◎ 논 문

화재 시뮬레이션을 위한 후처리장치 개발

허 성 범*1, 장 재 원*2, 허 남 건*3

Development of Post-Processor for Fire Simulation

S. Hur , J. Chang , N. Hur

When caught in a fire inside a building or a tunnel the generated smoke is the main cause of the bad visibility, which makes it difficult for a person to find escape route. Therefore of the fire simulation it is required to visualize the simulated results of smoke realistically form a viewpoint of a person caught in a fire. In the present study, developed is a CFD post-processor which can visualize the object through smoke from the results of CFD fire simulation. Examples of some applications of the program are shown in the paper.

Key Words: 화재해석(Fire Simulation), 후처리장치(Post-processor), 광선추적(Ray Tracing), 색혼합(Blending), 가시도 함수(Visibility Function), 투시투영(Perspective View)

1. 서 론

화재는 사람의 의도에 반하거나 고의에 의하여 발생하는 연소현상으로서 건물이나 터널 내부에서 화재가 발생할 경우 연소에 의해 발생된 연기는 안에 있던 사람을 질식시킬 뿐 아니라 시각능력을 저하시키는 주원인으로 작용하여 대피 및 인명구조에 많은 어려움을 야기시킨다.

따라서 과거 화재 시 건물이나 터널 내부의 연기 농도 분포를 알기 위해 많은 노력이 있었다. 그러나 이제까지의 화재 시뮬레이션 연구는 연기의 농도를 그래프의 형태로 나타내어 실제 화재 시 내부에 있는 사람이 느끼는 시각 상태를 나타내는데에 한계가 있다.

본 연구에서는 건물이나 터널 내부에서의 화재발생 시 연기의 농도와 사람의 시각능력과의 관계를 알아보고, 계산된 결과를 이용하여 내부의 연기농도를 3차원 투명도를 사용하여 시각화하는 후처리장치를 개발하였다. 또한 가상현실(Virtual Reality) 기술을 적용하여 시점과 시선의 변화에 따른 연기농도의 변화를 나타내었다.

프로그램 개발은 본연구실에서 CFD Package 프로그램으로 개발중인 범용 3차원 유동해석용 전/후처리 장치에 기능을 삽입하는 방법으로 진행하였다.

* 2002년 7월 11일 접수

*1 학생회원, 서강대학교 대학원 기계공학과
*2 학생회원, 서강대학교 대학원 기계공학과
*3 정회원, 서강대학교 대학원 기계공학과

2. 후처리 장치

2.1 개발환경

본 프로그램의 개발환경은 OS Windows 2000 Professional, 1.4GHz P-IV CPU, 512MB RAM으로 개발을 수행하였으며, 사용된 Compiler는 Windows 프로그래밍에 보편적으로 사용되는 Visual C++ 6.0과, MFC(Microsofr Foundation Class Library) 6.0[1]을 사용하였다. 또한 3D 그래픽스 Library는 OpenGL v1.1[2-3]

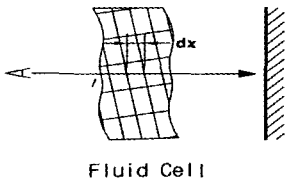


Fig. 1 The relationship between the fluid cell and the line of sight

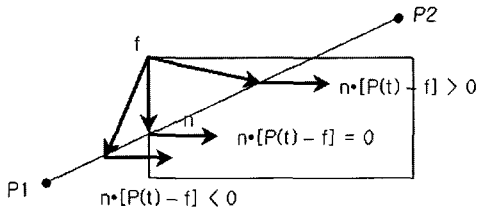


Fig. 2 2-Dimensional Cyrus-Beck algorithm

을 사용하였다. 80년대 초반 그래픽스 기술의 필요에 의해 E&S와 Silicon Graphics에 의해 Graphics API로 개발이 시작된 OpenGL은 산업계에서 가장 광범위하게 사용되는 3D Graphics API로 Unix상의 X-Window 및 Windows NT, Windows 98 등에서 사용이 가능하다. 또한 2D Graphic과는 달리 3차원 그래픽이 표준인 OpenGL은 RealTime Rendering, Shading, Texture Mapping등 다양한 기능을 가지고 있으며 물체의 표현방법은 3차원 좌표값을 기준으로 하고 있다.

