Journal of the Optical Society of Korea, Vol. 8, No. 3, September 2004, pp. 115-121

Efficient Algorithms to Generate Elemental Images in Integral Imaging

Sechan Oh, Jisoo Hong, Jae-Hyeung Park, and Byoungho Lee*

National Research Laboratory of Holography Technologies, School of Electrical Engineering, Seoul National University, Seoul 151-744, KOREA

(Received August 2, 2004)

In this paper, we propose a new algorithm to generate elemental images in a computer generated integral imaging system. By comparing the computing time of this algorithm with that of the existing algorithm, we prove the efficiency of this algorithm. Two more algorithms considering the finite size of each pixel are also proposed. These algorithms enhance the quality of the integrated image while generating the elemental image as fast as the existing algorithm.

OCIS codes : 100.6890, 200.4560, 220.2740

I. INTRODUCTION

Integral imaging (II) that is also referred to as integral photography, was first proposed by Lippmann in 1908 [1]. It attracted a lot of researchers because of its various advantageous features such as full-color and real-time display of a three-dimensional (3D) image within a certain continuous viewing angle without any supplementary devices [2-13]. II has two steps, pickup and display processes, to display 3D images. In Fig. 1, a detailed description of these two steps is given. In the pickup process, each elemental lens of the lens array picks up two-dimensional (2D) scenes of the object from various directions. Then the pickup device such as a charge-coupled device (CCD) captures these images. A set of these captured 2D scenes is called the elemental image. In the display process, the elemental image is displayed by a display panel, such as a liquid crystal display (LCD), and the rays emitted from the elemental image retrace the original routes to form the 3D integrated image.

In computer generated integral imaging (CGII), the pickup process is replaced by an imaginary process [2]. Elemental images of CGII are generated from the 3D data of the imaginary object, considering the characteristics of the imaginary system which will be used in the display process. The imaginary pickup process is based on ray-optics, which is a merit over the real pickup process because it is free from the diffraction patterns of the elemental lenses. Each lens in the lens

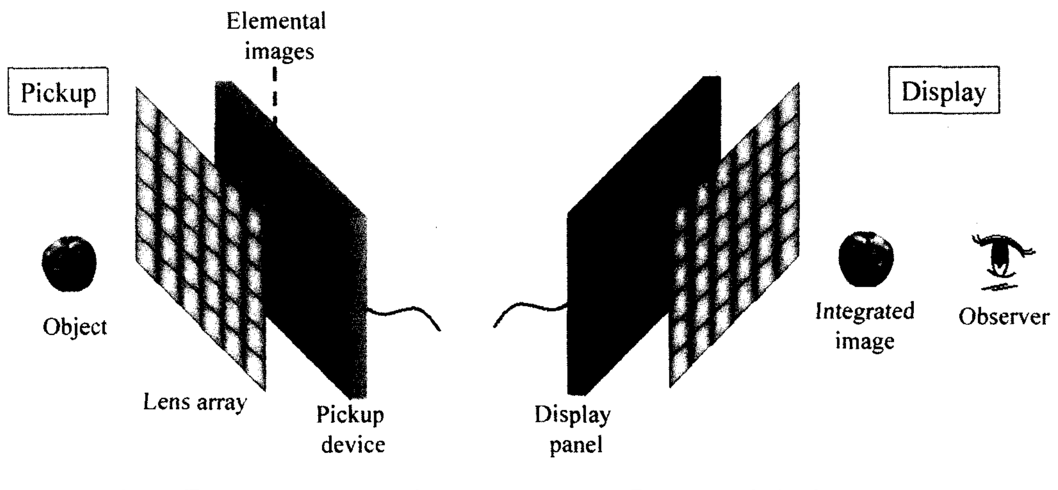

FIG. 1. The basic concept of integral imaging. Pickup and display processes.

array has its own image region on which the corresponding information is recorded or displayed.

