 CMC, vol.42, no.2, pp.125-140, 2014

Fast and High-Resolution Optical Inspection System for In-Line Detection and Labeling of Surface Defects

M. Chang[1,2,3], Y. C. Chou[1,2], P. T. Lin[1,2] and J. L. Gabayno[2,4]

Abstract: Automated optical inspection systems installed in production lines help ensure high throughput by speeding up inspection of defects that are otherwise difficult to detect using the naked eye. However, depending on the size and surface properties of the products such as micro-cracks on touchscreen panels glass cover, the detection speed and accuracy are limited by the imaging module and lighting technique. Therefore the current inspection methods are still delegated to a few qualified personnel whose limited capacity has been a huge tradeoff for high volume production. In this study, an automated optical technology for in-line surface defect inspection is developed offering high performance in spatial resolution and detection speed for any surface. The inspection system consisting of an LED array which illuminates a wide inspection area on the test object captures scattered light from surface defects using a 12288-pixel line CCD at 12 kHz acquisition rate. A 3.5 μm per pixel resolution of the line CCD provides a detection width capability of at most 43 mm which is equivalent to 147 megapixels image data acquired per second. To handle the large volume of data per acquisition cycle, the data are transmitted from a host CPU to multiple GPU devices where CUDA-based image processing kernels are adopted to perform detection and labeling of surface defects in parallel. The processed data is sent back to the CPU to display user-defined defect maps. 2-D inspection of back-coated flat mirrors, 43 mm x 70 mm^2 in size, using a single CCD module and multiple GPU reveals that surface flaws such as bubbles, cracks, and edge defects are detected accurately. The acquisition time to capture and load the data to a CPU is 1.7 s while the processing time to transmit the same data for surface defect detection in a GPU is 248 ms. The latter time scale is considerably

[1] Department of Mechanical Engineering, Chung Yuan Christian University, Chung Li, Taoyuan, Taiwan 32023.

[2] Center for Biomedical Technology, Chung Yuan Christian University, Chung Li, Taoyuan, Taiwan 32023.

[3] The State Key Laboratory of Digital Manufacturing Equipment and Technology, Huazhong University of Science and Technology, Wuhan, China 430074.

[4] Mapua Institute of Technology, Muralla St., Intramuros, 1002 Manila, Philippines.

faster compared to minute-long computations in solely CPU-based processing algorithm of the same test object. The minimum width of detected surface defects is about 10 μm with true detection rates above 94%. Moreover, the inspection system is easily configurable by tasking multiple CCD imaging modules to different GPU devices to allow inspection of larger test objects. This flexibility can improve both acquisition and detection speeds to boost in-line circuit chips, packaging, and touchscreen panel inspection systems.

Keywords: surface defects, automated in-line inspection, linear CCD, parallel computing, Compute Unified Device Architecture (CUDA).

1 Introduction

Manufactured products are tested prior to delivery to ensure that they remain reliable and meet customer expectations. Under such environment, it is essential that the tests being implemented do not compromise the properties and performance of the final product, i.e. non-destructive [Blitz (1991)]. The decision to test the quality of a product is usually defined by economic considerations in the part of the manufacturer, where cost of the inspection system, expenses of replacing a failed component, and inconvenience to the end users are weighed in. Prohibitive costs of test systems often lead manufacturers to take calculated risks and only test a small sample from a given batch of products. However, for sensitive components such as ICs, imagers, etc., in-line inspection is strictly enforced during the early stages of manufacturing and at specific intervals in order to monitor critical parts of the product. Early detection of defects at these stages help manufacturers implement corrective actions prior to delivery thereby improving overall productivity by reducing re-works, line disruptions, and eventual shipment delay.

Automated optical inspection (AOI) systems based on computer or machine vision technology (MVT) have been employed to equip a production line with more reliable measurements. The fast acquisition speeds of the instruments compared to manual inspection saves the manufacturer both time and resources in troubleshooting, quality evaluation, and yield. Depending on the size and surface properties of the materials, the inherent concerns on the performance of AOIs are the processing speed, accuracy, and resolution of the imaging module.

Computer vision technology is a method that relies on image or motion analysis to provide automated optical inspection, robot control, remote navigation, and dynamic visual recognition. In general, it consists of an image acquisition module such as a camera, lenses, and illumination source and a software package that can interpret the acquired images by means of digital image-processing (DIP) techniques [Gonzalez and Woods (2008)]. The software package is developed specifi-

cally to extract key features from the images where decisions for succeeding actions of the system are based on. In recent years, the technology has been widely used in automotive camera systems, remote surveillance, materials sorting, and product packaging.

