韓國컴퓨터情報學會 論文誌
第15卷 第10號, 2010. 10.

2010-15-10-2-7

Google Earth를 이용한 비행정보 시각화 시스템의 구현

박 명 철*, 허 화 라**

Implementation of the Flight Information Visualization System using Google Earth

Myeong-Chul Park*, Hwa-ra Hur**

요 약

본 논문은 항공기의 비행정보를 효과적으로 시각화하기 위하여 Google Earth를 이용한 시스템 구현에 대해 기술한다. 이 시스템은 Google Earth 에서 제공하는 상세한 위성 영상 정보를 이용하기 위하여 COM API 를 이용하였다. 제안하는 시스템은 항공기의 다양한 장치에서 얻어진 비행정보를 OpenGL을 이용하여 각종 계기판에 표시하고 실제 비행모습을 가시화하기 위하여 Google Earth Map에 연동하여 표시하였다. 본 연구결과는 비행평가 및 개선에 이용될 수 있고 향후, 비행소프트웨어 개발에 적용될 수 있다.

Abstract

This paper presents implementation of a system for effective visualizing flight information of aircraft using Google Earth. This system in order to use a detailed satellite image which provide from Google Earth used COM API. This system appeared the various flight information of the aircraft in the instrument panel using OpenGL and the aircraft flight condition is visible in the Google Earth Map. This research result used to flight evaluation and improvement. In future will be able to apply to flight software development.

▸ Keyword : 비행시뮬레이션(Flight Simulation), 구글어스(Google Earth), 시각화시스템(Visualization System)

• 제1저자 : 박명철
• 투고일 : 2010. 08. 11, 심사일 : 2010. 08. 27, 게재확정일 : 2010. 08. 28.
* 송호대학 컴퓨터정보과 ** 송호대학 보건의료전자과

Ⅰ. 서 론

게임 등 에서 사용되는 대부분의 비행 시뮬레이션은 사용자 인터페이스를 통한 비행 모습 가시화가 대부분이다. 이는 사용자의 행위를 비행정보로 이용할 뿐 비행정보를 가시화하여 비행을 평가하고 개선하기 위한 도구로 이용되기는 어렵다.

비행의 결과로 수집된 정보는 더 나은 비행을 위한 정보로 이용될 수 있으므로 평가 및 개선을 위한 시각화 도구가 요구된다. 그러나 대부분의 시뮬레이션 도구는 후 처리를 위한 분석도구가 아닌 게임 분야와 같이 동작에 따른 가시적 효과를 중심으로 사용되어 왔다[1,2,3].

또한 기존의 시뮬레이션 도구들은 대부분 고가의 도구로서 사용자의 조정을 통한 동작만을 시각화 할 뿐 비행정보를 가시화 하지는 못하기 때문에 비행정보를 평가하는 도구로서 사용되지 못하는 문제점을 가지고 있다. 비행체의 비행데이터 가시화 소프트웨어는 이미 몇 가지가 개발되었지만 대부분 비행체의 동작을 제어하기 위한 가시화 도구이고 특정한 목적을 위해 생성되는 데이터이므로 일반 사용자가의 요구사항을 만족시키는 것은 매우 어렵다[4,5]. 이러한 문제를 해결하기 위하여 본 논문에서는 공개된 Google Earth의 Open API (Open Application Program Interrace)를 이용하여 누구나 쉽게 사용할 수 있는 저가의 시각화 시스템을 제안하고 비행정보의 후처리를 통하여 비행정보를 평가할 수 있는 도구를 제안한다[6,7]. 사용자는 논문에서 제시하는 비행정보 데이터의 구조에 맞게 데이터베이스를 구축하면 사용자가 원하는 비행정보에 대한 고도, 속도 및 각 정보의 변화 추이와 실제 비행모습을 확인하여 더 나은 비행을 위한 평가 정보로 이용할 수 있다. 그리고 비행정보의 변화를 직시할 수 있게 OpenGL 라이브러리를 이용하여 조종석의 계기를 표현하였고 효과적인 분석을 위하여 사용자 인터페이스로 모니터링 시스템을 함께 제공한다[8,9].

논문의 구성은 다음과 같다. 2장에서는 시스템 구현을 위한 배경에 대해 기술하고 3장에서는 조종석 계기의 표현에 대해서 설명한다. 4장에서는 계기와 실제 시뮬레이션을 연동하기 위한 시스템 구현에 대해 기술하고 마지막으로 5장에서 결론을 맺는다.

