3차원 샘플링에 기반을 둔 볼륨랜더링 프로그램의 설계 및 구현

박재영[†] · 이병일[††] · 최흥국[†††]

요 약

볼륨랜더링은 연속적인 2차원 영상들을 기반으로 하여 3차원 데이터로 만드는 것이다. 오브젝트의 내부 영역까지도 가시화 할 수 있는 장점 때문에, 최근 MRI, PET, SPECT 같은 의료 영상의 경우 볼륨랜더링을 이용해서 진단에 많이 사용하고 있다. 본 논문에서는 볼륨랜더링을 쉽게 할 수 있도록 2차원 데이터를 바탕으로 볼륨데이터를 만드는 방법을 제시하고, 볼륨랜더링 기법을 이용해 의료 영상에 적용시켜 보았다. 또한 2차원 데이터를 추출하는 샘플링단계에서 해상도를 향상시키기 위해 linear interpolation과 cubic interpolation을 통해 볼륨랜더링 된 영상의 공간 해상도를 조절하도록 설계 및 구현하여 보았으며, 변형함수(transfer function)를 이용하여 각각의 결과를 비교하였다. 2차원 영상의 샘플링에 사용되는 interpolation 방법을 3차원 영상에 적용하여 구현하였다. 의료영상의 볼륨랜더링 기법은 3차원 입체 데이터로 구현되는 것이므로 영상 분석을 통한 진단에 크게 기여 할 것으로 기대된다.

A Design and Implementation of Volume Rendering Program based on 3D Sampling

Jae-Young Park[†], Byeong-Il Lee[††] and Heung-Kook Choi[†††]

ABSTRACT

Volume rendering is a method of displaying volumetric data as a sequence two-dimensional image. Because this algorithm has an advantage of visualizing structures within objects, it has recently been used to analyze medical images i.e, MRI, PET, and SPECT. In this paper, we suggested a method for creating images easily from sampled volumetric data and applied the interpolation method to medical images. Additionally, we implemented and applied two kinds of interpolation methods to improve the image quality, linear interpolation and cubic interpolation at the sampling stage. Subsequently, we compared the results of volume rendered data using a transfer function. We anticipate a significant contribution to diagnosis through image reconstruction using a volumetric data set, because volume rendering techniques of medical images are the result of 3-dimensional data.

Key words: volume rendering, 3D sampling, cubic interpolation

1. 서 론

최근 polygon 방식의 기존의 그래픽 기술보다 오브젝트 내부 영역도 같이 렌더링 할 수 있는 볼륨랜더링이 의료 진단에서 많이 이용되고 있다. MR 장치는 대조 해상율이 우수하고 단면 영상을 선명하게 만들 수 있어서 인체 내부 모습을 연속적인 2차원 슬라이스 영상으로 촬영해서 진단에 쓰는 경우가 많은데, 이런 영상들을 모아서 3차원 영상으로 재구성하면 더욱더 객관적이고 정확하게 진단에 쓰일 수가 있다. MRI 특징은 오브젝트 내부의 서로 다른 조직을 볼 수 있는 것이 가장 큰 장점인데 [1,2], 이런 슬라

접수일 : 2000년 9월 27일, 완료일 : 2002년 7월 12일

[†] 준회원, (주)삼건베리클 SI팀 주임
[††] 준회원, 인제대학교 전산학과 (박사과정)
[†††] 종신회원, 인제대학교 정보컴퓨터공학부 조교수

이스 영상들을 3차원으로 재구성하면 인체 내부 조직을 보다 사실성 있게 관찰 할 수 있다. 그러나 polygon 방식의 기존의 그래픽 렌더링 방법은 점과 선으로 연결하여 오브젝트를 생성하기 때문에 surface 부분만 렌더링 되어서 오브젝트 내부 영역을 가시화 할 수가 없다 [3-5]. 하지만 볼륨랜더링은 2차원 영상들을 재구성 할 때 픽셀들을 z축으로 쌓은 후 viewing ray에 따라서 샘플링과정을 거친 복셀기반으로 렌더링을 하기 때문에 오브젝트의 내부 영역까지도 렌더링 할 수가 있다. 특히 최근 몇 년간 많은 병원들이 PACS (Picture Achieving and Communication System)를 도입하면서 예전과는 달리 슬라이스 영상들을 필름으로 현상해서 일일이 보관하는 것이 아니고 디지털화 된 이미지 파일 형태로 저장하고 네트웍을 통해서 빠르게 전송하고 있다. 따라서 원하는 영상들을 언제든지 빠른 시간에 찾아 볼 수가 있고, 연속적인 슬라이스 영상들이 많이 필요한 MRI 같은 영상들도 하나의 단위로 묶어서 편리하게 볼 수 있다 [6]. 이러한 환경에서는 2차원 슬라이스 영상들을 쉽고 빠르게 모을 수가 있기 때문에 3차원 영상으로 렌더링 해서 진단에 쓰는 경우가 많이 늘어나고 있다. 2차원 영상을 기반으로 3차원 영상을 재구성하여 활용 및 진단에 사용하는 경우가 있지만 초기 영상화를 위한 기법의 조합은 결과 영상에 대한 목적에 영향을 받는다. 시뮬레이션이나 그래픽만의 목적이라면 원 데이터의 손실이나 변동이 있어도 표현만 잘 되면 되지만 의료영상의 표현에 있어서는 원 영상 데이터의 손실없이 표현될 수 있도록 하는 것이 바람직하다. 프랙탈이나 스플라인을 이용한 영상화기법들은 그래픽 특성의 효과적인 표현을 위해서 손실데이터를 인정하는 부분이 있다. 확률적인 접근을 통한 방법과 원 영상데이터의 최적의 표현을 위한 목적에서 수행되는 샘플링의 효율적인 적용이 필요하다.

