집적영상 기술에서의 요소영상 배열을 생성하기 위한 Fast 알고리즘

정회원 권 영 만*, 종신회원 김 은 수**

Fast Algorithm to Generate the Array of Elementa l Image in Integral Imaging Systems

Young-Man Kwon* *Regular Member*, Eun-Soo Kim** *Lifelong Member*

요 약

본 논문에서는 컴퓨터적으로 생성하는 집적영상 시스템에서 요소영상 배열을 생성하기 위한 새로운 fast 알고리즘을 제안한다. 이 알고리즘은 깊이 정보를 사용해서 요소영상 배열을 생성하며, 경계 영역 개념을 도입하여 경계 영역 내에 있는 복셀에 대해서만 연산을 수행하여 알고리즘의 계산 속도를 개선하였다. 제안된 알고리즘과 기존 알고리즘의 계산 시간을 이론적 및 실험적으로 비교한 결과 제안된 알고리즘이 더 효율적임을 증명하였다.

Key Words : Elemental image array, Depth information, Lens array, Integral imaging system, 3D image

ABSTRACT

In this paper, we propose a fast algorithm to generate the array of elemental image in a computer generated integral imaging system. It generates the array of elemental image using depth information, needs less computing time to produce the result by using the concept of boundary area and computing the voxel within boundary area. By comparing the computing time of proposed algorithm with that of the existing algorithm theoretically and experimently, we proved the efficiency of this algorithm.

Ⅰ. 서 론

3차원 영상을 디스플레이하기 위한 방식에는 스테레오스코피(stereoscopy), 홀로그래피(holography), 집적영상(II : Integral Imaging) 등 여러 가지가 있다. 특히, 1908년에 Lippmann에 의해서 처음 제안된 집적영상 방식을 이용하면 백색광을 이용하여 3차원 물체를 저장하고 재생할 수 있다[1-8].

한편 2002년부터 2004년까지 유럽에서 수행된 ATTEST(Advanced Three-Dimensional Television System Technologies)과제에서는 3DTV 방송을 위하여 실제 방송환경에서 2D 비디오 영상과 이에 동기화된 깊이 정보를 전송하고 이로부터 다시점 영상을 생성하여 3차원 방송을 하게 된다[2]. 즉, 다시점 영상을 전송하기 위해서는 많은 대역폭을 필요로 하고, 각각의 3DTV에 따라 콘텐츠의 구조가 바뀌어야 하기 때문에 2차원 비디오 영상과 깊이 정보를 전송해 주고 이로부터 각각의 3DTV에 맞는 콘텐츠를 생성하여 입체 영상을 보게 된다. 따라서 이와 같은 시스템에서 3차원 디스플레이 방식의 하나인 집적영상 기술을 사용하기 위해서는 각각의 집적영상 TV가 같은 특성을 갖는 렌즈 배

※ 본 연구는 지식경제부 및 정보통신연구진흥원의 대학 IT연구센터 육성·지원사업 (IITA-2008-C1090-0801-0018) 지원에 의하여 수행되었습니다.

* 을지대학교 의료산업학부 시스템 연구실(ymkwon@eulji.ac.kr), ** 광운대학교 차세대 3D 디스플레이 연구센터(eskim@kw.ac.kr)

논문번호 : kics2008-07-331, 접수일자 : 2008년 7월 30일, 최종논문접수일자 : 2008년 10월 13일

열을 사용한다고 볼 수 없기 때문에 깊이정보로부터 요소영상 배열을 생성하는 기법은 실질적인 관점에서 매우 중요한 요소라고 할 수 있다.

요소영상 배열을 컴퓨터적으로 생성하는 기법은 컴퓨터 그래픽에서 가시면(visible surface)을 검출하는 원리와 같다. 따라서 물체의 정의 내역을 다루는 것인가 혹은 투영된 상을 다루는 것인가에 따라 크게 물체 공간(object space) 기법과 이미지 공간(image space) 기법으로 구분된다. 이 중 물체 공간 기법에 근거하여 요소영상 배열을 생성하는 알고리즘으로 direct 알고리즘이 제안되었고[3], 이미지 공간 기법을 이용한 efficient 알고리즘이 제안되었다[4].

