工 學 碩 士 學 位 論 文

분산처리 네트워크 방식을 이용한 실시간 선박조종시뮬레이터의 개발에 관한 연구

A Study on Construction of Real Time Ship-Handling Simulator Using the Distributed Processing Network Method

指導敎授 孫景浩

2001年 2月

韓國海洋大學校 大學院

造船工學科

洪 基 榮

목 차

A Study on Construction of Real Time Ship-Handling Simulator Using the Distributed Processing Network Method

by

Ki Young Hong

Department of Naval Architecture

Graduate School of Korea Maritime University

Abstract

As sea traffic has been complicated and ships have been enlarged recently, we require a safety of ship in the coastal and areas. Although the maneuvering performance of ship herself almost clearly become, the important control quality of human being is not exactly grasped. The real time ship-handling simulator can be used as the most useful way when we understand the safety of ship maneuvering performance in the harbor area. The most important things of real time ship handling simulator are mathematical model and 3 dimensional image. We can manage the 3 dimensional image by using the 3 dimensional graphic technique. The distributed processing method is thought as one of the most effective ways to network communication system. We developed the real time ship handling simulator using the distributed processing network method.

Nomenclature

D	Fractal Dimension
I_{zz}	Moment of inertia about the z axis
J_{zz}	Added moment of inertia about the z axis
K	Total Length
K_n	Length of Triadic Koch Curve
m	Mass of Ship
m_x	Added mass in the x direction
m_y	Added mass in the y direction
N	Yaw moment
N_H	Yaw moment due to hull
N_P	Asymmetrical yaw moment acting on a hull by reserving rotation of a propeller
N_R	Yaw moment due to rudder
N_T	Yaw moment due to tugboat
N_W	Yaw moment due to wind
S	Length of Unit
r	Angular velocity
$\dot{r}$	Angular acceleration
u	Longitudinal component of ship velocity
$\dot{u}$	Longitudinal component of ship acceleration
u^*	Longitudinal component of apparent current velocity

v	Lateral velocity
$\dot{v}$	Lateral acceleration
v^*	Lateral component of apparent current velocity
X	Surge force
x_G	Distance from midship to center of gravity
X_H	Longitudinal force due to a hull
X_P	Longitudinal force due to a propeller
X_R	Longitudinal force due to a rudder
X_T	Longitudinal force due to a tugboat
X_W	Longitudinal force due to a wind
Y	Sway force
Y_H	Lateral force due to a hull
Y_P	Asymmetrical longitudinal force due to a hull by reversing rotation of a propeller
Y_R	Lateral force due to a rudder
Y_T	Lateral force due to a tugboat
Y_W	Lateral force due to a wind

Greek

α	x coordinate of center of m_y
ψ	Heading angle
ψ_c	Angle of current

List of Figures

List of Tables

1. 서언

최근에 해상교통이 복잡하고 선박이 대형화됨에 따라 선박의 항행안전의 문제가 중요시되고 있다.[1] 그래서 IMO(국제해사기구의 약칭)는 선박에 있어서 최소한도의 조종성능(조종성 기준)을 요구하게 되었다. 해상교통이 복잡하고 해역의 제약이 큰 항만과 같은 해역에서는 출입항 선박조종시의 성능을 고려할 필요가 있다. 그러므로 이와 같은 장소에서의 안전성은 선박자체의 조종성능과 항해자의 제어특성을 더하여 Man-Machine계와 항구의 항행환경과의 적합성 관점에서 파악하지 않으면 안된다. 오늘날, 선박자체의 조종성능에 관해서는 꽤 명확하게 알려지게 되었지만, 중요한 인간의 제어특성에 관해서는 정확하게 파악되지 않았다. 선박조종시뮬레이터는 특성이 명료하지 않은 인간을 System의 일부로 조합하여 선박조종현상을 재현하고, 항내 선박조종의 안전성을 검토할 때 가장 적합한 수단으로 사용될 수 있다.[2] 또한 항해자의 훈련에 있어서도 새롭게 항해교육을 받는 사람이나 도선사와 항해사에 대해서도 출입항 선박조종은 가장 중요하다고 생각된다.

본 연구는 위에서의 배경으로부터 출입항 선박 조종 시뮬레이션 하는 것을 주목적으로 실시간 선박 조종 시뮬레이터를 개발하였다. 현장감 있는 시계를 작성하는 방법으로 여러 가지 방법이 제안되고 있고 최근에는 고성능의 컴퓨터가 비교적 안정적인 가격으로 구입할 수 있기 때문에 정확한 화면을 비교적 간단히 얻을 수 있다. 본 논문에서는 현실감을 재현하는 방법으로 가상현실기법(Virtual Reality)을 이용하여 항내 선박조종의 주된 정보원이 되는 본선의 선수영상과 주위의 영상을 현장감 있게 재현하고 투영하는 것을 시험할 수 있었다.[3]

2. 출입항 선박 조종 시뮬레이터의 개요

2.1 시뮬레이터 설계방침

본 연구는 출입항시의 선박조종의 영상을 재현하는 것에 역점을 두고 시뮬레이터의 사양을 검토하였다. 중점이 된 것은 실시간 영상을 재현하기 위해 화면갱신시간을 정하였고 또한 항내의 모형과 같은 넓은 시계를 어떻게 확보하는지에 역점을 두었다. 또한 항내에서의 복잡 다양한 선박운동의 재현하기 위한 조종운동수학모델에 역점을 두어 시뮬레이터를 개발하였다.

2.2 시뮬레이션 처리방법

출입항 선박조종시 본선과 항만의 위치관계가 상당히 중요한 정보가 되기 때문에 조종운동방정식 계산에 의한 정확한 선체운동의 영상이 필요하게 된다.[2] 그 방법으로서 CGI방식의 3차원 그래픽 기법이 필요하게 된다. 또한 해역의 물체 뿐만 아니라 정확한 선수부 영상을 위해 상갑판상의 주요한 구조물에 대한 데이터베이스를 만들고 화상을 투영하는 방법으로 시계영상을 처리하였다. 항만이나 선박과 같은 대용량의 그래픽을 처리하기 위해서는 계층화된 데이터베이스가 필요하다. 또한 시뮬레이션은 가상 환경(Virtual environment)을 실세계와 같이 자연스럽게 관찰할 수 있도록 하는 응용분야로 정의된다. 다음은 가상현실(Virtual reality)에 대한 개념과 시뮬레이션의 적용에 대해 설명한다.

2.3 가상현실(Virtual Reality Method)

가상현실의 세계란 상상의 세계를 현실과 같이 만들어 내고 인체의 모든 감각기관(눈, 코, 귀, 입, 피부 등)이 인위적으로 창조한 세계에 몰입됨으로써 자신이 바로 그 곳에 있는 것처럼 느낄 수 있는 공간을 의미한다. 가상현실 세계에서는 현실 세계에 대한 시뮬레이션 뿐만 아니라 현실세계에서 불가능한 체험을 가능하게 해 줄 수 있다. 이런 가상현실의 실현은 멀티미디어의 기술 발전에 있다고 하겠다.

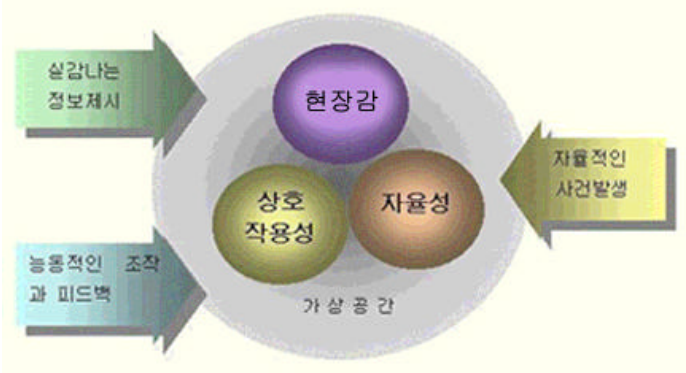


Fig. 1 Virtual Reality Systems

2.3.1 가상현실 시스템의 종류

(1) 몰입형 가상현실 시스템(Immersive VR System)

컴퓨터에 의해 만들어진 3차원 환경에 HMD(Head Mounted Display)등의 몰입형 장비를 착용하여 가상의 세계를 경험하고 상호 대화식으로 주고받는 시스템을 말하며 사용자가 현실과는 완전히 차단된 가상 환경만을 볼 수 있도록 하는 것이 가장 이상적인 형태의 가상현실이라고 할 수 있다. 그러나 고가의 장비를 필요로 하기 때문에 주로 연구 실험용으로 사용된다.

Fig. 2 Virtual Reality Example

(2) 비 몰입형 시스템(Non-immersive VR System)

탁상형 가상현실 시스템(Desktop VR System)이라고도 부르며, 모니터 화면에 나타난 영상을 사용자가 보면서 가상현실을 체험하는 것을 말한다. 가상 세계에 대한 몰입감이 떨어지는 등 부족한 면은 많으나 PC등 저가의 장비를 이용해 쉽게 사용이 가능하며 현재 대중적으로 많이 보급되어 있다. 비행 시뮬레이션과 선박 조종시뮬레이션은 일종의 비 몰입형 시스템에 속한다.

(3) **증강 현실** (Augmented Reality)

실세계와 가상의 이미지가 중첩되어 복합형 가상현실 시스템(Hybrid VR System)**을 구현하는 것을 말하며 사용자가 보는 현실세계와 부가 정보를 갖는 가상세계를 합쳐 하나로 보여 주는 것으로 최근 활발한 연구가 진행되고 있는 분야다. 가상현실 기법만으로 현실 세계를 완벽하게 대처하기 보다는 현실세계를 가상세계로 보완해 주는 개념의 기법이다.**

Fig. 3 Graphic Management for Augmented Reality

2.3.2 시뮬레이션을 구현하기 위한 그래픽 기법

시뮬레이션의 세계를 구현하기 위해 3차원 그래픽 기법은 중요하다. 3차원 그래픽 기법에는 기하변환(Geometric Transformation)이나 투영(Projection), 은닉면제거(Hidden Surface Removal), 이면제거(Backface Culling), 절단(Clipping) 등이 있다. 하지만 여기서는 현실감 있는 영상을 재현하기 위한 방법으로서 셰이딩(Shading)과 텍스쳐맵핑(Texture mapping)에 대해 알아보겠다.[4]

(1) Shading

조명에 의해서 표면 메쉬(Mesh) 위의 한 점이 지닌 수직벡터에 의해 반사광이 결정되고, 따라서 해당 점의 색이 결정된다. 이것을 기준으로 메쉬 내부 구석구석의 색을 결정하는 작업을 셰이딩(Shading, 陰影을 주기)이라 한다. 물체를 랜더링(Rendering)함에 있어서는 물체를 얼마나 현실감있게 보이는 지가 중요한 요소이다.

(Ⅰ) Flat Shading

단순하기 때문에 매우 빠른 렌더링 방법으로 Constant Shading이라고도 한다. 이 방법에서는 주어진 하나의 표면 메쉬를 전체를 동일한 색으로 칠한다. Flat Shading의 문제점은 메쉬 사이의 경계선이 필요 이상으로 뚜렷해지는 데 있다. 마하밴드효과(Mach Band Effect)라고 부르는 이 현상은 두 개의 면이 만나는 경계선 부근에서 어두운 면은 더욱 어두워지고 밝은 면은 더욱 밝게 보이는 일종의 착시현상이라 할 수 있다. 그 결과, 경계선 부근에서는 명암이 대조(Contrast)를 이루어 어둡고 밝은 띠(밴드, Band)가 형성된다. 이는 사람의 시각적 인식체계가 윤곽선(Outline)을 더욱 철저하게 잡아내고자 노력하기 때문에 일어나는 현상이다.

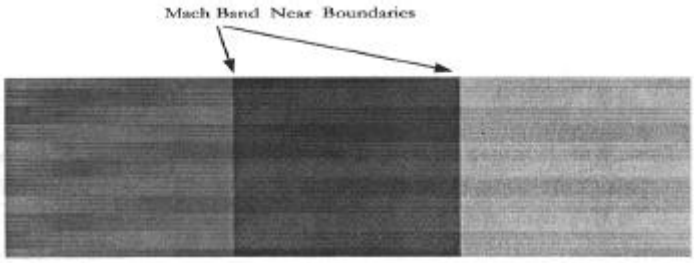


Fig. 4 Mach Band Effect

일반적으로 그래픽에서는 하나의 곡면을 여러 개의 평면 메쉬로 근사 시켜서 나타낸다. 이때, 메쉬 평면간에 경계선이 마하밴드효과로 인해 너무 뚜렷하게 노출된다는 것은 시각적으로 바람직하지 않다. 즉 평면 사이의 색 변화가 미세하여 느끼지 못하도록 부드럽게 이어가는 것이 가장 좋다. Fig. 5는 Flat Shading에 의해 그린 것이다. 메쉬 경계선에 뚜렸한 윤곽이 눈에 띄는 것을 볼 수 있다.

Fig. 5 Flat Shading

(Ⅱ) Gouraud Shading

Gouraud Shading에서는 하나의 메쉬면 내에서도 음영을 달리한다. 즉, 개별 메쉬면 안에서 음영이 부드럽게 변화하도록 하여 마치 하나의 메쉬면이 다시 여러개의 더 작은 메쉬들로 나뉘어져서 조명 모델이 가해진 것과 유사한 효과를 창출한다. 이는 선형보간(Linear Interpolation)을 사용하여 가능하다.

표면의 조명에 있어서 표면 수직벡터(Surface Normal Vector)는 매우 중요하다. 이것은 확산광(擴散光), 경면광(鏡面光) 등의 성분이 표면 수직벡터의 방향에 따라 좌우되기 때문이다.

Gouraud Shading에서는 먼저, 메쉬면들에 의해 둘러싸인 정점에서의 표면 수직벡터를 구한다. 이것은 면에 수직인 벡터를 그대로 이용하는 Flat Shading과는 다르다. 정점에 수직인 벡터는 그 정점을 둘러싼 4개의 수직벡터를 평균함으로써 구해진다.
Fig. 6에 보인 네 면의 수직벡터를 각각 Na, Nb, Nc, Nd라 하면 정점 "1"에서의 수직벡터는 Navg(average)의 방향을 갖는다. 수직벡터의 방향이 주어졌으므로 이것은 이 정점에서 광원의 방향, 눈의 방향을 감안하여 조명 모델에의 반사광의 세기를 구할 수 있다.

이때, 주어진 수직 벡터를 기준으로 경면광 성분의 크기를 구할 수도 있으나 일반적으로 Gouraud Shading에서는 경면광을 계산하지 않는다. 이는 정점"1"에서 평균적으로 근사된 수직벡터 방향이 일반적으로 실제 그 점에서의 수직벡터와 정확히 일치하지 않기 때문이다. 따라서 Gouraud Shading에서는 부정확한 고밀도광의 성분을 배제하고, 확산광과 주변광만을 계산해 주는 것이 일반적이다.

Gouraud Shading의 두 번째 단계는 화소 단위로 메쉬면을 칠하면서 적용된다. 이것은 Fig. 7과 같이 네 개의 정점에서의 반사광 크기를 기준으로 메쉬 내부에 있는 모든 화소의 밝기를 결정하는 단계이다. 실수 값으로 표시된 정점 위치는 래스터화(Rasterization, Scan Conversion)에 의해 그 정점에 가장 가까운 화소로 매핑된다. Fig. 7의 $I1, I2, I3, I4$는 네 개의 정점에서 계산된 화소의 반사광 세기(Intensity)를 말한다. 이 값은 각각 인접한 네 개 면의 수직벡터의 평균벡터를 기준으로 얻은 값이다.

