공학석사 학위청구논문

OpenGL 기반 그래픽 서브시스템의 성능평가에 관한 연구

A Study on the Performance Evaluation of OpenGL-based Graphics Sub-System

지도교수 정 재 현

2004년 2월

한국해양대학교 대학원 기계공학과

이 동 훈

목 차

1. 서 론 ····· 1

1.1 연구배경 ····· 1

1.2 연구내용 ····· 1

1.2 연구목적 ····· 2

2. OpenGL 3D API 성능 평가 환경 설정 ····· 3

2.1 Graphics Sub-system ····· 3

2.2 OpenGL 3D API ····· 4

2.3 Specviewperf ····· 6

2.4 테스팅 플랫폼 ····· 12

3. OpenGL 3D 성능 SPECViewperf 테스트 ····· 13

3.1 OpenGL 3D Performance 테스트 ····· 13

3.1.1 테스트 계획 ····· 14

3.1.2 OpenGL Settings 테스트 ····· 17

3.1.3 Default Color Depth, Buffer-Flipping Mode, Vertical Sync 테스트 ····· 22

3.2 OpenGL 그래픽 서브시스템 최적화 성능테스트 ····· 27

3.3 OpenGL 그래픽 서브시스템 성능저하 환경테스트 ····· 29

4. SPECViewperf 테스트결과 분석 및 평가 ····· 31

5. 결 론 ····· 35

A Study on the Performance Evaluation of OpenGL-based Graphics Sub-System

Lee Dong Hoon

Department of Mechanical Engineering

Graduate School, Korea Maritime University

Abstract

The company, SGI, developed the new 3D API, OpenGL and opened the specification of OpenGL to the public. Many other companies which try to develop the graphic hardware system are maintaining and advancing the products related OpenGL with OpenGL specification.

The softwares which are accelerated by OpenGL show various results according to the detail function in the accelerating device, OpenGL. However, there is a loss in the performance of the system hardware because users are not fully prepared for understanding the graphic hardwares and 3D API.

In this thesis, 3D performance environmental settings supporting the OpenGL 3D API Graphic Sub-system were tested with SPECviewperf 7.1 based on Pro/Engineer 2001 which includes two models in shaded, wireframe, and hidden-line removal, three modes and seven viewsets.

Finally, this research aims for organizing the optimal environment of major application which used by users with the result of 3D performance analysis.

1. 서 론

1.1 연구배경

실리콘 그래픽스사가 다양한 그래픽 하드웨어에서 작동하는 그래픽 소프트웨어의 제작을 위해 OpenGL을 새로운 3D API로 개발하여 공개한 이후, 현재 OpenGL은 CAD/CAM/CAE 어플리케이션은 물론 일반 3D 그래픽 어플리케이션 제작에 있어서도 표준적인 소프트웨어 툴 키트로 인식되게 되었다.

OpenGL이 지금까지 개발된 수많은 3D API 가운데 이러한 성과를 가지게 된 것은 다양한 하드웨어에서 구동이 가능하도록 스펙이 공개되었다는 점과 이러한 스펙을 이용하여 많은 그래픽 하드웨어 시스템 개발 업체에서 OpenGL를 하드웨어적으로 가속 처리할 수 있도록 전용 하드웨어를 개발하고 지속적으로 그 성과를 발전시키고 있기 때문이다.

하지만 OpenGL에 의해 가속되는 소프트웨어는 그 가속 장치의 세부적인 기능에 따라 다양한 결과를 제공한다. 이러한 가속 시스템은 개발 업체의 독자적인 그래픽 프로세서로 연구되어 컴퓨터 본체 시스템을 능가하는 성능과 가격을 요구하는 경우도
흔하다.[1]

현재 CAD/CAM 등 3D 그래픽 어플리케이션의 운용이나 다양한 그래픽 기능의 응용이 필요한 많은 공학 분야에서 사용되는 고가의 OpenGL 그래픽 서브시스템은 사용자 운용 측면에서 그 성능이 제대로 발휘되기에는 여러 가지 문제가 있다. 이것은 그래픽 서브시스템 지원 어플리케이션의 운용 뿐만 아니라 개발자 측면에서 그래픽 하드웨어 및 3D API의 인식 및 이해 부족에 기인하고 있는 점이 많다.[2] 실제로 많은 OpenGL 3D API 전용 그래픽 서브시스템들은 다양한 환경 설정 기능을 제공하고 있다. 그러나 앞서의 언급처럼 그 환경에 대한 운영 능력 부족은 시스템 하드웨어 성능 저하를 초래하게 될 수 있다는 것이다. 또한, 그래픽 소프트웨어 및 하드웨어 연구 개발에 있어서도 이러한 내용에 대한 준비는 매우 필수적이다.[2][4] 그러나, 현재 국내 공학 분야에서의 OpenGL 3D API 지원 하드웨어 및 소프트웨어에 대한 평가나 분석은 거의 사용자 후기 수준에 불과한 실정이다.

1.2 연구내용

본 연구는 이러한 상황에서 OpenGL 3D API 지원 그래픽 서브시스템에 대한 3D 기능 환경 설정 및 성능 분석에 대한 내용을 시도한다. 실제로 OpenGL 3D API 그래픽 서브시스템은 지금까지 다양한 하드웨어로 개발되어 왔지만, 현재 가장 보편적인 장치로 3DLabs의 Wildcat 시리즈, nVidia Quadro 시리즈 그리고 ATI Fire GL 시리즈를 들 수 있다.

본 연구에서는 이 중 가장 그 사용 빈도가 높고 기능이 다양한 nVidia Quadro 그래픽 프로세서에 기반한 OpenGL 가속 그래픽 서브시스템에 대한 OpenGL 환경 설정 및 성능 평가 분석을 시도하였다. 이를 위해 가장 보편적인 그래픽 워크스테이션의 환경 구성으로 IA-32 Pentitum 4 기반 Windows 2000 운영체제를 선택하였다. 실제 워크스테이션 환경은 현재 UNIX를 운용체제로 하는 RISC 기반 워크스테이션과 마이크로소프트의 Windows 워크스테이션과 GNU /Linux 운영체제로 하는 IA-32 및 호환 프로세서 기반 워크스테이션을 사용하는 경우가 있지만, 많은 경우 가격적인 문제와 운영 측면에서 Windows 워크스테이션 환경을 선택하고 있다는 점에서 일차적인 조건을 결정하였다.

1.3 연구목적

본 연구에서는 선택된 그래픽 서브시스템에 대한 OpenGL 지원 어플리케이션 운용 능력에 대한 벤치마킹 프로그램을 이용하여, 그 결과로 성능 평가를 수행한다. 이를 위해 그래픽 서브시스템의 주요 환경 설정 및 선택 사양에 대한 제어로 각각의 평가를 수행하여 최고 성능 및 최적 성능 그리고 부하가 요구되는 환경에 대한 설정을 분석한다. 이를 통하여 사용자가 사용하는 주요 어플리케이션의 최적 환경에 적합한 환경 구성을 할 수 있도록 하는 것을 목표로 한다. 이를 위해 본 연구에서는 통합 CAD/CAM 어플리케이션의 대표 중에 하나인 PTC의 Pro/Engineer에 기반한 SpecViewperf 프로그램을 평가하였다.

2. OpenGL 3D API 성능 평가 환경 설정

2.1 그래픽 서브-시스템

워크스테이션이 등장하기 전 개인용 컴퓨터는 특별한 업무용이기 보다는 개인 생활에 필요한 다양한 분야에 적용되어 있었다. 하지만, 개인용 컴퓨터의 발전과 메인프레임의 기능을 필요로 하는 연구 개발 분야의 확장은 보다 향상된 성능의 스탠드-얼론 시스템을 요구하게 되었고, 그 결과 Apollo에 의해 엔지니어링 워크스테이션이라는 새로운 컴퓨터 머신이 등장하게 된다.

엔지니어링 워크스테이션이 개인용 컴퓨터와 구별되는 것은 32-비트 이상의 빠른 마이크로프로세서와 즉각적인 다양한 네트웍 환경 접속 기능 그리고 빠른 3차원 그래픽 처리 능력을 제공한다는 것이다. 특히, 3차원 그래픽 처리 능력의 제공이 이전 메인프레임이나 슈퍼 미니컴퓨터 시스템에서의 고성능 작업을 데스크 탑으로 이전할 수 있게 되어, 워크스테이션의 보급에 급속한 배경이 되었다.[3][5]

워크스테이션의 3차원 그래픽 하드웨어는 이를 뒷받침하는 새로운 의미의 라이브러리인 3D API의 등장으로 보다 그 활용성이 커지게 되었다. 주요 3D API로 사용되었던 제품은 GKS에서 시작하여 PHIGS/PHIGS+, PEX 등 범용 3D API와 HP Starbase 등 전용 3D API 등 여러 종류가 사용되었다. 이러한 3D 그래픽 하드웨어와 3D API를 일반적으로 그래픽 서브-시스템이라고 한다.

그래픽 서브-시스템이라고 한 것은 기본적으로 본체 시스템과 분리된, 즉 CPU를 중심으로 한 메인 프로세서와 별도로 존재하는 시스템이라는 의미이지만, 이것은 물리적인 구분과 함께 논리적인 구분도 함께 포함하고 있다. 최근 개인용 컴퓨터 수준에서 볼 때 CPU와 메모리 시스템이 존재하는 마더보드의 확장 슬롯을 통하여 직접 그래픽 컨트롤러가 장착된 형태이기 때문에, 하드웨어 본체와 그래픽 시스템의 물리적인 구분과 함께 운영체제에서 작동하는 그래픽 드라이버를 통한 논리적인 구분을 함께 고려할 수 있다.[7]

현재 고성능 워크스테이션의 경우, 이미 OpenGL이 3D API의 표준으로 유지되고

있는 현실에서, 그래픽 서브-시스템은 OpenGL를 하드웨어적으로 가속 처리할 수 있는 그래픽 컨트롤러를 지칭한다고 볼 수 있다. 물론, 앞서의 언급에서처럼 현재 대부분의 워크스테이션용 3D 그래픽 컨트롤러에서는 OpenGL 3D API를 지원하고 있다.[8]

2.2 OpenGL 3D API

OpenGL은 Windows NT 기반 하에서 그래픽 카드를 직접 제어하는 표준 인터페이스를 의미한다. OpenGL을 개발한 실리콘 그래픽스사(SGI)는 전통적으로 강력한 3D 그래픽 기능을 제공하는 워크스테이션과 서버의 공급으로 유명하지만, 자사의 IRIX 운영체제 환경을 위한 IRIS-GL을 통한 다양한 그래픽 소프트웨어 개발 지원으로 그 유명세를 더하게 된다. SGI는 3차원 그래픽 워크스테이션과 소프트웨어의 보다 광범위한 보급을 위해 SGI 시스템에 최적화되어 있던, IRIS-GL을 범용 시스템에서 운용할 수 있도록 그래픽 운영체제와 분리된 새로운 3D API로 개발하게 되고 OpenGL이라는 이름으로 공개하게 된다.

