

모바일 게임 : 현황과 전망

성균관대학교 한정현

LG 전자기술원 김태성

1. 서론

국내 게임 산업과 시장은 최근 폭발적인 증가세를 보여왔다. 이를 주도한 것은 리니지로 대표되는 온라인 게임이지만, 이에 못지 않게 모바일 게임 분야의 성장세 역시 두드러졌다. 2002년 모바일 게임 시장은 2001년에 비해 180% 가량 성장한 1000억 원 이상의 규모였다. 또한, 국내 게임 시장이 세계 시장에서 차지하는 비율 역시, 온라인 게임 6.7%, 모바일 게임 5.3%의 순서를 보인다[1].

모바일 게임 플랫폼은 휴대폰, PDA, 그리고 '게임 보이'와 같은 휴대형 전용 게임기로 구분할 수 있다. 하지만 국내에서 PDA 및 휴대형 전용 게임기 시장은 매우 미미하다. 따라서, 본 논문은 휴대폰으로 논의를 집중하여 모바일 게임 개발 기술의 현황을 살펴보고 미래를 전망한다. 2절에서는 모바일 게임 플랫폼을 정리하고, 3절에서는 모바일 게임 구현 기술의 간단한 예를 살펴본다. 한편, 3D 모바일 게임은 이제 막 시장 진입 단계에 들어갔는데, 4절에서는 모바일 3D의 소프트웨어 및 하드웨어 구현 현황을 정리하고, 5절에서는 3D 게임 엔진 및 개발 사례를 살펴본다. 마지막으로 6절에서 모바일 게임의 미래를 전망해 보고 글을 맺는다.

2. 모바일 게임 플랫폼

표 1 모바일 게임 플랫폼

플랫폼	개발언어	수행방식	서비스사(개발사)
GVM/GNEX	Mobile C	VM	SK 텔레콤 (신지소프트)
SK-VM	Java	VM	SK 텔레콤 (XCE)
KVM	Java	VM	LG 텔레콤(Sun)
BREW	C/C++	Native Binary	KTF (Qualcomm)
WIPI	Java/C/C++	통합	[KWISF, TTA, ETRI]

모바일 게임 구동 방식은 크게 Virtual Machine (VM) 방식과 Native Binary 방식으로 나뉜다. 시스템

안정성과 보안성에서 강점을 보이는 VM 방식은 SK 텔레콤의 GVM/GNEX와 SK-VM이 대표적인 예이고, 빠른 실행 속도의 강점을 보이는 Native Binary 방식은 KTF의 BREW가 대표적인 예이다(표 1 참조.)

2.1 VM 방식

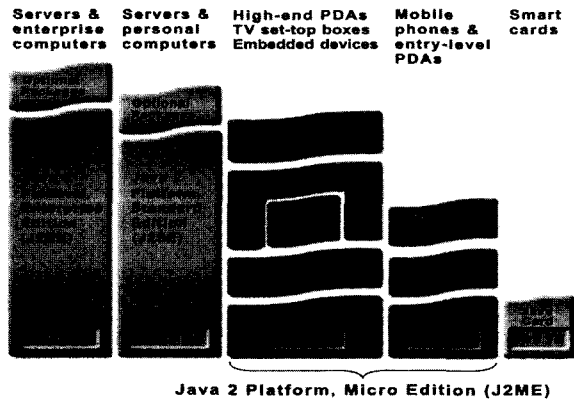


그림 1 Java 플랫폼 구조 [18]

VM 방식은 Java를 통해 대중화되었는데, 그림 1에 도시된 바와 같이, J2EE(Java 2 Enterprise Edition)는 기업용 컴퓨터, J2SE(Java 2 Standard Edition)는 PC, J2ME(Java 2 Micro Edition)는 휴대폰과 PDA와 같은 소형 기기에서의 표준화된 Java API이다. J2ME는 이른바 Configuration과 Profile, 그리고 Optional Package로 구성되는데, VM 및 기본적인 클래스 라이브러리로 구성되는 J2ME Configuration 중 휴대폰에 쓰이는 것은 Connected Limited Device Configuration (CLDC)이다. 한편, CLDC는 상위 레벨 API를 제공하는 Mobile Information Device Profile(MIDP)과 결합되어 Java 실행 환경을 제공한다.

XCE사가 개발한 SK-VM[32]은 J2ME 구조를 따라, XVM(XCE 자체 개발 VM), M-Configuration (CLDC) 그리고 M-Profile(MIDP)로 구성되어 Java 애플리케이션을 지원한다. 한편, SK-VM보다 많은 개

발자들을 확보한 VM으로, 신지소프트가 개발하여 역시 SK 텔레콤에서 채택한 GVM과 GNEX[31]가 있다. (GNEX는 GVM의 후속 버전이다.) GVM/GNEX는 C 언어의 모바일 용 축소판인 Mobile C를 지원한다.

2.2 Native Binary 방식

개발된 프로그램을 프로그램이 실행될 CPU에서 동작되는 기계어로 컴파일하여 실행시키는 것이 Native Binary 방식이다. 2001년 1월 쉘컴은 Native Binary 방식의 BREW(Binary Runtime Environment for Wireless) [26]를 발표하였고, 이를 세계 최초로 채택한 업체는 KTF였다. BREW에서 제공하는 C++ 프로그래밍 환경은 Visual C++ 환경과 매우 유사하므로, Visual C++ 프로그래밍에 익숙한 사용자는 쉽게 BREW 프로그래밍을 수행할 수 있다. 다만, 객체 지향 언어의 기본적인 사항 중 클래스 상속과 가상 함수 등을 지원하지 않으며, 전역 변수 등도 사용할 수 없는 제약 사항이 있다[7,8].

또 하나의 대표적인 Native Binary 방식은 Symbian이다. Symbian은 Ericsson, Nokia, 삼성전자 등 모바일 산업을 주도하는 기업들이 공동 설립한 컨서시엄으로, 2G, 2.5G 및 3G 휴대폰을 위한 공개 표준 운영체제를 지향하는 Symbian OS를 개발하고 있다[34]. Symbian OS는 현재 버전 7.0까지 개발되었다. Symbian OS를 채택한 대표적인 휴대폰은 Nokia의 Series 60 및 90 휴대폰들로, C++ 프로그래밍을 지원한다. (최근 출시된 Nokia의 휴대폰 겸 게임기 N-Gage 역시 Series 60 플랫폼을 채택하고 있다[23].)