2.2 Ray Tracing에 의한 연기농도 적분

본 연구에서는 격자 내부의 연기 농도 분포를 3차원 투명도로 나타내었다. 각 격자 내부의 연기의 농도를 C_i라 하고, 시점과 물체 표면 또는 벽면에 위치한 셀의 중심을 연결하는 선분이 관통하는 유체 셀의 개수를 n, 이 선이 셀 내부를 통과하는 선분의 길이를 dx로 하여 사람이 느끼는 연기농도의 강도를 Fig. 1과 다음의 식으로 나타내었다.

$$S = \sum dx_i C_i \qquad (1)$$

셀 내부를 통과하는 시선의 길이(d_x)는 3차원 Cyrus Beck Algorithm[4-5]을 사용하여 계산을 하였다. Cyrus Beck Algorithm은 한 꼭지점까지의 벡터를 f라 하고, 이 꼭지점에서부터 벡터 $P(t)$ 위의 임의의 점까지의 벡터를 $P(t)-f$라 하여 벡터 $P(t)-f$와 한 면의 Inner Normal Vector(n)의 내적 관계를 사용한 Algorithm이다. 본래 이 알고리즘은 Clipping을 위하여 개발되었지만, 본 연구에서는 이를 변형하여 셀 내부를 통과하는 선분의 길이를 구하는 데 사용하였다. 이 두 벡터의 내적 관계는 Fig. 2와 같이 나타나며, 이 중 두 벡터의 내적이 0인 것을 이용하여 선과 면이 만나는 점을 계산하였다.

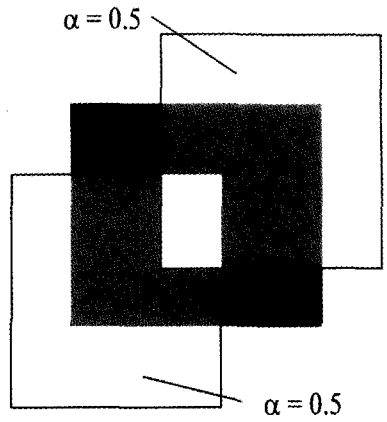


Fig. 3 Alpha blending method

2.3 색혼합에 의한 연기농도 가시화

연기농도 적분 방법은 시선의 개수와 시선이 통과하는 셀의 개수가 증가함에 따라 렌더링 시간이 길어지게 된다. 따라서 본 연구에서는 렌더링 속도향상을 위하여 본 프로그램에서 사용한 3D Graphic Library인 OpenGL내의 α 색혼합[6] 기능을 이용하여 연기의 농도를 표현하였다.

α 색혼합이란 면의 색을 결정할 때 색상정보와 함께 완전 투명인 0과 완전 불투명인 1사이의 α값을 지정함으로써 면을 투명도를 가지는 색으로 표현하는 방법으로서 간단한 예를 Fig. 3에 나타내었다. Fig. 3에서 완전 불투명인 검정색과 0.5의 α값을 가지는 흰색의 면 두 개가 겹쳐지는 경우 겹쳐지는 면의 α값으로 인해

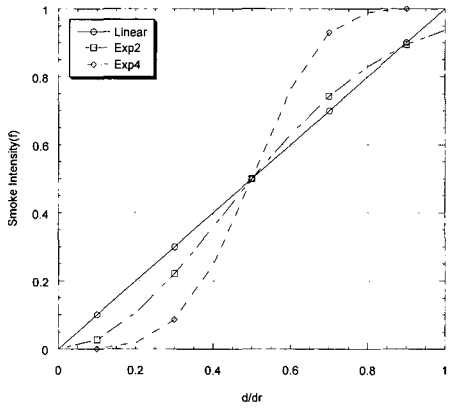


Fig. 4 Visibility function

최종적으로 렌더링되는 색은 다음과 같이 회색이나 흰색으로 보이게 된다. 따라서 격자내부의 유체셀을 연기의 색과 함께 농도값을 α값으로 가지는 면으로 렌더링 함으로써 연기의 농도를 표현할 수 있다.