초기 IRIS-GL과 마찬가지로 IRIX 기반 SGI 워크스테이션에서 OpenGL이 운용되던 시절에서는 PHIGS/PHIGS+와 PEX 등 강력한 X-Window System 기반 3D API와 경쟁하게 되고, 특별한 성과를 보이지는 못하고 일반적인 3D API의 하나로 시작된다. 하지만, PHIGS/PHIGS+와 PEX는 공급 체계의 구조적인 문제점을 가지고 있었다. 게다가 새롭게 워크스테이션 운영체제로 등장한 Microsoft의 Windows NT 기반 운영체제에서의 OpenGL을 채택함으로 OpenGL이 제공하는 3D API의 호환성은 가장 큰 그래픽 어플리케이션 개발 요소로 등장하게 되었다. Windows NT 기반 퍼스널 워크스테이션의 보급과 개인용 컴퓨터의 급속한 발전은 PEX 등 그 작동 환경이 상대적으로 제약적인 3D API에 비하여 높은 시장성을 가지게 되어, 현재 대부분의 그래픽 하드웨어 및 소프트웨어 공급 업체들은 OpenGL을 핵심 3D API로 선택되고 있다.

Open GL은 원래 그래픽 처리 라이브러리이다. 실제 우리가 3D를 처리하는 프로그램을 만들기 위해서는 대단히 까다로운 그래픽 루틴을 만들어야 하지만, 이것만으로도 엄청난 작업이기 때문에 단지 3D 그래픽을 실행하고 처리하기 위한 소프트

웨어를 만드는 것도 쉽게 할 수 있는 성질의 것은 아니다. 이렇듯 까다로운 3D 그래픽을 쉽게 하기 위해서 만들어진 라이브러리가 OpenGL이다. OpenGL 라이브러리를 사용함으로써 프로그래머는 3D 그래픽을 어떻게 계산하고 화면에 나타내어야 하는가에 대한 어려움에서 벗어나서, OpenGL과 같은 라이브러리에서 준비된 함수들을 사용하여 라이브러리에서 이에 필요한 계산 처리들을 해주고 프로그래머는 단지 그래픽 자체에만 골몰하는 창조적인 작업에 더 많은 시간을 들일 수 있게 되었다.[8][10][11]

SGI사에 따르면 OpenGL은 '그래픽 하드웨어를 위한 소프트웨어 인터페이스'라고 정의되어 있다. 이 말은 OpenGL이 3차원 모델을 만들거나 렌더링 하는, 우리가 잘 알고 있는 3D 스튜디오 같은 그래픽 소프트웨어나 3차원 그래픽을 하기위한 프로그래밍 언어가 아니라는 것을 의미한다.

OpenGL은 3차원 그래픽을 이용하는 응용프로그램을 만들기 위한 API이다. 원래 게임이 아닌 그래픽전문가용 라이브러리로 개발된 것으로 게임에서 주로 사용되는 DirectX에 비해 월등한 3D 알고리즘을 제공하고 있다. 하지만 Full OpenGL을 수행하기 위해서는 그래픽칩셋 자체의 부동소수점연산기능을 지닌 Geomerty엔진이 탑재 되어있어야 하며, ICD(Installable Client Driver)드라이버 지원도 필수적 이다.[11]

이렇게 까다로운 실행환경이 있었지만 게임을 제작하는 회사인 ID사에서 새로운 3D 전략물인 Quake 엔진을 OpenGL API기반으로 제작하기 시작하였고, OpenGL이 객체모델을 사용하지 않는다는 점과 게임에서 OpenGL의 모든 기능이 사용되지 않는 점을 착안해 새로운 형식의 OpenGL 시스템 파일을 만들게 되었다. 이것은 완전한 Full OpenGL이 아닌 OpenGL의 일부만 사용하는 새로운 형식의 알고리즘을 개발하였던 것이다. 결과는 Full OpenGL 드라이버인 ICD (installable Client Driver)에 대한 제한적으로 지원하게끔 만든 MCD(Mini Client Driver)였다.[4][10][11]

이렇게 게임과 같은 대중적인 힘을 바탕으로 OpenGL은 한정된 사람들만이 쓴다는 고정적 이미지를 벗어나기 시작했다. 또한 전문성에 입각한 OpenGL도 계속적인 발전이 있었다. 이는 3DLabs의 Wildcat, nVidia Quadro, FireGL Series와 같은 고급 하드웨어 출현과 함께 다양한 고급 그래픽 어플리케이션 들이 등장하게 되었으며, 현존하는 최고의 그래픽 소프트웨어 솔루션으로서 자리잡게 되는 결과를 낳았다. OpenGL은 실리콘그래픽스는 물론이고 여러 유닉스 시스템과 윈도우 NT에도

포팅 되어 있으므로 다른 운영체제들과의 호환성에서도 뛰어난 면을 보여준다.[12]

OpenGL의 보급과 전용 그래픽 가속 시스템의 등장은 이전 범용 3D API의 표준인 PEX의 쇠퇴를 가져왔다. 또한 많은 그래픽 하드웨어 업체들은 OpenGL을 표준으로 인정하여, 최근 하드웨어들은 이전 3D API의 하드웨어 지원 대신 OpenGL에 대한 지원이 증가하고 있는 실정이다.

현재 OpenGL은 개인용 컴퓨터에서 시작하여 그래픽 서버 그리고 슈퍼 컴퓨터에 이르는 다양한 실행 환경을 가지게 되었다. OpenGL 등장 이후, 실제 기존 3D API의 시장 방어를 위한 노력과 새로운 3D API의 등장으로 OpenGL은 계속적으로 경쟁 속에서 발전되고 있다. 현재 OpenGL은 개인용 컴퓨터와 Windows NT 기반 워크스테이션 시장에서는 Microsoft의 Direct 3D의 강력한 도전으로, CAD/CAM 등 3D API를 필요로 하는 그래픽 소프트웨어 시장에서 게임 등 다양한 멀티미디어 환경으로의 보급에 대한 희망이 큰 위험을 맞이하고 있다. 그러나, 여전히 워크스테이션 이상의 고성능 그래픽 하드웨어에서는 거의 경쟁이 없는 실정이라는 점에서, 본 연구에서의 OpenGL의 채택은 당연한 결과로 볼 수 있다.

OpenGL에서의 새로운 스펙의 추가와 수정은 SGI를 포함한 대부분의 그래픽 하드웨어와 소프트웨어 개발 업체들이 참가하고 있는 비영리 기구인 OpenGL ARB에 의해 유지 발전되고 있다. 현재 OpenGL은 Specification 1.4가 공개되어 있으며, 그래픽 하드웨어 공급 업체들의 최신 스펙에 대한 지원이 서서히 증가하고 있다.

2.3 SpecViewperf

OpenGL이 주요 3D API의 하나로 등장한 이후, 수많은 그래픽 하드웨어 개발사들은 OpenGL 전용 그래픽 서브-시스템을 위한 장치들을 출시하게 된다. 특히, 개인용 컴퓨터 환경과 Window NT 운영 환경에 기반한 저가 그래픽 워크스테이션 분야에서는 출시되는 그래픽 하드웨어 절대량이 기존 RISC 기반 UNIX 워크스테이션에 비해 엄청나게 증가하게 되었다. 하지만, OpenGL 3D API 가속 처리를 위해 동일한 그래픽 프로세서를 사용하는 경우에도 PCB의 설계와 추가적인 유틸리티 기능으로 그 성능의 차이가 크다는 점은 OpenGL 그래픽 서브시스템 시장에 대한 사용

자의 선택을 크게 혼란시키고 있다.

이에 대해 다양한 하드웨어에 대한 벤치마킹을 주관하고 있는 SPEC은 CAD/CAM 등 3D 그래픽 하드웨어에 대한 공정한 평가를 위한 방법을 제시하게 된다. 현재 SPEC의 그래픽 하드웨어에 대한 평가를 분류하면 크게 표준적이며 범용적인 OpenGL 환경에서의 3D 가속 처리를 시험하기 위한 SPECViewperf 시리즈와 개별적인 CAD/CAM 등 컴퓨터 그래픽 어플리케이션에서의 성능 평가를 위한 SPECapc 시리즈를 들 수 있다. 실제 두 테스트 간에는 어느 정도의 차이가 있는 것이 사실이지만, 일반적으로 SPECViewperf를 통한 평가가 일반적 성능의 평가 척도로 이해되고 있으며, SPECViewperf 프로그램 자체의 성능과 기능 개선으로 그 신뢰성은 계속적으로 증가하고 있다.

본 연구에서는 SPECViewperf 7.1의 CAD/CAM 관련 벤치마킹 테스트 중에 하나인 proe-02 viewset를 선택하여 OpenGL 환경의 평가와 그 설정 환경의 변화를 검증하였다. proe-02 viewset는 가장 뛰어난 3D 모델링 기능을 제공하고 있는 통합 CAD/CAM 프로그램의 하나인 PTC Pro/Engineer 2001에서 생성된 모델에 대한 가속 처리 테스트를 각 그래픽 하드웨어에 대해 주요 기능을 평가하고 있다.

SPECViewperf 7.1의 proe-02는 Pro/Engineer 2001에 의해 생성된 두개의 모델을 세가지 렌더링 모드에 대해 평가한다.

첫 모델인 PTC World Car는 4~6백만개의 정점으로 이루어져 있고, 쉐이딩 모드, 은선-처리 모드 그리고 와이어 프레임 모드에 대한 가속 처리를 평가한다. 또한 와이어 프레임 모드에 대해서는 일반적인 모드와 안티-알리아싱 모드에 대해 각각 평가한다.

두 번째 모델인 복사기는 50만개에서 150만개의 정점을 가지도록 구성되고 있고, 쉐이딩 모드와 은선-처리 모드에 대해 평가한다. 각 viewset은 Pro/Engineer 2001에서 렌더링 과정을 통해 그래픽 서브시스템의 상태 변화를 점검하게 된다.[13]

각 viewset의 지정 값은 Table 1과 같다.

Table 1 각 viewset의 지정값

Test	Weight	Description
1	13.5	PTC world car shaded
2	13.5	PTC world car shaded to show model detail
3	13.5	PTC world car using hidden line removal mode
4	16.0	PTC world car using wireframe mode
5	16.5	PTC world car using anti-aliased lines
6	13.5	PTC copier shaded
7	13.5	PTC copier using hidden line removal mode

각각의 viewset에 대한 실행 화면은 Fig. 1.1~Fig. 1.7의 그림과 같다.