2.3 WIPI

서 정리한 VM 방식은 CPU 성능과 자원이 제한된 휴대 전화기에서 상대적으로 느린 속도를 보이는 단점을 가진다. 반면, Native Binary 방식은 VM 기술과 달리 탑재된 CPU의 성능을 최대한 활용할 수 있어 VM 방식에 비해 빠른 실행 속도를 제공하지만, Java가 아닌 C/C++를 사용하는 데서 야기되는 이식성 문제 및 취약한 보안성 등의 단점을 가진다.

WIPI(Wireless Internet Platform for Interoperability)는 한국 무선인터넷 표준화포럼(KWISF), 한국전자통신연구원(ETRI), 한국정보통신기술협회(TTA) 주도로 개발한 무선 인터넷 표준 플랫폼으로, VM 방식과 Native Binary 방식의 문제점을 동시에 해결하는 것을 목표로 한다. 즉, Java 언어의 장점을 수용함과 동시에 Native Binary의 실행 성능을 갖도록 하는 것이다[4]. 따라서 WIPI 플랫폼은 Java와 C/C++를 기본언어로 채택하여 프로그램 개발이 가능하도록 하며 단말기에 탑재

되는 애플리케이션은 Native Binary로 만들어져 실행 성능을 보장하는 것이다. C/C++ 언어로부터 Native Binary를 생성하는 데에는 문제가 없다. 하지만, Java는 기본적으로 VM을 대상으로 하는 언어이기 때문에 Java를 컴파일하여 단말기에서 직접 실행 가능한 Native Binary를 만들어 주어야 된다. 이는 서버에서 Ahead Of Time Compile (AOTC) 방식으로 구현된다. 즉, 바이트 코드 형태로 되어 있는 자바 애플리케이션을 미리 컴파일하여 단말기의 CPU에 최적화된 바이너리 코드를 생성하는 것이다.

현재 WIPI 플랫폼 게임은 주로 Java로 구현된다. C 프로그래밍도 가능하지만 보안성 및 UI 컴포넌트 미비 등의 문제로 Java가 선호되는 것이다. 2003년 하반기부터 WIPI가 장착된 휴대폰이 출시되기 시작하였고 (삼성전자 SPH-X9300) KTF를 중심으로 다수의 WIPI 기반 게임이 서비스 중에 있다.

3. 모바일 게임 구현 기술

모바일 기기는 저성능 프로세서와 소용량 메모리, 그리고 소형 액정 화면 등의 한계를 가진다. 이 같은 저 사양 하드웨어 상에서의 게임 구현 방식은 PC에서의 구현 방식과 상당한 차이를 보인다. 몇 가지의 대표적인 예를 살펴 보도록 하자.

SK-VM에서 제공되는 툴[32]을 이용해 2D 게임 그래픽 구성 과정을 설명하면 다음과 같다. 그림 2-(a)에 보인 것처럼 모바일 게임의 배경 화면은 대체로 단순하다. 따라서 이를 구성하는 기본 단위를 그림 2-(b)에서와 같은 '타일'들로 정의한 뒤, 이 타일들을 붙여 나가면서 배경 화면을 구성한다. 즉, 배경화면을 2차원 배열로 정의한 뒤, 배열의 각 cell에 타일 번호를 지정하면 된다. 한편, 이러한 배경 화면에 움직이는 캐릭터를 구현하기 위해서는 sprite를 사용한다. Sprite란 화면 상에서 이동 가능한 2차원 이미지를 통칭하는 것으로, 마우스 커서가 대표적인 예이다. 그림 2-(c)는 곤충 캐릭터를 위한 sprite들의 예이다. 연이은 동영상 프레임(frame) 각각에 이들 sprite들을 적절히 배치하면 애니메이션 효과를 얻을 수 있다. 그림 4-(d)는 하나의 프레임 예이다.

GVM/GNEX, SK-VM, BREW 등 모든 모바일 게임 플랫폼은 다양한 애플리케이션/게임 개발 툴(리소스 편집기, 에뮬레이터 등)을 제공한다. C/C++ 혹은 Java에 익숙한 개발자는 이러한 툴들을 이용하여 비교적 짧은 시간에 게임을 개발할 수 있다.

대부분의 모바일 단말기는 부동 소수점 (floating point) 연산을 지원하지 않는다. 하지만, 정수 연산만을

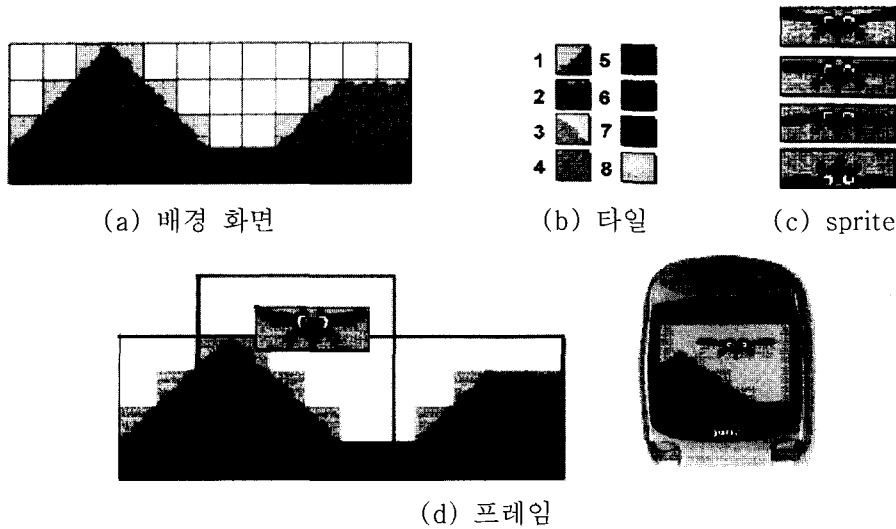


그림 2 2D 게임 화면 구성 예 [32]

