'91 韓國自動制御學術會議論文集 1991. 10.22~24

PHIGS를 지원하는 그래픽 시스템의 개발

박재현, 노갑선, 박정우, 장래혁, 구경훈, 이재영, 권욱현

서울대학교 공과대학 제어계측공학과

Development of Graphics System Supporting PHIGS

Jaehyun Park, Gab Seon Rho, Jung Woo Park, Naehyuck Chang, Kyunghoon Koo,

Jaeyoung Lee, and Wook Hyun Kwon

Department of Control and Instrumentation Engineering

Seoul National University

ABSTRACT

In this paper, a graphics system supporting PHIGS and PHIGS+ is suggested and its hardware structure and software are described. The developed graphics system is a multi-processing system that uses 6 i860 RISC CPU's and supports PHIGS and PHIGS+ language in a hardware level. The developed system under tested is able to draw 160,000 3-D polygones in one second when each polygon has 100 pixels and is shaded with Gouraud shading.

1. 서론

자동제어 분야가 발전되고 다양한 형태를 가지면서 자동화 시스템의 적용 결과를 미리 예상하기 위하여 시스템의 시뮬레이션은 필수적인 요소로서 인식되어왔다. 과거의 시뮬레이션은 주로 수치해석에 따른 수치적 시뮬레이션이 주종을 이루고 있으나 컴퓨터의 그래픽 성능이 향상됨에 따라 점차로 수치적 결과 나열에서 실제 현상을 그대로 그래픽으로 표현하는 그래픽 시뮬레이션으로 발전하는 추세에 있다. 이러한 추세는 공장 자동화의 대표적인 기기인 로보트나 수치제어기(NC)등에서 두드러지게 나타난다. 예를들면 로보트의 실시간 동작 모습 등을 시뮬레이션하거나 전체 생산 라인의 동작 상태를 실시간으로 시뮬레이션하는등의 작업들은 자동화 분야의 하나로서 중요한 역할을 하게 되었다. 그러나 이러한 자동화 시스템의 대부분은 고속의 반응 속도를 가지며 따라서 이들 자동화 시스템의 시뮬레이션을 위해서는 시뮬레이션 결과를 실시간으로 표현해주는 그래픽 시스템을 필수적으로 필요로 하게 된다. 시뮬레이션 결과를 그래픽으로 표현하기 위해서는 3가지 요소를 모두 갖춘 그래픽 시스템이 필요하게 되는데 우선 첫째로 실제 모델(로보트 등)을 원래 모습 그대로 표현하기 위한 고화질의 그래픽 시스템이 필요하다. 실제 모델을 섬세하게 표현하기 위해서는 우선 1280 * 1024 화소 이상의 고 해상도를 가져야 하며 표현할 수 있는 색도 16,000,000가지 이상의 다양한 색을 표현해야 한다. 그리고 실제 환경에서의 모델을 표현하기 위하여 주변의 빛과 모델의 재질, 반사도 등을 고려하여 색의 변화를 주어야 하며 모델이 3차원 공간에서 동작하는 모습을 적절하게 표현해야 한다. 둘째 요소는 위와 같은 섬세한 그래픽을 실시간으로 표현할 수 있는 고속의 동작 성능을 가져야 한다는 것이다. 대부분의 자동화 기기가 고속으로 움직이는 것을 고려할때 아무리 섬세한 그래픽 모델이라도 실시간으로 표현할 수 없다면 시뮬레이터로서는 아무런 가치를 가질 수 없는 것이다. 따라서 시뮬레이터로서의 기능을 수행하기 위해서는 고속의 그래픽 기능이 필요하게 된다. 셋째로 복잡한 모델을 손쉽게 표현할 수 있는 그래픽 언어가 필요하며 이 언어는 다른 종류의 그래픽 시스템에 쉽게 이식이 가능해야 한다. 아무리 고성능의 그래픽 시스템을 가지고 있어도 최종적으로 시뮬레이션 모델을 작성하기 위한 그래픽 언어가 복잡하면 시뮬레이션 프로그램을 작성하기가 어렵게 된다. 또한 하나의 시스템에서 개발된 시뮬레이터가 다른 시스템에 그대로 이식되기 위해서는 세계적으로 표준적으로 사용되는 언어를 이용해야 한다.