Fig. 1.1 PTC world car shaded

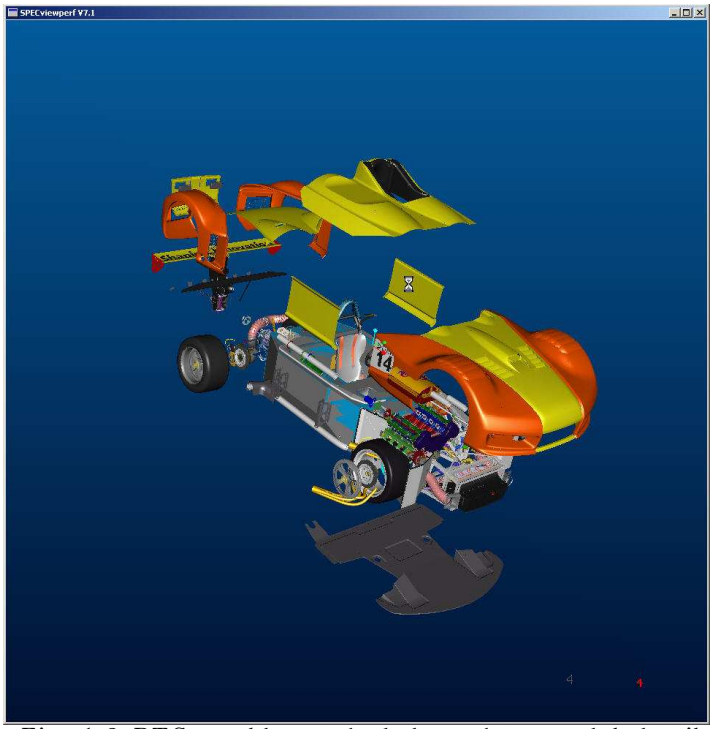

Fig. 1.2 PTC world car shaded to show model detail

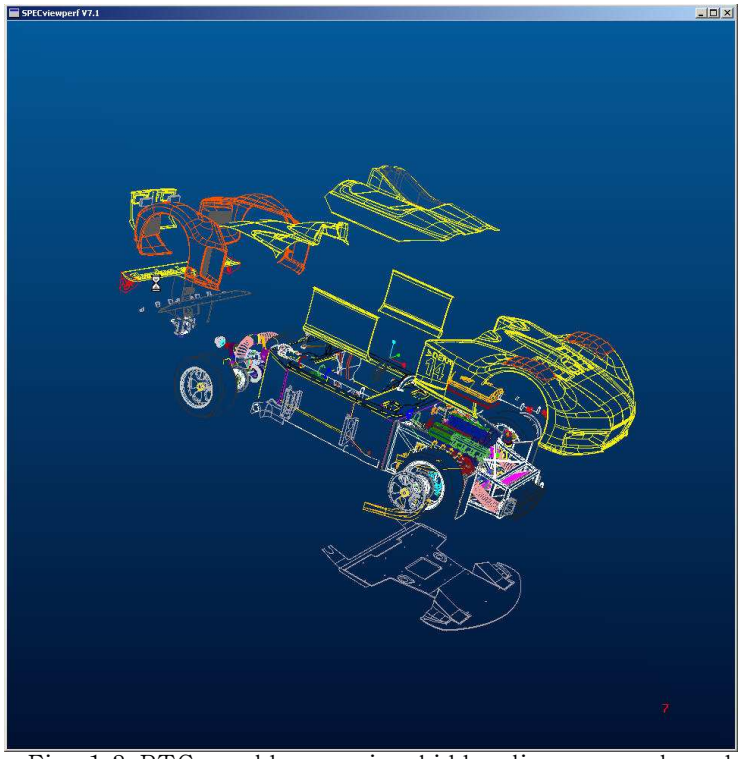

Fig. 1.3 PTC world car using hidden line removal mode

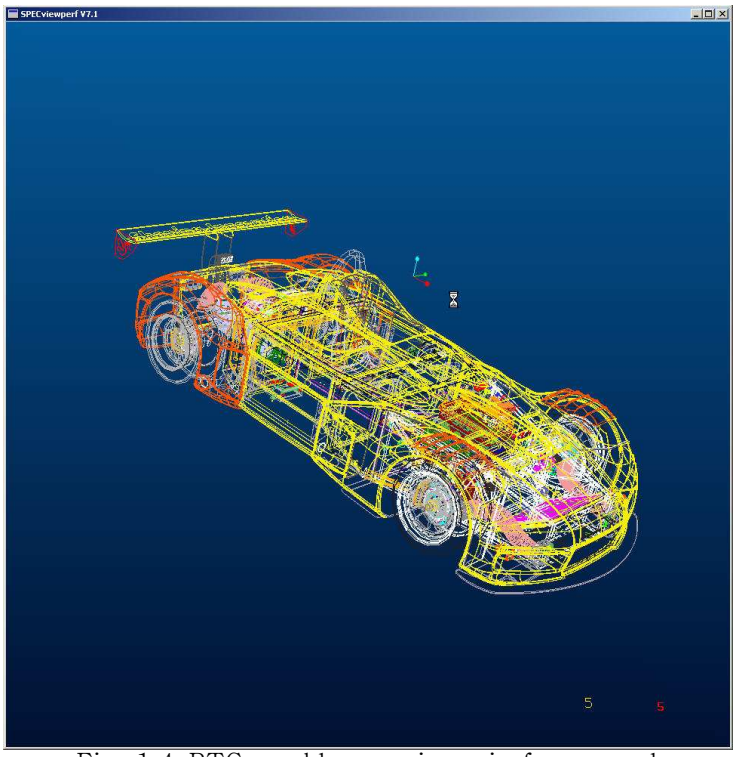

Fig. 1.4 PTC world car using wireframe mode

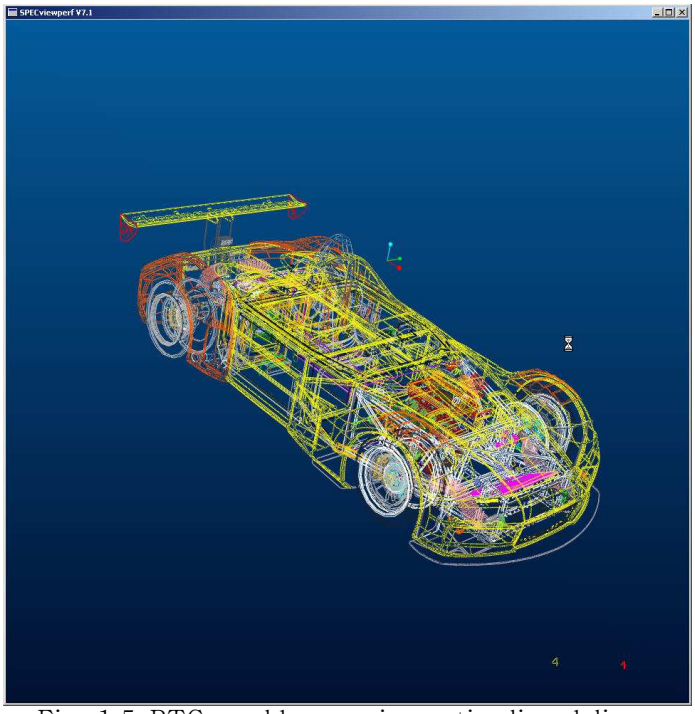

Fig. 1.5 PTC world car using anti-aliased lines

Fig. 1.6 PTC copier shaded

Fig. 1.7 PTC copier using hidden line removal mode

2.4 테스팅 플랫폼

본 연구에서 이상의 평가를 위해 intel E7205 칩셋에 기반한 워크스테이션 마더보드를 기준으로 Pentium 4 2.4GHz, 512MB SDR, 18GB SCSI 하드 드라이브를 갖춘 시스템을 구성하였으며, OpenGL 그래픽 서브-시스템으로 nVidia의 Quadro4 칩셋을 기반으로 한 그래픽 카드를 장착하였다. 테스트를 위한 운영 체제로서 Microsoft Windows 2000 Professional Service Pack 2를 선택하였고, 그래픽 서브-시스템의 드라이버는 Microsoft WHQL 기준 nVidia Quadro 그래픽 드라이버 45.23를 선정하였다.

테스트를 위해 구성된 시스템은 그래픽 드라이버가 제공하는 OpenGL 3D API의 최적 설정 조건을 파악하기 위해, SPECViewperf 외 다른 어플리케이션은 설치하지 않았으며 Microsoft의 특성상 잦은 업데이트도 필수 항목에 대해서만 설치를 수행하였다.

3. OpenGL 3D 성능 SPECViewperf 평가

3.1 OpenGL 3D 성능 테스트

그래픽 서브 시스템의 3D API로 제공되는 OpenGL 설정값의 제어에 따른 성능 변화를 테스트하였다. 사용자는 OPenGL Settings, Default Color Depth For Texture, Buffer-Flipping Mode, Vertical Sync의 4가지 부문의 선택항목을 가진다. 이 4개의 항목은 각각의 경우에 따라 아래의 표와 같은 하위옵션들을 가진다.

Table 2 OpneGL Settings의 하위옵션

OpenGL Settings	Disable support for enhanced CPU instruction set
	use unified back/depth buffer
	Enable quadbuffered stereo API
	Enable Overlays
	Maximize texture memory
	Enable conformant OpenGL texture clamp behavior

Table 3 Default Color Depth for textures의 하위옵션

Default Color Depth for textures	Use desktop color depth
	Always use 16 bpp
	Always use 32 bpp

Table 4 Buffer - Flipping Mode의 하위옵션

Buffer - Flipping Mode	Use block transfer
	Auto - select

Table 5 Vertical Sync의 하위옵션

Vertical Sync	Always off,
	Off by default
	On by default

3.1.1 테스트계획

OpenGL Settings, Default Color Depth [for textures], Buffer - Flipping Mode, Vertical Sync를 한꺼번에 테스트하기에는 각 항목의 하위옵션에 따른 경우의 수가 기하급수적으로 늘어나게 된다. 따라서 각각의 옵션에 대해 기준을 제시하여 테스트를 효율적으로 수행할 수 있도록 테스트 계획안이 필요하게 되었다.

우선적으로 각 조건에 따라 OpenGL 벤치마크 테스트 속도를 비교하여, 가장 큰 영향을 줄 수 있는 OpenGL Settings의 세부항목에 따른 프레임 처리속도 변화를 우선적으로 측정했다. OpenGL Settings는 복수선택이 가능한 옵션이므로 경우의 수가 수십 가지가 넘게 된다. 그러므로 먼저 OpenGL Settings의 옵션을 아무것도 선택하지 않은 경우를 먼저 테스트 하고 다음으로 각 하나씩의 옵션을 선택한 경우, 두개의 옵션을 동시에 선택한 경우, 세 개의 옵션을 선택한 경우 등의 순으로 테스트 하였다.

반면에 Default Color Depth [for textures], Buffer - Flipping Mode, Vertical Sync의 하위 항목들은 무조건 한 가지씩 선택되어져 있어야 하므로 OpenGL Settings의 옵션에 따른 값을 측정할 때에 각각의 값에 대한 하위옵션을 임의의 한 가지로 설정하여 고정하였다. Default Color Depth [for textures]의 하위항목은 Use desktop color depth, Buffer - Flipping Mode 하위항목은 Use block transfer, Vertical Sync의 하위항목은 Always off를 기준으로 고정 후 측정하였다.

다음은 OpenGL Settings를 제외한 다른 3개의 상위옵션의 세부항목에 따른 프레임 처리속도를 테스트해 보았다. 이를위해 OpenGL Settings를 임으로 설정하고 3개의 상위옵션의 세부항목을 바꾸어가며 테스트를 시행하였다.

위와 같은 테스트를 수행한 후 나온 결과를 분석하여 3D API 기반 OpenGL 그래픽 서브시스템에 최적화된 성능이 테스트의 결과와 일치하는지 알아보았다. 그리고 같은 방법으로 3D API 기반 OpenGL 그래픽 서브시스템에 부하가 가장 크게 걸리는 세부항목에 대해서도 테스트 해보았다. 이를 정리한다면 다음과 같다.