Although recent computer processors can generate an elemental image in a short time, the time needed to generate elemental images is crucial in real application of II. This paper provides new algorithms to generate elemental images in CGII. Computer experiment shows the efficiency of these algorithms.

II. SIMPLE SAMPLING ALGORITHMS

1. Direct Algorithm

In CGII, several computer images (objects) are imagined to be located freely in 3D space. All points of the imaginary objects are picked up as the elemental image under the ray-optic assumption. The basic concept of the imaginary pickup process is shown in Fig. 2. The rays from a point of the object penetrate the centers of elemental lenses and are recorded as the elemental image. The whole elemental image is of the same size as the overall lens array and the image region of each elemental lens is the region right behind the lens.

The existing direct algorithm to generate an elemental image is as follows [2]. First, calculate the physical locations of every pixel inside the object images. Second, by using pre-calculated center locations of all elemental lenses, find out the locations where the rays from the object should be focused on the elemental image. Third, check if the focused points are inside the image region of each elemental lens. Finally, convert the physical location of the point to the pixel index of the elemental image and record the information. Because we must watch the nearest object in the display process when multiple objects overlap, this process must be executed on the farthest object first. We show the pseudo code of this algorithm in the following:

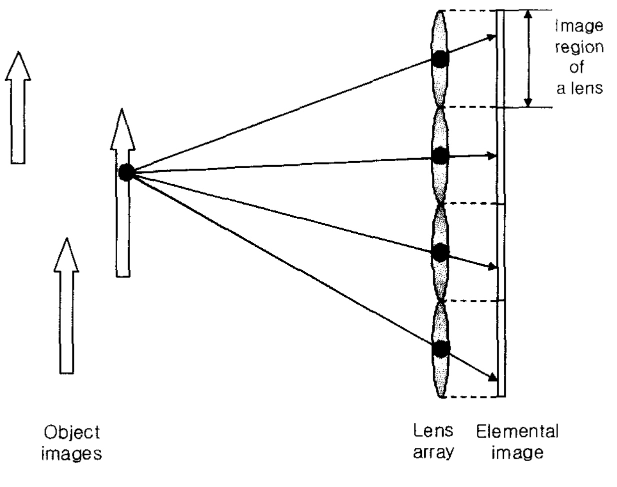

FIG. 2. Basic concept of the imaginary pickup process in CGII.

```
Direct-Algorithm
g = distance_between_display_panel_and_lens
For every_object : from_the_farthest
  L = distance_between_object_and_lens
  For every_pixel_in_the_object
    (x,y) = physical_location_of_a_pixel
    For every_elemental_lens
      (cX,cY) = center_of_the_elemental_lens
      sX = cX+(cX-x) · g/L
      sY = cY+(cY-y) · g/L
      If |sX-cX| & |sY-cY| < half_lens_pitch
        (iX,iY) = index_of_(sX,sY)_in_elemental_image
        save_the_color_of_the_pixel_at_(iX,iY) : End if
```

This algorithm critically depends on the total size of object images while the number of actually recorded pixels is limited. Moreover, several pixels of the objects over-record one pixel of the elemental image, which is also a wasteful calculation. If the total number of pixels in the objects is N, the computing time takes $O(N)$.

2. Efficient Algorithm

To minimize the wasteful calculations of the direct algorithm, we can sample points of the object in the opposite way. The detailed algorithm is as follows. First, calculate the physical location of a pixel of the elemental image. Second, find the center location of the elemental lens whose image region includes this pixel. Third, under ray-optic assumption check if the ray from the point that penetrates the center of the appropriate elemental lens meets an object. This checking process must be done from the nearest object to the farthest object and we can stop the process if we find a matched point. Finally, after finding a matched point in an object image, convert the physical location to the pixel index and record it on the elemental image. Executing this process on all pixels of the elemental image completes the whole process. The computing number needed in this algorithm does not depend on the total size of the objects and no pixel in the elemental image records the pixel of the objects over one time.