Ⅱ. 시스템 구현을 위한 배경지식

1. 비행정보의 수집과 DB 구축

항공기의 비행정보는 군사적 목적으로 사용되는 특수성이 있기 때문에 실제 정보를 얻기에는 어려움이 있다. 본 연구에서는 대표적인 비행 시뮬레이터인 X-Plane의 SDK(Software Development Kit) 플러그인을 이용하여 시뮬레이터의 동작 시 생성되는 데이터를 경로별로 수집하였다[10]. 시각화에 사용된 비행 데이터는 수집 후 0.1초 단위로 정합과정을 거쳐 해당 DB에 기록되게 된다. 수집된 데이터는 기본적인 위치 정보인 경도, 위도, 고도와 Heading, Roll, Pitch등의 자세 정보를 포함하고 있다. 그리고 엔진을 비롯한 각 유닛들의 계기 정보가 부가적으로 수집되었다. 이 정보는 3차원 화면표시에 사용되면 계기의 관계성을 통하여 정형화시켜 사용된다. 표 1은 수집한 비행정보의 데이터 구조를 보인 것이고 [그림 1]은 수집된 원시 정보를 보이고 있다.

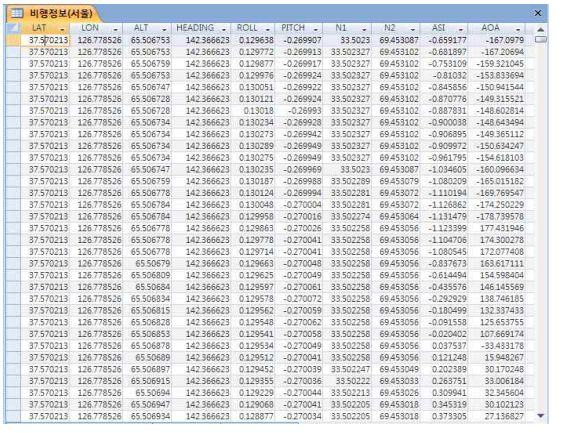
표 1. 비행정보의 데이터 구조
Table 1. Data Structure of Flight Information

필드명	데이터형	내용
Lat	double	Latitude
Lon	double	Longitude
Alt	double	Altitude
Heading	double	Azimuth angle
Roll	double	Roll Angle
Pitch	double	Pitch Angle
Yaw	double	Yaw Rate
N	double	RPM Engine
ASI	double	Indicated Airsoeed
AOA	double	Angle of Attack
VX	double	Heding Velocity
VY	double	Drifr Velocity
VZ	double	Vertical Velocity
ITT	double	ITT Engine Temperature
OP	double	Oil pressure
OT	double	Oil Temperature
FF	double	Fuel Flow

2. Google Earth의 COM API

Google Earth는 위성 영상뿐 아니라 전 세계의 상세한 지도와 지형 정보를 무료로 제공하고 있다. 또한 다양한 운영체제에서 동작하며 Google Earth COM API을 이용하여 자유롭게 Google Earth을 제어할 수 있다. 응용프로그램의 질

의 정보를 Google Earth에 명령으로 전달하여 사용자 뷰 위치를 바꿀 수 있고 경도, 위도 정보 등을 변경하면 자연스러운 비행정보 시각화 도구를 구현할 수 있다.


비행정보(서울)