1. 4개의 상위옵션(OpenGL Settings, Default Color Depth [for textures], Buffer - Flipping Mode, Vertical Sync) 중에서 OpenGL Settings를 우선적으로 테스트
2. OpenGL Settings를 제외한 다른 3개의 상위옵션은 임의의 값으로 고정

3. OpenGL Settings의 옵션을 선택하지 않은경우, 하나씩을 선택한 경우, 두개를 동시에 선택한 경우... 등으로 조건을 바꾸어 가며 테스트
4. OpenGL Settings의 임의의 값에 다른 3개의 상위옵션을 바꾸어가며 테스트
5. 1~4의 결과를 토대로 최적화된 성능의 옵션값을 대입하여 테스트
6. 1~4의 결과를 토대로 부하가 가장 크게 걸리는 옵션값을 대입하여 테스트
7. 테스트 결과 분석

테스트에 체크된 세부항목은 이해를 돕기 위해 각 테스트에 번호를 부여하고 테스트 번호에 따른 세부 체크항목은 Table 로 정리해 나타내었다.
각 테스트 항목에 따른 OpenGL Settings의 옵션 체크항목은 Table 6 과 같다.

Table 6 각 테스트별 OpenGL Settings 선택현황

	OpenGL Settings						
	Disable support for enhanced CPU instruction set	use unified back/depth buffer	Enable quad buffered stereo API	Enable Overlays	Maximize texture memory	Enable conformant OpenGL texture clamp behavior	옵션선택개수
Test 1	×	×	×	×	×	×	선택하지 않음
Test 2	○	×	×	×	×	×	한가지씩의 옵션만 선택
Test 3	×	○	×	×	×	×	
Test 4	×	×	○	×	×	×	
Test 5	×	×	×	○	×	×	
Test 6	×	×	×	×	○	×	
Test 7	×	×	×	×	×	○	
Test 8	○	○	×	×	×	×	두개의 옵션을 동시에 선택
Test 9	○	×	○	×	×	×	
Test 10	○	×	×	○	×	×	
Test 11	○	×	×	×	○	×	
Test 12	○	×	×	×	×	○	
Test 13	×	○	○	×	×	×	
Test 14	×	○	×	○	×	×	
Test 15	×	○	×	×	○	×	
Test 16	×	○	×	×	×	○	
Test 17	×	×	○	○	×	×	
Test 18	×	×	○	×	○	×	
Test 19	×	×	○	×	×	○	
Test 20	×	×	×	○	○	×	
Test 21	×	×	×	○	×	○	
Test 22	×	×	×	×	○	○	

	OpenGL Settings						
	Disable support for enhanced CPU instruction set	use unified back/depth buffer	Enable quad buffered stereo API	Enable Overlays	Maximize texture memory	Enable conformant OpenGL texture clamp behavior	옵션선택개수
Test 23	○	○	○	×	×	×	세 개의 옵션을 동시에 선택
Test 24	○	○	×	○	×	×	
Test 25	○	○	×	×	○	×	
Test 26	○	○	×	×	×	○	
Test 27	○	×	○	○	×	×	
Test 28	○	×	○	×	○	×	
Test 29	○	×	○	×	×	○	세 개의 옵션을 동시에 선택
Test 30	○	×	×	○	○	×	
Test 31	○	×	×	○	×	○	
Test 32	○	×	×	×	○	○	
Test 33	×	○	○	○	×	×	
Test 34	×	○	×	○	○	×	
Test 35	×	○	○	×	×	○	
Test 36	×	○	×	○	○	×	
Test 37	×	○	×	○	×	○	
Test 38	×	○	×	×	○	○	
Test 39	×	×	○	○	○	×	
Test 40	×	×	○	○	×	○	
Test 41	×	×	○	×	○	○	
Test 42	×	×	×	○	○	○	
Test 43	○	○	○	○	×	×	네 개의 옵션을 동시에 선택
Test 44	○	○	○	×	○	×	
Test 45	○	○	○	×	×	○	
Test 46	○	○	×	○	○	×	
Test 47	○	○	×	○	×	○	
Test 48	○	○	×	×	○	○	
Test 49	○	×	○	○	○	×	
Test 50	○	×	○	○	×	○	
Test 51	○	×	○	×	○	○	
Test 52	○	×	×	○	○	○	
Test 53	×	○	○	○	○	×	
Test 54	×	○	○	○	×	○	
Test 55	×	○	○	×	○	○	
Test 56	×	○	×	○	○	○	
Test 57	×	×	○	○	○	○	
Test 58	○	○	○	○	○	×	다섯 개의 옵션을 동시에 선택
Test 59	○	○	○	○	×	○	
Test 60	○	○	○	×	○	○	
Test 61	○	○	×	○	○	○	
Test 62	○	×	○	○	○	○	
Test 63	×	○	○	○	○	○	
Test 64	○	○	○	○	○	○	모두선택

Default Color Depth [for textures] - Use desktop color depth
Buffer Flipping Mode - Use block transfer
Vertical Sync - Always off

3.1.2 OpenGL Settings 테스트

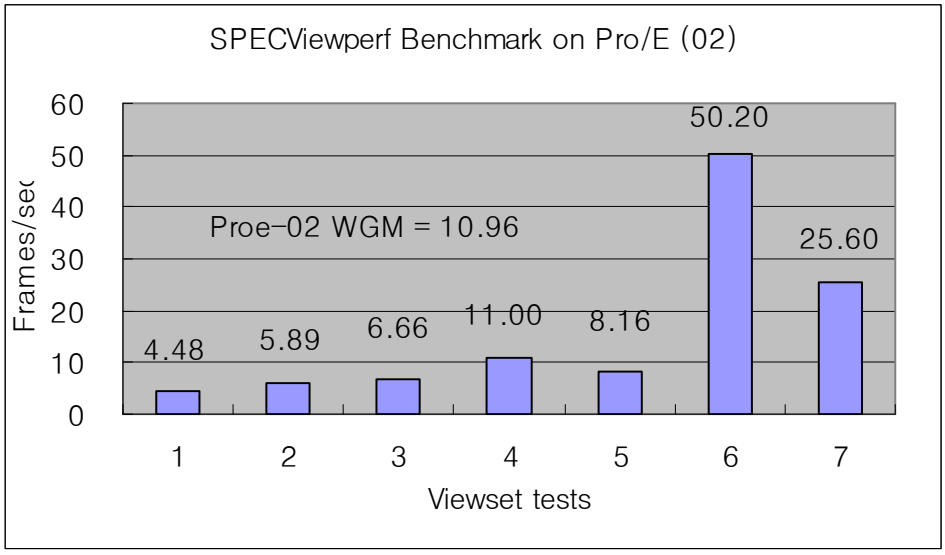

Fig. 3.1 Test 1 기본 OpenGL 설정 모드 (옵션 사항 선택 없음)

Fig. 3.1을 보면 한번의 테스트에서 7가지 viewset에 대해 각각의 값을 그래프로 나타내 주고 있음을 알 수 있다. 가로축은 각각의 viewset을 나타내며 세로축은 Frames/sec 로 각각의 viewset에 대한 처리량을 나타낸 것으로 숫자가 높을수록 단위시간당 프레임 처리량이 많았음을 의미한다.

가로축 viewset tests의 1번에서 5번까지가 4백에서 6백만개의 정점으로 이루어진 PTC World car 모델의 viewset이며 6번, 7번은 50만에서 150만개의 정점을 가진 copier 모델의 viewset이다. Fig. 3.1 그래프안의 Proe-02 WGM이란 Proe-02 Weighted Geometric Mean을 뜻하는 것으로 전체 테스트의 기하학적 평균을 나타내준다. 이 값은 SPEC사의 SPECviewperf 개발진들이 전체 viewset에 대한 평균값을 알아보기 쉽게 나타낸 것으로 SPECviewperf 프로그램 내부의 자체함수를 사용하여 평균값을 구하고 있다.

Fig. 3.1에서 보는 바와 같이 OpneGL Settings 테스트에서 아무런 옵션을 지정하지 않았을 때 Geometric Mean은 10.96이 나왔음을 알 수 있다. 이 값은 본 OpneGL기반 3D API 그래픽 서브시스템에서 OpenGL Settings의 옵션이 선택되지 않았을 때 평균적으로 초당 10.96개의 프레임을 구현할 수 있음을 나타낸다. 다음으로는 옵션을 한가지씩만 선택했을 때의 경우를 알아보았다.

이후로는 각각의 테스트결과를 알아보기 쉽게 그래프가 아닌 Table로 나타내었다.

한 개씩의 옵션만을 선택한 OpenGL Settings 세부항목에 대한 테스트 결과는 Table 7 에 나타내었다.

Table 7. 한 개의 옵션만을 선택한 OpenGL Settings 테스트 결과

	Viewset							WGM
	1	2	3	4	5	6	7	
Test 2	4.95	6.47	7.32	11.00	8.19	46.70	25.60	11.29
Test 3	4.66	5.22	7.17	10.90	8.24	50.50	25.20	11.20
Test 4	4.84	6.36	7.25	10.90	8.22	50.50	25.30	11.31
Test 5	4.82	6.37	7.26	10.80	8.21	50.50	25.30	11.29
Test 6	4.65	5.97	6.90	10.90	8.12	48.50	24.70	10.96
Test 7	5.20	6.63	7.54	11.00	8.15	50.10	25.60	11.55

Test 2~Test 7에 나타난 값들을 분석해보면 Test 1에서와 같이 아무런 옵션이 주어지지 않았을 때보다 OpenGL Settings의 하위옵션을 하나씩 선택했을 때에 프레임 처리속도가 향상된 것을 알 수 있다. 그러나, Test 6 에서와 같이 Maximize texture memory를 선택했을 때는 프레임 처리속도에 거의 변화가 없음을 볼 수있다. 대신 Test 7의 Enable conformant OpenGL texture clamp behavior를 선택했을 때에 OpenGL Settings의 옵션을 하나씩만 선택한 경우에 있어서 프레임 처리성능의 향상이 가장 크게 이루어 진 것을 볼 수 있다. Table 8은 OpenGL Settings의 하위옵션을 두개씩 동시에 선택했을 때의 경우이다.

Table 8 OpenGL Settings의 하위옵션을 두개씩 동시에 선택

	Viewset							WGM
	1	2	3	4	5	6	7	
Test 8	5.39	6.90	8.00	10.90	8.28	47.10	25.30	11.66
Test 9	5.28	6.80	7.00	10.80	8.26	47.10	25.30	11.56
Test 10	5.33	6.81	7.87	10.90	8.27	47.20	25.30	11.60
Test 11	4.79	6.16	7.13	11.00	8.17	46.30	24.70	11.06
Test 12	5.60	6.95	7.96	11.00	8.10	46.70	25.60	11.68
Test 13	5.35	6.87	7.97	10.90	8.22	50.50	25.30	11.63
Test 14	5.34	6.87	7.90	10.90	8.26	50.60	25.30	11.63
Test 15	4.67	6.00	7.01	10.80	8.21	48.90	24.40	10.99
Test 16	5.37	6.86	7.91	10.80	8.23	50.60	25.30	11.71
Test 17	5.32	6.55	7.90	10.90	8.24	50.50	25.20	11.63
Test 18	4.69	6.01	7.06	10.90	8.25	48.90	24.40	11.04
Test 19	5.33	6.86	7.90	10.80	8.24	50.50	25.30	11.70
Test 20	4.69	6.02	7.06	10.80	8.22	48.80	24.40	11.01
Test 21	5.33	6.87	7.97	10.90	8.24	50.50	25.30	11.73
Test 22	4.79	6.13	7.12	11.00	8.14	48.60	24.70	11.11

Table 8에서 두개씩의 옵션을 동시에 선택한 모든 경우의 수를 분석해보면 OpenGL Settings의 하위옵션이 아무것도 주어지지 않았을 때보다 그래픽 서브시스템 프레임 처리성능의 향상이 크게 이루어 진 것을 볼 수 있고, OpenGL Settings의 하위옵션이 한 개씩만 주어졌을 경우 보다도 더욱 큰 프레임 처리성능의 향상이 있었음을 알 수 있다.