2.4 연기의 농도와 시각과의 관계

화재가 발생한 건물이나 터널 안쪽에 있는 사람이 물건이나 벽을 바라보면 이 색은 연기에 의해 뚜렷하게 보이지 않게 되는데, 이 정도를 가시도 함수(Visibility Function) f로 정의하여 사용하였다. 이 때 사람은 거리에 따라 가시도가 변하기 때문에 가까운 거리에서는 가시도가 높고, 거리가 멀수록 가시도가 낮아지는 가중치를 부여하여 좀 더 사람의 시각상태에 가깝게 하였다. 본 논문에서는 OpenGL내에서 거리에 따른 안개의 농도를 표현하는 방법[6]과 대기중의 공기 질에 따른 인간의 시야[7]를 참고 하여 가시도 함수를 정의하였다. 여기서 f는 연기농도 적분 방법과 색혼합 방법에 따라 다른 정의를 사용하였다. 연기농도 적분 방법의 경우는 시선이 통과하는 셀의 연기 농도값을 적분하여 연기에 의해 물체가 완전히 보이지 않는 연기의 농도(S_r)값을 기준으로 가중치를 부여하는데 반해, 색혼합 방식의 경우는 셀마다 연기의 농도가 거리에 따른 가중치가 없는 거리(d_r)를 기준으로 농도에 가중치를 주기 때문이다. 연기농도 적분방법의 f는 다음의 식(2)와 (3)과 같이 선형함수와 지수함수의 간단한 형태로 나타낼 수 있다.

선형함수:

$$f = a\frac{S}{S_r} \quad (S \le S_r)$$
$$f = 1 \quad (S \ge S_r) \tag{2}$$

지수함수:

$$f = 1 - \exp\left(-b\left(\frac{S}{S_r}\right)^d\right) \tag{3}$$

단, 여기서 f는 0과 1사이의 값을 가지며, 위의 식에서 a, b, c는 상수로서 빛의 강도나 연기의 종류에 따라 결정하게 된다. 본 연구에서는 $a=1$, $d=2$(Exp2) 또는 4(Exp4)를 사용하였고, b의 경우는 Exp2일 경우 2.773를, Exp4일 경우는 11.09를 사용하였다. 이것은 S/S_r이 0.5일 때 Exp2와 Exp4의 f값이 0.5로 일치시키기 위해서이다. f가 0일 때는 연기가 없는 상태로 모든 것이 뚜렷이 보이는 상태를 의미하며, f가 1이면 연기의 농도가 가장 강한 상태로 모든 것이 보이지 않게 되는 상태를 의미한다.

색혼합에 사용된 가시도 함수를 식(4)와 (5)에 나타내었다.

선형함수:

$$f = a \cdot S\frac{d_{cell}}{d_r} \quad (d < d_r)$$
$$f = S \quad (d \ge d_r) \tag{4}$$

지수함수:

$$f = 1 - \exp\left(-d \cdot \left(S\frac{d_{cell}}{d_r}\right)^d\right) \tag{5}$$

이때, a, b, c는 식(2)와 (3)에 쓰인 값을 사용하였고 d_r은 광선 추적에 의한 연기농도 적분 방식의 결과와 비교하기 위하여 15를 사용하였다. 거리와 농도에 대한 f값의 변화를 Fig. 4에 나타내었다. 그래프의 가로축은 기준거리(d_r)와 시점과 셀과의 거리의 비를 나타낸다. 지수함수의 경우 가까운 곳은 선형함수에 비하여 더욱