To make this new algorithm be more powerful than the direct algorithm, the process of finding the center of the appropriate elemental lens should be done in a short time. Because the arrangement of elemental lenses in a lens array is regular, we can find the accurate center location of an image region that is the center of the appropriate lens, by mathematical modular function. However, the modular function has complex internal instructions and it does not fully utilize the geometrical relation between neighboring pixels. In the rectangular lens array system, the probability that two neighboring pixels of the elemental image are included in the same image region is very high. Moreover, even though they are included in different image regions, the distance between the the center locations of the two image regions is the pitch of the elemental lens. These ideas can be represented by 'center' and 'bound' variables. The 'center' variables have the center coordinate of the elemental lens whose image region includes the running point of the elemental image and the 'bound' variables have the boundary coordinate of the image region. The basic concept of the efficient algorithm is shown graphically in Fig. 3.

```
Efficient-Algorithm
g = distance_between_display_panel_and_lens
centerX = x_coordinate_of_first_elemental_lens
boundX = centerX + half_lens_pitch
For every_x_in_the_elemental_image : from_left_to_right
  If x>boundX
    centerX += lens_pitch
    boundX += lens_pitch : End if
  centerY = y_coordinate_of_first_elemental_lens
  boundY= centerY + half_lens_pitch
  For every_y_in_the_elemental_image
    If y>boundY
      centerY += lens_pitch
      boundY += lens_pitch : End if
    For every_object : from_the_nearest
      L = distance_between_object_and_lens
      sX = centerX+(centerX-x) · L/g
      sY = centerY+(centerY-x) · L/g
      If (sX,sY)_is_inside_the_object
        sample_the_color_of_(sX,sY)
        break_innermost_for_loop : End if
```

This is the pseudo code that fully utilizes the geometrical relation between pixels. Generally, the pitch of one pixel is much shorter than the pitch of an elemental lens. Therefore, most pixels just pass the *if* ($x > boundX$) and *if* ($y > boundY$) statement without modifying the center and bound variables. Moreover, this code uses the fact that pixels of the same x-index and different y-indexes have the same value in *centerX* and *boundX* variables. In conclusion, by less than 4 elemental instructions of the computer processor, one pixel can find the correct center location of the appropriate lens, which makes the efficient algorithm be powerful. The computing time of the efficient algorithm takes $O(c)$, where c is a constant.

3. Computer Experiment

We compared the time needed to generate elemental images of the two algorithms. The lens array of the system used in the experiment consists of 13 by 13 rectangular elemental lenses, and the pitch of an elemental lens is 10 mm. The focal length of the elemental lens is 22 mm, and the pickup device is assumed to be located in the plane 26.48 mm behind the lens array. The horizontal pitch of a pixel in the display device is 0.31 mm and the vertical pitch is 0.32 mm. Figure 4 is the experimental result for several objects.

As expected, the efficient algorithm can generate elemental images in a constant time regardless of the total number of pixels in the objects. However, the time needed to generate elemental images in the direct algorithm increases linearly as the number of pixels in the objects increases. When the total number of the pixels in the objects exceeds approximately 4950, the efficient algorithm shows a better performance than the direct algorithm.

In the efficient algorithm, the ray tracing processes are done only once for each pixel of the elemental

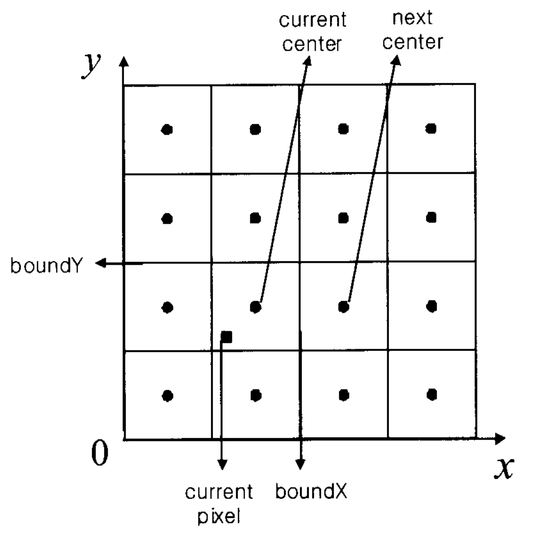

FIG. 3. Basic concept and notation of the efficient algorithm.

