가구 배치를 위한 인터렉티브 3D Visual 디스플레이 개발

김지연, 손상수, 김병수
건양대학교 IT 학부
union13@konyang.ac.kr

A Development of interactive 3D Visual Display for Furniture Arrangement

Ji-Yeon Kim, Sang-Su Son, Byungsoo Kim
Department of IT, Konyang University

요 약

본 논문에서는 맞춤 가구 제작 및 제작된 가구의 배치를 위한 효율적이면서도 사용하기 용이한 3D Visual 디스플레이 시스템을 개발하고자 한다. AutoCAD로 만들어진 평면파일(*.dxf), 그리고 3D 제작툴로 만들어진 오브젝트 파일(*.3ds)을 이용하여 현실감 있도록 시뮬레이션 하는 것을 목표로 한다. 2D로 제작된 AutoCAD 평면 설계 파일을 이용하여 평면 파일을 3D 투시도로 변환하고 3DS파일로 구축한 객체들을 배치한 뒤 텍스처매핑, 렌더링 등을 구현하여 최대한 현실감을 낼 수 있도록 개발한다.

1. 서론

기업의 환경은 시간이 흐름에 따라 빠르게 변해가고 있다. 변화하는 예로서 공간비용을 최소화하고 업무효율을 극대화하는데 필요한 여러 가지 정보와 시설들이 활용되고 있다. 이제 과거와 같이 몇 개의 가구를 간단히 조합하여 오피스를 구성하는 시대는 지나갔다. 따라서 간단한 견적서 1장으로 사무기기를 판매하고 구매자들의 욕구를 충족시키기에는 턱없이 부족한 것이 현실이다.

하지만 국내의 많은 영세 가구업체들은 나름대로의 방법을 가지고 활용하고 있으나 극히 초보적인 수분이다. 따라서 새로운 가구 신제품에 대해 지속적인 업그레이드가 어렵고 최신의 다양한 기법들이 활용되지 못하고 있다. 또한 전문 프로그램은 가구 제작 및 배치 프로그램이기 보다는 오피스 인테리어에 초점을 맞춘 기능을 제공하고 있기 때문에 전문지식이 없는 영세 가구업체들은 그러한 기능을 익히기 또한 어려운 실정이다.

본 논문에서는 위와 같은 영세가구업체들의 당면한 문제점들을 충분히 고려하여 맞춤 가구 제작 및 제작된 가구의 배치를 위한 효율적이면서도 사용하기 용이한 3D Visual 디스플레이 시스템을 그림 1과 같이 구축하고자 한다. 3D 프로그래밍을 위하여 OpenGL[6]과 MFC를 사용하고 AutoCAD로 만들어진 평면 파일(*.dxf), 그리고 3D 제작툴로 만들어진 오브젝트 파일(*.3ds)을 이용하고자 한다.

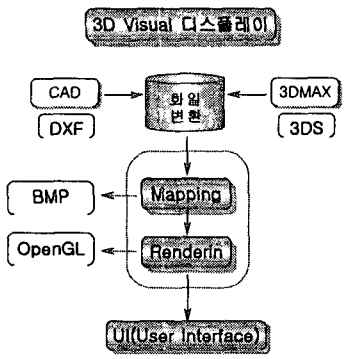

그림1. 3D Visual 디스플레이

2D로 제작된 AutoCAD 평면 설계 파일을 이용하여 평면 파일을 3D 투시도로 변환하고 3DS파일로 구축한 객체들을 배치한 뒤 텍스처 매핑, 렌더링 등을 구현하여 최대한 현실감을 낼 수 있도록 개발한다.

매핑 된 후에 사용자가 확대, 축소, 회전 등의 효과를 줄 수 있게 하는 편리한 사용자 인터페이스를 제공 하는 3D Visual 디스플레이를 개발한다.

2. AutoCAD의 역사와 저장 구조

2.1 AutoCAD의 역사

AutoCAD란 Autodesk사에서 만든 CAD(Computer Aided Design)프로그램이다. 컴퓨터를 이용하여 건축, 기계, 금형, 인테리어 디자인 등을 가장 효율적인 도면 작업 및 설계를 할 수 있고 편리하게 관리할 수 있다.