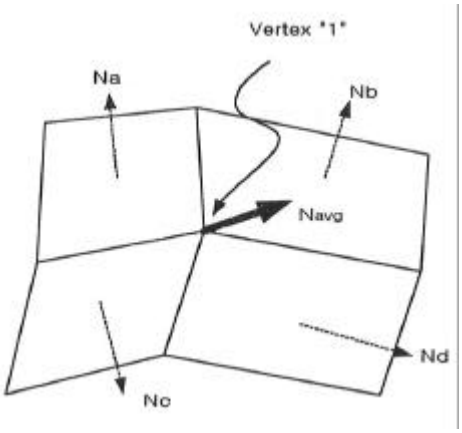


Fig. 6 The normal vector at the apex

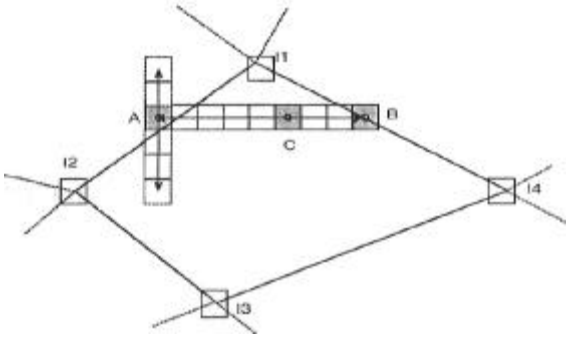


Fig. 7 Linear interpolation of Gouraud Shading

$I1, I2$ 를 알고 있을 때, 화소 A의 밝기는 $I1$ 화소와 $I2$ 화소의 선형 보간에 의해 구한다. 화소 사이의 거리를 화소 중심점 간의 거리라고 가정하고, 화소 A의 밝기를 그림에서 y축 방향으로 아래위로 표시된 화살표를 기준으로 설명하면 다음과 같다.

y축 방향으로 화소 A와 화소 $I1$ 의 거리를 2라고 할 때, 화소 A와 화소 $I2$ 의 거리는 3이 된다. 따라서 이것을 선형 보간하면 A의 반사광 세기는 $I1 \times \frac{3}{5} + I2 \times \frac{2}{5}$ 로 계산된다. 즉, 화소가 $I1$ 에 가까울수록 그 밝기는 $I1$ 과 유사해 진다. Fig 8은 $I1, I2$ 의 선형보간을 나타낸다.

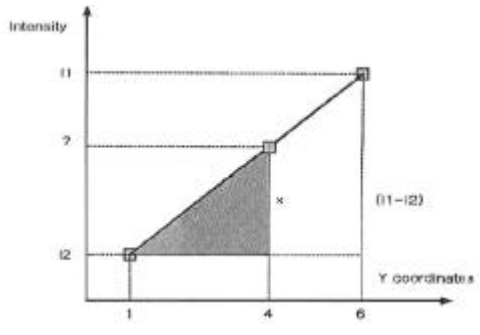


Fig. 8 Linear interpolation at the y axis direction

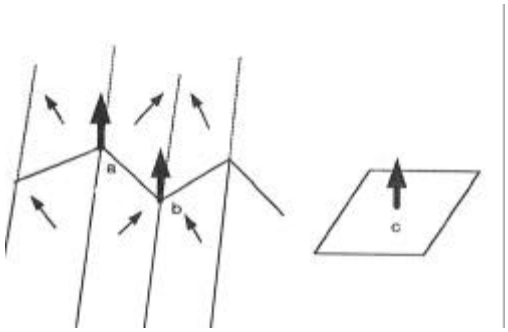


Fig. 9 Error of Gouraud Shading

보간 계산으로 인해 Gouraud Shading에 걸리는 시간은 Flat Shading에 비해 약간 더 걸리지만 더 나은 음영이 디스플레이될 수 있다. 그러나 이 방법에서는 확산광은 감안되지만, 경면광이 무시된다. 또 개별 메쉬면 안에서의 음영은 부드럽게 이어지나 메쉬간의 경계선에서는 여전히 마하 밴드가 나타난다는 단점을 안고 있다. Gouraud shading이 결정적으로 잘못된 결과를 초래할 때도 있다. Fig. 9와 같이 접힌 종이의 정점 a 혹은 b에서의 평균 수직 벡터 방향을 계산해 보면 c처럼 평평한 면에서의 수직 벡터 방향과 일치하게 된다. 이러한 결과로 정점 a, b에서의 Shading은 마치 평평한 곳처럼 칠해지는 오류를 범하게 된다.

그러나 일반적으로 Gouraud Shading은 Fig. 10처럼, Flat Shading보다는 훨씬 부드러운 음영을 그려낸다. 그러나 그림의 표면을 자세히 관찰하면 마하 밴드가 조금은 희미하게 나마 보이는 것을 알 수 있다.

Fig. 10 Gouraud Shading

(Ⅲ) Phong Shading

Bui-Tong Phong Shading에서는 수직 벡터를 보간한다. Gouraud Shading에서 정점의 반사광 세기를 먼저 계산하고 이를 보간 하는데 비해, Phong Shading에서는 반사광 세기 계산을 제일 마지막 단계로 미룬다. 즉, 이 방법에서는 먼저 정점에서의 수직 벡터를 보간해서 개별 화소에서의 수직 벡터를 계산한다. 화소별 수직 벡터를 계산하는 방법은 Gouraud Shading에서의 보간 방법과 동일하다. 예를 들어, 벡터 (2, 4, 6)과 벡터 (4, 4, 8)

을 보간하여 중간위치 화소의 벡터를 계산하면 (3, 4, 7)이 된다. 즉, 벡터를 x, y, z 성분별로 각각 보간하면 된다. 보간된 벡터 방향은 Fig. 11과 같다.

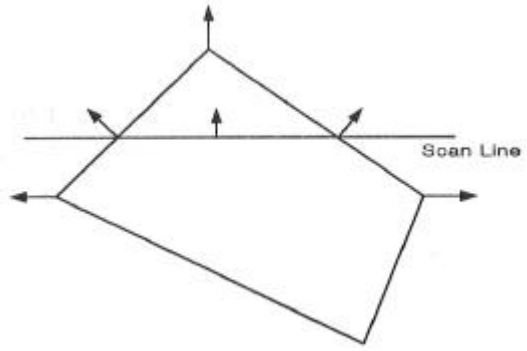


Fig. 11 Interpolation of normal vector

일반적으로 Phong Shading은 Gouraud Shading보다 정밀한 랜더링 결과를 보여준다. Fig. 12는 곡면을 몇 개의 평면 메쉬로 근사시킨 모습의 측면도이다. 여기서 곡면 $a-b-c-d$는 바깥 쪽 두 개의 평면인 $a-e$, $e-d$로 근사되어 있다. 메쉬 정점에서의 수직 벡터 계산에 의하여 a에서는 수직 벡터 Na가, e에서는 수직 벡터 Nc가 계산된다. 수직벡터는 면에 수직인 벡터이므로, 이 벡터의 방향은 f점에서 면이 서 있는 방향이 마치 g 모양으로 서 있는 것을 의미한다.

그런데 이런 g의 방향은 실제 곡면의 해당점인 b에서의 곡면의 방향과 일치한다. 결과적으로 보간에 의해 계산된 벡터 방향이 원래의 곡면이 서있는 방향을 복원하고 있음을 볼 수 있다. 모델링 시에 굴곡이 있는 곡면을 평면에 의해 근사 시키지만, 랜더링 단계에서 실제 곡면의 방향을 복원하여 반사광 세기를 계산함으로써 Phong Shading이 정밀도를 더하게 된다.

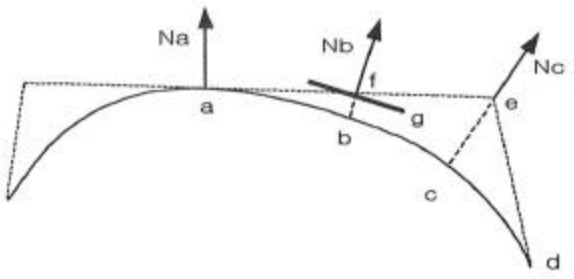


Fig. 12 The curve rate of Phong Shading

Phong Shading에 걸리는 시간은 대략적으로 Gouraud Shading의 3배 정도가 된다. 이것은 Gouraud Shading이 반사광 세기를 보간하는데 비해 Phong Shading은 벡터를 보간하기 때문이다. 반사광 세기가 하나의 값(Scalar)이라면 공간상의 벡터는 x , y , z 의 세 개의 성분으로 이루어지기 때문에 세 개의 성분 각각에 대해 보간을 가해야 하기 때문이다. 물론 Phong Shading에서도 해당 화소에서의 수직 벡터의 방향이 결정되면 최종적으로는 이를 바탕으로 조명모델에 의해 반사광의 세기를 계산해야만 한다. Phong Shading의 조명모델에서는 확산광, 주변광외에도 경면광을 계산할 수 있다. 이것은 Phong Shading에 의해 면이 서있는 방향이 정확히 평가될 수 있기 때문이다. Fig. 13은 Phong Shading의 결과이다. Gouraud Shading에 비해 첫째, 경면광에 의한 고밀도광 부분이 보이며, 둘째, 마하밴드 효과가 현저히 감소되었음을 보여준다. 이는 곡면의 복원에 의해 경계면이 부드러워졌기 때문이다.

Fig. 13 Phong Shading

(Ⅳ) Blinn Shading, Metal Shading

표면의 거칠기(Roughness)를 모델링하기 위한 방법으로 미세면 모델(Micro facet Model)이 있다. 이 모델에서 물체 면은 Fig. 14와 같이 서로 다른 방향의 오리엔테이션을 지닌 작은 미세면들로 구성되어 있다고 본다. 그러나 전체적으로 봐서는 평균적인 오리엔테이션을 지니고 있다고 간주한다. 이 경우의 반사광은 미세 표면의 거칠기를 통계적으로 처리해서 계산한다. Fig. 15에 보인 것 같이 표면의 거칠기가 증가할수록 경면광의 분포영역이 점차 넓어져서, 모든 방향으로 반사되는 확산광 성분처럼 작용한다.

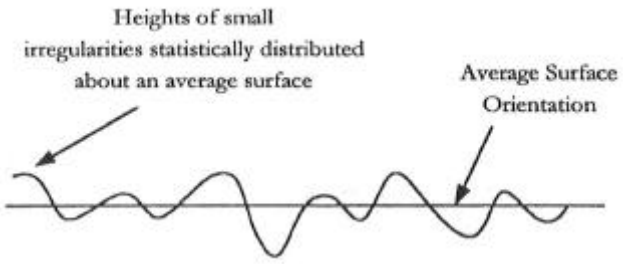


Fig. 14 Minute surface

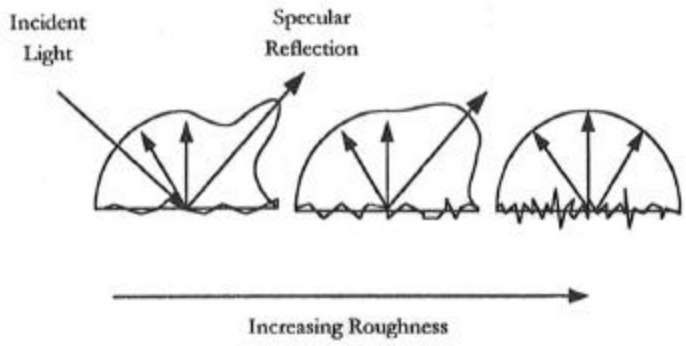


Fig. 15 Change of reflection light by the roughness

미세면 모델 중 대표적인 것으로 Blinn Mode과 쿡 토렌스에 의한 금속 모델(Metal Model, Cook/Torrence Model)이 있다. 미세면 모델에서는 표면의 거칠기라는 매개변수를

사용해서 미세면의 굴곡이나 모양을 제어 할 수 있다. 따라서 모델간의 차이점은 이 매개변수를 어떤 수학적인 모델로 표현하느냐에 따라서 달라진다. Blinn Model은 일반적으로 Phong Model과 유사한 결과를 보이지만 경면광 성분이 더욱 부드럽게 멀리 퍼져서 나타난다. 즉 경면광의 광원이 물체 면에 수직이 아니라 거의 물체 면과 나란히 예각으로 입사하였을 때의 결과에 가깝게 된다.

Phong Model의 경면광이 반짝이는 플라스틱 재질을 표현하기에는 좋으나 금속 표면의 은은한 경면광을 표현하기는 어렵다. 쿡 토렌스 모델은 금속 표면의 경면광 처리에 유리하다. 쿡 토렌스 모델에서 금속 표면의 광택은 광원의 색상이라기보다 금속 자체의 확산광 색상과 경면광 분포 곡선에 따라서 결정되도록 되어 있다.

(2) 텍스쳐 맵핑(Texture Mapping)

재질 혹은 질감을 의미하는 텍스쳐(Texture)를 물체 면에 부여 함으로써 현실감이 더해질 수 있다. 일반적으로 조명과 음영(Shading)만을 적용한 물체는 반질거리는 인상을 준다. 따라서 원래의 텍스쳐가 지닌 의미는 물체 면에 인위적으로 미세한 굴곡을 부여함으로써 주름면(Wrinkled Surface)을 형성하는 것이다.
그러나 이처럼 3차원적으로 주름진 면이 최종적으로 2차원 화면에 뿌려질 때는 명암에 의해 그 주름이 느껴질 수밖에 없다. 따라서, 아예 주름진 2차원 그림을 물체 면에 그대로 입힘으로써 재질을 느끼게 하는 작업까지도 텍스쳐의 범주에 넣는다. 즉, 물체 면을 3차원적으로 조각하여 재질을 느끼게 하는 것이 아니라, 주름진 3차원 물체에 스캐닝한 2차원 사진을 그대로 입히는 이론이다. 이와 같이 텍스쳐에는 여러 가지 이론적 측면이 있다. 다음은 이것에 대해 설명한다.

(Ⅰ) 범프맵 (Bump Map)

범프(Bump: 융기, 튀어나옴) 매핑(Mapping)에서는 물체 면에 실제로 굴곡을 부여한다. 3차원 곡면을 $S(u, v)$ 라는 매개변수 함수로 표시하고, 조정할 텍스쳐의 양을 $T(u, v)$ 라 하면, 범프 매핑 결과의 곡면은 $S(u, v) + T(u, v) \cdot N$ 으로 표시된다. 그 결과, 원래 곡면의 위치가 수직벡터(N)방향으로 $T(u, v)$ 만큼 높아지거나 낮아진다. 수직 벡터의 조절량인 $T(u, v)$ 에 랜덤 인수(Random Factor)를 도입하여 군데군데마다 조절량을 서로

다르게 하면 Fig. 17와 같은 범프 맵 효과를 얻을 수도 있다.

Fig. 16 Before the Bump map

Fig. 17 Appliance the Bump map

범프 맵에는 약간은 기교(Trick Play)에 가까운 방법들도 사용된다. 예를 들어 표면의 확산광 성분을 계산할 때, 확산계수(Diffusion Coefficient)를 부분적으로 바꿈으로써, 이것에 따른 명암 변화에 따라 굴곡처럼 보이게 할 수도 있다. 또한, 경면광 성분을 계산할 때 표면의 거칠기(Roughness)를 변화시켜 경면광 세기를 변화시킬 수도 있다.

변화된 재질을 나타내는 텍스쳐 함수 T 는 그것이 1차 함수인지 2차, 3차 함수인지에 따라 $T(u)$, $T(u, v)$, $T(u, v, w)$ 로 표시된다. 1차 함수는 가하고자 하는 재질을 곡선의 형태로 제시하는 것이며, 2차 함수는 하나의 평면 그림으로 제시한 것이다. 3차 함수는 재질 자체가 하나의 입체 모습으로 제시된 경우를 말한다. 그래픽에서 자주 쓰이는 것은 2차 텍스쳐 함수이다.

(Ⅱ) 2차원에서 3차원으로 매핑(2D to 3D Mapping)

텍스쳐가 2차원 그림으로 표시될 때 이를 3차원 물체에 입히는 작업은 주의를 요한다. 즉, 평면 벽지에 그려진 무늬를 3차원 원구에 입힐 때에는 어떤 식으로 입혀야 하는지가 문제가 된다. Fig. 18의 텍스쳐는 체크 무늬(Checkerboard Pattern)모습의 2차원 평면 그림이지만 이 그림이 입혀질 대상은 오른쪽의 3차원 곡면이다. 따라서 곡면의 모든 부분에 대해서 어떤 색이 되어야 하는지를 밝혀야 한다. 이 경우 평면이 곡면으로 바뀌어야 함은 물론, 둘 사이의 표면적 차이나 모양차이를 어떻게 처리해야 하는지가 문제가 된다. 이는 평면

지도를 가지고 원구형의 지구의(地球儀)를 완전히 감싸라는 주문과 같다고 볼 수 있다.