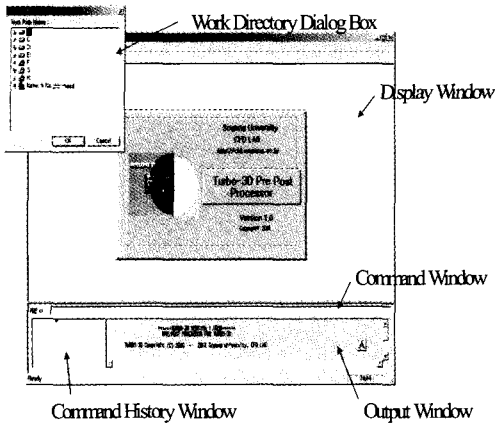


Fig. 5 Window structure of pre/post processor

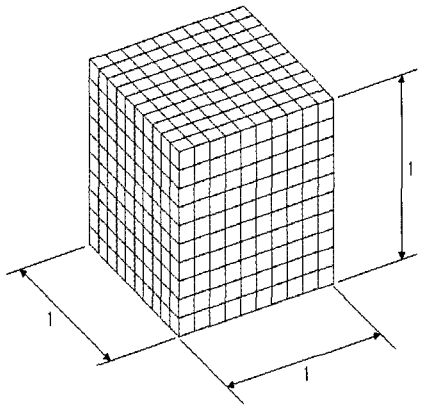


Fig. 6 Mesh for the test cell

낮은 값을 보이고, 거리가 먼 경우는 높은 값을 가진다. 이것은 사람의 시야가 거리가 가까울수록 더욱 선명한 것을 잘 나타내고 있다.

컴퓨터는 기본적으로 R(Red), G(Green), B(Blue)를 조합하여 다양한 색깔을 결정한다. 특히 OpenGL에서는 각 R, G, B색 이외에 투명도를 결정하는 α값을 가진다. 물건이나 벽면의 실제 색을 C_r이라 하고, 연기의 색을 C_f라고 할 때, 실제 내부에서 사람이 느끼는 색(C_r')은 다음 식과 같이 나타난다.

$$C_r' = C_r + (C_f - C_r)f \tag{6}$$

이것을 각각 R, G, B에 적용하면 다음 식과 같이 나타낼 수 있다.

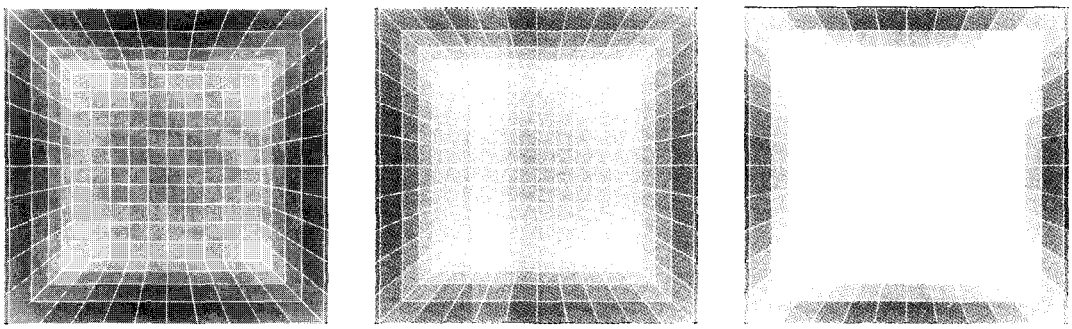
$$\begin{aligned} R_r' &= R_r + (R_f - R_r)f \\ G_r' &= G_r + (G_f - G_r)f \\ B_r' &= B_r + (B_f - B_r)f \end{aligned} \tag{7}$$

여기서 α값인 f의 값이 0이면 실제 사람이 느끼는 색은 본래의 색이 되고, 1이면 연기의 색으로 느끼게 되는 것이다.

2.5 프로그램 구성

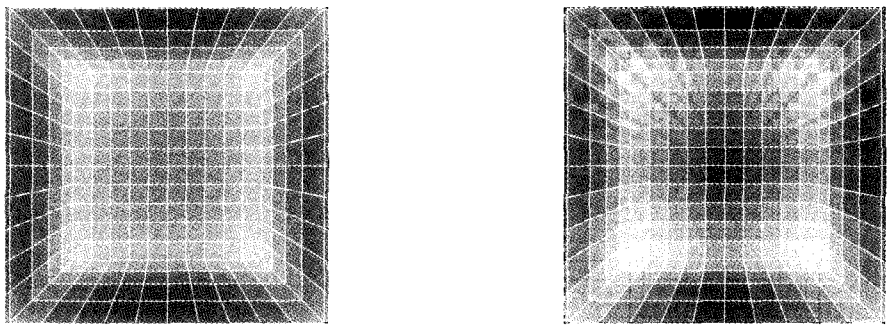
본 프로그램을 실행하면 Fig. 5와 같이 간단한 프로그램 로고가 나타난 후 작업할 디렉토리를 선택하는 대화상자가 나타난다. 작업 디렉토리를 선택하면 메인윈도우가 나타난다. 메인 윈도우는 크게 4개로 나누어져 있는데 3차원 그래픽 처리를 위한 Display Window와 명령을 입력받는 Command Window, 사용된 명령을 저장하는 Command History Window, 명령의 결과를 출력하는 Output Window로 구성되어 있다. 기본적으로는 MFC 구조를 사용하고 있으며, Display Window는 CView class를 상속받아 OpenGL을 사용할 수 있도록 수정하였다. 입체감을 위해 투시투영을 사용하였으며, Fig. 6은 투시투영을 사용하여 본 연구실에서 장대터널 해석[8]에 사용한 터널 형태를 내부에서 본 그림이다. 모든 변수의 저장을 위해 CDocument Class를 상속받아 사용하였다.

