차 례

로봇 콘텐츠 소프트웨어 개발 학습모듈의 개요

학습모듈의 목표

로봇의 동작과 디지털 멀티미디어 콘텐츠가 연동되도록 로봇 콘텐츠를 설계하고 개발할 수 있다.

선수학습

C 프로그래밍, 로봇공학, ROS(robot operating system)

학습모듈의 내용 체계

학습	학습 내용	NCS 능력단위 요소	
		코드번호	요소 명칭
1. 로봇 콘텐츠 설계서 작성하기	1-1. 로봇 콘텐츠 설계서 작성	1903080309_14v1.1	로봇 콘텐츠 설계서 작성하기
2. 로봇 콘텐츠 개발하기	2-1. 로봇 콘텐츠 개발	1903080309_14v1.2	로봇 콘텐츠 개발하기
3. 로봇 콘텐츠 실행 소프트웨어 개발하기	3-1. 로봇 콘텐츠 실행 소프트웨어 개발	1903080309_14v1.3	로봇 콘텐츠 실행 소프트웨어 개발하기
4. 로봇 콘텐츠 저작 도구 개발하기	4-1. 로봇 콘텐츠 저작 도구 개발	1903080309_14v1.4	로봇 콘텐츠 저작 도구 개발하기

핵심 용어

로봇 콘텐츠, 로봇 드라마, 시나리오, 콘텐츠 정의서, 콘텐츠 설계서, 콘텐츠 저작 도구, 메타모델, 콘텐츠 실행 소프트웨어

학습 1	**로봇 콘텐츠 설계서 작성하기**
학습 2	로봇 콘텐츠 개발하기
학습 3	로봇 콘텐츠 실행 소프트웨어 개발하기
학습 4	로봇 콘텐츠 저작 도구 개발하기

1-1. 로봇 콘텐츠 설계서 작성

학습 목표

- 로봇 콘텐츠 구성 요소에 따라 로봇 콘텐츠 정의서를 작성할 수 있다.
- 로봇 사용자에게 제공되는 서비스 형태에 따라서 정의된 로봇 콘텐츠의 시나리오 기반 작업 단위를 정의할 수 있다.
- 로봇 콘텐츠 시나리오에 따라 로봇 콘텐츠의 저작 및 실행 방법에 대한 작업 절차서를 작성할 수 있다.
- 로봇 콘텐츠의 실행 시나리오를 기획하고, 로봇 콘텐츠 설계서를 작성할 수 있다.

필요 지식 /

[1] 로봇 콘텐츠 설계서 작성

1. 로봇 콘텐츠

(1) 정의

로봇 콘텐츠는 동영상, 음원, 이미지를 포함하는 멀티미디어 데이터와 로봇의 모션, 멀티미디어 재생, 디스플레이 등의 다양한 기능이 결합되어 사람에게 특정 서비스를 제공하거나 스토리 또는 메시지를 전달하는 것을 의미한다.

한정혜(2011)는 로봇 콘텐츠를 '로봇에 탑재되어 있으면서 로봇이 외부와 상호 작용하는데 필요한 모든 표현(말, 표정, 행위, 멀티미디어 자료, 지식 등)의 체계' 라고 정의했다.

(2) 구성 요소

로봇 콘텐츠의 구성 요소는 일반적으로 멀티미디어 데이터(파일)와 로봇 동작, 그리고 메타모델(meta-model)을 포함한다. 여기서 메타모델이란 멀티미디어 데이터와 로봇 동작을 시나리오 기반으로 실행하기 위해서 정의한 절차서를 통칭한 것이다.

한편, 현은자・윤현민・장시경 외(2009)는 로봇 콘텐츠가 다음과 같이 구성되어 있다고 분석했다.

- 로봇의 행동: 제스처, 이동, 인간과 로봇의 상호 작용
- 명령/작업 처리: 태스크, 시나리오, 지능, 상황 인지
- 멀티미디어 콘텐츠: 플래시, 영화, 음악

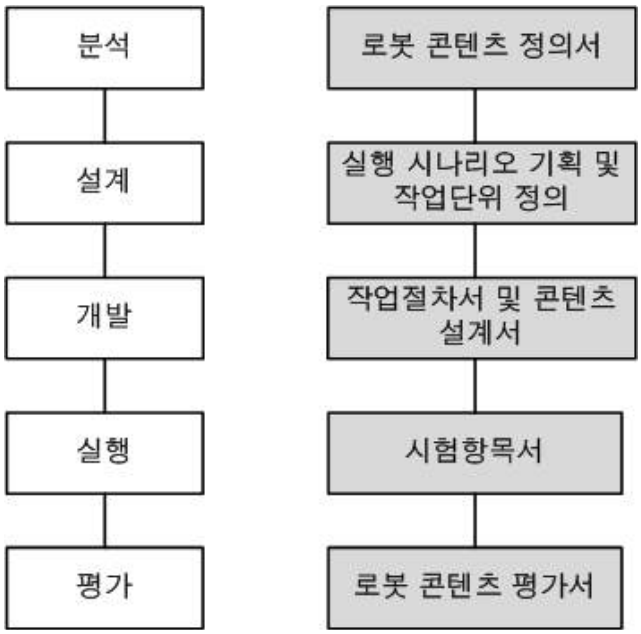
(3) 종류

오상록(2008)에 따르면 로봇 콘텐츠의 종류에는 정보 서비스, 홈케어 서비스, 맞춤형 서비스, 기본 서비스, 교육 서비스, 오락 서비스, 가사 서비스, 감성 서비스, 로봇 기능 서비스, 기관 정보 서비스 등으로 나눌 수 있다.

국내외에서 현재 개발 중이거나 시중에 선보이고 있는 로봇 콘텐츠 서비스로는 교사 보조 서비스, 안내/홍보 서비스, 보안 로봇 서비스, 외식 도우미 서비스 등이 있다.

2. 개발 과정

로봇 콘텐츠는 멀티미디어 콘텐츠의 일종이므로 멀티미디어 개발의 일반적 단계를 참고할 필요가 있는데, 멀티미디어 개발 단계는 분석(analysis), 설계(design), 개발(development), 실행(implementation), 평가(evaluation)의 단계로 구성 된 ADDIE 모델을 따른다(한정혜, 2011). ADDIE 모델을 기반으로 로봇 콘텐츠 제작을 위해 필요한 문서들을 산출해 보면 다음과 같다.



[그림 1-1] 로봇 콘텐츠 제작 단계별 필요 문서

(1) 로봇 콘텐츠 정의서(분석 단계)

개발할 로봇 콘텐츠가 적용되는 서비스 분야가 무엇이며, 콘텐츠의 사용자 또는 대상자의 성별, 연령대, 선호도, 사용 특성과 콘텐츠를 통해 얻고자 하는 목표, 주제, 요구사항, 기대하는 효과 등을 정의하고 분석한다.

또 콘텐츠에서 로봇이 의인화 될 경우 사용자 입장에서 바라 본 로봇의 신상특성(성별, 나이, 국적, 성격, 가치관, 직업, 사용자와의 관계 등)에 대해서도 필요시 정의한다.

로봇과 사용자 간의 효과적인 의사소통을 위해 필요한 방식을 언어(verbal), 준언어(paraverbal), 비언어(nonverbal) 방식 중 선택한다. 이 중에서 준언어는 전달되는 목소리의 음색, 고저, 리듬, 속도 등을 의미하고, 비언어는 응시, 표정, 몸짓, 신체접촉 등을 포함한다.

이러한 정의와 분석 결과에 적합한 하드웨어(로봇의 신체 구조, 모터, 센서, 디스플레이 모듈 등)와 소프트웨어(OS, 미들웨어, 애플리케이션 프로그램, 사용자 인터페이스 등)를 가진 로봇 플랫폼을 선정한다.

이 모든 것들이 정해지면 로봇 콘텐츠 개발에 소용되는 총 비용과 인원, 기간 등을 산정한다.

(2) 시나리오 기획(설계 단계)

시나리오 기획의 가장 첫 단계는 콘텐츠의 시놉시스(synopsis)를 작성하는 것인데, 시놉시스란 콘텐츠의 시나리오를 쓰기 전에 전체적인 개요와 내용을 간략하게 작성한 것이다. 시놉시스에는 콘텐츠의 제목, 주제, 작가의 의도, 장르, 주인공과 등장인물, 전체 줄거리 등이 포함되어야 한다.

시놉시스가 완성되면 이를 기초로 시나리오를 기획하고 작성해야 한다. 로봇 콘텐츠의 경우 사용자에게 메시지를 전달해야 하는데 메시지를 전달하는 플롯(plot, 사건의 논리적인 패턴과 배치)인 메시지 시나리오를 설계해야 한다(한정혜, 2011). 교육용 로봇 콘텐츠를 예로 들면 메시지 시나리오는 교수학습 활동과 관련한 교안(또는 교수법)이라고 할 수 있다.

스토리 보드는 메시지 시나리오를 바탕으로 로봇의 입출력 방식, 상호 교환하는 언어・준언어・비언어적 메시지, 로봇이 처한 환경, 상호작용하는 상대방을 모두 포함하여 로봇이 사용자와 상호작용하는 시나리오를 세부적으로 작성하는 것이다.

(3) 작업 절차서(개발 단계)

스토리 보드가 작성 되었으면 로봇이 구현해야 할 태스크 별로 작업단위를 정의하고, 각 작업단위 별로 개발해야 할 로봇공학 기술 및 알고리즘에 대한 작업 절차서를 작성해야 한다.

작업 절차서는 선택한 로봇 플랫폼의 기구적 구조와 구성요소, 기능, 하드웨어 사양, 사용 가능한 소프트웨어에 따라서 작성해야 한다.

일반적으로 설계의 경우는 하향식(Top-Down)식으로, 개발의 경우는 상향식(Bottom-Up)으로 진행되어야 하며, 작업절차 중 로봇이 사용자와 상호작용하는 GUI(Graphic User Interface) 또는 TUI(Tangible User Interface, 촉감형 사용자 인터페이스) 개발과 관련한 부분을 제일 먼저 개발한 후 나머지 작업 절차를 그에 맞춰 개발해야 한다.

(4) 콘텐츠 시험・평가서(실행 및 평가 단계)

콘텐츠 개발이 완료되었으면 콘텐츠 개발 프로듀서(책임자)는 콘텐츠의 프로토타입 버전을 로봇에서 실행하며 로봇 콘텐츠 설계서를 모두 만족하는지, 음성・영상 멀티미디어와의 동기화가 잘 이뤄졌는지, 로봇 모션에서 부자연스럽거나 잘못된 동작이 발생하는지 등의 기술적 부분을 모두 체크해서 이상 여부를 확인한다. 만약 기대 이하의 성능이나 오류가 발생하면 해당 작업의 작업 절차서로 돌아가서 작업을 다시 수행해야 한다.

콘텐츠의 객관적인 평가를 위해 콘텐츠 설계서에 대응되는 콘텐츠 평가서를 만들고 목표 사용자를 대상으로 시범적으로 콘텐츠를 실행한다. 이때 콘텐츠에 다양한 기능이 있는 경우 로봇이 처음부터 끝까지 서비스를 하게 해서 소집단의 전문가들이 사용자 경험을 평가하고, 이 과정을 통해 사용자가 요구하는 다양한 기능을 모두 만족시키도록 수정・보완하는 방법을 사용한다.

최종 완성 된 콘텐츠는 로봇에 설치하여 사용자/대상자에게 실행하며, 사용자의 정량적・정성적 평가 및 만족도를 피드백 받아서 로봇 콘텐츠의 완성도를 지속적으로 향상시켜야 한다.

수행 내용 / 로봇 콘텐츠 설계서 작성하기

재료 · 자료

- 로봇 콘텐츠 요구 사항 정의서
- 로봇 하드웨어 사양서
- 로봇 소프트웨어 플랫폼의 설명서
- 로봇 기능 정의서

기기(장비 · 공구)

- 컴퓨터, 프린터, 인터넷
- 문서 작성용 소프트웨어

안전 · 유의 사항

- 로봇 콘텐츠 설계 시 사용자의 안전과 사용의 편의성을 중요하게 고려해야 한다.
- 로봇 콘텐츠 사용자의 요구와 시나리오를 적극적으로 수용하는 태도를 가져야 한다.

수행 순서

1 로봇 콘텐츠 정의서를 작성하고 적절한 로봇 플랫폼을 선정한다.

1. <표 1-1>에 나타낸 로봇 콘텐츠 정의서 양식에서 로봇 콘텐츠의 서비스 분야, 사용자 특성, 콘텐츠 특성, 로봇 특성, 메시지 전달 방식에 대해 작성한다.
2. 로봇 콘텐츠 개발 예제로서 로봇 드라마(로보라마)를 선정한 경우, <표 1-2>와 같이 작성할 수 있으며, 이때 <표 1-1>에서 해당사항이 없는 줄은 삭제한다.
3. 작성한 로봇 콘텐츠 정의서에 따라 적합한 로봇 플랫폼을 결정한다. 로보라마의 경우 인간과 유사한 신체구조를 가진 휴머노이드 로봇이 가장 적합하다.
4. 로봇 콘텐츠 개발에 소용되는 총 비용과 인원, 기간 등을 산정한다.

