디지털산업정보학회 논문지 제7권 제3호-2011년 9월

비행 정보 시각화를 위한 DMM 시스템의 구현

허 화 라* · 박 명 철**

Implementation of the DMM System for Flight Information Visualization

Hur, Hwa Ra · Park, Myeong Chul

〈Abstract〉

The flight information visualization of the aircraft is the system which is widely used to the threat against low altitude tasks and terrain altitude. But, it is difficult to implement the system because of restrictions that GPS data and huge geographic information should be stored. In this paper, it proposes economic DMM (Digital Moving Map) system for flight information visualization from open-source-base. First, the flight information is transferred from X-Plane through UDP and then demonstrated on the DMM system. In the proposed DMM system, flight information is visualized on the map information downloaded from an ArcGIS Map server using the mapping data between the present altitude of the aircraft and the terrain altitude. The result of this paper could be used an economic tool in the field of flight information visualization and the game algorithm research.

Key Words : Flight Information, DMM(Digital Moying Map), Visualization System

Ⅰ. 서론

항공기의 시각화는 상공에서의 다양한 위협과 돌출적인 상황변화를 더욱 효과적으로 대응하기 위한 목적으로 널리 활용되고 있다[1-3]. 그리고 비행경로를 시각적으로 표현하기 위하여 지형고도 정보를 고려한 경로점을 자동으로 생성하는 연구도 이루어지고 있다[4-5]. 비행경로를 표현하기 위해서는 먼저 DMM (Digital Moving Map, 이하 DMM) 시스템이 구축되어야 하는데 이는 사실적인 지도 및 지형정보와 더불어 비행경로를 안내하는 도구로 이용되고 있다. 그러나 대부분의 DMM은 방대한 지리정보와 지형정보를 데이터베이스로 구축하여 사용해야 하는 제약조건을 가지기 때문에 게임이나 연구적인 목적으로 사용되는 단위 시스템에서는 적용하기가 어렵다. 본 논문에서는 오프소스 기반의 여러 소프트웨어를 이용하여 경제적인 DMM을 구축하여 비행경로를 시각화하는 시스템을 제안한다. 비행 시각화에 필요한 정보는 X-Plane[6]에서 UDP 형태로 전송받는다. 대부분의 비행 시뮬레이션 제품이 비행기의 시각적 모델과 비행절차 등의 특화에 주안점이 있었다면, X-Plane은 다양하고 치밀한 비행운동모델(Flight Dynamics Model)을 제공한다. 본 연구에서 사용되는 비행기의 속도, 경도, 위도 등의 정보를 UDP 프로토콜에 의해 전달받아 사용하게 된다. 취득한 고도와 방향정보를 이용하여 예상되는 진행방향으로 경로를 표시하

* 강릉원주대학교 교수
** 송호대학교 보건의료전자과 교수(교신저자)

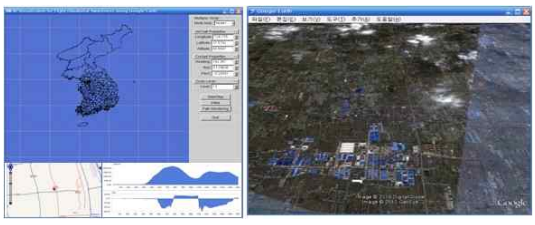
고 다양한 비행정보를 화면상에 표시한다. 시각화 시스템을 구현하기 위하여 OpenGL API[7-8]을 이용하고 지도정보는 ArcGIS의 맵서버에서 전송받아 실시간으로 표시한다[9]. 논문의 구성은 다음과 같다. 2장에서는 비행 시각화와 관련된 연구와 전체 시스템의 구조를 살펴보고 3장에서는 제안하는 시스템의 설계와 구현 및 결과에 대해 기술한다. 마지막으로 4장에서 결론을 맺는다.

Ⅱ. 관련연구

박명철[1] 등은 <그림 1>과 같이 비행정보를 구글 어스을 통하여 사실적으로 시각화하는 도구를 소개하였다. 그러나 이 도구는 DMM 시스템을 가지고 있지 않으며 비행계기를 중심으로 비행정보를 시각화 하였다.

