블랙아웃과 레드아웃을 적용한 비행 시뮬레이션

Flight Simulation based on Blackout and Redout

유기엽, 송창우*, 정경용, 김성호
상지대학교 컴퓨터정보공학부,
인하대학교 컴퓨터정보공학부*

Ki-Youp Yu(skireu@nate.com),
Chang-Woo Song(ph.d.scw@hanmail.net)*,
Kyung-Yong Chung(kyjung@sangji.ac.kr),
Sung-Ho Kim(kimsh1204@sangji.ac.kr)

요약

비행 시뮬레이션은 항공기와 같은 비행용 장비를 조종하는 훈련이나 탑승체험 등을 목적으로 실제 비행 상황을 흉내 내는 것을 의미한다. 이와 같은 기존의 비행 시뮬레이션은 비행기의 조종과 직접적으로 관련된 시각적이고 물리적인 제어에 국한된 연구 개발에 치중하여 각종 게임이나 시뮬레이터 등에 활용하였다. 그러나 비행 시뮬레이션에 조종사의 생리적 현상을 고려하여 적용한 현실적인 연구들은 거의 없었다. 그러므로 본 논문에서는 조종사의 블랙아웃과 레드아웃 현상을 가정한 비행 시뮬레이션을 간단히 구현해보고, 발전 가능성을 가늠해본다.

Abstract

The flight simulation the fact that imitates the real situation of flight for the training or boarding experience which manages a same flight equipment with the aircraft in goal. The flight simulation of existing attached importance in the research and development which is limited in the control which is visible and physical relate with the management of the airplane directly. So, the flight simulation applied in various game or simulator etc. But, there were not researches which are actual applies in flight simulation with considers the physiological phenomenon of pilot. Therefore this paper tries the flight simulation which assumes the physiological phenomenon as the blackout and a redout of pilot. In conclusion, predicts a future.

Ⅰ. 서론

지금까지 연구 개발된 비행 시뮬레이션들은 비행 물체에서 주로 발생되는 물리적인 현상을 시각적으로 가시화하거나 인공 지능 기술을 가미한 것들이 대부분[7, 8]이다. 즉, 물리적인 현상을 가시적으로 보여주기 위한 것은 비행기의 움직임을 보여주기 위함이 가장 큰 목적이라고 볼 수 있을 것이다. 그러나 최근에 연구 개발된 신전투기 등에서는 이미 조종사의 신체적 한계를 뛰어넘는 성능을 가지고 있는 기종들이 대다수 출시되고 있다. 약 30년 전인 1980년대부터 개발된 전투기들은 과학기술 발전을 통해 최고 속력과 가속도, 기동성 등의 엄청난 발전을 거듭하여 왔었다. 그런 이유로 조종사의 신체적인 한계를 훨씬 뛰어넘는 비행의 경우, 대형 사고의 원인으로 이어지는 사실들이 많아졌다. 한 예로 F-20 타이거샤크는 1980년대 우리나라에 F-5E 제공호를 대체할 전투기로 주목받고 있었지만, 고 기동 비행으로 조종사가 실신하여 추락 사고로 이어져는 바람에 안타깝게 후보에서 탈락되었던 역사가 있었다.

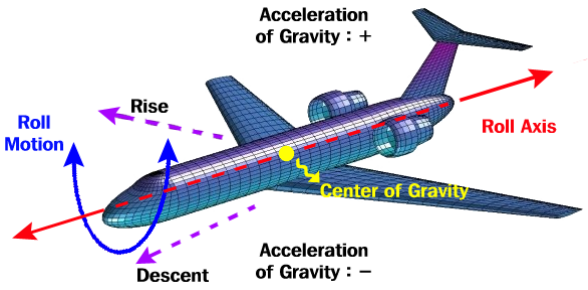
본 논문에서는 이와 같이 비행 시에 발생되는 물리적 환경에 따른 조종사의 신체적 반응을 가시적으로 표현해주기 위한 방법을 알아보고 간단히 구현해보고자 한다. 본 논문의 2장에서는 관련 용어에 대하여 살펴보고, 3장에서는 시뮬레이션 개발 환경, 4장에서는 비행 시뮬레이션의 구현과 관련된 부분에 대하여 기술하도록 한다. 그리고 마지막 5장에서는 결론 및 향후 연구과제에

대하여 기술한다.