<표 1-1> 로봇 콘텐츠 정의서 양식

구분		내용
콘텐츠 서비스 분야		정보 서비스, 홈케어 서비스, 맞춤형 서비스, 교육 서비스, 오락 서비스, 가사 서비스, 감성 서비스, 로봇 기능 서비스, 기관 정보 서비스 중 선택
사용자 특성	신상	성별: 연령: 직업:
	선호도	
	사용특성	
콘텐츠 특성	목표	
	주제	
	요구 사항	
	기대효과	
로봇 특성	신상	성별: 연령: 직업:
	성격	
	가치관	
	사용자와의 관계	
	기타	
메시지 전달방식	언어	
	준언어	
	비언어	

<표 1-2> 로봇 드라마의 로봇 콘텐츠 정의서 작성 예

구분		내용
콘텐츠 서비스 분야		오락 서비스
사용자 특성	신상	연령: 10세 이상
	선호도	로봇이 인간의 동작을 자연스럽게 구현하는 것을 좋아함
	사용특성	드라마를 편안하게 관람
콘텐츠 특성	목표	로봇이 인간 배우처럼 감동적인 드라마를 구현할 수 있음을 보임
	주제	로봇 배우들을 통해 진정한 가족애가 무엇인지를 깨닫게 함
	요구 사항	최대한 현실감을 느낄 수 있도록 로봇의 동작이 안정적이고 자연스러워야 하며 의상과 소품, 배경이 사실적이어야 함
로봇 특성	신상	성별: 남, 연령: 40대, 직업: 평범한 직장인
	성격	소심하고 수동적이지만 가족을 지키고자 함
메시지 전달방식	언어	일상 생활 언어 사용
	준언어	성격과 감정, 상황에 따라 목소리의 고저, 리듬, 속도를 조절
	비언어	희노애락의 감정을 적절한 자세와 동작으로 표현

2 로봇 콘텐츠 정의서에 따라 시놉시스와 시나리오를 작성한다.

1. 로봇 콘텐츠의 정의서에 따라 로봇 콘텐츠의 시놉시스를 작성한다. 구현할 로봇 드라마의 주제가 '가족애' 이므로 다음과 같은 시놉시스를 일례로서 작성할 수 있다 (https://www.youtube.com/watch?v=YtksgM53k0A&feature=youtu.be).

제목: "아버지의 이름으로"

주제: 평범한 직장인의 애환을 가족애를 통해 극복함으로써 가정의 소중함을 일깨움

주인공과 등장인물: 아버지 로봇(주인공), 직장상사 로봇, 어머니 로봇, 아들 로봇

전체 줄거리는 다음과 같다.

<표 1-3> 로봇 콘텐츠 시놉시스 예

장면 번호	장면별 줄거리
Scene 1	회사 내의 업무에서 큰 실수를 한 주인공이 직장 상사에게 호된 질책을 듣는다. 흥분한 직장 상사는 주인공에게 화를 내며 사표를 쓰라고까지 한다.
Scene 2	풀이 죽은 채 집에 귀가한 주인공은 아내에게 위로를 받고 싶었으나 생활비 문제 때문에 오히려 핀잔을 듣는다.
Scene 3	아들은 주인공의 충고를 잔소리로 듣고 반항함으로써 주인공에게 더욱 더 상처를 준다.
Scene 4	주인공이 집에 없는 사이 회사의 동료에게서 온 전화를 받은 아내와 아들은 주인공이 정신적으로 힘든 상태라는 것을 알고 자신의 행동을 후회한다.
Scene 5	다시 집으로 들어온 주인공에게 아내와 아들은 따뜻한 관심을 보이며 위로하고, 질책한 상사에게서 사과의 전화를 받는다.
Scene 6	가수의 노래에 맞춰 온가족이 다함께 춤을 추며 행복하게 마무리한다.

2. 로봇 콘텐츠의 시놉시스에 따라 각 씬 별로 시나리오를 작성한다. "아버지의 이름으로" 의 Scene 1에 대한 시나리오를 문장으로 작성한 예는 다음과 같다. 아버지 로봇은 배대리로, 직장상사 로봇은 최과장이라는 이름을 붙인다.

긴장감이 도는 음악과 함께 직장상사인 최과장이 의자에 앉은 채 화를 내며 허공에 주먹을 휘두르고 손으로 가슴을 친다. 긴장감이 도는 음악이 서서히 줄어들며, 전화벨이 울린다. 주인공인 배대리가 전화 받는 모션을 취한다.

최과장 : (몹시 흥분하여) 배대리, 당장 내방으로 오게!

배대리 : (머리를 긁적이며 풀죽은 목소리로) 알겠습니다. 지금 가겠습니다.

배대리 : (노크를 한다) 과장님, 배대리입니다. 들어가도 되겠습니까?

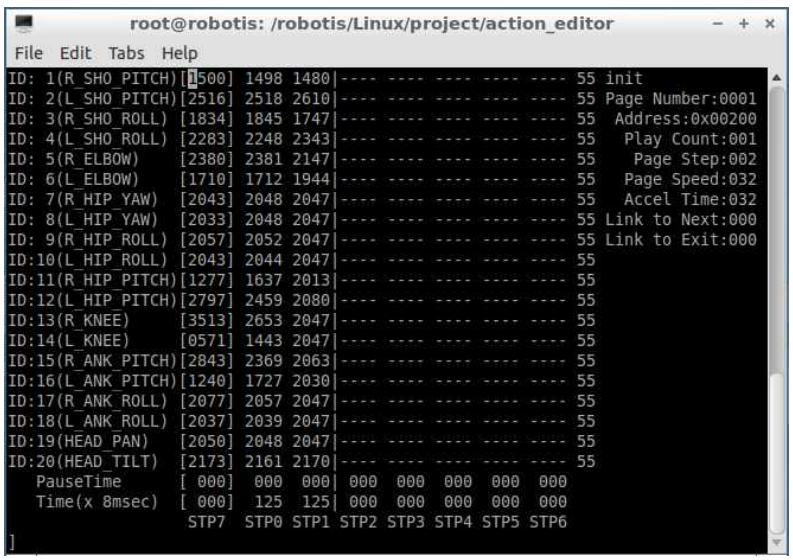
최과장 : (화난 목소리로) 들어 오게!
배대리가 문을 열고 조심스럽게 들어온다.
최과장 : (삿대질을 하며) 배대리, 지금 자네가 저지른 일이 회사에 얼마나 큰 손실인지 아는가?
배대리 : (고개를 숙이며) 죄송합니다. (고개를 들고 한쪽 팔을 살짝 올리며) 하지만 그 일은...
최과장 : (흥분해서 일어서서 삿대질을 하며) 어디서 변명을 하나? 배대리 때문에 회사가 큰 어려움에 빠졌어. 꼴도 보기 싫으니까 당장 나가게. 이번 달 월급도 없는 줄 알고, 불만 있으면 회사를 나가든가 아니면 그냥 사표를 쓰게나.
배대리 : (무릎을 꿇고 흐느끼며) 과장님, 정말 죄송합니다. 제게는 처자식이 있습니다. 한 번만 용서해 주십시오. 다음부터 절대 이런 실수 없도록 하겠습니다.
최과장 : (한쪽 팔로 문을 가리키며) 당장 나가게!
비장한 음악이 흐르며 배대리는 방을 나간다.

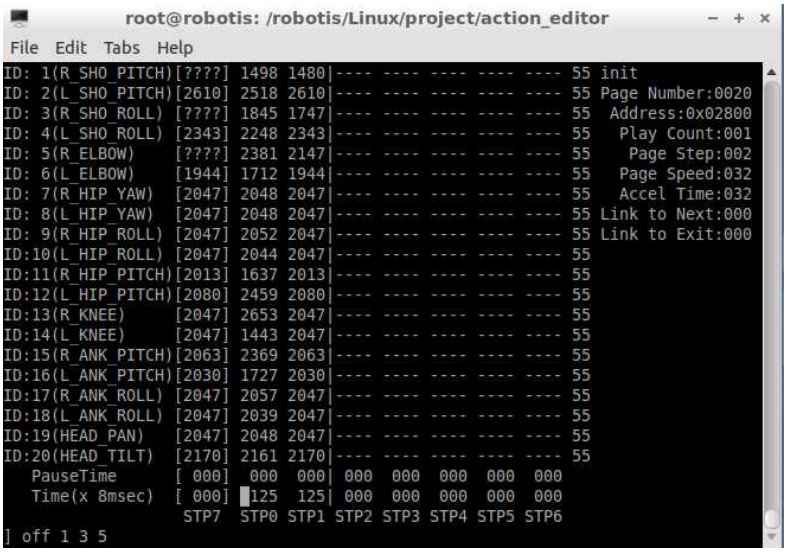
3 로봇 콘텐츠의 시나리오에 따라 작업절차서와 로봇 콘텐츠 설계서를 작성한다.

1. 시나리오에 따라 로봇이 구현해야 할 태스크 별로 작업 단위를 정의한다. '아버지의 이름으로' 라는 시나리오의 Scene 1에 대해 필요한 작업 단위들을 다음과 같이 분류한다.
   (1) 로봇 전신 모션: 팔 휘두르기, 전화받기, 걸어가기, 의자에 앉기/일어서기, 무릎꿇기/일어서기 등
   (2) 음성: 각 대사를 TTS(Text to Speech)를 이용해서 음성 파일(wave 파일 또는 mp3 파일)로 제작
   (3) 음향효과: 전화벨 소리 파일, 긴장이 도는 음악과 비장한 음악 파일 다운로드
2. 정의한 작업 단위를 기반으로 작업 절차서를 작성한다.
   작업 단위에 대한 작업 흐름도를 [그림 1-2]와 같이 구성한다.
   각 작업 단위에 대해 작업 절차서를 작성한다. <표 1-4>는 로봇의 전신 모션을 만들기 위해 휴머노이드 로봇의 모션 편집기 소프트웨어를 사용하는 것에 대한 작업 절차서의 예이다.

[그림 1-2] 로봇 콘텐츠 작업 흐름도

<표 1-4> 로봇 모션 만들기의 작업 절차서

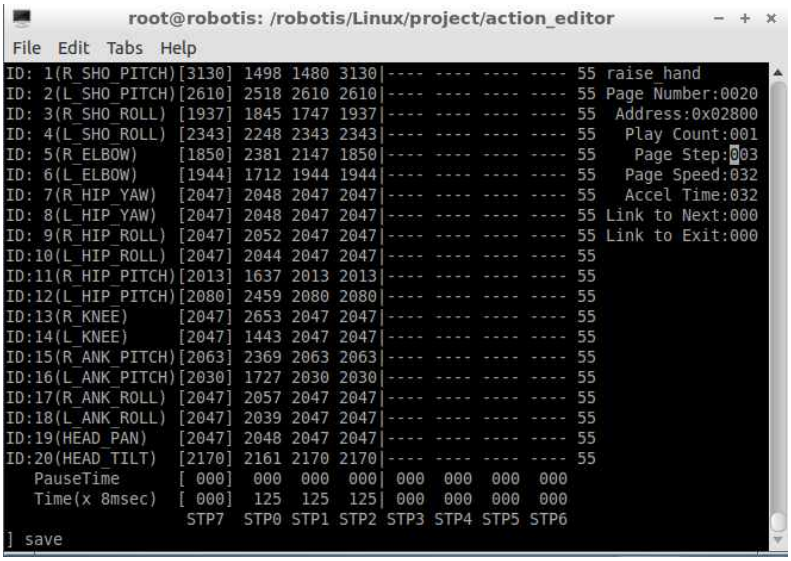
구분	내용
1. 작업 순서	1.1 구현할 모션의 특징(모션 프리미티브, 동작 시간, 속도 등)에 대해 파악한다. 1.2 휴머노이드 로봇의 모션 편집기 소프트웨어를 실행한다. 1.3 손으로 모션 프리미티브를 직접 만들고 관절각 데이터를 저장한다. 1.4 저장된 데이터를 재생해서 모션이 안정적이면서 목표한 대로 움직이면 다음 동작으로 넘어간다.
2. 작업 설명도	휴머노이드 로봇의 모션 편집기 소프트웨어(ROBOTIS-OP의 경우 액션에디터)를 사용해서 오른팔을 상하로 움직이기 위한 모션 프리미티브를 제작하고 재생을 통해 확인한다.  출처: 액션에디터. 스크린샷. [그림 1-3] 액션에디터의 init 모션 화면



출처: 액션에디터. 스크린샷.

[그림 1-4] 액션에디터에서 휴머노이드 로봇의 오른팔에 해당하는 1, 3, 5번째 관절모터의 토크를 off로 만들고 오른팔의 원하는 자세를 세팅

2. 작업 설명도



출처: 액션에디터. 스크린샷.

[그림 1-5] 새로 세팅한 자세를 save 명령으로 저장하고 play 명령으로 새로 제작한 모션 확인

3. 주요 착안 사항	새롭게 제작한 자세들을 연결하여 모션을 재생하면서 모션의 정확도와 함께 로봇의 안정성(넘어지지 않음)이 유지되는지를 확인해야 한다.

3. 로봇 콘텐츠 설계서를 작성한다.

이전 단계에서 작성한 로봇 콘텐츠 정의서와 작업 절차서를 기반으로 <표 1-5>와 같은 로봇 콘텐츠 설계서를 작성한다.

로봇 콘텐츠 설계서로부터 Scene 1을 위해 만들어야 할 모션의 개수(10개)와 음성 파일의 개수(10개), 음악/음향 효과 파일의 개수(4개) 등을 일목요연하게 볼 수 있으며, 이에 따라 콘텐츠 제작 작업을 분업화 할 수 있다.

