모바일 단말을 위한 인터렉티브 미디어 시스템의 확장

한승진, 류은석, 유 혁
고려대학교 컴퓨터학과
e-mail : sjhan@os.korea.ac.kr

An Extension of Interactive Media System for Mobile Device

Seung-Jin Han, Eun-Seok Ryu, Hyuck Yoo
Dept. of Computer Science and Engineering, Korea University

요 약

현재의 세계적인 트렌드인 HCI(Human Computer Interaction)에서 사용자의 기호나 의견 등을 반영하는 인터렉티브 미디어(Interactive Media)는 빠질 수 없는 주제다. 본 연구팀은 모바일 단말환경에서 사용자의 인터렉션을 통한 서비스를 제공할 수 있는 IMS(Interactive Media System)를 설계하고, 이를 PDA 상에 구현하였다. 기존의 연구들이 보여주는 링크의 형태로만 미디어를 지원하는 방식은 CPU 등의 자원이 부족한 모바일 환경에서는 부담이 될 수 있다. IMS 는 이를 벗어나 내부적으로 미디어 오브젝트를 지원하는 방식을 사용하여 모바일 환경에 적합하게 연산속도를 개선하고 있다. 또한 이러한 방식으로 인하여 생길 수 있는 문제인 미디어 포맷의 지원에 대한 제약을 극복하기 위해 확장성 있는 구조로 설계되어 이미지와 텍스트, 백터그래픽 만을 제공하던 단순한 시스템에서 H.264 와 MPEG4 AAC 와 같은 여러 모듈들이 더해졌다. 또한 OpenGL 모듈이 추가되고 3D 오브젝트들이 새롭게 정의됨으로써 IMS 는 IML 을 통해 마크업 언어차원에서 3D 그래픽을 지원할 수 있게 되었고 2D 와 3D 를 함께 사용한 다양한 비쥬얼 구성이 가능하게 되었다.

본 논문에서는 IMS 의 확장성 있는 구조와 OpenGL 을 추가하고 새로운 미디어 오브젝트를 정의하는 과정 등을 다루며 언급한 내용을 자세히 소개한다.

1. 서론

유비쿼터스라는 말이 낯설지 않은 시대의 흐름과는 달리 대부분의 인터렉티브 미디어에 대한 연구는 아직도 PC 나 셋탑박스와 같은 유선환경에 편중되어 있다. 모바일 단말이 보급화되고 성능이 개선되었지만 이러한 유선상의 시스템을 모바일 단말환경에 직접 적용시키기에는 무리가 있다. 본 연구팀은 이러한 문제점을 해결하기 위해 모바일 단말환경에 적합한 인터렉티브 미디어 시스템(IMS)을 설계하고 이를 PDA 상에 구현하였다. 무선 단말의 상대적으로 적은 리소스는 기존의 SMIL[1]과 같은 시스템이 보여주는 미디어를 링크하는 방식이 부담이 될 수 있다. IMS 에서는 내부적으로 미디어 오브젝트를 지원하는 방식을 사용하여 오버헤드를 줄이며 동시에 각각의 미디어 오브젝트에 대한 직접적인 제어성을 높이고 있다. 그러나 내부적으로 오브젝트를 지원하는 방식은 다양한 미디어 포맷의 지원에 제약점이 될 수 있다. 새로운 형태의 미디어가 계속 개발되고 있고 사용자의 요구가 다양해지는 시점에 한정적인 미디어 포맷만을 지원하게 되는 결과는 바람직하지 않다. IMS 는 이를 극복하기 위해 확장성을 염두에 두고 설계되었다. 최초의 비트맵 형식의 이미지 오브젝트와 파일 형식의 텍스트 오브젝트, 그리고 벡터그래픽만을 지원하던 단순한 형태의 시스템에서 WAV 형식과 MPEG-4 AAC 의 오디오, H.264 의 비디오까지 다양한 미디어 오브젝트의 재생이 가능하도록 확장되었다. 또한 사용자의 요구가 발전해감에 따라 우리는 3D 그래픽 지원의 필요성을 느끼고 3D 그래픽 모듈을 새롭게 추가하였다.

본 연구는 한국정보통신대학교 디지털미디어연구소의 정보통신연구개발사업의 연구비 지원에 의하여 수행되었음

본 논문에서는 먼저 IMS 의 전체적인 시스템을 살펴보고 OpenGL 모듈을 추가하며 3D 오브젝트를 정의하는 과정을 토대로 IMS 의 구조와 확장되는 과정에 대해 설명한다.