다른 프로그램과의 호환성을 위해 셀 데이터, 격자점 데이터 및 계산결과 중 연기농도 데이터인 스칼라 데이터를 읽어오는 명령을 제작하였다. 화면에 표시하기 위해서는 먼저 표시할 격자의 종류를 선택한 후 화면에 표시하게 되는데, 이때 먼저 격자점 데이터를 검색하여 각 좌표축의 최대, 최소값을 계산하여 격자의 크기를 결정한다. 다음에는 시점의 위치를 설정하고 실제 화면에 표시하게 된다. 또한 가상현실(Virtual Reality) 개념을 도입하여 키보드의 화살표 키를 이용하여 시점의 위치와 시선의 방향을 변화시킬 수 있도록 제작하였으며, 변화된 위치에서 사람이 느끼는 연기의 농도를 다시 계산하여 표시하게 하였다.



(a) Linear function(C=0.5) (b) Exponential function (d=2, C=0.5) (b) Exponential function (d=4, C=0.5)

Fig. 7 View for different visibility function



(a) Smoke intensity integral method(C=0.5) (b) Blending method(C=0.5)

Fig. 8 Compare smoke intensity integral and blending method in constant concentration distribution condition

3. 결과 및 고찰

3.1 적용예 : 등농도 분포

먼저 프로그램을 테스트하기 위해 가장 간단한 형태인 정육면체에 대해 내부의 연기 농도가 0.5로 일정한 경우에 적용하였다. 격자의 형태는 Fig. 6과 같고, 각 축으로 10개씩 1000개의 셀을 사용하였다. 광선 추적에 의한 연기농도 적분방법을 이용하여 테스트한 결과를 Fig. 7에 나타내었다. 3개의 그림 모두 농도가 일정하기 때문에 거리가 증가함에 따라 S의 값이 증가하여 면이 흐리게 보이는 것을 잘 나타내고 있다. 또한 선형함수의 경우 연기농도가 선형적으로 변하고 있는 반면에 지수함수인 Fig. 8(b)와 (c)의 경우는 시점에 가까울수록 더 선명해짐을 볼 수 있다. 특히 Exp4인 경우가 더 심한 것을 알 수 있다. 이것은 Fig. 4에 나타난 그래프와 잘 일치하고 있음을 나타낸다. Fig. 8은 같은 조건에서 선형함수에 대하여 광선 추적에 의한 연기농도 적분 방법과 색혼합을 이용한 방법을 나타낸 그림이다. 대체적으로 연기분포에 대한 경향은 비교적 잘 맞고 있다. 이것은 지수함수를 사용한 경우도 두가지 방식 모두 비교적 잘 일치하고 있다. 그러나 색혼합 방식의 경우 체크무늬가 나타나고 있음을 확인할 수 있었는데, 그 이유는 육면체의 셀이 겹치는 과정 때문이다. 이 현상은 특히 지금처럼 셀의 개수가 적고 연기의 농도가 비교적 낮은 경우에 특히 심해진다.