이용해 게임 구현을 하는 데에는 한계가 있다. 이러한 한계는 3D 게임 구현 시 확연해진다. 따라서, 정수 연산을 사용하면서도 어느 정도의 정확도를 제공하는 고정 소수점 (fixed-point) 연산을 사용한다. 고정 소수점 변수로는 대개 4 바이트 정수 타입을 사용한다. 소수점이 중앙에 있다고 가정하면, 앞 2 바이트는 부호 및 정수 부분으로, 뒤 2 바이트는 소수 부분으로 사용된다. 정수와 고정 소수점 간 변환 및 덧셈 연산을 위해서 다음과 같은 macro를 사용한다.

```
#define INT2FXD(X) ((INT32)((X) << 16))
#define FXDADD(A, B) ((A) + (B))
#define FXD2INT(X) ((INT32)((X) >> 16))
```

고정 소수점 연산이 끝난 뒤에는 원래의 데이터 타입으로 (위 경우는, 정수로) 변환되어야 함은 물론이다. 실제 고정 소수점 연산은 부동 소수점 연산을 대체하기 위해 쓰이는 것으로, 위의 경우보다 복잡한 macro를 통해 수행된다. 하지만, 이에 관한 논의는 본 논문의 범위를 넘어서는 것으로, 관심 있는 독자들은 참고 문헌 [5]를 참조하라.

4. 모바일 3D 그래픽스

2D 일변도였던 모바일 게임 분야에 최근 3D 게임이 선보이기 시작했다. 이 절에서는 3D 게임의 핵심 요소인 소프트웨어 및 하드웨어 3D 렌더링(rendering) 기술 및 그 API를 검토한다.

4.1 렌더링 파이프라인 및 소프트웨어 렌더링

3차원 그래픽스는 흔히 “모델링에 이은 렌더링”으로 정의된다. 가장 많이 쓰이는 모델링 기법은 그림 3에 도

시된 바와 같은 다각형 메쉬(polygon mesh)이다. 메쉬를 구성하는 다각형들은 이른바 렌더링 파이프라인이라고 하는 순차적인 연산 과정을 거쳐 화면에 그려진다 [29]. 렌더링 파이프라인의 핵심은 다양한 기하 변환 및 라이팅(lighting) 계산 등을 포함하는 “기하(geometry) 단계”와 변환된 다각형 데이터 및 색상, 텍스처 정보를 이용하여 화면에 픽셀을 생성하는 “래스터라이징(rasterizing) 단계”로 구분된다.

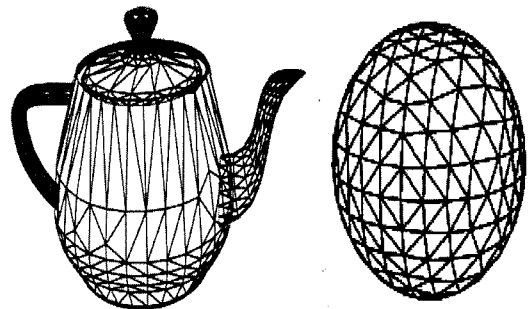


그림 3 다각형 메쉬

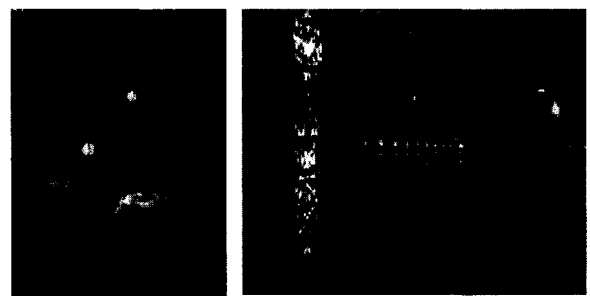


그림 4 PocketGL 스크린 샷

현대의 PC 그래픽스 카드는 이러한 기하 및 래스터라이징 단계를 모두 하드웨어적으로 지원하지만, 대부분의 모바일 기기에서는 이를 소프트웨어로 구현한다. 소프트웨어

렌더러 구현은 휴대폰보다 상대적으로 고사양인 PDA에서 먼저 시작되었는데, PDA 전용 소프트웨어 렌더러의 예로는 Pocketgear사의 PocketGL이 있다[25]. 이는 데스크탑 3D API의 국제 표준인 OpenGL(Open Graphics Library)[24]을 PDA 사양에 맞게 축약한 것으로 WinCE 환경에서 작동한다. PocketGL은 고정 소수점 연산 등을 이용해 렌더링 파이프라인의 기본 기능을 모두 수행한다. 그림 4는 PocketGL의 스크린 샷으로 음영처리(shading), 원근투영(perspective projection) 등이 구현되었음을 보여준다.

한편, PDA에서의 그래픽스 라이브러리로 인텔의 GPP(Graphics Performance Primitives)가 있다[16]. 이는 인텔 PCA 아키텍처[17]에 최적화된 그래픽스 라이브러리로, 렌더링 파이프라인 구현을 위해 필요한 루틴을 제공한다. 기하 단계 구현을 위한 고정 소수점 연산과 행렬 및 벡터 연산 관련 루틴, 레스터라이징 단계 구현을 위한 스캔라인(scanline) 알고리즘 등을 이용해 소프트웨어 렌더러를 구현할 수 있다.

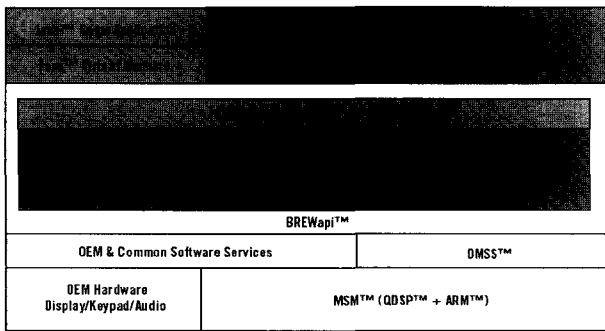


그림 5 Q3D 아키텍처

한편, 휴대폰에서의 소프트웨어 렌더러 구현도 최근 활발히 진행되어 왔다. 대표적인 예가 퀄컴의 MSM6xxx™ 시리즈 칩셋에서 제공되는 Qualcomm Q3Dimension™ Gaming Platform(이하, Q3D로 약칭)으로 렌더링 파이프라인의 기본 API를 제공한다[27]. Q3D는 BREW API의 일부로, BREW SDK™를 통해 지원된다(그림 5 참조.)