현재까지 사용되는 그래픽 시스템은 여러 가지가 있으

나 많이 사용되는 PC나 일반 웍스테이션 등은 위와 같은 요소를 만족시키지 못하며 위와 같은 조건을 만족시키는 그래픽 시스템으로는 아주 고가의 그래픽 전용 장비가 일부 사용되고 있다.

본 논문에서는 위에서 보인 세가지 요소를 모두 만족하고 비교적 적은 가격으로 구현할 수 있는 그래픽 시스템을 제안하고 제안된 그래픽 시스템에서 사용되는 하드웨어 및 소프트웨어의 특징을 제시한다.

2. 그래픽 시스템과 3차원 그래픽스 개요

그래픽스 시스템에서 하는 기능은 크게 세가지로 나눌 수 있다. 첫째는 그리고자 하는 그림의 형태를 모델링하고 구조(Structure)를 작성하는 과정으로 3차원 그래픽 명령을 사용하여 데이타베이스를 구축하는 과정이다. 둘째는 데이타 베이스에 저장되어 있는 그래픽 모델을 해석하여 화면에 보여줄 수 있는 형태의 래스터 그래픽 영상을 작성하여 프레임 버퍼에 기록하는 과정이다. 세번째 과정은 프레임 버퍼에서 나온 래스터 그래픽 영상을 색상 변화을 거친 후 D/A 컨버젼 하여 R,G,B 의 아날로그 전압을 CRT로 공급하는 것이다. 이러한 일련의 그래픽스 구현 과정은 그림 1에서 보는 것과 같이 여러 단계로 세분 될 수 있으며 모든 과정은 파이프 라인 구조를 가지고 있다[5,6,7,8].

2.1. PHIGS 와 PHIGS PLUS

컴퓨터 그래픽를 위한 최초의 표준화된 그래픽스 스펙은 GKS이며 이는 2D 그래픽에 제한 되어있었다[1]. GKS의 3D 확장은 1988년에 이루어져 GKS-3D로 확정되었다[2]. GKS는 논리적으로 서로 연관된 라인, 폴리곤(Polygon), 문자열 등의 프리미티브(Primitive)들을 그룹지을 수 있으며, 이 그룹의 어트리뷰트를 세그먼트(Segment)라 부른다. 이 세그먼트는 평면적인 구조만을 취할 수 있다. 이러한 평면적인 구조의 컴퓨터 그래픽스 모델은 정적인 그래픽을 구성하는 데는 용이하나 동적인 그래픽 즉 에니메이션 등을 구현할때 어려움이 따르며 그림의 수정 및 사용자와의 대화 형태의 운용에도 어려움이 따른다.

반면에 1988년에 ISO를 통하여 확립된 PHIGS (Programmer's Hierachical Interactive Graphics System)는 스트럭쳐(Structure) 단위의 계층적인 구조를 수용할 수 있도록 되어있으며 PHIGS의 모든 스트럭쳐는 각기 스케일링, 회전, 이동을 통하여 다이나믹한 움직임을 할 수 있고, 데이타베이스가 변하면 자동으로 화면을 갱신한다[3,4]. 따라서 좀더 동적이며 복잡한 형태의 3차원 그래픽을 손쉽게 작성하기 위하여 PHIGS는 많이 사용되며 PHIGS를 발전시켜 오브젝트의 빛의 반사를 고려하여 더욱 실제적인 모델을 할 수 있는 기능 (Pseudorealistic Implementation)을 추가한 것이 PHIGS+이며 이는 1991년에 Standard로서 확정되었다. PHIGS 및 PHIGS+의 기능을 요약하면 아래와 같다.