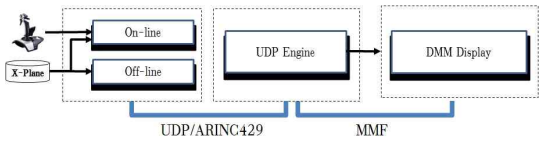
<그림 1> 구글 어스를 이용한 비행정보 시각화 시스템[1]

박석규[2] 등은 <그림2>와 같이 비행 상황을 인식하기 위한 도구를 소개하였다. 이 도구는 이동 경로만을 개괄적으로 보여주는 수준에 그치기 때문에 실제 비행정보를 맵 상에 표시하거나 조종사가 조종시에 참고할 수 있는 시스템으로 사용되기에는 많은 제약이 따른다.



<그림 2> 비행 상황인식을 위한 3차원 시각화[2]

본 논문에서 제안하는 DMM 시스템은 이러한 문제를 해결하기 위하여 실제 조종사가 비행 중에 참고할 수 있는 정보를 제공한다. 기본적인 위치정보와, 속도, 방향, 고도정보가 이에 해당한다. 본 논문에서 제안하는 시스템의 전체적인 구조도 <그림 3>과 같다.



<그림 3> 전체 시스템 구조

모의 비행데이터를 생성하기 위한 X-Plane의 UDP 패킷 엔진과 DMM을 위한 시각화 엔진이 주 구현 내용이 된다. UDP 엔진에서는 X-Plane으로 부터 전송받은 비행정보를 DMM 시스템에 적합하게 정합하고 해당 정보를 특정 메모리 공간에 순차적으로 저장한다. 저장된 정보는 MMF(Memory Mapped File, 이하 MMF)을 이용하여 DMM 시스템에서 사용하게 되는데 생산자와 소비자 관계를 고려하여 특정 변수값을 세마포어로 사용함으로 새로운 정보가 메모리상에 저장되어 있음을 식별한다.

Ⅲ. 제안 시스템의 구현

본 절에서는 제안하는 DMM 시스템의 구현과정에 대해 설명한다. 먼저, 정보취득을 위한 UDP 엔진에 대해 설명하고 DMM 디스플레이의 구조에 대해 설명한다. 그리고 구현된 시스템의 동작을 확인하기 위하여 실제 데이터와 비교 분석하여 구현 시스템의 적합성을 증명한다.

3.1 UDP 엔진의 설계

UDP 엔진의 구조는 크게 UDP 서버 모듈과 패킷을 분석하고 정합하는 모듈로 구성된다. UDP 서버 모듈에서는

UDP 전송을 위한 상세정보를 초기화한다. 내부 네트워크 IP을 사용하고 전송 포트는 9000번으로 설정한다. <그림 4>는 UDP 서버 모듈의 일부 코드를 보이고 있다.

```
WSADATA wsa;

if(WSAStartup(MAKEWORD(2, 2), &wsa) != 0) return -1;
SOCKET sock = socket(AF_INET, SOCK_DGRAM, 0);
if(sock == INVALID_SOCKET) err_quit("socket()");
SOCKADDR_IN serveraddr;

ZeroMemory(&serveraddr, sizeof(serveraddr));
serveraddr. sin_family = AF_INET;
serveraddr. sin_port = htons(9000);
serveraddr. sin_addr. s_addr = htonl(INADDR_ANY);
retval = bind(sock, (SOCKADDR *)&serveraddr, sizeof(serveraddr));
if(retval == SOCKET_ERROR) err_quit("bind()");
```

<그림 4> UDP 서버 모듈의 일부 코드

패킷을 분석하고 정합하는 모듈에서는 X-Plane 에서 전송된 패킷을 해당 인덱스에 따라 DMM 디스플레이에서 사용할 수 있도록 정합하는 과정을 담당한다. <표 1>에서는 X-Plane 으로 부터 전송되어 오는 패킷의 인덱스에 따른 정보를 보이고 있다. 패킷에는 인덱스를 포함하여 최대 8개의 독립된 데이터가 있으며 각각의 데이터는 4바이트 실수형으로 정의되어 있다. <표 1>의 순번이 8개의 독립된 데이터에서 해당되는 순서를 의미한다.