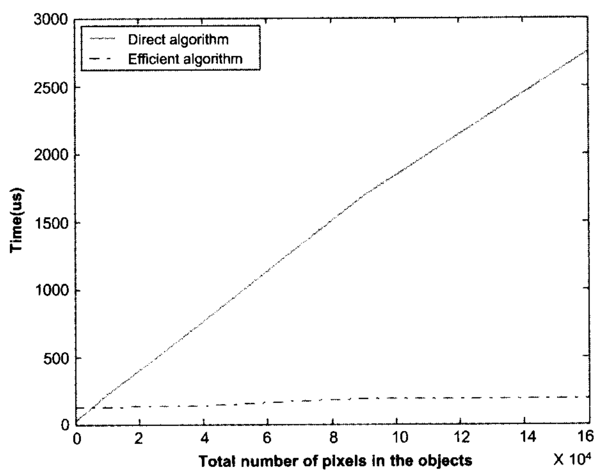

FIG. 4. Graph (input number vs. microseconds) of direct algorithm and efficient algorithm.

image, which means $13 \times 13 \times$ (*number_of_pixels_in_one_elemental_image_region*) times in this experiment. On the other hand, the direct algorithm traces all rays from a pixel in the objects that penetrate every elemental lenses and this should be done for all pixels of the objects, which means $13 \times 13 \times$ (*number_of_pixels_in_the_objects*) times. Since the number of pixels in one elemental image region is about 1010 in this experiment, 1010 pixels is a critical size of the object that determines the efficiency of the proposed algorithm and the conventional direct algorithm. In the real experiment, however, due to another processes to load the object image and build the elemental image, the size of 4950 pixels was measured as a critical point as shown in Fig. 4. Therefore, roughly speaking, it can be said that efficient algorithm can generate elemental images in a shorter time when total size of objects is larger than the size of 4~5 elemental lenses.

III. OPTIMAL SAMPLING ALGORITHMS

1. Concepts of Optimal Sampling

The algorithms of simple sampling assume that a pixel of the elemental image represents a point located at the center of the pixel, and the sampled value is also the color information of that point in the objects. However, a pixel of the elemental image has finite (i.e., non-zero) size and it corresponds to not a point but a sizable area of the object including several pixels in the object plane. We call it the sampling area of the elemental image pixel. This concept is shown in Fig. 5.

The variance between the original sampling area and the sampled value can be represented as,

$$V_s = \frac{1}{n}\sum_k (x[k]-s)^2 , \tag{1}$$

where $x[k]$ is the value of a pixel in the objects. The

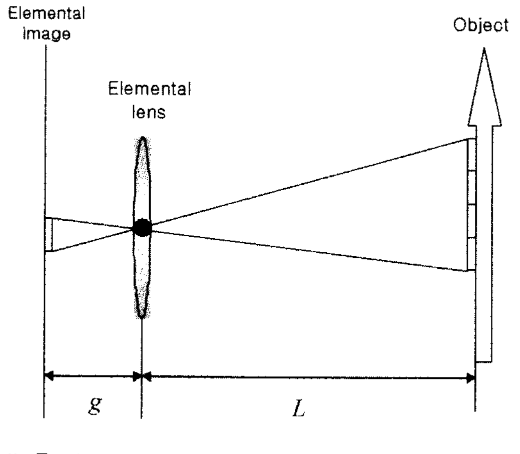

FIG. 5. Basic concept and notation of optimal sampling.