direct 알고리즘은 요소영상 배열을 생성하기 위해서 물체를 구성하고 있는 모든 점들을 요소 렌즈 배열을 통해서 영상 면에 투영한다. 따라서 객체를 구성하고 있는 픽셀의 수가 N개이면 계산 시간이 O(N)이 된다. 더욱이 객체를 구성하고 있는 여러 개의 픽셀들이 영상 면의 같은 위치에 투영되는 경우가 있어서 비효율적인 측면이 있다. 이러한 비효율적인 측면을 개선하고자 efficient 알고리즘이 제안되었다[4]. efficient 알고리즘은 영상 면의 각 픽셀에 대해서 공간에 있는 물체의 대응점을 찾아서 영상면에 기록함으로써 계산 속도를 향상시킬 수 있는 알고리즘이다.

하지만 기존의 direct 혹은 efficient 알고리즘은 물체를 구성하고 있는 면이 요소 렌즈에 수직한 아주 특별한 경우에 대하여 논하였다. 따라서 3DTV와 같이 2D 비디오 정보와 동기화된 깊이정보를 전송받아서 요소영상 배열을 생성할 경우에 알고리즘의 효율성은 달라질 수 있다. 따라서 본 논문에서는 깊이정보에 기존 알고리즘들을 적용해 본 후에 기존 알고리즘보다 더 빠른 새로운 fast 알고리즘을 제안하고, 이 알고리즘의 유효성을 실험 결과를 통해 확인하였다.

Ⅱ. 직접영상 시스템

집적영상 기술은 그림 1과 같이 크게 픽업(Pickup) 과정과 재생 과정으로 나눌 수 있다. 집적영상의 픽업 과정에서는 3차원 물체의 정보를 렌즈배열과 CCD와 같은 2차원 영상 감지기(Image sensor)를 이용하여 2차원 요소영상 배열(Elemental image array)로 기록한다. 그리고 재생 과정에서는 픽업과정에서 얻어진 2차원 요소영상

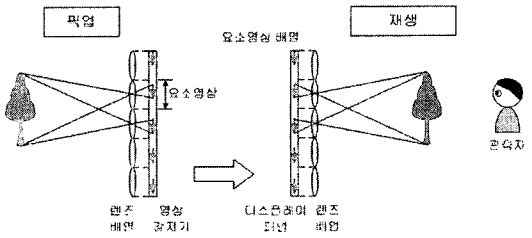


그림 1. 집적영상 시스템

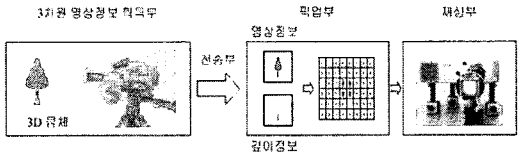


그림 2. 일반적인 집적영상 시스템

배열을 LCD와 같은 디스플레이 패널에 표현하고 이들을 다시 렌즈 배열을 통과시켜 3차원 영상으로 재생한다.

그러나 방송 시스템이나 비디오 디스플레이 시스템에서 그림 1에서와 같이 광학적으로 픽업된 요소영상 배열을 직접 사용자에게 전송하여 디스플레이하면 문제가 발생하게 된다. 이는 모든 집적영상 시스템들이 같은 특성을 가진 렌즈 배열을 사용한다고 가정할 수가 없기 때문이다. 일반적인 집적영상 시스템은 크게 실제 3차원 물체의 3차원 영상정보를 얻는 획득부, 이를 전송하는 전송부, 전송된 3차원 정보로부터 요소영상 배열을 생성하는 픽업부, 그리고 생성된 요소영상 배열을 이용하여 3차원 영상을 디스플레이를 하는 재생부로 나누어지며 그림 2와 같다.

특히 본 논문에서는 획득된 3차원 정보는 2차원 영상과 이에 동기화된 2차원 깊이정보로 표현된다고 가정한다. 따라서 재생부에서 디스플레이를 하기 전에 전송된 2차원 영상정보와 깊이정보를 사용해서 시스템에 맞는 적합한 요소영상 배열을 컴퓨터적으로 생성한 후에 이를 사용해서 디스플레이를 하는 시스템으로 가정한다. 그러므로 깊이정보로 컴퓨터적으로 요소영상 배열을 생성하는 알고리즘은 집적영상 시스템에서 매우 중요한 부분이다.

