in both images. However, images captured at different times always differ from one another with different lighting environment, even if the object remains fixed in space. Our scheme for estimating the location of an object can still obtain the accuracy displacement between images based on their correlation coefficient.
To select a source image block for use as the basis block is of great importance when a correlation coefficient is being used to estimate the displacement of an object. Location and size are two of the major properties of an image block, and the values of these properties that are appropriate for computing correlation coefficients are efficiently estimated using the proposed HTP-based evaluation method. Even if the displacement undergone by an object is very small, the correlation coefficient can still clearly reveal it, based on the difference between the source and target images. The feasibility of the proposed method is demonstrated by a numerical simulation of a photographic experiment in which the structure is at a short distance from the camera. In another experiment, a small three-story frame was mounted on a Quanser Shake Table II and a linear variable differential transformer (LVDT) was placed on its second floor (Fig. 2). The results of both these experiments have been previously published (Lu *et al.* 2014).

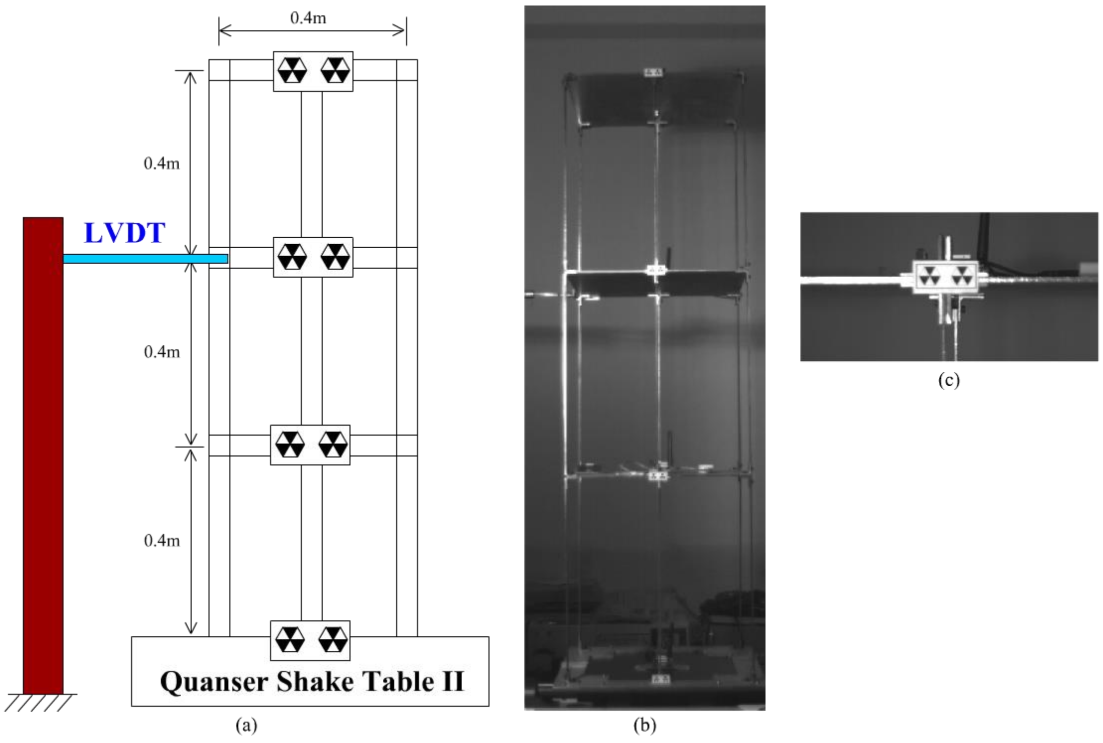

Fig. 2 (a) Configuration of small three-story frame with an LVDT on its second floor; (b) an image frame captured using measurement system with low and (c) high resolution.