Manufacturing industries have employed MVT as a non-destructive test (NDT) protocol for reliable inspection and localization of surface defects. In such systems, one or multiple DIP solutions are applied after image acquisition to assess the integrity of the product. Among the popular DIP operations include gamma correction, stitching, binarization, labeling, edge detection, color and texture analyses, and pattern recognition. Surface defect inspection systems based on these operations have been previously proposed in combination with advanced imaging techniques. Inspection platforms using linear CCDs were developed to discriminate texture variation in metal slabs [Ryu, Choi, Jeon, Lee, Yun, and Kim (2014)] and surface flaws in decorative labels [Busin, Vandenbroucke, and Macaire (2013)]. Contrast based image processing algorithm [Li, Lu, and Zhang (2012)] and image recognition technique [Chen, Lin, Han, and Liang (2013)] were also proposed to localize defect in steel bars and ceramic surfaces, respectively. Scanning systems for in-line inspection were developed for surface inspection of metal slabs [Zhao, Ouyang, Chen, and Wen (2011)] and textures of plastic surfaces [Michaeli and Berdel (2011)]. Other schemes featured automatic detection of cracks on solar wafers [Li and Tsai (2011); Tsai, Chang and Chao (2010); Chiou, Liu and Liang (2011)] and real-time inspection of automotive parts [Rosati, Boschetti, Biondi, and Rossi (2009)] and reflective metal surfaces [Zhang, Ding, Lv, Shi, and Liang (2011)]. To inspect surface imperfections on LCD panels and other glass samples, the solutions employed were optical flow-based motion analysis [Tsai and Tsai (2011)], binary feature histogram operation [Zhao, Kong, Zhao, Liu, and Liu (2011)], and color space selection method [Nishu and Agrawal (2012)]. In terms of screening products based on pre-defined criteria, these techniques have delivered reliable performance. However, a compromise between resolution and detection speed often creates a bottleneck for in-line inspection.

This paper presents an inspection system that can achieve significant and reliable performance in speed and resolution for extraction and labeling of micron-sized surface defects. The system features a lighting solution combining a linear CCD with LED array and a parallel computing algorithm for simultaneous detection and labeling of the surface defects. These defects are reconstructed from light scattered on the surface of the test objects. The information is collected by the line CCD and stored in a host CPU. To deliver fast computation, the image data is transmitted to a GPU device where computations using Compute Unified Device Architecture (CUDA) kernel functions such as gamma correction, Gaussian pyramid, binariza-

tion, labeling, and edge detection, are completed in parallel. The processed data is sent back to the CPU allowing the user to display a defect map. To our knowledge, this is the first non-destructive optical inspection system that combines parallel computing architecture to support both high resolution and fast detection speed of surface defects for in-line measurements of large test objects.

2 Optical Inspection Setup

The optical inspection system consists of an imaging module, a motorized inspection stage, and image processing software such as shown in Fig. 1. Each imaging module features a 5W broadband light source and a 12288 pixel-line CCD camera (Basler raL12288-66km). The pixel resolution of the CCD is 3.5 μm hence the detectable width is roughly 43 mm per CCD module. At 12 kHz line rate, the image data acquired by one CCD alone is approximately 147 megapixels per second. The detectable width can be improved to accommodate even bigger samples by investing on additional imaging modules or increasing the pixel number of each module. Huge data capacity can be expected from either method which can be handled by parallel processing on a common integration server combining CPU and multiple GPU devices.

The inspection stage is motorized with a maximum translation speed of 70 mm/s. It has a built-in movable holder for mounting the samples. Two opposite levers can be used to lock-in different sizes of test samples on the stage. The levers are also fitted with adjustment knobs to latch on the samples and control their position and tilt. The camera and linear stage are connected to a common host computer. The images acquired are also stored on the same computer.

In order to demonstrate the detection speed and resolution capability of the test setup, this study will focus on only one imaging module. The line CCD is oriented parallel to the width of the test object as shown. Two sets of LED array are mounted on the light source holder and are juxtaposed to the CCD. The LED illuminates the sample at an angle from the normal. In principle, the incident light impinging the test object is reflected, transmitted, or absorbed by the surface. In our case, the normal orientation of the CCD with respect to the sample is optimized to receive the most light which are scattered from artificial or geometrical defects on the surface. Thus, if the surface is perfectly smooth the camera will record a mostly dark image since the majority of the light reflected is specular. Conversely, if there is a defect on the surface it will scatter light in the direction of the camera which in turn appears as a bright area in a dark background on the recorded image.