Ⅲ. 기존의 직접영상 생성 방법

깊이정보가 주어졌을 때 이를 요소영상 배열로 생성하는 방법을 생각해 보자. 깊이정보는 물체의

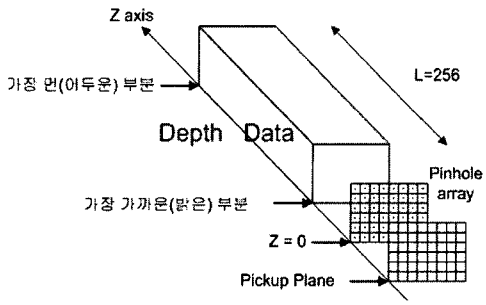


그림 3. 깊이정보로부터 요소영상 배열을 생성하는 개념적인 구조

거리 정보를 가지고 있으며, 이를 표현하기 위하여 일반적으로 명암도(gray scale) 이미지 파일로 나타낸다. 이 명암도 이미지에서 밝을수록 가까이 있는 것을 뜻하고 어두울수록 멀리 있다는 것을 뜻한다. 깊이정보가 명암도 이미지로 나타낼 때 이로부터 요소영상 배열을 생성하는 개념적인 구조는 그림 3과 같다. 깊이정보로부터 요소영상 배열을 생성하기 위해서 먼저 z=0 위치에 핀홀(pinhole) 배열을 두고 z축의 음의 방향으로 영상면을 둔다. 그리고 핀홀 배열에서 적당한 거리를 두고 깊이정보의 밝은 부분을 가까이 두고 어두운 부분을 먼 쪽에 둔다. 그런 후에 기하 광학적 레이 분석을 통해 요소영상 배열을 컴퓨터적으로 생성한다.

3.1 Direct 알고리즘

direct 알고리즘은 물체 공간 기법에 근거하여 요소영상 배열을 생성하는 알고리즘이다. 이 알고리즘은 먼저 물체내의 모든 픽셀의 위치를 구한다. 다음 단계는 모든 요소렌즈(elemental lens)의 중심 좌표를 사용해서 객체에서 발생한 광선이 요소영상 배열에서 투영되는 지점을 찾는다. 그런 후에 투영되는 지점이 각 요소렌즈의 영상범위 안에 있는 지를 체크한다. 마지막으로 투영되는 지점에 범위 안에 있으면 물체 점이 투영된 위치를 요소영상 배열의 픽셀 인덱스로 변환하고 그 위치에 물체 점의 픽셀 컬러 값을 저장한다. 이 알고리즘의 pseudo-code를 그림 4에 나타낸다[4].

```
g = distance_between_lens_and_pickup_device
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, xY) = center_of_the_elemental_lens
      sX = cX +(cX-x)*g/L
      sY = cY +(cY-y)*g/L
      If |sX-cX| < half_lens_pitch &&
          |sX - cY| < half_lens_pitch
        (iX, iY) =
            index_of_(sX, sY)_in_the_elemental_image
        save_the_color_of_the_pixel_at_(iX, iY)
      End If
    End For
  End For
End For
```

그림 4. direct 알고리즘

3.2 Efficient 알고리즘

efficient 알고리즘은 이미지 공간 기법을 이용한 알고리즘이다. efficient 알고리즘의 처리 순서는 다음과 같다. 먼저 요소영상 배열의 픽셀 위치를 선택한다. 두 번째로 이 픽셀을 포함하고 있는 요소렌즈의 중심 위치를 찾는다. 세 번째로 기하 광학의 법칙에 따라 점으로부터 선택된 요소렌즈의 중심을 관통하는 광선이 물체와 접하는 지를 체크한다. 이 체크 과정은 가장 가까이에 있는 물체로부터 먼 물체순서로 체크하며 만일 대응하는 점이 존재하면 그 과정을 정지할 수 있다. 마지막

```
g = distance_between_lens_and_pickup_device
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 = centerX + lens_pitch
    boundX = boundX + lens_pitch
  End If
  centerY = y_coordinate_of_first_elemental_lens
  boundY = centerY + half_lens_pitch
  For every_y_in_the_elemental_image :
      from_bottom_to_top
    If y>boundY
      centerY = centerY + lens_pitch
      boundY = boundY + lens_pitch
    End If
    For every_object : from_the_nearest
      L = distance_between_object_and_lens
      sX = centerX +(centerX - x)*L/g %sampling point
      sY = centerY +(centerY - y)*L/g
      If (sX, sY)_is_inside_the_object
        sample_the_color_of_(sX, sY)_in_the_object
        break_innermost_for_loop
      End If
    End For
  End For every_y_in_the_elemental_imag
End For every_x_in_the_elemental_image
```