<표 1-5> 로봇 콘텐츠 설계서

Scene 번호	1	작성날짜		작성자	
콘텐츠 개요	'아버지의 이름으로' 로봇 드라마의 Scene 1 설계서이며, 최과장이 배대리의 업무상 실수에 대해 화를 내고 문책하는 상황				
시간(분)	로봇				
	모션		음성 파일	음악/음향 효과	눈 LED
5분	(최) 의자에 앉은 채 손으로 가슴을 친 후 전화를 건다.			긴장감 있는 음악 파일 재생	(최) 화가 난 상태를 의미하는 붉은색으로 눈 LED 점등
	(배) 전화를 받는다.		(최) "배대리, 당장 내방으로 오게"	전화벨 소리 재생	
	(배) 다른 손으로 머리를 긁는다.		(배) "알겠습니다. 지금 가겠습니다."		
	(배) 문쪽으로 걸어가서 노크한다.		(배) "과장님, 배대리입니다. 들어가도 되겠습니까?"		
	(배) 문을 천천히 열고 들어간다.		(최) "들어오게"		
	(최) 배대리를 향해 삿대질을 한다.		(최) "배대리, 지금 자네가 저지른 일이 회사에 얼마나 큰 손실을 끼친지 아는가?"		
	(배) 고개를 숙인다.		(배) "죄송합니다."		
	(배) 고개를 들고 한쪽 팔을 올린다.		(배) "하지만 그 일은..."		
	(최) 자리에서 일어서며 삿대질을 한다.		(최) "어디서 변명을 하나? 배대리 때문에 (중략) 그냥 사표를 쓰게나."		
	(배) 무릎을 꿇는다		(배) "과장님, 정말 죄송합니다. 다시는 실수 없도록 하겠습니다."		
	(최) 한쪽 팔로 문을 가리킨다.		(최) "당장 나가게! "	비장한 음악 파일 재생	(최) 슬픈 상태인 흰색으로 눈 LED 점등

2-1. 로봇 콘텐츠 개발

학습 목표

- 로봇 콘텐츠 설계서를 이용하여, 해당 로봇의 소프트웨어 플랫폼에 적용가능 하도록 로봇 콘텐츠 개발 계획을 수립할 수 있다.
- 로봇 콘텐츠가 실행될 로봇의 운영체제와 API(Application Programming Interface) 라이브러리 및 SDK(Software Development Kit)를 제공받아 로봇 콘텐츠 저작 환경을 구축할 수 있다.
- 운영체제에 따라서 웹용 콘텐츠나 앱용 콘텐츠 개발을 결정하고, 소프트웨어 편집기와 컴파일 환경을 결정할 수 있다.
- 로봇 콘텐츠의 메타모델을 정의하고, 타임라인이나 로봇 이벤트기반의 로봇 동작을 추가할 수 있다.
- 로봇 행위와 멀티미디어는 동기화되어 실행되도록 설정할 수 있다.
- 개발한 로봇 콘텐츠는 설계에 맞게 구현되었는지 테스트하고, 수정 점검할 수 있다.

필요 지식 /

[1] 로봇 콘텐츠 메타모델

로봇 콘텐츠 메타모델이란 멀티미디어 데이터와 로봇 동작을 시나리오 기반으로 실행하기 위해서 정의한 절차서를 의미한다.

배우의 음성과 효과음, 배경음악 등의 멀티미디어 데이터는 로봇 드라마와 같은 콘텐츠에서 대사를 전달하고 극적인 효과를 높이는 데 있어 중요한 역할을 하므로 로봇의 동작과 잘 동기화되어 실행할 수 있어야 한다.

1. 로봇 콘텐츠 저작 소프트웨어

로봇 콘텐츠 제작을 위해 멀티미디어와 동작을 편집해서 통합 구현할 수 있는 소프트웨어로는 한국의 OPRoS(Open Platform for Robotic Services), 로보이드 스튜디오(Roboid Studio), ROCOS(Robot Contents Organizing Software), 미국의 ROS(Robot Operating System), EU의 OROCOS 등이 있다.

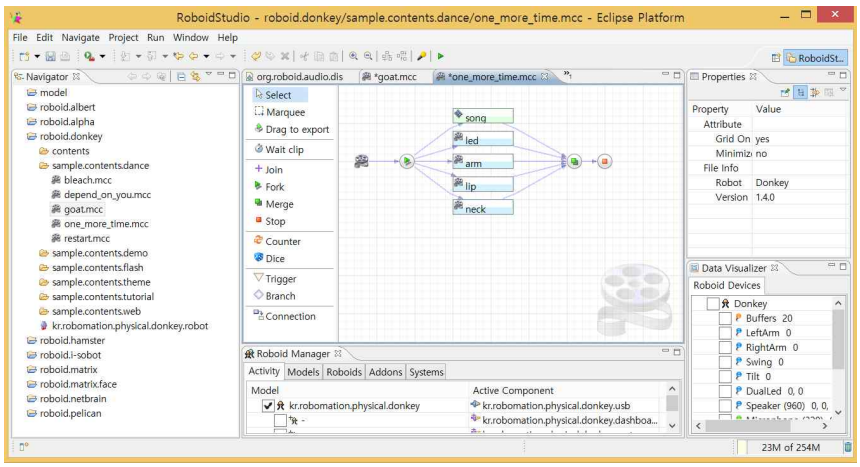
(1) OPRoS

OPRoS는 2007년 12월부터 개발이 시작 된 컴포넌트 기반의 오픈소스 플랫폼이며 통합 개발환경(integrated development environment, IDE)과 로봇에서 동작하는 프레임워크, 서버, 컴포넌트 및 소프트웨어의 시험 및 검증도구로 구성되어 있다(http://ropros.org/display/opros/OPRoS+Wiki).

개발자는 통합개발환경에서 컴포넌트와 콘텐츠를 개발하고 모니터링 및 디버깅, 그리고 시뮬레이션을 통합적으로 수행할 수 있으며, 프레임워크에서는 로봇의 컴포넌트 실행 관리와 결함 관리 등을 수행하며, 서버는 컴포넌트 저장소와 에이전트 또는 로봇 컴포넌트의 프록시 서비스 실행을 관리한다.

(2) 로보이드 스튜디오

로보이드 스튜디오는 로봇 콘텐츠를 GUI(graphic user interface)로 손쉽게 구성할 수 있는 도구로서, 2008년 5월에 첫 버전이 공개되었으며 현재 버전 1.4.2까지 발표되었다(www.roboidstuio.org). 로보이드 스튜디오는 철저하게 콘텐츠 중심이며, 일반인도 쉽게 사용할 수 있도록 시간 축을 따라 동작을 지시하는 편집 도구(타임라인 모션 에디터)와 이러한 동작들을 이용하여 상황에 따라 지능적으로 행동하게 하는 저작 도구(콘텐츠 컴포우저) 등을 제공한다. 또 비트맵 제어 방식을 사용하여 명령어로는 표현하기 어려운 복잡한 동작도 표현할 수 있으며, 멀티미디어 데이터와 로봇 동작 간의 동기화를 기본적으로 제공한다. [그림 2-1]은 로보이드 스튜디오를 컴퓨터에 설치하고(한정혜, 2011) 예제를 실행했을 때 나타나는 화면이다.



출처: 로보이드 스튜디오. 스크린샷.
[그림 2-1] 로보이드 스튜디오의 예제 실행 화면

(3) 스크립트 기반 로봇 콘텐츠 저작

멀티미디어가 사용되는 로봇 콘텐츠 제작의 핵심 기술 중 하나는 멀티미디어 재생과 로봇 모션을 정확히 동기화하는 것이며, GUI로 제작된 로봇 콘텐츠 제작 소프트웨어가 없는 경우 텍스트 기반의 로봇 콘텐츠 메타모델 파일을 작성한다.

2 로봇 콘텐츠 실행 로봇

로봇 콘텐츠를 개발하기 위해서는 콘텐츠를 구현할 로봇의 기구적 구조와 하드웨어, 소프트웨어에 대한 사전 지식이 필요하므로 로봇 콘텐츠를 구현할 로봇 플랫폼을 먼저 정해야 한다. 학습모듈 1-1에서 다루었던 로봇 드라마와 같이 로봇 콘텐츠는 인간의 삶과 문화, 역사, 가치들을 주제로 하는 경우가 많기 때문에 로봇의 구조도 가급적 사람과 비슷하여야 실감 있게 구현하는 데 유리하다. 이러한 로봇 플랫폼으로 휴머노이드 로봇이 가장 적합하다.

휴머노이드 로봇은 인체의 구조를 본 따서 머리와 사지, 몸통으로 구성되어 있다. 대표적인 성인급(adult size) 휴머노이드 로봇으로 일본에는 HONDA의 ASIMO, AIST의 HRP, 한국에는 KAIST의 HUBO와 로보티즈의 똘망(THOR-MANG), 미국에는 BOSTON DYNAMICS의 ATLAS, 이탈리아에는 IIT의 iCog, 독일에는 IRM의 TORO 등이 있으며, 어린이급(kid-sized) 휴머노이드 로봇으로는 프랑스 알데바란로보틱스의 나오(NAO), 한국 로보티즈의 ROBOTIS-OP가 있다. 또 상체는 휴머노이드 로봇이지만 하체에는 바퀴가 장착되어 있어 보다 안정된 이동성을 보장하는 휴머노이드 로봇으로 소프트뱅크의 페퍼(PEPPER)와 로보케어의 실벗(SILBOT) 등이 있다.

본 학습모듈에서는 휴머노이드 로봇 중 일반적인 로봇 콘텐츠를 구현하는 데 필요한 기본적인 센서(카메라, 마이크, 관성센서 등)와 표현모듈(모터, 스피커, 눈 LED 등)을 모두 갖춘 ROBOTIS-OP 시리즈를 대상으로 로봇 콘텐츠를 구현한다.

1. ROBOTIS-OP

ROBOTIS-OP(이하 OP)의 이전 이름은 DARwIn-OP였으며 2011년에 첫 번째 모델을 출시한 이후 전 세계적으로 수백 대가 판매되었다. ROBOTIS-OP2(이하 OP2)는 DARwIn-OP의 하드웨어 사양을 좀 더 향상시켜 2015년에 새롭게 출시되었다.

(1) 운영체제

OP2의 상체에 내장되어 있는 SBC(single board computer)의 운영체제(operating system, OS)는 Lubuntu 리눅스 12.04 32비트(Kernel 3.2.66, “Precise Pangolin”)이다.

(2) 하드웨어

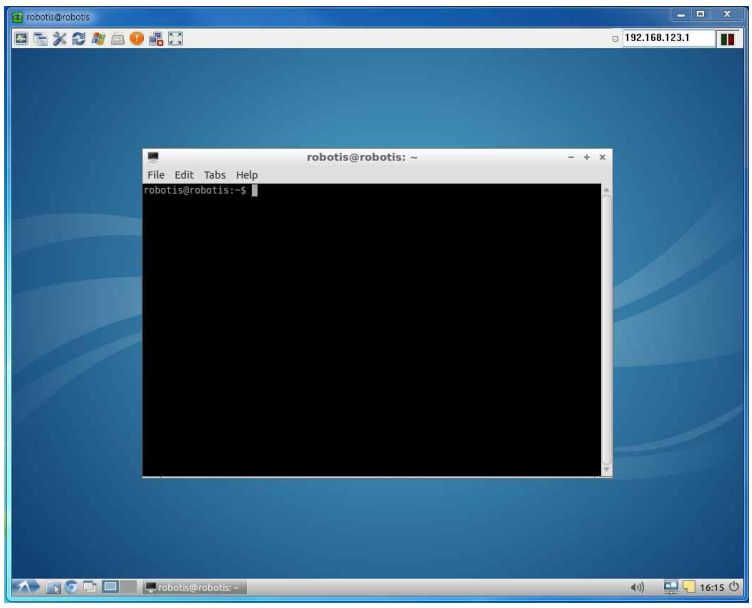
OP2는 키 45cm, 무게 2.9kg이며 하드웨어 사양은 <표 2-1>과 같다.

<표 2-1> OP2의 하드웨어 사양

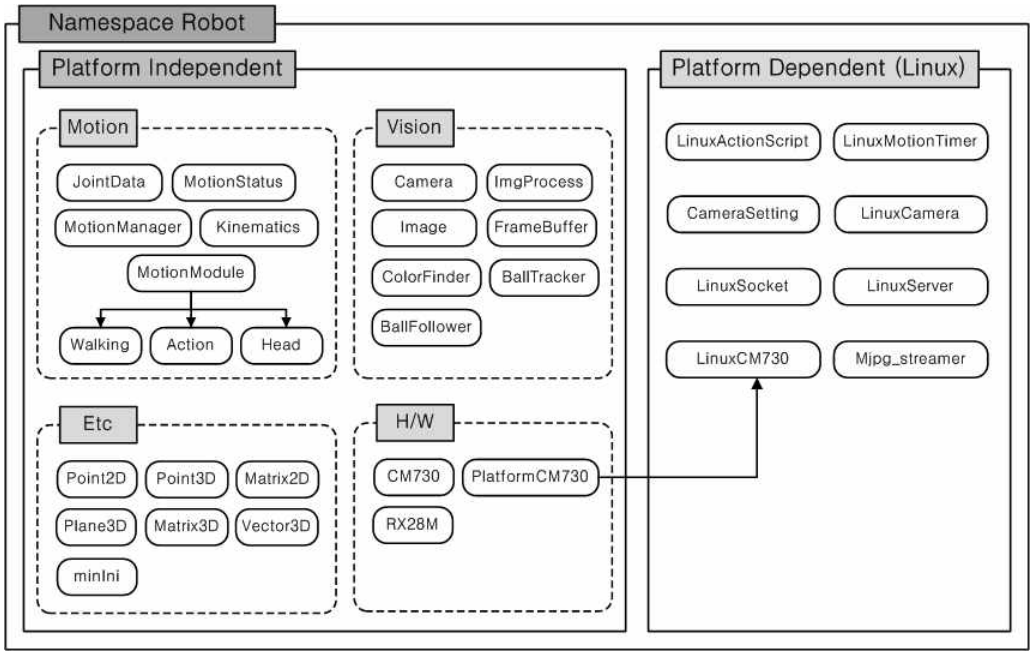
구분	부품 및 사양
CPU	Intel Atom N2600, 1.6 GHz, dual core
RAM	4GB DDR3 204핀 SO-DIMM 모듈
저장장치	32GB half-size mSATA 모듈
LAN 속도	1Gbps
OS	32비트 Linux 배포판 또는 32비트 Windows 버전
제어기	CM-740(ARM Coretex-M3 STM32F103RE 72 MHz)
액추에이터	DYNAMIXEL MX-28T, 20개
센서	3축 자이로스코프 센서, 3축 가속도 센서, 3개의 버튼, 2개의 감지 마이크

(3) 소프트웨어

OP2에서 모션과 멀티미디어를 동작시키기 위해서는 내장된 PC의 운영체제 상에서 동작하는 애플리케이션 프로그램을 사용자가 작성할 수 있어야 한다. 이를 위해 프로그램 개발이 용이한 데스크톱 PC에서 VNC(virtual network computing) 소프트웨어를 이용해서 OP2에 내장된 PC에 접속해야 한다. VNC 중에서 Windows에서 동작하는 오픈소스 응용프로그램인 UltraVNC Viewer를 사용하여 OP2의 내장 PC에 원격으로 연결하여 데스크톱 PC의 키 입력으로 OP2를 제어할 수 있다. [그림 2-2]는 UltraVNC 뷰어로 OP2에 접속하여 터미널을 실행한 화면이다.