또한 Test 11, 15, 18, 20. 22에서 보는바와 같이 Maximize texture memory가 포함된 항목은 앞에서 살펴본 것과 유사하게 프레임 처리속도가 느려졌으며 대신 Test 12, 16, 19, 21, 22에서와 같이 Enable conformant OpenGL texture clamp behavior가 포함된 항목은 프레임 처리속도가 빨라졌음을 볼 수 있다. 여기서 특히 Test 22의 Maximize texture memory와 Enable conformant OpenGL texture clamp behavior가 만난 옵션에서는 프레임 처리속도가 저하된 것으로 보아 프레임 처리성능을 향상시키는 옵션과 프레임 처리성능을 저하시키는 옵션이 만났을 경우에 성능이 저하되는 방향으로 영향을 많이 받음을 알 수 있다.

다음의 Table 9는 OpenGL Settings의 하위옵션 3개를 동시에 선택했을 경우의 테스트 결과표이다.

Table 9 OpenGL Settings의 하위옵션 3개를 동시에 선택

	Viewset							WGM
	1	2	3	4	5	6	7	
Test 23	5.24	6.73	7.65	10.90	8.24	47.10	25.20	11.49
Test 24	5.30	6.82	7.60	10.90	8.24	47.10	25.30	11.53
Test 25	4.62	5.98	6.67	10.90	8.26	46.60	24.40	10.90
Test 26	5.16	6.67	7.56	10.80	8.26	47.20	25.20	11.43
Test 27	4.65	6.15	5.89	10.90	8.25	47.10	25.20	11.02
Test 28	4.43	5.46	6.47	10.80	8.25	46.60	24.40	10.60
Test 29	4.62	6.22	7.13	10.90	8.27	47.20	25.30	11.09
Test 30	4.55	5.86	6.74	11.00	8.27	46.70	24.40	10.84
Test 31	5.15	6.62	7.53	10.90	8.27	47.20	25.30	11.43
Test 32	4.70	6.07	8.95	11.00	8.90	46.30	24.70	10.97
Test 33	5.13	6.54	7.54	10.80	8.22	50.50	25.20	11.48
Test 34	4.45	5.69	6.50	10.80	8.18	48.70	24.40	10.72
Test 35	4.60	6.10	7.08	10.80	8.21	50.40	25.20	11.10
Test 36	4.52	5.60	6.50	10.80	8.21	46.80	24.40	10.73
Test 37	4.66	6.24	7.10	10.80	8.24	50.5	25.30	11.18
Test 38	4.45	5.64	6.50	10.80	8.24	48.80	24.40	10.73
Test 39	4.45	5.65	6.49	10.09	8.25	46.80	24.40	10.74
Test 40	5.12	6.56	7.50	10.90	8.26	50.50	25.20	11.50
Test 41	4.80	5.91	6.86	10.90	8.26	48.70	24.40	10.94
Test 42	4.60	5.90	6.86	10.80	8.24	48.90	24.40	10.92

Table 9를 분석해 보면 Table 7이나 Table 8과 비교해 보았을 때 상반되는 결과가 나타나지 않으므로 Table 7이나 Table 8의 분석과 비슷한 분석결과를 보인다. OpenGL Settings의 하위옵션이 3개가 동시에 선택되었을 경우 OpenGL Settings의 하위옵션이 한 개나 두 개가 동시 선택되었을 경우에 비해 전체적으로 프레임 처리속도가 조금씩 느려졌다. Table 9의 결과는 앞선 테스트 결과표인 Table 7이나 Table 8과 마찬가지로 Maximize texture memory가 포함된 항목은 프레임 처리속도가 느려졌으며 Enable conformant OpenGL texture clamp behavior가 포함된 항목은 프레임 처리속도가 빨라졌음을 볼 수 있다.

다음 Table 10은 계속해서 OpenGL Settings의 하위옵션 네 개를 동시에 선택했을 경우의 결과표이다.

Table 10 OpenGL Settings의 하위옵션 네 개를 동시에 선택

	Viewset							WGM
	1	2	3	4	5	6	7	
Test 43	4.68	6.14	6.86	11.00	8.30	47.10	25.20	11.04
Test 44	4.38	5.36	6.32	10.90	8.28	46.70	24.40	10.55
Test 45	4.51	5.95	6.71	10.90	8.26	47.10	25.20	10.88
Test 46	4.33	5.33	6.26	10.80	8.26	46.60	24.40	10.49
Test 47	5.16	6.53	7.42	10.90	8.26	47.20	25.30	11.39
Test 48	4.60	5.92	6.35	10.90	8.26	46.70	24.40	10.88
Test 49	4.59	5.89	6.82	10.90	8.25	46.70	24.40	10.86
Test 50	5.10	6.55	7.43	10.90	8.26	47.20	25.20	11.37
Test 51	4.65	5.94	6.85	10.90	8.26	46.60	24.40	10.89
Test 52	4.56	5.83	6.83	10.80	8.27	46.70	24.40	10.82
Test 53	4.58	5.68	6.63	10.90	8.24	48.80	24.40	10.92
Test 54	5.06	6.50	7.41	10.80	8.21	50.50	25.20	11.42
Test 55	4.56	5.87	6.84	10.90	8.25	48.80	24.40	10.91
Test 56	4.41	5.37	6.37	10.90	8.26	48.80	24.40	10.63
Test 57	4.48	5.64	6.52	10.90	8.21	48.90	24.40	10.75

Table 10의 OpenGL Settings의 하위옵션 네 개를 동시에 선택했을 때를 분석해 보았을 때 가장 눈여겨 보아야 할 대목은 OpenGL Settings의 모든 하위옵션을 조합한 경우의 수를 통틀어 가장 느린 프레임 처리속도의 수치가 나타났다는 것이다. 다음 테스트를 수행하며 Table 11과 Table 12에서도 나타나겠지만 미리 테스트 결과를 살펴보면 OpenGL Settings의 하위옵션을 조합한 테스트 전체에서 Test 46의 Disable support for enhanced CPU instruction set, use unified back/depth buffer, Enable Overlays, Maximize texture memory라는 OpenGL Settings의 하위옵션 네

개를 동시에 선택 했을 때 가장 느린 프레임 처리속도인 10.49의 WGM이 나왔다. 이것은 OpenGL 3D API 기반 그래픽 서브시스템의 성능에 부하가 가장 크게 걸리는 OpenGL Settings의 하위옵션이라는 것을 나타내는 것이다. 따라서 계속되는 성능결과 분석에서 Test 21의 그래픽 서브시스템의 최적화된 환경과 더불어서 이번의 Test 46의 결과를 가지고 성능이 가장 저하되는 시스템 환경도 알아보고자 한다.

계속해서 Table 11은 다섯 개의 OpenGL Settings 하위옵션을 동시에 선택했을 경우이다.

Table 11 다섯 개의 OpenGL Settings 하위옵션을 동시에 선택

	Viewset							WGM
	1	2	3	4	5	6	7	
Test 58	4.70	5.64	6.50	10.90	8.26	46.70	24.40	10.69
Test 59	4.64	6.23	7.00	10.90	8.27	47.20	25.20	11.06
Test 60	4.43	5.55	6.44	10.90	8.28	46.80	24.40	10.64
Test 61	4.46	5.52	6.49	10.90	8.29	46.70	24.40	10.68
Test 62	4.45	5.61	6.49	10.90	8.29	46.60	24.40	10.68
Test 63	4.43	5.53	6.47	10.90	8.25	48.80	24.40	10.70

Table 11에서 OpenGL Settings의 5개 하위옵션을 동시에 선택했을 때는 특별히 두드러지는 결과는 나타나지 않고 있다. 다만, 프레임 처리속도가 많이 저하된 것을 알 수 있다.

다음의 Table 12는 마지막으로 6개의 옵션을 모두 선택했을 경우이다.

Table 12 여섯 개의 OpenGL Settings 하위옵션을 모두 선택

	Viewset Tests							WGM
	1	2	3	4	5	6	7	
Test 64	4.43	5.53	6.47	10.90	8.24	46.60	24.40	10.63

Table 12에서도 마찬가지로 특별하게 두드러지는 결과는 없으며, 전체적인 프레임 처리속도의 저하가 나타나고 있다.

3.1.3 Default Color Depth [for texture], Buffer-Flipping Mode, Vertical Sync 테스트

이번에는 OpenGL Settings테스트결과를 토대로 Default Color Depth [for texture], Buffer - Flipping Mode, Vertical Sync 의 OpenGL Settings를 제외한 다른 3가지 옵션 값에 따른 프레임 처리속도의 변화를 알아보았다. 이번 연구를 수행하기 위해서는 OpenGL Settings가 어떠한 경우로든 하위옵션 값이 선택되어져 있어야 하므로 3.1.2의 OpenGL Settings 테스트에서 주로 프레임 처리속도를 빠르게 해준 것으로 나타난 OpenGL Settings의 하위옵션 값 Enable conformant OpenGL texture clamp behavior가 선택되었을 경우를 대표적으로 선정하여 Default Color Depth [for texture], Buffer - Flipping Mode, Vertical Sync 의 하위옵션 값의 변화에 따른 그래픽 서브시스템의 성능을 알아본다. 이 경우에도 각 옵션에 대한 하위옵션의 변화에 따라 경우의 수가 많으므로 효율적인 테스트를 위해 일정한 기준을 두고 테스트를 시행하였다.

1. OpenGL Settings의 하위옵션을 Enable conformant OpenGL texture clamp behavior로 선택
2. Default Color Depth [for textures] 의 옵션을 Use desktop color depth로 고정한 후 Buffer - Flipping Mode, Vertical Sync 의 옵션을 변화시켰을 때의 경우를 테스트
3. Default Color Depth [for textures]의 옵션을 Always 16 bpp 로 두고 Buffer - Flipping Mode, Vertical Sync 의 옵션변화에 따라 테스트
4. 마지막으로 Default Color Depth [for textures]의 옵션을 Always 32 bpp 로 두고 Buffer - Flipping Mode, Vertical Sync 의 옵션을 바꾸어가며 테스트

Table 13은 각 테스트 번호에 따른 Default Color Depth [for texture], Buffer - Flipping Mode, Vertical Sync 의 하위옵션 값에 따른 사항을 표로 나타내었다.