그림 5. efficient 알고리즘

으로 물체 이미지에서 대응하는 점을 찾으면 물체의 위치를 픽셀 인덱스로 바꾸어 물체의 영상값을 찾아서 요소영상 배열에 기록한다. 이 과정을 모든 요소영상 배열의 픽셀에 대하여 수행한다. 이 알고리즘의 pseudo-code를 그림 5에 나타낸다[4].

Ⅳ. Fast 알고리즘

이 장에서는 깊이정보에 대해서 Ⅲ장에서 설명한 direct 알고리즘과 efficient 알고리즘을 사용하여 요소영상 배열을 생성하는 대신에 요소영상 배열을 빠르게 생성할 수 있는 새로운 fast 알고리즘을 제안한다.

4.1 Fast 알고리즘

fast 알고리즘은 direct 알고리즘과 같이 물체 공간 기법의 알고리즘이다. fast 알고리즘의 실행 순서는 다음과 같다. 첫 번째로 거리가 가장 먼 곳으로부터 임의의 한 점을 선택한다. 선택된 점으로부터 깊이 정보인 거리 L과 (x, y)값을 알 수 있다. 두 번째 단계는 거리 L을 사용해서 선택된 점에 대한 boundX와 boundY에 대한 경계 값들을 구한다. 이 값들이 경계(boundary)에 대한 하한 값과 상한 값이고 이를 그림 6에 나타낸다. 세 번째는 요소영상 배열에 대해서 각 요소영상의 중심점 좌표 (cY, cX)의 값을 구한다. 네 번째로 이 중심점 좌표 값이 경계 영역 내에 있는 지를 체크한다. 만일 경계 영역 내에 있으면 물체의 (x, y) 좌표가 요소영상 배열에서 대응되는 (iX, iY) 점을 찾는다. 그런 후에 물체의 영상 정보를 (iX, iY)에 저장한다. 거리가 먼 곳에 있는 픽셀 순서로 모든 픽셀에 대해서 위의 과정을 수행하면 요소영상 배열을 생성할 수 있다. 이 알고리즘의 pseudo-code를 그림 7에 나타낸다.

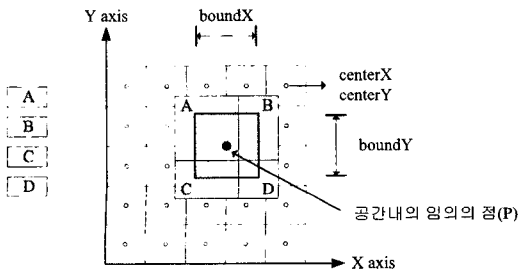


그림 6. 경계 영역

```
extract the points from the depth information
sort every_pixel_in_the_object_from_the_farthest
For every_pixel_in_the_object : from_the_farthest
  L = distance_between_pixel_and_lens    % z coordinate
  (x, y) = the coordinate of pixel in the space
  rY_lower_bound = y - y_pitch*L/(g*2)
  rY_upper_bound = y + y_pitch*L/(g*2)
  rX_lower_bound = x - x_pitch*L/(g*2)
  rX_upper_bound = x + x_pitch*L/(g*2)
  For every_elemental_lens_in_y_direction % bottom_to_top
    cY = center_y_coordinate_of_the_elemental_lens
    If  (rY_lower_bound < cY) && (cY < rY_upper_bound)
      For every_elemental_lens_in_x_direction % left_to_right
        cX = center_x_coordinate_of_the_elemental_lens
        If  (rX_lower_bound < cX) &&
            (cX < rX_upper_bound)
          sY = cY +(cY-y)*g/L
          sX = cX +(cX-x)*g/L
          (iX, iY) =
            index_of_(sX, sY)_in_the_elemental_image
          save_the_color_of_the_pixel_at_(iX, iY)
        End If
      End For
    End If
  End For
End
```