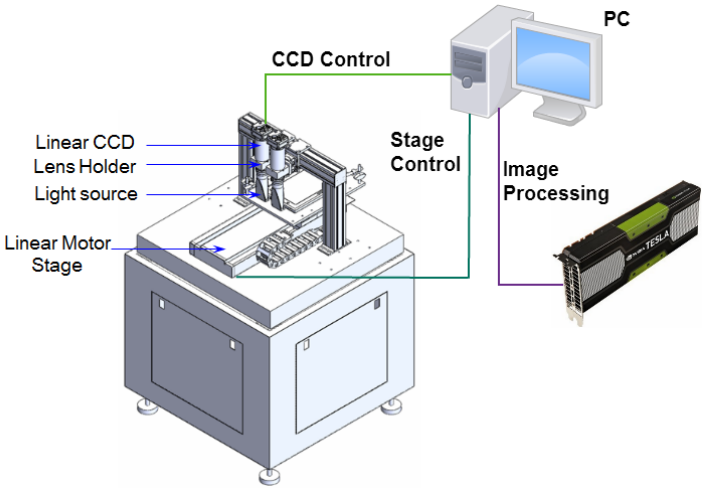


Figure 1: The hardware components of the optical inspection system.

3 System Flow Chart

As shown in Fig. 2, the surface detection software initially prompts the user to input the parameters settings which include the capture rate of the CCD, speed of the motorized stage, and acquisition time. A command string is sent out next to initialize the CCD and the stage. The line CCD acquires the surface scattered light from the test object as it moves on the translation stage. The image data is stored on the host server (CPU) after the set acquisition time has elapsed. The image pre-processing and defect inspection tasks are assigned as CUDA-based kernel functions to a GPU device. With this design and depending on the size of the object, multiple GPU devices can be configured to perform the same kernel operations in parallel.

In a GPU device, the image pre-processing and defect detection tasks are performed as pixel-wise operations on a designed block of CUDA threads. An image pixel is represented by a CUDA thread which can be varied depending on the resolution requirements. Initially, the raw data is copied from the CPU's main memory to the GPU's global memory through a Message Passing Interface (MPI) command. Separate memory is also allocated in the GPU to contain the processed data after each kernel function, i.e. dst_1, dst_2, etc. in Fig 2. The kernel functions are also initiated by MPI commands. After the CUDA threads complete the kernels, the processed data (i.e. dst_out) is transferred back to the CPU. The processed data from multiple GPUs are combined to generate a defect map where the size, number, and features of surface defects are displayed for defect classification or quality evaluation of the test object.

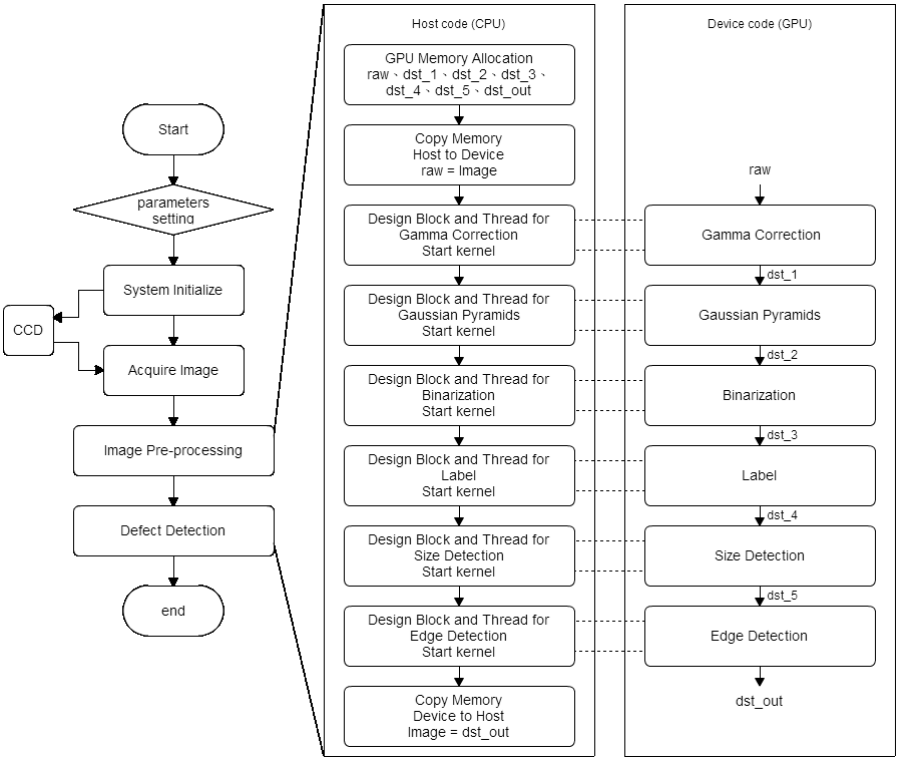


Figure 2: The system flowchart from acquisition to transmission and detection of surface defects.

4 Kernel Functions for Surface Defect Inspection

Six kernel functions are carried out by the CUDA threads. These include two image pre-processing operations and four successive defect detection algorithms. These kernels can be completed iteratively on a single GPU or in parallel on multiple devices depending on the size of the image data.