출처: UltraVNC 뷰어. 스크린샷.
[그림 2-2] UltraVNC 뷰어로 OP2에 접속하여 터미널을 실행하는 화면



출처: 김종욱(2015a).『함께 즐기는 휴머노이드 로봇: 로보티즈 OP편』. 홍릉과학출판사. 249페이지.
[그림 2-3] OP2 프레임워크의 클래스와 인터페이스 구조

[그림 2-3]은 OP2 소프트웨어 프레임워크의 클래스와 인터페이스 구조를 나타낸다. 그림에서 이름 공간(namespace)이란 적용 분야가 비슷한 클래스, 구조체, 인터페이스, 열거 형식 등을 하나의 공간 안에 통합한 것을 의미한다. 그리고 플랫폼(platform)이란 OP2의 운영체제의 종류를 가리키는 용어로서, 플랫폼 독립(platform independent) 클래스란 OS와는 상관없는 라이브러리 파일을, 플랫폼 종속(platform dependent) 클래스란 OP2의 OS인 Linux와 관련된 클래스를 각각 의미한다. 각 클래스에 대한 자세한 자료는 로보티즈의 이매뉴얼(http://support.robotis.com/ko/)과 김종욱(2015a)의 교재를 참고하기 바란다. 이후부터 OP2의 프레임워크 상에서 로봇 콘텐츠를 개발하는 과정을 설명한다.

수행 내용 / 로봇 콘텐츠 개발하기

재료 · 자료

- 로봇 하드웨어 사양서
- 로봇 운영체제 사양서
- 로봇 기능 정의서
- 로봇 소프트웨어 플랫폼의 설명서
- 로봇 소프트웨어 개발 매뉴얼
- 오픈소스 지적재산권 지침서

기기(장비 · 공구)

- 컴퓨터, 프린터, 인터넷
- 로봇 소프트웨어 플랫폼의 소프트웨어 개발 환경툴(SDK)
- 프로그램 개발용 소프트웨어
- 컴파일러
- 로봇 콘텐츠를 실행할 로봇
- 문서 작성용 소프트웨어

안전 · 유의 사항

- 로봇 콘텐츠에도 저작권 관련 규정을 준수하여야 한다.
- 오픈 소스의 지적재산권 규정을 준수해야 한다.

수행 순서

① 로봇 콘텐츠 설계서를 이용하여, 콘텐츠를 구현할 로봇의 소프트웨어 플랫폼에 적용 가능하도록 로봇 콘텐츠 개발 계획을 수립한다.

1. 제작된 로봇 콘텐츠를 구현할 로봇을 선정한다.

   학습모듈 1에서 설명한 로봇 드라마('아버지의 이름으로')를 구현할 수 있는 휴머노이드 로봇으로 안정적인 동작에 필요한 기본적인 센서와 모듈을 갖춘 로보티즈의 OP2를 선정한다.

2. 선정한 로봇의 운영체제와 API(application programming interface) 라이브러리 및 SDK (software development kit)를 제공받아 로봇 콘텐츠 저작 환경을 구축한다.

OP2는 내장된 PC에 운영체제로 리눅스와 윈도우즈 중 하나를 설치할 수 있다. 로봇 콘텐츠 저작 도구로 ROS를 활용하기 위해 OP2 PC에 우분투 리눅스 Hydro 버전을 설치한다.

로보티즈는 OP2의 API 라이브러리로서 [그림 2-3]과 같이 다양한 클래스와 인터페이스로 구성 된 소프트웨어 프레임워크를 오픈소스로 제공하며, C++로 작성 된 소스 코드는 http://sourceforge.net/projects/darwinop 에서 다운로드할 수 있다.

3. 로봇의 소프트웨어 플랫폼에 적용 가능한 로봇 콘텐츠 개발 계획을 수립한다.

<표 1-5>의 로봇 콘텐츠 설계서에 대해 OP2를 위한 로봇 콘텐츠 개발 계획을 수립한다. 이로부터 로봇 콘텐츠 개발에 필요한 비용과 시간을 산정한다.

로봇 콘텐츠 개발팀을 구성하는데, 로보라마의 경우 크게 프로듀서, OP2 모션 제작팀(모션 제작 및 눈 LED 색깔 설정), 멀티미디어(음성과 음향 효과 파일) 제작팀으로 구분할 수 있다.

<표 1-5>의 Scene 1 로봇 콘텐츠 설계서로부터 만들어야 할 모션의 종류는 11가지, 음성 파일은 10개, 음악/음향 효과 파일이 4개임을 파악하고 그에 따라 모션 제작팀과 멀티미디어 제작팀의 인원수를 정한다.

<표 2-2>는 모션 제작팀의 개발 계획표를 나타낸다. Scene 1에 필요한 모션의 단위 모션과 난이도를 분석하고 예상 제작 시간을 계산해 보면 총 370분(6시간 10분)이 필요하다. 이로부터 몇 명이 몇 시간 동안 모션 제작 작업을 해야 할지 예측할 수 있다. 복합 모션을 단위 모션으로 구분하는 이유는 중복된 단위 모션의 경우 새로 만들 필요 없이 만들어 둔 모션을 재활용하면 효율적이기 때문이다. 예를 들어 <표 2-2>에서 '오른팔로 가리키기' 동작은 세 번 재사용됨을 알 수 있다. 그리고 이 동작은 가리키는 대상을 영상으로 인식한 후 그 방향으로 가리켜야 하므로 난이도가 상급이다.

OP2의 모션 제작팀은 눈 LED 색깔 조정도 함께 수행하며, 복합적인 모션은 단위 모션을 이어붙이거나 동시에 수행함으로써 효율적으로 만들 수 있다.

멀티미디어 제작팀은 <표 1-5>의 최과장과 배대리의 음성을 녹음할 사람 두 명을 섭외하고, 그렇지 못할 경우를 대비해 TTS 소프트웨어를 컴퓨터에 설치한다.

TTS 소프트웨어로 많이 사용되는 것은 Google의 TTS API의 파이썬(Python) 인터페이스인 gTTS(https://github.com/pndurette/gTTS)와 카네기멜론대학교의 Flite(festivial-lite)에 기반하여 Mycroft A.I.와 VocaliD가 개발한 Mimic(https://github.com/MycroftAI/mimic)이 있다.

<표 2-2> 로봇 콘텐츠 개발 계획: 모션 생성 개발 계획

Scene 번호	1	작성 날짜		작성자	
콘텐츠 개요	'아버지의 이름으로' 로봇 드라마의 Scene 1 설계서이며, 최과장이 배대리의 업무상 실수에 대해 화를 내고 문책하는 상황				

로봇 모션 제작 계획

모션	단위 모션	구현 난이도	예상 제작 시간
(최) 의자에 앉은 채 오른손으로 가슴을 친 후 전화를 건다.	- 의자에 앉기 - 오른손으로 가슴 치기 - 왼손으로 전화 걸기 - 눈 LED: 붉은색	상 (복합 모션)	40분
(배) 전화를 받는다.	- 왼손으로 전화 받기	하	20분
(배) 다른 손으로 머리를 긁는다.	- 오른손으로 머리 긁기	하	20분
(배) 문쪽으로 걸어가서 노크한다.	- 보행(1) - 노크하기	상 (보행과 연계)	40분
(배) 문을 천천히 열고 들어간다.	- 문열기 - 보행(2)	상 (보행과 연계)	40분
(최) 배대리를 향해 삿대질을 한다.	- 오른팔로 가리키기(1)	상 (영상인식)	40분
(배) 고개를 숙인다.	- 고개 숙이기	하	20분
(배) 고개를 들고 한쪽 팔을 올린다.	- 고개 들기 - 한쪽 팔 올리기	중	30분
(최) 자리에서 일어서며 삿대질을 한다.	- 자리에서 일어서기 - 오른팔로 가리키기(2)	상 (영상인식)	40분
(배) 무릎을 꿇는다.	- 무릎 꿇기 - 눈 LED: 흰색	상 (자세 안정성)	40분
(최) 한쪽 팔로 문을 가리킨다.	- 오른팔로 가리키기(3)	상 (영상인식)	40분
	합계		370분

2 로봇 콘텐츠 개발 계획에 의거하여 로봇 콘텐츠를 개발한다.

1. 로봇 콘텐츠의 메타모델을 작성한다.

로봇 콘텐츠의 메타모델은 로봇 동작과 멀티미디어 데이터를 시나리오 기반으로 실행하기 위해 정의한 절차서이며, <표 2-3>과 같이 타임라인 기반의 표로 작성하면 GUI 방식과 스크립트 방식 모두에 적용할 수 있다.

<표 2-3> 로봇 콘텐츠 메타모델

Scene 번호	1	작성 날짜		작성자	
콘텐츠 개요	‘아버지의 이름으로’ 로봇 드라마의 Scene 1 설계서이며, 최과장이 배대리의 업무상 실수에 대해 화를 내고 문책하는 상황				

컷별 모션과 멀티미디어 파일

시간	모션	음성 재생	배경 음악 /효과음 재생	눈 LED 색깔 설정
00:00:00	(최) 합성모션1: 의자에 앉기, 손으로 가슴 치기		음악파일1 (긴장감 있는 음악)	
	전화 걸기		음향파일1 (전화벨 소리)	
00:00:15	(배) 전화 받기	(최) 음성파일1 (“배대리, 당장 내 방으로 오게”)		
00:00:30	(배) 오른 손으로 머리 긁기	(배) 음성파일2 (“알겠습니다. 지금 가겠습니다.”)		
00:00:50	(배) 합성모션2: 문쪽으로 걸어가기			
	노크하기		음향파일2 (노크소리)	
		(배) 음성파일3 (“과장님, 배대리입니다. 들어가도 되겠습니까?”)		(최) 붉은색 (배) 파란색 (기본)
00:01:30	(배) 합성모션3: 문을 천천히 열기, 들어가기			
00:02:00	(최) 배대리를 향해 삿대질 하기	(최) 음성파일4 (“배대리, 지금 자네가 저지른 일이 회사에 얼마나 큰 손실을 끼쳤는지 아는가?”)		
00:02:30	(배) 고개 숙이기	(배) 음성파일5 (“죄송합니다.”)		
00:03:00	(배) 합성모션4: 고개 들기, 한쪽 팔 올리기	(배) 음성파일6 (“하지만 그 일은”)		
00:03:15	(최) 합성모션5: 자리에서 일어서기, 삿대질 하기	(최) 음성파일7 (“어디서 변명을 하나? 배대리 때문에 (중략) 그냥 사표를 쓰게나.”)		
00:03:40	(배) 무릎꿇기	(배) 음성파일8 (“과장님, 정말 죄송합니다. 다시는 실수 없도록 하겠습니다.”)		(배) 흰색
00:04:20 00:04:40	(최) 한쪽 팔로 문 가리키기	(최) 음성파일9 (“당장 나가게!”)		

로봇 콘텐츠의 메타모델은 교육용 로봇 콘텐츠와 같이 사용자와의 인터랙션이 발생하는 경우에 로봇 이벤트 기반으로도 정의할 수 있다. 이를 위해서는 Soar와 같은 인지 에이전트 아키텍처가 사용되어야 하는데, 이에 대해서는 3-1절에서 다룬다.

2. 로봇 모션과 멀티미디어를 동기화한다.

로봇 모션과 멀티미디어를 동기화하기 위해서는 멀티미디어 파일들을 다운로드하거나 생성하여 각 파일의 시간을 측정 후 그에 맞춰 모션을 생성해야 한다.

<표 2-3>에서 멀티미디어로는 음성과 효과음, 배경음악 파일이 있다.

효과음 1과 2는 전화벨 소리와 노크 소리로서 웹사이트에서 다운로드할 수 있다.

배경음악 파일은 긴장감이 감도는 음악파일을 구한다.

음성파일은 두 명의 성우 또는 학생이 직접 녹음하거나 TTS를 사용해서 mp3 파일 또는 wav 파일을 생성한다.

로봇 모션을 만들 때는 생성한 멀티미디어 파일의 시간을 측정한 후 그에 맞춰서 적절한 시간으로 만든다.