summation is conducted on all pixels which are inside the sampling area and n is the number of pixels inside the area. Algorithms of simple sampling take the value of center pixel in the sampling area as the value s. To enhance the quality of the integrated image, we have to find the optimal value that makes the smallest variance. We can simply prove that the average value of the area is always the optimal value. If we take the average value as the sampled value, the variance is

$$V_{ave} = \frac{1}{n}\sum_k \left(x[k] - \frac{1}{n}\sum_l x[l] \right)^2 . \tag{2}$$

The difference between arbitrary V_s and V_{ave} can be represented as,

$$\begin{aligned} V_s - V_{ave} &= \frac{1}{n}\sum_k (x[k]-s)^2 - \frac{1}{n}\sum_k \left(x[k] - \frac{1}{n}\sum_l x[l] \right)^2 \\ &= \frac{1}{n}\sum_k \left\{ s^2 - 2s\cdot x[k] + \frac{2}{n}x[k]\sum_l x[l] - \left(\frac{1}{n}\sum_l x[l] \right)^2 \right\} \\ &= s^2 - 2s\cdot\frac{1}{n}\sum_k x[k] + \left(\frac{1}{n}\sum_k x[k] \right)^2 \\ &= \left(s - \frac{1}{n}\sum_k x[k] \right)^2 \geq 0 . \end{aligned} \tag{3}$$

The formula above equals 0, when the value s is the average value, so the variance is minimal when we take the average value of the sampling area. We call it optimal sampling.

2. Efficient Algorithm for Optimal Sampling

We can embody the optimal sampling by simple application of the efficient algorithm. As shown in Fig. 5, the size of the sampling area depends on the z-coordinate of the object, so it is constant for an object whose thickness is much shorter than the distance between the object and the lens array. Let L be the distance between an object and the lens array and g be the distance between the display panel and the lens array. Then the width of sampling area is Lx/g and the height is Ly/g if the width of one pixel is x and the height is y. Moreover, the center of the sampling area is the point that we found in the original efficient algorithm. Therefore, we can modify the original algorithm as follows.

First of all, calculate the width and height of the sampling area for all 2D objects. For a pixel in the elemental image, after we find a matched point in the objects using the original efficient algorithm, we can easily find the pixels inside the sampling area by use of the width and height of the sampling area and its center. Then, sum the values of pixels and also record the number of summed pixels. Finally, divide the summed value by the number of summed pixels. This algorithm needs more computing than the original efficient algorithm, but the upper bound of the computing time is also limited by $O(c)$ regardless of the object size where c is a constant.

3. Direct Algorithm for Optimal Sampling

In the original direct algorithm, the fact that several pixels of the objects over-write one pixel of the elemental image and many traced rays are not actually recorded, makes the algorithm inefficient. However, the pixels of the objects that over-write one pixel of the elemental image are exactly the pixels of the sampling area explained in the last section. So, we can embody the optimal sampling by simply modifying the direct algorithm.

First, make a 2D array for summation of color information and another integer array, of the same size as the elemental image. We name this array the color array. For a pixel of the objects, after we trace the rays that penetrate the centers of elemental lenses, we sum the color information of that pixel to appropriate location of the color array and add 1 to the integer array of appropriate index. After adding the values of all pixels of one object, we divide the summed values of color array by the values of integer array which has the same index. Finally, we record the values of color array in the elemental image if it is not black (=empty). If we repeat this process for all objects in depth order (object farthest from the observer first), we can complete the algorithm.

This algorithm has almost the same number of computations with the original direct algorithm while embodying the optimal sampling.

4. Computer Experiments

Figure 6 is the graph of the time needed to generate elemental images using direct algorithm and efficient algorithm with optimal sampling under the same experimental condition used in section II. 3. Like the simple sampling algorithms discussed in section II. 3, the direct algorithm for optimal sampling shows a linearly increasing computing time and the efficient algorithm for optimal sampling shows a constant upper bound. The graph of the efficient algorithm is flattened as the number of pixels in the object increases.

