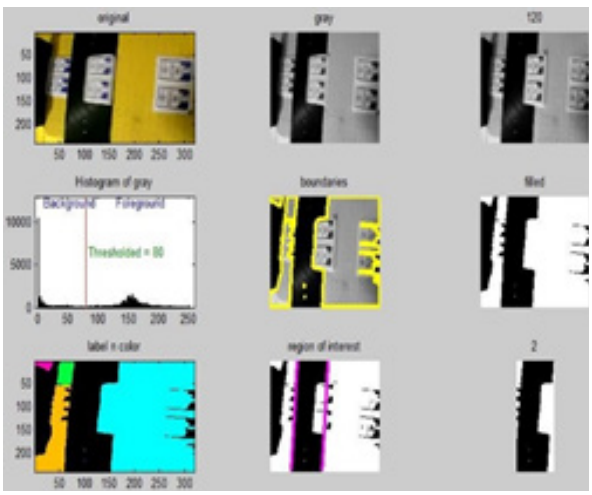

Figure 9.1 *Overall View of Analysis Window*

A complete view of the analysis and counting process visualization window is shown in Figure 9.1 where each item and its corresponding process will be given next. Here, both for simplicity and for paying the main attention on the image/video processing and counting algorithm a conveyor belt video prepared by www.4SmartMove.com where I have downloaded and used.

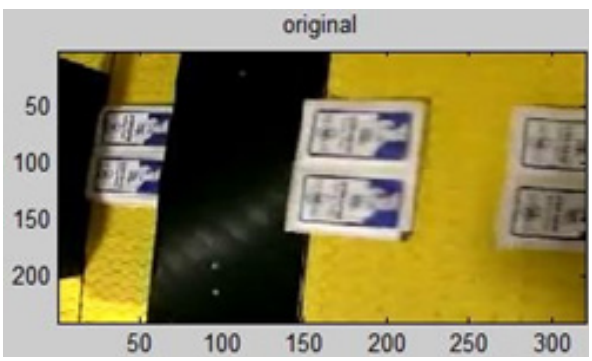

Figure 9.2 *Original Video*

There are several similar commercial videos prepared by the mentioned company and in detail a frame of the one of the incorporated videos can be seen in Figure 9.2. On a yellow background of the conveyor belt, products pass by. Resolution of the videos are 240x320 pixels, thus when I read the video in Matlab and separate it in frames, I obtain a 240x320x3 sized frame, thus it has RGB components.

Figure 9.3 *Gray Frames*

To apply processing easier and faster, RGB images (each one of the frames) are transformed to gray level images as can be seen in Figure 9.3. So to again for the computational concerns, a sampling process is conducted such that not using every frame but capturing one of each four frames to process further. Therefore image processing procedure becomes much more feasible for real-time applications in particular.

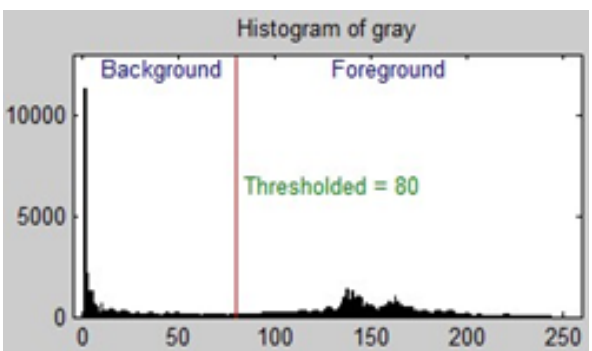

Figure 9.4 *Histogram of Gray-level Distributions*

After obtaining sampled gray-level images, a better way to simplify images is to transform them into binary valued images. In order to accomplish this transformation, here comes the histogram analysis part. By incorporating a histogram analysis, we can have information on the distribution of gray-levels in an image. For instance, in Figure 9.4 histogram values of a frame can be seen.

Figure 9.5 *Binary valued image*

Figure 9.6 *Filled version of the image*

Viewing of the gray-level distributions of an image yields a better understanding for the determination of the threshold value and/or values for binary-valued version image. From Figure 9.5 thresholded, i.e. binary version of the image can be seen. Chosen level, 80, as threshold value gives best result which is determined by trial and error and justified by the histogram plot as mentioned above. Lower or higher values do not yield adequate results, especially when filling 22 To further advance image processing in order to provide a smoother ground for counting application, filling small but irrelevant parts in contrast with seemingly big and/or available parts, an image filling approach is done. As a result a satisfactory image for consequent steps has obtained. The filled image that comes after the binary valued version can be seen in Figure 9.6. Before beginning to count products one more step is added to processing in order to visualize better the process, which is the labeling of the structures, that can be seen by eye, in our final filled images.