<표 1> X-Plane 의 UDP 패킷 항목[10]

패킷 인덱스	내용	순번
UDP 03	속도	0
UDP 04	상승속도	1
UDP 18	피치, 롤, 방향	0~3
UDP 20	위도, 경도, 고도	0~3

정합된 패킷 정보는 MMF(Memory Mapped File, 이하 MMF)을 이용하여 프로그램 간에 통신을 하게 된다. 해당 데이터와 특정 메모리 객체를 연결하고 명명된 명칭을 이용하여 서로 다른 프로그램 간에 데이터 액세스가 가능하다. 기본적으로 MMF는 파일을 이용한 메모리 고유 방법이지만 파일명을 적는 곳에 아래와 같이 INVALID_HANDLE_VALUE 라고 해주면 메모리 객체만 생성이 된다. 그리고 이 메모리 객체에 이름을 설정해 주면(AIR_INFO) 공유 메모리의 생성이 완료된다.

Code 1:

```
HANDLE AirFileMap = ::CreateFileMapping
        (INVALID_HANDLE_VALUE, NULL,
        PAGE_READWRITE, 0,1024,
        "AIR_INFO");
```

위 코드에서 정보 생산자인 UDP 엔진에 의해 수행되는 코드로서 AIR_INFO라고 명명된 메모리 객체가 생성되는며 크기가 1024 바이트의 읽기, 쓰기 가능한 상태가 된다. 그리고 MapViewOfFile 메소드를 이용하여 생성된 메모리 객에에 접근하여 시작 위치를 얻어 낸다.

아래 코드는 얻어진 시작위치를 이용하여 실제 비행 정보를 공유메모리 영역에 복사하는 부분이다. Flight_info 변수는 문자형 변수로서 정합된 패킷정보를 담고 있다.

Code 2:

```
AirData = ::MapViewOfFile (AirFileMap,
        FILE_MAP_ALL_ACCESS, 0, 0, 1024);
strcpy((char *)AirData, Flight_info);
```

이렇게 공유메모리에 저장된 정보는 3.2절에서 설명하는 DMM 디스플레이 엔진에서 소비자의 역할로 사용하게 된다. 생산자가 동작하지 않으며 소비자가 동작할 수 없기 때문에 아래 코드와 같이 UDP 엔진에 의해 공유메모리가 생성되었는지를 검사하게 된다. 그리고 아직 공유메모리가 생성되어있지 않으면 프로그램을 종료한다.

Code 3:

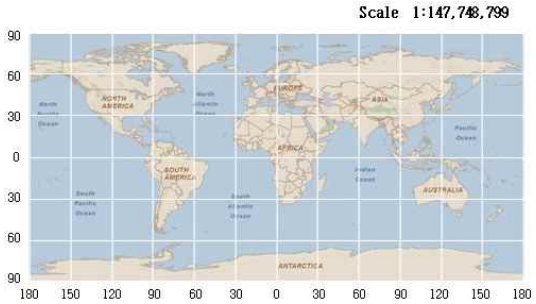
```
AirFileMap = ::OpenFileMapping
        (FILE_MAP_ALL_ACCESS, FALSE,
        "AIR_INFO");
if(!AirFileMap){
        MessageBox(NULL, "UDP 모듈이 동작하지
        않습니다. 확인하세요.", "확인",MB_OK);
        return 0;
}
```

정상적으로 공유메모리 객체가 생성되었다면 해당 메모리의 위치를 읽어 오게 된다. 이후에는 해당 공유메모리 객체에 접근하여 시작 위치를 얻어 오고 원하는 데이터를 읽고 쓸 수 있게 된다. 데이터를 액세스하는 방법은 생산자인 UDP 엔진에서와 동일하다.

한 가지 문제점은 생산자인 UDP 엔진이 넣어둔 데이터를 소비자인 DMM 시스템이 사용한 이후에 공유메모리에 해당 데이터는 이미 사용되었음을 알리고 UDP 엔진에게 새로운 데이터를 넣어 주기를 알려야 한다. 본 도구에서는 DMM 시스템이 데이터를 사용하게 되면 해당 영역에 문자 "0"를 저장함으로서 데이터를 사용했음을 알린다. 그리고 생산자인 UDP 엔진은 이 정보를 보고 소비자인 DMM 시스템이 새로운 데이터를 원한다는 것을 식별할 수 있게 구현되었다.

