특집 04

TTS기반 모바일 3D 토킹헤드 시스템 구현

목 차

1. 서 론
2. 토킹헤드 동기화 기법
3. 한글 음소간 합성 규칙
4. 모바일 토킹헤드 시스템
5. SAPI 한글 음소 맵핑
6. 실험 결과
7. 결 론

김상완 · 박순영 · 최경호
(목포대학교)

1. 서 론

토킹헤드 시스템(Talking-head systems)이란 얼굴 애니메이션에 말하는 기능이 추가된 시스템을 의미한다. 토킹-헤드 기술은 가상현실, 3-D 게임, 비디오 메신저, 스토리텔링, 휴먼-컴퓨터 인터페이스 등의 분야에서 중요성이 더욱더 강조되고 있으며, 유비쿼터스 환경에서 보다 현실감 있는 서비스를 제공하기 위한 핵심 기술의 하나이다[1-6]. (그림 1)은 다양한 멀티미디어 응용 시스템에서 활용되고 있는 토킹-헤드 기술의 적용 예를 보여준다.

토킹-헤드 서비스를 제공하기 위해서는 우선 2-D 혹은 3-D 얼굴 모델을 생성해야 하며 생성된 얼굴 모델을 음성신호 혹은 TTS(Text to Speech) 등을 기반으로 구동해야 한다.

3차원 얼굴 모델 생성은 토킹-헤드 시스템의 개발에 있어서 가장 먼저 개발해야하는 기술 중의 하나이다. 3차원 얼굴모델의 생성을 위한 다양한 기법 중에 대표적인 기법으로 다음의 3가지 기술이 있다. 첫째, 가장 단순한 방법으로 하나의 전면(Frontal view) 영상을 이용한 방법이

(그림 1) 토킹-헤드 기술의 적용 예

있다. 이 방법은 전면 영상 한 장만을 이용하여 3차원 모델을 생성하기 때문에 얼굴의 3차원 정보를 정확하게 복원할 수 없다는 단점이 있다[7]. 두 번째 방법으로는 스위스 제네바대학에서 제안한 전면과 옆면 영상 2장을 이용한 방법이

있다. 이 방법은 전면과 옆면 영상에서 얼굴 특징점의 위치를 사용자가 선정해 줌으로써 3차원 얼굴 형상을 비교적 정확히 재구성할 수 있다는 장점이 있다[8]. 마지막으로 마이크로소프트에서 제안한 방법은 첫 번째와 두 번째 방법과는 다르게 임의의 포즈를 취하고 있는 한 장의 영상만 있으면 데이터베이스에 있는 다수의 3차원 얼굴모델을 이용하여 가장 비슷한 3차원 얼굴모델을 생성하는 기법이다[9]. 첫 번째 방법과 두 번째 방법의 단점은 얼굴 영상이 정해진 포즈, 즉 정면과 측면을 바라보고 있어야만 한다는 단점이 있으며 세 번째 방법은 3차원 얼굴 데이터베이스의 구축이 필요하다는 단점이 있다. 본 연구실에서는 사용자가 카메라 앞에서 자연스럽게 움직이고 있는 동안에 자동으로 3차원 얼굴 모델을 생성하는 연구를 진행하고 있다. 이를 위해 사용자의 얼굴에서 MPEG-4에서 정의한 FDP (Facial Definition Points)에 해당하는 특징점을 자동으로 추출하고 추출된 특징점을 3차원 얼굴 생성 엔진에 입력하여 얼굴 모델을 생성하여 사용하였다.

토킹-헤드 시스템의 구동을 위해서 입력으로 사용할 수 있는 정보에는 사람의 라이브 음성 입력 혹은 TTS 엔진을 이용하여 텍스트를 기반으로 생성된 음소 및 viseme 정보가 있다. 라이브 음성에서의 입모양 파라미터 추출 기술은 음성정보에 숨어있는 비디오 정보(입모양 파라미터 등)를 추출하여 얼굴 애니메이션에 사용하고자 하는 기술로 다양한 분야에서 활용이 가능하다. 예를 들어, 대역폭이 작은 일반전화기를 이용하여 청각장애자에게 음성정보와 해당 음성정보에서 추출한 입모양 파라미터를 전달하면 그 정보를 이용하여 얼굴 애니메이션을 생성할 수 있고 이를 통해 청각장애자가 상대방의 대화를 이해할 수 있도록 도움을 줄 수 있게 된다. 이외에도 3차원 게임이나 비디오 메신저에서 나의 얼굴이나 혹은 나의 얼굴이 아닌 특정사람(예를 들어, 내가/상대방이 좋아하는 연예인)의 얼굴을 수신자에게 전달하고 나의 음성으로 얼굴 애니메이션을 제공하게 되는 것도 가능하게 된다. 음성신호에서 비주얼 파라미터를 추출하여 얼굴 애니메이션을 수행하기 위하여 벡터 양자화 기법[10], 신경망 이론[11], Hidden Markov Model Inversion (HMM)[12-13] 등의 여러 가지 방법이 있다. 본 논문에서는 모바일 환경에서의 한글 토킹-헤드 서비스 구현을 위한 아키텍처와 응용서비스에 대해서 제안하고자 한다. 구체적으로, Speech API(SAPI)를 이용해서 TTS를 사용할 때 생성되는 음소 정보와 발음 시간 등을 이용해서 오디오와 3-D 얼굴 모델 간 동기를 맞추는 방법을 소개하고, 모바일 기기에서 토킹헤드 서비스 구현을 위한 새로운 아키텍처를 소개하고자 한다. 모바일 기기는 연산량과 저장 공간 등이 데스크탑 컴퓨터보다 많이 작다. 이러한 문제를 극복하고자 클라이언트-서버 기반의 모바일 토킹헤드 시스템을 제안하고자 한다.