OP2의 액션에디터(Action Editor)로 모션을 만드는 경우 [그림 1-3]~[그림 1-5]에 보인 액션에디터 페이지 창 아래 부분의 'Time(x 8msec)' 값을 조정하여 원하는 실행 시간을 정할 수 있다. 예를 들어 모션 한 스텝의 실행 시간을 1초(=125×8ms)로 정하고 싶다면 해당 스텝의 'Time' 값에 125를 입력하면 된다.

MATLAB 시뮬레이터로 모션을 만드는 경우 전체 모션 수행의 목표 시간을 입력하면 그에 맞춰 모든 관절의 회전각 궤적이 조정된다.

<표 2-3>의 메타모델의 컷(cut)별로 멀티미디어 재생과 모션을 동시에 수행하면서 동기화가 잘 되는지 확인한다.

3. 개발한 로봇 콘텐츠는 설계에 맞게 구현되었는지 테스트하고 수정한다.

지금까지 학습모듈 1-1에서 작성한 로봇 콘텐츠 설계서의 첫 번째 씬에 대해 로봇 콘텐츠 메타모델을 수립하고 멀티미디어와 모션을 동기화하는 방법에 대해 설명하였다. <표 1-3>에서 보듯이 '아버지의 이름으로' 라는 로보라마는 6개의 씬으로 구성되어 있으므로 모든 씬에 대해 상기 작업을 수행하고 각 씬의 컷별로 멀티미디어와 모션 파일을 만들어 동기화를 테스트하고 수정한다.

학습2 교수 · 학습 방법

교수 방법

- 로봇 콘텐츠 설계서에 따라 콘텐츠를 구현할 로봇을 정하고, 사용되는 소프트웨어 플랫폼을 분석하여 그에 맞는 콘텐츠 개발 계획을 수립할 수 있도록 지도한다.
- 학생들이 로봇 콘텐츠를 저작할 수 있도록 콘텐츠를 구현할 로봇의 운영체제와 API 라이브러리 및 SDK에 대해 설명하고 설치하는 방법을 안내한다. 이때 운영체제에 따라 콘텐츠 개발 용도(웹용과 애플리케이션용), 소프트웨어 편집기, 컴파일 환경 등을 함께 고려해서 결정하도록 지도한다.
- 로봇 콘텐츠의 메타모델을 정의하는 방법을 설명하고, 타임라인이나 로봇 이벤트 기반의 로봇 동작을 추가하는 방법에 대해 가르친다.
- 로봇 콘텐츠 메타모델에서 로봇의 동작과 멀티미디어가 정확히 동기화되어 실행될 수 있도록 시간을 배정한 것인지를 확인한다.

학습 방법

- 로봇 콘텐츠 설계서에 따라 콘텐츠를 구현하기에 적절한 로봇을 검색과 토론을 통해 정할 수 있고, 이 로봇이 사용하는 소프트웨어 플랫폼을 분석하여 콘텐츠 개발 계획을 수립할 수 있다.
- 콘텐츠를 구현할 로봇의 운영체제와 API 라이브러리 및 SDK를 설치하는 법을 배우고 매뉴얼과 책, 각종 자료를 통해 사용법을 익힐 수 있다.
- 로봇 콘텐츠의 메타모델을 정의할 수 있고, 설계한 메타모델에 타임라인이나 로봇 이벤트를 기반으로 새로운 로봇 모션을 추가할 수 있다.
- 로봇 콘텐츠 메타모델에서 로봇의 동작과 멀티미디어가 정확히 동기화되어 실행되도록 설정할 수 있다.

학습2 평 가

평가 준거

- 평가자는 학습자가 학습 목표를 성공적으로 달성하였는지를 평가해야 한다.
- 평가자는 다음 사항을 평가한다.

학습 내용	평가 항목	성취수준		
		상	중	하
로봇 콘텐츠 개발	- 로봇 콘텐츠 설계서를 이용하여, 해당 로봇의 소프트웨어 플랫폼에 적용 가능토록 로봇 콘텐츠 개발 계획을 수립할 수 있다.			
	- 로봇 콘텐츠가 실행될 로봇의 운영체제와 API 라이브러리 및 SDK를 제공받아 로봇 콘텐츠 저작 환경을 구축할 수 있다.			
	- 운영체제에 따라서 웹용 콘텐츠나 앱용 콘텐츠 개발을 결정하고, 소프트웨어 편집기와 컴파일 환경을 결정할 수 있다.			
	- 로봇 콘텐츠의 메타모델을 정의하고, 타임라인이나 로봇 이벤트 기반의 로봇 동작을 추가할 수 있다.			
	- 로봇의 모션과 멀티미디어는 동기화되어 실행되도록 설정할 수 있다.			
	- 개발한 로봇 콘텐츠는 설계에 맞게 구현되었는지 테스트하고, 수정 점검할 수 있다.			

평가 방법

- 포트폴리오

학습 내용	평가 항목	성취수준		
		상	중	하
로봇 콘텐츠 개발	- 로봇 콘텐츠 개발 계획서			
	- 로봇 콘텐츠의 저작 환경 구축 설명서			
	- 웹용 또는 앱용 콘텐츠의 소프트웨어 편집기와 컴파일 환경 설명서			
	- 로봇 콘텐츠의 메타모델 설명서			
	- 로봇의 모션과 멀티미디어 동기화 과정 설명서			
	- 로봇 콘텐츠 테스트 및 점검 수행 일지			

● 서술형 시험

학습 내용	평가 항목	성취수준		
		상	중	하
로봇 콘텐츠 개발	- 로봇 콘텐츠 개발 계획 시 중요 사항 이해			
	- 로봇 콘텐츠 저작 환경에 대한 지식			
	- 웹용 또는 앱용 콘텐츠의 차이점 이해			
	- 로봇 콘텐츠 메타모델의 정의 및 요구 사항			
	- 로봇의 모션과 멀티미디어 동기화의 필요성 이해			
	- 로봇 콘텐츠 테스트 및 점검 방법 이해			

● 사례 연구

학습 내용	평가 항목	성취수준		
		상	중	하
로봇 콘텐츠 개발	- 로봇 콘텐츠 개발 계획서 사례			
	- 로봇 콘텐츠 저작 환경 구축 사례			
	- 우수한 웹용 콘텐츠와 앱용 콘텐츠 사례			
	- 우수한 로봇 콘텐츠 메타모델 사례			
	- 로봇의 모션과 멀티미디어 동기화의 성공/실패 사례			
	- 로봇 콘텐츠 테스트 및 수정/점검 방법 사례			

● 구두 발표

학습 내용	평가 항목	성취수준		
		상	중	하
로봇 콘텐츠 개발	- 로봇 콘텐츠 개발 계획 발표			
	- 로봇 콘텐츠 저작 환경 구축 과정 발표			
	- 콘텐츠 개발 과정 및 소프트웨어 개발 환경 결정 과정 발표			
	- 설계한 로봇 콘텐츠 메타모델 발표			
	- 로봇의 모션과 멀티미디어 동기화 방안 발표			
	- 로봇 콘텐츠 테스트 및 점검 방법 발표			

피 드 백

1. 포트폴리오
   - 로봇 콘텐츠를 개발하기 위해 작성한 설명서의 품질에 대한 수준을 서로 비교하고 평가한 결과를 공유한다.

2. 서술형 시험
   - 평가 결과 성적이 저조한 학생은 로봇 콘텐츠 개발에 관한 보고서 제출 기회를 부여한다.

3. 사례 연구
   - 로봇 콘텐츠 개발과 관련한 주요 사례에 관해 연구한 내용을 평가한 후 우수한 사례는 정보를 공유한다.

4. 구두 발표
   - 로봇 콘텐츠 개발 프로젝트를 수행하여 그 결과물을 발표한 후 문제점을 지적하여 개선 방안을 논의한다.

3-1. 로봇 콘텐츠 실행 소프트웨어 개발

학습 목표

- 저작된 로봇 콘텐츠를 로봇 소프트웨어 플랫폼에서 실행 가능토록 개발 계획을 수립할 수 있다.
- 로봇 소프트웨어 플랫폼에서 적합한 로봇 콘텐츠 실행 소프트웨어를 설계할 수 있다.
- 로봇 콘텐츠의 메타모델 해석기로부터 로봇 동작과 멀티미디어를 각각 실행 가능토록 수행할 수 있다.
- 콘텐츠를 실행하여 로봇 동작과 멀티미디어 구현에서 에러 발생 유무를 테스트할 수 있다.

필요 지식 /

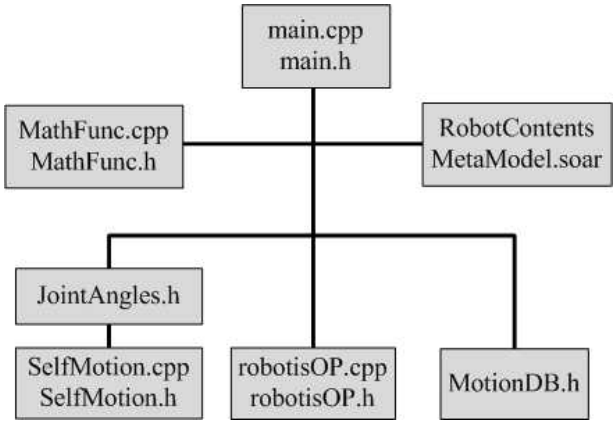
1 로봇 콘텐츠 실행 소프트웨어 제작

1. 휴머노이드 로봇용 실행 소프트웨어 제작

(1) 소프트웨어 패키지 구조

김종욱(2015a)은 이족보행을 포함한 다양한 모션을 OP2에서 구현할 수 있도록 optWalk라는 패키지를 C++로 작성하고 소스코드를 공개했다.

휴머노이드 로봇으로 로봇 콘텐츠를 구현할 경우 <표 2-3>의 로봇 콘텐츠 메타모델에서 볼 수 있듯이 보행 기반 동작(문을 열고 들어가기)이 많이 들어가게 되므로 본 학습모듈에서는 optWalk에 로봇 콘텐츠 메타모델 해석기를 추가한 roborama-op 패키지를 사용한다. [그림 3-1]은 roborama_op 패키지의 파일 관계도를 나타내는데, 이 패키지는 휴머노이드파크 웹사이트(http://humanoidpark.kr)의 자료실에서 다운로드할 수 있다.



[그림 3-1] roborama_op 패키지의 파일 관계도

[그림 3-1]에 보인 roborama_op 패키지의 각 파일의 내용 및 기능은 다음과 같다.

- main.cpp, main.h: 텍스트 기반 UI(user interface)와 로봇 콘텐츠 메타모델을 이용한 전체 콘텐츠 흐름 제어
- MathFunc.cpp, MathFunc.h; 각종 수치 연산에 필요한 수학적 함수 정의
- RobotContentsMetaModel.soar: 구현할 로봇 콘텐츠의 메타모델을 포맷에 맞춰 작성한 Soar 코드
- JointAngles.h: OP의 모든 관절각 관련 파라미터 정의
- SelfMotion.cpp, SelfMotion.h: OP의 다양한 동작 생성 및 제어
- robotisOP.cpp, robotisOP.h: OP의 규격 정보 저장 및 기구학/동역학 연산 수행
- MotionDB.h: OP의 모든 동작과 관련된 관절변수와 관절궤적 데이터 저장

로봇 콘텐츠 메타모델용 Soar 파일을 제외한 각 함수에 대한 상세한 설명은 김종욱의 책(2015a)을 참고하기 바란다.

(2) 로봇 모션 생성 소프트웨어

로봇 콘텐츠에서 로봇으로 인간의 다양한 모션을 생동감 있게 표현하는 것이 대단히 중요하다. 휴머노이드 로봇은 인간의 신체를 이용한 모션을 가장 잘 표현할 수 있는 로봇 플랫폼이며, 대부분의 휴머노이드 로봇은 로봇에 대한 전문 지식이 없어도 쉽게 로봇의 자세와 모션을 만들 수 있는 모션 편집기 소프트웨어를 제공한다.

OP를 이용해서 모션을 만드는 방법에는 크게 두 가지가 있다. 첫 번째는 로보티즈에서 제공하는 모션 편집 소프트웨어인 액션에디터를 사용하는 것이고, 두 번째는 김종욱(2015a)이 제안한 모델 기반 모션 생성 기법을 사용하는 것이다. 두 번째 방법은 MATLAB과 SelfMotion.cpp(roborama_op 패키지 구성) 파일에 C++ 함수(SerialFrameMotion)로 구현되어 있다.