Table 13 각 테스트 번호에 따른 Default Color Depth [for texture], Buffer - Flipping Mode, Vertical Sync 의 하위옵션 값에 따른 사항

	Default Color Depth for textures			Buffer - Flipping Mode		Vertical Sync		
	Use desktop color depth	Always use 16 bpp	Always use 32 bpp	Use block transfer	Auto - select	Always off	Off by default	On by default
Test 65	○	×	×	○	×	○	×	×
Test 66	○	×	×	○	×	×	○	×
Test 67	○	×	×	○	×	×	×	○
Test 68	○	×	×	×	○	○	×	×
Test 69	○	×	×	×	○	×	○	×
Test 70	○	×	×	×	○	×	×	○
Test 71	×	○	×	○	×	○	×	×
Test 72	×	○	×	○	×	×	○	×
Test 73	×	○	×	○	×	×	×	○
Test 74	×	○	×	×	○	○	×	×
Test 75	×	○	×	×	○	×	○	×
Test 76	×	○	×	×	○	×	×	○
Test 77	×	×	○	○	×	○	×	×
Test 78	×	×	○	○	×	×	○	×
Test 79	×	×	○	○	×	×	×	○
Test 80	×	×	○	×	○	○	×	×
Test 81	×	×	○	×	○	×	○	×
Test 82	×	×	○	×	○	×	×	○
Test 83	○	×	×	○	×	○	×	×
Test 84	○	×	×	○	×	×	○	×
Test 85	○	×	×	×	○	○	×	×
Test 86	○	×	×	×	○	×	○	×
Test 87	×	○	×	○	×	○	×	×
Test 88	×	○	×	○	×	×	○	×
Test 89	×	○	×	×	○	○	×	×
Test 90	×	○	×	×	○	×	○	×
Test 91	×	×	○	○	×	○	×	×
Test 92	×	×	○	○	×	×	○	×
Test 93	×	×	○	×	○	○	×	×
Test 94	×	×	○	×	○	×	○	×
Test 95	○	×	×	○	×	×	×	○
Test 96	○	×	×	×	○	×	×	○
Test 97	×	○	×	○	×	×	×	○
Test 98	×	○	×	×	○	×	×	○
Test 99	×	×	○	○	×	×	×	○
Test 100	×	×	○	×	○	×	×	○

Table 14는 우선적으로 Default Color Depth [for textures] 의 옵션을 Use desktop color depth로 두고 나머지 Buffer - Flipping Mode, Vertical Sync 의 옵션만 변화시켰을 때의 경우를 테스트 해 보았다.

Table 14 Default Color Depth [for textures] 의 하위옵션을 Use desktop color depth로 고정한 후 Buffer - Flipping Mode, Vertical Sync 의 하위옵션에 따른 테스트

	Viewset							WGM
	1	2	3	4	5	6	7	
Test 65	4.74	6.26	7.15	10.90	8.11	50.10	25.60	11.21
Test 66	4.76	6.37	7.15	10.90	8.12	50.20	25.60	11.25
Test 67	4.76	6.03	6.67	10.00	7.51	35.00	20.00	9.938
Test 68	4.90	6.40	7.30	11.00	8.15	50.20	25.60	11.37
Test 69	4.90	6.60	7.30	11.00	8.15	50.20	25.60	11.37
Test 70	4.76	6.12	6.66	10.00	7.90	35.70	20.00	9.963

* OpenGL Settings - Enable conformant OpenGL texture clamp behavior

위의 결과를 분석해 보면 전체적으로는 성능이 어느 선에서 일정하게 나왔으나 Test 67, Test 70의 Vertical Sync 의 옵션이 On by default 상태에서 프레임 처리 속도가 급격히 감소하고 있음을 볼 수 있다. Vertical Sync 의 또 다른 옵션인 Always off 와 Off by default 의 선택에 따른 변화는 Test 65와 Test 66, 그리고 Test 68과 Test 69 에서 보는 것처럼 별 다른 차이를 나타내지 않고 있어 그 선택에 따른 결과는 무시할 수 있을 정도이다. 그러나 Vertical Sync 의 옵션이 Off by default 일 때에 Always Off를 선택한 경우보다 조금 더 나은 프레임 처리속도를 보여주고 있다. Buffer - Flipping Mode는 Test 68~Test 70 에서와 같이 하위옵션이 Auto - select 일 경우가 하위옵션이 Use block transfer를 선택한 경우보다 프레임 처리속도가 더 빠르게 나타났다. 위에서 나타난 Test 65~Test 70까지의 결과만을 놓고 본다면 Vertical Sync 의 옵션은 Always off 와 Off by default인 경우 그리고, Buffer - Flipping Mode는 Auto - select 인 경우에 그래픽 서브시스템의 성능을 향상시켜주는데 도움이 되는 하위옵션일 것이라 예상할 수 있다.

다음의 Table 15는 Default Color Depth [for texture] 의 옵션을 Always use 16 bpp 로 고정하고 다른 옵션들에 변화를 주었을 때의 경우이다.

Table 15 Default Color Depth [for textures] 의 하위옵션을 Always use 16 bpp로 고정한 후 Buffer - Flipping Mode, Vertical Sync 의 하위옵션에 따른 테스트

	Viewset							WGM
	1	2	3	4	5	6	7	
Test 71	4.88	6.40	7.23	10.90	8.13	50.10	25.60	11.32
Test 72	4.77	6.39	7.60	11.00	8.15	50.20	25.60	11.28
Test 73	4.99	6.49	7.49	10.00	7.49	35.60	20.20	10.26
Test 74	5.23	6.67	7.56	11.00	8.14	50.20	25.60	11.57
Test 75	5.17	6.62	7.75	11.00	8.13	50.10	25.60	11.54
Test 76	4.99	6.20	7.22	9.99	7.51	35.70	20.00	10.21

* OpenGL Settings - Enable conformant OpenGL texture clamp behavior

Test 71~Test 76의 Default Color Depth [for textures] 의 하위옵션을 Always use 16 bpp로 고정한 후 Buffer - Flipping Mode, Vertical Sync 의 하위옵션의 변화에 따라 테스트를 해 본 결과는 앞선 테스트인 Test 71~Test 76의 Default Color Depth [for textures] 하위옵션을 Use desktop color depth로 고정한 후 Buffer - Flipping Mode, Vertical Sync 의 하위옵션에 따른 테스트와 비슷한 양상의 특징을 보이고 있으며, 다만 전체적인 프레임 처리속도 면에서 더 향상된 성능을 보여주었다.

좀 더 자세하게 살펴보면 Test 73, Test 76에서 보는바와 같이 Default Color Depth [for texture] 의 옵션이 Always use 16 bpp 인 경우도 Default Color Depth [for texture] 의 옵션이 Use desktop color depth 인 경우와 마찬가지로 Vertical Sync 의 옵션이 On by default 상태에서 프레임 처리속도가 급격히 감소하고 있다.

Vertical Sync 의 다른 옵션인 Always off 와 Off by default의 선택에 따른 변화도 Test 71, Test 72, Test 74, Test 75에서 보는바와 같이 마찬가지로 무시할 수 있을 정도이다. 여기서도 앞에서의 Test 65와 Test 66, 그리고 Test 68과 Test 69와 마찬가지로 Vertical Sync 의 옵션이 Off by default를 선택한 경우가 Always Off를 선택한 경우보다 조금 더 나은 프레임 처리속도를 보여주고 있다. Buffer - Flipping Mode는 Auto - select 일 경우가 Use block transfer 인 경우보다 속도가 더 빠르게 나타났다.

앞서 말한바와 같이 전반적으로 Default Color Depth [for texture] 의 옵션이 Always use 16 bpp 인 경우가 Use desktop color depth로 고정했을 때 보다 프레임 처리성능이 향상된 것으로 나타났다.

다음의 Table 16은 Default Color Depth [for texture] 의 옵션을 Always use 32 bpp 로 했을 때이다.

Table 16 Default Color Depth [for textures] 의 하위옵션을 Always use 32 bpp로 고정한 후 Buffer - Flipping Mode, Vertical Sync 의 하위옵션에 따른 테스트

	Viewset							WGM
	1	2	3	4	5	6	7	
Test 77	5.20	6.65	7.49	11.00	8.15	50.20	25.60	11.55
Test 78	5.18	6.60	7.49	10.90	8.14	50.20	25.60	11.51
Test 79	5.01	6.32	7.07	10.00	7.51	35.60	20.00	10.16
Test 80	5.14	6.62	7.50	10.90	7.73	50.20	25.60	11.50
Test 81	5.16	6.59	7.50	10.90	8.13	50.20	25.60	11.50
Test 82	5.00	6.38	7.20	10.00	7.49	35.40	20.00	10.18

* OpenGL Settings - Enable conformant OpenGL texture clamp behavior

Test 77~Test 82의 Default Color Depth [for textures] 의 하위옵션을 Always use 32 bpp로 고정하고 Buffer - Flipping Mode, Vertical Sync 의 하위옵션의 변화에 따라 테스트를 해 본 결과 역시 앞선 테스트인 Test 65~Test 76의 Default Color Depth [for textures] 하위옵션이 Use desktop color depth 일 때나 Default Color Depth [for textures] 하위옵션이 Always use 16 bpp 일 때에 따른 테스트와 큰 차이점은 보이지 않고 있다.

Default Color Depth [for texture] 의 옵션이 Always use 32 bpp 일 때는 Default Color Depth [for texture] 의 옵션이 Use desktop color depth 일 때는 물론이고 Always use 16 bpp 일 때보다도 더 전체적으로 프레임 처리성능이 고르게 향상된 것으로 나왔다. 여기에서도 Test 79, Test 82에서 보는바와 같이 Vertical Sync의 옵션이 On by default 가 선택되었을 때는 프레임 처리속도가 급격하게 떨어진 것으로 나타났다. 나머지 Vertical Sync의 하위옵션이 Always off 또는 Off by default의 경우에서나 Buffer - Flipping Mode의 하위옵션이 Use block transfer 나 Auto-select의 옵션이든 간에 가리지 않고 엇비슷하게 향상된 프레임 처리성능을 나타내 주었으며 Vertical Sync의 하위옵션이 Off by default일 때, 그리고 Buffer - Flipping Mode의 하위옵션이 Auto-select일 때에 더 나은 프레임 처리결과를 나타내주었다.

3.2 OpenGL 그래픽서브시스템 최적화 성능테스트

이 절에서는 이번 연구에 사용된 그래픽 서브시스템에서 최적화된 가장 빠른 프레임 처리성능을 보여줄 수 있는 조건을 찾기 위한 테스트를 실험해 보았다. 이번 테스트에서는 OpenGL Settings의 하위옵션 값을 앞에서 이미 테스트해본 3.1.2의 OpenGL Settings 테스트에서 프레임 처리속도가 가장 빠르게 나온 Test 21의 OpenGL Settings의 하위옵션 값 Enable Overlays와 Enable conformant OpenGL texture clamp behavior가 동시에 선택된 경우의 옵션 값을 적용하여 테스트 해 보았다.

앞에서 미리 행한 테스트 결과를 분석해 보면 각 옵션에 대한 하위옵션들이 그래픽 서브시스템에 미치는 전체적인 윤곽은 어느 정도 드러났으므로 3.1.2의 OpenGL Settings와 3.1.3 Default Color Depth [for texture], Buffer-Flipping Mode, Vertical Sync 테스트의 결과를 가지고 분석해 본다면 이번 그래픽 서브시스템의 최적화 시스템의 테스트 결과도 전망해 볼 수 있을 것이다. 앞선 3.1.3의 테스트 분석결과에서는 Default Color Depth [for textures] 의 하위옵션은 Always use 32 bpp로 설정했을 때, Buffer - Flipping Mode의 하위옵션은 Auto-select가 설정되었을 경우, 그리고 Vertical Sync의 하위옵션이 Off by default가 설정되었을 경우가 본 그래픽 서브시스템 성능 테스트에서 최적화된 결과를 나타내어 줄 것이라고 조심스럽게 예상해 볼 수 있다.