3.2 DMM 시스템의 설계 및 구현

DMM을 위한 지도정보는 ArcGIS Online 에서 제공하는 2차원 맵을 사용한다. 해당 서버의 맵정보는 <그림 5>와 같이 위도는 90° 에서 -90° 까지, 경도는 -180° 에서 180° 까지의 맵을 축척에 따라 제공하고 있다. 축적은 레벨 0에서 레벨 10까지 제공하는데 각 축척별 맵 정보는 <표 2>와 같이 단위 이미지로 분할되어 있다. 레벨 0의 축척 지도정보는 실제 구현에는 사용하지 않고 레벨 1의 지도 정보를 가장 하위 수준의 축척으로 사용한다.



<그림 5> ArcGIS 맵 서버에서 제공하는 맵의 경도와 위도

초기 축척 레벨은 4단계에서 시작되는데 4단계의 단위 이미지의 경위도 범위 값은 11.25° 이다. 이는 4단계의 이미지가 경도를 기준으로 32장으로 분할되어 있기 때문에 360° 를 32로 나눈 11.25° 가 하나의 이미지의 경도 범위 값이 되는 것이다. 그리고 레벨이 올라가면 범위 값이 절반으로 감소하고 레벨이 내려가면 반대로 범위 값이 두 배로 증가한다.

<표 2> 축척에 따른 수준별 단위 분할 수

Level ID	위도 분할수	경도 분할수	축척
0	1	2	1:147,748,799
1	2	4	1:73,874,399
2	4	8	1:36,937,199
3	8	16	1:18,468,599
4	16	32	1:9,234,299
5	32	64	1:4,617,149
6	64	128	1:2,308,574
7	128	512	1:1,154,287
8	256	1024	1:577,143
9	512	2028	1:288,571
10	1024	4096	1:144,285

DMM 시스템에서는 UDP 엔진으로 부터 전송받은 실제 비행정보의 경위도 값을 이용하여 해당 이미지를 결정하고 웹을 통하여 이미지를 실시간으로 전송받아 화면상에 표시한다. 비행 위치 정보를 이용한 분할 이미지 중

해당 이미지를 결정하는 코드는 다음과 같다.

Code 4:

```
x_stno=abs((int)(((-180)-lon)/zoom_ra));
y_stno=abs((int)((lat-90)/zoom_ra));
```

위 코드에서 *lon*은 비행 경도 정보이고 *lat*는 비행 위도 정보이다. 그리고 *zoom_ra*는 경위도의 범위 값이다. 그리고 *x_stno* 는 경도의 분할 영역의 인덱스이고 *y_stno* 는 위도의 분할 영역의 인덱스 값이 된다. 이렇게 하면 비행중인 위치에 해당하는 이미지의 분할 영역을 구할 수 있고 해당 이미지를 아래 코드와 같이 구현하여 취득한다.

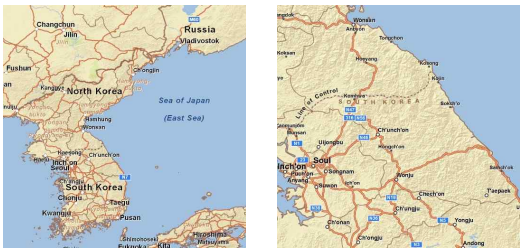
Code 5:

```
sprintf(str1,"http://server.arcgisonline.com/ArcGIS/rest/
        services/ESRI_StreetMap_World_2D/MapServe
        r/tile/%d/%d/%d.jpg",zoom_in, y_stno,x_stno);
```

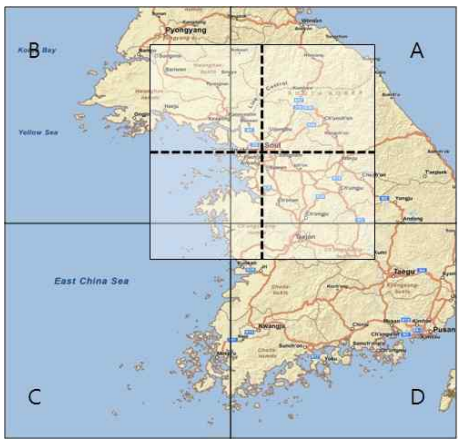
가령, 서울 시청의 경위도 값(경도 : 126.58°, 위도 : 37.33°)을 축척 레벨 *zoom_in*을 4로 지정하고(*zoom_ra*= 11.5) 코드 4에 대입하면 *x_stno*는 27, *y_stno* 는 4가 된다. 그리고 축척 레벨 *zoom_in*을 6으로 지정하고(*zoom_ra*= 2.81255) 대입하면 *x_stno*는 109, *y_stno* 는 18이 된다. 이를 코드 5에 대입하여 얻어진 지도 정보를 <그림 6>에 보이고 있다.