LAT	LON	ALT	HEADING	ROLL	PITCH	N1	N2	ASI	AOA
37.570213	126.778526	65.506753	142.366623	0.129638	-0.269907	33.5023	69.453087	-0.659177	-167.0979
37.570213	126.778526	65.506753	142.366623	0.129772	-0.269913	33.502327	69.453102	-0.681897	-167.20694
37.570213	126.778526	65.506759	142.366623	0.129877	-0.269917	33.502327	69.453102	-0.753109	-159.321045
37.570213	126.778526	65.506753	142.366623	0.129976	-0.269924	33.502327	69.453102	-0.81032	-153.833694
37.570213	126.778526	65.506747	142.366623	0.130051	-0.269922	33.502327	69.453102	-0.845856	-150.941544
37.570213	126.778526	65.506728	142.366623	0.130121	-0.269924	33.502327	69.453102	-0.870776	-149.315521
37.570213	126.778526	65.506728	142.366623	0.13018	-0.26993	33.502327	69.453102	-0.887831	-148.602814
37.570213	126.778526	65.506734	142.366623	0.130234	-0.269928	33.502327	69.453102	-0.900038	-148.643494
37.570213	126.778526	65.506734	142.366623	0.130273	-0.269942	33.502327	69.453102	-0.906895	-149.365112
37.570213	126.778526	65.506734	142.366623	0.130289	-0.269949	33.502327	69.453102	-0.909972	-150.634247
37.570213	126.778526	65.506734	142.366623	0.130275	-0.269949	33.502327	69.453102	-0.961795	-154.618103
37.570213	126.778526	65.506747	142.366623	0.130235	-0.269969	33.5023	69.453087	-1.034605	-160.096634
37.570213	126.778526	65.506759	142.366623	0.130187	-0.269988	33.502289	69.453079	-1.080209	-165.015182
37.570213	126.778526	65.506778	142.366623	0.130124	-0.269994	33.502281	69.453072	-1.110194	-169.769547
37.570213	126.778526	65.506784	142.366623	0.130048	-0.270004	33.502281	69.453072	-1.126862	-174.250229
37.570213	126.778526	65.506784	142.366623	0.129958	-0.270016	33.502274	69.453064	-1.131479	-178.739578
37.570213	126.778526	65.506778	142.366623	0.129863	-0.270026	33.502258	69.453056	-1.123399	177.431946
37.570213	126.778526	65.506778	142.366623	0.129778	-0.270041	33.502258	69.453056	-1.104706	174.300278
37.570213	126.778526	65.506778	142.366623	0.129714	-0.270041	33.502258	69.453056	-1.080545	172.077408
37.570213	126.778526	65.50679	142.366623	0.129663	-0.270048	33.502258	69.453056	-0.837673	163.617111
37.570213	126.778526	65.506809	142.366623	0.129625	-0.270049	33.502258	69.453056	-0.614494	154.598404
37.570213	126.778526	65.50684	142.366623	0.129597	-0.270061	33.502258	69.453056	-0.435576	146.145569
37.570213	126.778526	65.506834	142.366623	0.129578	-0.270072	33.502258	69.453056	-0.292929	138.746185
37.570213	126.778526	65.506815	142.366623	0.129562	-0.270059	33.502258	69.453056	-0.180499	132.337433
37.570213	126.778526	65.506828	142.366623	0.129548	-0.270062	33.502258	69.453056	-0.091558	125.653755
37.570213	126.778526	65.506853	142.366623	0.129541	-0.270058	33.502258	69.453056	-0.020402	107.669174
37.570213	126.778526	65.506878	142.366623	0.129534	-0.270049	33.502258	69.453056	0.037537	-33.433178
37.570213	126.778526	65.50689	142.366623	0.129512	-0.270041	33.502258	69.453056	0.121248	15.948267
37.570213	126.778526	65.506897	142.366623	0.129452	-0.270039	33.502247	69.453049	0.202389	30.170248
37.570213	126.778526	65.506915	142.366623	0.129355	-0.270036	33.50222	69.453033	0.263751	33.006184
37.570213	126.778526	65.50694	142.366623	0.129229	-0.270044	33.502213	69.453026	0.309941	32.345604
37.570213	126.778526	65.506947	142.366623	0.129068	-0.270041	33.502205	69.453018	0.345319	30.102123
37.570213	126.778526	65.506934	142.366623	0.128877	-0.270034	33.502205	69.453018	0.373305	27.136827

그림 1. 항공기의 비행정보(서울)
Fig. 1. Flight Information of Aircraft (SEOUL)

Google Earth는 태그 기반의 KML(Keyhole Markup Language) 파일 형식에 지리정보를 저장 및 표현하여 Google Earthd에 표시할 수 있다. 그러나 본 논문에서는 KML 파일을 사용하지 않고 API를 통하여 직접 표시 정보를 전달한다. 정보의 전달은 계기판과 Google Earth의 연동을 위하여 공유 메모리를 이용한다. 그리고 Google Earth의 카메라 위치 이동과 애니메이션을 위해 ApplicationGE 클래스를 인스턴스화 하고 Google Earth COM API의 주 영역에 해당하는 EARTHLib 라이브러리의 IApplicationGE를 사용하고 카메라의 뷰를 표현하기 위하여 ICameraInfoGE 클래스 포인터를 이용한다. 본 논문에서 사용하는 IApplicationGE 인터페이스의 주요 멤버 함수는 표 2에 보인다[11].