따라서 본 논문에서는 MRI 슬라이스 영상들을 보다 쉽게 볼륨랜더링에 적용할 수 있도록 볼륨데이터를 만드는 방법을 제시한다. 또한 볼륨랜더링은 오브젝트 내부 영역을 가시화 하는 것이 가장 큰 장점이므로 transformation matrix를 통해서 공간해상도를 향상시키는 방법을 살펴보고, 샘플링을 적용시킬 때 transformation matrix를 효과적으로 이용하여 샘플링이 보다 정확하게 되도록 interpolation 방법을 2차원 cubic interpolation 방법에서 3차원 cubic interpolation 적용을 하여 결과를 비교하였다. 이와 같은 기법들을 이용하여 윈도우즈의 일반 PC 기반에서 향상된 해상도를 가지는 볼륨랜더링이 효과적으로 수행되도록 프로그램을 설계 및 구현하여 보았다. 또한 렌더링을 위해 구성되어야 하는 볼륨데이터의 제작을 위해 2차원 영상을 3차원 데이터 형식으로 변환하여 주는 프로그램을 만들어 실험하였다. 본 논문의 구성은 다음과 같다. 2장에서는 관련 연구 및 볼륨랜더링에 대해서 소개하고 3장에서는 interpolation과 transformation matrix를 이용한 3차원으로 샘플링 방법에 대해서 소개하고, 볼륨랜더링 알고리즘에 적용시키는 방법을 제시한다. 4장에서는 볼륨랜더링 과정에서 본 논문에서 제시하고 있는 볼륨데이터를 제작하는 방법과 프로그램 설계에 대해서 살펴보고 결과 예를 보인다. 5장에서는 결론과 향후 연구에 대해 기술한다.

2. 관련 연구

볼륨랜더링은 2차원 data set를 기본으로 3차원 그래픽 이론들을 적용시켜서 3차원 data set으로 만드는 것이다. 여기에 필수적으로 사용되는 이론들은 ray tracing, shading, transformation matrix, 원근 투영, composition등이 있다. 렌더링에 필요한 기본 데이터가 polygon이 아닌 샘플링 된 포인터인 복셀이기 때문에 기존의 컴퓨터 그래픽 렌더링 방식과는 조금 다른 과정을 거친다. 본 절에서는 기존의 볼륨랜더링 과정을 살펴보고, 보다 효과적으로 볼륨랜더링 하는데 필요한 기반 기술들에 대해서 살펴보기로 한다.

2.1 Volume rendering

볼륨랜더링은 연속적인 2차원 슬라이스들을 기본으로 하는 3차원 data set을 z축으로 쌓아서 3차원 영상으로 재구성하는 것이다 [7]. 대표적으로 영상공간(image space) 렌더링 방법과 물체공간(object space) 렌더링 방법이 있다. 영상 공간 렌더링 방법은 viewing 시점을 결정하고 영상 평면(image plane)의 각 픽셀에 ray을 통과시켜서 일정한 간격으로 픽셀의 값과 위치를 샘플링할 때, 각각의 ray가 일직선상에 놓이는 복셀들의 intensity와 opacity를 합성해서 영상을 생성해내는 알고리즘이다. 대표적인 렌더링 방법에는 광선 추적법(ray cast)이 있다 [8,9]. Ray마

다 복셀의 값을 저장해야 하므로 많은 메모리를 요구하지만, viewing ray에 따라서 자유롭게 3차원 공간에 투영할 수 있다. 영상공간 렌더링 방법에는 볼륨데이터를 순차적으로 순회하면서 복셀을 영상 평면으로 투영하여 영상을 생성하는 알고리즘으로 splatting이 이에 속한다 [10-12]. 이 방법은 구현이 힘들고, view mode에 종속적이기 때문에 3차원 공간에 투영할 때마다 다시 리샘플링을 해야한다. Shear-warp 알고리즘은 볼륨데이터의 전처리 과정에서 런 길이 부호화(run-length encoding)를 하고, 다음 단계에서 shear-warp 분해 방법을 이용하여 계산 시간을 줄일 수 있는 방법이다 [13,14]. 다음 그림 1은 일반적인 볼륨랜더링 파이프라인을 보여주고 있는데, 렌더링을 위해서는 제일 먼저 3차원 영상으로 만들 기본 데이터들을 획득하는 과정이 필요하다.

일반적으로 볼륨랜더링은 그림 1과 같은 과정을 거치는데 shading단계에서는 복셀의 intensity값을 결정하며, 랜더링 하고자하는 object의 surface와 컬러값도 이 단계에서 결정한다. Classification단계에서는 복셀의 opacity를 결정하는 단계이며 오브젝트 내부의 서로 다른 재질로 인해서 opacity 값이 다양하게 변하는데, 이런 정보들을 모아서 렌더링 할 때 내부 경계 영역을 구분할 수 있게 함으로써 오브젝트의 내부 모습을 가시화 할 수 있다. Compositing은 shading과 classification단계에서 얻어진 값들을 최종적으로 누적해서 하나의 최종 픽셀값으로 결정하는 단계이다.

2.2 Viewing ray와 voxel

Ray cast방법에서는 viewing ray와 교차하는 2차원 영상들의 픽셀들을 ray에 일직선상에 놓이도록 하기 위해서 샘플링과정을 거치면서 복셀을 생성하는데, 이 샘플링단계에서 ray에 놓일 복셀의 위치가 결정된다. 따라서 적당한 interpolation을 사용해서 리샘플링을 해주면 보다 정확하게 복셀의 위치를 결정할 수가 있으므로 공간 해상도를 향상시킬 수 있다. 다음 그림2는 viewing ray와 복셀값이 3차원 공간에서 서로 어떻게 상관 관계를 가지고 있는지 보여주고 있다.