- ○ Structure hierarchy
- ○ Traversal-time binding
- ○ Name sets and filters
- ○ Transformation pipeline
  - Modeling transformation
  - Viewing transformation
  - Clipping and view mapping
  - Workstation transformation
- ○ Structure edting

2.2 PHIGS의 구성

PHIGS에서 사용되는 그래픽 데이타 및 응용 데이타들로 이루어진 모델에 대한 정보는 CSS(Centralized Structure Store)라는 장소에 위치한다. 응용 프로그램은 모델의 그림을 표시하고 대화하기 위하여 그래픽 장치 (PHIGS 용어로는 워크스테이션)를 이용하게 된다. 여기서 워크스테이션이란 그래픽 출력이나 그래픽 입력을 제공하는 하드웨어 및 소프트웨어의 조합을 의미한다.

2.2.1 PHIGS의 적용과정

먼저 응용 프로그램은 모델을 정의한다. 정보를 저장하기 위한 데이타의 기본 단위는 구조요소 (structure element)라고 하고, 그 집합을 구조 (structure)라고 한다. 응용 프로그램은 서로 다른 구조들을 모두 연결하여 계층구조 (hierarchies)를 형성함으로써, 한 모델의 서로 다른 부분들간의 논리적 관계를 정의한다. 이러한 계층구조를 이용하면, 한 모델에 자주 나타나는 데이타라도 단 하나의 복사본만을 저장해도 된다는 잇점이 있다. 일단 모델이 정의되면, 응용 프로그램은 모델 중 표시할 부분(Window)과 적절한 시점 (view point), 그리고 사용할 워크스테이션을 지정하여 그림을 생성할 수 있게 된다. 즉, PHIGS가 모델의 정의를 읽어들여 필요한 그래픽 정보를 추출하고 그림을 그리는 것이다. 이러한 표시 과정을 트래버스 (traverse)이라고 한다. 만약, 응용 프로그램이 구조의 내용이나 시점을 변경하여 모델을 편집하는 경우, 표시되는 그림은 변경 사항을 반영하여 자동적으로 바뀌게 된다.

2.2.2 그래픽 출력 및 속성 (attribute)

PHIGS는 출력 기본요소 (primitive)라고 불리우는 그래픽 출력의 기본 형식을 제공한다. 이것들로 2차원 및 3차원 그림 (모델의 그래픽 데이타)를 구성할 수 있다. 그래픽 출력 기본 요소의 종류는 8가지이다. 한편 그래픽 출력의 형태와 형식은 기하정보 (geometric information)와 속성 (attribute)에 의해서 결정된다.

2.2.3 모델

많은 그림들이 본래부터 계층구조를 가지고 있다. 이러한 자연스러운 계층구조는 구조망 (structure networks)이라고 불리우는 연결된 구조들의 집합으로 나타낼 수 있다. 그리고, 한 구조는 또다른 구조를 참조할 수 있어 자주 사용되는 구조의 경우 저장 공간을 줄일 수 있다. PHIGS에서는 그림의 각 부분들을 일종의 지역 좌표계인 모델링 좌표계로 분리시켜 정의할 수 있는데, 모델링 변환을 거쳐 그림의 각 부분들을 하나의 세계 좌표계 (world coordinate system)로 조합한다. 구조들이 워크스테이션으로 포스트 (post)되면 트래버스가 일어나 워크스테이션에 표시될 그림이 생성된다. 한편, 구조내의 구조요소들이 삽입, 수정될 수 있도록 요소 포인터 (element pointer) 및 라벨 요소 (label element)가 그 위치를 지정하게 된다.

2.2.4 좌표변환 및 시점

PHIGS에서는 실행단계에 따라 모두 5가지의 좌표계가 사용되며 각 단계에 적합하도록 변환된다. 그리고 시점변화에 따라 3차원 영상이 변화될 수 있도록 시점기준 좌표계 (view rerference coordinate system)를 제공한다.

2.2.5 그래픽 입력

응용 프로그램과 그림과의 대화가 가능하도록 PHIGS는 그래픽 입력을 지원한다. 응용 프로그램에서 대화가 필요한 경우 화면상의 프롬프트 (prompt)나 메뉴의 형태로 나타나며, 각 입력장치를 통해 데이타가 입력된다.

2.3 PHIGS PLUS

PHIGS PLUS는 좀 더 진보된 그래픽 모델링과 컴퓨터 그래픽스를 제공하기 위해 국제표준인 PHIGS의 확장으로 제안되었다. PHIGS PLUS는 현재까지도 개발중에 있다. 현재까지의 확장 내용을 보면 다음과 같다.