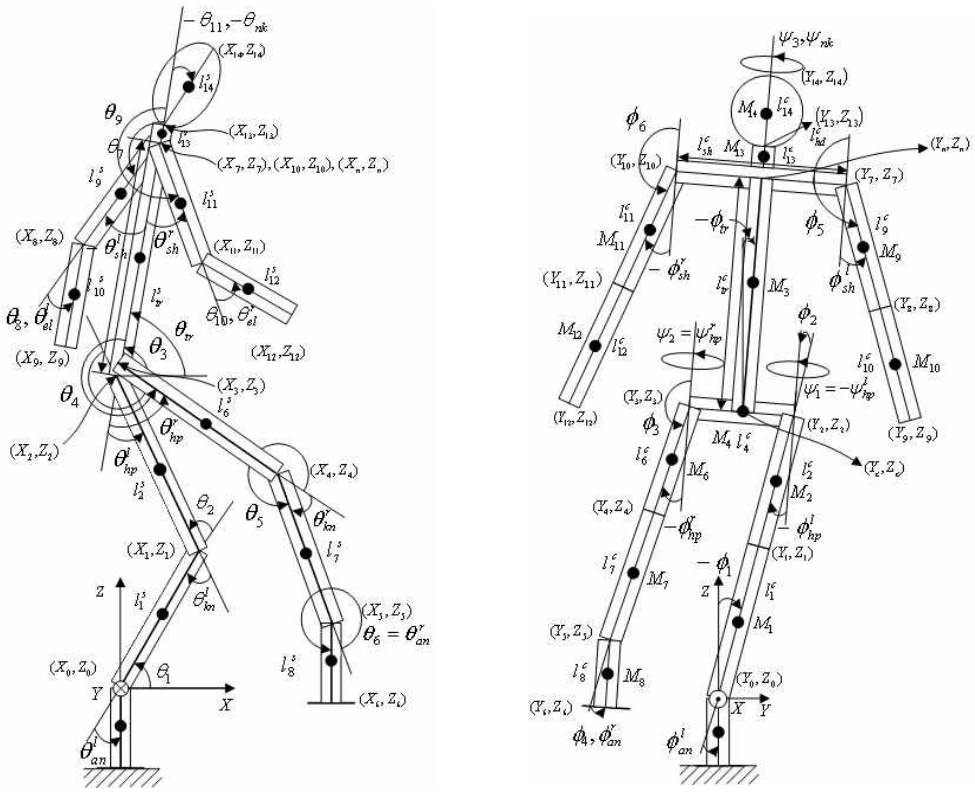
(가) 액션에디터

액션에디터는 로봇을 수동 조작 모드(액추에이터의 토크를 OFF 상태로 전환)로 설정한 상태에서 원하는 모션의 각 주요 자세(motion primitive)를 사용자가 손으로 직접 만들고 그 자세에 대한 전체 액추에이터 회전각을 제어기가 저장하는 과정을 매 주요 자세 별로 반복 진행한 후 재생(play)을 하면 OP가 주요 자세 사이의 프레임(frame)별 자세에 내삽(interpolation) 관절각을 적용하여 부드럽게 움직이는 원리로 모션을 편집 및 제작할 수 있게 한다.

이 방법은 로봇에 대한 전문 지식 없이도 쉽게 복잡한 동작을 만들 수 있지만, 미리 만들어 놓은 모션을 재사용 또는 편집해서 쓸 수 없고, 다양한 보행 특성(속도, 보폭, 정서 표현)을 가진 이족보행을 구현할 수 없으며, 로봇 콘텐츠 제작을 위해 모션과 멀티미디어 데이터를 연동할 수 없는 단점이 있다.

(나) 모델 기반 모션 생성

김종욱(2015a)은 DH(Denavit-Hartenberg)법과 투영법(projection-based method)을 기반으로 OP의 전신을 [그림 3-2]와 같이 모델링하고 이로부터 토크와 영모먼트점(zero moment point)을 계산하고 이족보행을 포함한 다양한 모션을 생성하는 방법을 MATLAB과 C++로 구현한 코드를 작성해서 공개했다.



출처: 김종욱(2015a).『함께 즐기는 휴머노이드 로봇: 로보티즈 OP편』. 홍릉과학출판사. 288페이지.
[그림 3-2] 왼발로 서 있는 상태인 휴머노이드 로봇의 3차원 전신 모델

이를 위해 전신을 시상면(sagittal plane, 옆에서 볼 때의 평면), 관상면(frontal plane, 앞에서 볼 때의 평면), 횡평면(transversal plane, 위에서 볼 때의 평면)으로 구분하고 시상면에는 $\theta_1 \sim \theta_{11}$, 관상면에는 $\phi_1 \sim \phi_6$, 횡평면에는 $\psi_1 \sim \psi_3$의 상대관절변수(relative joint variable)를 설정했다.

한 가지 주의할 점은 휴머노이드 로봇의 경우 지지하는 다리에 따라 기준 좌표계가 바뀌므로, 걸어갈 때 하체의 상대관절변수들의 위치가 왼쪽에서 오른쪽으로 서로 교환되는 특성이 있다. 예를 들어 [그림 3-2]에서 왼발로 서 있는 경우에는 왼쪽 발목관절각이 θ_1이지만, 오른발로 서 있는 경우에는 오른쪽 발목관절각이 θ_1이 되고 왼쪽 발목관절각은 θ_6로 바뀌게 된다.

이를 해결하기 위해 각 관절에 고유한 절대관절변수(absolute joint variable)를 할당할 필요가 있다. 절대관절변수는 [그림 3-2]에서 보듯이 위첨자는 해당 관절의 위치에 따라 l(왼쪽) 또는 r(오른쪽)을 붙이고 아래첨자는 관절의 위치를 나타내는 an(발목), kn(무릎), hp(대퇴부), sh(어깨), el(팔꿈치), nk(목) 중의 하나를 붙인다. 이렇게 하면 모션 제작 시 관절변수의 이름만 봐도 직관적으로 그 관절의 위치를 알 수 있게 되어 매우 편리하다.

1) 모션 프리미티브를 이용한 모션 생성

모션 프리미티브(motion primitive)란 어떤 동작을 이루는 최소의 주요 자세를 의미한다. 예를 들어 반가움을 표시하기 위해 왼쪽 팔을 흔드는 모션은 직립 자세, 왼쪽 팔을 수직으로 들어 올린 자세, 왼쪽 팔을 좌측으로 60° 들어 올린 자세만 있으면 이 자세들을 연결하여 구현할 수 있으므로 이 세 자세가 반가움을 표시하는 동작의 모션 프리미티브가 된다.

[그림 3-2]를 참고로 하여 각 자세에 대해 다음과 같은 시상면, 관상면, 횡평면 절대관절변수 벡터의 값을 적절히 정한다.

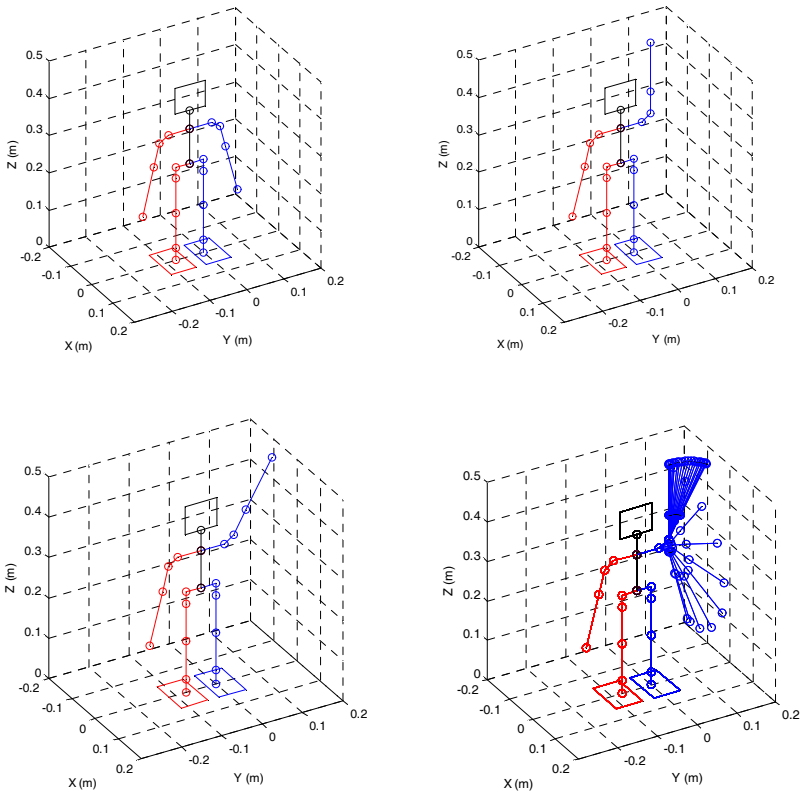
$$\Theta = \left[\theta^l_{an}\ \theta^l_{kn}\ \theta^l_{hp}\ \theta^r_{hp}\ \theta^r_{kn}\ \theta^r_{an}\ \theta^l_{sh}\ \theta^l_{el}\ \theta^r_{sh}\ \theta^r_{el}\right]$$

$$\Phi = \left[\phi^l_{an}\ \phi^l_{hp}\ \phi^r_{hp}\ \phi^r_{an}\ \phi^l_{sh}\ \phi^r_{sh}\right],\ \Psi = \left[\psi^l_{hp}\ \psi^r_{hp}\ \psi_{nk}\right] \qquad \text{식 (1)}$$

상기 세 가지 모션에 대한 모션 프리미티브의 절대관절변수 벡터가 다음과 같이 정해졌다고 하자.

- 직립 자세: $\Theta_{stand}, \Phi_{stand}, \Psi_{stand}$
- 왼팔을 수직으로 들어 올린 자세: $\Theta_{lh90\mathrm{deg}}, \Phi_{lh90\mathrm{deg}}, \Psi_{lh90\mathrm{deg}}$
- 왼팔을 60° 들어 올린 자세: $\Theta_{lh60\mathrm{deg}}, \Phi_{lh60\mathrm{deg}}, \Psi_{lh60\mathrm{deg}}$

이러한 모션 프리미티브 절대관절변수각의 내삽 계산값을 이용하여 각 샘플 타임별로 부드러운 자세를 생성하기 위해 각 관절의 궤적을 혼합다항식(blending polynomial)으로 표현한다.



[그림 3-3] 반가움을 표시하는 동작 시뮬레이션

작업지능 소프트웨어 개발하기(LM1903080308_14v1.1) 학습모듈의 식 (6)과 [그림 2-6]에 표현 된 혼합다항함수 궤적이 m개의 세그먼트로 구성되어 있고, 경유점의 시간벡터가 $[t_0, t_1, \cdots, t_m]$, 각도 벡터가 $[\theta_0, \theta_1, \cdots, \theta_m]$, 각속도벡터가 $[\omega_0, \omega_1, \cdots, \omega_m]$이라고 할 때, 관절각의 전체 궤적을 다음과 같이 표현하자.

$$q(t) = BP_m(t, [t_0, t_1, \cdots, t_m], [\theta_0, \theta_1, \cdots, \theta_m], [\omega_0, \omega_1, \cdots, \omega_m]) \quad \text{식 (2)}$$

이제 반가움을 표시하는 모션을 구현하기 위해 시상면의 $k(k=1,\cdots,11)$번째 절대관절변수의 궤적을 혼합다항함수로 표현하면 다음과 같다.

$$\theta_k(t) = BP_4(t, [0,1,3,5,6], [\Theta_{stand}(k), \Theta_{lh90\deg}(k), \Theta_{lh60\deg}(k),$$
$$\Theta_{lh90\deg}(k), \Theta_{stand}(k)], [0,0,0,0,0]), \ 0 \le t \le 6 \quad \text{식 (3)}$$

여기서 경유점 시간벡터가 $[0,1,3,5,6]$인 것은 팔을 들어 올리고 내리는 시간은 1초씩, 팔을 흔드는 시간은 2초씩 정해준 것을 의미하며, 상황이나 기분에 따라 다르게 조정할 수 있다. [그림 3-3]은 반가움을 표시하는 동작의 세 가지 모션 프리미티브의 자세를 보이고, 혼합다항함수를 사용하여 이 자세들을 연결하여 생성한 모션을 시뮬레이션한 것이다.

2) 동작 합성

<표 2-3>에서 보듯이 로봇 콘텐츠로 구현하려는 일상적인 사람의 동작은 두 가지 이상의 동작을 연결하거나 동시에 수행하는 경우가 많다. 먼저, 두 동작을 부드럽게 연결해야 할 때 두 세트의 관절궤적을 어떻게 처리해야 하는지 알아보자.