그림 2와 같이 3차원 공간에서 각각의 ray가 슬라이스 영상들을 지나면서 ray와 일직선상에 놓이는 2차원 이미지의 픽셀들을 z축으로 쌓아서 최종 하나의 픽셀값으로 결정을 한다. 이때 viewing ray는 고정이 되어 있지 않으며 간격도 2차원 영상의 픽셀들의 위치와 다르다. 따라서 정확하게 복셀을 지나가지 않고 조금씩 오차가 생기기 때문에 주변의 복셀 값을 고려해서 ray의 일직선상에 복셀이 놓이도록 적당한 interpolation을 사용해서 리샘플링을 해주어야 한다. 이렇게 리샘플링과정을 거친 point들을 다시 복셀로 결정해서 렌더링 하면 viewing ray에 따라서 3차원 공간에 오브젝트를 보다 정확하게 투영할 수 있다. 이때 2차원 영상과 마찬가지로 어떤 interpolation을 사용하느냐에 따라서 최종 이미지의 해상도가 결정되는데, 볼륨랜더링은 3차원 공간에서 interpolation이 되어야 하므로 x, y축으로 2차원 interpolation하고 다시 z축으로 한번 더 interpolation을 해주면 오브젝트 내부의 공간 해상도를 개선할 수 있다.

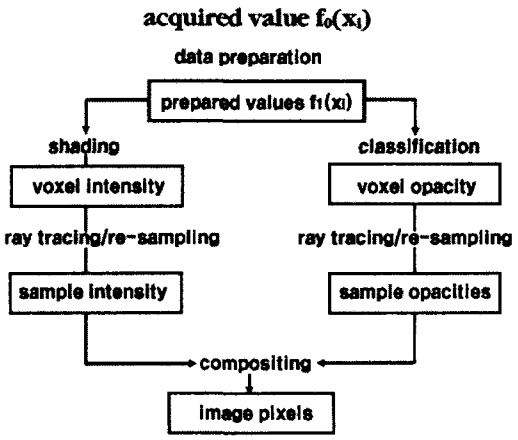


그림 1. Overview of Volume Rendering Pipeline

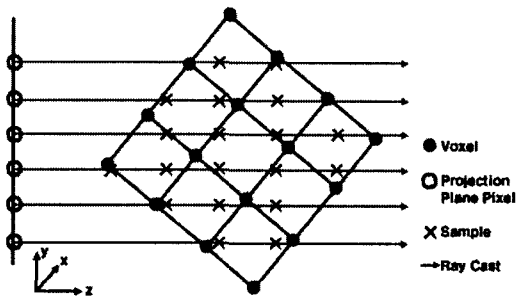


그림 2. Transformed Image Space and Mapping

3. 볼륨랜더링을 위한 3차원 tri-interpolation

앞에서 언급한 바와 같이 볼륨랜더링은 오브젝트 내부 정보를 가시화 하는 것이 가장 큰 목적이기 때

문에 여러 가지 전처리 과정을 거친다. 이러한 과정 중에서 본 절에서는 공간 해상도를 결정하는 리샘플링과정에서 cubic interpolation과 같은 고차수 interpolation을 쉽게 적용시키는 방법과 일반적으로 많이 쓰이는 linear interpolation방법과 cubic interpolation방법을 적용시켰을 때의 공간 해상도에 대해 비교 설명한다.

3.1 Interpolation을 위한 요구 사항

먼저 3차원 공간에서 linear나 cubic같은 고차수 interpolation을 적용시키려면 복셀의 위치를 정확하게 알아야 한다. Ray cast방식으로 rendering하는 영상 공간방식에서는 viewing ray에 따라서 샘플링 될 복셀이 결정되기 때문에 transformation matrix를 쉽게 이용 할 수 있다.

그림 3처럼 ray가 오브젝트 내부의 어느 지점을 지나고 있는지를 알 수 있으면 이 지점을 기준으로 interpolation을 적용시킬 수 있다. 3차원 공간에서 복셀의 위치를 얻기 위해서 다음과 같은 transformation matrix를 정의해서 이용한다. z'은 현재 ray가 지나가는 위치이며, x',y'는 이미지 픽셀의 위치를 나타낸다.

$$\begin{bmatrix} a & e & i & m \\ b & f & j & n \\ c & g & k & o \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} x \\ y \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix}$$

x-vector z-vector

식 (1)

맨 왼쪽 매트릭스는 현재 projection plane에서 오브젝트의 위치를 나타내며, 샘플링 point된 지점을 (x ,y, 0)로 정의해서 projection matrix와 곱하면 현재 ray가 지나가고 있는 지점에서 복셀 point를 (x',y',z')로 나타낼 수 있다.

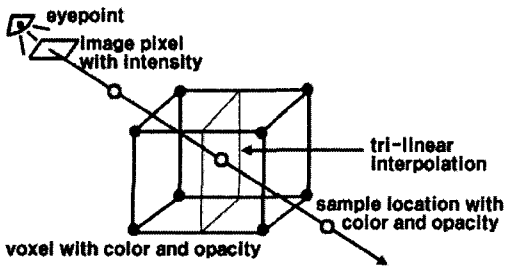


그림 3. Ray Tracing/Resampling Steps

3.2 3차원 cubic interpolation

Cubic convolution interpolation 커널은 다음 그림 4와 같이 새로운 픽셀값을 생성하기 위해서 네 개의 가장 이웃한 픽셀들을 요구한다. 따라서 한 차원에 대해서 4번의 곱셈이 필요하며, 샘플링하고자 하는 주변의 포인터를 보다 많이 고려해서 계산을 하기 때문에 좀더 정확한 값을 얻을 수가 있다 [15].

Cubic convolution function은 식 2와 같으며, 식에서 상수 a는 그림 4에서 보듯이 곡선의 모양을 결정한다. 보통 a는 -3에서 0값을 많이 사용하는데, -3에 가까울수록 영상이 sharpening하여지며 0에 가까울수록 blurring해진다.

$$\begin{cases} (a+2)\,|x|^3-(a+3)\,|x|^2+1 & 0\le|x|\le1 \\ a\,|x|^3-5a\,|x|^2+8a|x|-4a & 1\le|x|\le2 \\ 0 & 2\le|x| \end{cases}$$

식 (2)

만약 x0, x1, x2, x3 4개의 샘플링된 포인터가 있을 때, 이들 값 사이에 리샘플링이 될 새로운 값 d를 얻기 위해서 다음 그림 5와 같이 주변의 포인터를 고려해서 그림 4와 같은 커널을 적용시켰을 때 식 3과 같이 모두 더하면 d값을 결정 할 수 있다.