3.2 적용예 : 화재해석결과

다음으로는 단순한 화재해석결과에 적용하였

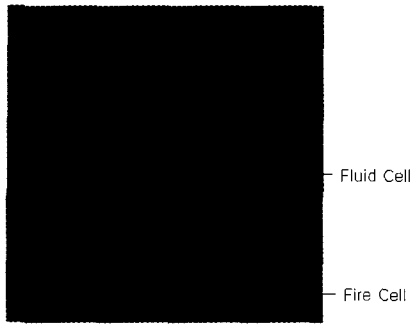


Fig. 9 Fire cell at the center of the domain

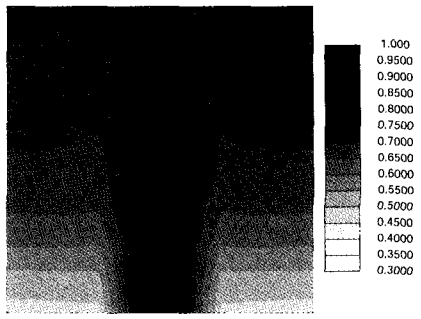
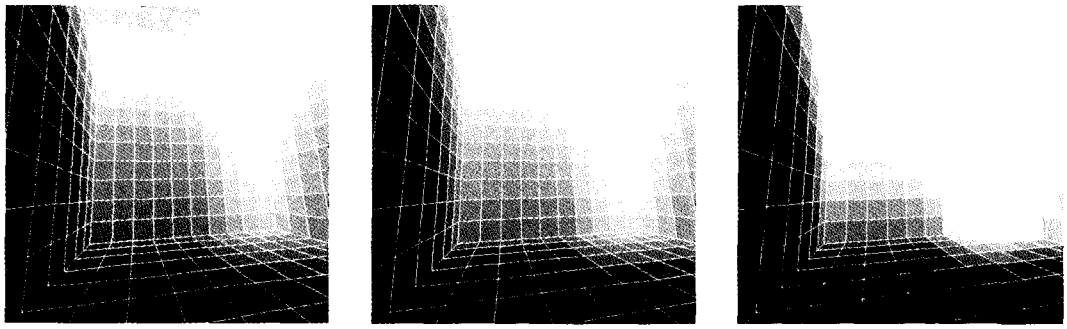
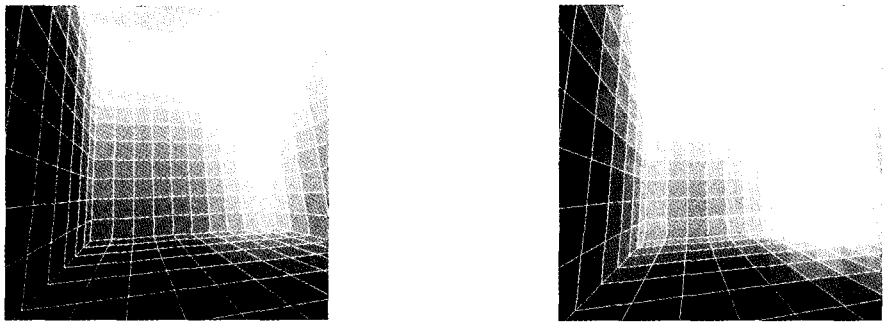


Fig. 10 Results showing the smoke concentration



(a) Linear function (b) Exponential function(d=2) (c) Exponential function(d=4)

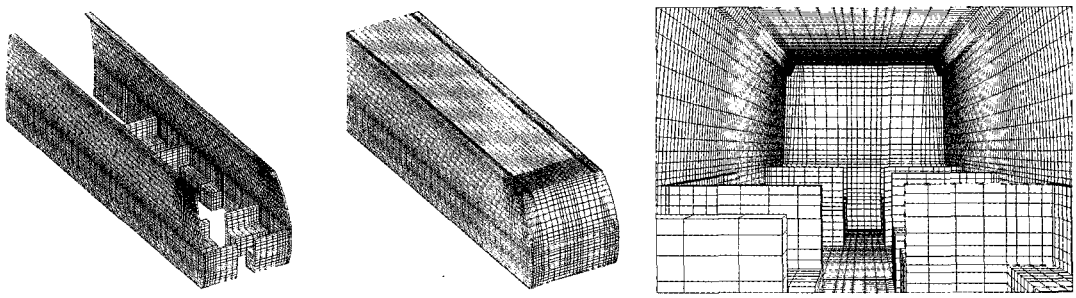
Fig. 11 Upward view for different visibility function