Ⅱ. 관련 용어

1. 중력 가속도

중력 가속도(Gravity)는 지구 중력을 기준으로 동일 질량일 때 더 무거워지거나 가볍게 되는 것을 표시하는 단위이다. 예를 들어 2G는 일반적인 지구 중력의 2배수를 의미한다. 그러나 중력가속도는 단순 중력만이 아니라 가속도 변화로 인해 관성의 법칙으로 생기는 물리적 힘의 변화를 의미하기도 한다. 이러한 관성에 의한 중력가속도의 생성은 그림 1과 같이 설명할 수 있다.



▶▶ 그림 1. 항공기의 Roll Motion과 중력가속도

즉, 중력가속도는 일반적으로 방향성을 가지는데 항공기가 정면으로 날아가고 있는 상태에서 힘의 방향이 아래쪽을 향할 때에는 +G, 위쪽을 향할 때에는 -G로 표현할 수 있다. 또한 본 논문의 핵심인 블랙아웃과 레드아웃이 발생되기 위해서는 그림 1과 같이 Roll Motion이 발생되어야 하는데, 이에 대해서는 비행 시뮬레이션 구현 부분에서 언급하기로 한다.

2. 블랙아웃(Blackout)과 레드아웃(Redout)

인간은 중력가속도의 영향을 받게 되면 신체 내부에 유동성을 가진 체액이 이동하게 된다. 일반적으로 우리 인간의 신체는 약 70%가 물(액)이기 때문에 중력가속도에 따른 영향이 매우 크다. 극단적인 예로 +G 상황에서 인간의 신체는 블랙아웃(Blackout) 현상이 발생한다. 블랙아웃(Blackout)이라 함은 기체가 급선회하거나 급상승할 경우 인간의 신체에 미치는 가속도가 인간이 보유하고 있는 몸무게의 5~6배에 가깝게 되는데, 이때 인간의 몸속에 존재하고 있는 피 성분들이 양쪽 다리 부분으로 한꺼번에 집중적으로 쏠리면서 그 반대 방향의 머리 부분에는 피 성분이 매우 부족해지게 되어 발생되는 현상을 의미한다. 블랙아웃(Blackout) 현상이 발생하면, 인간의 판단 능력이 급속히 떨어지고, 시야가 흐려지며, 심할 경우에는 산소 부족으로 실신하는 현상(G-loc : Gravity Loss Of Consciousness)이 발생되기도 한다. 따라서 이와 같은 블랙아웃 현상을 방지하기 위해서는 몸이나 허벅지 등을 꽉 조임으로써 물리적인 피의 쏠림 현상을 차단하는 특수 복장(G-suit, 가속도 방호복)을 입는데, 이 또한 장치가 풀리게 되면 전문 조종사 일지라도 블랙아웃 상태에서 빠져나오지 못하게 된다. 일반적으로 신체 건강한 일반인이 견딜 수 있는 최대 중력가속도는 약 4G이다.

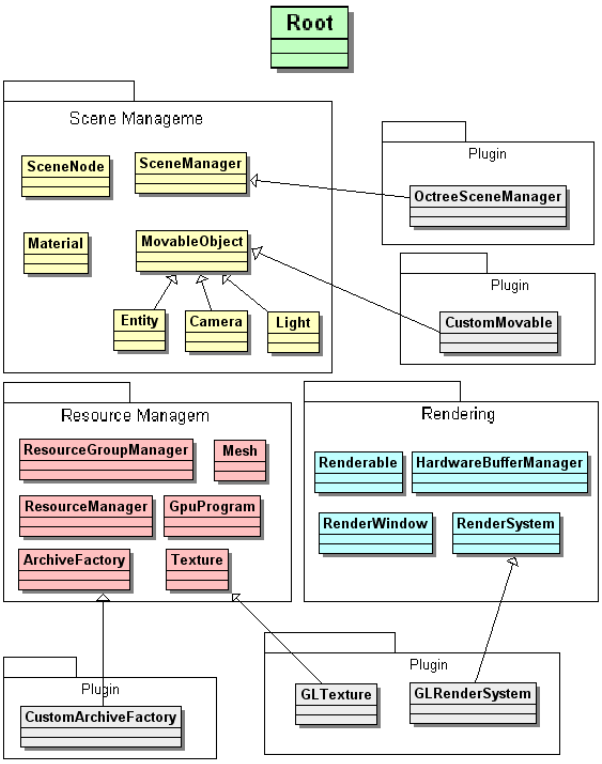
이와는 반대로 극단적인 -G 상황에서는 레드아웃(Redout) 현상이 발생한다. 레드아웃(Redout) 현상에서는 인간의 머리 부분에 피가 급속히 몰려들어 안구안쪽의 모세혈관이 터지거나 심하면 뇌혈관까지 터지는 뇌출혈을 일으킬 수 있어 블랙아웃(blackout)보다 훨씬 더 위험한 것으로 알려져 있다. 일반적으로 신체 건강한 일반인이 견딜 수 있는 최대 중력가속도는 약 -3G이다.