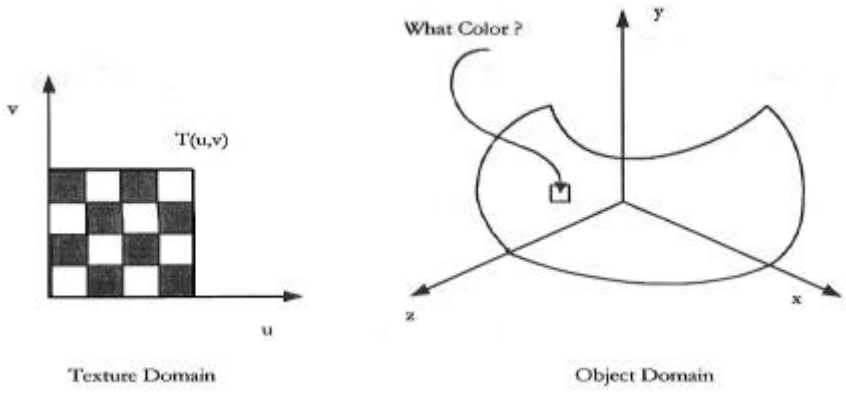


Fig. 18 The curved surface of plane texture

이 문제는 역으로 곡면을 평면에 매핑할 때의 어려움을 생각해 보아도 알 수 있다. Fig. 19은 평면 → 평면, 원통형 → 평면, 원구면 → 평면 매핑을 보여준다. 첫 경우는 어려움이 없다. 둘재 원통의 옆면은 펼치면 평면으로 되니 어려움이 없다. 그러나 세 번째 경우는 사실상 불가능하다. 원구를 펼 친 것이 4각형이 될 수는 없기 때문이다. 이런 어려움 때문에 지도책의 경우에는 메르카토르 도법을 사용하기도 한다. 그림의 잘라낸 부분에 해당하는 아래 위로 뾰족한 타원 모양 단위로 부분 평면도를 표시하되 여러 개의 평면도로 전체를 묘사하는 것이다. 어떤 방법을 사용하든 이런 경우에는 필연적으로 왜곡(Distortion)이 수반된다. 문제는 어떤 방법으로 이를 최소화하느냐는 것이다.

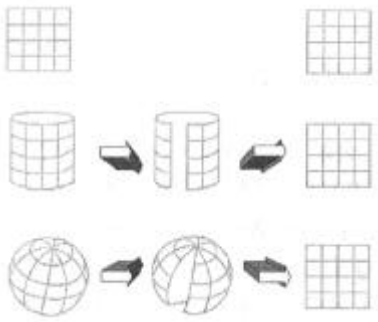
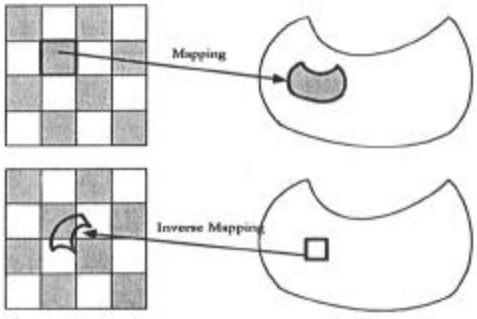


Fig. 19 The plane work of curved surface

곡면이 매개변수로 표현되었을 때의 해결책은 쉽다. 왜냐하면, 이 경우에는 텍스쳐

(Texture Domain)나 실제 물체(Object Domain)나 모두 매개변수 u, v ($0 \leq u, v \leq 1$)의 함수로 표시되어 있기 때문이다. 따라서 물체의 u, v가 하나의 화소를 덮을 크기가 될 때까지 세분하고, 그 u, v범위가 텍스쳐 도메인에서 어떤 색인지를 찾아서 그 색깔을 그 화소의 색으로 칠하면 되기 때문이다. 예를 들어, 물체 면에서 $0.5 \leq u \leq 0.6$, $0.3 \leq v \leq 0.4$가 하나의 화소영역에 해당하고 텍스쳐에서 이 범위가 검은색이라면 해당 화소를 검은색으로 칠하면 된다.

또한 텍스쳐 매핑에서는 항상 에일리아싱(Aliasing)문제를 고려해야 한다. Fig. 20에서 텍스쳐 도메인의 평면 사각형이 곡면에 매핑될 때에는 더 이상 사각형이 아니다. 모양 뿐만 아니라 면적도 확대되거나 축소되어야만 한다. 다시 말하면, 곡면 일부를 나타내는 정사각형 화소가 어떤 텍스쳐인지를 발견하기 위해 역 함수(逆, Inverse Mapping)를 가해보면, 텍스쳐 도메인에서는 사각형이 아니라는 것이다. 이때, 화소에 적용할 텍스쳐 값으로서 텍스쳐 도메인의 곡선 사변형의 한 가운데를 샘플링 하면 그 화소는 회색이지만, 구석에서 샘플링하면 흰색이 된다. 따라서 이 경우의 화소의 색깔을 결정하는 데에는 어떤 방법이든 앤티 에일리아싱(Anti-Aliasing)기법을 쓰는 것이 바람직하다.



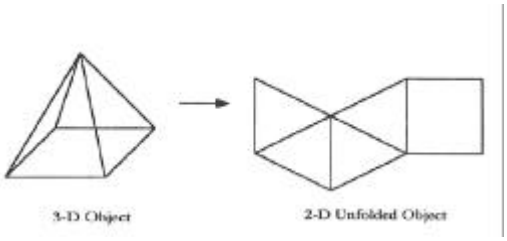


Fig. 20 Texture mapping and reverse mapping Fig. 21 3 dimensional object unfolding

곡면이 다각형 메쉬(Polygon Mesh)로 표현될 때에는 Fig. 21처럼 펼침(Unfolding)을 가하고 이를 다시 접을 수도 있다. 곡면에 직접 텍스쳐 패턴(Pattern)을 적용하는 것이 어려운 반면, 평면에 텍스쳐를 적용하는 것은 상대적으로 쉽기 때문이다. 그러나 이 방법은 모든

곡면이 이처럼 다각형 평면 메쉬로 펼칠 수 있는 것이 아니라는데 어려움이 있다. 이와 같은 경우 S 매핑(Mapping)과 O매핑(Mapping)을 사용하여 처리한다.

2.4 3차원 지형 모델링을 위한 방법

불규칙한 해안선을 가진 지형을 모델링 하기 위해서는 지형 자료 데이타베이스를 기반으로 하여 3차원 지형 모델링을 한다. 지형 자료 데이터베이스에는 우리가 손쉽게 접근할 수 있는 해도, 등고선이 나와 있는 항공 입체 사진, 인공위성에서 촬영한 지형 사진 등이 있다. 이런 지형 자료 데이터베이스를 근간으로 하여 3차원 지형 모델링을 하는데 프랙탈 지형 모델링(Fractal Terrain Modeling)의 원리를 이용하여 불규칙한 해안선을 모델링 하였다.

2.4.1 Fractal Terrain Modeling의 이론적 배경

컴퓨터 그래픽의 주요 관심사는 인간세계에서 볼 수 있는 여러 가지 형상들을 어떻게 닮은 형태로 잘 나타낼 수 있느냐에 있다. 또한 표현된 기본 모델을 이용하여 보다 응용된 모델로 나타낼 수 있도록 하여야 한다.[5]
그래픽 분야에서 그 동안 연구되어 온 과제는 직선이나, 완만한 선들로 표면을 구성하고 있는 물체들을 잘 표현하고 응용하는 방법에 관한 연구였다.

그러나 실세계에 분포되어 있는 자연현상들은 불규칙적이며 임의의 형태로 존재하고 있다. 이 문제를 해결하는데 최근에 많이 연구되고 있는 것이 Benoit Mandelbrot에 의해 제안된 프랙탈 그래픽 방법이다. 프랙탈 그래픽 상에서 문제시하고 있는 형태들은 산들의 형태, 나무들의 생성구조, 파도의 부딪히는 모양, 해안선의 모양, 빛의 운동, 분자나 원자들의 운동형태 등을 다루고 있다. 그런데 이들은 저마다 불규칙한 특성으로 인하여 자연스럽게 모델링하는 데에는 많은 어려움이 따른다.

예를 들어, 지도상의 해안선 구조를 생각해 보면, 해안선의 길이를 측량하는 데에는 측정자의 간격기준에 따라 정밀도가 달리 측정되어지는 것을 알 수 있다. 왜냐하면 해안선 구조는 매우 불규칙하게 형성되어 있으므로 측정자의 간격기준이 커질수록 정밀도가 떨어지기 때문이다.

이와 같은 자연의 미묘한 구조에 대해 Benoit Mandelbrot는 자연현상들의 변화된 모양이 전 단계의 모양과 닮은꼴의 축소형으로 변화하고 있는 점에 착안하여 불규칙속의 규칙적 요소를 잘 표현할 수 있는 프랙탈 개념을 만들어 냈다.
각각의 프랙탈 그래픽 영상들은 DNA와 같이 자기 유사성에 의한 복제능력이 뛰어난 특성을 가지고 있다.

프랙탈은 크게 두 분야로 나누어져 있는데 기하학적 프랙탈은 물체의 규칙성에 중점을 두어 동일하게 자기 모양이 축소되면서 변화한다. 그러나 이 기본적인 기하학적 표현으로는 무수히 변화하는 자연현상의 대상들을 정확하게 표현하는 데에는 한계가 있다. 그러므로 기하학적 프랙탈과는 원리가 비슷하나 부분적으로 균일하게 생성되는 형상들을 표현하는 데에는 임의적 프랙탈 개념을 응용해야 한다.

2.4.2 Fractal 기본이론

일반적으로 현실 세계에서 지형공간을 구성하고 있는 물체들을 인식하는 데에는 선, 다각형, 원뿔, 구와 같은 유클리드 기하학의 기본적인 형태를 가지고 정수형 차원을 사용하고 있다. 즉, 점, 선, 면인 경우에는 3차원으로 구분한다.

그러나 Hansdorff Besicovitch는 종래의 유클리드 기하학과는 다른 공간적 개념인 프랙탈 차원을 정의하였다. Table 1은 유클리드 기하학과 프랙탈 기하학의 특징이다.

Table. 1 The comparison of Euclidean and Fractal geometry

Euclidean geometry	Fractal geometry
1. 보통 간단한 공식에 의한 전개	1. 알고리즘 전개
2. 특정크기, 축척 등에 의존	2. 특정크기, 축척 등에 의존하지 않음
3. 인조물 해석에 적합	3. 자연현상 표시에 적합

Benoit Mandelbrot에 의해 발전된 프랙탈 기하학은 유클리드의 차원과는 대조적으로 자연의 물체는 비정수형으로 구성되어 있음을 시사하고 있다. 프랙탈 그래프의 곡선들은 제

안된 공간 안에서 그들이 가지고 있는 모습과 비슷하게 크기가 작아지면서 무한히 자기 모습을 복제해 가는 기본적인 특징이 있다. 그러므로 이와 같은 반복적인 특성은 컴퓨터의 표현 능력에 많은 영향을 받게 된다.

프랙탈의 차원을 정의하는 기본적인 개념은 다음과 같다.
주어진 공간 안에서 L 이라는 선분의 길이가 존재할 때 L 은 동일한 길이를 갖는 최소의 기본단위 S 로 N 등분되어 질 수 있다. 이것을 유클리드 기하학의 1 ,2, 3차원과의 관계를 비교하면 각각의 길이 $S = L / N$ 의 관계를 갖고 L 의 길이를 정규화 시키면 L 은 1이 된다. 그러므로 N 은 $1/S$ 의 비율에 따라 변화하게 된다. 1차원 상에서 S 가 감소하면 N 은 역비례 관계에 의해 증가한다. 즉, $1/S$ 를 1차원에서 변화의 기본 요소로 정의하면 2차원 상에서 N 의 비율은 $(1/S)^2$, 3차원 상에서는 $(1/S)^3$ 에 비례하여 증가하게 된다. 그러므로 이런 관계를 갖는 것을 프랙탈차원으로 정의하여 표현하면 임의의 차원 D 는 $N = (1/S)^D$ 의 형태를 갖게 된다. 이것을 대수적으로 이용하면 다음과 같은 차수를 구할 수 있다.

$$D = \log N / \log (1/S) \tag{1}$$

이 식을 보면 일반적으로 프랙탈차원 D 는 비정수형의 값을 갖는다는 것을 알 수 있다 . 그런데 앞으로 프랙탈은 자기 유사한 특성이 있으므로 그 개념을 여기다 도입하면 여러 번 반복 계산을 수행할 때마다 결국 S 가 감소하게 되므로 최종의 차원은 S 가 무한히 작아지는 경우가 된다.

$$D = \lim_{S \to 0} \log N / \log (1/S) \tag{2}$$

일반화된 프랙탈 차원 D 를 유도해 낼 수 있다. 1900년 초 H. Von Koch에 의해 만들어진 Triadic Koch Curve의 구조를 살펴보면 Fig 22에서와 같이 선분의 길이가 계속 감소하면 첫 번째 단계 (K_0) 에서 주어진 길이를 S_0 라 할 때 S_0 의 길이가 $1/3$ 으로 줄어들 선분의 수인 N 은 4개로 증가되고 S 의 길이는 $(1/3) \times S_0$ 가 되며 총길이는 $(4/3) \times S_0$ 가

된다.

Koch Curve의 일반식을 구해 보면 총 길이 K 는

$$K = \lim_{n \to \infty} K_n \tag{3}$$

여기서, K_n = n 차에서의 Triadic Koch Curve의 길이

이때, K_1 은 K_0 의 직선의 길이 S_0 에 $(4/3) \times S_0$ 의 크기만큼 늘어났으며 K_2 는 $(16/9) \times S_0$ 만큼 늘어난 것을 볼 수 있다. 횟수 n 이 무한히 증가하면 총길이 K 는 무한대로 증가하며 발산하게 된다. 프랙탈 차원으로 Triadic Koch Curve를 정의하면 D = log 4 / log 3 = 1.2619 차원이 된다.

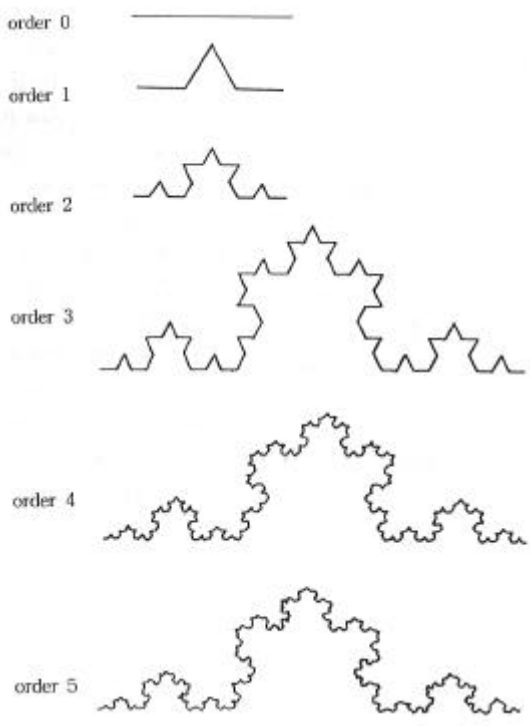


Fig. 22 Triadic Koch curves

여기에서 보듯이 K_0 에서 출발한 단순한 형태가 K_n 번째에는 복잡한 형상을 띄게 됨을

알 수 있다. 프랙탈 기하학의 기본원리는 원래 여러 번 계산한 경우 길이가 무한히 늘어나는 형태를 표현하는데 적합한 것을 의미한다. Fig 22에서 Koch Curve의 총길이는 프랙탈 차원이 증가함에 따라 무한히 증가하며, 형태도 자기복제에 의해 변화가 되며 표현이 되어 나가 복잡한 형식을 구성할 수 있음을 나타내 주고 있다. 또한 이 형태는 Hansdorff의 정의에 의해 실수형의 차원을 갖게 되어 평면 내에서의 각기 다른 형태에 대한 특성을 쉽게 표현되어 물체의 식별에 도움이 될 수가 있다.

프랙탈 곡선의 차원 D 는 1과 2사이의 값을 갖는다. $D = 1$ 이라는 의미는 곡선이 지극히 평탄한 직선이며 다항식으로 완전하게 접근할 수 있다. 따라서 모든 축척에서 길이는 일정하다. D 값이 1보다 크면 선분이 혼돈(fuzziness or uncertainty)을 포함하기 때문에 2차원 속성을 약간 갖는다. D 가 2일 때는 '혼돈'이 커져서 선분을 포함하는 면적이 선분의 궤적으로 완전히 메워져서 사실상 면적이 된다.

2.4.3 지형 모델링

불규칙한 해안선의 지형 모델링을 위해서는 위에서의 프랙탈 이론을 근간으로 하여 해안선의 형태를 생성하고 생성된 해안선을 바탕으로 지형을 모델링한다. 지형 모델링 방법에는 사각 그리드(Rectangle Grid)에 의한 방법과 해도 데이터에서 중요한 의미 점을 추출하여 불규칙 삼각망 형태로 구성된 TIN(Triangle Irregular Network)에 의한 방법으로 구분된다. 사각 그리드에 의한 방법은 해도 데이터에서 표현된 고도 값들에게 단순히 선형 보간법(Linear interpolation)을 적용하여 3차원 지형을 모델링하므로 빠르고 간단하며 평지 모델링에 적합하다. 그러나 데이터 양이 많고 굴곡이 심한 해안 지형의 특징을 정확히 표현하기 힘들다. 반면 TIN에 의한 방법은 해도 데이터로부터 지형적으로 의미있는 점들만을 추출함으로써 지형의 모습을 사실적으로 표현할 수 있어 사실적인 지형 모델링 및 랜더링에 많이 이용되고 있다. 본 논문에서는 해안선의 사실적인 묘사를 위해 TIN에 의한 방법을 사용하여 항만지형을 모델링 하였다.