(1) 2차원 및 3차원적 곡선과 곡면 (surface)을 위한 부가적인 출력 기본요소
(2) 조명원 (source of illumination)을 정의하기 위한 함수
(3) 출력 기본요소의 묘사를 제어할 수 있게 하는 음영법 (shading)
(4) 좀 더 유연한 색상의 지정을 위한 일반 색상 (general colour)

이로써 음영처리된 광범위한 영상을 만들어 낼 수 있다.

3. SGM-1의 하드웨어 구조

고속의 3차원 그래픽을 목적으로하는 SGM-1(Seoul National University Graphic Machine Model 1)의 전체 시스템은 고속의 64비트 RISC CPU인 i860을 6개 사용하여 구성되어 있다. 6개의 CPU중 1개는 주 CPU로서 그래픽 이미지의 생성을 담당하며 다른 1개의 CPU는 그래픽 구조 분석 (Traversal)을 담당하고 4개의 CPU는 2개씩 나누어 그래픽의 묘화(Rendering)을 담당한다. 묘화가 끝난 그림은 고속의 화면 제어기를 통하여 화면에 나타나게 된다. 전체 시스템의 블록 다이어 그램은 그림 2와 같다.

3.1 호스트 프로세서

호스트 프로세서(Host Processor)는 그래픽 이미지를 생성하는 프로세서로서 일반적인 범용 컴퓨터로 구성된다. 따라서 일반적인 구조로서 i860 CPU를 1개 사용하고 추후에 UNiX O/S를 사용하기 위하여 필요한 ROM, RAM, Serial Port, HDD, Tape, Ethernet, Keyboard 등으로 구성되어 있다. 호스트 프로세서에서는 그리고자 하는 그래픽 모델을 생성하고 전체 그림의 스트럭쳐를 형성한다. 모든 그림은 PHIGS와 PHIGS+ 형태의 스트럭쳐로 구성되며 구성된 그림은 CSS(Centralized Structure Store)에 저장이 된다. 일단 CSS에 저장된 그래픽 이미지는 구조 분석기(Traversal Processor)를 통하여 그래픽 시트템의 내부로 전달된다. 현재 구성된 호스트 컴퓨터는 SUN PHIGS와 비슷한 형태의 PHIGS와 PHIGS+의 C Binding을 지원하며 CSS는 256K Byte로 구성되어 있다.

SGM-1에서 구현된 CSS는 용도에 따라 크게 GIA (Global Information Area), IPC(Interprocessor Channel), GOS(Graphic Object Store)의 세부분으로 나누어진다. GIA는 PHIGS의 PDT(Phigs Description Table), PWDT(Phigs Workstation Description Table), Pre-defined Variable등을 저장해 두는 상수 영역과 TSL(Traversal State List)와 같은 변수 영역으로 나누어 사용된다. IPC는 모든 프로세서 사이의 통신을 위한 영역으로 일반 채널과 긴급 채널을 따로 두어 이용한다. GOS는 실제로 PHIGS 명령에 따른 스트럭쳐 또는 GIA영역의 변화 여부를 표현하는 영역으로 PHIGS 명령어중 그래픽 구조에 관한 정보는 GOS에 저장되지만 속성 변화에 대한 명령은 호스트 CPU에 의하여 GIA가 변경되고 속성 변

화에 관한 정보만 GOS에 표현함으로써 실제적인 메모리 공간의 절약과 고속의 구조 분석을 가능하게 한다.

3.2 구조 분석기(Traversal Processor)