어떤 두 모션에 대해 휴머노이드 로봇의 k번째 관절의 목표 회전각 궤적이 $q_k^{(1)}(t)$와 $q_k^{(2)}(t)$이고, 각 궤적이 다음과 같이 상이한 3개의 세그먼트와 2개의 세그먼트로 구성된 혼합다항함수로 표현된다고 하자.

$$q_k^{(1)}(t) = BP_2\left(t, \left[t_0^{(1)}, t_1^{(1)}, t_f^{(1)}\right], \left[\theta_0^{(1)}, \theta_1^{(1)}, \theta_f^{(1)}\right], \left[\omega_0^{(1)}, \omega_1^{(1)}, \omega_f^{(1)}\right]\right)$$

$$q_k^{(2)}(t) = BP_1\left(t, \left[t_0^{(2)}, t_f^{(2)}\right], \left[\theta_0^{(2)}, \theta_f^{(2)}\right], \left[\omega_0^{(2)}, \omega_f^{(2)}\right]\right) \quad \text{식 (4)}$$

두 동작을 연결시키기 위해 k번째 관절의 두 회전각 궤적을 부드럽게 이어붙이려면 두 번째 궤적의 시작시간($t_0^{(2)}$)이 첫 번째 궤적의 종료시간($t_f^{(1)}$)과 같아야 하며, 첫 번째 궤적의 마침각도($\theta_f^{(1)}$)와 두 번째 궤적의 시작각도($\theta_0^{(2)}$)의 차이값($\theta_d^{(12)} = \theta_f^{(1)} - \theta_0^{(2)}$)을 두 번째 궤적의 각도벡터($\left[\theta_0^{(2)}, \theta_f^{(2)}\right]$)에 더해야 한다. 또 궤적을 부드럽게 잇기 위해 두 번째 궤적의 시작각속도($\omega_0^{(2)}$)를 첫 번째 궤적의 마침각속도($\omega_f^{(1)}$)와 동일하게 해야 한다. 다음 식은 이 두 가지 조건을 적용하여 두 궤적 $q_k^{(1)}(t)$와 $q_k^{(2)}(t)$를 이어붙인 궤적 $q_k^{12c}(t)$를 수식으로 표현한 것이다.

$$q_k^c(t) = \begin{cases} BP_2\left(t, \left[t_0^{(1)}, t_1^{(1)}, t_f^{(1)}\right], \left[\theta_0^{(1)}, \theta_1^{(1)}, \theta_f^{(1)}\right], \left[\omega_0^{(1)}, \omega_1^{(1)}, \omega_f^{(1)}\right]\right), & t_0^{(1)} \le t \le t_f^{(1)} \\ BP_1\left(t, \left[t_f^{(1)}, t_f^{(1)} + (t_f^{(2)} - t_0^{(2)})\right], \left[\theta_d^{(12)} + \theta_0^{(2)}, \theta_d^{12} + \theta_f^{(2)}\right], \left[\omega_f^{(1)}, \omega_f^{(2)}\right]\right), & \\ \qquad\qquad t_f^{(1)} \le t \le t_f^{(1)} + (t_f^{(2)} - t_0^{(2)}) \end{cases}$$

식 (5)

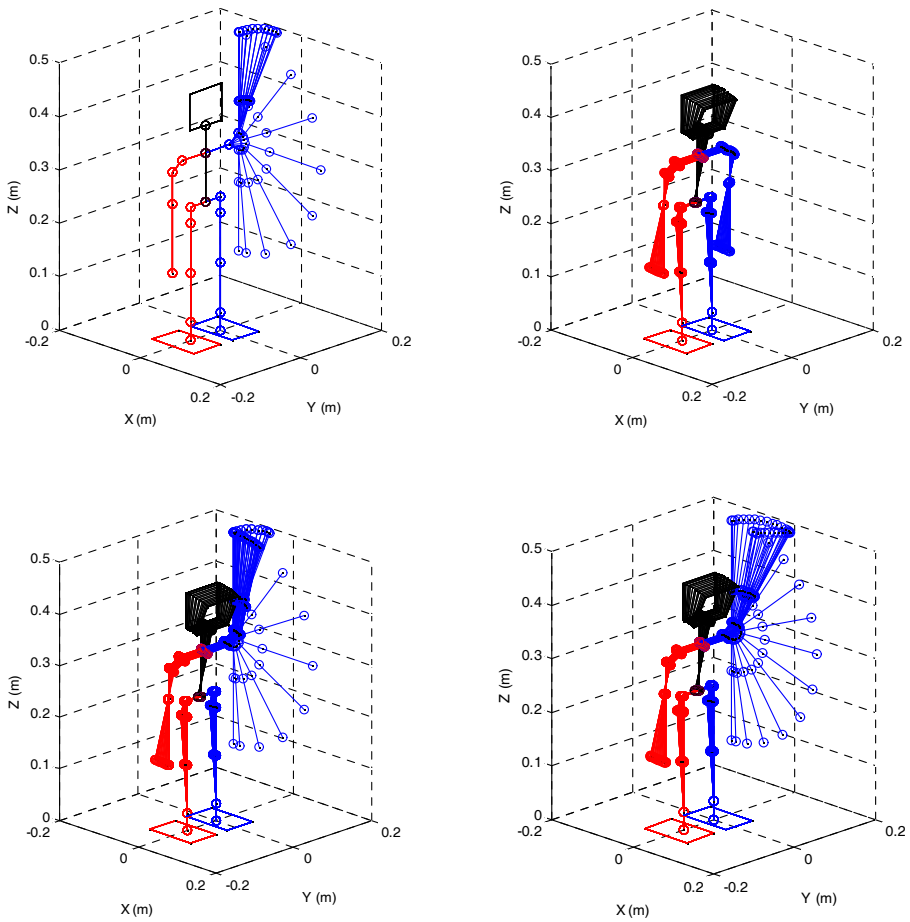
그 다음으로 두 동작을 합성하여 동시에 표현하기 위해서는 두 궤적을 더해야 한다. 그런데 두 궤적의 종료시간이 다를 경우, 더 긴 궤적의 종료시간에 맞춰 더 짧은 궤적을 늘여준 다음에 더한다. 이를 위해 시간비율을 다음과 같이 계산한다.

$$\alpha_k^{12} = \max(t_f^{(1)}, t_f^{(2)}) / \min(t_f^{(1)}, t_f^{(2)}) \qquad \text{식 (6)}$$

식 (4)에서 두 번째 궤적이 더 짧은 경우($t_f^{(2)} < t_f^{(1)}$), 이 시간비율을 이용하여 두 궤적을 합하는 것은 다음 수식으로 표현된다.

$$q_k^s(t) = BP_2\left(t, \left[t_0^{(1)}, t_1^{(1)}, t_f^{(1)}\right], \left[\theta_0^{(1)}, \theta_1^{(1)}, \theta_f^{(1)}\right], \left[\omega_0^{(1)}, \omega_1^{(1)}, \omega_f^{(1)}\right]\right) + \qquad \text{식 (7)}$$

$$BP_1\left(t, \left[t_0^{(2)}, \alpha_k^{12} t_f^{(2)}\right], \left[\theta_0^{(2)}, \theta_f^{(2)}\right], \left[\omega_0^{(2)}, \frac{\omega_f^{(2)}}{\alpha_k^{12}}\right]\right), \; t_0^{(1)} \le t \le t_f^{(1)}$$



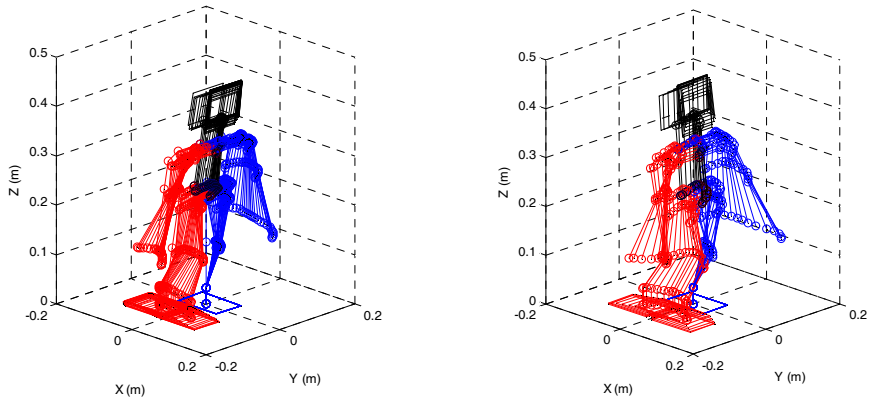
[그림 3-4] 동작 합성 시뮬레이션(손 흔들기, 인사, 손 흔든 후 인사하기, 손 흔들면서 인사하기)

[그림 3-4]는 손 흔드는 동작과 인사하는 동작에 대한 각 관절의 궤적을 만든 후 식 (5)를 이용해서 두 동작을 이어서 수행하는 것과 식 (7)을 이용해서 두 동작을 동시에 수행하는 것을 시뮬레이션으로 보인 것이다.

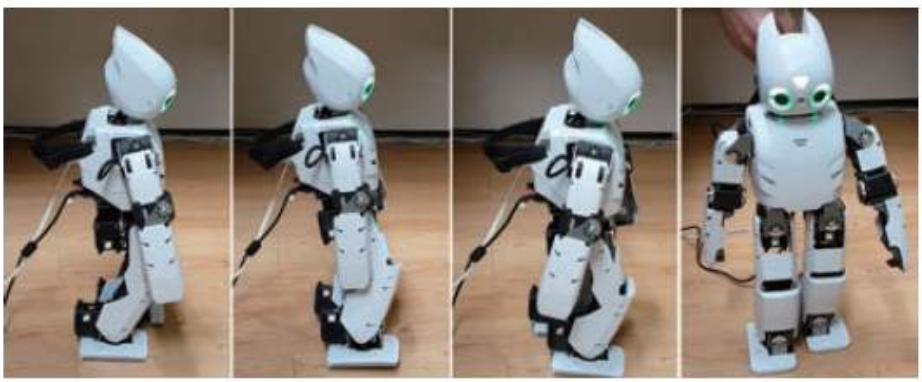
3) 정서적 보행

휴머노이드 로봇의 안정적인 이족보행을 위해서는 설정된 보폭과 발높이 만큼 걸어가면서 안정적인 ZMP(zero moment point, 영모멘트점) 궤적을 따라갈 수 있도록 하체의 주요 관절변수 궤적을 혼합다항함수로 표현하고, 궤적 경유점의 파라미터들을 최적화 알고리즘인 PSO(particle swarm optimization)로 최적화하는 것이 중요하다(최낙윤 · 최영림 · 김종욱, 2013).

김종욱(2015b)은 전신의 관절궤적 파라미터들을 조정하여 보행 속도와 보폭, 상체 각도 등을 조정하고 눈 LED색깔을 적절히 변화시킴으로써 로봇 배우의 감정 상태를 알고리즘으로 용이하게 표현할 수 있음을 보였다. 관절궤적 조정 원리를 설명하기 위해 m개의 세그먼트로 구성된 혼합다항함수의 식 (2)를 다음과 같이 수정해서 표현하자.



[그림 3-5] 슬픈 상태와 기쁜 상태에서의 보행 시뮬레이션



[그림 3-6] 슬픈 상태에서 걸어가는 OP2의 스냅샷

$$q'(t) = BP_m(t, \frac{1}{\eta}[t_0, t_1, \cdots, t_m], \kappa[\theta_0, \theta_1, \cdots, \theta_m], \eta\kappa[\omega_0, \omega_1, \cdots, \omega_m]) \quad \text{식 (8)}$$

여기서 η는 궤적의 시간축을 조정하는 계수로서 η가 1보다 크면 관절의 회전속도가 증가하고, 1보다 작으면 감소한다. 그리고 κ는 각도의 배율을 나타내며 κ가 1보다 크면 각도값이 확대되고, 1보다 작으면 축소된다. 그러므로 이족보행을 위한 하체 관절궤적의 η 값을 1 이상으로 증가시키면 보행속도가 비례해서 빨라지고, κ 값을 1 이상으로 증가시키면 보폭이 그만큼 커진다. [그림 3-5]는 다음과 같이 설정함으로써 정서적 보행을 시뮬레이션한 것을 보인다(θ_{tr}: 상체각).

- 슬픈 상태: $\eta = 0.7, \kappa = 0.8, \theta_{tr} = 80^\circ$
- 기쁜 상태: $\eta = 1.4, \kappa = 1.2, \theta_{tr} = 95^\circ$

[그림 3-6]은 상기 시뮬레이션을 통해 검증한 전신의 관절궤적을 OP2에 적용하여 슬픈 상태에서 걸어가는 것을 실험으로 구현한 사진이다. OP2는 두 눈에 LED가 있기 때문에 감정에 따라 색깔을 조정함으로써 더 효과적으로 감정을 표현할 수 있다. 상기 실험에서는 슬플 때 LED가 흰색이 되도록 설정했으며, 화가 나 있는 경우는 붉은색으로 변화시켰다.

2. 로봇 콘텐츠 메타모델 구현

OP2를 위한 roborama_op 패키지는 리눅스를 기반으로 프로그래밍되어 있으므로 로봇 콘텐츠 메타모델을 스크립트 기반으로 작성하는 것이 적합하다.

<표 2-3>에서 보듯이 로봇 콘텐츠의 메타모델은 타임라인 기반 또는 로봇 이벤트 기반으로 다양한 모션과 함수, 파일들을 호출하므로, 모든 요소들이 계획대로 잘 동기화(synchronization)되어야 한다. 또 드라마 공연 중 예기치 못한 상황이 발생했을 때 로봇들이 정지하는 것이 아니라 대안이 되는 다른 모션이나 파일들을 선택함으로써 적절히 대응할 수 있어야 한다. 이를 위해서는 단순한 스크립트 기반 알고리즘이 아니라 다양한 상황을 고려하고 추론하며 학습할 수 있는 인공지능 에이전트 아키텍처가 필요하다.

이를 위해 여러 가지 인공지능 에이전트 아키텍처 중 작업지능 소프트웨어 개발하기(LM1903080308_14v1.1) 학습모듈에서 소개한 Soar(state, operator, and result)를 사용한다.

(1) Soar를 이용한 로봇 콘텐츠 메타모델 구현

Soar는 장기적으로 기억을 저장하기 위해 제작 메모리(production memory), 시맨틱 메모리(semantic memory), 일화성 메모리(episodic memory) 모듈을 갖추고 있다. 로봇 콘텐츠를 구현하는 로봇은 콘텐츠 메타모델을 기반으로 타임라인이나 로봇 이벤트에 따라 적절한 모션이나 멀티미디어 파일을 선택할 수 있어야 하므로, 메타모델의 내용을 담은 시맨틱 메모리를 제작 메모리에 연결하여 사용하면 된다.

(가) Soar의 요소

로봇 콘텐츠 메타모델 생성을 위해 Soar를 사용하면 각 요소들은 다음과 같이 해석될 수 있다.

1) state(상태): 현재 실행 중인 콘텐츠 시나리오의 경과 시간과 로봇의 모션 수행, 음성·음향 효과 재생 등의 상태
2) operator(연산자): 메타모델 타임 라인이나 로봇 이벤트에 따라 적절한 라이브러리, 함수 또는 파일을 불러와서 로봇 콘텐츠를 연속적으로 실행
3) result(결과 또는 목표): 자연스럽고 극적인 로봇 콘텐츠 구현을 통한 시청자의 만족도 및 감동

(나) Soar의 프로세스

Soar는 기본적인 결정(decision) 사이클이 있는데 로봇 콘텐츠 메타모델에 적용하면 다음과 같다.