4.1 Image pre-processing

The first kernel is gamma correction [Asadi and Hassanpour (2012)]. This is applied to the raw image I_{in} to enhance the contrast and is defined by the relation $I_{out} = \alpha I_{in}^{\gamma}$. The coefficient αgs arbitrarily assigned a value of 1 and γg 0.84 is chosen after optimization. The output data is normalized from 0 to 255. After correction, the data is stored in the allocated memory dst_1 as shown in Fig. 2.

The second kernel implements a Gaussian pyramid [Jain and Sharma (2013)] scal-

ing operation to further reduce the computation time. The processed data in dst_1 is resized by convolving with the kernels $g_R(x,y,\sigma) = \frac{1}{16}[1\ 4\ 6\ 4\ 1\ ]$ along the rows and $g_C(x,y,\sigma) = \frac{1}{16}\begin{bmatrix} 1 \\ 4 \\ 6 \\ 4 \\ 1 \end{bmatrix}$ on all the columns. The computation is repeated until the pixel size approaches a set threshold, e.g. 25 μm. All the even rows and columns are removed after convolution and the processed output is stored in the allocated memory dst_2.

4.2 Defect Detection

Four kernel functions for defect detection are carried out in succession by the CUDA threads. These include binarization, labeling, size detection, and edge detection. The latter two operations can be combined into one kernel.

4.2.1 Binarization

The processed data in dst_2 is binarized based on a preset threshold. As illustrated in Fig. 3, if a pixel value is larger than the threshold, the pixel value is set to 255. Otherwise, the pixel value is set to zero. The binarized output is stored in the allocated memory dst_3.

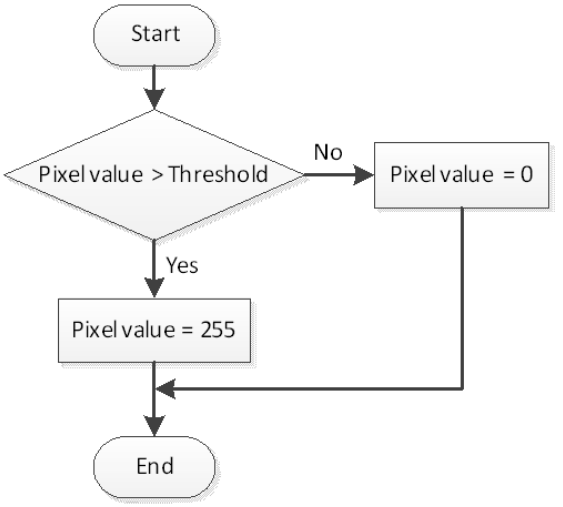


Figure 3: The binarization algorithm.

4.2.2 Labeling

In order to obtain the number of defects in an image, each are assigned a label. The idea is to locate the starting and terminal points of defects, and then eliminate the redundant starting and terminal points. After elimination, the number of defects should be the same as the number of starting or terminal points.

The algorithm for determining the starting point of a defect follows the diagram shown in Fig. 4(a). If a pixel value is equal to zero, the pixel value is not changed. However, if a pixel value is larger than zero, and if the pixel value's left, upper-left, upper, and upper-right pixel values are all zeroes, that pixel value is set to 1. Otherwise, the pixel value is set to 9. Image pixels with a value of 1 are then considered the starting points of defects.

The algorithm for determining the terminal point of a defect is illustrated in Fig. 4(b). If a pixel value is either equal to zero or 1, that pixel value is not changed. However, if a pixel value is larger than zero but not equal to 1, and if the pixel value's right, bottom-right, bottom, and bottom-left pixel values are all zeroes, the pixel value is set to 2. Otherwise, the pixel value is still set to 9. Image pixels with a value of 2 are then considered the terminal points of defects.

The algorithm for elimination of redundant defect starting points is shown in Fig. 5(a). This operation is specifically targeted toward pixels with a value equal to 1. In such a case, if there is any pixel on its left side along the same row with a value of 1, the target pixel's value is changed from 1 to 9.

The algorithm for elimination of redundant defect terminal point is illustrated in Fig. 5(b). In contrast to the previous operation, this is targeted toward pixels with a value equal to 2. For a target pixel, if there is any pixel on its right side along the same row with a value of 2, the target pixel's value is changed from 2 to 9. The processed output after labeling is stored in the allocated memory dst_4.

4.2.3 Size and Edge Detection

Finally, this operation is implemented to obtain the size and geometric shapes or contours of defects, i.e. the left, upper, right, and bottom edges of defects. In addition, after the labeling operation, all nonzero pixel values are changed back to 255, which is the original value obtained after binarization. Moreover, an array of zeroes, which has the same size as the image, is allocated in the CUDA global memory for the edge detect algorithm.

The algorithm for left edge detection of defect is shown in Fig. 6(a). For a target pixel with a value of 255, if the values of its left and right pixels are zero and 1, respectively, the target pixel's value in the new array is set to 255. Otherwise, the target pixel's value in the new array is not changed.