한편, 국내에서는 SK 텔레콤이 기가(GIGA: Graphic Instruction Graphic Acceleration) 프로젝트를 통해 3D 구현을 위한 소프트웨어 렌더러를 채택하였다. 이의 API 이름은 SKT-GL로, 4.2절에서 설명될 모바일 3D 표준인 OpenGL ES와 유사한 API이다. SKT-GL은 2003년 하반기에 출시된 최초의 기가폰(SKY IM-6400)에서 1000개 다각형에 대해 7~8 fps(frames per second) 정도의 렌더링 성능을 보인다. 일본의 경우, HI사의 Mascot Capsule Micro3D Engine이 소프트웨어 렌더링에서 독보적인 위치를 차지하고 있다[14]. 이

는 C와 Java 언어로 구현되었으며, SKT-GL과 함께 SK 기가폰에 채택되었다.

4.2 하드웨어 가속기 및 API

데스크탑 컴퓨터와 모바일 기기의 하드웨어 사양 차이가 워낙 크기 때문에, 모바일 기기에서 데스크탑 수준의 3D 영상을 기대할 수는 없다. 하지만, 데스크탑에서의 화려한 3D 그래픽에 익숙해져 있는 사용자의 욕구를 낮은 수준에서나마 충족시키기 위해서는 모바일 3D 처리를 위한 하드웨어를 추가하는 것이 필수적이라는 데 많은 사람들이 공감한다. 이를 위해서 제시되는 방안은 크게 세가지로 나뉘어진다. TI사의 OMAP[35]으로 대표되는 애플리케이션 프로세서, 부동 소수점 연산 등 상대적으로 고성능 연산을 수행하는 DSP 칩, 그리고 PC의 그래픽스 카드에 대응하는 3D 전용 칩이 그것이다[3].

3D 게임 구현만을 생각한다면 3D 전용 칩을 장착하는 것이 최선이지만, 휴대폰에서 요구되는 다양한 멀티미디어 기능을 충족시키기 위해서는 범용성을 가진 애플리케이션 프로세서를 선택하는 것이 합리적인 방안일 수 있다. 하지만, 애플리케이션 프로세서는 생산 단가가 높은 단점이 있어, DSP 칩이 선호되기도 한다. 한편, post-PC의 대표주자인 휴대폰에서 PC에서와 같은 3D 콘텐츠를 기대하는 사용자 층이 늘어날 것을 전망해 보면, 3D 전용 칩이 대중화될 가능성 역시 존재한다고 볼 수 있다.

SK 기가폰에서는 현재 나조미사의 Java 가속칩인 JA108[21]을 채택하고 있다. (이런 이유로 현재로서는 Mobile C를 지원하는 GNEX보다 Java를 지원하는 SK-VM에서의 3D 구현이 합리적인 선택이다.) 하지만, Java 가속칩을 DSP 칩으로 대체한 기가폰이 곧 선보이게 될 것이다. 한편, 3D 전용 칩 개발은 PDA 분야에서는 미국의 MeidaQ[19]가, 휴대폰 분야에서는 일본의 산신전기(GSHARK)[30]와 미쯔비시(Z3D)[2] 그리고 핀란드의 bitboys (Acceleon Gxx™ 시리즈)[6] 등이 선도하고 있다. MeidaQ사의 칩은 HP iPAQ h2200 시리즈에 탑재되었다. 산신 전기는 HI사의 Mascot Capsule Micro3D Engine을 GSHARK에 포팅 중이고, Z3D가 장착된 휴대폰(일본 NIT DoCoMo D505i 등)은 이미 2003년 하반기에 일본에서 출시되었으며, Acceleon 칩은 NEC의 모바일 3D 솔루션으로 채택되었다. 한편, 국내의 우수 단말기 제조사들은 2004년 중하반기 출시를 목표로 3D 전용 칩이 탑재된 휴대폰을 설계 및 제조 중에 있다.

위에서 언급한 3D 구현을 위한 세가지 흐름 중 어느 것이 대세가 되더라도, 게임 개발자의 입장에서 이의 소프트웨어 API가 필요한데, 이의 표준으로 2003년 7월에

발표된 것이 OpenGL ES(OpenGL for Embedded Systems)이다[9]. 이는 Nokia, Ericsson, sgi, SK 텔레콤 등을 주축으로 한 Khronos Group이라는 컨서시엄에 의한 제안된 휴대용 단말기 3D API 표준으로, OpenGL API 중 모바일 기기에 적합한 것들을 선별하여 만들어졌다. OpenGL ES API의 몇 가지 예를 들어 보이면 다음과 같다.

GLAPI void APIENTRY glColor4x(GLfixed red, GLfixed green, GLfixed blue, GLfixed alpha);

GLAPI void APIENTRY glRotatex(GLfixed angle, GLfixed x, GLfixed y, GLfixed z);

glColor4x()는 (red, green, blue, alpha)로 컬러 값을 설정하는 함수이고 (alpha는 불투명도를 나타냄) glRotatex()는 (x, y, z) 벡터 축을 중심으로 angle 만큼의 회전 변환을 설정하는 함수이다. OpenGL에 익숙한 독자는 표 2로부터 OpenGL API 중 OpenGL ES 1.0으로 채택(In)된 것과 탈락(Out)한 것의 예를 볼 수 있을 것이다.

표 2 OpenGL ES [9]

In	Out
2D Textures	1D, 3D, Cube Maps, Proxy
Wrap repeat, edge_clamp	clamp, border_clamp Texture Borders
Compressed Texture (optional)	GetCompressedTexImage
TexSubImage, CopyTexImage	
Multitexture (optional)	Combine Environment
RGBA / packed formats, L, LA	All other image formats
All Filters	

In	Out
Vertex Arrays	Begin/End, Edge Flags
Most Geometric Primitives	Quads, Quad Strips, Polygons
Matrix Stack	Transpose Mult/Load
Viewport, DepthRange	TexGen, User Clip Planes
Vertex Lighting	Back Materials Local Viewer, Secondary Color
ShadeModel	

현재 상당수의 소프트웨어 렌더러 및 하드웨어 가속기가 OpenGL ES 표준에 따라 구현되고 있다. 국내에서는 ㈜넥서스칩스와 ㈜와우포엠이 휴대폰 전용 3D 가속기(ISKRA II)[22]를 개발하면서 OpenGL ES를 채택하였고, 국내 단말기 제조사에서 개발 중인 3D 가속기 역시 대부분 OpenGL ES에 기반하여 진행 중에 있다. 또한, 전술한 bitboys의 Acceleon GxxTM 시리즈 그리고 후술할 게임 엔진 Swerve 등도 모두 OpenGL ES에 기반해 있다. 물론, 데스크탑 3D API로 공인받지 않은 “사실상의 (de facto) 표준” DirectX가 존재하는 것처럼, 모든 소프트웨어 렌더러 및 하드웨어 가속기가 OpenGL ES 표준을 따르는 것은 아니다. 예를 들면, 전술한 미쯔비시의 모바일 3D 전용칩 Z3D는 OpenGL

ES와 다른 독자적인 API를 제공한다.