2. 관련연구

2-1 ETRI : MPEG-4 대화형 멀티미디어 방송

ETRI 에서 연구되었던 대화형 멀티미디어 방송기법 연구는 MPEG-4 의 BIFS 를 이용하여 진행하게 될 미디어 데이터의 시나리오 트리를 구성한다. 그리고 사용자와의 인터렉션을 위해 upchannel stream 을 사용한다. 하지만, 이 시스템은 TV 셋탑박스에서의 동작을 위해 설계되었고, 무선망과 같은 환경을 고려하지 않고 있다. 따라서, 본 연구에서 목표하는 환경인 PDA, 핸드폰 등의 무선 단말에서는 적용하기 힘들다[2].

2-2 SMIL

SMIL(Synchronized Multimedia Integration Language)은 인터넷 방송과 같이 멀티미디어 자료를 처리할 때, 쌍방향 서비스의 지원을 위해 개발되었다. 이는 XML 에 기반하여 HTML 과 아주 유사한 구조를 가지고 있다. 이에 대한 규약은 W3C 에서 담당하였고, SMIL 2.0 까지 정리되어있다[3]. PocketSMIL 과 같은 모바일 환경을 위한 시스템들이 나와있지만 기본적으로 미디어를 링크하는 방식으로 재생되어 모바일 환경에서 만족할 만한 성능을 보여주지 못하고 있다[4].

3. Interactive Media System

기본적인 IMS 의 구성은 그림 1 과 같다.

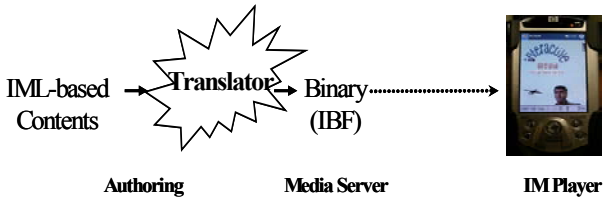

그림 1. Interactive Media System 의 구성

IMS 는 크게 IML, Intermediate Translator, IM Player 로 구성되어있고 각 파트는 밑에서 간략히 소개한다.

3-1 Interactive Media Language

본 연구팀은 SMIL 에 대한 연구를 통해 모바일 환경에 적합한 IML 을 정의하였다. IML 은 설계 과정에서부터 경량 단말기 환경을 고려하였고, 다양한 사용자 인터렉션을 제공하기 위해 정의 되었다. IML 은 SMIL 과 유사한 형태로 사용하기 쉽고 멀티미디어 데이터의 시간적, 공간적 동기화를 제공한다. 하지만, 벡터그래픽과 같은 미디어 오브젝트를 파일로 링크하는 것이 아닌 직접 내포(Embed)하고 있으며 이러한 오브젝트에 대한 직접적인 컨트롤이 가능하도록 설계되었다. 또한 IML 은 밑에 제시한 것처럼 모듈화되어 설계되었기 때문에 확장에 용이한 구조를 가지고 있다.

- Structural and Media Object Declare Module
- Animate Module
- Content Control Module
- Prefetch Control Module
- Layout Module
- Linking Module
- Media Object Module
- Meta-information Module
- Time Manipulation Module & Transition Effects Module

IML 의 구체적인 명세 및 언어의 설계에 대한 세부적인 사항은 앞서 발표된 논문에 자세히 실었다[5].

3-2 Intermediate Translator

Intermediate Translator 는 IML 로 작성된 컨텐츠를 재생하는 플랫폼이 상대적으로 리소스가 부족한 모바일 단말기라는 점을 조금이나마 극복하기 위해 설계되었다. XML 형태로 작성된 마크업 언어 파일을 저작시에 미리 바이너리화 함으로써 재생시에 불필요한 XML 파싱(Parsing)을 제거하였다. 이는 클라이언트 측에 XML 파서 모듈이 불필요하게 해줄 뿐 아니라, 전달되는 파일 사이즈를 줄이고, 연산을 위한 CPU 자원도 절약할 수 있게 해준다. 그리고 바이트(byte)단위의 정렬을 통해 비트(bit)연산을 줄이도록 노력하였다.

3-3 Interactive Media Player

IM 플레이어의 구조는 그림 2 와 같다. IML 로 저작되고 바이너리 포맷으로 변형되어 전달된 미디어 데이터는 재생을 위해 내부에서 다시 파싱작업을 거친다. 이 작업을 하는 IBF(Intermediate Binary Format) 파서 모듈은 텍스트 기반을 처리하는 파서처럼 각각의 테이블을 유지할 필요가 없기 때문에 경량 단말기환경에서도 빠르게 동작할 수 있다. Object Scheduler 는 파싱된 오브젝트들을 Waiting List 와 Current List 로 관리하며 동기화를 제공한다. Current List 에 옮겨져 재생될 미디어 오브젝트는 벡터 데이터는 렌더링되고, 그 외 비디오 및 오디오 데이터는 각각에 맞는 코덱 디코더에 전달되어 처리한다. IM Player 에 대한 더욱 자세한 내용은 전에 발표된 논문을 참조한다[6].