$$f(d) = c0 * p0 + c1 * p1 + c2 * p2 + c3 * p3$$

식 (3)

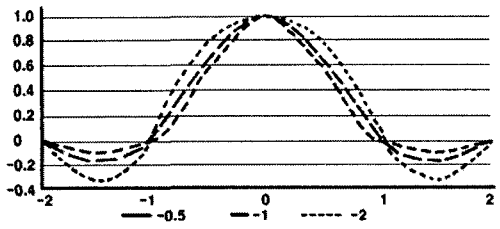


그림 4. Cubic Convolution Interpolation Kernel

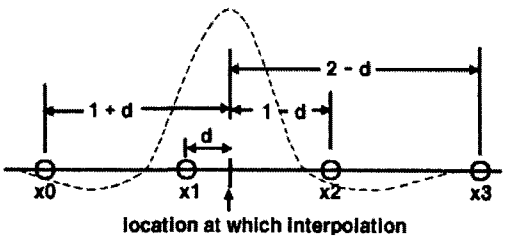


그림 5. Distance From Known Points to Interpolated Point

위의 식에서 x0 = p0, x1 = p1, x2 = p2, x3 = p3이고 c0, c1, c2, c3은 커널 계수이며 식 4와 같이 계산된다.

$$C_0 = a(1+d)^3 - 5a(1+d)^3 + 8a(1+d) - 4a$$
$$C_1 = (a+2)^3 - (a+3)d^2 + 1$$
$$C_2 = (a+2)(1-d)^3 - (a+3)(1-d)^3 + 1$$
$$C_3 = a(2-d)^3 - 5a(2-d)^2 + 8a(2-a) - 4a$$

식 (4)

식 4에서 구한 커널 계수를 식 3처럼 주위에 인접한 복셀들과 곱해서 모두 더하면 그림 5처럼 interpolation된 d값을 구할 수 있다. 위와 같은 interpolation이 3차원 공간에서 이루어지도록 하기 위해서 먼저 x축으로 interpolation을 적용시켜서 결과를 얻고 그 결과 값을 가지고 그림 6(a)처럼 다시 y축으로 interpolation 하면 된다. 마지막으로 이렇게 2차원으로 얻어진 값을 이용해서 그림 6(b)처럼 z축으로 적용시키면 3차원 tricubic interpolation이 된다.

이때 3차원으로 interpolation이 일어나기 때문에 앞에서 언급한 바와 같이 ray casting할 때 현재 어느 위치에서 리샘플링을 해야하는지 z 값을 알아야 한다. 즉 3차원 공간 속에서 현재 ray가 어느 지점을 지나고 있는지 알아야만 주변의 포인터 값을 고려해서 interpolation을 적용시킬 수가 있고 커널 계수를 식 4와 같이 구할 때 변수 d를 결정할 수 있다. 따라서 샘플링된 좌표의 위치를 알기 위해서 식 1같은 transformation matrix를 ray가 지나가는 위치를 결정할 때 사용하였다. 다음 식 5는 복셀의 위치를 나타내는 오프셋 값의 계산식이다.

$$\text{offset} = Z \times \text{ImageWidth} \times \text{ImageHeight} + Y \times \text{ImageWidth} + X \qquad \text{식 (5)}$$

식 5에서 ray의 depth는 위의 식 1에 의해서 vector.z가 되고 vector.x와 vector.y는 ray하고 만나는 슬라이스 이미지의 좌표 값이 된다. 따라서 offset는 3차원 공간에서 ray가 통과하는 지점의 좌표 값으로 나타낼 수 있다. 그림 3과 같이 현재 몇 번째 슬라이스에서 interpolation을 적용시킬지 해당 슬라이스의 좌표 값을 구할 때 위의 식 5에서 구한 vector.z로 현재 ray의 depth를 구하고 다음에 vector.x와 vector.y로 주변의 인접한 복셀의 좌표 값을 구할 수 있다.

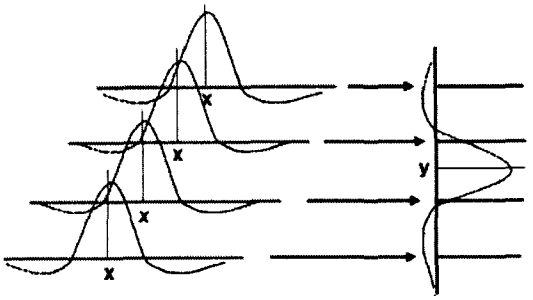


그림 6(a). Extending Cubic Convolution Dimension

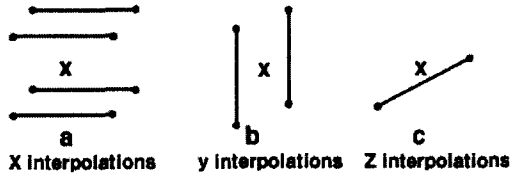


그림. 6(b) (a-c) Interpolation in Three Interpolation to Two Dimensions.

식 5에 의해서 ray가 지나가고 있는 지점을 offset이라고 한다면 식 4에 필요한 나머지 변수들은 식 6처럼 나타낼 수 있다.

$$2 - d = 2 - \text{offset}$$
$$1 - d = 1 - \text{offset} \qquad \text{식 (6)}$$
$$1 + d = \text{offset} + 1$$
$$2 + d = \text{offset} + 2$$

이제 cubic interpolation을 위한 변수들이 모두 준비되었으므로 볼륨랜더링 파이프라인에서 3차원 cubic interpolation을 적용시키는 프로시저는 다음과 같다.

4. 구현 및 결과

앞에서 제시한 볼륨랜더링 프로그램의 구현을 위해서 볼륨데이터의 획득, classification, 샘플링, shading에서 display에 이르는 일련의 과정을 다음과 같이 그림 7과 같이 구조적으로 설계를 하였다.

먼저 volume data set를 만들기 위해서 슬라이스 영상을 하나씩 읽어와서 하나의 파일에 축적해서 저장을 한다. 이렇게 만들어진 볼륨데이터에 다음과 같은 파일 헤더를 붙여서 볼륨랜더링 알고리즘을 적용시킬 때 여러 가지 변수 정보로 사용한다.