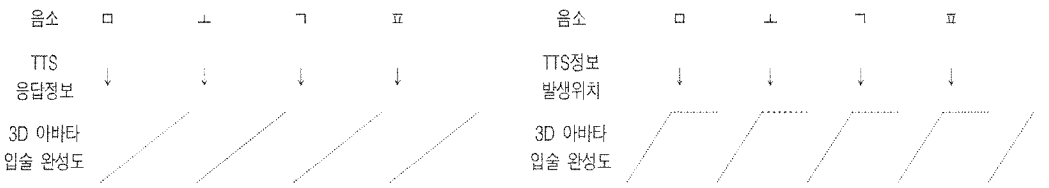
2. 토킹헤드 동기화 기법

SAPI를 이용해서 TTS기능을 사용하면 SAPI는 텍스트를 표준어 발음법에 따르는 음성으로 출력해주고 〈표 1〉과 같은 음소정보들을 제공한다.

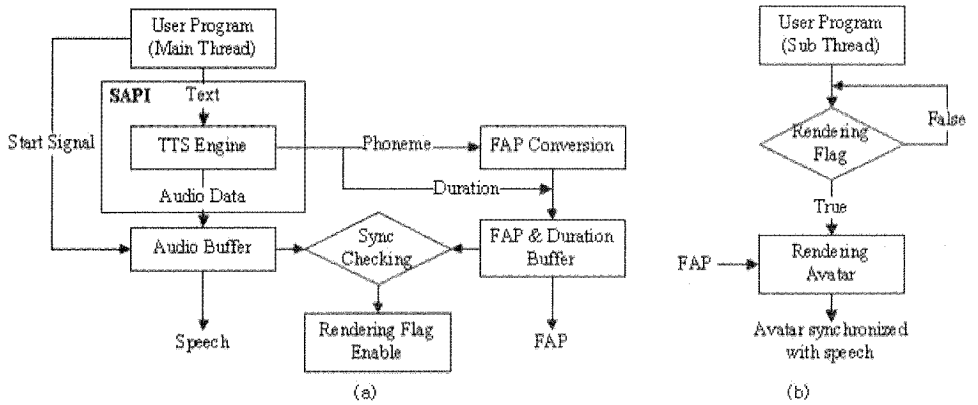
〈표 1〉 SAPI 에서 얻을 수 있는 음소정보들

Type	Note
Phoneme	입력 텍스트를 표준어 발음으로 발음된 음소
AudioStream Offest	Phoneme의 AudioStream에서 발음시작 위치
Viseme	Phoneme에 맵핑된 입모양 정보

위의 정보들은 음소하나당 얻을 수 있다. 입력된 문장(텍스트)에 대해 음소정보들은 AudioStreamOffset과 출력되는 음성의 위치가 일치되는 순간에 음소하나에 대해서만 실시간으로 얻을 수 있다.



(그림 2) 기존의 시스템과 제안된 시스템의 비교



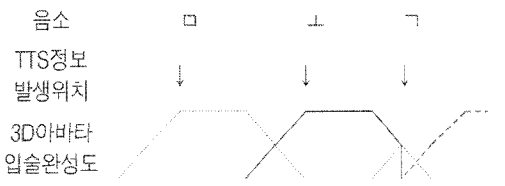
(그림 3) SAPI기반 토킹헤드 시스템 아키텍처 (a) 동기화 파라미터 생성 모듈, (b)렌더링 모듈

기존의 SAPI의 TTS기능을 이용하는 토킹헤드 시스템은 텍스트를 입력하면 바로 스피커를 통해 음성이 출력되고 획득한 음소정보를 사용하여 얼굴 애니메이션을 하도록 했다. 이러한 방식은 각 음소마다 맵핑된 한 장의 정지된 영상을 화면에 출력할 때에는 문제가 되지 않지만, 프레임을 이루어 애니메이션을 할 때에는 음성과의 동기문제가 생긴다.