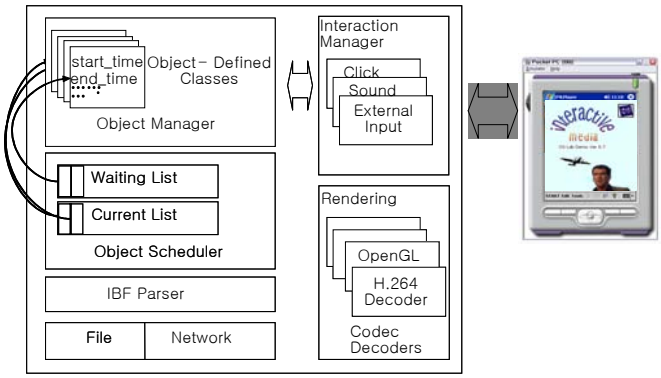

그림 2. 인터렉티브 미디어 플레이어의 내부 구조

4. IMS 의 구조와 확장성

위에서 언급했듯이 IMS 는 확장이 용이하도록 모듈별로 구현되었다. 새로운 미디어 오브젝트를 추가할 때의 과정은 다음과 같다.

(1) Structural and Media Object Declare Module 과 Media Object Module 에 Object ID 를 정의한다.
(2) 추가될 미디어 오브젝트의 클래스를 만들고 Render 에 실행루틴을 추가한다.
(3) 라이브러리를 포팅해서 코덱 디코더에 넣는다.

확장에 있어서 다른 모듈에 신경을 쓸 필요가 없기 때문에 필요에 따라 위의 과정을 거쳐 간편하게 새로운 오브젝트의 추가가 가능하다. 미디어 오브젝트를 추가하는 과정은 5 장에서 다시 살펴보기로 하고 먼저 Codec Decoders 에 대해 자세히 소개하도록 한다.

4-1 Codec Decoders

IMS 에서의 코덱 디코더는 부호화된 데이터를 받아 복호하는 디코더와 형식화된 데이터를 가지고 명시적인 표현으로 바꾸어주는 라이브러리, 또는 프로세스를 합쳐서 지칭한다. 이것은 IMS 의 확장성 지원에 있어서 핵심적인 부분이다. IMS 에서는 Object Management 와 디코딩, 그리고 화면에 디스플레이(Display)하는 과정을 분리하여 확장성을 높이고 있다. 각각의 미디어 오브젝트는 해당되는 디코더에 넘겨질 때까지 완전히 독립적으로 관리되기 때문에 필요한 라이브러리들이 다른 구현내용에 신경 쓰지 않고 정의된 인터페이스에 따라 모듈화되어 쉽게 추가 될 수 있다. 디코딩된 데이터는 독립적으로 처리되어 최종적으로 공유된 화면 메모리에 뿌려지게 된다. 이렇게 스칼라 그래픽(Scalar Graphic)과 벡터 그래픽(Vector Graphic)이 하나의 화면메모리를 공유함으로써 속도를 높이고 자원을 절약할 수 있다. 현재 IMS 에서 모듈로서 추가되어 지원되는 미디어 오브젝트의 종류는 다음과 같다.

- H.264/MPEG4 Part-10
- AAC(Advanced Audio Codec)
- WAV Audio
- Vector Graphics
- Bitmap Image
- Text Stream

이 중 비디오 코덱 H.264 는 모바일 환경에 알맞은 Baseline Profile 을 지원하고 있으며, 오디오 코덱 AAC 역시 LC(Low Complexity) Profile 로서 지원하고 있다. Vector Graphic 렌더러는 SVG 와 비슷한 형태로 간단한 벡터 그래픽을 표현할 수 있도록 도와준다[7].

이와 같이 IM Player 가 지원하는 여러 미디어 코덱들은 라이브러리 형태로 존재하며, 각각의 코덱들은 PDA 에서 돌아갈 수 있도록 여러 최적화 작업을 통해 포팅되었다. 라이브러리들은 IM Player 내부에서 정한 표준적인 API 를 통해 호출되며, 크게 보아 Init, Decode, Close 의 세 단계 함수로 구현되어 있다.