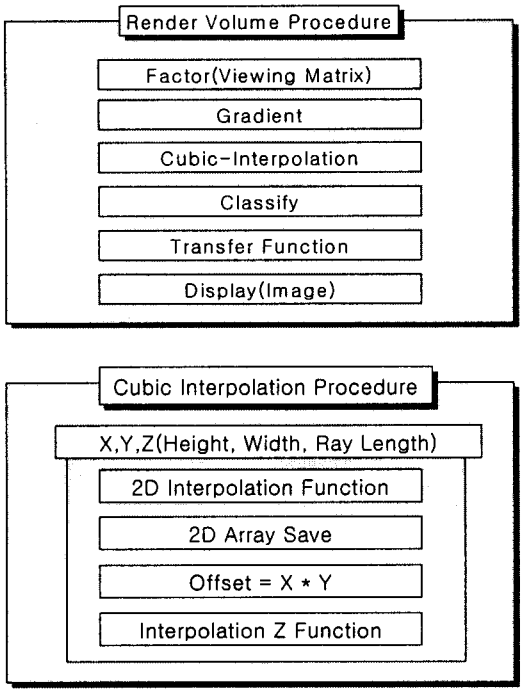


그림 7. Cubic interpolation을 적용시키는 중요한 두 개의 프로시져

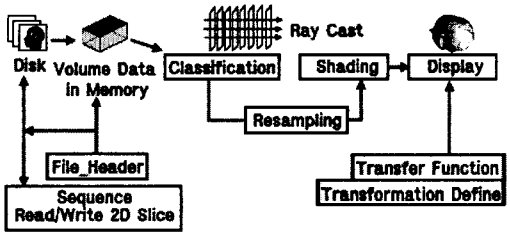


그림 8. 볼륨랜더링 프로그램 구현 설계

그림 9처럼 정의된 파일 헤더에서 width, height images는 2D 슬라이스 영상의 크기와 개수를 나타내는데, 이들 변수는 수많은 이미지로 구성된 볼륨데이터를 컴퓨터 메모리로 읽을 때 메모리 할당함수의 중요한 함수 인자로 쓰인다. 다음 리스트는 파일 헤더를 가지고 동적으로 메모리를 어떻게 할당하는지 보여준다.

```
pixelsPerSlice = _width * _height ;
pixelsPerVolume = bits_per_voxel *_images ;
bytesPerSlice = pixelsPerSlice * bytesPerPixel;
                //create buffer big enough for the volume
_Volume = (unsigned char *) calloc(pixelsPerVolume, bytesPerPixel);
```

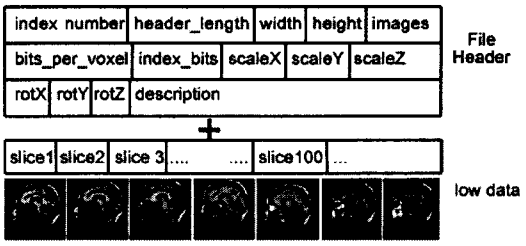


그림 9. 파일 헤더 정의 및 Low Data

그림 9와 같이 정의된 파일헤더를 바탕으로 볼륨데이터를 생성하고 그림 8과 같은 단계로 볼륨데이터를 읽어와서 위 리스트같이 파일 헤더정보를 구현에 이용하면 다음과 같은 장점이 있다.

- 연속적인 슬라이스 영상만 있으면 쉽게 볼륨데이터를 만들고 볼륨랜더링 프로그램에 적용시킬 수 있다.
- 볼륨랜더링은 아주 수많은 2차원 데이터를 바탕으로 하기 때문에 하나의 파일 형태로 묶으면 작업 단계를 간소화 할 수 있고 응용프로그램 개발에도 용이하다.
- 이진 데이터 형식이 다른 운영체제(unix, 윈도우즈)에서 볼륨데이터를 제작할 때, 파일 헤더 정보에 있는 비트수를 조절함으로써 볼륨데이터를 쉽게 읽을 수 있도록 변환 할 수 있다.

이렇게 완성된 볼륨데이터를 가지고 shading, classification, 샘플링을 거쳐서 랜더링을 수행한 뒤, 최종적으로 3차원으로 투영을 할 때는 그림 8와 같이 파일 헤더 뒷부분에 있는 scale.X 부터 rot.Z까지의 transformation matrix 정보를 이용해서 가장 중요한 부분이 화면에 나오도록 초기 값으로 이용할 수 있도록 설정하였다.

4.1 Volume Data 정의 프로그램

연속적인 2차원 슬라이스 영상만 있으면 볼륨랜더링 알고리즘에 적용될 수 있도록, 볼륨데이터를 만들고 파일 헤더를 자유롭게 정의할 수 있는 프로그램을 Visual C++ 6.0을 이용하여 제작했으며, 다음 그림은 인터페이스 예를 보여주고 있다.

그림 10에서 왼쪽 부분의 슬라이스 영상들은 볼륨데이터를 만드는 과정을 보여주고 있으며, 오른쪽 상단의 하나의 슬라이스 영상은 다음에 계속해서 추가

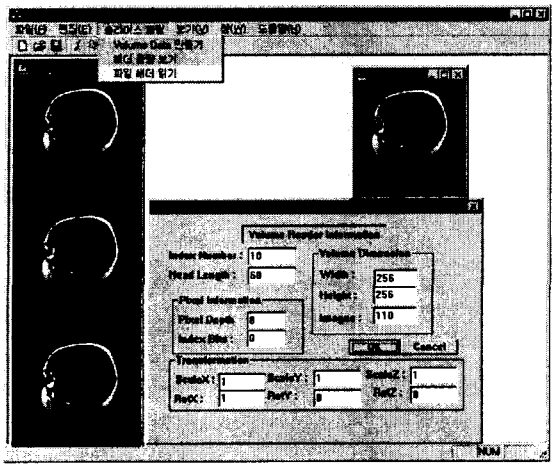


그림 10. 헤더 Volume Data Make 프로그램 인터페이스

할 이미지이다. 이런 식으로 연속적으로 2차원 영상의 픽셀값을 하나의 파일로 저장해서 그림 10의 대화상자에서 보듯이 볼륨데이터의 파일 헤더 정보를 정의해서 볼륨데이터 파일의 맨 앞에 추가하면 된다. 프로그램으로 볼륨데이터를 만드는 과정은 다음과 같다.