2.5 3차원 모델링 과정(3 Dimensinal Modeling Procedure)

해상을 운항하는 선박의 조종 운동 특성을 실시간으로 재현하기 위해서는 3차원 모델을 기하학적이고 계층적으로 묘사하는 과정이 필요하다. 이와 같은 모델을 생성할 수 있는 3차원 모델 저작도구로서 MultiGen을 이용하여 모델을 생성하였다. MultiGen은 실시간 시뮬레이션이 요구하는 고유한 모델링 시스템인 OpenFlight 데이터 포맷을 사용하고 선박의 3차원 표면 모델링 및 특수효과 모델링을 처리하는데 있어서 효과적이다.

Fig. 23은 3차원 모델링 과정을 나타내고 있다. 항만의 3차원 모델링을 하는 경우 스캐너를 이용하여 해도를 스캐닝한 후 스캔된 Bitmap이미지를 해안선의 Vector 이미지를 추출하기 위해서 AutoCad Overlay를 사용하여 해안선 이미지를 구축한다. 이 과정에서 해도상의 Scale에 주의해야한다. 여기서 만들어진 해안선이미지를 3차원으로 지형 모델링 하기 위해 TIN(Triangle Irregular Network)에 의한 방법으로 지형의 사실적인 묘사를 하였다. 이렇게 고도를 가진 3차원 지형 모델을 3D Max를 이용하여 좀 더 사실적인 묘사를 하고 마지막 단계로 MultiGen을 이용하여 실시간 시뮬레이션을 재현할 수 있는 다양한 옵션을 이용하여 3차원 모델을 생성한다.[15] 선박의 모델링이나 다른 구조물(크레인, 부표, 등등)도 이와 같은 방법을 통하여 생성할 수 있다.