3. GRAPHICS PROCESSING UNIT AND CUDA

Parallel computing executes multiple instructions concurrently, and its feature known as multi-threading can simultaneously execute multiple processes or threads in a multi-core CPU to accelerate scientific computing (Hung & Adeli 1992, 1994, Adeli &

Hung 1993). The thread is the fundamental building block of a parallel program; most traditional programs are of a single-thread type because the primary hardware on which they are executed is a single-core CPU.
OpenMP is an application-programming interface (API) that supports multi-platform, shared-memory multiprocessing programming in C, C++, and FORTRAN. It consists of a set of compiler directives, library routines, and environmental variables that affect run-time behavior. A programmer can easily use OpenMP to develop multi-core CPU parallel programs. Advances in GPU technology have opened a new avenue for increasing computing power. GPU-based techniques can be hundreds of times faster than conventional sequential computation without sacrificing solution quality. CUDA is a parallel computing platform and API that was created by NVIDIA. It allows software developers to use a CUDA-enabled GPU for general-purpose processing – an approach known as GPGPU. The CUDA programming model is a variant of the single instruction multiple data (SIMD) model, also known as the single program multiple data model, which uses an instruction model called single instruction multiple threads (Cook 2013). A flowchart of the CUDA programming model (Figure 3) includes the following four steps: copy host data to device, invoke kernel function, execute kernel function on GPU, and copy result from device to host. Digital image data must be copied to the GPU. The DIC method is coded into the kernel function, and executed on a GPU. The computational result from the GPU is copied to the CPU memory and written into an external file.

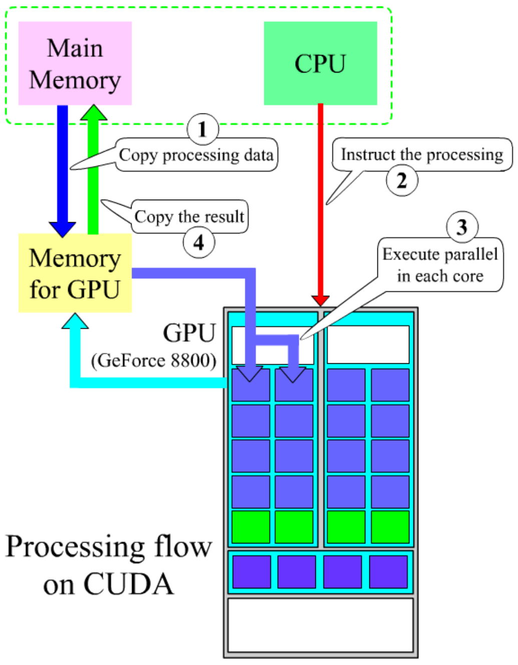

Fig. 3 Flowchart of execution on CUDA.

Modern GPUs have highly parallel structures, which make them more effective than general-purpose CPUs in executing algorithms to process large blocks of visual data in parallel. The NVIDIA GPU was chosen to solve displacement-evaluation problems in the present work because, as a CUDA-enabled GPU, it simplifies parallel programming for GPU architectures. OpenMP is used to retrieve image files to reduce the execution time of the system, and DIC is implemented with CUDA to improve computing performance.

4. METHODS

The proposed system relies upon a novel HTP suitable for painting on the surface of a structure. The location of this pattern affects the accuracy of estimation of the displacement of the structure. Figure 4 presents a method for estimating the original location (x0, y0) of the top-left corner of the HTP. First, two acquired images are selected and denoted as *S* and *P*, with *S* being an undeformed image, and *P* being an ideal HTP. A simple difference method compares *S* against *P* pixel by pixel, and returns an image *D* of the differences between the images under comparison, consistent with Eq. (2). The difference in pixel brightness is represented as an integer value within the range 0 to 255. Two pixels are considered exactly identical if the difference between the intensities of the two images is zero. Though the differences in the background will tend to be close to zero, the differences between non-background pixels will be relatively large.

$$D = |S - P| \tag{2}$$

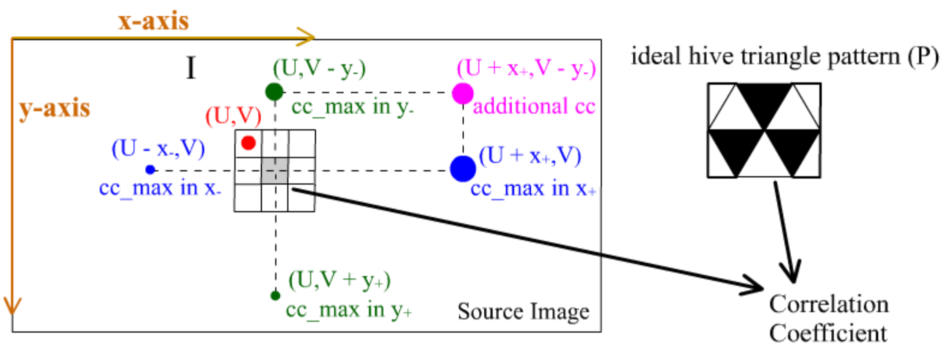

Fig. 4 Pattern location is estimated by comparing the source image with the ideal HTP using DIC coefficient. The three largest circles (green, blue, and pink) indicate the three largest correlation coefficients. The highest correlation coefficient is calculated in x_+ direction.