PHIGS는 구조적인 그래픽 언어로서 모든 그래픽 모델은 그래픽 구조(Graphic Structure) 형태로 존재한다. 이는 계층적인 그래픽 모델을 가능하게 하여 효과적인 그래픽의 변경 및 동작 표현이 가능하게 된다. 따라서 PHIGS의 그래픽 구조로 작성된 그림을 실제로 화면에 그리기 위해서는 계층적으로 구성된 그림을 순차적으로 해석하여 구조를 분석하고, 최종적으로 그림을 그릴 수 있는 형태로 변경할 필요가 있다. SGM-1에서는 구조 분석기(Traversal Processor)가 CSS에 저장되어 있는 그래픽 구조를 해석하여 최종적인 그래픽 오브젝트를 추출하게 되며 추출된 그래픽 오브젝트에 대하여 3차원 공간에서의 위치 결정을 한다. 3차원 공간에서의 위치 결정은 그리고자 하는 물체의 위치와 그 물체을 바라보는 시점의 위치의 상대적인 위치를 계산하여 3차원 물체를 이동, 회전시켜 절대 위치를 구한다. 또한 3차원 그래픽스에서 절대 공간 위치 결정에는 프로그램 작성자가 사용하는 모델링 좌표(Modeling Coordination)과 월드 좌표(World Coordination)을 정규 좌표(Normalized Projection Coordination)으로 변경하는 좌표계 변환도 동시에 수행되어야 한다. SGM-1의 구조 분석기의 블록다이어 그램은 그림 3와 같다.

3.3 묘화 장치(Rendering Processor)

묘화 장치는 구조 분석기에 의하여 추출된 최종 그래픽 오브젝트를 주변의 광원과 물체의 색, 물체 표면의 광택, 반사율 등을 고려하여 채색하는 일을 맡는 장치로서 실제적인 그림을 그리는 역할을 한다. 따라서 묘화 장치가 행해야 하는 계산은 매우 많으며 실제로 묘화 장치가 행하는 알고리즘에 따라 최종적인 그림의 화질이 좌우 된다.

SGM-1에서는 4개의 CPU가 묘화를 담당하는데 2개씩 나누어 2개의 병렬 처리 경로를 가지며 각각의 처리 경로는 다시 2단계의 파이프라인 구조로서 병렬 처리를 행한다. 이는 묘화 과정의 두 단계인 Lighting과 Shading을 파이프라인으로 처리할 수 있다는 그래픽의 특성을 이용한 것이며 처리 경로를 두개로 나눈것은 묘화 장치 이후의 화소 생성 장치의 성능을 하나의 CPU가 처리할 수 있는 성능의 2배 이상으로 설계함으로써 병렬 처리에 있어서의 효율을 극대화 시키기 위한 것이다. 4개의 i860 프로세서를 병렬처리 함에 있어서 모든 프로세서가 최상의 성능을 발휘하고 프로세서 상호간의 데이타 교환이 원활히 이루어 지도록 위하여 각각의 프로세서마다 시스템 컨트롤러 전용칩을 개발 사용하였다. 시스템 컨트롤러 칩의 내부 블럭 다이아그램을 보면 그림 4와 같다.

묘화 장치의 2단계 파이프라인 구조중 첫번째 CPU는 주로 Lighting을 담당하는 CPU로서 PHIGS+의 4가지 Lighting 모델인 Ambient, Directional, Positional, Spot의 4가지 종류의 광원을 지원하며 각각의 광원과 표시 물체의 표면 반사율 등을 고려하여 공간 내에서 보이는 색을 구하는 일을 한다. 그리고 두번째 CPU는 Shading을 담당하는 프로세서로서 기본적으로 Flat Shading, Gouraud Shading, Phong Shading에 따른 Shading과 Interpolation을 수행한다. 두개의 병렬처리 경로의 마지막 프로세서는 고속의 Interpolation 알고리즘을 수행하며 120nSec 에 하나의 화소를 생성함으로써 두개의 프로세서가 번갈아 화소를 생성하는 경우 최고 60 nSec에 하나의 화소를 그릴수 있다.

묘화 장치의 최종단에는 묘화 장치에서 사용한 좌표계를 실제로 표시할 화면의 좌표계로 변환 시켜주는 Display Cordinator Converter가 존재한다. 이는 SGM-1의 화면의 가로가 1280 화소로 구성되어 있고 고속의 화면 처리를 위하여 이를 5:1 Interleaving을 하도록 설계되어 있기 때문에 화면 메모리의 어드레싱 스페이스가 5진수로 구성되기 때문이다. 이를 위하여 SGM-1에서는 좌표계 변환을 실시간으로 처리하는 전용 칩인 Display Cordinator Convert를 설계 사용하여 메모리 낭비를 줄이고 좌표 변환에 걸리는 시간을 대폭 단축 시켰다.