The direct algorithm for optimal sampling is superior to the efficient algorithm for optimal sampling if the number of input pixels is less than 21000. If we recall that the boundary was 4950 for the original algorithms, the direct algorithm shows a better performance in optimal sampling than in simple sampling.

To confirm that the optimal sampling idea improves the quality of the integrated image, we performed a computer simulation. Because we cannot take the integrated image without any noise experimentally, computer simulation gives us more accurate data about the integrated image.

Figure 7 is the simulation result of an apple image. The image of optimal sampling shows a smoother boundary that seems more similar to the original image than the image of simple sampling. To evaluate the quality of the integrated image exactly, we use the peak signal-to-noise ratio (PSNR) that is generally accepted measure in the field of the image processing [3].

Table 1 shows the results of the calculated PSNR of

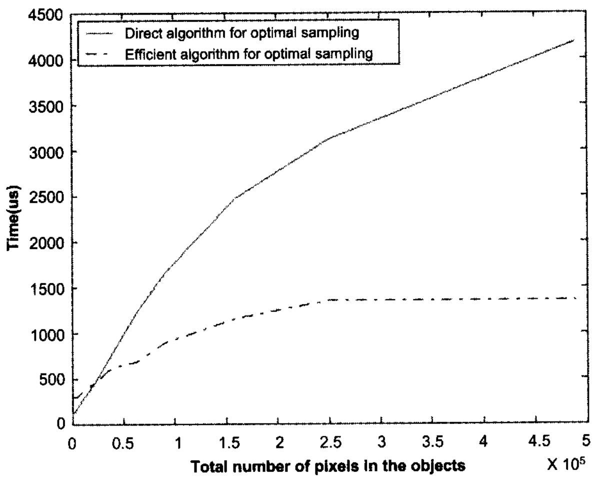

FIG. 6. Graph (input number vs. microseconds) of direct algorithm and efficient algorithm for optimal sampling.

FIG. 7. Simulation results of simple sampling and optimal sampling. (a) Reconstructed image by the simple sampling idea. (b) Original object. (c) Reconstructed image by the optimal sampling idea.

TABLE 1. Simulation results for PSNR in two sampling ideas. Simple sampling and optimal sampling.

Images	PSNR for Simple Sampling (dB)	PSNR for Optimal Sampling (dB)
Image1	21.4	24.2
Image2	15.6	19.1
Image3	20.6	23.6
Average	19.2	22.3

3 images. All images used in this simulation are 300 by 300 bmp files. It shows that the optimal sampling improves the quality of integrated images.

IV. EFFICIENT ALGORITHM VS. DIRECT ALGORITHM

The basic idea of the efficient algorithm is to record only essential information. Because the quantity of the information that an elemental image can record is limited, the computing time of the efficient algorithm has a constant upper bound. On the other hand, the direct algorithm has advantages over the efficient algorithm in that it has a simpler code and it can generate an optimal elemental image within the same time with simple sampling.

The curved lens array, the curved screen and lens array, and the embossed screen are viewing angle enhanced II systems [4-7]. Instead of planar lens array and display panel, they use curved elements to increase the size of image region of an elemental lens. Elemental images of these systems can be generated in two ways. One is the direct algorithm that picks up the information from objects to the elemental image, and the other is the efficient algorithm that functions in the opposite way. Like the planar system, if we utilize the center and bound variables in the efficient algorithm, we can find the center location of the elemental lenses in a finite time. Therefore, the efficient algorithms of these curved systems also have a constant upper bound in computing time.