다음의 Table 17은 OpenGL Settings의 하위옵션이 Enable Overlays와 Enable conformant OpenGL texture clamp behavior가 동시에 선택된 경우에 있어 Default Color Depth [for texture], Buffer-Flipping Mode, Vertical Sync의 하위옵션들을 변화시켜가며 그래픽 서브시스템 성능의 최적화환경을 테스트한 결과표이며, 여기서는 성능이 가장 향상된 옵션 값을 찾기 위한 것이므로 프레임 처리속도를 급격하게 저하시키는 옵션인 Vertical Sync의 하위옵션 On by default 는 제외하고 테스트 하였다.

Table 17 OpenGL 그래픽 서브시스템 최적화 성능 테스트

	Viewset							WGM
	1	2	3	4	5	6	7	
Test 83	5.08	6.55	7.45	10.80	8.24	50.50	25.30	11.46
Test 84	5.04	6.53	7.43	10.90	8.25	50.50	25.30	11.45
Test 85	4.68	6.19	6.93	11.00	8.29	54.50	26.30	11.35
Test 86	4.57	6.08	6.80	10.80	8.21	54.40	26.30	11.20
Test 87	4.49	5.64	6.56	11.00	8.30	50.60	25.30	10.91
Test 88	4.74	6.30	7.13	10.90	8.25	50.30	25.30	11.24
Test 89	4.76	6.35	7.80	10.90	8.25	54.40	26.30	11.45
Test 90	4.71	6.32	7.16	10.80	8.22	54.40	26.30	11.39
Test 91	4.78	6.47	7.30	10.90	8.25	50.60	25.30	11.34
Test 92	5.26	6.73	7.75	10.90	8.25	50.50	25.20	11.63
Test 93	5.24	6.87	7.89	10.80	8.20	54.30	26.30	11.84
Test 94	5.34	6.35	7.86	10.80	8.21	54.30	26.30	11.86

* OpenGL Settings - Enable Overlays,
Enable conformant OpenGL texture clamp behavior

Table 17에서 가장 빠른 프레임 처리속도는 Test 94에서 Default Color Depth [for texture] 의 옵션을 Always use 32 bpp 로 했을 때 Buffer - Flipping Mode 의 옵션이 Auto - select, Vertical Sync가 Off by default 인 경우로 나타났다. 이 결과는 테스트를 시행하기 전에 예상한 결과와도 일치하는 하위옵션 값들을 보여준다. OpenGL Settings의 변화에 따른 결과 값도 큰 폭의 성능변화를 보여주었다.

Test 94와 Default Color Depth [for texture], Buffer-Flipping Mode, Vertical Sync 의 하위옵션 값이 동일하면서 OpenGL Settings의 하위옵션만이 다른 Test 81의 WGM이 11.50이라는 프레임 처리속도를 보여준 것에 비해, Test 94에서는 Table 17에서 보는바와 같이 WGM이 11.86이라는 다소 여유 있는 차이의 프레임 처리속도를 보이고 있다. 이는 OpenGL Settings의 하위 옵션 값에 따른 그래픽 서브시스템의 프레임 처리 성능의 변화 역시 크다는 것을 나타내준다. 또한 Vertical Sync가 Off by default인 Test 94의 모드는 Test 93의 Vertical Sync가 Always Off 인 경우보다 프레임 처리속도 면에서 근소하게 앞선 값을 가졌다.

따라서 본 논문의 전체 테스트에서 OpenGL Settings의 하위옵션이 Enable Overlays, Enable conformant OpenGL texture clamp behavior인 경우에서 Default Color Depth [for texture] 의 하위옵션이 Always use 32 bpp, Buffer - Flipping Mode의 하위옵션이 Auto - select, Vertical Sync의 하위옵션이 Off by default인 경우가 가장 빠른 프레임 처리성능의 향상을 가져왔다.

3.3 OpenGL 그래픽서브시스템 성능저하 환경테스트

다음은 시스템의 프레임 처리움직임을 가장 둔화시키는 옵션을 알아보았다. 성능저하 옵션 값을 테스트하기 위해 OpenGL Settings의 하위옵션 값을 3.1.2의 OpenGL Settings 테스트에서 프레임 처리성능을 가장 저하시킨 Test 46의 OpenGL Settings의 하위옵션 값 Disable support for enhanced CPU instruction set, use unified back/depth buffer, Enable Overlays, Maximize texture memory 의 4개의 하위옵션이 동시에 선택된 경우를 선택하여 테스트 하였다.

본 연구는 그래픽 서브시스템에 부하가 가장 크게 걸리는 환경설정을 알아보기 위한 것이므로 Default Color Depth [for texture], Buffer-Flipping Mode, Vertical Sync 의 하위옵션에 대한 설정은 앞선 3.1.3의 테스트 결과에서 다른 하위옵션들보다 급격하게 시스템의 성능을 둔화시킨 Vertical Sync의 하위옵션 On by default인 경우만 테스트하였다.

이번 연구의 결과 역시 3.1.2의 OpenGL Settings와 3.1.3 Default Color Depth [for texture], Buffer-Flipping Mode, Vertical Sync 테스트의 결과를 분석해 보았을 때 Default Color Depth [for textures]의 하위옵션이 Use desktop color depth인 상황에서 Buffer-Flipping Mode의 하위옵션이 Use block transfer, Vertical Sync의 옵션은 On by default인 경우가 성능에 부하가 가장 클 것이라고 어느 정도 결과를 예상할 수 있으나 결과 분석에 따른 예상과 동일한 결과를 얻을 수 있을 것인지 알아보고 또한 더욱 정확한 연구결과를 위해 테스트를 시행하였다.

Table 18 OpenGL 그래픽 서브시스템 성능저하 환경테스트

	Viewset							WGM
	1	2	3	4	5	6	7	
Test 95	4.27	5.00	6.00	10.00	7.51	30.00	20.00	9.206
Test 96	4.29	5.35	6.00	10.00	7.87	30.00	20.00	9.328
Test 97	4.29	5.46	6.00	10.00	7.52	30.00	20.00	9.324
Test 98	4.29	5.46	6.08	10.00	7.70	30.00	20.00	9.377
Test 99	4.29	5.50	6.00	10.00	7.55	30.00	20.00	9.327
Test 100	4.28	5.46	6.29	10.00	7.70	30.00	20.00	9.415

* OpenGL Settings - Disable support for enhanced CPU instruction set, use unified back/depth buffer, Enable Overlays, Maximize texture memory

OpenGL 그래픽 서브시스템의 성능저하 환경을 테스트한 Table 18의 결과를 보면 성능저하 환경 테스트를 시행하기 전에 예상했던 결과가 그대로 나타났음을 볼 수 있다. Default Color Depth [for textures]의 하위옵션이 Use desktop color depth, Buffer-Flipping Mode의 하위옵션이 Use block transfer, Vertical Sync의 하위옵션이 On by default인 Test 95 의 경우가 WGM이 9.206으로 그래픽 서브시스템의 프레임 처리성능이 가장 둔화된 옵션으로 나타났다.

3.2 OpenGL 그래픽 서브시스템의 최적화 성능테스트에서 Default Color Depth [for texture], Buffer-Flipping Mode, Vertical Sync 의 하위옵션 값이 동일하면서 OpenGL Settings의 하위옵션만이 다른 Test 81과 Test 94를 비교했던 것과 동일하게 이번 연구인 OpenGL 그래픽서브시스템의 성능저하 환경테스트 에서도 마찬가지로 Default Color Depth [for texture], Buffer-Flipping Mode, Vertical Sync 의 하위옵션 값이 동일하면서 OpenGL Settings의 하위옵션만이 다른 Test 67과 Test 95를 비교해 보면 Test 67은 WGM이 9.938라는 프레임 처리속도를 보여준 것에 비해, Test 95에서는 WGM이 9.206의 큰 폭의 차이가 나는 프레임 처리속도를 보이고 있다. 이 또한 OpenGL Settings의 하위 옵션 값에 따라서 그래픽 서브시스템의 프레임 처리 성능의 변화가 크다는 것을 재확인해주는 결과라고 할 수 있다.

Ⅳ. SPECViewperf 테스트결과 분석 및 평가

본 연구에서 앞에서의 여러 테스트를 통해 OpenGL Settings, Default Color Depth [for textures], Buffer - Flipping Mode, Vertical Sync의 하위 항목들의 세부사항에 따른 그래픽 서브시스템에 미치는 영향에 대해 알아보았다. 그러한 연구테스트의 여려 결과를 분석 및 평가 해보면 다음과 같다.

우선 OpenGL Settings의 하위옵션이 그래픽 서브시스템에 미치는 영향을 분석해 보면 Maximize texture memory를 선택했을 때는 프레임 처리속도가 극단적으로 느려지는 것을 알 수 있고, 대신 Enable conformant OpenGL texture clamp behavior를 선택했을 때 가장 프레임 처리성능의 향상이 크게 이루어 진 것을 볼 수 있다.

OpenGL Settings의 하위옵션 Maximize texture memory를 선택한 경우는 문자 그대로 Texture 즉, 질감을 표현하기 위해 그래픽 서브시스템이 사용할 수 있는 메모리를 최대한으로 사용하게 하였으므로 그와는 대조적으로 그래픽 서브시스템을 구동하는 메모리의 양의 부족하게 되어 전체적으로 그래픽 서브시스템의 프레임 처리량이 줄어든 것으로 분석할 수 있다. 이와 유사하게 OpenGL Settings의 하위옵션 Enable conformant OpenGL texture clamp behavior가 선택된 경우 프레임 처리량이 높아진 것 또한 질감의 표현을 강제로 조정하였으므로 Maximize texture memory를 선택한 경우와는 대조적으로 성능의 향상을 가져온 것으로 분석된다.

OpenGL Settings에서 한 개씩의 하위옵션을 선택한 경우를 분석해 보면 아무런 하위옵션을 선택하지 않은 경우에 보다 한가지의 경우 즉, Maximize texture memory를 제외하고는 프레임 처리성능이 향상되었음을 알 수 있다. 이는 OpenGL Settings에 있는 하위옵션이 Maximize texture memory를 제외한 다른 하위옵션들 Disable support for enhanced CPU instruction set, use unified back/depth buffer, Enable quadbuffered stereo API Enable Overlays, Enable conformant OpenGL texture clamp behavior가 전반적으로 프레임 처리속도를 향상시키는데 관련이 있기 때문이다.

OpenGL Settings에서 두개의 옵션을 동시에 선택했을 경우는 한 개의 옵션이 주어진 경우 보다 더욱 프레임 처리성능이 향상되었음을 알 수 있다. OpenGL

Settings의 전체 테스트에서 비교했을 때에도 옵션이 2개가 동시 선택되었을 경우에 전반적으로 높은 성능의 프레임 처리속도 증가가 나타났으며, 전체의 OpenGL Settings 테스트 결과중에서도 Test 21의 Enable Overays, Enable conformant OpenGL texture clamp behavior가 동시에 선택된 경우가 그래픽 서브시스템의 최적 그래픽 처리량을 보이는 OpenGL Settings 하위옵션으로 나타났다. 전체적으로 Maximize texture memory가 포함된 항목은 프레임 처리속도가 느려지고 Enable conformant OpenGL texture clamp behavior가 포함된 항목은 프레임 처리속도가 빨라졌다.