(그림 2)의 왼쪽과 같이 음소정보를 얻는 위치에서 애니메이션을 하게 되면 음성의 출력은 시작되었지만, 애니메이션의 입모양은 아직 갖춰져 있지 않기 때문에 입술의 움직임이 음성에 비해 늦어보이게 된다. 이를 해결 하기위해서는 (그림 2)의 오른쪽과 같이 해당 음소가 발음되는 시간보다 수ms 이전에 애니메이션이 시작되어야한다. 하지만 앞에서 지적한 것과 같이 음소는 텍스트를 Speech할 때 실시간으로 획득할 수 밖에 없고, 실시간으로 발생되는 사건을 미리 알고 애니메이션을 진행하여야 한다.

(그림 3(a))의 SAPI 기반의 동기화 파라미터 생성 모듈은 텍스트가 입력되면 TTS engine은 AudioData를 버퍼에 담고 Phoneme을 FAP로 컨버팅한 파라미터와 AudioStreamOffset에서 구한 발음시간을 버퍼에 담는다. SyncChecking 부분에서는 버퍼에 담긴 AudioData를 재생하면서 재생시간과 음소의 AudioStreamoffset을 비교체크해서 음소에 대한 애니메이션 시작 시간을 지나게 되면 Rendering Flag를 활성화 해준다. (그림 3(b))는 토킹헤드에서 3D모델을 렌더링하는 모듈의 아키텍처이다. SAPI 모듈에서 Redering Flag를 활성화 해주면 해당 FAP를 이용해서 3D 아바타를 렌더링 해준다.

(그림 2(b))와 같이 파라미터를 적용할 경우 새로운 음소에 대한 viseme을 적용할 때마다 아바타의 입모양이 초기화된 후 렌더링이 진행되므로 굉장히 부자연스럽게 보이게 된다. 새로운



(그림 4) 렌더링 수치 변화그래프

FAP가 입력될 경우 이 전의 FAP에 대한 것은 바로 삭제하지 않고 점점 감소하도록 하면 자연스러운 토킹헤드의 입모양을 얻을 수 있다. 그리고 (그림 4)의 'ㄱ'의 음소 부분을 보게 되면 새로운 FAP값을 증가시키고 이전 FAP는 감소시키는 도중에 또 다른 새로운 FAP가 입력되었다. 이 경우 감소시키던 FAP(실선)는 바로 삭제하고 증가시키던 FAP(점선)를 감소시키기 시작하고 새롭게 들어온 FAP(파선)에 대해서 증가시키도록 하였다. 이외에도 자연스러운 애니메이션을 위해 Spline 함수 등을 이용한 기법 등이 활용될 수 있다.

제안된 시스템은 텍스트 전체에 대한 오디오 데이터를 생성해야 하며 오디오 생성시간이 텍스트에 길이에 비례하여 증가한다. 즉, 장문의 텍스트 일수록 오디오 데이터 생성에 필요한 전처리 시간이 길어지는 것이다. 이러한 시간을 줄이기 위해서 입력되는 텍스트 전체를 SAPI로 즉각 보내는 것이 아니라, 마침표(. ? !)단위로 텍스트를 잘라서 SAPI로 전달하도록 한다.

3. 한글 음소 간 합성 규칙

음소정보에 따라 미리 애니메이션을 시작하여 음성과 동기를 맞추면서 어느 정도 자연스러운 토킹헤드를 만들 수 있다. 하지만 모든 음소에 대해 애니메이션을 하도록 하면 오히려 부자연스럽게 보이게 되기 때문에 다음의 규칙을 이용하여 음소에 대해 필터링 하도록 한다.

※ 음소 필터링 규칙

ㄱ. 자음이 "ㄱ[g], ㅋ[k], ㅇ[-ŋ], ㅎ[h], ㄹ[l]"일 경우 표현하지 않는다.

ㄴ. 자음 "ㄴ[n], ㄷ[d], ㅌ[t], ㅅ[s] ,ㅆ[s*], ㅈ[ʧ], ㅊ[ʧ*]"은 앞이나 뒤에 모음 "ㅗ[o], ㅛ[jo], ㅜ[u], ㅠ[ju], ㅡ[ɨ], ㅣ[i], ㅢ[ɨi], ㅘ[wa], ㅙ[wæ], ㅚ[we], ㅝ[uʌ], ㅞ[we], ㅟ[wi]"가 올 경우 표현하지 않는다.