- 첫 번째 슬라이스 영상을 읽어 와서 가로, 세로 크기, pixel type 정보들을 읽어 온다. 이 부분에서 볼륨데이터의 x, y 크기가 정해진다. 볼륨데이터 이름을 지정하고 저장을 한다.
- 다음으로 계속해서 두 번째 슬라이스를 읽어와서 그림 8처럼 첫 번째 볼륨데이터의 파일의 맨 끝에 두 번째 슬라이스 영상의 픽셀 정보를 기록을 한다. 이런 식으로 세 번째부터 슬라이스 끝까지 계속 누적해서 저장을 한다. 이 부분에서 슬라이스의 개수를 바로 볼륨데이터의 z로 정의하면 된다.
- 앞에서 얻어진 정보를 바탕으로 볼륨데이터의 파일 헤더 정보를 transformation matrix 정보와 함께 파일 맨 앞에 기록한다.

다음 그림 11은 위와 같이 정의해서 만들어진 볼륨데이터를 보여 주고 있다. 이렇게 파일헤더를 정의해서 볼륨데이터를 만들면 볼륨랜더링 알고리즘을 적용시킬 때 연속적인 슬라이스 영상만 있으면 쉽게 렌더링 할 수 있기 때문에 2차원 영상들을 필요로 하는 볼륨랜더링 프로그램에서는 데이터의 융통성을 가질 수 있다

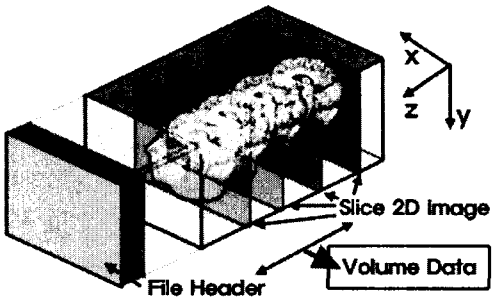


그림 11. Volume Data Structure

4.2 볼륨랜더링 구현 및 실험

오브젝트 내부 모습을 가시화 하기 위해서 변형함수를 적용시켜서 렌더링 하였으며 MRI-Head영상 110장을 기본 데이터로 사용하였다. 변형함수는 classification단계에서 이루어지는데, 픽셀의 opacity 속성을 할당하는 함수이다 [16]. Ray casting하면서 쌓아둔 픽셀의 정보를 최종 화면에 투영할 때 다음 식 7과 같이 intensity와 gradient를 같이 고려해서 변형함수에 적용시키면 오브젝트 내부의 모습을 보다 다양하게 관찰할 수 있다.

$$a_1 = O(I_i | \nabla_i | \ldots\ldots\ldots) \qquad \text{(식 7)}$$

위의 식에서 $(O\ldots)$ opacity transfer function, I_i = intensity value, $|\nabla_i|$ = gradient magnitude이며, 오브젝트 내부에서 픽셀의 변하는 정도를 나타낸다.

MIP (Maximum Intensity Processing)는 각각의 ray에서 최대 값을 가지는 복셀값을 최종 픽셀값으로 결정하는 변형함수이며 average compositing은 각각의 ray에 쌓인 복셀값의 평균을 구해서 최종 픽셀로 결정나는 변형함수이다. MRI Head영상의 경우 MIP는 인체의 skin이나 동맥에서 복셀이 최대 값을 가지므로 그림 12의 (a)처럼 surface나 혈관 모양을 볼 때 사용한다. 그림 12의 (b)는 뇌와 뇌관(brain stem)등의 구조를 좀 더 자세히 살펴보기 위해서 뇌와 뇌관 데이터가 가장 많이 있는 50-70 번째 슬라이스 20장만 따로 재구성을 해서 평균 변형함수를 사용해서 렌더링 한 결과이다. 평균변형함수는 오브젝트 내에 어떠한 조직이 가장 많이 포함되어 있는지를 볼 때 사용하는데, 그림 12의 (b)처럼 뇌와 척수 데이터가 가장 많은 픽셀정보를 가지고 있으므로 주로 뇌와 뇌관 부분의 조직을 볼 수가 있다.


(a) MIP

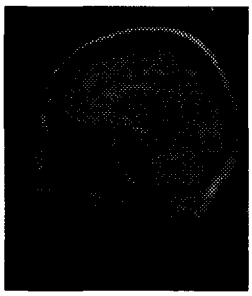
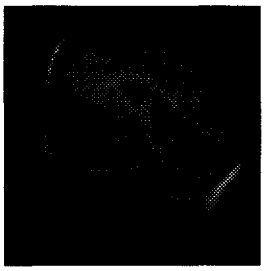

(b) Average compositing

그림 12. Transfer function 적용 이미지

4.3 Interpolation과 공간 해상도 결과

본 절에서는 3차원 cubic interpolation을 이용하여 샘플링을 하여 보았는데, cubic interpolation이 얼마나 많이 공간해상도 향상에 영향을 주었는지 linear interpolation과 비교해 보았다. 다음 그림 13는 3D-head dataset을 가지고 평균 변형함수를 적용시켜 렌더링 한 결과이다. Linear interpolation을 적용해서 랜더링 한 그림 13(a) 영상보다 cubic interpolation을 사용해서 랜더링 한 그림 13(b) 영상이 좀더 선명한 것을 볼 수 있다.