Ⅲ. 시뮬레이션 개발 환경

1. 게임 엔진 Ogre

게임 엔진 Ogre(이하 Ogre)[4]는 개방형 게임 엔진들 중의 하나이며, GNU의 공인 라이선스인 LGPL(Lesser General Public License)을 가지기 때문에 플러그인을 개발하는 데 제한이 없고, 수정에 대한 사실과 일자를 표기한다면 얼마든지 수정하여 사용할 수 있다는 큰 장점을 가지고 있다. 또한 유연한 객체 지향형의 구조를 가지고 있다는 장점도 가지고 있다. 이러한 Ogre는 C++ 언어를 기반으로 제작되어졌으며, 기본적으로 DirectX API와 OpenGL API를 지원하는 3D 렌더링(Rendering) 엔진이라고 볼 수 있다. Ogre의 또 다른 장점 중의 하나는 다른 응용프로그램과 쉽게 조합시킬

수 있다는 것이다. 예를 들면, PhysX와 같이 물리엔진을 함께 사용하여 다양한 물리 현상과 관련된 시뮬레이션 개발이 가능하다. 그러나 Ogre는 기본적으로 렌더링 기능이 주목적이기 때문에 음향과 같은 오디오 관련 기능은 별도로 구현을 해야만 하는 단점도 가지고 있다.



▶▶ 그림 2. Ogre의 핵심 클래스들

그림 2는 Ogre의 핵심 클래스들에 대한 도식도인데, 클래스는 크게 3가지로 구분되어진다, 즉, Scene Management, Resource Management, 및 Rendering 이 그것이다. Scene Management는 화면 구성과 각 객체의 위치를 조정한다. 그리고 Resource Management는 객체를 그리는데 필요한 자원을 관리한다. 마지막 Rendering은 화면을 모니터에 디스플레이 하는 것을 담당한다.

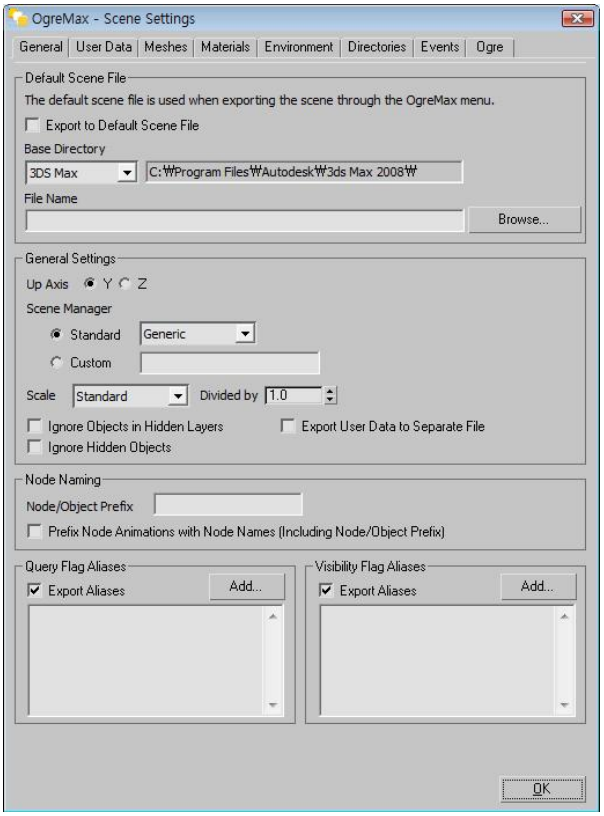
이와 같은 Ogre를 사용하기 위해서는 ExampleFrame Listener 및 ExampleApplication등을 상속받아 처음 응용프로그램이 시작될 때와 이벤트에 따른 Scene의 변화를 기술해야 한다. Scene은 SceneNode로 구성되는데 SceneNode는 여러 개의 ChildSceneNode나 Object를 가질 수 있고, 모든 SceneNode는 최상위 ParentsNode로 RootSceneNode를 가진다. 마지막으로 구현된 Scene은 Frame Listener를 통해 화면에 투상 된다.