ㄷ. 모음 ㅏ[a], ㅑ[ja], ㅓ[ʌ], ㅕ[jʌ], ㅐ[ae], ㅒ[jae], ㅘ[wa], ㅙ[wae], ㅚ[we], ㅝ[wʌ], ㅞ[we] 다음에 자음 ㄴ[n], ㄷ[d], ㅌ[t], ㅅ[s], ㅆ[s*], ㅈ[ʧ], ㅊ[ʧ*]이 올 경우(음절의 초성, 종성에 관계없이) 표현한다.

ㄹ. 이중모음은 음소발음 시간 내에 2번 morph 한다.

- ㅑ[ja] → ㅣ[i] + ㅏ[a]
- ㅕ[jʌ] → ㅣ[i] + ㅓ[ʌ]
- ㅒ[jæ] → ㅣ[i] + ㅐ[æ]
- ㅖ[je] → ㅣ[i] + ㅔ[e]
- ㅘ[wa] → ㅗ[o] + ㅏ[a]
- ㅙ[wæ] → ㅗ[o] + ㅐ[æ]
- ㅚ[we] → ㅗ[o] + ㅔ[e]
- ㅝ[wʌ] → ㅜ[u] + ㅓ[ʌ]
- ㅞ[we] → ㅜ[u] + ㅔ[e]
- ㅟ[wi] → ㅜ[u] + ㅣ[i]

ㅁ. 현재 음소가 자음이며 규칙ㄱ과 규칙ㄴ에 의해 표현하지 않고 다음 음소가 모음일 경우에 다음 모음에 대한 FAP를 현재 자음의 Audio-StreamOffset에서 표현하도록 한다.

4. 모바일 토킹헤드 시스템

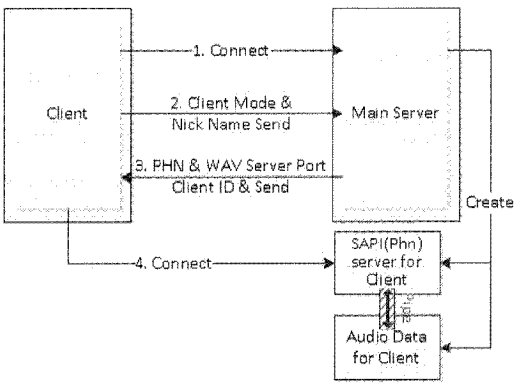
모바일 토킹헤드 시스템은 앞서 설명된 TTS 기능을 이용한 토킹헤드 시스템을 모바일 기기에서 구동한 시스템이다. 예를 들자면 모바일 기기에서 앞서 살펴본 토킹헤드 기능을 수행하거나, 스마트폰 등에 적용되어 문자 등을 상대방 모바일 기기에 전송하면 기기에서는 토킹헤드가 동작하여 음성과 얼굴애니메이션이 텍스트를 대

신해주는 등의 시스템이다.

그 동안 데스크탑 환경 기기 내에 TTS엔진을 설치하고 사용하여 토킹헤드를 구동했다. 하지만 모바일 기기는 데스크탑 기기에 비하여 하드웨어 성능이나 저장 공간 등이 적고 TTS엔진 또한 적게는 수십 메가에서부터 많게는 수백 메가까지 차지하기 때문에 모바일 기기 자체에서 바로 수행하기에는 무리가 있다. 이러한 문제점을 해결하기 위해서 서버-클라이언트 기반의 모바일 토킹헤드를 제안한다. 토킹헤드 시스템을 이용할 클라이언트는 서버에 텍스트를 전송한다. 서버는 클라이언트로부터 받은 텍스트를 오디오파일과 FAP 등의 TTS데이터를 클라이언트에게 전달해준다. 다시 클라이언트는 서버로부터 받은 데이터를 플레이하면서 모바일 토킹헤드 시스템이 완성된다.

4.1 서버-클라이언트

(그림 5)는 모바일 토킹헤드 시스템의 서버 구동에 관한 아키텍처이다.



(그림 5) 모바일 토킹헤드 시스템의 서버 아키텍처

먼저 서버에 클라이언트가 접속을 하면, 서버는 해당 클라이언트용 서브서버를 별도로 생성해준다. 이 서브서버는 클라이언트 1개당 하나씩 할당 된다. 메인서버는 단지 클라이언트의 서비스를 위한 접속과 종료만 담당하게 되는 것이다. 클라이언트마다 별도의 서브서버가 할당되는 이유는 메인서버가 받고 전달할 메시지와의 패킷의 블록크기가 완전히 상이하기 때문에 채널을 달리해줘야 하기 때문이다.

서브서버가 만들어지면 클라이언트에게 서브서버의 접속 포트번호를 가르쳐주고 클라이언트는 서브서버에 접속을 하면서 모바일 토킹헤드 시스템을 사용할 준비가 완료된다.