Auto CAD는 1982년 12월 라스베가스 컴덱스 쇼에서 Release 1을 처음 발표한 후, 1997년 5월에는 Windows95와 Windows NT 운영체제를 사용한 새로운 개념의 R14버전이 출시되었고, 현재AutoCAD 2002가 최신 버전이다

2.3 AutoCAD 저장구조

국내에서 범용으로 사용되고 있는 AutoCAD의 내부 데이터 저장 방식은 DWG이지만 그 저장 형태는 비공개이다. 따라서 AutoCAD 사용자와 AutoCAD를 사용하여 응용프로그램을 개발하고자 하는 개발자들은 AutoCAD로 작성된 파일의 내부 저장 형식을 파악할 수 없기 때문에 DWG로 작성된 파일의 데이터를 읽거나 저장할 수 없으며, 이것으로 인하여 AutoCAD로 작성된 데이터를 다른 응용프로그램에서 사용할 수 없다[2].

이러한 문제를 해결하기 위해 개발되어진 데이터 저장 방식이 DXF이다. DXF는 AutoCAD로 작성된 도면을 다른 시스템에 의해서 읽을 수 있는 파일 형식으로 변환하기 위하여 개발되었는데 특수한 서식의 문자열을 가진 ASCII 문자 파일로 구성되어 있어 파일 구조를 쉽게 파악할 수 있게 되어있다. 따라서 본 논문에서도 모든 AutoCAD 파일은 DXF 포맷에 의한 것 임을 밝혀둔다.

3. Loader 제작

AutoCAD로 제작된 2D 평면 설계도를 3D 투시도로 변환하여 임의의 가상공간을 만든 뒤, 3D MAX로 이미 만들어진 책상, 의자, 소파 등과 같은 사무용 제반 가구들과 형광등, 문, 액자 등과 같은 가구 이외의 3차원 오브젝트들을 기본 라이브러리로 구축한 후 MFC와 OpenGL을 이용하여 3D Visual 디스플레이를 제작한다.

단순히 3차원 공간에 3차원 물체를 배치하는 것에서 끝나는 것이 아닌 모든 공간, 물체들에 대해 이미지를 mapping시키고 이동, 회전, 확대, 축소 등의 간단한 기능을 지원한다. 그리고 다양한 재질과 조명, 반사 효과, 그림자 효과 등을 적용하여 좀더 사실적으로 렌더링 할 수 있도록 한다.

본 논문의 3D Visual 디스플레이가 지원하는 2D 평면 설계 파일은 *.DXF이고, 3D 모델은 *.3DS이다. 따라서 DXF File Loader에 대해서 그 다음은 3DS File Loader에 관해서 설명 한뒤 3D Visual 디스플레이를 만들어 보도록 하겠다.

3.1 DXF File Loader

3.1.1 DXF File 포맷

DXF File Loader를 제작하기 위해서는 DXF File의 포맷을 알아야 한다. DXF는 각 도면 요소를 표현하는 섹션(Section)으로 구성되며, 크게 Header Section, Class Section, Table Section, Blocks Section, Entities Section, Object Section이 있다. 버전 Up이 되면서 Section이 늘어나거나 기능이 추가되기는 하지만 가장 기본적인 Section들에 대해서만 알아보도록 하겠다. DXF File의 전체 구조는 다음과 같다[1][4].

① Header Section은 AutoCAD 데이터베이스 버전, 시스템 변수로 구성되어 있으며, 도면에 대한 일반적인 정보를 표현한다.

② Class Section은 응용 프로그램 정의 클래스에 대한 정보를 표현 한다

③ Table Section은 기호 테이블에 대한 정보를 표현하며 선종류, 도면층. 문자유형, 뷰의 테이블들로 구성된다.

④ Blocks Section은 도면의 각 블록 참조를 구성하는 블록 정의와 도면 요소를 표현한다.