5. IMS 의 확장

5-1 OpenGL Module

OpenGL SGI(Silicon Graphics)에서 개발된 2 차원 및 3 차원 그래픽 이미지를 정의하기 위한 표준 API 의 집합을 말한다. OpenGL ARB(Architecture Review Board)에서 주기적으로 표준을 개정하고 있으며 현재 2.0 specification 까지 발표되었다[8].

본 연구에서는 IMS 상에서 OpenGL 지원을 위한 라이브러리로 Mesa 3D Graphics Library 를 적용하였다. 이는 공개된 라이브러리로 최근 발표된 Mesa 6.0 버전이 SGI 의 OpenGL 1.5 specification 까지의 호환성을 제공한다[9]. 적용된 라이브러리는 PDA 환경에 적합하도록 포팅을 거쳐 가볍고 빠른 실행속도를 보여주고 있다.

5-2 Module ID 의 추가

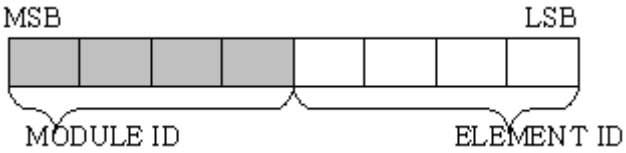

그림 3. Mudule ID 의 구조

IMS 는 모든 오브젝트를 그림 3 과 같은 형태의 고유한 모듈 ID 로서 관리한다. 새로운 미디어 오브젝트를 추가할 때는 2 가지 모듈에서 새로운 Element ID 를 정의하는 과정이 필요하다. 첫번째는 Structure and Media Object Declare 모듈로 문서 구조와 미디어 오브젝트의 초기선언부분을 표현하고 있고 0x00 부터 시작한다. 두번째는 Media Object 모듈로 선언된 미디어 오브젝트를 IML 컨텐츠 안에서 Reference ID 를 이용, 참조하여 사용되는 모듈이며 0x60 부터 시작한다. 본 연구에서 추가된 예인 Cube Object 에는 Element ID 값으로 0x0C 와 0x68 의 값을 부여하였다.

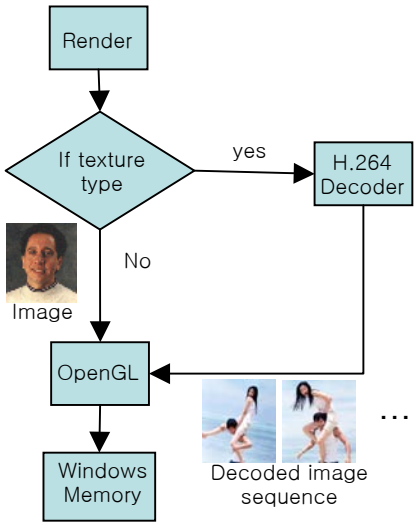

그림 4. 간략하게 표현된 Cube Object 의 재생과정

5-3 클래스 선언과 실행 루틴

추가되는 오브젝트는 클래스로 선언되고 렌더러에 실행 루틴(Routine)을 추가하는 과정이 필요하다. Cube 오브젝트의 경우 클래스정의에서 공통적인 요소들을

선언하고 렌더러에서 오브젝트에 입힐 텍스쳐(Texture)의 종류에 따라 분기하는 루틴이 추가되었다. Object Manager 에서 넘어온 오브젝트 데이터는 명시된 값에 따라 각각 다른 형태의 텍스쳐를 입히기 위해 진행되게 된다. 그림 4 는 Cube Object 가 생성되어 재생되는 경우를 간략하게 표현하고 있다. 다음은 이 그림을 참조하며 진행과정을 상세하게 설명한다.

5-3-1 초기화과정

어떤 형식의 포맷이 텍스쳐로 주어지던 간에 공통적으로 수행되는 부분은 OpenGL 라이브러리를 사용하는 객체의 초기화 과정이다. 초기화 과정의 수행 내용은 다음과 같다. 우선, 정점(Vertex)과 법선(Normal), 삼각형(Triangle)을 이용해 Cube 개체를 생성한다. 개체가 생성되면 시점과 투영방식, 광원 등을 설정하는 API 들이 호출된다.

5-3-2 Texturing and Display

일반 BMP 포맷이나 벡터그래픽 같은 고정적인 이미지가 텍스쳐로 입혀지는 과정은 다음과 같다. Cube 개체의 초기화를 거치면 바로 TextureLoad 루틴으로 넘어가고 OpenGL 의 API 를 통해 텍스쳐를 입히는 작업이 진행된다. 그리고 최종적으로 화면 메모리로 전해져 디스플레이 된다. 반면에 텍스쳐로 비디오가 선택된 경우 텍스쳐를 만들기 위해 먼저 H.264 디코더를 거치게 된다. 자세한 진행 과정은 다음과 같다.