본 논문에서 비행 시뮬레이션을 구현하기 위해 게임엔진 Ogre를 사용하는 이유는 기존에 사용되는 3D 프로그래밍 언어를 사용하여 프로그래밍 할 경우 Anti-aliasing 및 Geometry Shaders 등과 같은 기술을 직접 프로그래밍하기 보다는 좀 더 쉽게 사용할 수 있다는 장점 때문이다. 특히 Material Script를 통해 재질을 세밀하게 설정할 수 있다는 점은 Ogre의 또 다른 장점이라 할 수 있다.

2. OgreMAX

OgreMax[6]는 Ogre에 필요한 3D 모델들을 파일로 만드는데 사용되는 3DS MAX의 플러그인으로서, 3DS MAX를 이용해서 제작한 모델들을 Export하여 Ogre에서 바로 사용할 수 있도록 해준다. 3DS MAX에서 Export되는 파일 포맷은 *.mesh, *.material 및 *.scene 등으로 분리되어 생성된다. 여기서 *.mesh 파일은 모델링된 데이터가 저장된 파일이고, *.material 파일은 표면에 대한 설정이 저장된다. 마지막으로 *.scene 파일은 *.mesh와 *.material을 통합한 전체적인 장면이다. *.mesh 파일은 내부에 필요한 material의 설정 이름이 포함되고, 이 material 설정은 *.material 파일 이름과 관계없어 신경을 쓰지 않는다면, 동일한 이름의 material의 설정을 Ogre에 올림으로써 충돌할 가능성이 높다. 특히 구현 시에는 *.mesh 파일과 *.material 파일의 이름을 동일하게 만들어야만 해당 오류가 생겼을 때 찾기가 쉽다.

3D 객체를 모델링을 할 때 Export 하는 과정을 염두에 두고 제작하여야만 나중에 Export 과정에서 문제가 생기는 것을 방지 할 수 있다. 특히 Ogre Max에서 완벽히 지원하지 않는 기능을 사용한다면, Export 과정에서 *.Mesh 파일이 생성되지 않는 문제가 생길 수 있으므로 주의를 해야 한다. 그림 3은 OgreMAX의 Scene Setting 위도우로서 Export 되는 항목들에 대한 속성을 설정할 수 있다.



▶▶ 그림 3. OgreMax의 Scene Setting

Ⅳ. 비행 시뮬레이션의 구현

본 논문에서 비행 시뮬레이션을 구현함에 있어서 필요한 3D 우주 공간의 배경은 Skybox 원리를 사용하였다. 그리고 비행 시뮬레이션 실험에 사용한 항공기나 우주선은 전방으로 나아가면서 상승 및 하강을 통해 중력가속도가 발생하는 상황을 연출하였다. Skybox는 카메라를 중심으로 정지된 모습으로 보이기 때문에 각종 행성과 소행성들을 우주 공간에 적당히 배치시켜 항공기나 우주선이 이동하고 있는 입체적 속도감을 느낄 수 있도록 하였다.

레드아웃과 블랙아웃을 구현하는 방법은 카메라의 시야 밖에 반투명한 필터를 만들고 레드아웃과 블랙아웃의 조건에서 각 필터를 이동시켜 카메라의 시야에 들어오도록 만들어 흐릿하거나 붉은 빛이 보일 수 있도록 구현하였다. 그리고 이를 카메라와 연동시켜 함께 움직일 수 있도록 구축하였는데, 만약 필터가 카메라와 함께 움직이지 않을 때에는 카메라의 시야에 필터의 일부가 보일 수 있도록 하였다.

레드아웃과 블랙아웃이 발생되기 위한 전제 조건으로는 일정한 속도와 일정한 Roll 각도(그림 1 참조)가 있어야 하는데, 표 1은 비행 속도와 Roll 각도에 따른 중력가속도와의 관계를 보여주고 있다. 그리고 Roll 되는 방향에 따라 중력가속도의 부호가 결정된다.