표 2. IApplicationGE 인터페이스의 주요 멤버함수
Table 2. Member Functions of IApplicationGE Interface

멤버함수	기능
GetCamera	실행중인 현재 카메라 뷰을 반환
GetMainHwnd	Google Earth의 메인 윈도우를 위한 핸들러 반환
SetCameraParams	사용자 정보를 실제 뷰 영역에 설정하는 함수

3. 비행계기의 이해

비행시뮬레이션 관련 항공기의 계기정보는 각 부의 현상과 작동 상태를 지시하고 이상 유무를 경고하며 비행의 자세, 위치, 진로 지시를 위해 가장 기본이면서 중요한 정보이다. 일반적으로 항공기의 계기는 정확성과 경량성, 소형성, 내구성을 가져야 하며 상온오차 및 마찰오차가 적어야하고 누설오차는 전혀 없어야 한다. 그리고 방청 및 방균 처리가 필수적이다[12].

항공계기의 종류는 크게 세 가지로 구분할 수 있다. 비행계기(Flight Instruments)는 항공기의 비행 상태 즉, 고도, 속도, 자세 등을 지시하며 고도계, 속도계, 인공 수평의가 이에 해당한다. 항법계기(Navigation Instruments)는 항공기의 진로, 방위, 위치를 지시하는 계기이며 방위계, 자동무선 방향 탐지기가 대표적인 예이다. 그리고 엔진계기(Engine Instruments)는 엔진을 포함한 동력장치에 관계되는 계기로서 회전 속도계, 윤활유 압력계, 배출가스 온도계 등이 이에 해당한다. 또한 대부분의 항공계기는 기존의 아날로그와 독립식 계기방식에서 디지털 및 통합형 계기 시스템으로 전환되고 있는 추세이다. 그리고 GPS를 이용한 항법 시스템이 보편화되고 있다[13]. 최근의 계기 기술 동향은 시계 향상 기술을 중심으로 연구되고 있는데 강화 영상 시스템이 그 대표적 예이다. 이 시스템은 전방 적외선 센서를 통해 얻어진 원적외선, 근적외선, 가시대역 영상을 이용하여 시계정보를 융합하여 훨씬 정확한 시계정보를 제공하는 시스템이다. 그리고 합성 영상 시스템은 고도, 자세, 위치 등 데이터베이스를 이용하여 가상 이미지를 생성하여 시각에 의해 얻어진 주변 상황과 결합시켜 시계확보가 어려운 환경을 극복하여 안전한 이착륙을 유도하는 시스템이다[14]. 본 논문에서는 항공기의 운항정보를 이용한 PFD(Primary Flight Display)의 구현을 통하여 기존 Basic T를 대체하는 디지털 계기 정보를 제공한다.

PFD 에는 Air 데이터와 ADAHRS(Attitude Heading Reference System)정보가 결합된 형태로 구현된다. 그리고 MFD(Multi Function Display)을 통하여 항공기의 고도 및 속도, 자세를 확인할 수 있는 통합 화면을 제공한다. 비행시각화 시스템과 더불어 계기 정보를 사실적으로 표현함으로서 효과적인 정보 인식이 가능하다.

III. 비행계기의 구현

비행계기의 구현은 OpenGL 라이브러리를 이용하여 구현되었으며 텍스쳐 매핑과 블렌딩 방식을 통하여 사실적으로 표

현하였다. 텍스쳐 매핑은 단순한 도형들을 사실적으로 표현하는데 유용한 기법으로 다양한 다각형 폴리곤에 사용자가 원하는 이미지를 매핑할 수 있다.

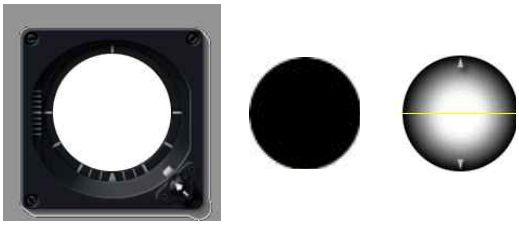


그림 2. 블렌딩을 위한 이미지(A.배경, B.원본, C.마스크)
Fig. 2. Images for Blending(A.DST, B.SRC, C.Mask)

특정 이미지를 화면에 표시할 때 본래의 이미지 위에 겹치게 됨으로 본래의 이미지가 지워지게 된다. 대부분의 비행계기는 안쪽에 위치한 부분만 동작하고 바깥쪽의 이미지는 고정되게 된다.