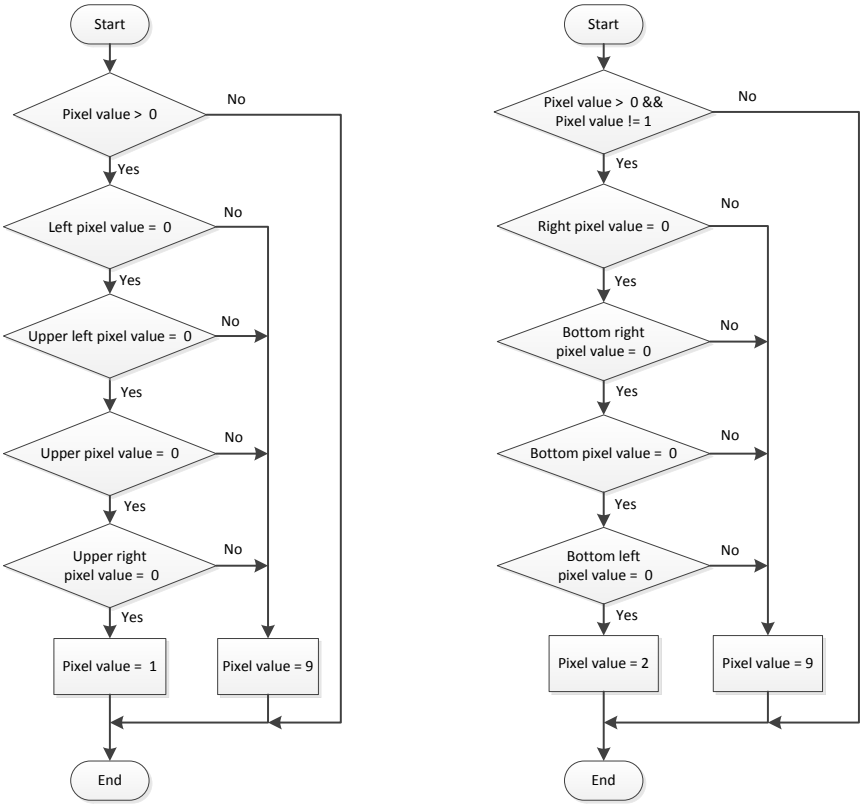


Figure 4: The defect (a) starting point and (b) terminal point determination algorithms.

The algorithm for right edge detection shown in Fig. 6(b) is the counterpart of the left edge detection procedure. For a target pixel with a value of 255, if the values of its right and left pixels are zero and 1, respectively, the target pixel's value in the new array is also set to 255. Otherwise, the target pixel's value in the new array is not changed.

The algorithm for upper edge detection is illustrated in Fig. 7(a). For a target pixel with a value of 255, if the values of its upper pixel and its corresponding pixel in the new array are both zeroes, and also the value of its bottom pixel is 255, then the target pixel's value in the new array is set to 255. Otherwise, the target pixel's value in the new array is not changed.