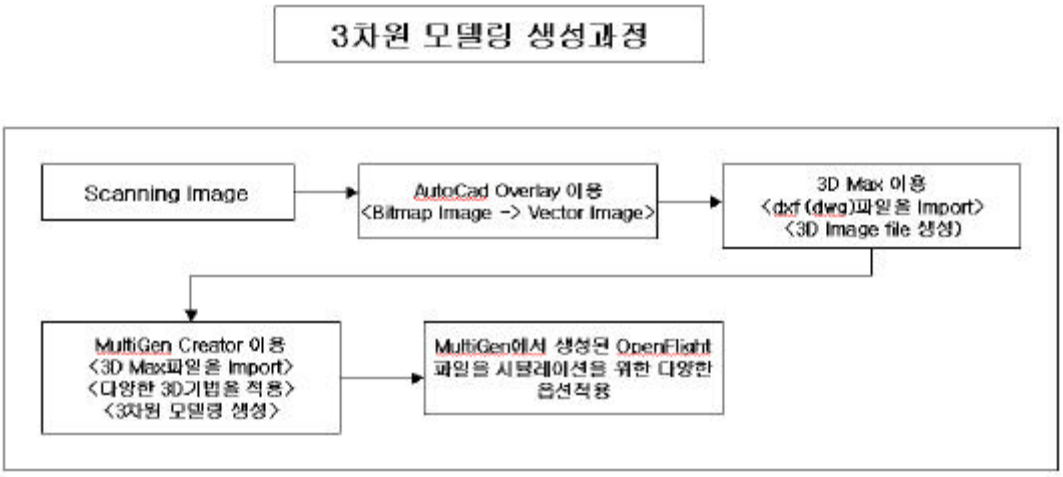


Fig. 23 3 Dimensional Modeling Procedure

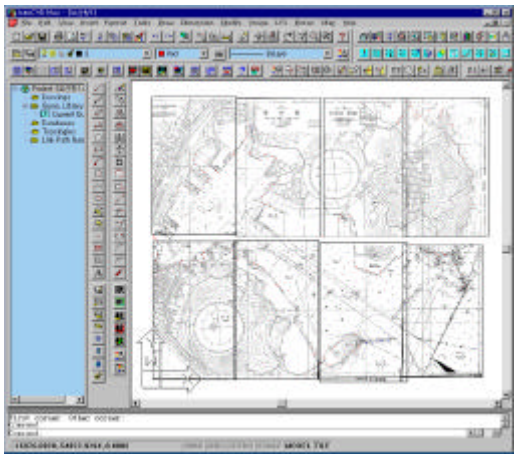


Fig. 24 Pusan Port Scanning Image

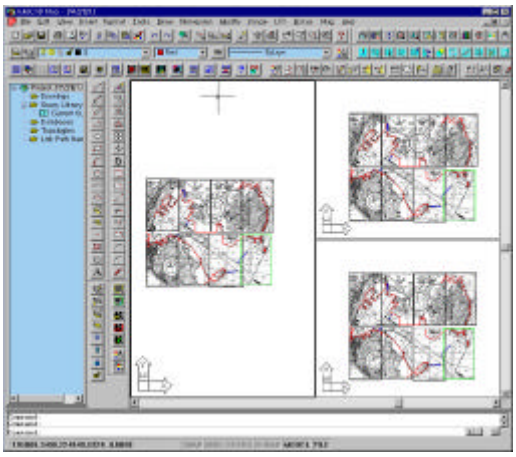


Fig. 25 Vector image

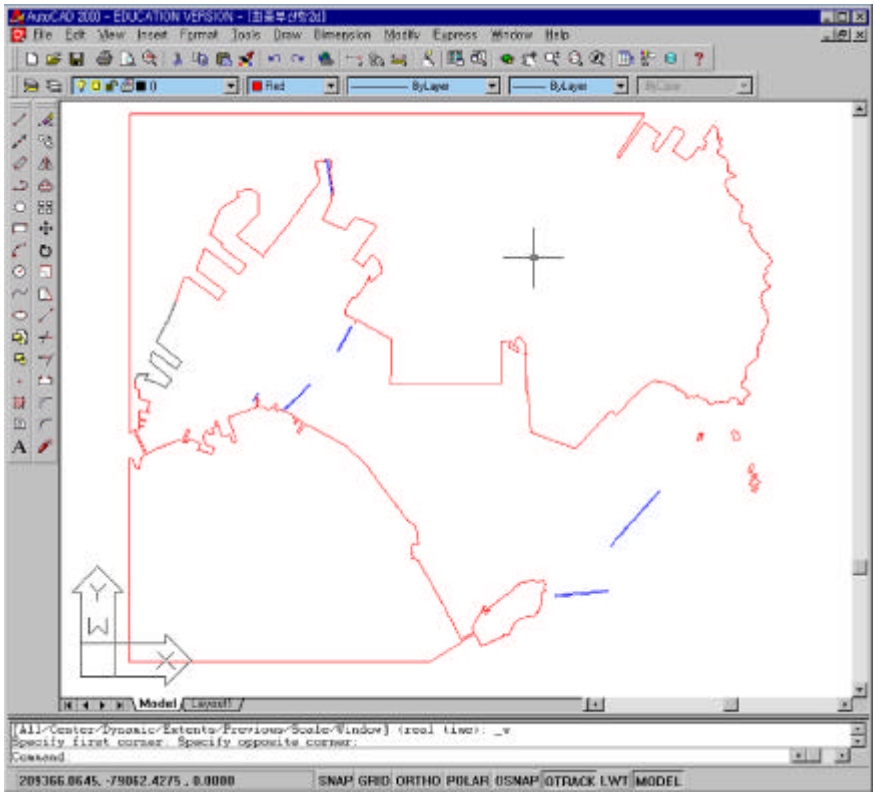


Fig. 26 Pusan Port Coastal Line

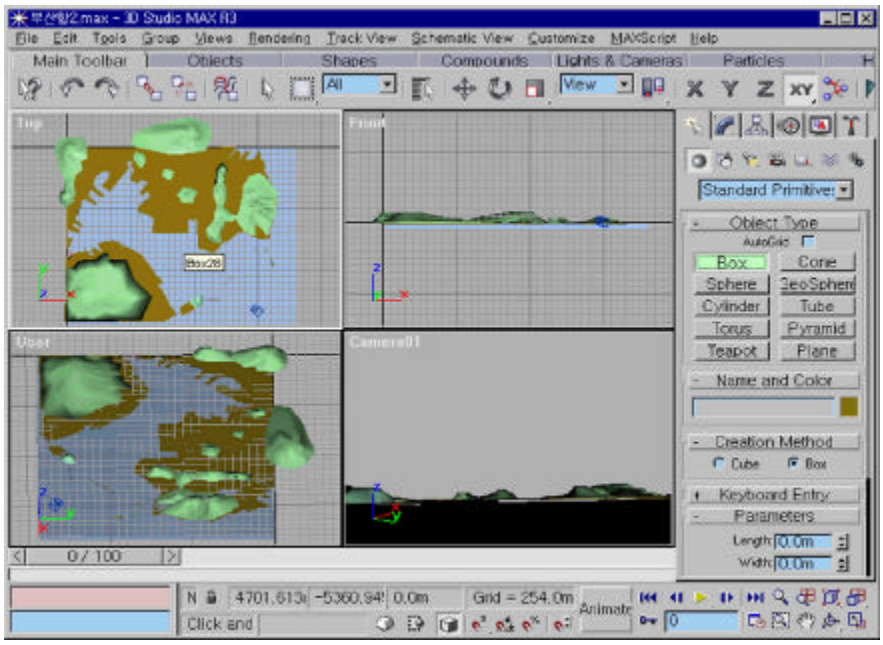


Fig. 27 3 Dimensional Image Making

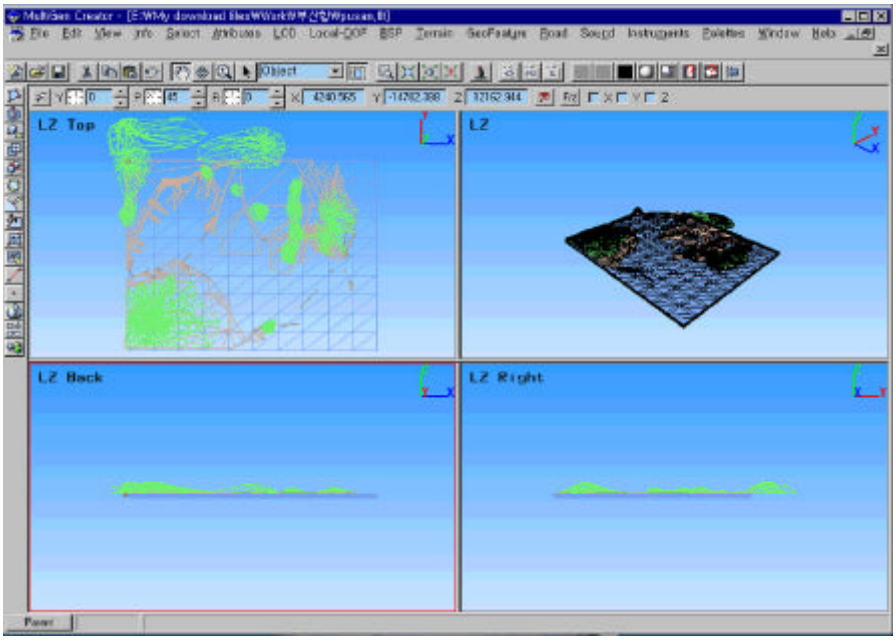


Fig. 28 3 Dimensional Image by the Multigen

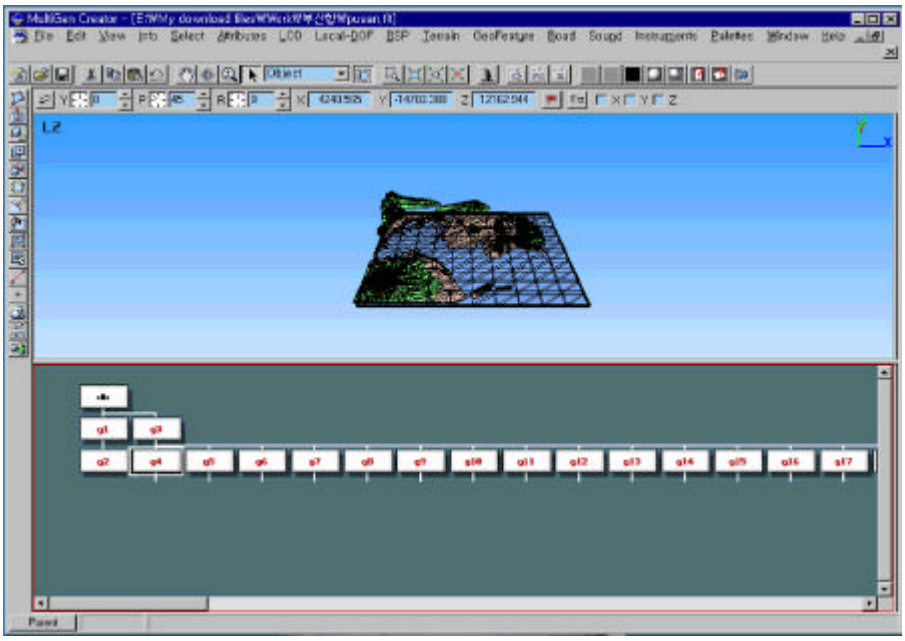


Fig. 29 3 Dimensional Image Database

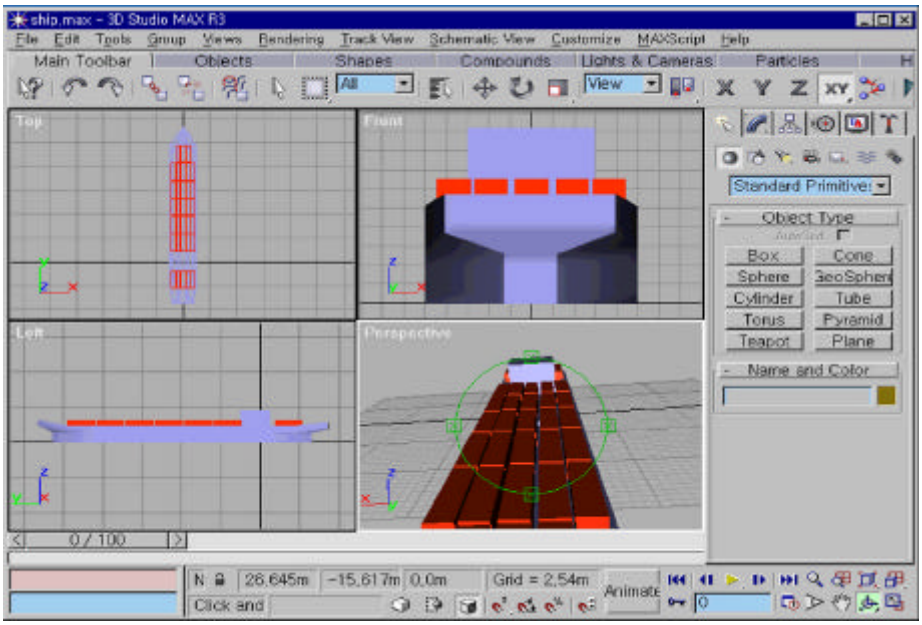


Fig. 30 3 Dimensional Ship Modeling

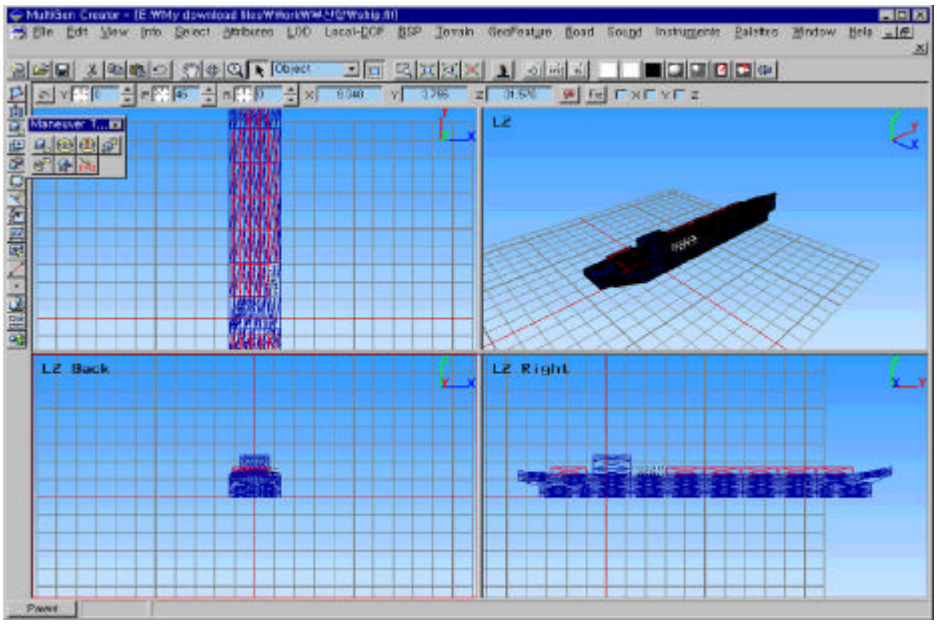


Fig. 31 3 Dimensional ship Image by the Multigen

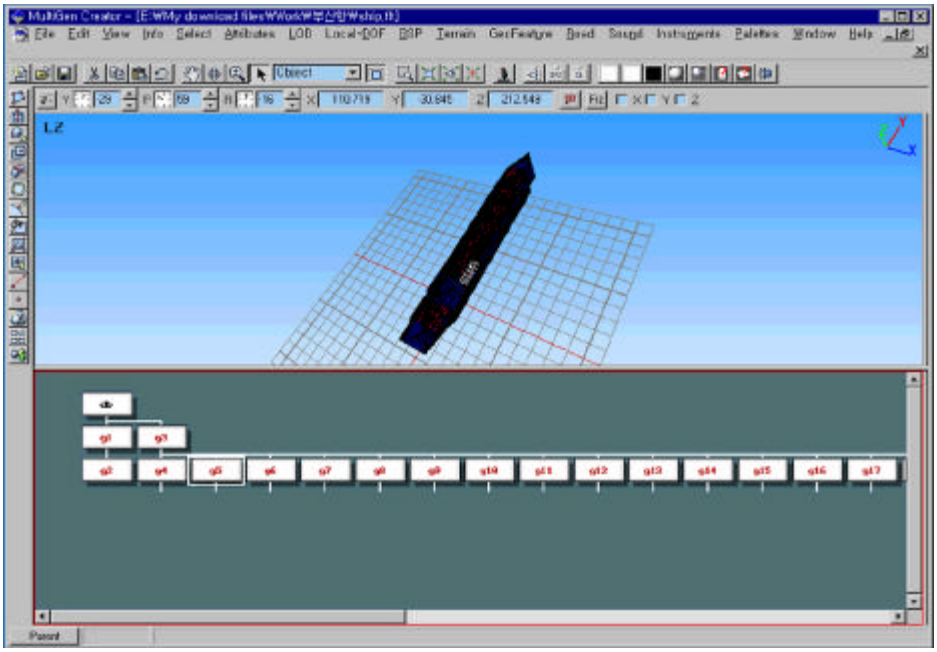


Fig. 32 Ship Modeling Database

2.6 분산처리기법 (Distributed Processing)

선박 조종 시뮬레이터를 구현하는데 있어서 구성부분간의 통신이 중요하게 된다. 이것에 대한 처리 방법으로 네트워크(Network)를 이용하여 서로 다른 구성부분간의 통신을 하였다. 다음은 네트워크를 이용하여 통신하는 방법에 대해 설명한다.[6]

2.6.1 TCP/IP

TCP(Transmission Control Protocol)/IP(Internet Protocol)프로토콜 그룹은 이 기종 컴퓨터와 서로 다른 운영체제(OS)간의 통신을 가능하게 한다. 이 통신 형식이 널리 이용되게 된 배경에는 프로토콜 그룹에 대한 정의와 구현이 무료로 제공되기 때문이다. 그래서, 이것을 개방 시스템이라고 한다. 하지만, 부정적인 측면으로 1970년에 제정된 국제 통신 표준인 OSI 7 계층 통신망 구조와 여러 면에서 일치하지 않고, 프로토콜 처리를 위한 오버헤드가 크기 때문에 널리 이용되지 않을 것이라 생각되었다. 하지만, 1993년 이후 TCP/IP를 기반으로 한 WWW(World Wide Web)서비스가 확산되면서 TCP/IP는 컴퓨터 통신망의 표준이 되었다. 또한, 논리적인 주소 할당을 할 수 있고 규모가 큰 네트워크에서도 개별 호스트가 각각의 고유 주소를 가질 수 있다. 이러한 논리적인 주소는 고정된 주소가 아니므로 호스트의 이동 및 주소 재할당, 네트워크 소속 변경 등이 용이하다.

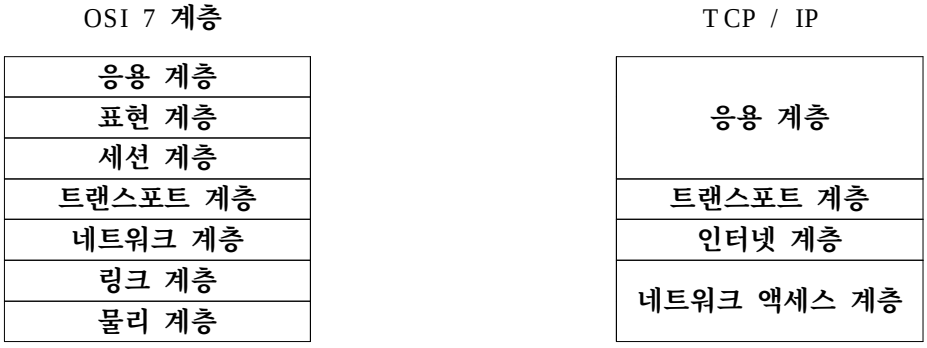


Fig. 33 Comparison layer protocol structure TCP/IP and OSI 7

1) TCP/IP의 각 계층에 따른 역할

① 네트워크 액세스 계층(Network Access Layer)

데이터 링크층 또는 네트워크 인터페이스 층이라고 하는데, 일반적으로 컴퓨터에 들어오는 네트워크 인터페이스 카드를 포함한다. 프레임을 물리적인 회선에 올리거나 내려받는 역할을 한다. 보통 전송매체의 물리적 인터페이스와 관련된 하드웨어 부분을 제어한다. OSI(Open System Interconnection)7 계층의 Physical layer(물리계층)는 LAN과 WAN으로 나누어진다.

② 인터넷 계층(Internet layer)

네트워크 상의 패킷 이동을 제어한다. 패킷 라우팅(routing:경로배정)등을 수행한다. 패킷이 목적지에 제대로 도착되었는지, 데이터의 손상 여부에 대해서는 상위 계층에서 처리한다. 예를 들어, ICMP, IP, ARP가 이 계층에 속한다.

③ 트랜스포트 계층(Transport layer)

상위 응용층에 대해 두 호스트간의 데이터 흐름을 제공한다. TCP/IP 프로토콜에는 TCP와 UDP(User Datagram Protocol)라는 2개의 트랜스포트 프로토콜이 있다. 여기서, TCP는 두 호스트간에 신뢰성 높은 데이터 흐름을 유지하는 기능을 하고, 반면, UDP는 데이터그램이라는 데이터 패킷을 한 호스트에서 다른 호스트로 전송하지만, 데이터그램이 목적지까지 가는 동안의 신뢰성은 제공하지 않는다. 보통 대용량 전송은 TCP를 사용하고, 소규모 데이터 전송은 UDP를 사용한다.

④ 응용계층(Application layer)

네크워크를 이용하여 서비스를 제공하는 계층이다. 웹브라우저에서 http(http 외에 Telnet, Gopher, FTP , SMTP, SNTP등)가 인터넷 기반의 유틸리티로서 winsock을 통해 하위 프로토콜로 전달된다. 응용계층(Application)은 사용자 응용 프로그램으로부터 요청을 받아들여서 이를 적절한 메시지로 변환하고 하위 계층으로 전달한다.

2.6.2 UDP(User Datagram Protocol)

UDP는 간단한 데이터그램-지향(datagram-oriented) 트랜스포트 계층 프로토콜이다. 프로세스의 각 출력 작업은 반드시 하나의 UDP 데이터그램을 생성하고, 하나의 IP 데이터그램으로 전송된다. 이것은 데이터 양이 연관이 없는 TCP와 같은 스트림-지향(Stream-Oriented) 프로토콜과 다른점이다. UDP는 신뢰성을 제공하지 않는다. 그러므로, 응용은 최종적인 IP 데이터그램의 크기에 대해서 주의할 필요가 있다. 하지만 데이터 패킷(Data Packet)을 빠른 속도로 보내기 때문에 실시간 시뮬레이션을 위해서는 속도를 보장할 수 있는 장점이 있다.

2.6.3 소켓(Socket)

소켓은 TCP/IP를 이용하는 API로서 1982년 BSD(Berkeley Software Distribution) 유닉스 4.1버전에서 소개된 이후 유닉스 계열의 운영체제에서 기본적으로 제공하고 있다. 윈속, 자바 소켓 등도 TCP/IP를 이용하기 위한 API를 제공하는데 모두 BSD 소켓과의 호환성을 갖도록 만들어 졌다. 이러한, API를 다른 시스템에서 사용하면 물론 호환이 되지 않는다. 하지만, 이 프로그램들이 IP 패킷을 서로 주고 받는데는 아무런 문제가 없으며 이것은 각 소켓 API들이 모두 TCP/IP라는 같은 통신 프로토콜을 사용하기 때문이다. 소켓은 응용 프로그램에서 TCP/IP 계층을 이용하는 창구 역할을 하고 있으며 응용 프로그램과 소켓 사이의 인터페이스를 소켓 인터페이스라고 한다.

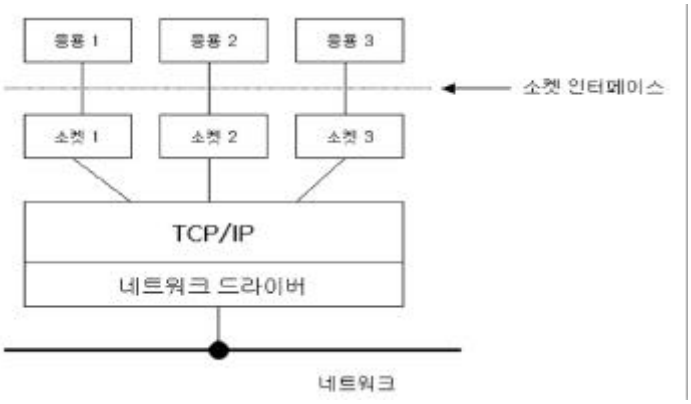


Fig. 34 Socket interface

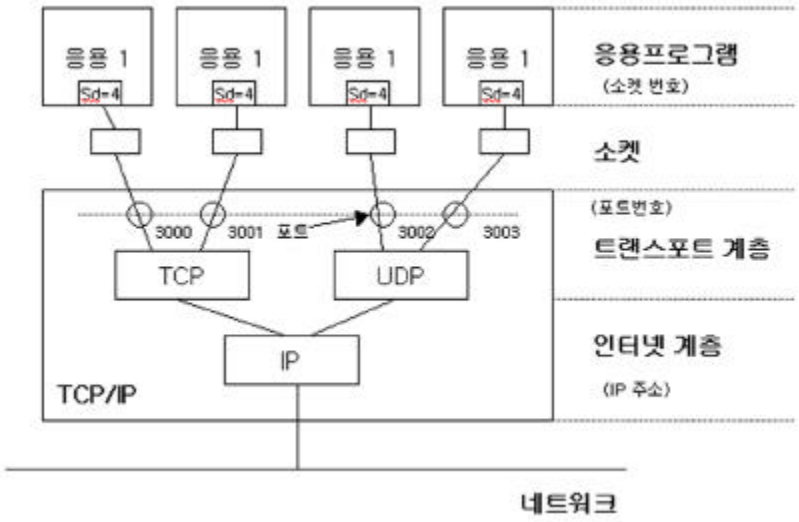


Fig. 35 A concern application and socket and TCP/IP

3. 출입항 모의를 위한 선박조종시뮬레이터의 구성 및 시뮬레이션의 예

3.1 항만내에서의 선박조종운동의 수학모델

1) 조종운동 수학모델의 정식화

일반적으로 선박의 조종 운동 방정식은 전후동요(surge), 좌우동요(sway), 선수동요(yaw)의 복합운동으로 표현 할 수 있다. Fig. 36과 같이 선체중앙(midship)에 좌표 원점을 둔 동좌표계를 $O-xyz$ 라 할 때, 평수중에서의 조종 운동의 기본 방정식은 다음과 같이 표현 할수 있다.[7]

$$\begin{aligned} m(\dot{u} - vr - x_G r^2) &= X \\ m(\dot{v} + ur + x_G \dot{r}) &= Y \\ I_{zz}\dot{r} + mx_G(\dot{v} + ur) &= N \end{aligned} \tag{4}$$

식에서 m 은 선체의 질량이고, I_{zz} 는 z 축 주위의 관성모멘트이다. u, v, r 은 각각 선체 중앙에서의 x 축 방향의 속도 성분, y 축 방향의 속도 성분, z 축 주위의 각속도 성분이다. $\dot{u}$, $\dot{v}$, $\dot{r}$ 는 각각 u, v, r 의 시간에 대한 1차 도함수이고, x_G 는 무게중심의 x 좌표이다. X, Y, N 은 각각 x, y 축 방향 유체력과, z 축 주위의 유체력 모멘트이다.

식(4)의 유체력중 부가질량력[8]을 좌변에 고려하고, 우변에는 부가질량력을 제외한 유체력으로서 MMG 모델[7]을 적용하면 조종 운동 수학모델은 다음과 같이 표시된다. 단, 관례에 따라 Munk 모멘트 등 일부 부가질량력은 우변의 유체력항에 포함시킨다.

$$\begin{aligned} (m + m_x)\dot{u} - (m + m_y)vr - (mx_G + m_y\alpha)r^2 &= X_H + X_P + X_R + X_T + X_W \\ (m + m_y)\dot{v} + (m + m_x)ur + (mx_G + m_y\alpha)\dot{r} &= Y_H + Y_P + Y_R + Y_T + Y_W \\ (I_{zz} + J_{zz})\dot{r} + (mx_G + m_y\alpha)\dot{v} + mx_G ur &= N_H + N_P + N_R + N_T + N_W \end{aligned} \tag{5}$$

여기서 첨자 H, P, R, T, W 는 각각 주선체, 프로펠러, 타, 예선 및 바람에 기인하는 유체력 또는 외력성분을 나타낸다. 그리고 m_x, m_y 는 x, y 축 방향의 부가질량, J_{zz} 는 z 축 방향의 부가관성모멘트, α 는 m_y 중심의 x 좌표를 의미한다.