Another method for estimating the location of a pattern is the six point reference (SPR) method. If we assume that the target image block is located in the center of the HTP, the sum of the corresponding six weighted pixel values can be obtained from the predefined distance from the center to the bottom of six triangles in the HTP. The six pixels (P1~P6) located at a predefined distance (d) from the center and with predefined weights (+1, -1). Generally, the six pixels consist of three black dots and three white dots. In an ideal situation, the grey values of the black and the white dots are respectively 0 and 255, and the reference value pursuant to Eq. (3) is 765. The maximum reference value that corresponds to the coordinates is located close to the center of the HTP. As the specific distance (d) is within the range of the HTP, the difference between the reference values for different distances is not excessively large, and the common statistical concept of standard deviation can therefore be used to analyze multiple reference values for various distances. In this case, a smaller standard deviation indicates that the central position is more likely to be the real center of the HTP. However, when the central position c is located in an entirely black or white block, the standard deviation is also very small, and this may lead to a false positive being obtained. Therefore, a threshold level of SPR is selected as a baseline to assist with locating the center of the HTP. Based on the results of the present and previous experiments, a threshold of 70% of the ideal reference value (765) works well.

$$SPR_d = P2 + P4 + P6 - P1 - P3 - P5 \tag{3}$$

The approximate coordinates of the pattern as determined by Eqs. (4)-(6) are located using a mean-max method. This involves calculating the means of all columns and rows in the image D. The maximum means of the columns and rows are used as the x and y coordinates, respectively.

$$\overline{X}_i = \frac{\sum_{j=1}^{n} D(i,j)}{n}, \overline{Y}_j = \frac{\sum_{i=1}^{m} D(i,j)}{m} \tag{4}$$

$$\tilde{X} = U \quad \text{where } \overline{X}_U > \overline{X}_i \text{ for } i = 1 \sim m \text{ and } i \neq U \tag{5}$$

$$\tilde{Y} = V \quad \text{where } \overline{X}_V > \overline{X}_j \text{ for } j = 1 \sim n \text{ and } j \neq V \tag{6}$$

Our algorithm creates an ideal HTP for use in calculating the coefficient of correlation between a source image and an ideal pattern. The correlation coefficient has many potential formulations, but the following one is used here:

$$C = \frac{\sum\sum\left[\left(f - \langle f \rangle\right)\cdot\left(g - \langle g \rangle\right)\right]}{\left[\sum\sum\left(f - \langle f \rangle\right)^2\right]\cdot\left[\sum\sum\left(g - \langle g \rangle\right)^2\right]^{1/2}} \tag{7}$$

In Eq. (7), *f* and *g* are pixels of image blocks that represent sub-blocks of the source image and the target image, respectively, and $\langle \cdot \rangle$ is the mean operator. The minimum difference between the coordinates of image blocks is 1; therefore, the precision of this method is expressed as an integer pixel value, in the case of original images. Target images, on the other hand, are broken down into a sub-pixel scale, as sub-pixel analysis can improve the precision of the method.

Finally, a bilateral search can determine the two maximal correlation coefficients that correspond to x and y coordinates, and an additional coordinate can be determined from these. For example, the coordinate of the maximal correlation coefficient that corresponds to the x-axis is in the positive direction (right), and the coordinate of the maximal correlation coefficient that corresponds to the y-axis is in the negative direction (up). An additional point is set at the aforementioned coordinates and the correlation coefficient at that point is computed (Fig. 4). Notably, the correlation coefficient of the new point may be less than the previous maximal correlation coefficient. Therefore, the three largest correlation coefficients are evaluated simultaneously to determine the adaptive direction of motion; i.e., a case in which cc_max in x+ is 0.8 and cc_max in x- is 0.3 can indicate that the adaptive direction is a positive direction along the x-axis.