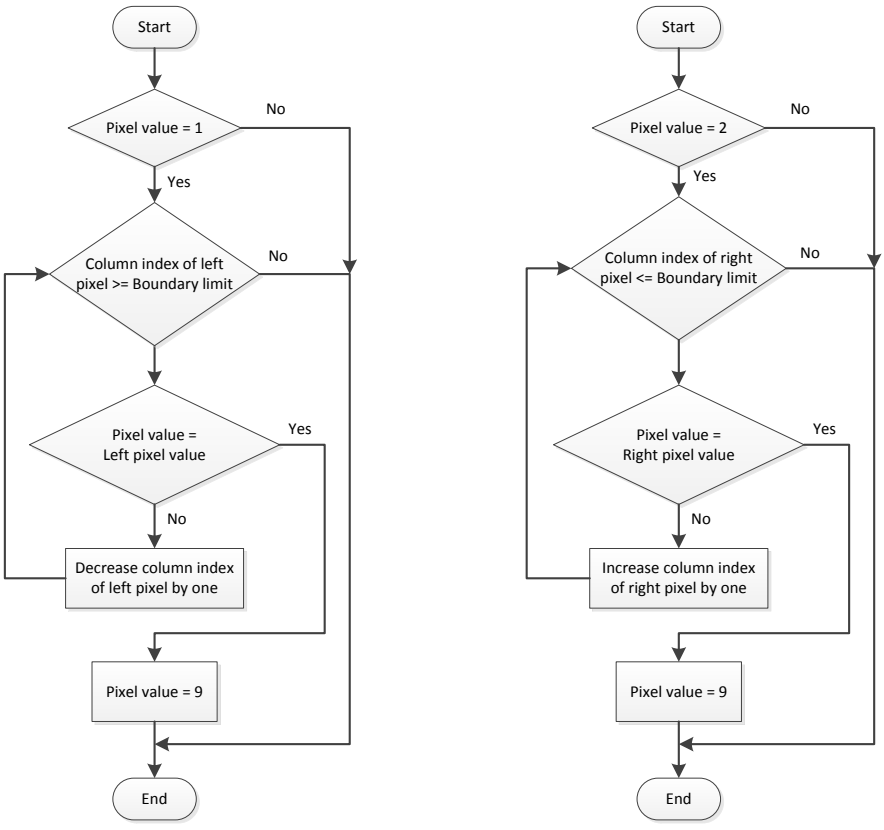
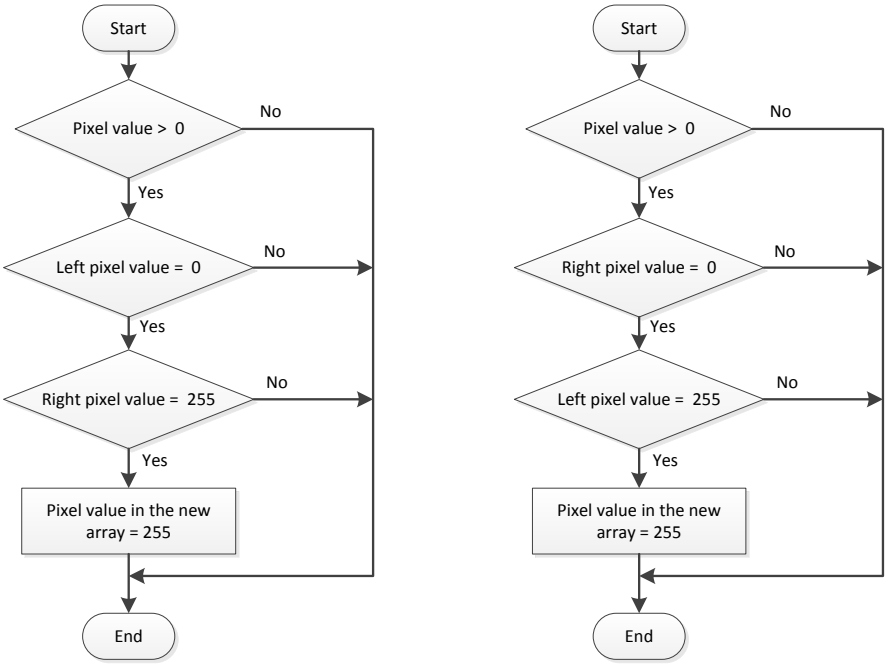


Figure 5: The redundant defect (a) starting point elimination and (b) terminal point elimination algorithms.

The algorithm for bottom edge detection shown in Fig. 7(b) is also the counterpart of the upper edge detection procedure. For a target pixel with a value of 255, if the values of its bottom pixel and its corresponding pixel in the new array are both zeroes, and also the value of its upper pixel is 255, then the target pixel's value in the new array is set to 255. Otherwise, the target pixel's value in the new array is not changed.

The size of the defects is calculated from the bounded edges. The processed output is stored in the allocated memory dst_out. This memory is directly accessible to the CPU.



(a) (b)

Figure 6: The defect (a) left edge and (b) right edge detection algorithms.

5 Experimental Results

The dark-field image of the upper left corner of a back-coated mirror object is shown in Fig. 8(a). The actual dimension is 43 mm (W) x 70 mm (L). The image on the left was the raw data stored on the CPU's main memory after acquisition. The data is transferred to the GPU's global memory for defect inspection following the algorithm discussed in Fig. 2. The processed output in dst_out is shown in Fig. 8(b). On the CPU, a defect map was created based on the geometrical shape, size, and position of the defects on the sample. The map is a specific area on the test object where a cluster of defects is detected. An example of a defect map on the corner of a test sample is shown in Fig. 10(c). The sizes and features of defects found on the sample are listed in Table 1. Point-like or circular defects ranging from 50 μm to 160 μm in diameter are labeled as bubbles. Thin and elongated features which are located far from the edge of the test object are labeled as cracks. The width and length of the cracks found on the sample is about 10 to 20 μm and 1 to