⑤ Entities Section에는 도면의 그래픽 객체를 표현한다.

⑥ Object Section은 도면의 비 그래픽 객체를 표현한다.

그림 2와 같이 간단한 사각형 하나만을 갖는 도면일지라도 완전한 DXF File을 생성 시키면 최소한 수십 페이지에 달하며, 실제 DXF File로 출력시킨 결과 대략 60페이지 정도의 분량이었다. 그림 3은 전체 내용의 극히 일부만을 보여주고 있다.

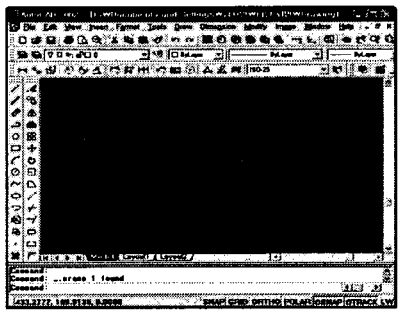

그림2. 사각형 하나만을 갖는 도면

그림3에서 보듯이 DXF File은 2개의 Line이 한 쌍을 이루어 하나의 정보 요소를 나타내는데, 첫 번째 Line은 Group Code로서 정수로 나타내고, 두 번째 Line은 Group Value로서 정수, 실수, 문자열로 표시한다.

0	100	20
SECTION	AcDbPolyline	221.4145687112256
2	90	10
ENTITIES	4	250.4745142374428
0	70	20
LWPOLYLINE	1	117.2508994201205
5	43	10
2B	0.0	89.55065311915246
330	10	20
1F	89.55065311915246	117.2508994201205
100	20	0
AcDbEntity	221.4145687112256	ENDSEC
8	10	
0	250.4745142374428	

그림3. DXF File 中 일부

그림 4와 같이 DXF에서의 자료 표현은 객체에 대한 정보의 유형을 의미하는 Group Code와 그 정보 유형에 대한 값 Group Value로 표현된다. 각각의 Section 마다 Group Code가 정의되어 있다. Group Code 들은 정수형, 실수, 문자열 등의 데이터 형식을 갖는다. 예를 들어 0에서 9까지는 문자열을, 10에서 59까지는 실수(double)라는 것을 나타내며, 그리고 60에서 79까지는 정수(integer)라는 것을 의미한다. 이러한 Group Code에 따라 구조체를 형성해 간다.[5]

3dface group codes

Group codes	Description
100	Subclass marker (AcDbFace)
10	First corner (in WCS) DXF: X value; APP: 3D point
20, 30	DXF: Y and Z values of first corner (in WCS)
11	Second corner (in WCS) DXF: X value; APP: 3D point
21, 31	DXF: Y and Z values of second corner (in WCS)
12	Third corner (in WCS) DXF: X value; APP: 3D point
22, 32	DXF: Y and Z values of third corner (in WCS)
13	Fourth corner (in WCS). If only three corners are entered, this is the same as the third corner DXF: X value; APP: 3D point
23, 33	DXF: Y and Z values of fourth corner (in WCS)
70	Invisible edge flags (optional; default = 0): 1 = First edge is invisible 2 = Second edge is invisible 4 = Third edge is invisible 8 = Fourth edge is invisible

그림4. DXF Group Code

3.1.2 구현

DXF File Loader의 구현은 3 단계로 나뉘며 단계별로 간략하게 알아보면 다음과 같다.

1단계: 2D 평면 DXF File의 각 Data를 임의의 구조체에 저장한다. 각각의 Section마다 표1과 같이 FACE Entity에 Data값을 담을 수 있는 구조체를 Group Code를 이용하여 만들어준 뒤 표2와 같은 방법으로 DXF File을 Open하여 연결리스트를 이용하여 Data를 읽어 들인다.

```
struct DXF_FACE {
  char subclass[255]; // AcDbFace
  double first_corner_x; // 좌표계의 첫번째 점x
  double first_corner_y; // 좌표계의 첫번째 점y
  double first_corner_z; // 좌표계의 첫번째 점z
  // ... 생략
  int invisible; // 1-first,2-second,4-third,8-fourth
  // 각 좌표를 지정하기 전에 i를 입력하면 그 좌표로부터 출
발하는 면의 모서리는 보이지 않게 된다.
  struct DXF_FACE *next;
  DXF_FACE() {
        next = NULL;
  }
};
```