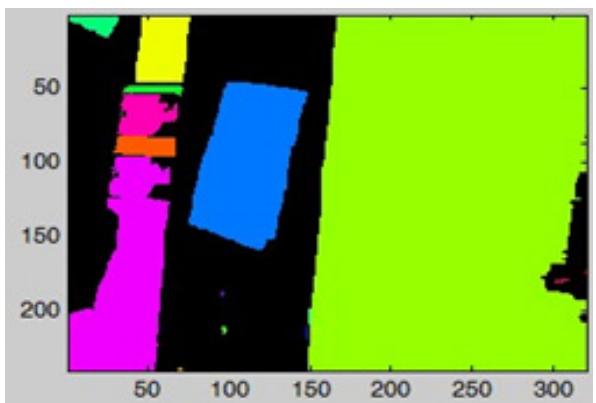

Figure 9.7 *Color-labeled image*

Before beginning to count products one more step is added to processing in order to visualize better the process, which is the labeling of the structures, that can be seen by eye, in our final filled images. Labeled image – parts then assigned random color codes as can be seen from Figure 9.7.

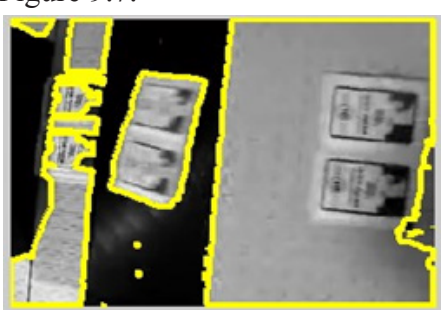

Figure 9.8 *Boundaries of Sub-structures*

A critique step comes here, which is the determination of boundaries in our image, which are the end lines of sub-structures. This step yielded a good determination of products on the conveyor belt, and an example frame for this step can be seen in Figure 9.8 where boundaries are plotted yellow onto the gray-level image.

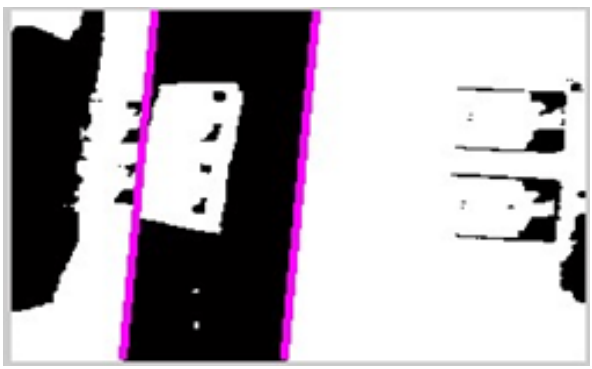

Figure 9.9 *Region of Interest*

Utilizing these boundaries, we can determine a 'region of interest' in order to apply counting process effectively. From Figure 9.9 it can be seen that the best possible region is selected, as it is shown in marble lines on the binary valued image.

Figure 9.10 *Target Area for Counting*

Finally, cutting this region gives us enough information for starting counting process. This novel part of the image can be seen from Figure 9.10. Resting part is a bit more straightforward, thus we have an almost black ground, where white 'products' pass on that surface.

REFERENCES

[1] R. C. Gonzalez and R. E. Woods, "Digital Image Processing". Gatesmark Publishing, 2008.

[2] R. C. Gonzalez, R. E. Woods, and S. Eddins, "Digital Image Processing UsingMatlab". Gatesmark Publishing , 2009.

[3] J. J. Ding, S. C. Pei, J. D. Huang, G. C. Guo, Y. C. Lin, N. C. Shen, and Y. S. Zhang,"Short response Hilbert transform for edge detection," CVGIP, 2007.

[4] S. C. Pei and J. J. Ding, "Improved Harris' algorithm for corner and edge detections," ICIP 2007.

[5] Thabit Sultan Mohammed and Nidal Ibrahim al-Tataie, "Artificial Neural Network as a Decision- Makers for Stereo Matching", GSTF-International Journal on ComputingVol. 1, No. 3, pp (89 – 94), August 2011.

[6] T.H. Lee, "Edge Detection Analysis", National Taiwan University, technical report 2008.

[7] J. Canny, "Finding Edges and Lines in Images," Massachusetts Institute of Technology, Cambridge, Massachusetts, USA, Tech. Rep. no. 720, 1983.

[8] R. Makkar, "Study of Image Matching Techniques", M.Sc. Thesis, Punjabi University, 2008.

[9] V. N. Radhika, B. Katikeyan, "Robust Stereo Image Matching for Space borne Imagery", IEEE Transactions on geosciences and remote sensing, vol. 45, No. 9, Sep. 2007.

[10] Thabit Sultan Mohammed, Nedhal Ibrahim Al-Taie, "Applying a Neural Network Framework for Stereo Matching", Annual Int. Conf. on Control, Automation and Robotics (CAR 2011), Singapore - 2011.