(그림 5)를 살펴보게 되면 메인서버가 생성하는 것이 서브서버 하나만이 아니라, 음소와 음원을 생성하는 서브서버를 각각 만들게 된다. 이는 서버-클라이언트 방식을 사용하면서 모바일 토킹헤드 시스템에 음소와 viseme 정보를 전송하기 위한 시간이 추가로 생겨서 전처리 시간이 증가하는 문제가 생기기 때문이다. 오디오 파일을 만들고 음소정보를 구하던 방식에서 음소정보를 구하는 것과 오디오 파일 생성을 동시에 시작하기 위한 것이다. 한 개의 엔진을 2개 이상 사용하기 위해서는 멀티쓰레드나 멀티프로세스 방식으로만 구현이 가능한데, TTS엔진 중에서 멀티쓰레드를 지원하지 않는 엔진도 있기 때문에 아키텍처에서는 멀티프로세스 방식으로 나타냈고, 프로세스간 통신은 네임드 파이프를 이용하여 통신을 한다.

4.2 전송

서브서버에서 클라이언트에 TTS데이터를 전송하는 부분은 다음과 같다. 서브서버는 음소정보와 오디오파일 2가지 종류의 데이터를 전송해야하는데, 일정크기의 패킷 블록에 음소정보와 오디오 파일을 넣도록 하는 Type-Length-Value(TLV)방식을 사용했다.

<표 2> 서브서버가 전송하는 패킷블록 포맷

	New	Phn Len	Phn Data	Wav Len	Wav Data
Byte	1	1	...	4	...

〈표 2〉는 서버-클라이언트간 TTS데이터 전송 패킷 포맷에 대해 보여주고 있다. 클라이언트는 전송받은 데이터를 플레이하는 피동적으로 토킹헤드가 동작되기 때문에 New의 1Byte가 필요하다. New가 1이면 서버가 새로운 텍스트에 대해 작업해서 전달해주는 것을 의미해서 클라이언트 토킹헤드의 초기화를 수행하도록 한다.

뒤에는 음소정보, 오디오데이터의 길이와 데이터필드가 각각 이어져 음소정보 청크와 오디오데이터 청크로 나뉘어져 전송되게 된다. 전송하는 시점은 SAPI가 음소정보를 전달해줄 때마다 데이터 패킷을 전달하도록 한다.

이러한 방식으로 TTS데이터를 전송하는 것은 클라이언트의 요청에 의해 서버가 서비스를 제공하는 중에 클라이언트의 새로운 요청이 들어오게 되면 즉각적으로 대처할 수 있는 장점이 있다. 오디오 데이터를 별도로 전송하게 되면 새로운 요청 작업이 들어올 경우 전송하던 작업을 초기화할 필요가 있지만, 음소정보 획득 시 데이터 패킷을 전달하게 되면 음소정보 구하는 모듈을 정지만 시키면 전송작업은 정지하게 되기 때문이다.

5. SAPI 한글 음소 맵핑

현재 MS사의 SAPI에는 한글의 음소에 대한 인덱스가 정의되어 있지 않기 때문에 SAPI의 TTS기능을 이용할 때 발생되는 음소의 인덱스와 한글에 대해 맵핑을 시킬 필요가 있다.

이를 위해, 커뮤니티 사이트 Nate와 Naver 뉴스의 정치와 스포츠 분야의 각각 3페이지씩 약 1만자의 텍스트를 가지고 TTS에 입력해서 진행하였다. 입력할 문장에 대해 초성, 중성, 종성으로 분리하고 인덱스와 음소에 대해 1차적으로 맵핑한 후, 표준어 발음법과 비교하여 최종적으로 맵핑하는 순서로 진행하였다.

〈표 3〉은 한글텍스트에 대한 SAPI의 음소 인덱스와 한글 음소간 맵핑 결과이다.

〈표 3〉 SAPI 한글 음소 인덱스

TTS Phn Idx	한글 음소	TTS Phn Idx	한글 음소	TTS Phn Idx	한글 음소	TTS Phn Idx	한글 음소
2	ㄱ	16	ㅊ	37	ㅑ	54	ㅖ
3	ㄲ	17	ㅋ	38	ㅒ	55	ㅟ
4	ㄴ	18	ㅌ	39	ㅕ	58	ㅠ
5	ㄷ	19	ㅍ	42	ㅖ	59	ㅡ
6	ㄸ	20	ㅎ	43	ㅕ	60	ㅢ
7	ㄹ	21	M ㄱ	44	ㅖ	61	ㅣ
8	ㅁ	22	M ㄷ	45	ㅗ	69	E ㄴ
9	ㅂ	23	M ㄹ	46	ㅘ	73	E ㄹ
10	ㅃ	24	M ㅂ	47	ㅙ	81	E ㅁ
11	ㅅ	25	M ㅈ	50	ㅚ	87	E ㅇ
12	ㅆ	26	M ㅎ	51	ㅛ	96	Silence
14	ㅈ	35	ㅏ	52	ㅜ		
15	ㅉ	36	ㅐ	53	ㅝ		

한글 음소 앞에 M이라고 붙어 있는 것은 Middle을 뜻하는 것으로 단어의 중간에 있을 때 나오는 것이다. 예를 들어 "김김"이라고 입력을 하게 되면 처음 '김'의 'ㄱ'은 2번의 음소 인덱스가 나오게 되고, 뒤의 '김'의 'ㄱ'은 21번의 인덱스가 나오게 된다.