그림 3. 블렌딩의 결과 이미지
Fig. 3. Result Image of Blending

이를 위하여 블렌딩 기법을 이용하는데 계기의 바탕을 검정색으로 하고 내부에는 원형 이미지가 있다. 이 이미지를 배경 이미지에 더하면 실제로는 배경이미지와 원형 이미지의 RGB 값이 더해져 바탕이 검은색인 부분은 배경에 아무른 영향을 미치지 않는다. 하지만 원형 이미지와 연산하는 배경 이미지의 부분이 완전한 검은색(RGB 0,0,0)이 아니면 색이 변질되는 문제점이 있다. 이러한 문제점은 배경 이미지에서 연산할 부분에 대해 마스크 이미지를 부가하여 연산하면 해당 부분이 검정색으로 바뀌게 된다. 마스크 이미지는 배경을 흰색으로 하고 원형 이미지 부분은 검은색으로 마스크 이미지를 배경에 논리 AND연산을 하면 된다. 그리고 원형 이미지를 OR연산하게 되면 원하는 영상을 계기정보로 표현할 수 있다.

[그림 2]는 블렌딩을 위한 배경 이미지와 원본 이미지, 마스크 이미지를 보인 것이고 [그림 3]은 최종적인 블렌딩 기법으로 생성된 이미지를 보인 것이다.

그림 4. 구현된 전체 계기판
Fig. 4. Instrument Panel of Aircraft

[그림 4]는 구현된 항공기의 전체 계기판을 보이고 있다. 좌우측의 MFD와 하단 중앙에 PFD에 대한 정보가 각각의 계기를 통해 보여진다. 중앙 하단의 계기판 정보는 좌측 MFD영역에 통합하여 표시되기도 하며 좌우측의 MFD는 더욱 식별이 용이하게 하기 위하여 팝업으로 확대 출력된다.

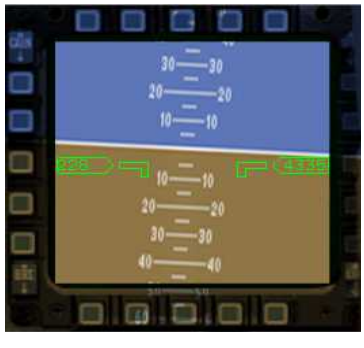


그림 5. LeftMFD의 PFD
Fig. 5. PFD of LeftMFD

[그림 5]는 왼쪽 MFD에 표시되는 PFD 로서 기본적인 비행정보를 디스플레이 한다. 여기에는 항공기의 속도와 고도, 기수의 상하, 좌우각도를 표시한다. 하단의 MAP 버튼을 클릭하여 이동되는 지형의 맵을 보이는 디스플레이로 변환이 가능하다. 지형에 대한 고도를 보일 수 있고 맵의 크기를 확대/축소 할 수 있다. 맵상의 기호들은 각 버튼을 통하여 그 기능을 알 수 있다.

그림 6. MFD의 System Display
Fig. 6. System Display of MFD

[그림 6]은 오른 쪽에 표시되는 시스템 디스플레이로서 엔인, 기어, 보조익과 주 날개에 부착된 고정식 작은 날개, 에어브레이크, 연료의 양에 대한 중요 정보를 보여주는 MFD이다. 비행정보를 화면 배치영역의 비율에 맞게 조정하여 표시한다.

그림 7. 각종 계기판
Fig. 7. Various instrument panel

[그림 7]은 현재 상태의 연료 유량(FF)과 두 개의 엔진 RPM(N1, N2), 항공기의 속도계를 보이는 계기판이다. 그리고 하단 중앙부에 받음각 지시계(AOA)와 자세방향지시계(ADI), 수직속도계(VVI)를 구현하였다. 배경의 이미지는 실제 계기판을 이미지를 사용하였고 중심의 지시계만 OpenGL을 이용하여 비행정보에 맞게 드로잉 하였다.