However, in these curved systems the sampling area of a pixel is not parallel to the objects, so the efficient algorithm cannot be applied to the optimal sampling. The direct method, on the other hand can be simply applied to the optimal sampling even in curved systems. So if we have to apply the optimal sampling idea to enhance the image quality, the direct algorithm is superior to the efficient algorithm in complex systems.

V. CONCLUSION

In this paper, we proposed a new algorithm to generate elemental images in II and named it the efficient algorithm. By computer experiments we compared the efficiency of this algorithm with the existing direct algorithm. We also proposed optimal sampling ideas for a quality enhancement of the integrated image. Analysis of these two algorithms gave us the standard to select an appropriate algorithm in several II systems.

ACKNOWLEDGMENT

This work was supported by the Next-Generation Information Display R&D Center, one of the 21st Century Frontier R&D Programs funded by the Ministry of Science and Technology of Korea.

*Corresponding author : byoungho@snu.ac.kr

REFERENCES

[1] G. Lippmann, "La photographie integrale," *Comptes-Rendus Acad. Sci.*, vol. 146, pp. 446-451, 1908.

[2] S.-W. Min, S. Jung, J.-H. Park, and B. Lee, "Three-dimensional display system based on computer generated integral photography," in *The 2001 Stereoscopic Displays and Applications Conference Photonics West, Proc. SPIE*, vol. 4297, pp. 187-195, San Jose, USA, 2001.

[3] J. Hong, J. Kim, J.-H. Park, and B. Lee, "Analysis of the expressible depth range of three-dimensional integral imaging system," *Journal of the Optical Society of Korea*, vol. 8, pp. 65-71, 2004.

[4] J.-H. Park, S.-W. Min, S. Jung, and B. Lee, "Analysis of viewing parameters for two display methods based on integral photography," *Appl. Opt.*, vol. 40, pp. 5217-5232, 2001.

[5] Y. Kim, J.-H. Park, H. Choi, S. Jung, S.-W. Min, and B. Lee, "Viewing-angle-enhanced integral imaging system using a curved lens array," *Opt. Express*, vol. 12, pp. 421-429, 2004

[6] Y. Kim, J.-H. Park, S.-W. Min, and B. Lee, "Viewing-angle-enhanced 3D integral imaging system using a curved screen and lens array," in *11th Conference on Optoelectronics and Optical Communications*, pp. 299-300, Daejeon, Korea, 2004.

[7] J. Kim, S.-W. Min, and B. Lee, "Viewing-angle-enhanced 3D integral imaging system using an embossed screen," in *11th Conference on Optoelectronics and Optical Communications*, pp. 309-310, Daejeon, Korea, 2004.

[8] B. Lee, S. Jung, and J.-H. Park, "Viewing-angle-enhanced integral imaging using lens switching," *Opt. Lett.*, vol. 27, pp. 818-820, 2002.

[9] S.-H. Shin and B. Javidi, "Viewing-angle enhancement of speckle-reduced volume holographic three-dimensional display by use of integral imaging," *Appl. Opt.*, vol. 41, pp. 5562-5567, 2002.

[10] J.-S. Jang and B. Javidi, "Improvement of viewing angle in integral imaging by use of moving lenslet arrays with low fill factor," *Appl. Opt.*, vol. 42, pp.

1996-2002, 2003.

[11] H. Choi, J.-H. Park, J. Hong, and B. Lee, "An improved stereovision scheme using single camera and a composite lens array," *Journal of the Optical Society of Korea*, vol. 8, pp. 72-78, 2004.

[12] B. Lee, S. Jung, J.-H. Park, and H. Choi, "Recent progress in three-dimensional display based on intergral imaging," *Journal of the Optical Society of Korea*, vol. 6, pp. 133-142, 2002.

[13] S. Jung, S.-W. Min, J.-H. Park, and B. Lee, "Study of three-dimensional display system based on computer-generated integral photography," *Journal of the Optical Society of Korea*, vol. 5, pp. 43-48, 2001./ vol. 5, pp. 117-122, 2001 (erratum).