한편, 현재의 저사양 휴대폰에서 OpenGL ES 1.0 API 전체를 구현하는 데에는 상당한 무리가 따른다. 실제로 OpenGL ES에 기반하여 소프트웨어 렌더러를 개발한 국내 개발사, 대학 및 연구소에서는 제안된 API의 50% 안팎을 구현하는데 그쳤으며, 구현된 함수 각각도 모든 option을 지원하지는 않는다. 예를 들면, 라이팅 계산을 이용한 음영처리(shading)는 대부분의 휴대폰에서 현저한 fps 저하 현상을 가져오게 되므로, 이른바 음영처리된 텍스처(shaded texture)를 사용하는 경우가 많은데, 이런 경우 OpenGL ES API 중 라이팅 관련 함수들은 구현하지 않게 된다. 앞서 SKT-GL이 기가폰에서 1000개 다각형에 대해 7~8 fps의 성능을 보

인다고 언급했는데, 이는 음영처리 없이 텍스처 매핑만을 수행했을 경우의 성능이다.

5. 모바일 3D 게임 엔진

5.1 게임 엔진 개요 및 scene graph API

게임 엔진은 통상 “게임 소프트웨어들이 공유하는 모듈로 구성된 라이브러리” 및 “게임 구현을 지원하는 툴

(tool)”의 집합으로 정의된다. 게임 엔진을 구성하는 서브 엔진으로는 렌더링 및 애니메이션 엔진, 사운드 엔진, 물리 엔진, 인공지능 엔진 등이 있으며, 게임에 필요한 각종 데이터를 읽고 편집하고 저장하는 기능을 제공하는 툴은 맵/지형 에디터, 캐릭터 및 애니메이션 에디터 등이 있다. 대표적인 데스크탑 용 게임 엔진은 Quake[15]와 Unreal[11]이 있다. 얼마 전 출시된 리니지2는 Unreal 엔진을 이용해 개발되었다.

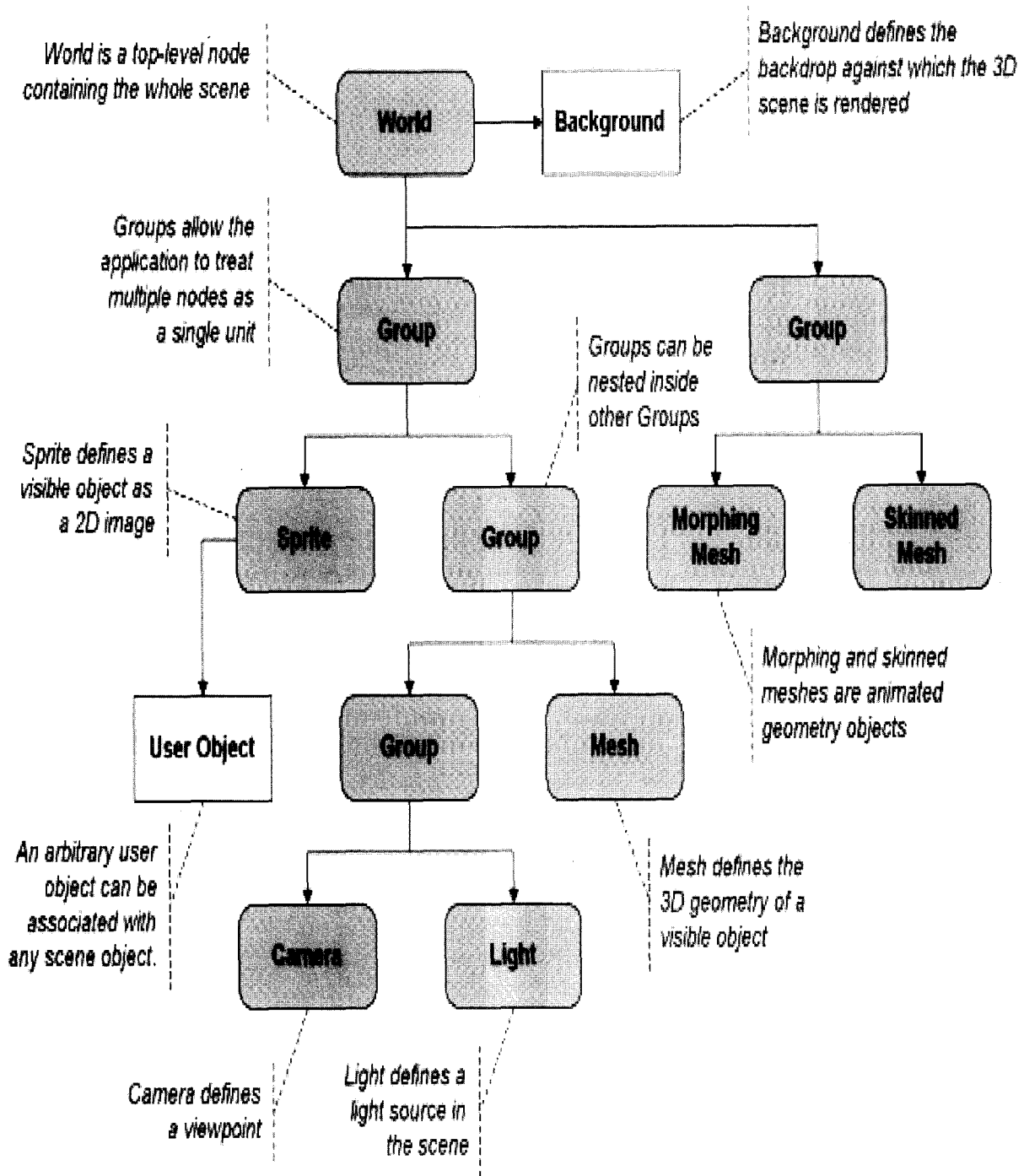


그림 6 Scene graph 예 [20]