[그림 8]은 블렌딩 기법을 이용하여 계기판 구현을 위한 코드의 일부분을 보이고 있다. 텍스쳐 맵핑을 위한 이미지를 로드하여 [그림 2]에서 설명한 블렌딩 절차에 따라 화면에 위치한다.

```
glPushMatrix ();
    glEnable(GL_DEPTH_TEST);
    glDisable(GL_BLEND);
    glBlendFunc(GL_DST_COLOR,GL_ONE_MINUS_DST_ALPH
    A);
    glBindTexture(GL_TEXTURE_2D, texture[0]);
    glBegin(GL_QUADS);
        glNormal3f(0.0f, 0.0f, 1.0f);
        glTexCoord2f(0.0f, 0.0f);
        glVertex3f(-0.5f, -0.5f, 0.0f);
        glTexCoord2f(1.0f, 0.0f);
        glVertex3f(0.5f, -0.5f, 0.0f);
        glTexCoord2f(1.0f, 1.0f);
        glVertex3f(0.5f, 0.5f, 0.0f);
        glTexCoord2f(0.0f, 1.0f);
        glVertex3f(-0.5f, 0.5f, 0.0f);
    glEnd();
    glPushMatrix ();
        glDisable(GL_DEPTH_TEST);
        glEnable(GL_BLEND);
        glBlendFunc(GL_DST_COLOR,GL_ONE_MINUS_DST_AL
        PHA);
        glBindTexture(GL_TEXTURE_2D, texture[2]);
        glRotatef(roll, 0.0, 0.0,1.0);
        glBegin(GL_QUADS);
            glTexCoord2f(0.0f, 0.0f);
            glVertex3f(-0.5f*0.6287, -0.5f*0.6287, 0.0f);
            glTexCoord2f(1.0f, 0.0f);
            glVertex3f(0.5f*0.6287, -0.5f*0.6287, 0.0f);
            glTexCoord2f(1.0f, 1.0f);
            glVertex3f(0.5f*0.6287, 0.5f*0.6287, 0.0f);
            glTexCoord2f(0.0f, 1.0f);
            glVertex3f(-0.5f*0.6287, 0.5f*0.6287, 0.0f);
        glEnd();
    glPopMatrix ();
glPopMatrix ();
```

그림 8. 계기판을 위한 코드의 일부분
Fig. 8. Part of Code for Instrument Panel

Ⅳ. 비행 시각화 도구의 구현

본 논문에서 구현하는 전체적인 시각화 시스템은 [그림 9]에서 보이고 있다. 수집한 비행정보 DB에서 사용자 UI을 이용하여 원하는 비행정보를 선택한다. 선택된 비행정보는 비행시각화 엔진과 계기판 엔진에 각각 제공되는데 데이터의 동기화를 위하여 공유 메모리를 사용하여 전달된다. 제안하는 시각화 시스템을 구현하기 위한 환경은 윈도우즈 기반에서 C++언어로 구현되었고 OpenGL을 이용하여 그래픽을 표현하였다. OpenGL은 산업 표준화된 그래픽 라이브러리로서

향후 내장형 시스템 등에 개발할 때 사용가능성이 유리하다는 장점을 가진다.

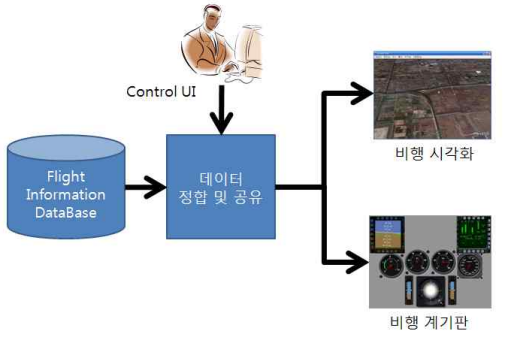


그림 9. Google Earth을 이용한 비행정보 시각화 시스템의 구조
Fig. 9. An architecture of Flight Information Visualization System using Google Earth