그림 7. fast 알고리즘

4.2 Fast 알고리즘의 효율성

이미지 공간 기법을 이용한 efficient 알고리즘의 경우, 물체 공간이 렌즈 배열에 대해서 몇 개의 수평면으로만 구성된 경우에는 효율적이다. 그러나 본 논문에서 사용하는 깊이 정보를 회색조 영상으로 생각하면 깊이 단계가 256개인 면으로 구성된 것으로 간주할 수 있고, 이와 같이 많은 수의 면으로 구성된 물체 공간에서는 이미지 공간 기법은 렌즈를 통과한 빛이 물체 면을 통과하는 지 혹은 반사하는 지를 모든 물체 면에 대해서 결정해야 되기 때문에 efficient 알고리즘은 효율성이 매우 낮다. 그래서 제안된 알고리즘의 효율성은 direct 알고리즘만을 비교한다.

direct 알고리즘과 본 논문에서 제안한 fast 알고리즘과의 차이점은 다음과 같다. 물체를 구성하는 임의의 점(복셀)이 주어질 경우에 direct 알고리즘은 모든 요소 렌즈에 대해서 sY = cY + (cY-y)*g/L, sX = cX + (cX-x)*g/L 연산을 수행한다. 이를 그림 8에 나타낸다. 반면에 fast 알고리즘은 먼저 rY_lower_bound=y-y_pitch*L/(g*2)와 같은 연산을 4번 수행한 후에 요소렌즈의 중심이 이 경계 영역에 속하는 경우에만 sY = cY

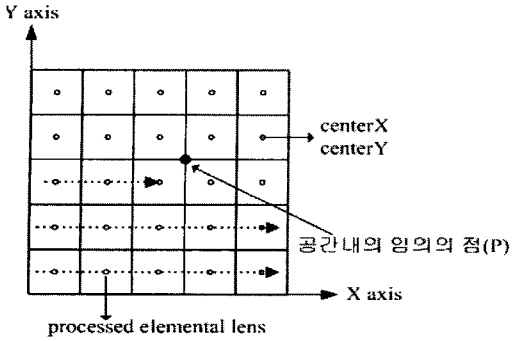


그림 8. direct 알고리즘의 처리 기법

+ (cY-y)*g/L, sX = cX + (cX-x)*g/L 연산을 수행한다. 알고리즘의 효율성을 이론적으로 고찰하기 위해서 sY = cY + (cY-y)*g/L 연산과 rY_lower_bound = y - y_pitch*L/(g*2) 연산을 같은 시간이 걸리는 연산으로 가정해 보자. 또한 3차원 물체를 구성하고 있는 점이 n개이고, 요소영상 배열이 pg개(p개의 행과 q개의 열)라고 가정한다. 그러면 direct 알고리즘은 2npq 개수만큼의 연산을 수행해야 한다. 하지만 fast 알고리즘에서는 n(4 + 2c) 만큼의 연산을 수행하면 된다. 여기서 c는 임의의 상수이고 pq에 비해서 매우 작은 수이다. 즉 c << pq를 만족하기 때문에 fast 알고리즘이 direct 알고리즘보다 매우 효율적이다.

V. 실험 및 결과 분석

본 논문에서 제안한 fast 알고리즘의 효율성을 검증하기 위해서 depth 카메라로 추출한 깊이정보와 3D MAX로 추출한 깊이 정보를 사용해서 알고리즘의 실행 속도를 측정하는 실험을 수행하였다.

5.1 Depth 카메라로 추출한 깊이정보 사용

depth 카메라를 사용하여 깊이정보를 추출하는 실험을 하였다. 깊이정보를 얻기 위해서 그림 9와 같은 'A', 'B' 그리고 'C'의 세 글자로 이루어진 3차원 물체를 사용하였다. 이들 글자의 실제크기는 약 8.5 cm × 8.5 cm이고, 깊이 카메라로부터 350 cm 위치에 놓여있다. 깊이정보는 256 단계 이상으로 양자화가 가능하나 실험에서는 256 단계로 양자화를 하였으며 추출된 깊이정보는 그림 10에 나타내었으며 크기는 740×468 픽셀이다.