표1. 3차원 공간에서 면을 생성하는 명령

```
void CDXF::DXFOpen(CString fileName) {
  // DXF File을 Open한다
}
int CLoadDXF::Load_DXF(char *filename) {
  // 파일의 끝(EOF)을 만날 때까지 Read
  while(EOF) {
      if(Header Section) {
        ReadHeaderSection();
      }
      // ... 생략
  }
}
void CLoadDXF::ReadEntitiesSection() {
  // Section의 끝(ENDSEC)을 만날 때까지 Read
  while(ENDSEC) {
     // ... 생략
     Read_3DFACD();
  }
}
void CLoadDXF::Read_3DFACE() {
  // 모든 line을 읽어 이미 만들어진 구조체에 연결리스트를
이용하여 모두 저장 한다
}
```

표2. DXF File Open

2단계: 저장된 Data를 가지고 컴퓨터 화면에 그리는 단계로 구조체에 저장된 표3의 Data를 가지고 그리게 된다. 그 과정을 보면 표4와 같다.

```
void CLoadDXFView::Read_3DFace() {
  // .... 생략
  fgets(str, 255, dxf_file);
  face->first_corner_x = atof(str);
  face->first_corner_y = atof(str);
  face->first_corner_z = atof(str);
}
```

표3. Read DXF File

```
void CDXF::Draw_3DFace() {
  // OpenGL 함수를 이용하여 화면에 그린다
}
```

표4. Draw 3DFace

3단계: 2D 평면 도면을 3D로 설계하는 단계로 화면에 출력된 2D 평면 설계도를 3차원상의 공간으로 바꾸어준다. 결과는 그림 5와 같다

이와 같이 DXF File Loader의 기능을 하는 CLoadDXF 클래스를 제작한다.

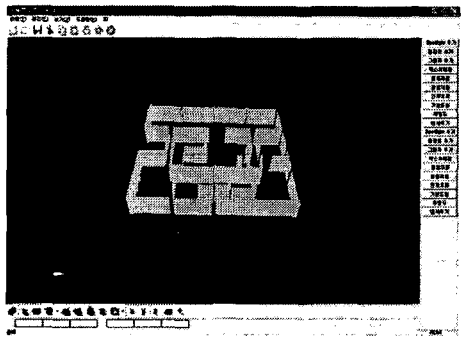

그림5. DXF File Loader

3.2 3DS File Loader 제작

3DS Loader를 제작하기 위해서는 역시 3DS File 포맷을 알아야 한다. 참고로 AutoDesk 사에서 아직까지 공식적인 3DS File 포맷스펙을 제공하지 않는다. 따라서 아직까지 밝혀지지 않은 부분이 있다. 본 논문에서도 인터넷 사이트의 문서와 다른 논문을 참고로 하여 3DS File 포맷을 살펴보기로 하겠다.

3DS File은 Chunk라고 하는 블록으로 이루어져 있다. 이 Chunk는 DXF File의 Section과 유사하다. Chunk는 데이터 흐름을 유도하고, 실제 데이터를 구성한다. 자신을 식별하는 ID와 다음 블록의 위치를 서술하고 있다. 표6과 같이 모든 Chunk 구조는 ID와 Chunk의 길이(Chunk ID의 위치로부터 다음 Chunk까지의 상대적 포인터)로 구성되어 있음을 알 수 있다.[3]

start	end	크 기	이 름
0	1	unsigned short(2)	Chunk ID
2	5	unsigned long(4)	Chunk의 길이

표6 Chunk 구조

Chunk들은 계층적으로 구성된 ID에 의해서 쉽게 식별된다. 하지만 DXF File에 비해 식별이 어려운 이유에서 아직도 많은 Chunk들이 미확인 상태이고 3DS File 포맷이 완벽하게 밝혀지지 않고 있는 이유 중의 하나이다. 3DS File의 최상위 Chunk ID는 4D4Dh로 이 최상위 Chunk ID를 이용하여 P_OBJECT, P_MATERIAL, P_LIGHT 구조체에 정보를 얻어 오게 한다.