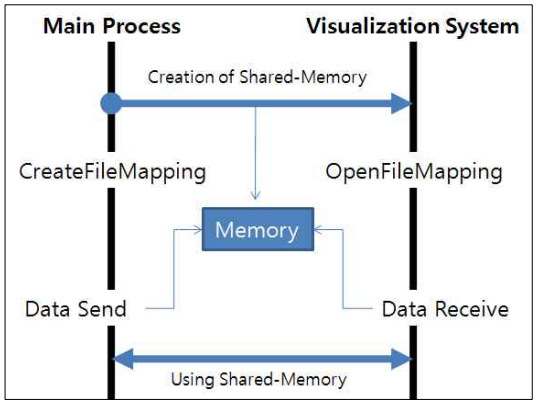


그림 10. MMF을 이용한 데이터 전달
Fig. 10. Data Transmission Using MMF

MMF(Memory Mapped File)을 이용하여 프로그램 간에 데이터를 주고받을 수 있는데 파일과 메모리 객체를 연결시킨 후 그 메모리 객체에 이름을 지어준다. 그 이름을 이용하여 메모리에 데이터를 읽고 쓰는 작업을 하게 되면 그 내용이 메모리와 연결되어 있는 파일에 똑같은 효과가 적용되는 구조이다. 본 시스템에서는 파일을 사용하지 않고 특정 메모리 객체를 생성하여 통신한다. 파일명을 적는 곳에 INVALID_HANDLE_VALUE 라고 해주면 메모리 객체만 생성이 된다. 그리고 이 메모리 객체에 이름을 설정해 주면 공유 메모리의 생성이 완료된다. 이렇게 하면 메모리상에 메모리 영역이 확보되고 각각의 프로세스들은 이 영역을 설정해 둔 이름으로 찾아서 접근(데이터 읽기, 쓰기)하여 작업을 하게 된다.

그림 11. 이륙상태의 비행 시각화
Fig. 11. Flight Visualization of Takeoff

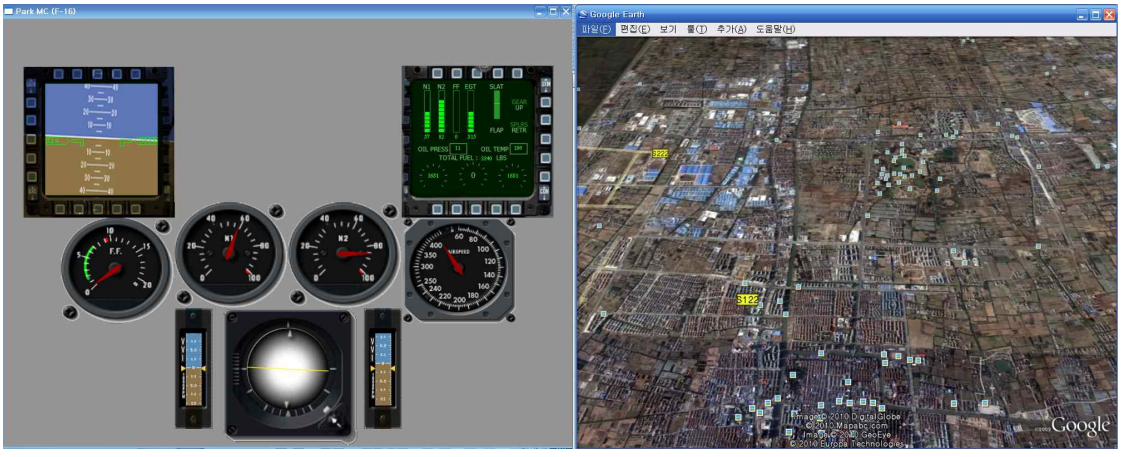


그림 12. 시각화 시스템의 통합
Fig. 12. Integration of Visualization System

[그림 10]은 MMF을 이용한 프로그램 간의 통신을 위한 구조도를 보이고 있다. 먼저 읽고 쓰기를 위한 메모리 객체를 생성하기 위해서는 CreateFileMapping (INVALID_HANDLE_VALUE, NULL, PAGE_READWRITE, 0, 1024, "FLIGHT_INFO")을 이용하여 "FLIGHT_INFO"라는 공유메모리를 만든다. 그리고 시각화 및 계기판 시스템에서는 OpenFileMapping (FILE_MAP_ALL_ACCESS, FALSE, "FLIGHT_INFO")을 이용하여 데이터를 참조한다. 데이터의 읽고 쓰기는 Map ViewOfFile을 이용하여 포인터를 생성하여 접근하면 된다.

비행 시각화 시스템은 Google Earth의 카메라 위치를 공유 메모리로 전달된 정보를 이용하여 IApplicationGE 인터페이스의 SetCameraParams() 메서드를 호출하여 구현하였다. [그림 11]은 비행정보에 따라 이륙하는 항공기를 시간차에 따라 화면 캡쳐 한 예이다. 그리고 [그림 12]는 비행 시각화 시스템과 계기 시스템이 연동되어 동작되는 모습을 보이고 있다. 시각화 시스템의 동작 중에 사용자에게는 실시간 비행정보의 변화 추이를 볼 수 있는 모니터링 화면이 제공된다. [그림 13]은 모니터링을 위한 사용자 화면인데 좌측의 각 파라미터에 대해 차트영역을 지정하면 해당 데이터가 시간흐름에 다라 하단 영역에 표시된다. 상단의 시트에는 비행정보가 기록되고 사용자가 클릭을 하면 해당 차트영역의 해당 위치가 표시된다.