The initial location (U,V) becomes the new location $(U+x_{+},V)$ in the next iteration. Through the proposed approach's iterative process, the final maximal correlation coefficient is determined and the coordinates of the pattern are found.
Figure 5 presents a formula for a particular pixel value at sub-pixel resolution. The maximal correlation coefficient can be evaluated in terms of displacement at sub-pixel accuracy. Given that pixel grayscale values in this work range from 0 to 255, a pixel value precision of 0.1 can be deemed acceptable. A target block that is closer to the pattern of a target image generally implies that the correlation coefficient of the target and reference blocks is larger. The algorithm attempts a bilateral process to obtain the maximal correlation coefficient. Much computing time was wasted in the former iterative process, so parallel processing can be deemed preferable.

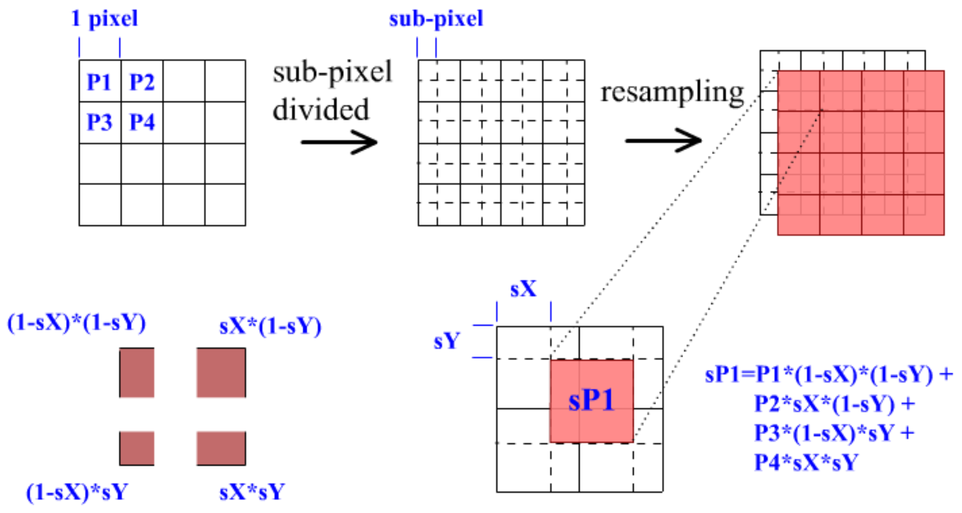

Fig. 5 Formula (sP1) for estimating the gray level of sub-pixels. For a block that must be resampled owing to sub-pixel displacement, the gray level of every pixel must be recalculated using this formula.

For acquired images, the original unit of length used in the measurement of image displacement is the integer pixel. The actual length (L) of the HTP is a known quantity, and the pixel width (W) of the HTP is evaluated using our measurement system. Actual structural displacement can be calculated using the product of the estimated pixel displacement and a pixel ratio (R_p), which is obtained from Eq. (8). Time-history displacements are thus all estimated in the digital image measurement system.

$$R_p = \frac{L}{W},\ (u,v) = (d_x, d_y) * R_p \tag{7}$$

5. EXPERIMENTAL RESULTS

In our experiment, the DIC method was implemented on an NVIDIA GTX-580 GPU with 512 CUDA cores and 1536MB of GDDR5X memory. In addition to the GPU, the workstation consisted of two Intel Xeon X5660 2.8 GHz, each of which had six physical cores, 12 logical cores, and 48GB RAM.

5.1 Results of simulation

This experiment involved two image resolutions, as shown in Figure 2. First, the camera lens focused on the second floor at a high resolution. Figures 1(b), 1(c) and 2(c) present sample captured images with high resolution. The 1940 El Centro earthquake was simulated using a Quanser Shake Table II, and the time-history displacement was estimated from the measurements made using our digital image measurement system. The curve of LVDT was very similar to that estimated using the digital image measurement system, and the peak frequencies calculated using FFT (fast Fourier transform) were 0.1258944579 and 0.1259031566, respectively. The relative error in the frequency was -6.90952E-05, and the root mean square (RMS) error was 0.020523298.

5.2 Performance of parallel computing

Five image-block sizes, 161×41, 201×41, 201×61, 201×101, 201×201, 301×201, 301×251, 301×301 and 401×401, were used in the analysis program, and the computing performance was evaluated for each of them. Figure 6 presents the execution times of the DIC method when using twice the amount of data about a simulated building for each image-block size. Three comparisons were made, using sequential processing and CUDA. In the chart, a larger image block reflects greater computational time; it reveals that CUDA outperformed sequential computation. The execution-time ratio is defined as the ratio of the execution time of sequential processing to that of CUDA. In ascending order from small to large image blocks, the execution time ratios we obtained were 11.8, 13.63, 19, 26.47, 40.99, 49.38 and 52.65.