6. 실험 결과

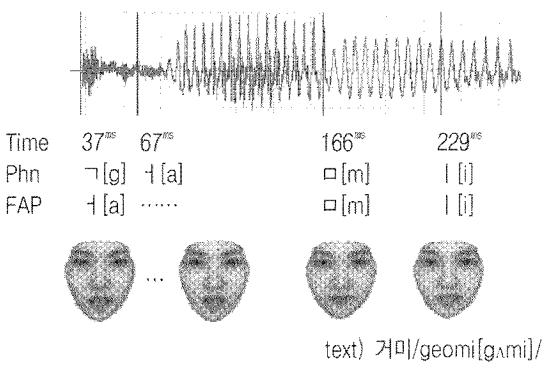
6.1 오디오-비디오 동기 실험

제안된 시스템의 구현을 위해 OS로 Windwos7, TTS엔진으로 NeoSpeech사의 Yumi엔진, SAPI5.1 버전을 사용해서 이루어졌다.

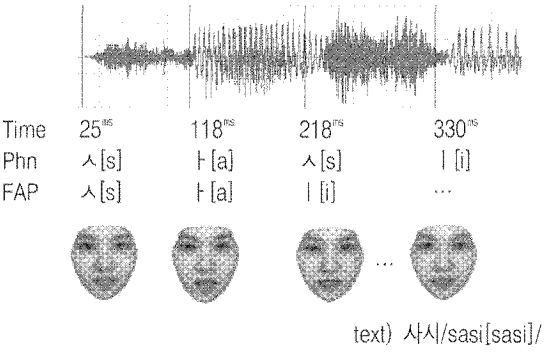
다음 그림들은 한글 음소간 합성 규칙에 따라 진행된 실험 결과이다.

(그림 6)은 합성규칙'ㄱ'에 따라서 표현하지 않는 자음에 대해 FAP를 처리하는 결과를 보여주고 있다. 거미에 대해 'ㄱ'의 입모양을 처리하지 않고 'ㄱ'의 위치에서 'ㅓ'의 FAP를 표현하도록 했다.

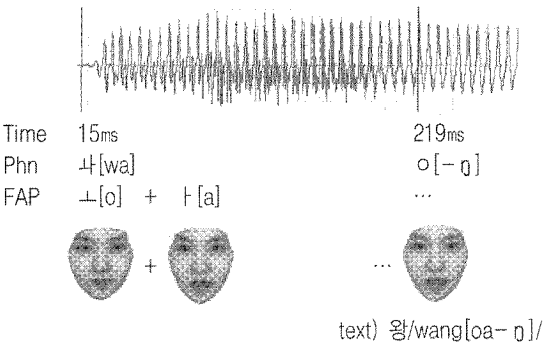
(그림 7)은 합성규칙 'ㄴ'과 'ㄷ'에 대한 결과이



(그림 6) 합성 규칙 'ㄱ' 에 대한 결과



(그림 7) 합성 규칙 'ㄴ' 과 'ㄷ' 에 대한 결과



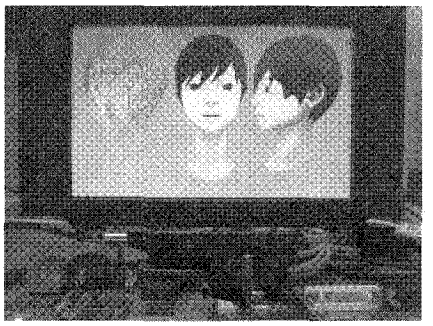
(그림 8) 합성 규칙 'ㄹ' 에 대한 결과

다. "사시" 단어에 대해 앞의 'ㅅ'은 FAP를 처리해주었고 뒤에 'ㅅ'은 무시를 하고 바로 모음 'ㅣ'의 FAP를 처리하도록 하였다.

(그림 8)는 이중모음에 대한 FAP를 처리하는 결과를 보여주고 있는 그림이다. 음소 '와'의 발음시간 내에 모음'ㅗ'와 모음'ㅏ'의 FAP를 모두 처리하게 하였다. FAP의 값의 증가는 평상시보다 2배씩 증가하도록 하였으며, 해당 음소의 전체 발음시간에 대해 1/3만큼은 선행되는 모음에 나머지 2/3시간 동안에는 뒤 따라오는 모음에 대한 FAP를 표현하도록 하였다.