[11] Z. Guoqing, Y. Bao~onga nd T. Xiaofaiig, "A Software Package of Stereo Vision", ICSP ' 1996.

[12] D. Anastasia and Y. Andreopoulos, "Software Designs of Image Processing Tasks with Incremental Refinement of Computation", Signal Processing Systems, 2009. SiPS 2009.

[13] Wim Abbeloos. Fast matlab stereo matching algorithm (sad). Master's thesis, sep 2010.

[14] Jean-Yves Bouguet. Camera calibration toolbox for matlab. Technical report, California Institute of Technology, jul 2010.

[15] Gary Bradski and Adrian Kaehler. Learning OpenCV. O'Reilly Media, 1st edition, 2008. ISBN 978-0-596-51613-0.

[16] Edward R. Dougherty and Roberto A. Lotufo. Hands-on Morphological Image Processing. SPIE Press, 1st edition, 2003. ISBN 0-8194-4720-X.

[17] Rafael C. Gonzalez and Richard E. Woods. Digital Image Processing. Pearson Education, 3rd edition, 2008. ISBN 978-0-13-505267-9.

[18] Stefano Mattoccia. Stereo vision: Algorithms and applications. Technical report, University of Bologna, feb 2011.

[19] Ayache, N., and Faugeras,O. (1986). "HYPER: A new aproach for the recognition and positioning of two-dimensional objects." IEEE Trans. Pattern Anal PAMI 8 (1B Press.

[20] Bolles, R. C., and Horaud, P. (1986). "A three-dimensional part orientation system." Int.J.Rob. Res. 5(3): 3-26.

[21] Brooks, R. A. (1981). "Symbolic reasoning among 3-D models and 2-D images. ." Artif Intell 17: 285-348.

[22] Brooks, R. A. (1983). "Model-based three dimensional interpertations of twodimensional imIEEE Trans. Pattern Anal PAMI 5(2): 140-150.

[23] Duda, R. O., aages." Forsyth, D. A., and Ponce J. (2003). Computer Vision A modern Approach Edition. London, PrentiHall. F): 44-54.

[24] Ellman, R., and Dreyfus, S. (1962). Applied dynamic programming . Princeton, Princeton University

[25] Hanson, A.J. (1991). "An optimization framework fro feature extraction." Mach, Vision Appl 4: 59-87.

[25] Kimme, C., Ballard, D., and Sklansky, J. (1975). "Finding circles by an array of accumulators." Communications of the ACM 18(2): 120-122.

[26] Mathworks (1998). RGB Images. http://visl.technion.ac.il/labs/anat//2ImageTypes/#RGB%20Images.

[27] Nölle, M., Jonsson, B., and Rubik, M. (2004). Coin Images Seibersdorf-Benchmark, ARC Seibersdorf research GmbH: 1-8. Provine, J.,

[28] McClintock,M. , Murray, K., and Chau, A. . (1999). "Automatic coin counter."

[29] Rosenfeld, A. (1969). Picture Processing by Computer. New York, Academic Press.

[30] Sonka, M., Hlavac,v., and Boyle, R. (1999). Image processing, analysis and machine vision. Londow, PWS.

[31] Suetens, P., Fua, P., and Hanson, A. J. (1992). "Computational strategies for object recognition." ACM Computing Surveys 24(1): 5-61.

[32] Tenenbaum, J. M., and Barrow, H.G. (1977). "Experiments in interpretation-guided segmentation." Artif. Intell 8: 241-274.

[33] Zhao, W., Chellappa, R., Phillips, P.J., and Rosenfeld, A. (2003). "Face recognition: a literature survey." ACM Computing Surveys 35(4): 399-458.

[34] J. Koenderik. The structure of images. Biol. Cybern, 50:363–370, 1984.

[35] O. Carlsson L. Nyberg and B. Schmidtbauer. Estimation of the size distribution of fragmented rock in ore mining through automatic image processing. In Technological and Methodological Advances in Measurement - Acta IMEKO 1982, volume 3 of Data processing and System aspects, pages 293–302, Berlin, West Germany, May 1983. North Holland Publishing Company.

[36] G. Von Borries L. Yachner, C. Gonzales and R. Nobile. Coarse particle size distribution analyzer. In IFAC Symposium on Automation for Mineral Resource Development, Brisbane Queensland, Australia, July 1985.

[37] T.B. Lange. Measurement of the size distribution of rocks on a conveyor belt using machine vision. PhD thesis, University of Witwatersrand, Faculty of Engineering, University of Witwatersrand, Johannesburg, 1990.

[38] N.H. Maerz. Image sampling techniques and requirements for automated image analysis of rock fragmentation. Proceedings of the FRAGBLAST 5, Workshop on Measurement of blast fragmentation, pages 115–120, August 1996.