P_OBJECT 구조체는 오브젝트가 가지고 있는 정점들의 좌표, 텍스쳐 좌표, 면의 구성방식, 면의 개수 등 오브젝트를 표현하는데 필요한 모든 정보를 가지고 있는 구조체이다. P_MATERIAL 구조체는 주변광, 발산광, 반사광의 색깔, 밝기, 투명도 등 오브젝트들이 가지고 있는 재질에 정보를 담고 있다. P_LIGHT는 조명의 위치, 조명의 종류, 조명의 색깔 등 조명에 대한 정보를 가지고 있는 구조체이다. 이 구조체들을 사용하여 오브젝트에 관련된 정보들을 채우는 정보들을 Load하는 기능을 하는 표7과 같은 Load3DS 클래스를 제작한다.[3]

```
class CLoad3DS {
private:
  . . . . . .
  // 각각의 오브젝트, 재질, 카메라, 조명 정보를 가지고 있는 변수
  P_OBJECT     P_OBJ, P_Object_HEAD;
  P_MATERIAL   P_MAT, P_Material_HEAD;
  P_LIGHT      P_LGT, P_Light_HEAD;
  . . . . . .
public:
  . . . . . .
  int Load3ds( ... )
  . . . . .
}
```

표7. CLoad3DS Class

3.3 BMP File Loader

BMP File 포맷은 가장 흔한 이미지 파일 형식 중 하나이다. 압축이 되어있지 않아서 이미지의 크기가 다른 파일 형식보다 크다라는 단점이 있지는 하지만, 이는 JPG나 TIF등과 같은 이미지 File 포맷보다 이미지를 저장하는데 있어 쉽다는 장점이 된다.

BMP File 구조는 비트맵 파일 헤더, 비트맵 정보 헤더, 비트맵 데이터의 세 부분으로 나뉘어져 있다. 이 부분 Microsoft BMP File을 읽을 때 비트맵 File 헤더, 비트맵 정보 헤더, 비트맵 데이터 순으로 차례대로 읽어야 함을 의미한다. BMP File 형식의 픽셀 비트수가 24비트이고, 압축되어 있지 않다고 가정하고, BMP File Loader의 알고리즘을 정리해 보면 그림 6과 같다.

3.4 3D Visual 디스플레이

3D Visual 디스플레이를 제작하기 위해 지금까지 3개의 Loader를 준비하였다. DXF File Loader는 CLoadDXF라는 이름의 클래스를 만들었고, 3DS File Loader는 CLoad3DS라는 클래스로, BMP File Loader는 CBmpLoader클래스를 만들었다.

```
// 비트맵 데이터를 저장할 구조체 정의
struct rgb_format
float *rgb    // 이미지 데이터
int  xsize:   int  ysize:// 이미지 너비와 높비

// 비트맵 데이터를 읽어오면 0, 그렇지 않으면 1 반환
int load_BMP(char *filename, GL_RGB_FORMAT* image)
File *fp = fopen(filename, "rb");
BITMAPFILEHEADERBMPH:   // 비트맵 파일 헤더
BITMAPINFOHEADERBMIH:   // 파일비트맵 정보 헤더

fread(&BMPH, sizeof(BITMAPFILEHEADER), 1,fp);//비트맵 파일 헤더를 읽는다.
if(BMFH.bfType != BM)   // BMP 파일 형식인지 확인한다.
return 1:

.... 예비용 필드가 0인지를 조사
fread(&BMIH, sizeof(BITMAPINFOHEADER), 1, fp);//파일 정보 헤더를 읽는다.
image->xsize = (int)BMIH.biWidth;  // 비트맵의 너비를 읽어 들인다.
image->ysize = (int)BMIH.biWidth;  // 비트맵의 높이를 읽어 들인다.

.. 색상 평면이 수가 1인지를 확인 후 압축 형식을 조사해서, 압축되어 있으면 0 반환
... 픽셀당 비트수(1, 4, 8, 16, 24, 32 비트) 등의 내용을 읽는다.

if(BMIH.biBitCount != 24)  // 픽셀당 비트수가 24가 아니면 0을 반환
return 0:

// 비트맵 데이터를 위한 메모리 할당
image->rgb = (float *)malloc(sizeof(float)*(image->xsize)*(image->ysize)*3);
..... 비트맵 데이터를 할당한 메모리에 넣는다.
fclose(fp);
return 0:
```