6.2 모바일 토킹헤드 구현

제안된 모바일 토킹헤드 시스템은 리버트론에서 제작한 Syslab II 보드와 PC를 이용해서 구현되었다. 타겟보드는 삼성 스마트폰 옴니아2의 CPU와 동일한 S3C6410 CPU를 사용하고 있으며 운영체제는 Windows Embedded CE 6.0을 사용하고 있다.



(그림 9) 타겟보드에서 토킹헤드 구동 장면

(그림 9)는 타겟보드에서 서버에 텍스트를 전송하고 받게된 TTS데이터로 토킹헤드를 구동하는 화면이다. 음소정보를 받을 때마다 TTS데이터를 전송하는 방식으로 구현되어 있는 시스템에서 패킷사이즈가 너무 작으면 FAP에 비해 오디오데이터가 제대로 전송이 못되어서 토킹헤드 도중에 버퍼링 시간이 생기게 된다. 오디오 품질 16Khz, 16bit 싱글채널을 사용해서 1Kbyte씩 패킷사이즈를 늘려서 실험해 본 결과 패킷의 크기가 4Kbyte이상이면 버퍼링이 없음을 확인하였다. 이는 보편적인 TCP/IP의 소켓버퍼의 크기가 8Kbyte인 것을 감안하면, 32Khz, 16bit 싱글채널까지 본 논문에서 제안하는 전송방식으로도 충분히 서비스가 가능함을 의미한다.

6.3 기타 응용 서비스

PC 기반의 토킹헤드 응용 서비스로 토킹헤드 프리젠테이션 서비스와 인터넷 뉴스 서비스를 구현하였다. 토킹헤드 프리젠테이션 프로그램의 구현을 위해 MS사에서 제공하는 오피스 2007 개발자 API를 이용하였다. 프리젠테이션 파일 중에 Microsoft (MS)의 PowerPoint 2007이상에서 기본 저장 포맷인 .pptx파일을 이용해서 진행하였다. pptx파일은 여러 개의 xml파일이 압축된 파일이다. 이 파일을 이용하면 쉽게 PPT파일에 접근하여 원하는 정보를 손쉽게 추출할 수 있는 장점이 있다. MS의 pptx파일을 압축해제하게 되면 폴더 구조로 해제가 되며, 이 중에서 ppt폴더 안의 noteslide*.xml파일에서 프레젠테이션 진행에 필요한 Note정보를 추출한다. 추출된 노트들은 슬라이드에 맞추어 노트의 텍스트를 TTS 엔진에 넘겨주고 토킹헤드 렌더링 모듈은 해당 슬라이드의 얼굴 애니메이션을 음성과 함께 제공하도록 구성하였다. 이외에도 인터넷의 뉴스를 자동으로 제공해 주는 토킹헤드 서비스도 구현하였다. 실험은 네이버의 스포츠 면을 가지고 진행되었다. 제안된 서비스는 해당 사이트에 뉴스 페이지의 소스를 요청하면서 시작된다. HTTP프로토콜을 통해서 전달받은 뉴스 페이지 소스를 받게 되면 Tag부분을 삭제하고 뉴스 부분의 텍스트를 추출하였다. 이렇게 추출된 텍스트를 TTS에 넘겨주고 그 결과를 토킹헤드 렌더링 모듈에 전달하여 인터넷 뉴스를 토킹헤드가 읽어주게 되는 것이다.

7. 결 론

본 논문에서는 토킹헤드 서비스 구현에 요구되는 기술적 요구사항과 연구동향에 대해서 소개하였고 모바일 환경에서 구현이 가능한 TTS 기반의 토킹헤드 서비스 아키텍처를 제안 하였다. 구체적으로는 TTS 기반의 토킹헤드 시스템의 립싱크를 위한 아키텍처와 자연스러운 토킹헤드 구현을 위한 음소간 합성규칙을 제안하였으며, 모바일 기기에서 토킹헤드 서비스의 구현을 위해 서버-클라이언트 기반 아키텍처를 제안하였다. 이 시스템은 클라이언트에 별도의 TTS 엔진의 설치를 요구하지 않는 장점이 있으나 서버에 탑재된 TTS엔진을 사용함으로써 추가적으로 시간지연이 발생하는 단점을 갖는다. 제안된 토킹헤드 시스템은 storytelling, e-book, 비디오 메신저, 게임, 차세대 휴먼-컴퓨터 인터페이스 등의 다양한 서비스에서 활용이 가능할 것으로 판단된다.