[39] N.H. Maerz. Reconstructing 3-d block size distributions from 2-d measurements on sections. Proceedings of the FRAGBLAST 5, Workshop on Measurement of Blast Fragmentation, pages 39–43, August 1996.

[40] N.H. Maerz. Aggregate sizing and shape determination using digital image processing. Center for Aggregates Research (ICAR) Sixth Annual Symposium Proceedings, pages 195– 203, 1998.

[41] N.H. Maerz and T.C. Palangio. Wipfrag system ii- online fragmentation analysis. FRAGBLAST 6, Sixth Internation Symposium for rock fragmentation by blasting, pages 111–115, August 1999.

[42] N.H. Maerz and W. Zhou. Optical digital fragmentation measuring systems-inherent sources of error. FRAGBLAST, The international journal for blasting and fragmentation, 2(4):415–431, 1998.

[43] N.H. Maerz andW. Zhou. Calibration of optical digital fragmentation measuring systems. FRAGBLAST, The international journal for blasting and rock fragmentation, 4(2):126–138, 2000.

[44] D.J. Mashao. Comparing svm and gmm on parametric feature-sets. Proceedings of the 14th Annual Symposium of the Pattern Recognition Association of South Africa, 2003.

[45] C. McDermott and N.J. Miles. The measurement of rock fragmentation using image analysis. Departmental magazine, Department of Mining Engineering, pages 49–61, 1988.

[46] D.H. Marimont M.J. Black, G. Sapiro and D. Heeger. Robust anisotropic diffusion. IEEE Transactions on Image Processing, 7(3):421–232, March 1998.

[47] S.G. Mkwelo. A comparative evaluation of evolutionary design methods in engineering. A 4th year undergraduate thesis in the department of electrical engineering, University of Cape Town, October 2001.

[48] T.J. Napier-Munn. An introduction to comparative statistics and experimental design for Minerals Engineers. Julius Kruttschnitt mineral research centre, The University of Queensland, 2 edition, 2001.

[49] T.C. Palangio N.H. Maerz and J.A. Franklin. Wipfrag image based granulometry system. Proceedings of the FRAGBLAST 5, Workshop on measurement of Blast Fragmentation, pages 91–99, August 1996.

[50] T.C. Palangio and N.H. Maerz. Case studies using the wipfrag image analysis system. FRAGBLAST 6, Sixth Symposium for rock fragmentation by blasting, pages 117–120, August 1999.

[51] P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. IEEE Transactions on Pattern Analysis and Machine Intelligence, 12(7):629 – 639, July 1990.

[52] T. Poggio and S. Smale. The mathematics of learning: Dealing with data. Notices of the AMS, 50(5), 2003.

[53] C. Pretorious and A.L. Nel. Rock size mintoring on conveyor belt systemsusing neural networks. Proceedings of the second South African Workshop on Pattern Recognition, pages 100–110, November 1991.

[54] W. Hue Q. Song and W. Xie. Robust support vector machine with bullet hole image classification. IEEE Transactions on Systems, Man, and Cybernetics-Part C, 32(4):440–448, November 2002.

[55] J.B.T.M. Roerdink and A. Meijster. The watershed transform: Definitions, algorithms and parallelization strategies. Fundamenta Informaticae, pages 187–228, 2000.

[56] G. de Jager S. Mkwelo and F. Nicolls. Watershed-based segmentation of rock-scenes and proximity-based classification of watershed regions under uncontrolled lighting conditions. Proceedings of the 14th Annual Symposium of the Pattern Recognition Association of South Afrca, 2003.

[57] J. Serra. Image analysis and mathematical morphology, volume 1. Academic Press, 1982.

[58] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. Proceedings of the International Conference on Computer Vision, 1998.

[59] R. van den Boomgard and J. van den Weijer. On the equivalence of local-mode finding, robust estimation and mean-shift analysis as used in early vision tasks. Proceedings of the 16th International Conference on Pattern Recognition, 3:927–930, 2002.

[60] L. Vincent and P. Soille. Watersheds in digital spaces: an effiecient algorithm based on immersion simulations. PAMI, 13(6):583–598, 1991.

[61] The Webmaster. The split-engineering website, February 2004. www.spliteng.com.

[62] The Webmaster. The wipfrag website, March 2004. www.wipware.com.

[63] David Wigeson. Fragmentation analysis using computer vision techniques. Master's thesis, University of Witwatersrand, 1987.

[64] B.A. Willis. Mineral Processing technology. Butterworth Heinenmann publishing (Ltd), 6 edition, 1997.

[65] A. Witkin. Scale-space filtering. Int. Joint Conf. Artificial Intelligence, pages 1019– 1021, 1983.

[66] B.A Wright. The development of a vision based flotation froth analysis system. Master's thesis, University of Cape Town, Faculty of Engineering and Built Environment, University of Cape Town, Rondebosch 7700, South Africa, 1999.