취득한 지도 정보를 화면상에 표시하기 위해서는 해당 위치를 중심으로 재배치하여 텍스쳐를 매핑하여야 한다. <그림 6>의 이미지는 해당 위치 값이 포함되어 있는 이미지이고 이를 <그림 7>과 같이 서울 시청을 화면에 중심에 배치해야 한다. A 이미지가 취득한 이미지 인데 해당 위치 값을 화면 중앙으로 배치하면 클리핑 영역이 발생하므로 B, C, D에 해당하는 세 장의 이미지를 추가로 매핑해야 한다.

구현된 도구에서는 맵 서버에서 한 번에 총 9장의 이



<그림 6> 서울 시청의 경위도 값을 이용한 ArcGIS 맵 서버을 통한 지도 정보 취득 (좌: 레벨 4, 우: 레벨 6)



<그림 7> 비행 위치를 중앙에 배치하기 위한 구조

<그림 8> 이미지를 최종 텍스쳐 매핑한 결과

미지를 얻어온다. 이는 비행의 위치가 연속적으로 변화하기 때문에 해당 이미지 외에 상하좌우 이미지와 대각

선 상의 이미지가 있어야 <그림 8>과 같이 원활한 매핑이 가능해 진다. 이는 한 장의 이미지가 아닌 4장의 이미지를 각각 클리핑하여 매핑한 결과이다.

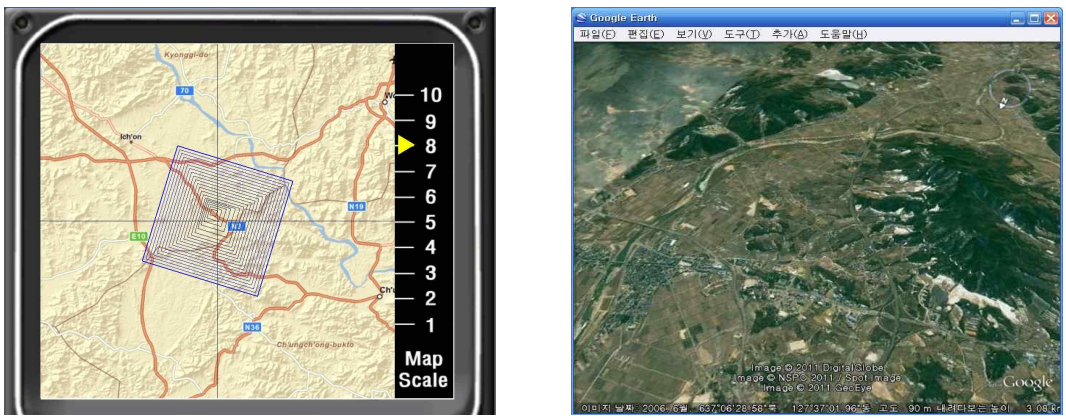
<표 2>에서 보는 것과 같이 축척 레벨이 4단계인 경우에 전체 이미지의 수가 512장(16*32)가 된다. 웹 환경에서 이미지를 취득하기 때문에 해당 레벨 이미지를 모두 가져오기에는 많은 부하가 따른다. 그리고 <그림 7>의 A 이미지 내에 비행 위치가 존재한다면 가져온 9장의 이미지만으로 매핑을 할 수 있기 때문에 모든 이미지를 가져올 필요는 없다. 만일 비행 위치가 A 이미지 영역을 벗어나면 다시 9장의 이미지를 맵 서버에서 가져오면 된다. 그리고 레벨이 커지면 단위 이미지에 해당되는 경위도 범위 값이 작아지기 때문에 빈번한 이미지 취득 작업이 발생한다. 하지만, 실험 결과 저속으로 비행하는 경비행기나 민항기의 경우에는 아무른 시간적 지연을 볼 수 없었고 고속 전투기의 경우에는 레벨 8 이상인 경우 로딩으로 인한 약간의 지연이 발생(깜박임) 하였지만 도구를 사용하는 데는 별 문제가 없음을 확인하였다.