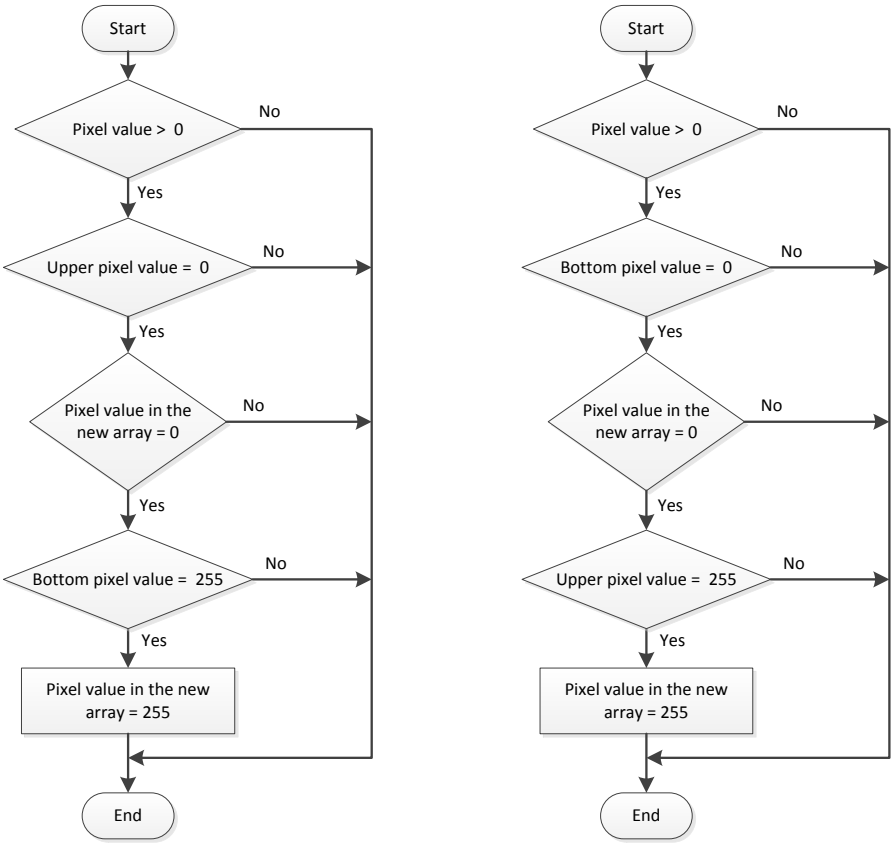
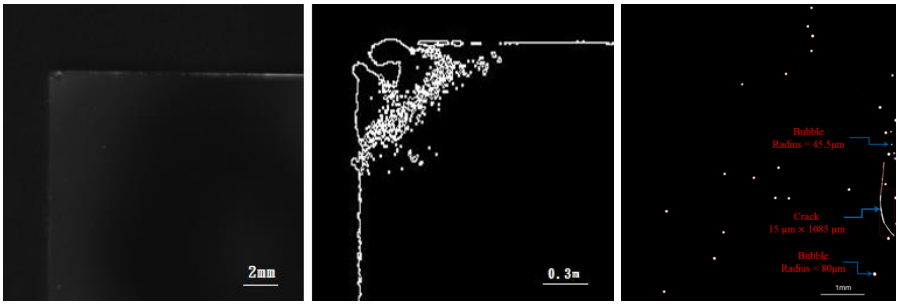


Figure 7: The defect (a) upper edge and (b) bottom edge detection algorithms.

10 mm, respectively. Jagged contours and voids that are detected on the perimeter of the test object are generally labeled as edge defects. Information about the shape and size of this particular defect can be utilized to gauge not only the quality of the test object but also the integrity of the tooling/cutting process on the sample.

Tab. 2 summarizes the detection sensitivity of the algorithm for each defect. The calculation compares the number of defects correctly detected by the algorithm with that of the actual defects. The results are significant and show that the accuracy of the algorithm is above 94% on the three defects category.

The processing time for each kernel function on a single GPU is listed in Table 3.



(a) (b) (c)

Figure 8: (a) Dark-field image of test object stored on CPUs main memory. (b) Processed output with defects detected near the upper-left edge. (c) A typical defect map with labeled bubbles and crack.

Table 1: Size and features found on the test sample.

	Features	Average size
Bubbles	Point-like or circular shape	50 to 160 μm
Cracks	Thin, elongated, located far from the perimeter of the test object	Width: 10 to 20 μm Length: 1 to 10 mm
Edge defects	Jagged lines, voids	

Table 2: True detection rates (TDR) of surface defect detection algorithm. TDR = 100x(N_d/N_a); N_d = defects identified from algorithm, N_a = actual defects = 100.

Surface Defects	TDR
Bubbles	96 %
Cracks	94 %
Edge defects	99 %

The total time to complete the image-processing tasks is 126.71 ms. Meanwhile, the total time to complete the defect detection tasks is 121.82 ms. In total, the combined processing times is significantly faster than computations on a CPU of the same tasks which exceeds more than 1 minute for the same test object.

Table 3: Computation time to complete surface detection.