3.4 화면 제어기(Display Controller)

PHIGS로 작성된 그림이 화면에 표현되기 위해서는 그래픽 모델을 분석하고 묘화를 마친후 최종적으로는 화소를 생성하여 화면에 표현하여야 한다. 화소의 생성은 묘화 프로세서중 마지막 프로세서가 Shading을 하면서 동시에 화면에 표현될 화소의 공간내의 위치와 색을 지정하는 과정에서 이루어진다. 일단 생성된 화소는 2개의 Frame Buffer중 현재 화면에 표시되고 있지 않는 Frame Buffer에 저장이 된다. 그런데 PHIGS로 작성된 그래픽은 3차원 공간내에 위치하게 되므로 시각점의 위치로부터 거리가 먼 물체는 가까운 물체에 의하여 가려져 보이지 않게 된다. 따라서 3차원 공간내에서 보이지 않는 물체와 보이지 않는 면을 제거하는 과정이 필요하게 되는데 이는 화면 제어기 내의 Z-Buffer를 이용하여 행해진다. 화면 제어기는 묘화 프로세서에 의하여 고속으로 생성된 화소의 Z 좌표의 값과 같은 위치에 존재하고 있던 점의 Z 좌표 값을 비교하여 두개의 화소중 가까운 쪽에 위치한 화소만을 그리

게 된다. 이런 Z-Buffer를 이용한 은면제거(Hidden Surface Elimination) 과정을 포함한 화면 제어기의 성능이 최종적인 그래픽 컴퓨터의 성능이 된다.

SGM-1에서는 묘화 프로세서가 생성할 수 있는 최대 속도인 60nSec이내에 생성된 화소를 처리하기 위하여 5개의 병렬성을 가진 그림 5와 같은 화면 제어기를 설계하였다. 화면 제어기 내의 FBCTRL(Frame Buffer Controller)은 최대 300nSec 이내에 모든 동작을 마치게 되는데 따라서 평균적으로는 60nSec 이내에 은면제거를 포함한 모든 동작을 마치게 되고 이는 초당 16M 화소를 그리는 능력을 가지게 된다. 이는 100화소의 Polygon을 그릴경우 초당 160,000개의 Polygon을 그릴 수 있음을 의미한다.

SGM-1에는 Z-Buffer가 24비트로 구성되어 있으며 각 화소의 색은 24비트로 16,000,000 가지의 색을 가질수 있다. 화면의 구성은 1280 × 1024개의 화소로 2개 화면분의 Frame Buffer가 마련되어 있어 화면에 보여주는 Buffer와 데이타 갱신을 위한 Buffer가 따로 있다. 이는 에니메이션 등의 동적 그림을 표현할 때 유용하게 사용될 수 있다. 따라서 SGM-1이 가지는 총 디스플레이 메모리는 10.8MByte이다.

일단 Frame Buffer에 저장된 데이타는 고속의 RAMDAC을 통과하여 Analog 신호로 변환되어 화면에 표시되는데 주어진 화면의 해상도를 지키기 위해서 화면의 수평동기는 72KHz, 수직 동기는 65Hz를 유지하고 이때 각 화소의 전송은 약 90MHz의 속도를 가진다. SGM-1에서는 Display Timing Controller Chip이 수평 동기 신호와 수직 동기 신호 등의 화면 제어 신호와 DRAM으로 구성된 Frame Buffer, Z-Buffer등의 리프레쉬(Refresh) 신호 등을 제어한다.