3.3 구현 도구 및 결과

본 논문에서 제안하는 시스템을 구현하기 위한 환경은 윈도우즈 기반에서 C++ 언어을 사용하였고 그래픽

<그림 9> 최종 구현된 DMM 시스템

라이브러리는 OpenGL을 이용하여 구현하였다. 비행정보는 실제 항공기 정보를 취득하기에 어려움이 있으므로 서론에서 언급한 것과 같이 X-Plane 을 이용하여 UDP 형태로 데이터를 취득하였다. 그리고 DMM 시스템에 원활한 비행을 위하여 진행 방향상의 지형 고도 정보와 경로 정보를 표시함으로서 비행 상태를 인지할 수 있게 구현하였다. 최종적으로 구현된 시스템의 동작 모습은 <그림 9>와 같다. 그리고 표시되는 도구의 사실성을 증명하기 위하여 동일한 위치 값에서 동작하는 구글 어스의 맵 모습을 비교한 결과 <그림 10>과 같이 동일함을 확인하



<그림 10> DMM 시스템과 구글 어스 맵의 비교(좌: 구현된 시스템, 우:구글 어스의 맵)

였다. 이는 구현 결과가 실제 맵 정보와 차이가 없음을 증명하는 것이다.

Ⅳ. 결론

본 논문은 항공기의 비행 시에 사용되는 지도정보를 오프소스 기반에서 구현하였다. 구현을 위하여 OpenGL을 이용하였고 웹 환경의 맵 서버을 통하여 가시화 하였다. 구글 어스의 맵과 비교한 결과, 실제 비행 상태의 맵 환경을 효과적으로 보이고 있음을 확인하였다. 본 연구 결과는 비행평가 및 개선 등에 활용할 수 있으며 향후 임베디드 시스템 기반의 비행 소프트웨어에 적용될 수 있다. 향후 연구는 맵 데이터베이스의 독립적인 구성과 실시간 비행정보 취득 기술에 대한 연구를 통하여 가상 비행 시뮬레이터 도구 개발을 지속할 예정이다.

참고문헌

[1] 박명철・허화라, “Google Earth를 이용한 비행정보 시각화 시스템의 구현,” 한국컴퓨터정보학회, 한국컴퓨터정보학회논문지, 제15권, 제10호, 2010, pp. 79-86.

[2] 박석규・박명철, “구글 어스를 이용한 비행 상황인식을 위한 3차원 시각화,” 한국컴퓨터정보학회, 한국컴퓨터정보학회논문지, 제15권, 제12호, 2010, pp. 181-188.

[3] FlightViz, http://www.simauthor.com

[4] 박정진・박상혁・유창경・신성식, “지형 회피를 위한 최적 경로점 자동 생성 알고리듬 연구,” 한국우주항공학회, 한국우주항공학회지, 제37권, 제11호, 2009, pp. 1104-1111.

[5] 송진오・박태진・김종석・최윤철, “항공기 상태전이정보를 이용한 비행경로 시각화 기법 연구,” 한국정보과학회 학술발표논문집, 제34권, 제2호(B), 2007, pp. 172-177.

[6] X-Plane Desktop manual, http://www.x-plane.com/files/manuals/X-Plane_Desktop_manual.pdf

[7] Shreiner and Dave, “OpenGL Programming Guide,” Addison-Wesley, 2009.

[8] Edward Angel, “Interactive Computer Graphics,” PEARSON, 2009.

[9] World Street Map, http://www.arcgis.com

[10] X-Plane SDK Documentation, http://www.xsquawkbox.net/xpsdk/docs/

■ 저자소개 ■

허 화 라
Hur, Hwa Ra

2011년 3월~현재
강릉원주대학교
2000년 3월~2011년 2월
송호대학교 보건의료전자과 부교수
2001년 2월 부산대학교 전자공학과 (공학박사)

관심분야 : ime-Delay, 모델예측제어, 원격제어로봇
E-mail : haru@gwnu.ac.kr

박 명 철
Park, Myeong Chul

2007년 7월~현재
송호대학교 보건의료전자과 전임강사
2007년 2월 경상대학교컴퓨터과학과(공학박사)

관심분야 : 시뮬레이션, 임베디드 소프트웨어, 병렬프로그래밍 및 디버깅
E-mail : africa@songho.ac.kr

논문접수일 : 2011년 7 월 15 일
수 정 일 : 2011년 8 월 12 일
게재확정일 : 2011년 8 월 20 일