한편, 조류에 기인하는 유체력은 별도의 항으로써 고려할 필요없이, 식(5)의 유체력 계산에서 운동 변수로서 상대유속 즉, 대수속도를 사용함으로써 조류력 영향이 포함될 수 있다. 조류의 유속 V_c 와 유향각 ψ_c 를 Fig. 36과 같이 정의할 때 대수속도와 대지속도와의 관계는 다음과 같다.

$$\begin{aligned} u &= u^* + V_c \cos(\psi_c - \psi) \\ v &= v^* + V_c \sin(\psi_c - \psi) \\ \dot{u} &= \dot{u}^* + V_c r \sin(\psi_c - \psi) \\ \dot{v} &= \dot{v}^* - V_c r \cos(\psi_c - \psi) \end{aligned} \tag{6}$$

여기서 u^*, v^*, $\dot{u}^*$, $\dot{v}^*$ 는 대지속도와 가속도를 u, v, $\dot{u}$, $\dot{v}$ 는 대수속도와 가속도를 나타낸다. 식(6)의 관계를 식(5)에 대입하면 다음과 같다.

$$\begin{aligned} (m + m_x)\dot{u}^* &= (m + m_y) vr + (m x_G + m_y \alpha) r^2 - (m + m_x) V_c r \\ &\quad \times \sin(\psi_c - \psi) + X_H + X_P + X_R + X_T + X_W \\ (m + m_y)\dot{v}^* + (m x_G + m_y \alpha)\dot{r} &= -(m + m_x) ur + (m + m_y) V_c r \\ &\quad \times \cos(\psi_c - \psi) + Y_H + Y_P + Y_R + Y_T + Y_W \\ (I_{zz} + J_{zz})\dot{r} + (m x_G + m_y \alpha)\dot{v}^* &= -m x_G ur + (m x_G + m_y \alpha) V_c r \\ &\quad \times \cos(\psi_c - \psi) + N_H + N_P + N_R + N_T + N_W \end{aligned} \tag{7}$$

식(7)로부터 축차근사적분법을 이용하여 대지속도 u^*, v^* 와 회두각속도 r 을 구한 후에, 다시 적분을 수행함으로써 선위(船位)와 회두각을 구할 수 있다.[9]

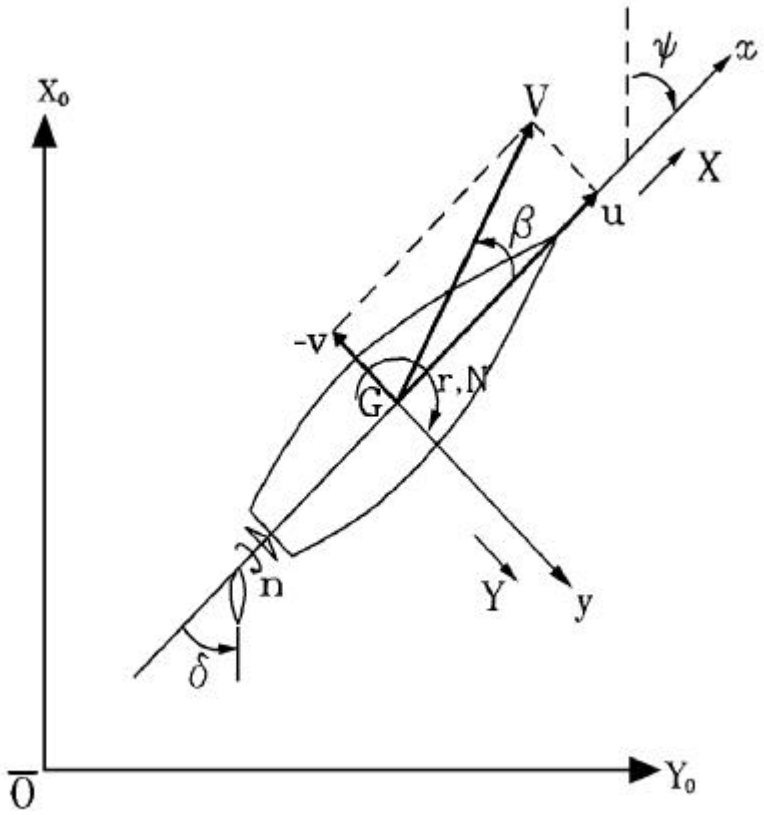


Fig. 36 Coordinate System

3.2 선박조종시뮬레이터의 구성

본 시뮬레이터는 Fig. 37에 나타낸 것과 같이 제어입력부(Contorl Panel), 운동계산부(Motion Solver), 정보출력부(Information Display), 3차원 시계재현부(3D Visual), 2차원조감도(2D Map Display)의 5 구성요소로 되어 있고 각 요소에는 각각 컴퓨터를 한대씩 배치하여 이들을 분산처리방식으로 연결하였다.

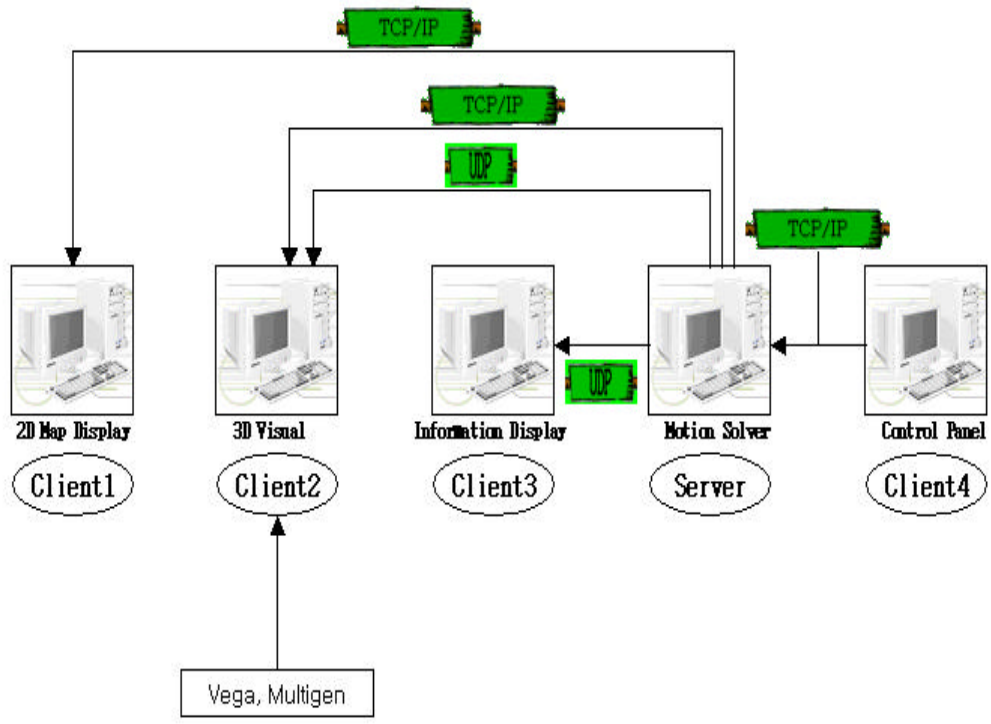


Fig. 37 Ship Handling Simulator Layout

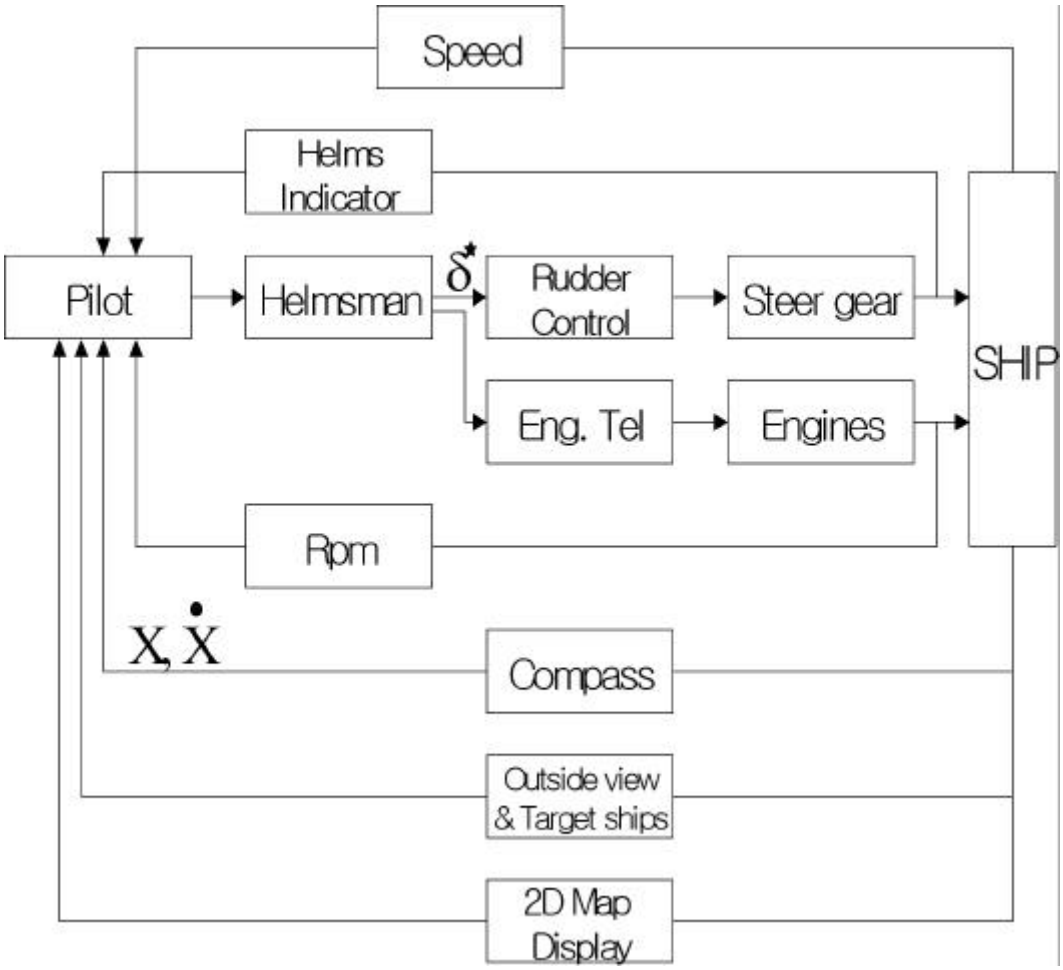


Fig. 38 Ship Handling Simulator diagram

(1) **운동계산부** (Motion Solver)

운동계산부는 자선과 예선 및 환경자료를 기반으로 하여 조종운동 수학모델을 수치계산하여 얻은 운동의 결과를 3**차원 시계재현부**(3D Visual)**와 정보출력부**(Information Display)**에 소켓**(Socket)**을 이용하여 자료를 보낸다. 운동계산에 있어서 자선의 주요목, 유체력미계수, 조류, 바람 등의 자료들은 중요한 요소로 작용한다. 또한 운동계산부에서는 제어입력부로부터** RPM, **타각**(Rudder Angle), **선박의 자료, 시계환경의 데이터를** TCP/IP**로 받아서** 3

차원 시계재현부의 기본 환경을 설정할 수 있다. 이와 같이 자료들을 효율적으로 관리하고 검색하기 위해서는 데이터베이스화하여 관리하면 체계적이고 계층적으로 이용할 수 있다. 특히 컴퓨터의 처리능력이 제한이 있는 경우라면 데이터베이스의 계층화하여 관리하여야 할 것이다. Fig. 38은 자선의 주요목, Fig 39는 유체력 미계수, Fig 40은 Isherwood의 풍압력 계수를 데이터베이스 하는 것을 보여준다.[12],[13] 데이터베이스는 Borland Delphi 5.0의 데이터베이스 엔진인 Paradox7을 이용하여 데이터베이스화하였다. 또한 조종운동의 수학모델을 프로그램밍하는데 있어서 Visual C++ 6.0을 사용하였다. [17]

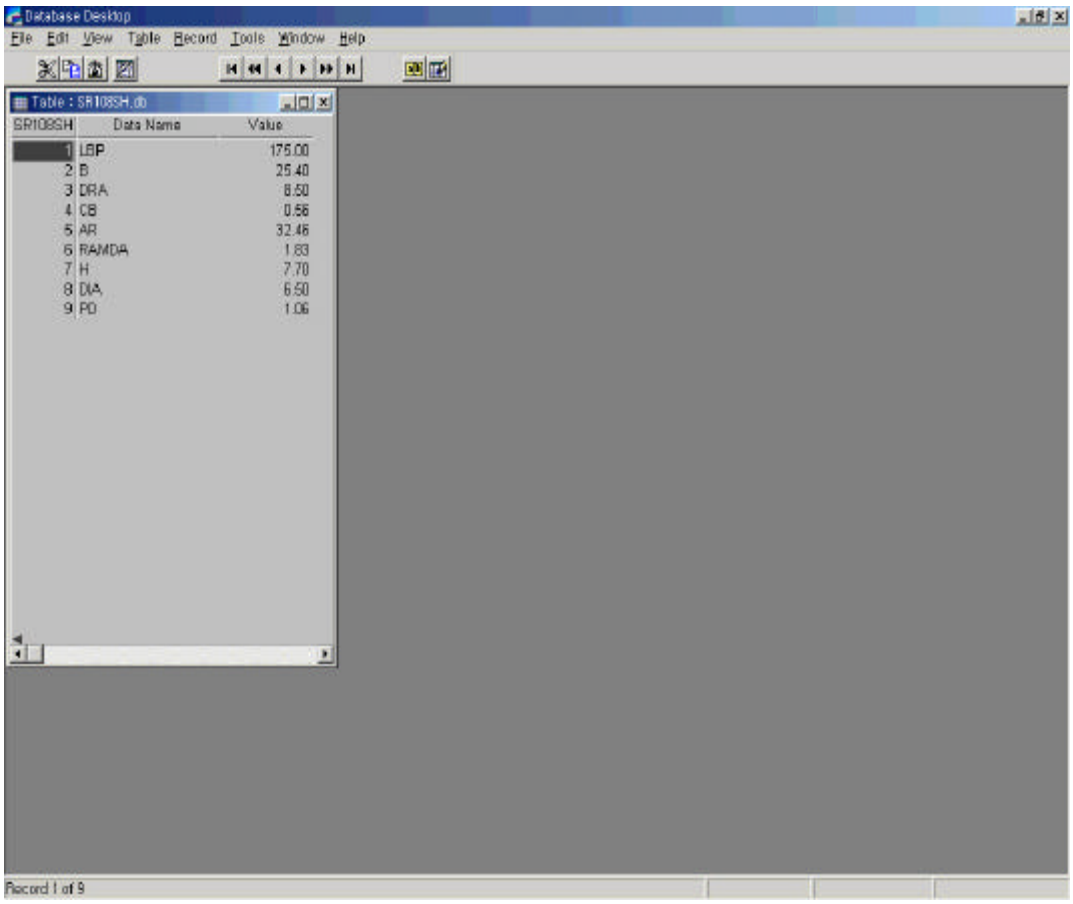


Fig. 39 Ownship Principal Database

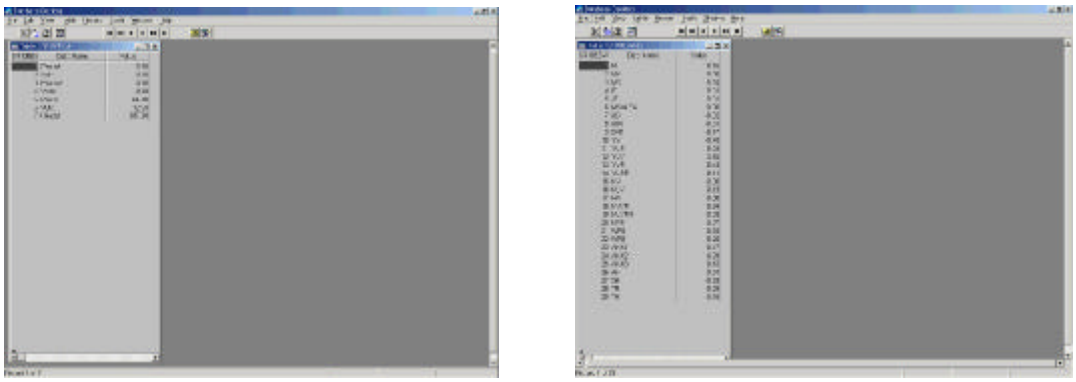


Fig. 40 SR 108 container ship coefficients database

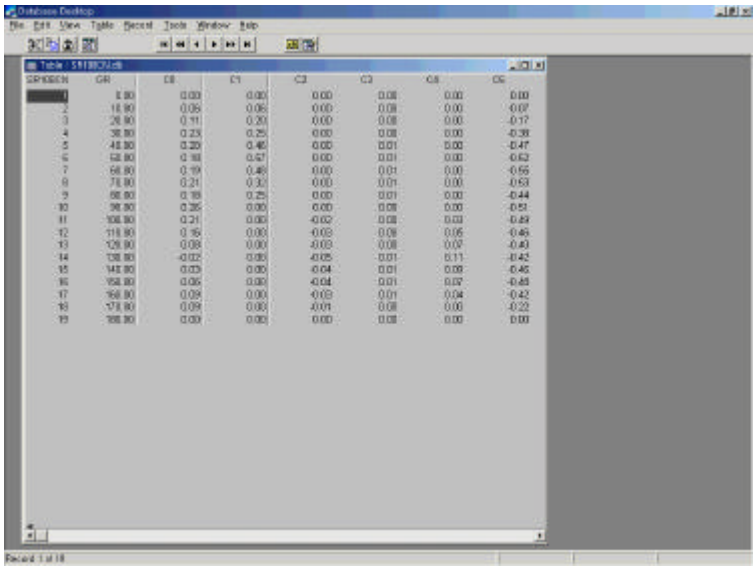


Fig. 41 Isherwood wind pressure coefficients Database

(2) **정보 출력부**(Information Display)

정보출력부는 운동계산부에서 계산된 결과 자료를 UDP **소켓**(Socket)**을 통해서 받는다.** **운동계산부에서는** UDP(User Datagram Protocol)**을 사용하여 자료를 보낸다. 정보 출력부**

에서는 빠른 속도로 계산되는 운동계산부의 자료를 받을 필요가 있기 때문에 UDP 소켓을 사용하여 데이터 통신을 하였다. UDP 소켓으로부터 받은 데이터를 이용하여 각종 정보를 디스플레이하여 항해자가 정보 출력부의 정보를 이용하여 선박을 조종하는데 유용하게 이용할 수 있다. 정보 출력부에는 자선의 침로각(Heading angle), 바람의 방향과 속도, 엔진의 Rpm이나 타각(Rudder Angle) 등의 정보를 보여준다. Fig. 41은 정보출력부의 각 부분에 대해 나타내고 있다.