(a) Smoke intensity inegral method (b) Blending method

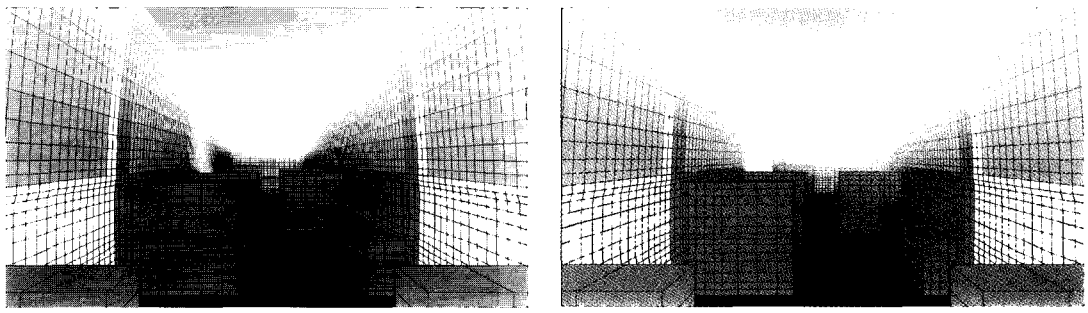
Fig. 12 Compare smoke intensity integral and blending method for fire simulation

다. Fig. 6과 같은 격자에 Fig. 9와 같이 아래쪽에 8개의 화재 셀(1MW)을 설치하였다. 계산은 Star-CD를 사용하여 계산을 수행하였으며, 시간에 따른 농도분표를 계산하여 시간이 40초일 때까지 계산하였다. 계산결과에 대한 연기농도의 분포를 Fig. 10에 나타내었다. 화재가 발생한 중앙아래부터 연기가 위쪽으로 올라가면서 연기기둥을 나타내는 것을 볼 수 있으며, 아래쪽 보다는 위쪽의 연기 농도가 진하게 나타났다. Fig. 11은 시점을 아래쪽 모서리에 위치시키고 테스트한 그림이다. Fig. 10에 나타난 것과 같은 연기의 분포를 확인할 수 있으며, 중앙의 연기기



(a) solid cells (b) fluid cells (c) Perspective view for grid test

Fig. 13 Computational grids for a train ventilation simulation



(a) Smoke intensity integral method (b) Blending method

Fig. 14 Concentration distribution inside train

둥을 잘 모사하고 있음을 확인할 수 있다. Fig. 12는 두 가지 방법을 비교한 그림이다. 색혼합 방법의 경우 등농도 분포와 마찬가지로 체크무늬가 발생하고 있음을 확인할 수 있다. 그러나 중앙의 연기기둥과 위쪽에 연기분포가 많은 것은 광선 추적을 이용한 연기농도 적분방법과 마찬가지로 잘 모사하고 있음을 확인할 수 있다.

3.3 적용예 : 경량전철 화재해석결과

실제적인 문제에 적용하기 위해 본 연구실에서 해석한 경량전철의 화재해석[9] 격자 이용하여 프로그램을 적용해 보았다. Fig. 13은 해석에 사용된 경량전철의 격자이고 총 셀의 개수는 약 150,000개이다. 화재발생시 60초가 지난 후 연기의 분포를 모사한 그림을 두 가지 경우에 대하여 Fig. 14에 나타내었다. Fig. 14는 시점을 열차의 중앙에 위치시키고 시선을 열차의 정면을 향하게 한 그림이다. 최초 화면 왼쪽부분에서 발생한 연기는 열차의 벽면을 따라 천정부로 이동한 후 천정을 따라 분포하면서 점차 아래로 내려오고 있는 형상을 잘 나타내고 있다. 특히 화재로 인한 연기가 상승하는 부분에서의 연기기둥과, 천정면에 연기농도가 높은 관계로 연기너머의 열차의 벽면이 가려져 보이지 않고 있는 것을 잘 표현하고 있다. Fig.14(b)에서는 등농도 분포나 단순 화재해석 결과와 비교해 보았을 때 체크무늬가 거의 보이지 않고 있다. 그 이유로는 연기의 농도뿐만 아니라, 셀이 많아짐으로써 격자의 겹치는 부분이 비교적 부드럽게 표현되기 때문이다.