이 모니터링을 통하여 불규칙적인 고도의 변화나 자세 변화를 확인하여 비행을 평가할 수 있고 향후 동일 영역비행에 대해 별도의 차트를 구성하여 비교 분석 작업을 거쳐 비행 개선에도 활용할 수 있다.

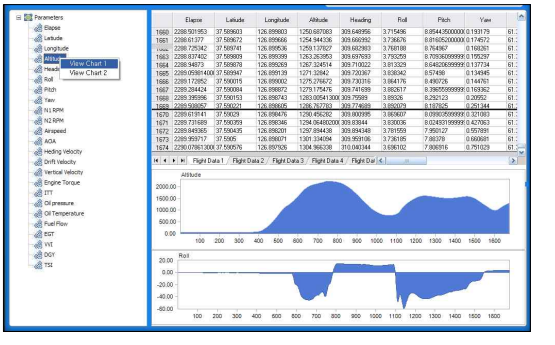


그림 13. 비행정보 모니터링 시스템
Fig.13. Flight Information monitoring System

V. 결론

본 논문은 항공기의 비행정보를 효과적으로 시각화하기 위하여 Google Earth를 이용한 시스템을 구현하였다. 구현을 위하여 OpenGL를 이용한 프로그램 모델을 사용하였고 Google Earth 에서 제공하는 상세한 위성 영상 정보를 이용하기 위하여 COM API 를 적용하였다. 제안하는 시스템은 항공기의 다양한 장치에서 얻어진 비행정보를 그래픽 라이브러리를 이용하여 각종 계기판에 표시하고 실제 비행모습을 가시화하기 위하여 Google Earth Map에 연동하여 표시하였다. 본 연구결과는 비행평가 및 개선에 이용될 수 있고 향후, 비행소프트웨어 개발에 적용될 수 있다. 향후 연구는 다양한 통신 프로토콜에 따른 전달 정보를 통합하여 시뮬레이션 할 수 있는 시스템 환경 구축에 대한 연구를 지속할 예정이다.

참고문헌

[1] 박명철, 김용해, 하석운, “혼돈이론을 응용한 예망어구에 대한 어류반응 행동모델의 수중현상 시각화,” 한국해양정보통신학회논문지, 제 8권, 제 3호, 645-653쪽, 2004년 6월.

[2] 박명철, 박석규, “OpenMP 병렬프로그램을 이용한 그물의 수중형상 시뮬레이션 구현,” 한국컴퓨터정보학회논문지, 제 13권, 제 2호, 11-17쪽, 2008년 3월.

[3] 박명철, 박석규 “혼돈이론을 이용한 선망어구에 대한 어군 유영 시각화,” 한국컴퓨터게임학회논문지, 제 17호, 145-149쪽, 2009년 6월.

[4] FlightViz, *http://www.simauthor.com*

[5] GRAF-VISION Flight Data Animator, *http://www.spirent-systems.com*

[6] Google Earth COM API, *http://earth.google.com/comapi*

[7] 최진우, 양영규, “Google Earth를 이용한 택시 텔레매틱스 운행 이력 데이터 가시화 시스템의 설계 및 구현,” 대한원격탐사학회지, 제 25권, 제 1호, 61-69쪽, 2009년 2월.

[8] Shreiner, Dave, *“OpenGL Programming Guide,”* Addison-Wesley, 2009.

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

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

[11] Martin C. Brown, *“Hacking Google Maps And Google Earth.”* John Wiley & Sons Inc, 2006.

[12] 이강희, *“계기비행,”* 비행연구원, 2009년.

[13] 한국항공우부산업(주) 개발본부 비행계획팀, *“T-50 항공기 개발 정험으로 쓴 실전 비행시험 계측,”* 청문각, 2008년.

[14] 이봉근 · 김찬조 · 김광원 · 한기혁 · 박상재, *“T-50 항공기 개발 정험으로 쓴 실전 비행시험,”* 청문각, 2007년.

저 자 소 개

박 명 철

2007년 : 경상대학교 컴퓨터과학과 공학박사

2007년 ~ 현재 : 송호대학 전임강사

관심분야 : 시뮬레이션, 임베디드 소프트웨어, 병렬프로그래밍 및 디버깅

허 화 라

2001년 : 부산대학교 전자공학과 공학박사

2000년 ~ 현재 : 송호대학 부교수

관심분야 : Time-Delay, 모델예측제어, 원격제어로봇