게임 엔진의 핵심 개념 중 하나는 scene graph이다. 하나의 장면(scene)을 렌더링하는데 필요한 모든 데이터를 계층적으로 저장해 놓은 것이 scene graph로서, scene을 구성하는 물체의 기하적인 정보와 이에 대한 속성, 그리고 이를 렌더링하는데 필요한 카메라 정보 등을 그래프(DAG: directed acyclic graph) 구조로 표현한다. 그림 6은 scene graph의 하나의 예를 보여 준다. 사용자와의 상호작용, scene을 구성하는 물체 간 충돌 등에 따라 매 프레임 scene graph는 수정되며, 렌더링 시스템은 이렇게 수정된 scene graph를 순회하며 렌더링을 수행한다. Scene graph 작성, 수정, 순회 등에 관련된 API들을 총칭하여 scene graph API라 한다.

모바일용 scene graph API로 Java Community Process(JCP)에 의해 제안된 JSR 184(Java Specification Requests 184)가 있다[20]. JSR 184는 J2ME/CLDC 용 3D API로, MIDP와 함께 사용되는 optional package의 하나로 제안되었다(그림 1 참조). Scene graph API는 통상 high-level API로 분류되며 실제 렌더링을 담당할 low-level API를 필요로 하는데, JSR 184는 OpenGL ES를 기반으로 설계되었다.

OpenGL ES와 마찬가지로, JSR 184 역시 전체를 구현하는 것이 어렵거나 불필요할 수 있다. 또한 Java 이외의 언어로 게임을 개발할 경우에는, 단지 하나의 참조 모델로 JSR 184를 국한시키기도 한다.

5.2 상용 게임 엔진

잘 알려진 모바일 3D 게임 엔진으로는 Fathammer사의 X-Forge[12]와 Superscape사의 Swerve[33] 등이 있다. 이들 엔진은 PDA 및 고사양의 휴대폰(스마트폰) 용 게임 개발에 사용되고 있다. X-Forge는 Symbian, Palm 등과 같은 다양한 플랫폼에서 사용될 수 있는 C++ 기반 엔진이며, Swerve 역시 다양한 플랫폼에서 Java와 C/C++ 프로그래밍을 지원한다. X-Forge와 Swerve를 이용한 다수의 모바일 3D 게임이 이미 시장에 선보였는데, 그림 7은 그 중 일부의 스크린 샷이다.

외국에서는 PDA와 전용 게임기(N-Gage 등)와 같이 상대적으로 고사양을 가진 모바일 기기에서의 3D 게임 개발이 주류를 이루고 있다. 하지만, 이와 같은 기기의 국내 시장 점유율은 매우 미약하여, 국내의 모바일 3D 게임 개발은 휴대폰에 집중되고 있다. 상대적으로 저사양을 가진 휴대폰에서의 3D 게임 구현을 위한 많은 노력이 ㈜와우포엠(NF3D)[36], ㈜고미드(G3)[13], ㈜리코스(ReakoLite M3D)[28], ㈜디지털아이엔터테

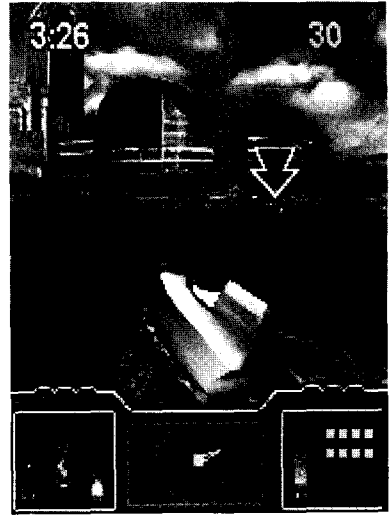
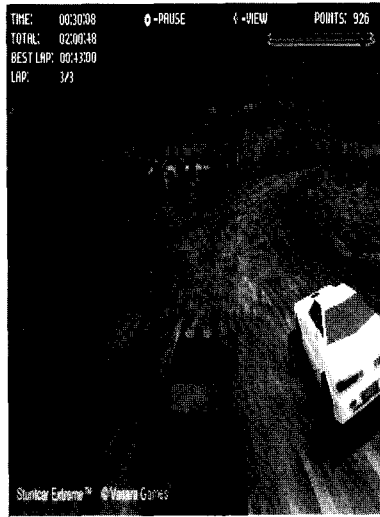
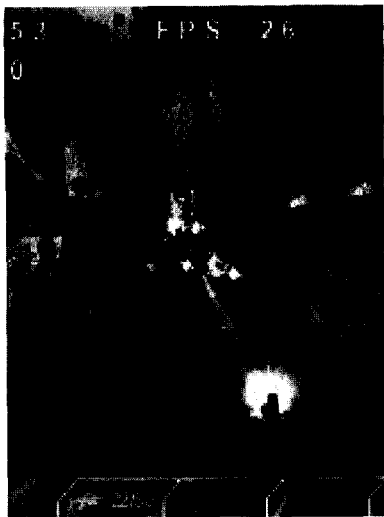
인먼트(QT)[10] 등의 선도적인 업체 중심으로 계속되어 왔으며, 이들 업체들은 기간 통신 업체, 단말기 또는 칩 제조 업체 등과 긴밀한 협력 관계 하에 게임 엔진을 개발하여 왔다.

이러한 노력의 결과로 2003년 하반기부터 SKY IM-6400 휴대폰과 삼성전자의 3D 모바일 게임폰(SPH-X9300) 등과 같이 3D 게임을 구동할 단말기가 출시되기 시작하였으며, 전술한 바와 같이 2004년에는 3D 가속기가 장착된 휴대폰이 출시될 예정이다. 이러한 환경 변화에 따라 (주)게임빌과 같은 모바일 게임 선두 업체를 비롯한 몇몇 개발사에서 3D 게임을 시장에 선보이고 있다.