그림6. BMF File Loaer 알고리즘

3D Visual 디스플레이에서 위의 Loader들을 이용한다. DXF File Loader를 이용하여 2D 평면 설계도를 3차원 투상도로 만들고, 3DS Loader를 이용하여 모델에 적용되는 해당 정보를 가져와 렌더링 하게 된다.

전체적인 구조를 설명하면자면 MFC 프로젝트를 생성한 뒤 메뉴와 툴바, 버튼 등을 사용하여 모든 기능을 시현할 수 있는 사용자 인터페이스를 제작한다. 인터페이스가 제작 되었다면 DXF File과 3DS File을 Load 하였을 때 이를 관리해주는 통합 클래스인 C3DS 클래스를 구현한다.

C3DS 클래스는 DXF File Loader와 3DS File Loader를 이용하여 DXF Data와 3DS Data를 Load하게 된다. 이때 C3DS 클래스는 불러온 파일을 오브젝트 단위가 아닌 파일 단위로 리스트를 구성한다. C3DS 클래스는 파일 단위의 리스트를 순회하면서 렌더링하게 된다. 결과를 보면 그림7과 같다.

4. 결론

본 논문에서는 3D Visual 디스플레이 시스템을 제작하는데 있어 2D 평면 설계도의 File 포맷을 DXF를 지원하였다. AutoCAD의 다른 저장 형태인 DWG는 아쉽게도 저장형태가 비공개이므로 아직까지 확장하지 못하였지만 DXF File 만이라도 완벽하게 지원할 수 있도록 더욱더 깊이 있는 연구가 필요하다.

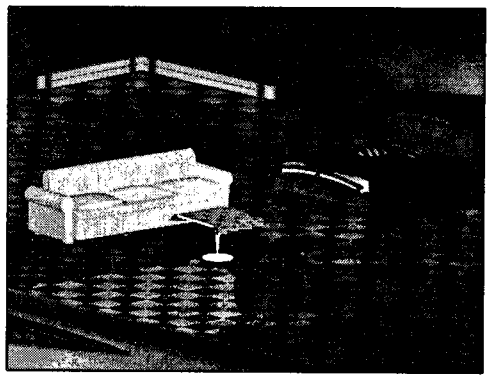

그림7 최종 결과 화면

오브젝트를 위한 File 포맷 또한 현재는 *.3ds File만을 지원하였고 이미지 포맷도 간단한 *.bmp File만을 지원하였지만 이를 더욱 확장하여 다양하게 지원 가능토록 하는 것이 앞으로의 과제이다.

[참 고 문 헌]

[1]김인한 외, "국제표준기반의 건설도면정보 교환모델에 관한 연구", 한국과학재단 산학협력연구, 1999

[2]서병길, "CAD, CAM, CAE SYSTEM상에서의 DATA 호환에 관한 고찰", 한남대학교 정보산업대학원, 2002

[3]이준구 외, "직물패턴 디자인을 위한 3차원 디스플레이 편집기 개발", 건양대학교, 2003

[4]http://inhavision.inha.ac.kr/%7Es970457/cad.htm

[5]http://www.autodesk.com/techpubs/autocad/acad2000/dxf/

[6]http://seojewoo.pe.kr/activex/opengl/opengl-9-1.html