2차원 이미지의 경우 영상을 확대하면 해상도의 유지를 위해서 원본 영상보다 더 많은 픽셀이 필요하다. 이때 새로운 픽셀을 만들기 위해서 원본 영상의 픽셀값을 기본 데이터로 interpolation해서 새로운 픽셀값을 결정하면 결과 영상은 원본 영상을 많이 왜곡시키지 않고 확대를 할 수 있다. 이때 nearest neighbor나 linear같은 interpolation 보다 cubic 같은 고차수 interpolation을 쓰면 보다 해상도가 좋은 결과를 얻을 수 있다 [12]. 볼륨랜더링과 같은 3차원에서 이미지 보정 작업이 필요한 경우 transformation matrix를 이용해서 확대해서 투영할 경우에도 2차원 영상의 interpolation의 경우처럼 원래 데이터를 바탕으로 새로운 값을 결정해서 렌더링을 해야하는데, 중간 값이나, linear와 같이 바로 이웃한 두 복셀의 정보를 가지고 interpolation하는 것보다 앞 절에서 설명한 cubic interpolation과 같이 주위의 보다 많은 복셀들을 고려해서 interpolation을 하면 보다 좋은 공간해상도를 가진 결과 영상을 얻을 수 있다. 다음 그림 14는 linear와 cubic interpolation을 적용해서 렌더링한 결과 영상이다. Interpolation이 얼마나 영향을 주었는지 영상을 확대해서 투영 해보았다.


(a) linear interpolation


(b) cubic interpolation

그림 13. Average composition 50-70번째 슬라이스


(a) linear, scale 1.5배


(b) cubic, scale 1.5배

그림. 14 Scale factor를 이용해서 확대한 영상

표 1. 그림 13의 결과 이미지의 contrast 비교

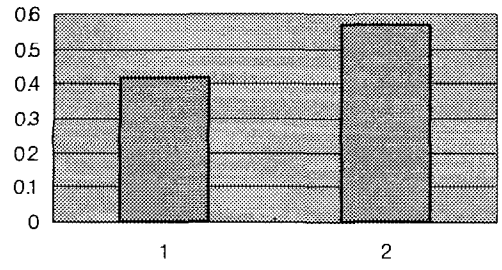


표 2. 그림 14의 결과 이미지의 contrast 비교

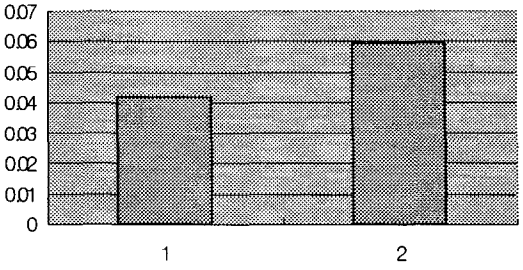
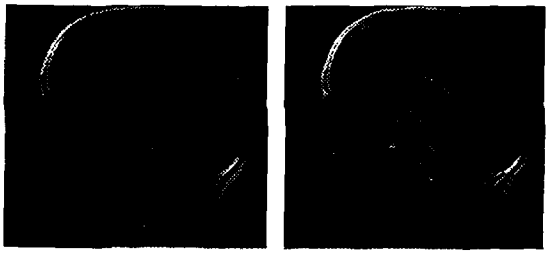


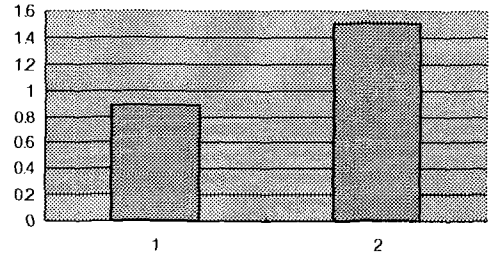
그림 14처럼 영상을 확대할수록 linear보다 cubic interpolation을 적용했을 때가 훨씬 더 공간 해상도가 좋은 것을 볼 수 있다. 좀더 자세하게 평가하기 위해서 위의 그림 14 영상을 소벨 필터를 사용해서 segmentation을 한 결과 영상을 그림 15에 나타내었다.



(a) linear interpolation (b) cubic interpolation

그림. 15 Sobel Filter 적용시킨 영상

표 3. 그림 15의 결과 이미지의 contrast 비교



위 그림 15와 같이 linear 보다 cubic interpolation을 사용해서 3차원 공간에 1.5배 확대 투영했을 때 이미지의 흐림 정도가 linear interpolation을 사용했을 때 보다 edge 부분에서 적게 나타나는 것을 볼 수 있다.

5. 결론 및 향후연구 과제

최근 MRI와 CT같은 단층 촬영 기기의 발달과 병원의 PACS 구축으로 3차원 영상에 대한 볼륨랜더링 기법의 사용이 더욱 확산될 것으로 기대되며, PC 성능이 매우 좋아짐에 따라 볼륨랜더링 같이 매우 많은 연산을 요구하는 computer graphic 프로그램도 쉽게 사용 할 수 있게 될 것으로 기대한다. 따라서 본 논문에서는 PC기반에서 볼륨랜더링 할 수 있는 프로그램을 설계 및 제작했으며, 2차원 슬라이스 영상을 볼륨 데이터로 쉽게 만들고 읽을 수 있도록 파일 헤더를 따로 정의해서 볼륨데이터에 정의해서 붙일 수 있도록 했다. 볼륨랜더링은 2차원 영상에서 볼 수 없었던 부분을 3차원영상으로 재구성하는 것이 가장 큰 장점이기 때문에 본 논문에서는 볼륨데이터를 쉽게 제작할 수 있도록 하여 슬라이스 영상들이 볼륨랜더링 프로그램에 융통성 있게 적용이 되도록 하였으며 리샘플링과정에서 cubic interpolation을 z축 방향으로 적용하여 3차원 공간에서 해상도를 향상시키고 확대 영상에서 좋은 결과를 보이도록 3차원 영상으로 재구성 하였다. 이 과정에서 오브젝트 내부 영역의 원하는 부분을 다양하게 관측 할 수 있도록 3차원 공간에서 복셀의 위치를 벡터값으로 표현해서 원하는 부분만 렌더링이 가능하도록 지원하고 있다. 또한 공간해상도 향상을 위해서 일반적으로 3차원 공간에 적용하기가 힘든 cubic같은 고차수 interpolation을 리샘플링단계에 적용하여 결과 영상이 보다 정확하게 렌더링 될 수 있도록 하였다.