3.4 렌더링 속도 비교

Fig. 15는 두가지 방식의 렌더링 시간에 대한 그래프이다. 경량전철의 중앙에서부터 앞쪽으로 시점을 전진시키면서 렌더링 속도를 비교하였다. 연기 적분 방법의 경우 시점이 앞쪽으로 이동되면서 시야에 보이는 면의 개수가 적어지게 되고 따라서 셀과 시선의 개수가 모두 줄어든다. 따라서 셀 내부를 통과하는 시선의 길이를 구하는 계산량이 줄어들고 그에 따라 렌더링 시

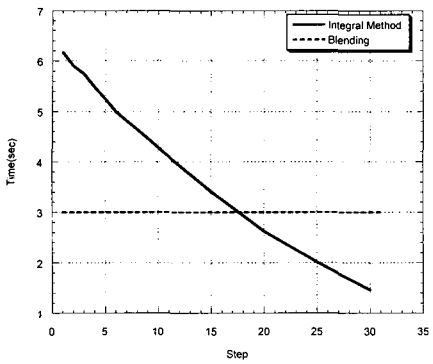


Fig. 15 Comparison of rendering time for smoke visualization

간도 비례해서 감소하고 있음을 확인할 수 있다. 그러나 스텝이 0인 경우 약 6초가 걸리고 이것은 시점을 뒤로 후진할 경우 더욱 많은 계산시간이 소요된다. 반면에 색혼합을 이용하여 연기농도를 모사하는 방법은 Fig.15에서 보듯이 시점이 이동하게 되더라도 렌더링 시간은 약 3초로 항상 일정함을 알 수 있다. 색혼합을 이용하는 경우는 단지 유체셀을 연기의 색과 농도로 렌더링 하기 때문에 시점의 이동과는 상관없기 때문이다. 따라서 렌더링 품질이 저하되더라도 셀이 많은 격자의 경우 시간을 단축시켜 렌더링 할 수 있다.

4. 결론

본 연구는 건물이나 터널 내부에 화재 발생시 내부의 연기 농도를 내부의 사람이 어떻게 느끼는지 알아보고자 3차원 투명도를 사용한 후처리 장치를 개발하였다. 각기 연기농도 적분법과 색혼합방법을 이용한 연기농도 모사 방법을 적용하였고, 각각 3가지의 가시도 함수를 정의하여 단순해석과 실제 해석 결과를 이용하여 테스트 하였다. 테스트 결과 등농도 분포 뿐만 아니라 실제적인 연기농도 계산의 경우에서도 연기의 분포를 잘 표현하고 있음을 확인하였다. 제시한 가시도 함수의 경우 선형적인 경우는 기울기가 급격히 변하기 때문에 사람의 시각과는 차이가 있다. 또한 사람은 가까운 곳에 있는 물체를 더 뚜렷이 볼 수 있다는 점에서 지수함수가 더욱 타당함을 예상할 수 있다.

또한 연기농도 적분법의 렌더링 시간 단축을 위하여 색혼합 방식을 적용하여 렌더링 시간을 단축하였다. 사용자는 경우에 따라 두가지 방법 중 하나를 선택하여 사용할 수 있다. 향후 빛의 영향과 연기 종류에 따른 가시도를 나타내는 계수를 포함함으로써 조명의 효과를 고려한다면, 좀 더 사실적인 모사를 수행할 수 있을 것이다.

후기

본 연구는 2001년도 서강대학교 교내연구비 지원과 서강대학교 산업기술연구소의 지원에 의해 수행되었으며, 이에 관계자 여러분께 감사드립니다.

참고문헌

[1] 이상엽, *Visual C++ Programming Bible*, 영진(2002).

[2] http://www.opengl.org.

[3] Kempf, F., Frazier, C., *OpenGL Reference Manual*, ADDISON-WESLEY DEVELOPERS PRESS, (1997).

[4] David, F., *Procedural Elements for Computer Graphics*, McGRAW-HILL, (1988).

[5] Cyrus, M., and Beck, J., "Generalized Two and Three Dimensional Clipping," Computers & Graphics, Vol.3, (1978) p.22-28.

[6] Richard S. Wright, Jr. Michael Sweet, *OpenGL SuperBible*, Waite Group, (2000).

[7] William C. Malm, *Introduction To Visibility*, CIRA, (1999).

[8] 권오승 외2인, "장대터널의 화재시뮬레이션 및 피난기술개발에 관한 연구," (주)대양설비기술단, (1999).

[9] 허남건, "경량전철 시스템 안전프로그램 계획을 위한 화재 방재 대책수립," 경량전철시스템 기술개발사업 2차년도 연구결과보고서, 한국철도기술연구원, (2000).