여기에서 눈여겨 볼 대목은 OpneGL Settings의 하위옵션 중 그래픽 서브시스템의 프레임 처리속도를 저하시키는 하위옵션 Maximize texture memory와 프레임 처리속도를 향상시켜주는 하위옵션 Enable conformant OpenGL texture clamp behavior가 만난 경우인 Test 22이다. 이 경우에서 그래픽 서브시스템의 프레임 처리속도가 낮게 나타난 것으로 보았을 때 OpenGL Settings의 하위옵션 중에서 프레임 처리성능을 향상시키는 옵션과 처리성능을 저하시키는 옵션이 동시에 선택된 경우에는 그래픽 서브시스템의 성능이 저하되는 방향으로 영향을 많이 받음을 알 수 있다. 이는 그래픽 서브시스템을 향상시켜주는 것으로 고려되는 OpenGL Settings의 하위옵션 값이라 할지라도 무조건으로 중복적으로 다수를 선택하는 것만이 최적의 성능향상을 나타내주지는 않음을 보여준다. 오히려 처리 성능을 높여주는 것처럼 보이는 하위옵션을 무분별하게 선택하는 것은 프레임 처리에 부하를 가져와 그래픽 서브시스템의 성능저하를 불러올 수도 있음을 알 수 있다.

OpenGL Settings의 하위옵션이 3개가 동시에 선택되었을 경우에는 OpenGL Settings의 하위옵션이 한 개나 두 개가 동시 선택되었을 경우에 비해 전체적으로 프레임 처리속도가 조금씩 느려졌다.

OpenGL Settings의 하위옵션 네 개를 동시에 선택했을 때는 OpenGL Settings의 하위옵션이 3개가 동시에 선택되었을 경우보다도 더욱 프레임 처리속도가 저하되었으며, 무엇보다 OpenGL Settings의 모든 하위옵션을 조합한 경우의 수를 통틀어 가장 느린 프레임 처리속도의 수치가 나타났다. OpenGL Settings의 하위옵션을 조합한 테스트 전체에서 Test 46의 Disable support for enhanced CPU instruction set, use unified back/depth buffer, Enable Overlays, Maximize texture memory의 OpenGL Settings의 하위옵션 네 개를 동시에 선택 했을 때 가장 느린 프레임

처리속도가 나왔다. 이것은 Test 46의 옵션을 조합한 경우가 그래픽 서브시스템의 성능에 부하가 가장 크게 걸리는 OpenGL Settings의 하위옵션이라는 것을 나타내는 것이다.

OpenGL 최적화 성능 테스트에서는 좀 더 복잡한 양상을 보이고 있다. 프레임 처리속도가 빠른 옵션은 Vertical Sync가 Always off 와 Off by default 일 때와 Buffer - Flipping Mode가 Auto - select, Default Color Depth [for texture] 의 하위옵션이 Always use 32 bpp 인 경우이다. Vertical Sync의 Always off 와 Off by default 는 선택에 따른 변화는 무시할 수 있을 정도이다. . 그러나 Vertical Sync 의 옵션이 Off by default 일 때가 Always Off를 선택한 경우보다 언제나 조금씩 더 나은 프레임 처리속도를 보여주고 있다.

Buffer - Flipping Mode는 Auto - select 일 경우가 Use block transfer 인 경우보다 빠른 것으로 나타났다.

Default Color Depth [for texture] 는 옵션이 Always use 16 bpp 인 경우가 Use desktop color depth로 고정했을 때 보다 프레임 처리성능이 향상된 것으로 나타났으며, Default Color Depth [for texture] 의 하위옵션이 Always use 32 bpp 일 때는 하위옵션이 Always use 16 bpp 일 때 보다 더욱 전체적인 프레임 처리성능이 향상되어 고르게 나타난다.

프레임 처리속도가 느려지는 옵션은 Vertical Sync의 하위옵션이 On by default 상태가 또 다른 Vertical Sync의 하위옵션인 Always Off나 Off by default 보다 프레임 처리속도가 극단적으로 감소하고 있다.

Buffer - Flipping Mode는 하위옵션이 Use block transfer가 선택된 경우가 하위옵션이 Auto - select 가 선택된 경우보다 프레임 처리성능이 더욱 둔화되었으며, Default Color Depth [for texture]는 하위옵션이 Use desktop color depth로 설정되었을 때 프레임 처리성능이 가장 저하되었다.

본 테스트 결과에서 그래픽 서브시스템에 가장 적합한 최적의 프레임 처리 하위옵션은 Default Color Depth [for texture] 의 옵션을 Always use 32 bpp 로 했을 때 Buffer - Flipping Mode의 옵션이 Auto - select, Vertical Sync가 Off by default 를 설정했을 경우 인 것으로 나왔다. Vertical Sync가 Always Off일 경우와 Off by default인 경우는 프레임 처리속도차가 그리 크지는 않으나 Off by default가 항상 조금씩 앞선 결과 값을 보여주었다.

시스템의 움직임을 가장 둔화시키는 옵션은 Test 95의 Default Color Depth [for textures] 의 옵션이 Use desktop color depth인 상황에서 Buffer-Flipping Mode 의 옵션이 Use block transfer, Vertical Sync의 옵션이 On by default 인 경우로 나왔다. Vertical Sync 의 옵션 On by default는 시스템의 성능에 부하를 가장 크게 하는 것을 알 수 있다. 이번 연구를 통한 최적화 OpenGL 3D API 그래픽 서브시스템 환경과 부하가 가장 큰 성능저하 OpenGL 3D API 그래픽서브 시스템 환경은 아래의 Table 19, 20 과 같다.

Table 19 최적화 OpenGL 3D API 그래픽 서브시스템 환경

OpenGL Settings	Default Color Depth for textures	Buffer - Flipping Mode	Vertical Sync
Enable Overlays, Enable conformant OpenGL texture clamp behavior	Always use 32 bpp	Auto - select	Off by default

Table 20 부하가 가장 큰 성능저하 OpenGL 3D API 그래픽서브 시스템 환경

OpenGL Settings	Default Color Depth for textures	Buffer - Flipping Mode	Vertical Sync
Disable support for enhanced CPU instruction set, Use unified back/depth buffer, Enable Overlays, Maximize texture memory	Use desktop color depth	Use block transfer	On by default

IV. 결론

본 연구서는 현재 일반적인 개인용 컴퓨터 하드웨어 및 소프트웨어 환경에서 OpenGL 3D API를 지원하는 그래픽 서브 시스템에서의 3D 성능 평가를 위한 테스트 과정을 수행하여, 운용 어플리케이션에 최적화된 환경 설정 기능과 제어에 대한 평가를 수행하였다. 그 결과 다음과 같은 결론을 얻을 수 있었다.

1. CAD/CAM/CAE 어플리케이션은 물론 일반 3D 그래픽 어플리케이션 제작에 있어서도 표준적인 소프트웨어 툴 키트인 OpenGL 기반의 윈도우 운영체제에서 3D Graphics 시스템을 구축할 때 세부적인 기능에 따라 다양한 결과를 분석하여 성능을 최대로 발휘할 수 있는 조건을 찾아내었다.

2. 개발자 측면에서 그래픽 하드웨어 및 3D API의 인식 및 이해 부족에 기인한 성능의 둔화를 각각의 세부사항에 따른 옵션 분석을 통해 성능향상을 모색하고 최적 환경에 적합한 환경구성을 개발자가 수행할 수 있도록 제안하였다.

3. OpenGL 기반의 윈도우 3D Graphics 의 환경최적화에 따른 데이터뿐만 아니라 현재 개발자 환경에서도 사용되고 있는 Unix 계열이나, 향후 차세대 OS로 주목받고 있는 리눅스 환경에서 OpenGL 3D API를 구현하고자 할 때 비교자료로 활용할 수 있다.

4. 본 연구에서 최적화 OpenGL 3D API 그래픽 서브시스템 환경은
OpenGL Settings의 옵션이 Enable Overays, Enable conformant OpenGL texture clamp behavior 일 때
Default Color Depth [for texture] 의 옵션이 Always use 32 bpp,
Buffer - Flipping Mode의 옵션이 Auto - select,
Vertical Sync의 옵션이 Off by default가 선택된 경우 그래픽 서브시스템에 가장 최적화 된 성능을 나타내었다.

5. 부하가 가장 큰 성능저하 OpenGL 3D API 그래픽서브 시스템 환경은
OpenGL Settings의 옵션이 Disable support for enhanced CPU instruction set, Use unified back/depth buffer, Enable Overlays, Maximize texture memory 일 때 Default Color Depth [for texture] 의 옵션이 Use desktop color depth,
Buffer Flipping Mode의 옵션이 Use block transfer,
Vertical Sync의 옵션이 On by default 가 선택된 경우 그래픽 서브시스템에 부하가 가장 많이 걸리는 것으로 나타났다.

참고 문헌

[1] Renate Kempf and Chris Frazier, "OpenGL Reference Manual Second Edition", Addison Wesley Developers press

[2] Mason Woo, Jackie Neider, Tom Davis, “OpenGL Programming Guide Second Edition”, Addison Wesley Developers press

[3] David F. Rogers, J. Alan Adams, “Mathematical Elements for Computer Graphics Second Edition”, McGraw-Hill International Editions

[4] Ron Fosner, “OpenGL Programming for Windows 95 and Windows NT”, Addison Wesley Developers press

[5] Vera B. Anand, “Computer Graphics and Geometric Modeling for Engineers”, Willey

[7] I.D FAUX, B.sc., Ph.D., and M.J.Pratt, M.A., M.Sc., “Computational Geometric for Design And Manufacture”, Ellis Horwood Limited

[8] Edward Angel, “Interactive Computer Graphics - A Top-Down Approach with OpenGL”, Addison Wesley Developers press

[9] Richard S, Wright, "OpenGL Super Bible second edition", 인포북

[10] Walnum.C, “3D Graphics Programming with OpenGL”

[11] Edward Angel, "OpenGL을 이용한 컴퓨터 그래픽스"

[12] Kilgard.M, "OpenGL Programming for the X-Windows System"

[13] http://www.spec.org/gpc/opc.static/proe02.html

맺음말

본 논문이 완성되기까지 부족한 저에게 끊임없는 지도와 관심으로 학문적인 면에서나 인간적인 면에서 항상 따뜻하게 지켜봐주신 정재현 지도교수님께 깊은 감사의 마음을 전합니다. 또한 바쁘신 와중에서도 저의 논문심사를 위해 많은 조언과 가르침을 주셨던 조종래 교수님과 박휴찬 교수님께도 감사드립니다.

이 논문을 내기까지 정말 많은 도움을 주신 같은 연구실의 김희중 선배님께도 감사를 전합니다. 또한 작년 한 해 동안 같은 연구실 생활을 하며 힘들 때 마다 도움과 배려를 함께 해준 김정욱 선배님께도 감사의 마음을 함께 드립니다. 동기 재혁이, 후배 종식이 에게도 깊은 수고와 감사의 마음을 전합니다. 그리고 논문을 쓰며 힘들고 지칠 때 마다 즐거움과 웃음을 선사하며 활력을 되찾게 해준 동아리 후배님들에게도 고마운 마음을 글로 대신하고자 합니다.

마지막으로 어떤 순간에도 항상 묵묵하게 지켜보시며 큰 힘과 안정을 주신 부모님과 가족들에게 더할 수 없는 감사의 마음과 함께 이 논문을 드립니다.