이 깊이 정보로부터 fast 알고리즘을 사용해서

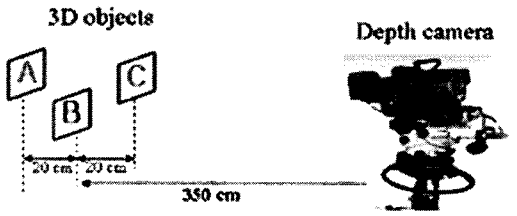


그림 9. 깊이정보를 추출하기 위한 실험 구성

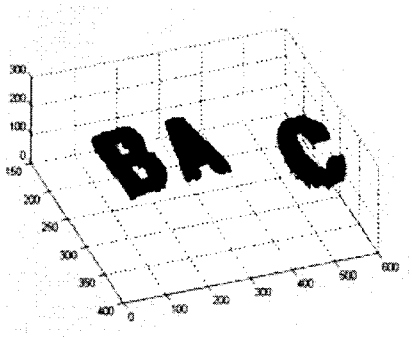
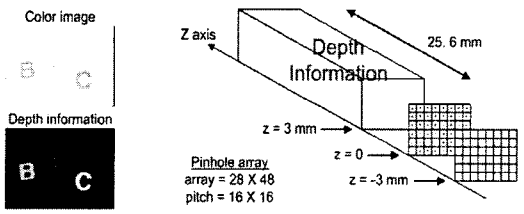
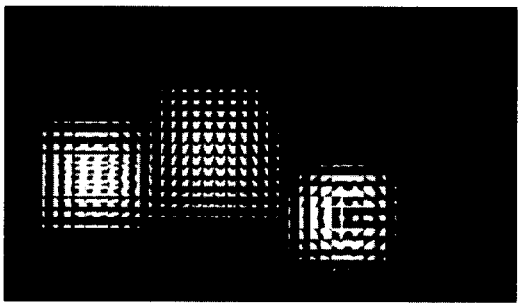


그림 10. 깊이정보



(a) 실험 구성



(b) 획득된 요소영상 배열

그림 11. fast 알고리즘에 의한 요소영상 배열

요소영상 배열을 생성하기 위한 실험 구성과 결과는 다음 그림 11과 같다.

먼저 depth 카메라의 깊이정보를 사용해서 본 논문에서 제안한 fast 알고리즘의 속도를 분석하기 위해서 사용된 연산의 수와 사용된 cpu 시간

을 비교해 보았다. 사용된 ABC 데이터의 점(복셀) 개수는 전처리가 된 후에 총 19759 개이며, 본문에서 언급한 연산의 개수는 direct 알고리즘 경우는 53112192(2*19759*28*48), fast 알고리즘 경우는 실제 카운트 한 결과 767350이었다. 본문에 제시한 연산 개수는 70배정도 적으나 Matlab 7.0에서 상대적인 시간을 측정해 본 결과 direct 알고리즘은 3.2969 sec, fast 알고리즘은 0.2188 sec가 되어 약 15배 빠른 것으로 나타났다.

5.2 3D MAX로부터 추출한 깊이정보 사용

3D MAX 프로그램으로 코끼리 두 마리를 설계하고, 이 3차원 물체인 두 마리 코끼리를 포함한 적당한 깊이에서 256 레벨로 깊이정보를 추출하였다. 추출된 깊이정보는 그림 12와 같고, 크기는 500x500 픽셀이며, 이를 사용하여 fast 알고리즘의 효율성을 실험하였다.

이 깊이 정보로부터 fast 알고리즘을 사용해서 요소영상 배열을 생성하기 위한 실험 구성과 결과는 그림 13과 같다.

3D MAX로부터 추출한 깊이정보를 사용해서 본 논문에서 제안한 fast 알고리즘의 속도를 분석하기 위해서 사용된 연산의 수와 사용된 cpu 시간을 비교해 보았다. 사용된 두 마리 코끼리 데이터의 점(복셀) 개수는 전처리가 된 후에 총 53421 개이며, 본문에서 언급한 연산의 개수는 direct 알고리즘 경우는 109406208(2*53421*32*32), fast 알고리즘 경우는 실제 카운트 한 결과 2471440이었다. 본문에서 제시한 연산 개수로는 44배정도 적으나 Matlab 7.0에서 상대적인 시간을 측정해 본 결과 direct 알고리즘은 6.9688 sec, fast 알고리즘은 0.6406 sec가 되어 약 11배 빠른 것으로 나타났다.