감사의 글

이 논문은 교육과학기술부의 재원으로 한국연구재단의 지원을 받아 수행된 광역경제권 선도산업 인재양성사업과 2009년도 대학중점연구소 지원사업(2009-0093828)으로 수행된 연구임

참고문헌

[1] Fabio Lavagetto, "Converting Speech into Lip Movement : A Multimedia Telephone for Hard of Hearing People," IEEE Transaction on Rehabilitation Engineering, vol. 3, no. 1,1995, pp. 90-102.

[2] T. Noma, L. Zhao, N.I. Badler, "Design of a virtual human presenter,"IEEE Computer Graphics and Applications, vol. 20 no. 4, 2000, pp. 79-85.

[3] Yura, S., Usaka, T., Sakamura, K., " Video avatar: embedded video for collaborative virtual environment," IEEE International Conference on Multimedia Computing and Systems, Vol. 2, 1999, pp. 433-438.

[4] Yao-Jen Chang, Chih-Chung Chen,

Jen-Chung Chou, Yung-Chang Chen, "Implementation of a virtual chat room for multimedia communications," 1999 IEEE 3rd Workshop on Multimedia Signal Processing, 1999, pp.599-604.

[5] Tian-Swee Tan, et al., "Photo-realistic text-driven malay talking head with multiple expression," Proc. Computer and Comm. Engineering, pp. 711-715, 2008.

[6] E. Cosatto, J. Ostermann, H.P. Granf, "Lifelike talking faces for interactive services", Proc. IEEE91(9), 1406-1428, 2003.

[7] C.J. Kuo, R.-S. Huang, and T.-G. Lin, "3-D Facial Model Estimation From Single Front-View Facial Image," IEEE Transactions on CSVT, vol. 12, no. 3, 2002, pp. 183-192

[8] Won-Sook Lee, Marc Escher, Gael Sannier, Nadia Magnenat-Thalmann, "MPEG-4 Compatible Faces from Orthogonal Photos," International Con- ference on Computer Animation, 1999, pp.186-194.

[9] Zicheng Liu, Zhengyou Zhang, Chuck Jacobs, Michael Cohen, "Rapid Modeling of Animated Faces From Video," Technical Report MSR-TR-2000

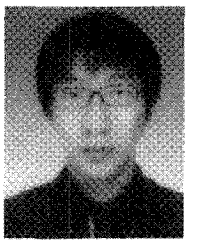
[10] Morishima, S. and Harashima, H., "A Media Conversion from Speech to Facial Image for Intelligent Man-Machine Interface", IEEE Journal on sel. areas in communications, vol. 9, No. 4, 1991, pp. 594-600.

[11] Fabio Lavagetto, "Time-Delay Neural Networks for Estimating Lip Movements From Speech Analysis: A Useful Tool in Audio-Video Synchronization," IEEE Transaction on Circuits and Systems For Video Technology, vol. 7, no. 5,pp. 786-800.

[12] K.H. Choi, Ying Luo, Jenq-Neng Hwang, "Hidden Markov Model Inversion For Audio-to-Visual Conversion in an MPEG-4 Facial Animation System, " Journal of VLSI Signal Processing - Systems for Signal, Image, and Video Technology, vol. 29, 2001, pp. 51-61.

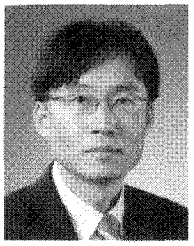
[13] K.H. Choi and Jenq-Neng Hwang, "Constrained Optimization for Audio-to-Visual Conversion," IEEE Transaction on Signal Processing, vol. 52, no. 6, June, 2004.

저자약력



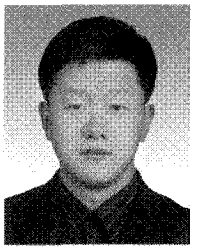
김 상 완

2009년 목포대학교 전자공학과(학사)
2010년 목포대학교 전자공학과 석사과정
관심분야 : 영상, 비디오 신호처리, 모바일 토킹헤드 시스템
이 메 일 : systemii@mokpo.ac.kr



박 순 영

1982년 연세대학교 전자공학과(학사)
1984년 연세대학교 전자공학과(석사)
1989년 State University of New York 전기및컴퓨터공학과 (박사)
1990년~현재 목포대학교 정보전자공학과 교수
관심분야 : 영상, 비디오, 컴퓨터그래픽스, 멀티미디어 응용 등
이 메 일 : sypark@mokpo.ac.kr



최 경 호

1989년 인하대학교 전기공학과(학사)
1991년 인하대학교 전자공학과(석사)
2002년 Univ. of Washington, Seattle, Dept. of Electrical Eng.(박사)
1991년~2005년 ETRI 텔레매틱스연구단/ 팀장
2005년~현재 목포대학교 전자공학과 조교수
관심분야 : 멀티미디어 시스템, Audio-to-Visual 변환, GIS/ITS/텔레매틱스
이 메 일 : khchoi@mokpo.ac.kr