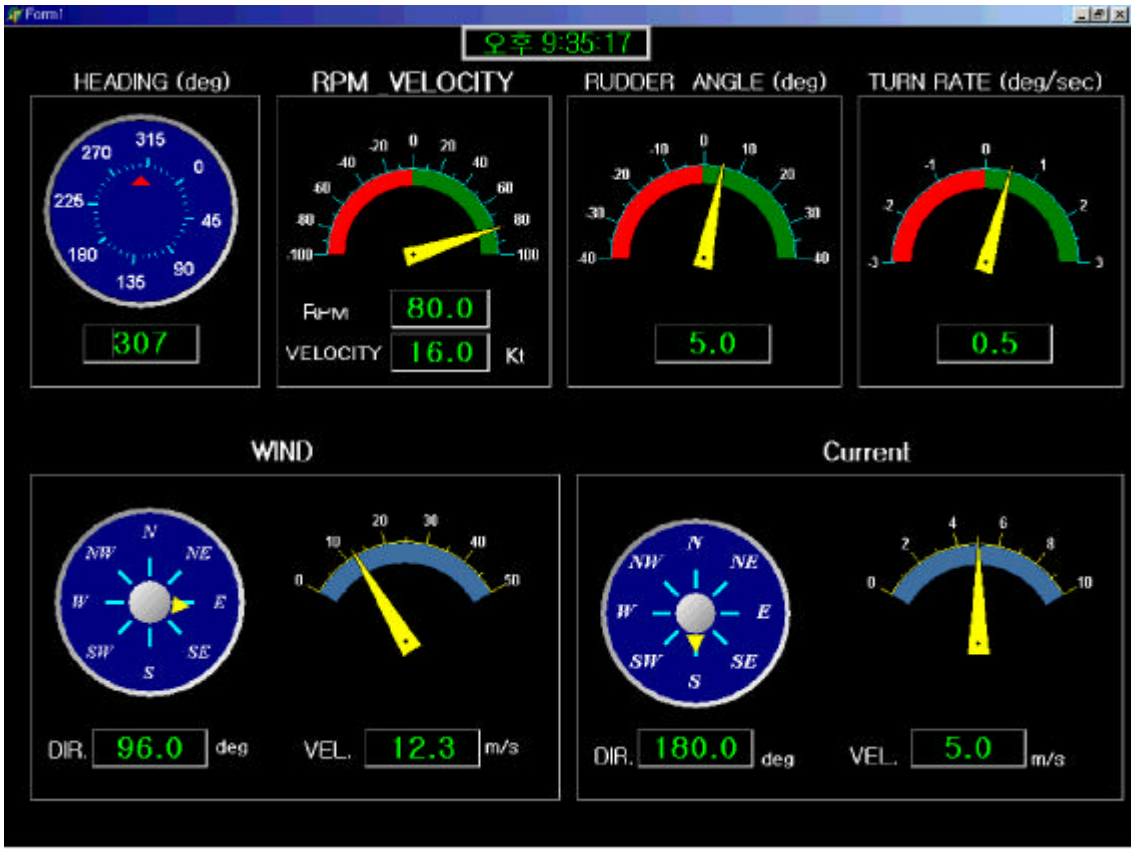


Fig. 42 Information Display

(3) 제어 입력부(Control Panel)

제어 입력부는 선박 조종에 필요한 엔진 RPM과 타각, 예선과 시뮬레이션에 필요한 주야의 조작, 자선의 선택, 타선의 선택과 위치를 제어 할 수 있는 부분으로 구성 되어 있다. 제어

입력부와 3차원 시계재현부는 운동계산부를 통해서 TCP/IP로 연결되어 데이터 통신을 할 수 있도록 설계하였다. 운동계산부에는 엔진 RPM과 타각, 예선등의 자료를 받아서 선박 조종시 필요한 명령을 입력받고 3차원 시계재현부에서는 자선의 선택, 자선의 주요목, 시계의 환경, 주야의 선택등을 제어입력부의 TCP/IP 소켓으로부터 데이터를 받아서 시계를 구성하도록 되어 있다.[14] 또한 제어 입력부에서는 바람과 Current를 조작할 수 있는 Panel을 구성하여 제어 할 수 있도록 하였다. Fig. 42, Fig. 43, Fig. 44, Fig 45, Fig 46, Fig. 47는 제어 입력부의 구성부분에 대해 나타내고 있다.

Fig. 43 Control Panel Initial

Fig 44 Own Ship Select Panel

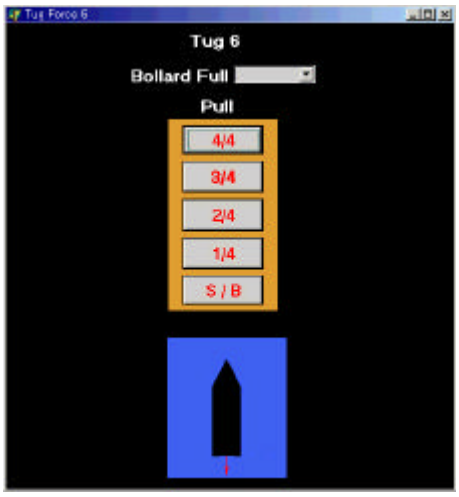


Fig. 45 Tug Force Input Panel

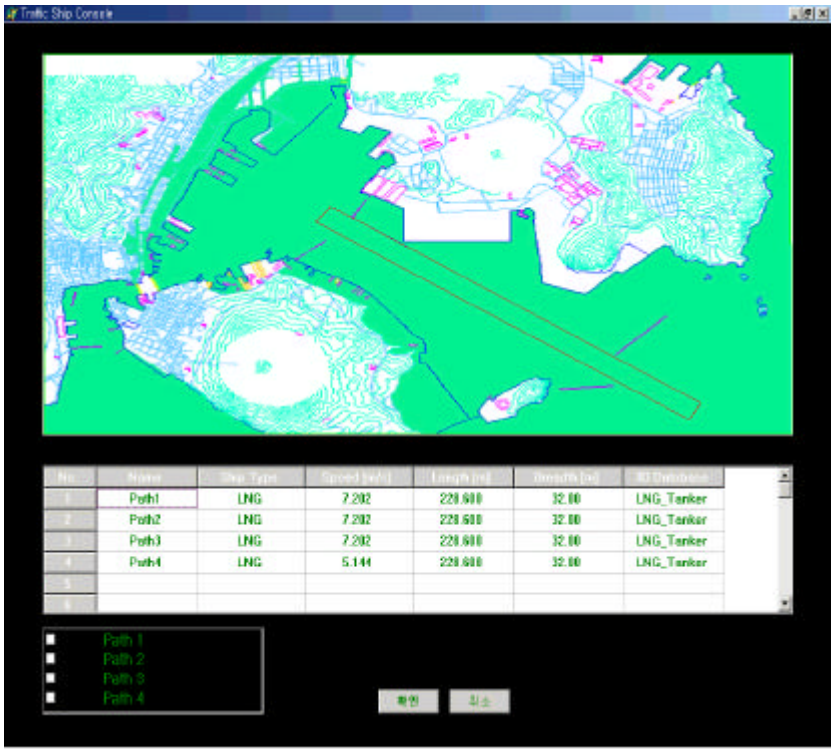


Fig. 46 Target ship select window

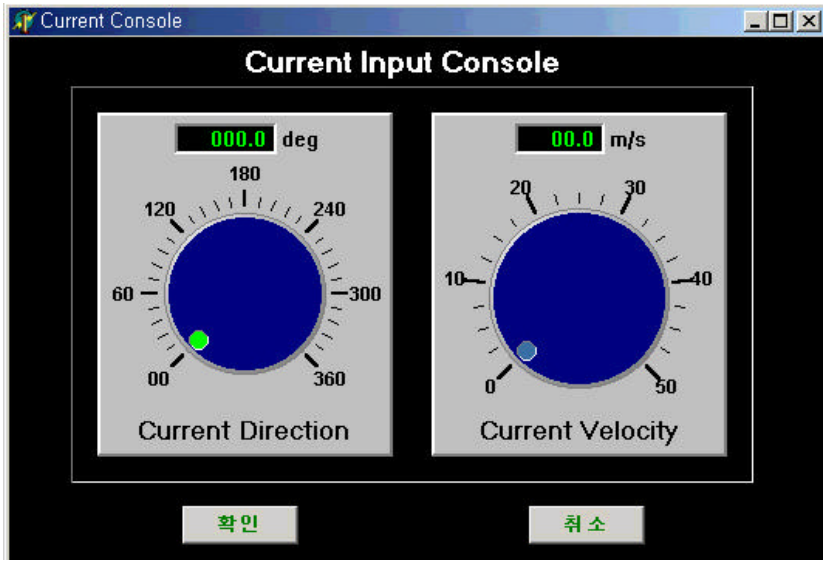


Fig. 47 Current input window

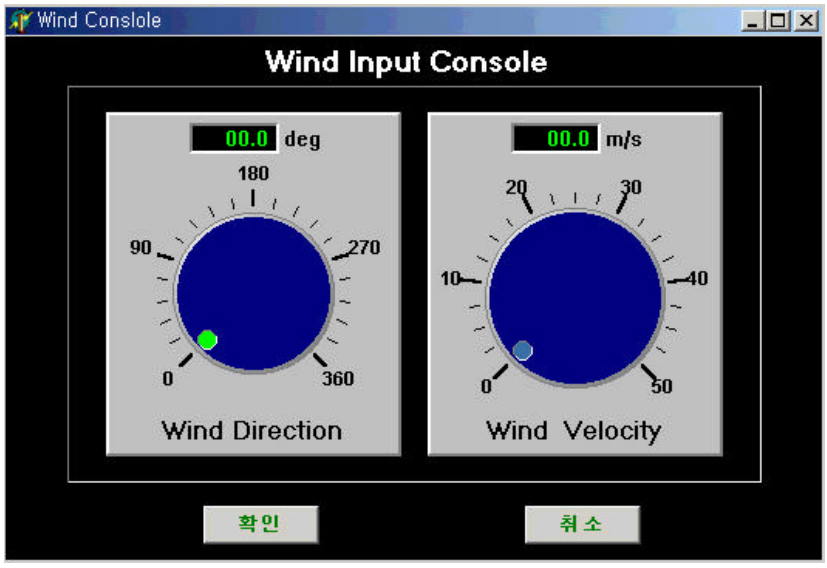


Fig. 48 Wind input window

(4) 3차원 시계재현부(3D Visual)

3차원 시계재현부는 운동계산부와 UDP(User Diagram Protocol) 소켓(Socket)으로 연결하여 실시간 시뮬레이션에 맞도록 구성하였다. TCP/IP 연결은 신뢰성과 대량의 데이터를 보내고 받을 수 있지만 신뢰성을 보장하는 대신에 시간 면에서 많은 지연이 있다. UDP 연결은 신뢰성 면에서는 TCP/IP보다는 떨어지지만 데이터를 빠르게 보낼 수 있다는 장점이 있다. 실시간 시뮬레이션에서 시계재현을 위해서는 초당 23 24 프레임 정도의 갱신율을 요구한다. 그런면에서 보면 UDP연결이 바람직하다. 또한 시계재현부에서는 시뮬레이션을 위한 3차원 그래픽 특성상, 실제 상황을 실시간으로 재현해야 하는 기능이 요구되기 때문에 CAD나 Animation용 3차원 그래픽 애플리케이션을 이용한 사진과 같은 사실감 있는 물체 형태의 정확한 묘사기능보다 시뮬레이션 대상 물체의 운동상태를 실시간으로 정확히 표현할 수 있는 3차원 컴퓨터 그래픽 기법이 더 요구된다. 실제해상에서 거동하고 시각적 현실감과 조종운동 수치계산의 결과를 정확하고 직관적으로 제공하는 기능이 필요할 것이다. 이런 기능을 위해서 본 시계재현부에서는 3차원 그래픽 개발용 S/W로 Silicon Graphics Inc(SGI)의 IRIS GL 개방형 시스템인 OpenGL 기반으로하는 Paradigm Simulation Inc의 VEGA의 개발 환경을 사용하여 시계 재현 시스템을 개발하였다. 또한 VEGA의 데이터베이스 처리를 원활하게 하기위해 Paradigm Simulation Inc의 MULTIGEN의 3차원 데이터베이스 모델링 프로그램을 이용하여 항만과 선박 등을 모델링하였다. MULTIGEN은 모델을 데이터베이스하는 계층적인 구조로 되어있어 모델 데이터를 최적화하는데 역할을 한다. 또한 시뮬레이션에서 실시간 랜더링은 중요하므로 다음을 고려해야 할 것이다.[16]

① Frame Rate : 데이터베이스의 모델링 테이터를 초당 24 프레임으로 전체 가상 환경을 비쥬얼라이제이션 해야 하며 이는 인간의 시각 능력을 고려한 시뮬레이션의 비쥬얼라이션의 최적의 속도이며 가상 환경 모델의 복잡도, 상세도, Texture의 요구 사항을 저하 시키지 않는 범위에서 운영해야한다.

② Bounding Volume과 Collision Detection : 선박의 충돌과 같은 실제 상황을 반영하기 위해서는 객체들 사이의 Collision Detection이 필요하다. 실시간 속도로 Collision Detection을 얻어내기 위해서는 모델내의 모든 Polygon을 계산하여 간섭 Point를 얻어내지 않고 Bounding Volume을 사용하여 처리한다.

③ Drawing Order : 항해자가 가상환경을 볼 때 근거리에 있는 물체는 자세하게 보이고 원거리에 있는 물체는 간략하게 보인다. 랜더링 순서로 본다면 원거리 객체는 먼저 그려지고 근거리는 시각적으로 후에 그려진다. 이러한 물체의 순서를 Drawing Order라고 한다. 이것은 Fixed List, Z-Buffer, BSP(Binary Seperating Plane)에 의해서 구현되며 BSP는 Fixed

List와 병행하여 사용한다. Z-Buffer는 항해자 시점으로부터 Pixel Depth를 결정한 후 Pixel에 우선 순위를 두어 가까운 객체의 Drawing을 보장한다 . 하지만 BSP보다 계산량이 많아진다. 랜더링 속도를 위해서 BSP와 Fixed List를 사용하여 처리하는 것이 좋다.

④ Level Of Detail : 물체의 정확도의 정도가 서로 다르면서도 비슷한 모양의 Object를 여러 개 만들어 화면에서 거리에 따라 어느 것이 보일지를 지정할 수 있다. 즉 가깝게 보이는 Object는 보다 정확해야 하고 멀리 보이는 Object는 덜 정확해도 무방할 경우 사용할 수 있다. 이렇게 하면 화면의 디스플레이 속도가 증가하는 효과를 얻을 수 있다. 그런 반면, 개체의 수가 늘어나 다운로딩 속도는 느려지게 된다. 그러므로 LOD(Level Of Detail)의 개체는 최대한 작게 해야 한다.

위와 같은 실시간 랜더링을 위하여 MULTIGEN과 VEGA를 사용하여 시계재현부를 구성하였다.

(5) 2차원 조감도 (2D Map Display)

운동 계산부의 데이터를 TCP/IP 소켓으로 받아서 2차원 지형모델에 선박의 항적을 표시하는 부분이다. 2차원 조감도를 이용하여 선박의 조종운동의 변화를 볼 수 있고 또한 항해자는 선박의 항적을 이용하여 정확한 항해를 하는데 이용한다. 2차원 조감도를 구성하는데 좌표의 정확한 표시가 필요하다. 정확한 좌표를 표시하기 위해서 수치해도를 이용하였고 이를 바탕으로 2차원 조감도를 구성하였다. Fig. 48는 2차원 조감도를 나타내고 있다.