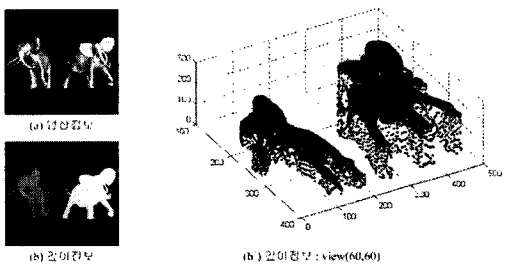


그림 12. 3D MAX로부터 추출한 영상정보와 깊이정보

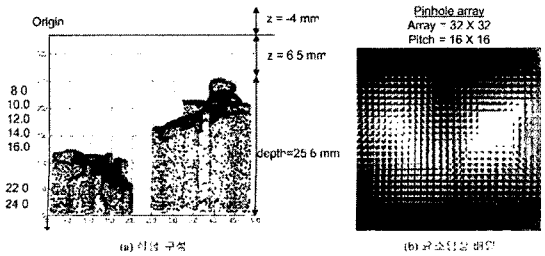


그림 13. fast 알고리즘에 의한 요소영상 배열

VI. 결 론

결론적으로, 본 논문에서는 3차원 집적 영상 기술에서 깊이 정보로부터 요소영상 배열을 생성하는 새로운 알고리즘 제안하고 이에 대해서 분석하였다. 제안한 알고리즘에 대해서 depth 카메라와 3D MAX로부터 획득한 깊이 정보에 대해서 요소영상 배열 생성 실험을 수행하고, 기존의 방법과 비교하였다. 이를 통하여 제안한 알고리즘의 계산 속도가 상대적으로 효율적임을 확인하였다.

참 고 문 헌

[1] 김은수, 이승현 공역, *3차원 영상의 기초*, 기다리, 1998

[2] Andre Redert, Mark Op de Breek, Christoph Fehn, Wijnand IJsselsteijn, Marc Pollefeys, Luc Van Gool, Eyal Ofea, Ian Sexton, Philip Surman, "ATTEST: Advanced Three-dimensional Television System Technologies," *Proceedings of the First International Symposium on 3D Data Processing Visualization and Transmission (3DPVT02) 2002*, IEEE

[3] 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 Optical Society of Korea*, Vol.5, No.3, pp.117-122, September 2001.

[4] S. Oh, J. Hong, J.-H. Park, and B. Lee, "Efficient Algorithm to Generate Elemental Images in Integral Imaging," *Journal of Optical Society of Korea*, Vol.8, No.3, pp.115-121, September 2004.

[5] F. Okano, H. Hoshino, J. Arai, and I. Yuyama, "Three-dimensional video system based on integral photography," *Opt. Eng.*, 38, 1072-1077, 1999.

[6] H. Arimoto and B. Javidi, "Integral three-dimensional imaging with digital reconstruction," *Opt. Lett.*, Vol.26, No.3, pp.157-159, 2001.

[7] S.-H. Hong, J.-S.Jang, and B. Javidi, "Three-dimensional volumetric object reconstruction using computational integral imaging," *Opt. Express 12*, 483-491, 2004.

[8] D.-H. Shin, M. Cho, K.-C. Park and E.-S. Kim, "Computational technique of volumetric object reconstruction in integral imaging by use of real and virtual image fields," *ETRI Journal*, Vol.27, No.6, pp.208-212, 2005.

권 영 만 (Young-Man Kwon) 정회원

1983년 2월 광운공과대학 전자공학과 졸업
1985년 2월 한국과학기술원 전기및전자공학과 석사
1998년 8월 한국과학기술원 정보및통신과 박사과정수료
2007년 2월 광운대학교 전자공학과 박사
1993년 3월~현재 을지대학교 의료산업학부 교수
<관심분야> 영상처리, 머신비전, 운영체제

김 은 수 (Eun-Soo Kim) 종신회원

한국통신학회 논문지 제 24권 9A참조