볼륨랜더링은 기존의 polygon처럼 아직 하드웨어 가속은 지원하지 않으므로, 소프트웨어적으로 연산시간을 줄일 수 있도록 방법을 연구 해야한다. 따라서 2개 이상의 CPU가 지원이 되도록 병렬 처리 방식으로 알고리즘을 개선하여, volume pipeline을 독립적으로 수행하게 하면 매우 좋은 성능 향상을 볼 수 있을 것이다. 최근 들어 고성능 그래픽 카드들이 OpenGL 라이브러리를 지원하여 출시되고 있다. 하드웨어적인 부분을 지원 받도록 하는 방법을 통해 수행시간을 최소화시킬 수 있을 것이다. 이러한 방법론의 개발과 더불어 의료 진단에 사용될 슬라이스들을 재구성 할 경우 인체의 어느 부분을 볼륨랜더링하면 의료 진단에 도움이 되는지 전문의의 의견을 반영하는 것이 중요하며, 의료 영상의 진단에 있어서도 정상과 비정상을 결정하는 기준이 2차원 슬라이스로 진단하는 경우와 볼륨랜더링을 한 경우와 다를 수 있으므로, 3차원 영상에서 정상과 비정상을 구분하는 기준을 결정해서 3차원으로 표현하고 결과 값을 수치로 표현하는 것이 중요하다. 이러한 결과 값은 볼륨랜더링이 향후 얼마나 의료 진단에 도움이 되고 객관적인 데이터를 제공해 줄 수 있는지의 중요한 요인이 되기 때문에, 의사들의 요구사항에 맞는 의료 진단용 프로그램을 개발하는데 중요한 변수로 작용하게 될 것이다.

참 고 문 헌

[1] 이성우, 은충기, 문치웅, 박수성, 자기 공명 영상학, 여문각, 1998.

[2] 윤옥경, 김동휘, 박길흠, "스케일 스페이스 필터링과 퍼지 클러스터링을 이용한 뇌자기공명영상 분할," 멀티미디어학회논문지, Vol. 3, No. 4, pp.339-346, Aug., 2000

[3] Lorensen, W.E. and Cline, H.E., "Marching Cubes: a high resolution 3D surface reconstruction algorithm," Computer Graphics, Vol. 21, No. 4, pp.163-169.

[4] William E. Lorensen and Harvey E. Cline, "A High Resolution 3D Surface Construction Algorithm," Computer Graphics, Vol. 21, No. 4, pp.163-169, July 1987.

[5] 김성진, 문상호, "그래픽스 전용 메모리 설계," 멀티미디어학회논문지, Vol. 2, No. 1 pp.80-87, Mar. 1999.

[6] Seoul National University Hospital, "Design and Implementation of PC-based Hospital Integrated PACS in Seoul National Univeristy Hospital," http://radhome.snu.ac.kr/PACS/pacs.htm

[7] Marc Levoy, "Volume Rendering, A Hybrid Ray Tracer for Rendering Polygon and volume Data," IEEE Computer Graphics and Applications, Vol. 10, No. 2, pp. 33-40, March, 1990.

[8] Marc Levoy, "Display of Surface from Volume Data," IEEE Computer Graphics and Applications, Vol. 8, No. 5, pp.29-37, May, 1988.

[9] Marc Levoy, "Efficient Ray Tracing of Volume Data," ACM Transaction on Graphic, Vol. 9, No. 3, pp.245-261, July, 1990.

[10] D. Laur and P. Hanrahan, "Hierachical Splattings: A Progressive Refinement Algorithm for Volume Rendering," Computer Graphics, Vol. 25, No. 4, pp.285-288, 1991.

[11] L. Westover, "Footprint Evaluation for Volume Rendering," Computer Graphics, Vol. 24 No. 4, pp.367-376, 1990.

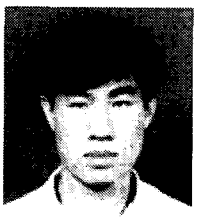
[12] I. Ihm and R. Lee, "On Enchancing the Speed of Splatting with Indexing," In Proceedings of Visualization 95, pp.27-34, Atlanta, GA October 1995.

[13] P. Lacroute and M. Levoy, "Fast Volume Rendering Using a Shear-Warp Factorization of the Viewing Transformation," Proceedings of SIGGRAPH 94, pp. 451-458, 1994.

[14] S.M. Kwon, J.K. Kim, H.W. Park, and J.B. Ra, "A Black-Based Volume Rendering Algorithm Using Shear-Warp Factorization," Journal of Biomedical Engineering Research, Vol. 21, No. 4, Aug, 2000.

[15] R. Crane, Image Processing classical & modern technique in C, 홍릉과학출판사, pp.137-148, 1997.

[16] 박재영, 이병일, 최현주, 최홍국, "3D Data Set에서 Transfer Function을 이용한 경계 영역의 가시화 방법," 멀티미디어 학회, Vol. 3, No. 1, pp.425-428, May, 2000.

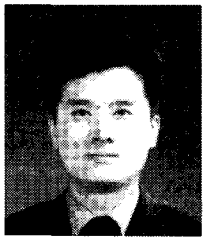


박 재 영

2001년 인제대학교 전산학과 (석사)

2001년~현재 (주)삼건베리클 SI팀 주임

관심분야 : 컴퓨터그래픽스



이 병 일

1999년 인제대학교 전산학과 (석사)

2000년~ 인제대학교 전산학과 (박사과정)

관심분야 : 영상처리 및 분석

최 흥 국

1988년 Linköping University, Computer Engineering Linköping, Sweden (공학사)
1990년 Linköping University, Computer Engineering Linköping, Sweden (공학석사)
1996년 Uppsala University, Computerized Image Analysis, Uppsala, Sweden (공학박사)
1997년~현재 인제대학교 정보컴퓨터공학부 조교수
관심분야 : 멀티미디어, 컴퓨터그래픽스, 영상처리 및 분석