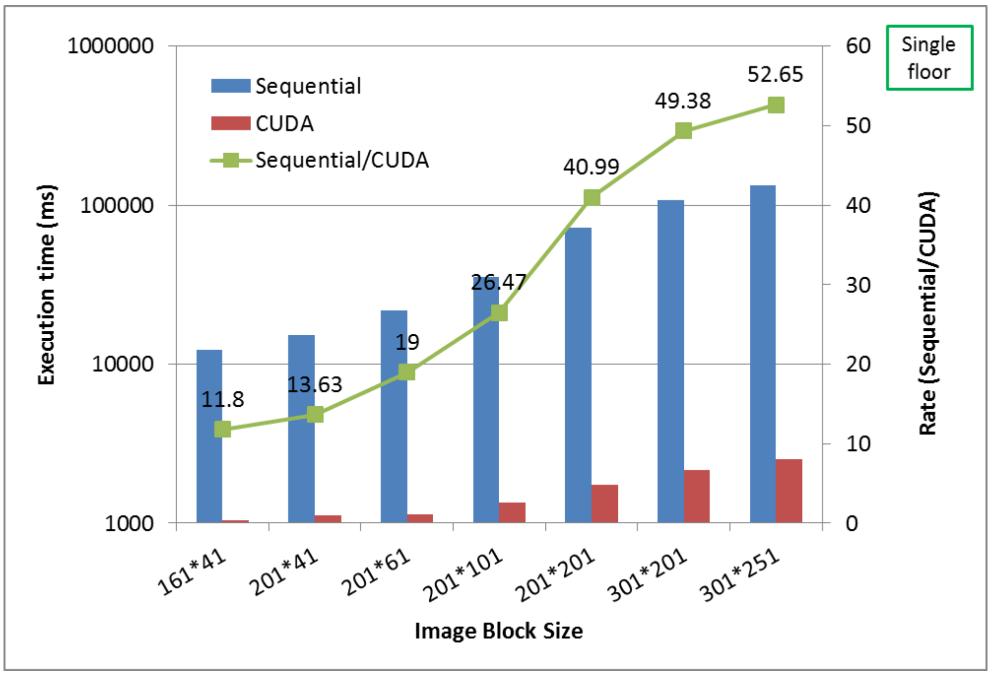

Fig. 6 Execution times of DIC in the proposed digital image measurement system, for the amount of data about a simulated building with various image-block sizes.

To increase the effectiveness of parallel computing, the amount of data of about the simulated building was increased to four times using data copying. Figures 6 and 7 present the execution time and performance, respectively, for one and four levels. The computing performance curve in Fig. 7 is similar to that in Fig. 6. In ascending order from small to large image blocks with four levels, the execution-time ratios were 14.29, 16.83, 22.17, 31.97, 46.54, 55.86, 59.83, 62.98 and 71.57. The ratios in Fig. 7 indicate that parallel computing is suitable for processing the large amounts of data required by our system.