4. SGM-1의 소프트웨어

SGM-1의 소프트 웨어는 크게 나누어 두가지로 나누어 지는데 첫번째는 사용자가 C 언어를 이용하여 PHIGS를 구현하는데 필요한 C-Binding Library이며 두번째는 실제로 그래픽을 구현하는 그래픽 구현 소프트웨어이다. SGM-1에서 사용한 C-Binding 형식은 PHIGS인 경우에는 이미 확정된 PHIGS Standard에 기초를 하고 PHIGS+인 경우에는 아직 Standard가 미확정 상태이므로 SUN PHIGS+와 비슷한 형식을 따르고 있다. 따라서 많은 경우 SUN 워스테이션에서 작성된 그래픽 소프트 웨어는 대부분 수용할 수 있는 형태를 따르고 있다. 그래픽 구현 소프트웨어는 5개의 i860 CPU가 나누어 계산하는 프로그램으로 3차원 그래픽을 구현하기 위하여 Traverse, Transformation, Clipping, Lighting, Shading의 5가지로 분류된다. Traverse 프로그램은 C-Binding을 통한 PHIGS형태의 그래픽 구조를 CSS로부터 읽어 가장 최소단위의 그래픽 물체를 추출하고 그 물체의 속성을 파악하는 과정으로 PHIGS의 계층적 구조를 해석하는 일을 주로 한다. Transformation은 그래픽 물체가 3차원 공간에서 변환을 일으킬 경우 그의 절대적인 위치를 구성하는 단계로서 Model Transformation, Local Transformation, Global Transformation의 3가지 Transformation을 행하며 각각의 Transformation에는 평행이동, 회전이동, 확대축소 등의 3가지 변환이 가능하다. Clipping은 Transformation을 마친 그래픽 물체 중 실제로 화면에 표시될 수 있는 영역 내의 그림만 추출하는 과정으로 3차원 Clipping을 행하게 된다. Lighting은 화면내의 광원으로부터의 빛에 따라 물체의 색을 결정하는 과정으로 광원의 위치, 종류, 광원과 물체와의 각도, 물체의 표면 성질, 물체의 색, 배경색 등을 고려하여 실제 상황과 가장 가깝게 물체의 색을 정하게 된다. Shading은 Lighting의 결과를 바탕으로 물체에 채색을 하는 과정으로 Flat Shading, Goraud Shading, Phong Shading의 세가지가 가능하다. 모든 소프트웨어는 속도를 높이기 위하여 기계어로 직접 작성되었으며 RISC CPU인 i860의 특성을 최대한으로 이용하여 CPU내부에서의 병렬 처리를 최대한으로 이용하여 작성되었다.

5. 결론

본 논문에서는 자동화 시스템의 실시간 시뮬레이션 등에 필요한 고속, 고화질의 그래픽 시스템의 개발 결과를 제시하였다. 개발된 그래픽 시스템은 고속의 RISC CPU인 i860 CPU를 6개 사용하였으며 각각의 CPU 시스템에는 고속의 자료 처리와 공유 자료의 전송을 위한 시스템 컨트롤러를 설계 사용하였으며 고속의 화소 처리를 위하여 화면 제어기는 6개의 전용 칩을 설계하여 제작 하였다. 현재 시험중인 개발 시스템의 최종 성능은 Gouraud Shading을 하는경우 100 화소의 폴리곤을 초당 160,000개 그릴 수 있을 것으로 예상된다.

개발된 그래픽 시스템은 그래픽 언어로서 세계 표준인 PHIGS 와 PHIGS+를 하드웨어 수준에서 지원하므로 고품질의 그래픽을 고속으로 그릴 수 있으며, 타 그래픽 시스템과도 호환성을 유지한다.

참고 문헌

[1] Peter R.Bono, "Guest Editor's Introduction Graphics Standards," *IEEE Computer Graphics and applications* Aug 1986 pp 12-16.

[2] Richard F.Puk, John I.McConnel, "GKS-3D: A Three-Dimensional Extension to the Graphical Kernal

System," *IEEE Computer Graphics and applications* Aug 1986 pp 42-49.

[3] David shuey, Thomas P.Morrissey, "PHIGS: A Standard, Dynamic, Interactive Graphics Interface," *IEEE Computer Graphics and applications* Aug 1986 pp 50-57.