Processing commands		Processing time (ms)
Image pre-processing	Copy memory from CPU to GPU	79.306
	GPU kernel function: Gamma correction and Gaussian pyramid	0.091
	Copy memory from GPU to CPU	47.309
Defect detection	Copy memory from CPU to GPU	61.581
	GPU kernel function: Binarization, Labeling, size and edge detections	0.144
	Copy memory from CPU to GPU	60.092

6 Conclusions

The paper presents an automated optical inspection system that combines a versatile imaging module with parallel computing platform to achieve fast and high resolution surface defect detection for in-line measurements. Fast processing time of large image data from the imaging module is realized by utilizing GPU structures to execute image processing functions in parallel. High resolution surface flaws such as bubbles, cracks, and edge defects are detected accurately from the image processed defect maps where the width of the smallest defect is about 10 μm. The sensitivity of the detection algorithm is above 94% for each defect category. The computation time to complete the defect inspection on a single CPU-GPU platform is at least 248 ms, which is significantly faster compared to purely CPU-based processing. As such, inspection of defects on large objects is possible by either multiplying the GPU platforms to complete the processing tasks or increasing the number of imaging modules to expand the inspection area.

Acknowledgement: This research is supported by the Ministry of Science and Technology under grant No. 102-2218-E-033-002-MY2 and Chung Yuan Christian University.

References

Asadi, S.; Hassanpour, H. (2012): A preprocessing approach for image analysis using gamma correction. *International Journal of Computer Applications,* vol. 38**,** no. 12, pp. 38-46.

Blitz, J. (1991): *Electrical and Magnetic Methods of Nondestructive Testing*, IOP

Publishing Ltd, New York. 1991. pp. 1.

Busin, L.; Vandenbroucke, N.; Macaire, L. (2013): Contribution of a color space selection to a flaw detection vision system. *Journal of Electronic Imaging* vol. 22, no. 3, pp. 17.

Chen, S. G.; Lin, B.; Han, X. S.; Liang, X. H. (2013): Automated inspection of engineering ceramic grinding surface damage based on image recognition. *International Journal of Advanced Manufacturing Technology,* vol. 66, no. 1-4, pp. 431-443.

Chiou, Y. C.; Liu, J. Z.; Liang, Y. T. (2011): Micro-crack detection of multicrystalline silicon solar wafer using machine vision techniques. *Sensor Review*, vol. 31, no. 2, pp. 154-165.

Gonzalez, R. C; Woods, R. E. (2008): *Digital Image Processing*, 3rd ed., Prentice Hall.

Jain, A.; Sharma, S. (2013): Entropy enhancement using image fusion techniques based on Laplacian and Gaussian pyramid decomposition. *International Journal of Engineering Innovation & Research*, vol. 2, no. 3, pp. 273-276.

Li, W. B.; Lu, C. H.; Zhang, J. C. (2012): A local annular contrast based real-time inspection algorithm for steel bar surface defects. *Applied Surface Science*, vol. 258 no. 16, pp. 6080-6086.

Li, W. C.; Tsai, D. M. (2011): Automatic saw-mark detection in multicrystalline solar wafer images. *Solar Energy Materials and Solar Cells* vol. 95, no. 8, pp. 2206-2220.

Michaeli, W.; Berdel, K. (2011): In-line inspection of textured plastics surfaces. *Optical Engineering*, vol. 50, no. 2, pp. 6.

Nishu; Agrawal S. (2012): Automated inspection of defects in glass by proper color space selection and segmentation technique of digital image processing. *International Journal of Computer Technology & Applications*, vol. 3, no. 3, pp. 1058-1063.

Rosati, G.; Boschetti, G.; Biondi, A.; Rossi, A. (2009): Real-time defect detection on highly reflective curved surfaces. *Optics and Lasers in Engineering*, vol. 47, no. 3-4, pp. 379-384.

Ryu, S. G.; Choi, D. C.; Jeon, Y. J.; Lee, S. J.; Yun, J. P.; Kim, S. W. (2014): Detection of scarfing faults on the edges of slabs. *ISJI International*, vol. 54, no.1, pp. 112-118.

Tsai, D. M.; Chang, C. C.; Chao, S. M. (2010): Micro-crack inspection in heterogeneously textured solar wafers using anisotropic diffusion. *Image and Vision Computing*, vol. 28, no. 3, pp. 491-501.

Tsai, D. M.; Tsai, H. Y. (2011): Low-contrast surface inspection of mura defects in liquid crystal displays using optical flow-based motion analysis. *Machine Vision and Applications*, vol. 22, no. 4, pp. 629-649.

Zhang, X. W.; Ding, Y. Q.; Lv, Y. Y.; Shi, A. Y.; Liang, R. Y. (2011): A vision inspection system for the surface defects of strongly reflected metal based on multi-class Svm. *Expert Systems with Applications*, vol. 38, no. 5, pp. 5930-5939.

Zhao, L. M.; Ouyang, Q.; Chen, D. F.; Wen, L. Y. (2011): Surface defects inspection method in hot slab continuous casting process. *Ironmaking & Steelmaking*, vol. 38, no. 6, pp. 464-470.

Zhao, J.; Kong, Q. J.; Zhao, X.; Liu, J.; Liu, Y. (2011): A method for detection and classification of glass defects in low resolution images. *Sixth International Conference on Image and Graphics*, pp 642-647