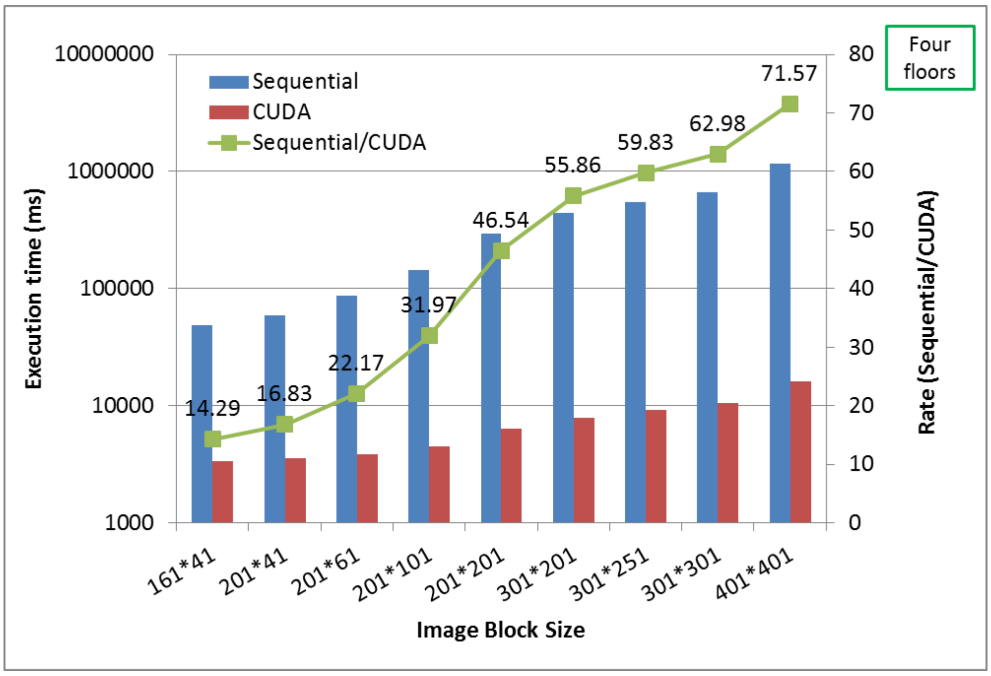

Fig. 7 Execution times of DIC in the measurement system, for four times the amount of data about a simulated building with various image-block sizes.

6. CONCLUSIONS

Our experimental results indicate that the novel digital image measurement system proposed in the present study is a workable means of making non-contact measurements, and reduces the overall time for estimating the displacement of structures by using GPU parallel processing. According to the results of the displacement-measurement experiments in our previous study, the time-history displacement estimated using our digital image system is very similar to that estimated using LVDT, indicating that the DIC method is efficient and accurate. System computing performance can be improved, relative to sequential processing, using GPU-based parallel computing. In our experiment, the GPU-based parallel computing technique was 71.57 times as effective as single-thread image processing.

In short, the DIC-based method is efficient, accurate, and suitable for parallel computing, and the CUDA-enabled GPU was easy to use for this purpose. Moreover, this GPU's performance improved as the amount of data increased, suggesting that the system did not reach its maximum performance during our experiment. However, more data require more storage memory, and excessive use of GPU memory causes the kernel function to crash. Therefore, the load balance of memory usage is a serious issue that must be adequately addressed in future research.

ACKNOWLEDGEMENTS

The authors would like to thank the Ministry of Science and Technology (former the National Science Council) of the Republic of China, Taiwan, for financially supporting this research under Contract No. NSC 100-2221-E-009-097 and MOST 104-2221-E-009-048-MY2.

REFERENCES

Adeli, H., Hung, S.L. (1993), "A Concurrent Adaptive Conjugate Gradient Learning Algorithm on MIMD Shared Memory Machines," The International Journal of Supercomputer Applications, **7**(2), 155-66.

Amodio, D., Broggiato, G.B., Campana, F., Newaz, G.M. (2003), "Digital speckle correlation for strain measurement by image analysis," Experimental Mechanics, **43**(4), 396-402.

Çelebi, M. (2002), "Seismic instrumentation of buildings (with emphasis on Federal buildings)," United States Geological Survey, Technical report No. 0-7460-68170, Menlo Park, Calif, USA.

Cook, S. (2013), "CUDA Programming: A Developer's Guide to Parallel Computing with GPUs," San Francisco: Morgan Kaufmann.

Hung, S.L., Adeli, H. (1992), "Parallel Backpropagation Learning Algorithms on Cray Y-MP8/864 Supercomputer," Neurocomputing, **5**(6), 287-307.

Hung, S.L., Adeli, H. (1994), "A Parallel Genetic/Neural Network Learning Algorithm for MIMD Shared Memory Machines," IEEE Transactions on Neural Networks, **5**(6), 900-909.

Lindholm, E., Nickolls, J., Oberman, S., Montrym, J. (2008), "NVIDIA Tesla: A Unified Graphics and Computing Architecture," IEEE Micro, **28**(2), 39-55.

Lu, Y.C., Hung, S.L., Lin, T.H. (2014), "Integrating a Hive Triangle Pattern with Subpixel Analysis for Noncontact Measurement of Structural Dynamic Response by Using a Novel Image Processing Scheme," The Scientific World Journal; Article ID 375210.

Pan, B. and Wang, Q. (2013), "Single-camera microscopic stereo digital image correlation using a diffraction grating," Optics Express, **21**, 25056-25068.

Peters, W.H., Ranson, W.F. (1982), "Digital imaging techniques in experimental stress analysis," Optical Engineering, **21**(3), 427-431.