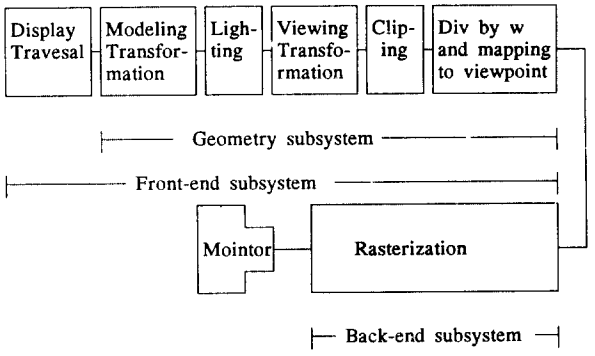
[4] T.L.J. Howard, W.T. Hewitt, R.J. Hubbold, K.M. Wyrwas, *A Practical Introduction to PHIGS and PHIGS PLUS*, Addison-Wesley Publishing Company, 1991.

[5] Andrew S. Glassner, *Graphics Gems*, Academic Press Inc, 1990.

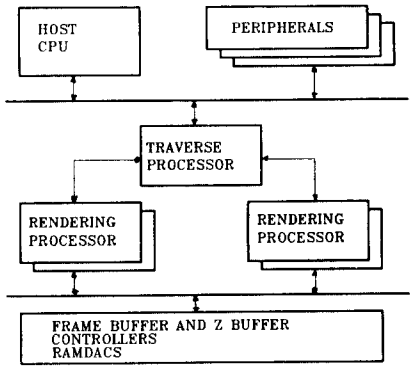
[6] James Arvo, *Graphics Gems II*, Academic Press Inc., 1991.

[7] James D Foley, Andries Van Dam, Steven K.Feiner, John F.Hughes, *Computer Graphics Principle and Pratice* ,Second Edition, Addison-Wesley Publishing Company, Inc.

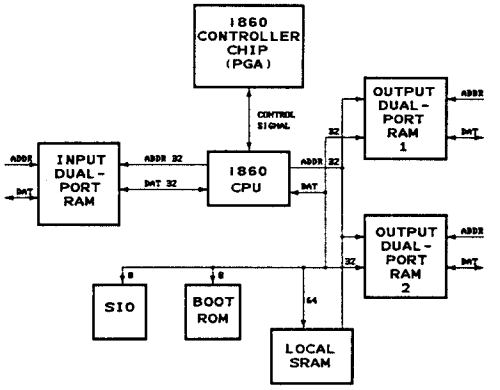
[8] Steven Harrington, *Computer Graphics a Programming Approach*, Second Edition, McGraw-Hill Internal Editions.



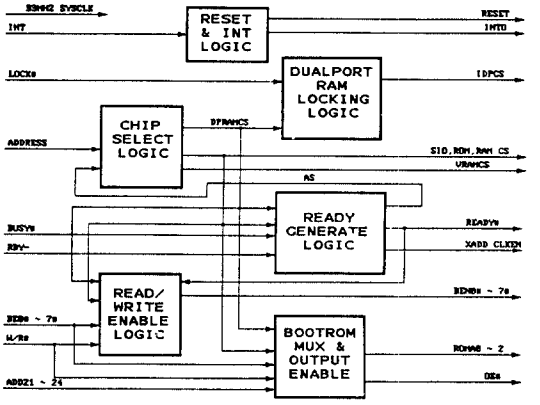
(그림 1) 그래픽스 파이프라인



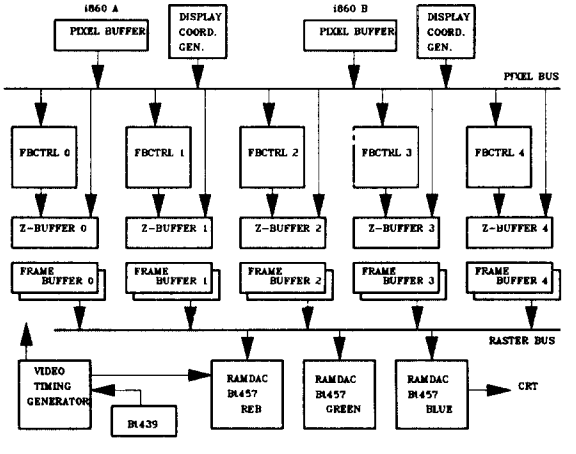
(그림 2) 전체 시스템 Block Diagram



(그림 3) 구조 분석기 Block Diagram



(그림 4) 시스템 콘트롤러 내부 Diagram



(그림 5) 화면 제어기 Block Diagram