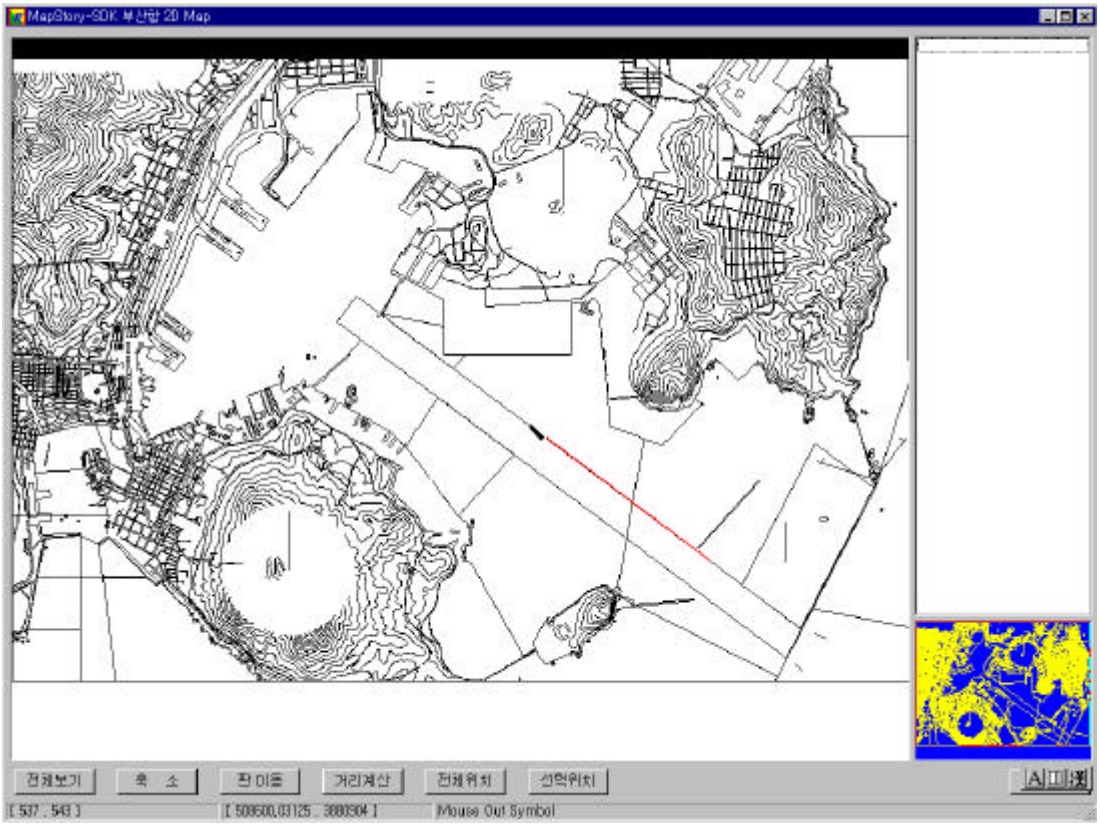


Fig. 49 2D Map Display

3.3 실시간 시뮬레이션 예

본 선박조종시뮬레이터를 이용하여 부산항에 대한 조종 시뮬레이션을 수행하였다.[11] 부산 외항 방파제 부근에서, 11.0 *kt* (half ahead)이며, 입항시 침로각 307° 도 상정하였다. Table 6.1은 시뮬레이션에 이용한 조건들이다. Fig. 46 와 Fig. 47, Fig. 48은 부산항에 적용한 예이다.

Table. 2 Simulation Condition

조건	내용
항만	부산항 (Pusan Port)
자선	SR108 Container Ship (Table 3)
외란	없음

Table. 3 Ship Principal dimensions of SR-108 Container Ship

Length b.p	175 *m*
Breadth	25.4 *m*
Draft	8.5 *m*
CB	0.559
Rudder area	32.4625 m^2
Rudder height	7.7
Propeller diameter	6.5
Pitch ratio	1.005

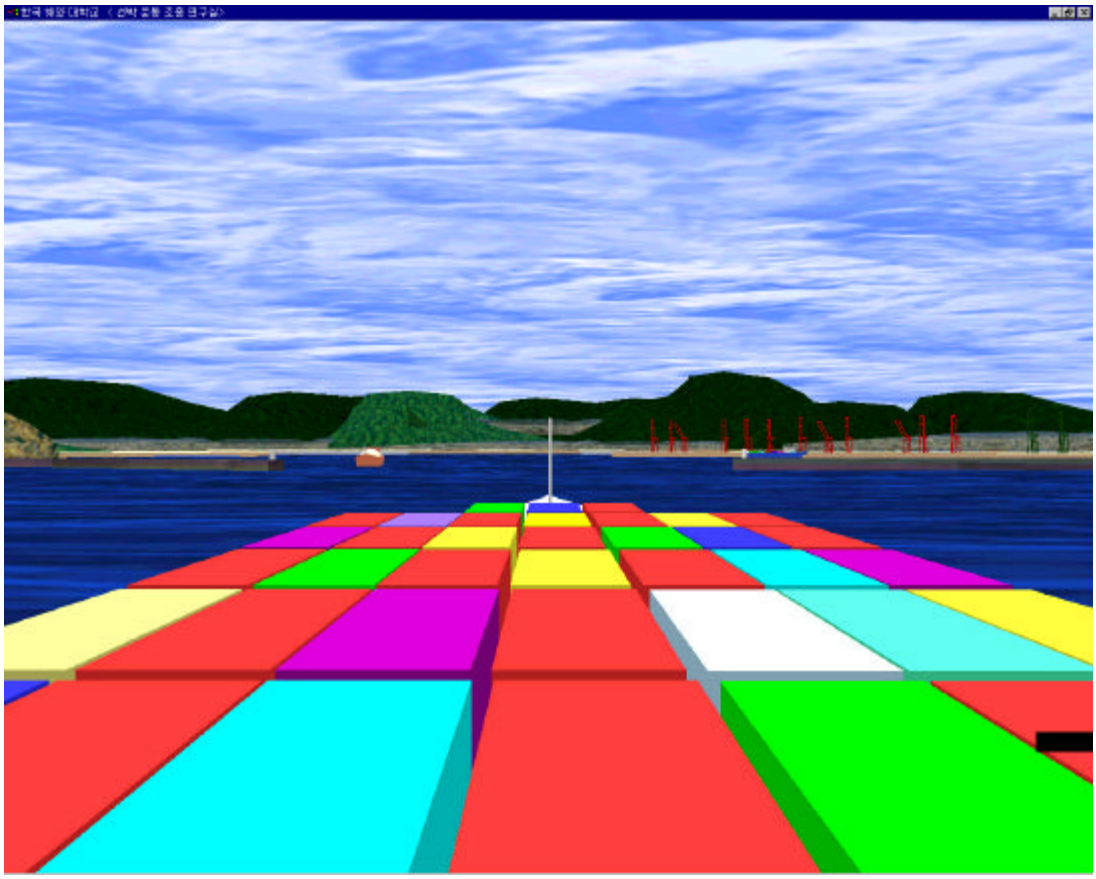


Fig. 50 Pusan harbor entry initial

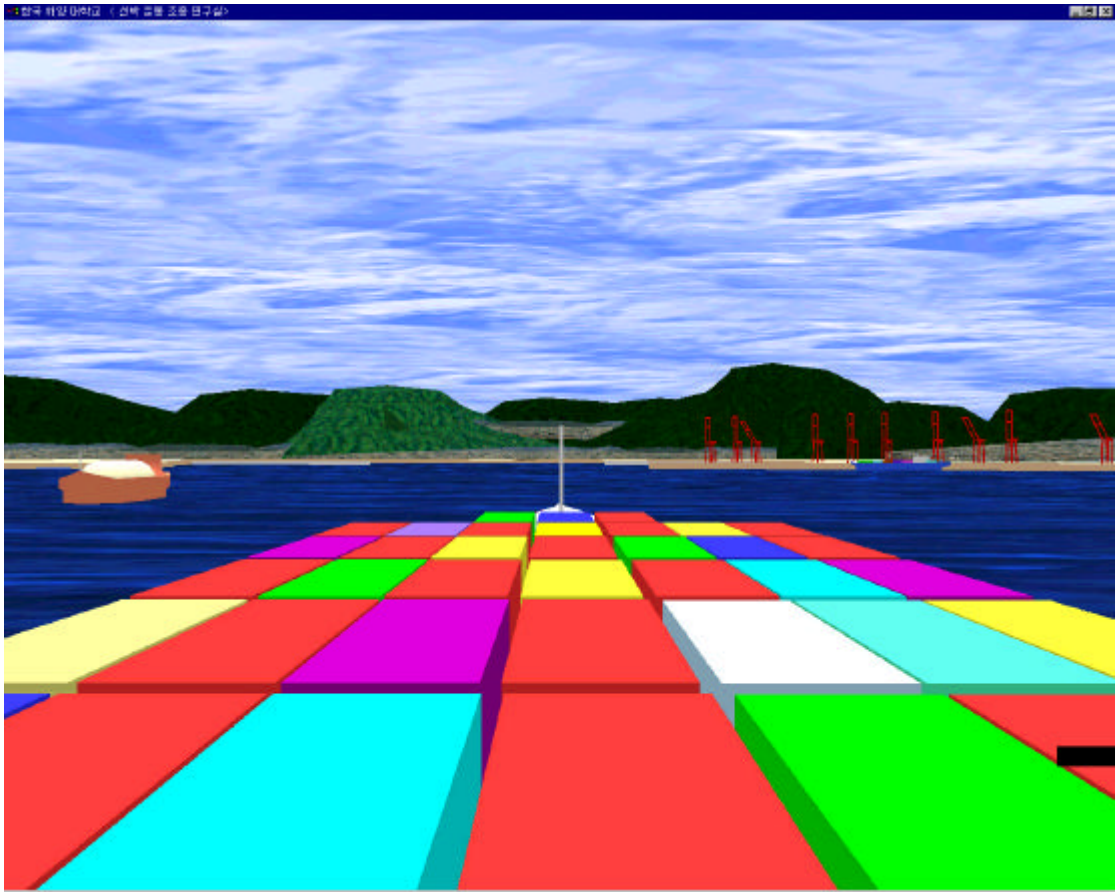


Fig. 51 The view of target ship intersection

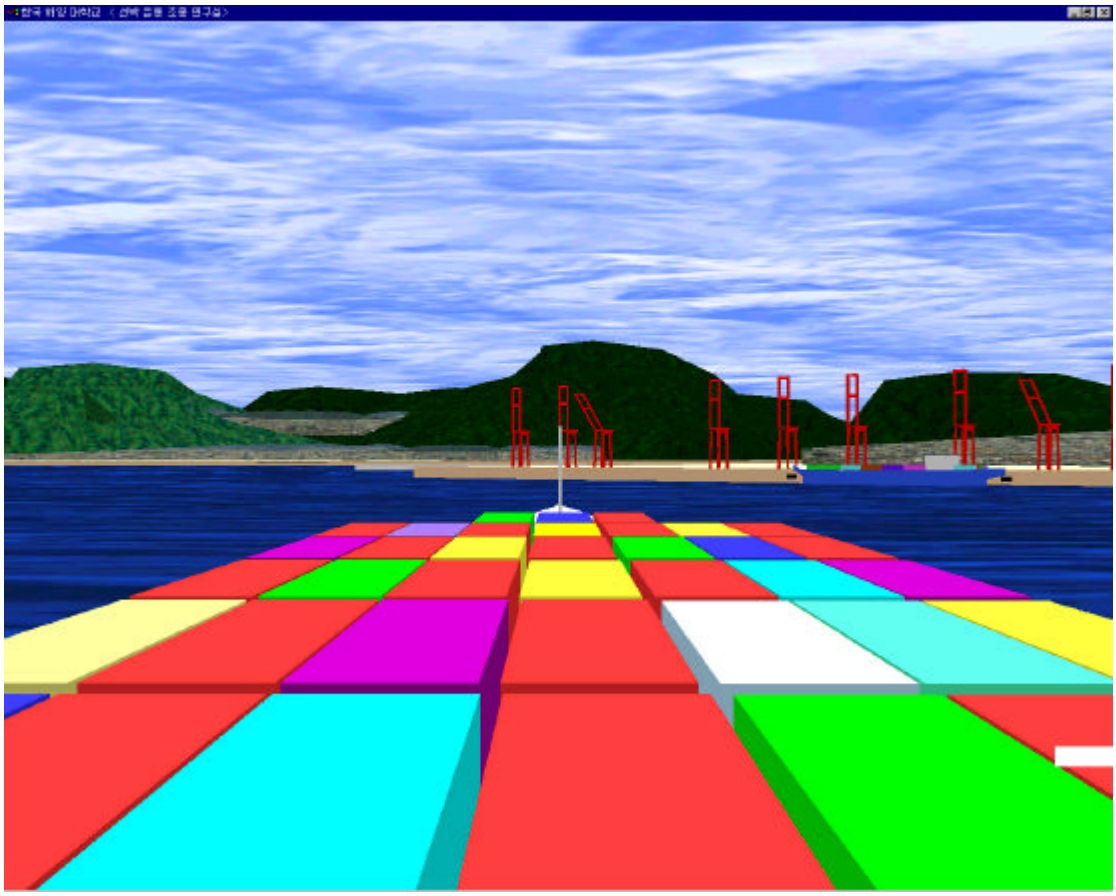


Fig. 52 The view of alongside pusan harbor

4. 결언

본 논문에서는 분산처리방식을 이용한 실시간 선박 조종 시뮬레이터의 개발에 대해서 논의하였다. 분산처리기법을 이용하여 실시간으로 선박의 영상을 재현할 수 있었고 가상현실 기법이 시뮬레이터를 개발하는데 어떻게 활용 되는지도 살펴보았다. 시뮬레이터는 가상현실을 구현하는데 초기단계로 볼 수 있고 다양한 가상현실 기법을 이용하여 실제로 몰입하여 조종할 수 있는 HMD(Head Mounted Display)나 Crystal eyes 등의 장비를 이용하여 몰입감을 증대할 수 있을 것이다.[10]

본 논문에서는 분산처리기법 중에 네트워크를 이용하여 PC간의 데이터를 활용하여 시뮬레이터를 효율적으로 사용할 수 있는 방법에 대해 논하였다. 또한 본 시뮬레이터를 이용하여 항해사의 선박 조종 훈련용과 항만 수로 설계시 안전성 평가용으로 유용하게 활용될 수 있을 것이다. 앞으로 증강현실 기법을 활용하여 자연스러운 실세계 영상을 만들어야 할 것이고 시뮬레이션의 중요한 요소인 수학모델에서 다양한 외력의 영향을 고려한 수학모델을 개발해야 할 것이다. 또한 레이다(Radar)와 전자해도(Electronic Chart Display Infomation System)를 이용한 더 실제에 가까운 브릿지 정보를 표현해야 할 것이다. 또한 시뮬레이터의 제어 입력부를 별도의 하드웨어로 제어할 필요가 있고 다양한 항구의 지형 모델링을 확보할 필요가 있다.

참고문헌

〔1〕 **강 창구**, “IMO **선박 조종성 기준**”, **대한조선학회지**, **제**30**권 제** 2**호**, 1993.

〔2〕 Kose, K. et al., “**出入港操船 ツミュレタ**”, **日本航海學會 第**78**回 講演會**, 1988.

〔3〕 http://203.31.130.95/~hskim/multimedia/ch11/ch11-02-body.htm.., “**가상현실 시스템의 구성**”,

〔4〕 **주우석**., “3**차원 컴퓨터 그래픽스**”, **그린출판사**, 1999.

〔5〕 **윤 철규**, “3**차원 지형모델링의 효율적 기법에 관한 연구**”, **충북대학교**, 1997.

〔6〕 **김 화종**, “**컴퓨터 네트워크 프로그래밍**”, **홍릉과학출판사**, 2000.

〔7〕 Ogawa, A., et al., "MMG**情報**(Ⅰ) - **操縱運動の數學**Model**について**-", **日本造船學會誌**, **第**575**号**, 1977.

〔8〕 Newman, J. N., “Marine Hydrodynamics", The MIT Press, Cambridge, 1978.

〔9〕 **손 경호**, “Hydrodynamic Force and Manoeuvring Characteristics of Ship at Low Advance Speed", **대한조선학회논문집**, **제**29**권**, **제**3**호**, 1992.

〔10〕 **클로드 카도즈**., “**가상현실**(**도미노총서**2)”, **영림카디널**, 1997.

〔11〕 **이 성욱**, “**출입항 조종의 실시간 시뮬레이션 기법 및 데스크탑형 시뮬레이터의 개발에 관한 연구**”, **한국해양대학교**, 1998.

〔12〕 Isherwood, R, M., “Wind Resistance of Merchant Ship", Trans. RINA, Vol.114, 1972.

〔13〕 van Lammeren, W.P.A, et al., "The Wageningen B-Screw Series", Trans. SNAME, Vol. 77, 1969.

〔14〕 Kobayashi, H., et al., "**曳船による조종운동に關する硏究**“, **日本航海學會論文集**, **第**88**号**, 1992.

〔15〕 MultiGen Inc., “MultiGen Creator User's Guide", The REALTIME 3D Company, 1997.

〔16〕 MultiGen・Paradigm Inc., “VEGA Programmer's Guide", The REALTIME 3D Company, 1999.

〔17〕 **이상엽**., “Visual C++ Programming Bible version 6.0", **영진출판사**, 1999.