(주)고미드의 게임 엔진 G3 SDK를 이용해 3D 게임 제작 단계를 간략히 살펴보자. 그림 8-(a)에 도시된 바와 같이, G3 SDK는 크게 (1) 게임 캐릭터 제작을 위한 3ds max exporter, (2) 게임 리소스를 관리하기 위한 utility, (3) 게임 애플리케이션이 탑재되는 G3 game platform, 그리고 (4) PC용 에뮬레이터 등으로 구성된다. 게임 개발자는 3ds max를 이용하여 게임에 필요한 각종 오브젝트를 생성하고 애니메이션을 정의한다. 3ds max exporter에서는 3ds max로 작업한 오브젝트의 정보를 G3에서 사용하는 형태로 저장한다. 3D 오브젝트는 기본적으로 다각형 메쉬로 표현되며, 2D 이미지는 2m*2n 사이즈의 jpg, tga, bmp 등으로 표현된다. 애니메이션은 3ds max에서 제공하는 키 프레임(key frame) 애니메이션을 기본으로 한다. 키 프레임 애니메이션이란, 애니메이션을 정의하는 전체 프레임 중에 특정의 몇몇 프레임들(키 프레임)에 오브젝트의 변형(움직임)을 각각 정의하고, 키 프레임들을 보간(interpolation)하여 중간 프레임들을 생성하는 기법이다.

G3 utility는 3ds max exporter에서 생성한 오브젝트 정보와, 각종 2D 이미지 파일(jpg, tga, bmp) 그리고 사운드 파일(wav) 등의 리소스를 일괄적으로 관리하는 툴로, 각 리소스를 타겟 모바일 플랫폼에서 구동 가능한 형태로 가공한다. (현재 G3 utility에는 scene graph editing 기능이 없지만, 곧 추가될 예정이다.)

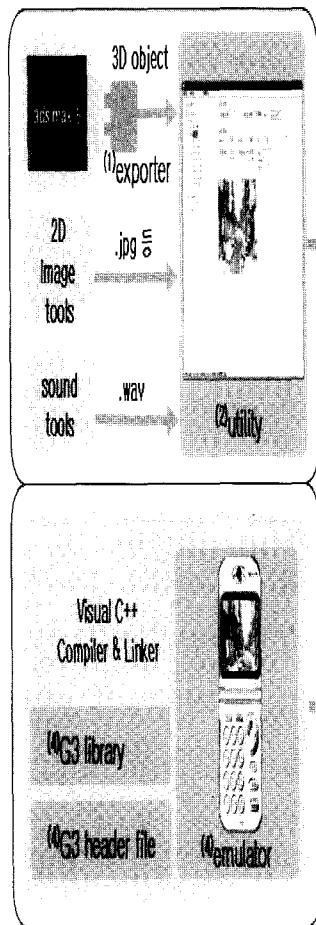
G3 애플리케이션은 Visual C++ 환경에서 프로그래밍할 수 있는데, G3 SDK에서 제공되는 G3 라이브러리와 헤더 파일, 에뮬레이터를 이용하여, 테스트 및 디버깅이 가능하다. 그림 8-(b)는 에뮬레이터를 이용한 3D 장면 스크린 샷을 보여준다. 마지막으로, 게임 애플리케이션을 모바일 기기에 탑재하기 위해서는 타겟 디바이스의 컴파일러를 이용하여 바이너리 파일을 생성하면 된다.



(a) X-Forge

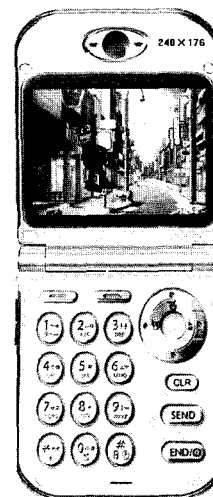
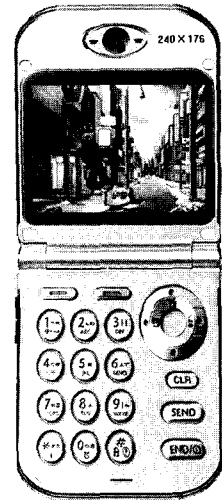
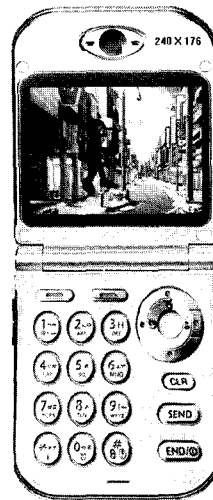
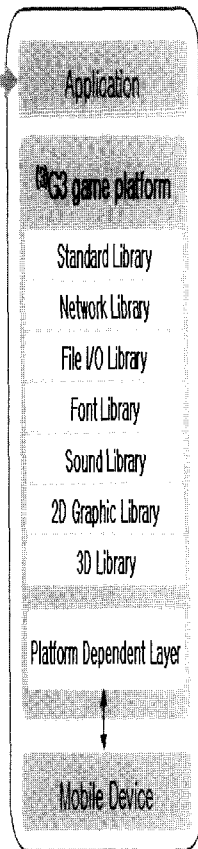
(b) Swerve

그림 7 X-Forge 및 Swerve 엔진 기반 게임 스크린 샷



리소스

실행 프로그램



(a) G3 SDK

(b) 에뮬레이터 스크린 샷

그림 8 G3 이용 게임 개발

6. 결론

지난 몇 년간 모바일 게임의 시장 점유율은 가파르게 상승하여 왔다. 본 논문에서는 2D 위주의 모바일 게임 개발 기술의 현황을 살펴보고, 이제 막 시장 진입 단계에 들어간 3D 게임 기술의 핵심 사항을 휴대폰을 중심으로 정리해 보았다.

소형 액정 화면, 저성능 프로세서, 소용량 메모리 등으로 대표되는 모바일 기기의 하드웨어 사양은 PC와 콘솔 게임기(Xbox와 Playstation2 등)의 그것과는 근본적으로 다르다. 따라서, 이미 3D 게임이 주류가 된 PC/콘솔 게임기와는 달리, 모바일 기기에서는 당분간 2D 게임이 우월한 지위를 차지할 것이 틀림없다.