(a) 컨텐츠에 명시된 비디오 파일을 초기화
(b) H.264 디코더에서 YUV 형식으로 프레임생성
(c) 생성된 프레임을 다시 RGB 형식으로 변환
(d) 이미지의 경우와 같이 OpenGL 의 초기화 수행
(e) TextureLoad 루틴 수행
(f) 공유화면 메모리에 전해져 디스플레이

비디오의 경우 재생이 되면서 텍스쳐 정보가 계속 새로운 프레임으로 갱신되어야만 한다. 따라서 처음에 위 과정을 거친 후에는 루프를 돌며 (b),(c),(e),(f)의 과정을 반복하여 수행하게 된다. 이 외에도 오브젝트에 회전이나 크기조정과 같은 효과를 줄 경우 (d)와 (f)의 과정이 재수행 될 수 있다.

6. 결론 및 향후 과제

본 연구에서는 모바일 단말에 적합한 인터렉티브 시스템의 미디어 지원방식에 대한 해결방법을 제시하고 이를 과정에 따라서 분석해보았다. IMS 는 확장성 있는 설계로 시스템 내부적으로 미디어를 지원하여 오버헤드를 줄이면서 동시에 다양한 미디어 포멧의 지원요구를 충족시키고 있다. 모듈화된 설계에 따라 미디어 오브젝트를 정의하고 필요한 라이브러리의 포팅, 그리고 실행 루틴을 추가하는 과정으로 새로운 미디어 오브젝트가 간편하게 추가될 수 있다.

이런 방식으로 다양한 미디어 오브젝트가 추가됨에 따라 IMS 점점 확장되어 현재는 텍스트, 이미지, 오디오, 비디오, 벡터그래픽까지 다양한 미디어 포맷을 지원하고 있다. 게다가 확장의 예로 든 OpenGL 모듈의 추가로 인해 2D 와 3D 를 함께 사용한 표현이 가능하게 되었으며 특히, 추가된 3D 오브젝트는 이미지뿐만 아니라 비디오까지 텍스쳐로 사용할 수 있는 강력한 비쥬얼 기능을 제공한다. 그리고 마크업 언어차원에서 3D 를 지원하기 때문에 사용자 측면에서는 OpenGL 에 대해 자세히 알지 못해도 편리하게 3D 오브젝트를 이용한 컨텐츠의 저작이 가능해진다. 이러한 확장과정을 통해 IMS 에서 다양한 볼거리를 제공하는 컨텐츠 구성이 가능해지고 응용 분야가 넓어진다. 예를 들면 3D 영상이 가미된 인터렉티브 시네마[10] 또는, 애니메이션의 저작이 가능해지고 인터렉티브 게임도 더욱 다양한 비쥬얼로 구성하여 제작할 수 있다.

현재 다자간의 인터렉션을 지원할 수 있도록 무선망에 적합한 스트리밍 모듈을 추가하는 작업을 계획중에 있어 지속적인 인터렉티브 미디어에 관한 연구를 통해 활용 범위를 계속 확대해 나갈 것이다.

그림 5. 2D 이미지를 배경으로 Cube Object 상에서 비디오를 텍스처링하여 재생하는 장면

참고문헌

[1]Rutledge, L, “SMIL 2.0: XML for Web multimedia”, Internet Computing, IEEE vol.5, Sep/Oct 2001, 78-84.
[2]Kyuheon Kim, “MPEG-4 based authoring system for interactive broadcasting”, 2002. 8.
[3]“Synchronized Multimedia Integration Language(SMIL 2.0)”, W3C Recommendation, 7 August, 2001, http://www.w3.org/TR/smil20/
[4]http://www.w3.org/AudioVideo/, “Synchronized Multimedia”
[5]윤민홍, 류은석, 유혁, “모바일 환경을 위한 대화형 언어의 정의”, 정보과학회 춘계 학술발표논문집, 2003.
[6]Eun-Seok Ryu, Chuck Yoo, “An Approach to Interactive Media System for Mobile Devices”, Proceedings of the 12th ACM International Conference on Multimedia (MM 2004), Oct, 2004.
[7]http://www.w3.org/Graphics/SVG, “Scalable Vector Graphics(SVG)”
[8]http://www.opengl.org, “Open Graphics Library”
[9]http://www.mesa3d.org, “Mesa 3D Graphics Library”.
[10]http://ic.media.mit.edu, “Interactive Cinema”.