표 1. 비행 속도와 Roll 각도에 따른 중력가속도

	속도 빠를 때	속도 느릴 때
Roll 클 때	높은 G	낮은 G
Roll 작을 때	낮은 G	G 없음

만약 키보드 인터페이스를 사용할 경우, 키가 입력되는 것과 입력되지 않는 두 가지의 경우 밖에 없다. 이와 같은 경우에는 키가 입력될 때 각도가 일정하게 변할 수밖에 없기 때문에 순간적인 Roll 각도가 변화되지 않는다. 이러한 이유로 마우스 인터페이스를 이용하여 가능한 아날로그에 가까운 모습을 보일 수 있도록 구현하였다.

▶▶ 그림 4. 비행 시뮬레이션의 초기 화면

그림 4는 구현된 비행 시뮬레이션이 실행된 초기 화면이다. 항공기나 우주선이 고속으로 비행 중인 상태에서 하강이나 상승으로 인한 -G와 +G가 생길 때는 그림 5와 같은 화면으로 전환되어짐을 확인할 수 있다.

▶▶ 그림 5. 블랙아웃과 레드아웃을 위한 필터 적용 상태 - 항공기의 상승 및 하강에 의한 변화

즉, 그림 5에서는 항공기나 우주선이 급작스러운 상승이나 하강 시에 발생되는 블랙아웃과 레드아웃을 가시적으로 잘 표현해주고 있음을 확인할 수 있었다.

V. 결론

조종사가 비행 시뮬레이션 시스템을 사용하여 다양한 비행 훈련을 할 수 있다는 것은 고가의 고급 연료를 사용하지 않고, 안전하게 비행 훈련을 충분히 할 수 있다는 장점 때문이다. 그러나 기존의 비행 시뮬레이션은 비행기의 물리적인 상태에 관한 연구 개발에 집중되어져 있었으며, 비행 시 발생하는 생리적 현상에 대한 가시적인 구현 결과는 거의 없었다. 그러므로 본 논문에서는 비행기의 물리적인 상태가 아니라 조종사의 생리적인 현상을 비행 시뮬레이션에 적용시켜보았으며, 그 결과로 조종사에게는 비행 시뮬레이션의 훈련 효과가 상승할 것이라 판단되어진다. 특히 고기동성 고속의 전투기는 가혹한 G와 비행슈트에 의한 압력으로 인한 사고의 위험성이 항상 내재되고 있는 것이 사실이다. 그러므로 더욱 높은 기술 수준을 겸비한 차세대 비행 시뮬레이션의 연속적인 연구 개발은 치명적인 사고의 위험을 최대한 줄여줄 수 있다고 볼 수 있다. 결국 현실에서 조종사들이 보고 듣고 느끼는 모든 물리적이고 생리적인 현상들을 차세대 비행 시뮬레이션에서 구현하고 실험하여 실전과 같은 훈련을 할 수 있다면, 불의의 사고로부터 귀중한 생명을 구하는데 매우 큰 도움이 될 것으로 판단된다.

감사의 글

"본 연구는 지식경제부 및 정보통신연구진흥원의 대학 IT 연구센터 지원사업의 연구결과로 수행되었음" (IITA-2009-C1090-0902-0020)

■ 참 고 문 헌 ■

[1] 세계 최신예기 '96, 모델 팬 편집부, 1996.
[2] Luna, Frank D., "Direct X를 이용한 3d game 프로그래밍 입문", 정보문화사, 2004.
[3] 주우석, "OpenGL로 배우는 컴퓨터 그래픽스", 한빛미디어, 2006.
[4] Ogre Game Engine, http://www.ogre.org
[5] 백광우, "3DS MAX 5 고급 활용 : 아트에서 애니메이션까지", 성안당, 2003.
[6] OgreMax, http://www.ogremax.com
[7] Randolph M. Jones, John E. Laird, Paul E. Nielsen, Karen J. Coulter, Patrick G. Kenny, Frank V. Koss, "Automated Intelligent Pilots for Combat Flight Simulation", AI Magazine 20(1): 27-41, 1999.
[8] 김형철, 김강수, 노재춘, 윤석준, "MAV (Micro Air Vehicle) 비행 시뮬레이터의 설계", 한국시뮬레이션학회 2003년 춘계학술대회논문집, pp.49-54, 2003.