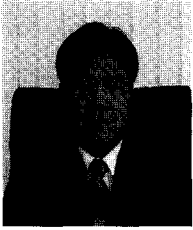
PC에서 3D 게임이 대중화된 것은 거의 전적으로 Nvidia와 ATI 등의 그래픽스 카드 덕분이었다. 마찬가지로, 모바일 3D 게임의 성장은 모바일 3D 가속기 기술 발전과 시장 형성에 달려 있다. 모바일 3D 가속기가 언제 어떤 양상으로 단말기에 대중적으로 장착될 지는 정확히 예측할 수 없다. 다만, 분명한 것은 이 흐름은 현재 매우 빠르게 진행되고 있다는 것이다. 최근 Nvidia가 모바일 3D 가속기 제조사인 MediaQ를 인수해 모바일 시장에 대한 공략을 개시한 것, 그리고 OpenGL ES 공개에 따라 국내외 유수의 단말기 제조사들이 일제히 3D 가속 단말기 개발에 착수한 것은 우리에게 많은 시사점을 던져 준다. 머지 않은 장래에 모바일 기기에서 2D 게임과 3D 게임이 양립할 시기가 올 것으로 보인다.

참고 문헌

- [1] 2003 대한민국 게임백서, (재)한국게임산업개발원, 2003.
- [2] 3D Graphics LSI Core for Mobile Phone Z3D, Graphics Hardware 2003, <http://www.graphicshardware.org/program.html>
- [3] 그래픽 가속칩, 모바일 3D 최적구현 길라잡이, 모바일컴 10월호, 2003.
- [4] 대한민국 WIPI관련 강좌란, <http://www.mobilejava.co.kr/bbs/temp/lecture/wipilec.htm>
- [5] 한정현, 모바일 게임 기술: 현황과 전망, 한국정보과학회 가을 학술발표회 튜토리얼, 2003.
- [6] Bitboys Acceleon, <http://www.acceleon.com>
- [7] 천귀호, BREW 모바일 프로그래밍, 한빛미디어, 2002.
- [8] 모바일브루(www.mobilebrew.net) 운영진, BREW Mobile Programming, 영진닷컴, 2002.
- [9] OpenGL ES Overview, <http://www.khronos.org/opengles>
- [10] Digital Eye Entertainment QT, <http://www.dieye.com>
- [11] Epic Games, <http://www.epicgames.com>
- [12] Fathammer X-Forge, <http://www.fathammer.com/x-forge/index.shtml>
- [13] Gomid, <http://www.gomid.com>
- [14] HI Mascot Capsule Engine, <http://www.mascotcapsule.com>
- [15] Id software, <http://www.idsoftware.com>
- [16] Intel Graphics Performance Primitives, <http://www.intel.com/design/pca/applicationsprocessors/swsup/gppv1.htm>
- [17] Intel PCA Overview, <http://www.intel.com/pca/developernetwork/overview/>
- [18] Java2™ Platform, Micro Edition, <http://java.sun.com/j2me/docs/j2me-ds.pdf>
- [19] MediaQ, <http://www.mediaq.com>
- [20] Mobile 3D Graphics API for J2ME, <http://www.jcp.org/en/jsr/detail?id=184>
- [21] Nazomi Communications, <http://www.nazomi.com>
- [22] NexusChips ISKRAII, <http://www.nexuschips.com>
- [23] Nokia N-Gage, <http://www.n-gage.com>
- [24] OpenGL, <http://www.opengl.org>
- [25] PocketGear PocketGL, http://www.pocketgear.com/software_detail.asp?id=1858
- [26] Qualcomm BREW, <http://www.qualcomm.com/brew>
- [27] Qualcomm Q3Dimension™ Gaming Platform, <http://www.cdmatech.com/solutions/products/q3dimension.jsp>
- [28] Reakosys ReakoLite™ M3DIM Engine, <http://www.reakosys.com>
- [29] Tomas Akenine-Möller and Eric Haines, Real time Rendering, A K Peters, Ltd., 2003.
- [30] Sanshin Electronics GSHARK, <http://www.gshark.com>
- [31] SinjiSoft GNEX, http://www.sinjisoft.co.kr/html/gnex_gnex.htm
- [32] SK-VM 2.x, <http://developer.xce.co.kr>
- [33] Superscape SWERVE, <http://www.superscape.com>

[34] Symbian OS, <http://www.symbian.com>

한 정 현



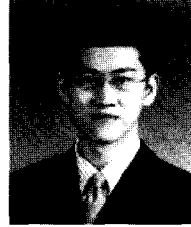
1988 서울대학교 컴퓨터공학과 학사
1991 미국 University of Cincinnati,
Department of Computer Science
석사
1996 미국 University of Southern Cali
fornia, Department of Computer
Science 박사
1996~1997 미국 상무성 National Insti-
tute of Standards and Tech-

nology (NIST) Manufacturing Systems Integration
Division 연구원
2002 미국 Drexel University, Department of Computer
Science 연구조교수
1997~현재, 성균관대학교 정보통신공학부 컴퓨터공학전공 부
교수
2001~현재, 산업자원부 지정 게임기술개발 지원센터 (성균관대
소재) 센터장
관심분야 : computer graphics, realtime rendering 등
E-mail : han@ece.skku.ac.kr

[35] Texas Instruments OMAP, <http://www.ti.com/sc/omap>

[36] WOW4M NF3D, <http://www.wow4m.com>

김 태 성



2001 성균관대학교 전기전자 및 컴퓨터공학부
학사
2002 미국 Drexel University, Depart-
ment of Computer Science 방문
연구원
2004 성균관대학교 정보통신공학부 컴퓨-
터공학전공 석사
2004~현재 LG전자기술원 Mobile Multi-
media 연구소 연구원

관심분야 : mobile 3D graphics, geometric modeling, real-
time rendering, collaborative design 등
E-mail : falcons@ece.skku.ac.kr

The 14th Joint Conference on Communications & Information (JCCI 2004)

- 일 자 : 2004년 4월 28 ~ 30 일
- 장 소 : 금호 충무 마리나 리조트(충무)
- 주 최 : 정보통신연구회
- 상세안내 : KAIST 이용훈 교수(Tel. 042-869-4411)
<http://www.jcci21.or.kr>