1) 환경으로부터의 입력(input): 콘텐츠 실행 경과 시간 또는 사용자의 상호 작용
2) 현재 상태의 정교화(state elaboration): 현재 실행 중인 콘텐츠 컷의 경과 시간과 로봇의 모션, 재생 중인 멀티미디어 파일들을 확인하고, 모션 실행에 오류가 있거나 해당 멀티미디어 파일이 없는 경우 현재 상태를 나타내는 WME(working memory element) 수정
3) 연산자(operator) 제안(proposal) 및 평가(evaluation): 현재 모션이나 멀티미디어 파일 재생이 종료되면 다음 모션과 멀티미디어 파일을 메타모델 시맨틱 메모리에서 읽어 옴. 만약 해당 모션 함수/변수나 멀티미디어 파일이 없으면 대체 가능한 모션이나 파일을 모두 검색하여 그 적절성을 평가함. 예를 들어 휴머노이드 로봇의 플랫폼이 이족보행 로봇에서 모바일 베이스 로봇으로 바뀔 경우 Soar가 이를 감지하여 '걸어가기' 모션에 대해 이족보행이 아니라 바퀴를 구동하는 함수를 호출하여 제안하고 적용 가능성을 평가함.
4) 연산자 선택(operator selection): 제안된 모션과 멀티미디어 파일 후보들의 선호도(acceptable, reject, better/worse, best, worst, indifferent 중 하나)를 기반으로 최적의 로봇 모션과 음성·음향 효과 파일 하나씩을 선택함.
5) 연산자 적용(operator application): 선택 된 모션의 실행 및 멀티미디어 파일 재생 명령을 OP2에 전송
6) 출력(output): OP2의 모션 실행 및 멀티미디어 파일 재생

수행 내용 / 로봇 콘텐츠 실행 소프트웨어 개발하기

재료 · 자료

- 로봇 하드웨어 사양서
- 로봇 운영체제 사양서
- 로봇 소프트웨어 플랫폼의 설명서
- 로봇 소프트웨어 개발 매뉴얼
- 오픈소스 지적재산권 지침서

기기(장비 · 공구)

- 컴퓨터, 프린터, 인터넷
- 로봇 소프트웨어 플랫폼의 소프트웨어 개발 환경툴(SDK)
- 프로그램 개발용 소프트웨어
- 컴파일러
- 로봇 콘텐츠를 실행할 로봇
- 문서 작성용 소프트웨어

안전 · 유의 사항

- 안전을 위해 로봇의 작동 매뉴얼을 숙지하여 로봇을 실행해야 한다.
- 오픈 소스의 지적재산권 규정을 준수해야 한다.

수행 순서

1 제작된 로봇 콘텐츠 메타모델을 이용하여 로봇 소프트웨어 플랫폼에 적합한 로봇 콘텐츠 실행 소프트웨어를 제작할 수 있다.

1. 로봇의 모션 제작

로봇 콘텐츠 메타모델의 컷별 모션을 로봇으로 구현하기 위해 다음의 단계를 거쳐야 한다.

(1) 모션 프리미티브 제작

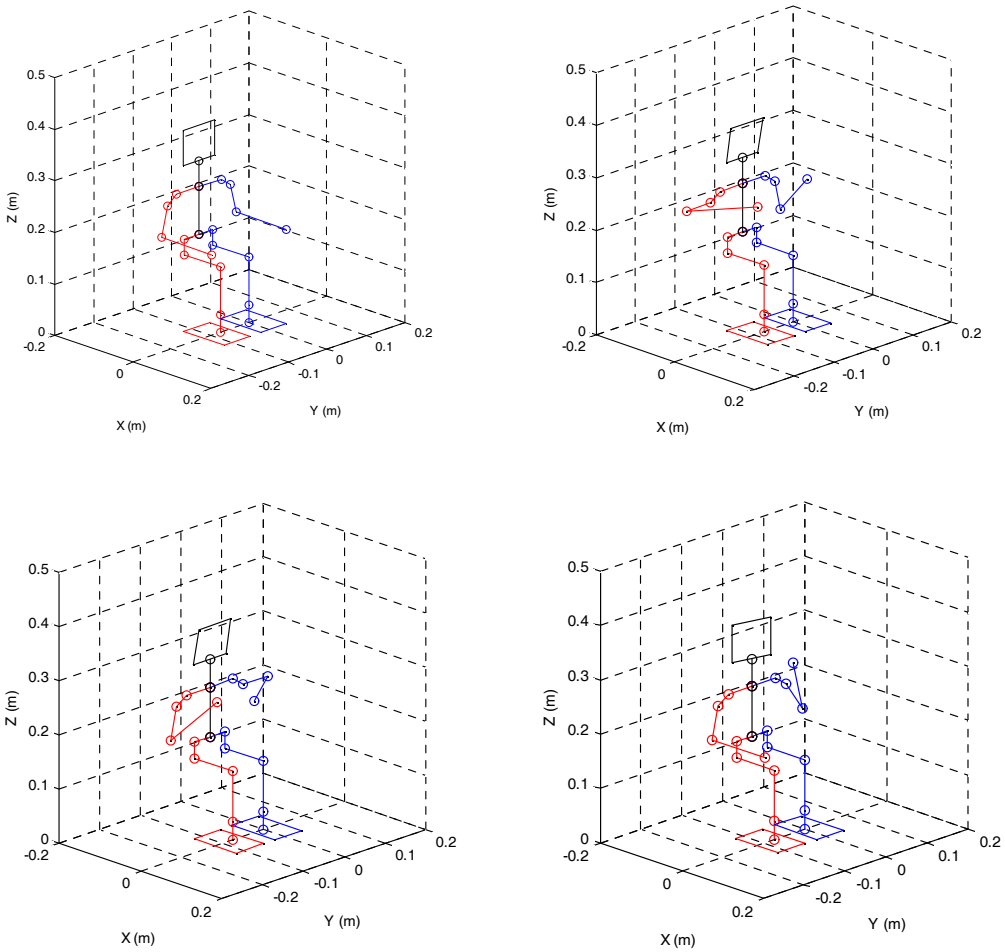
로봇의 전신에 있는 관절모터의 회전각을 적절히 설정함으로써 로봇 콘텐츠 메타모델의 컷별 동작에 대한 전신 모션 프리미티브를 얻는다.

OP2 로봇을 보유하고 있는 경우 액션에디터를 실행하여 로봇의 각 링크를 손으로 직접 조작하며 모션 프리미티브를 얻고, 그렇지 않은 경우 MATLAB 시뮬레이터를 사용해서 모션 프리미티브를 생성한다.

휴머노이드파크 웹사이트의 '로봇기술 자료실' 페이지에서 'robot drama (matlab)' 폴더를 다운로드하고, 이 폴더의 sim_pose.m 파일에 시상면, 관상면, 횡평면의 절대관절변수 벡터를 아래와 같이 정해줌으로써 [그림 3-7]과 같이 메타모델(<표 2-3>)의 Scene 1의 첫 번째 컷에 필요한 '의자에 앉기', '오른팔로 가슴 치기', '왼팔로 가슴 치기', '왼손으로 전화 걸기/받기' 모션 프리미티브들을 생성한다.

```
                        ⋮
if no_pose == 1          % 직립한 자세
    mth = [0 0 0 0 0 0   0 0 0 0   0];  % 시상면 관절각
    mphi = [0 0 0 0 14 -14];            % 관상면 관절각
    mpsi = [0 0 0];                     % 횡평면 관절각
elseif no_pose == 2      % 의자에 앉은 자세
    mth = [0 90 90 90 90 0   0 90 0 90   0];
    mphi = [0 0 0 0 14 -14];
    mpsi = [0 0 0];
elseif no_pose == 3      % 오른팔로 가슴 치기
    mth = [0 90 90 90 90 0   0 135 0 145 -10];
    mphi = [0 0 0 0 14 -90];
    mpsi = [0 0 0];
                        ⋮
elseif no_pose == 5      % 왼팔로 전화 걸기/받기
    mth = [0 90 90 90 90 0   10 155 0 90   0];
    mphi = [0 0 0 0 30 -14];
    mpsi = [0 0 -10];
else
                        ⋮
ang = TransMotorAng2GeomAng(mth, mphi, mpsi, 'non', 'deg', sf);
th = ang(1:11);    phi = ang(12:17);      psi = ang(18:20);

res = GenMotionsWithDH(sf, sp, th, th_1d, th_2d, phi, phi_1d, phi_2d, psi, psi_1d, psi_2d, fig_no);
```



[그림 3-7] 메타모델의 Scene1의 첫 번째 컷에 대한 모션 프리미티브(앉기, 오른팔로 가슴 치기, 왼팔로 가슴 치기, 전화 걸기 자세)

(2) 연속 모션 시뮬레이션

앞에서 얻은 모션 프리미티브의 절대관절각 벡터를 이용해서 식 (3)과 같은 방법으로 각 관절별 회전각 궤적을 만든 후 MATLAB 시뮬레이터로 모션을 생성한다.

휴머노이드파크 웹사이트의 '로봇기술 자료실' 페이지에서 'robot drama (matlab)' 폴더를 다운로드하고, 이 폴더의 sim_roborama1.m 파일을 열면 다음 페이지와 같은 코드를 볼 수 있다.

times1 벡터는 첫 번째 컷에서 각 모션 프리미티브가 수행되는 시간을 의미하며, <표 2-3>에서 이 시간을 15초로 계획했으므로 여기에 맞춰서 각 단위 모션의 간격을 조절한다. 만약, 계획보다 시간이 더 걸린다거나 덜 걸리면 메타모델의 시간을 전체적으로 조절하면 된다.

```
                        ⋮
no_motion = 2;      % 1: 손 흔드는 모션    % 2: Scene1 Cut1 모션
                        ⋮
%%%  의자에 앉은 자세
mth_sit_down = [0 90 90 90 90 0   0 90 0 90   0];
mphi_sit_down = [0 0 0 0 14 -14];
mpsi_sit_down = [0 0 0];
%%%  오른팔로 가슴 치기
mth_hit_chest_right_arm = [0 90 90 90 90 0   0 135 0 145 -10];
mphi_hit_chest_right_arm = [0 0 0 0 14 -90];
mpsi_hit_chest_right_arm = [0 0   0];
                        ⋮
%%%  왼팔로 전화 걸기/받기
mth_phone_call_left_hand = [0 90 90 90 90 0   10 155 0 90   0];
mphi_phone_call_left_hand = [0 0 0 0 30 -14];
mpsi_phone_call_left_hand = [0 0   -10];
                        ⋮
elseif no_motion == 2  % 씬1 컷1에 대한 모션 생성
    times1 = [0 2 4 8 10 12 15];
    mth_motion1 = [mth_stand_up; mth_sit_down; mth_hit_chest_right_arm; mth_hit_chest_left_arm;
                   mth_hit_chest_right_arm; mth_hit_chest_left_arm; mth_phone_call_left_hand];
    mphi_motion1    =    [mphi_stand_up;          mphi_sit_down;          mphi_hit_chest_right_arm;

                   mphi_hit_chest_left_arm; mphi_hit_chest_right_arm; mphi_hit_chest_left_arm;
                   mphi_phone_call_left_hand];
    mpsi_motion1 = [mpsi_stand_up;   mpsi_sit_down;   mpsi_hit_chest_right_arm;
                   mpsi_hit_chest_left_arm; mpsi_hit_chest_right_arm; mpsi_hit_chest_left_arm;
                   mpsi_phone_call_left_hand];
end
                        ⋮
for i=1:ns
    t = (i-1)*st+Ti;
    ts(i) = t;
    [sag_an_l(i), sag_an_l_1d(i), sag_an_l_2d(i)] = CalcAngIn_NSegBP(t, times1,
            deg2rad(mth_motion1(:,1)), zeros(1, n_pos));
    [sag_kn_l(i), sag_kn_l_1d(i), sag_kn_l_2d(i)] = CalcAngIn_NSegBP(t, times1,
            deg2rad(mth_motion1(:,2)), zeros(1, n_pos));
    [sag_hp_l(i), sag_hp_l_1d(i), sag_hp_l_2d(i)] = CalcAngIn_NSegBP(t, times1,
            deg2rad(mth_motion1(:,3)), zeros(1, n_pos));
                        ⋮
    res = GenMotionsWithDH(sf, sp, th, th_1d, th_2d, phi, phi_1d, phi_2d, psi, psi_1d, psi_2d, 1);
end
```

(3) OP2에 적용하여 모션 최종 확인

지금까지 시뮬레이터로 구현한 첫 번째 컷 모션을 실제 로봇 플랫폼인 OP2에 적용하여 안정되고 구현 가능한 모션인지 확인한다.

휴머노이드파크 웹사이트의 '로봇기술 자료실' 페이지에서 roborama_op 패키지를 다운로드하고, 파일 중에서 SelfMotion.cpp 파일을 열면 다음 페이지의 SerialFrameMotion 함수를 찾을 수 있다.

roborama_op 패키지의 main.cpp 파일의 main 함수를 보면 아래와 같이 인사하는 모션을 SerialFrameMotion으로 구현한 코드를 볼 수 있다. 코드에서 인사하는 동작을 위한 모션 프리미티브의 절대관절변수 벡터 mths_bow와 mphis_bow는 MATLAB sim_roborama1.m 파일과 유사한 방식으로 MotionDB.h 파일에 기록되어 있다.

```
int main()
{
                        ⋮
    SelfMotion im;
                        ⋮
    im.no_frame = 2;
    for(fr=0; fr<=im.no_frame; ++fr){
        im.times[fr] = times_bow[fr];
        im.delays[fr] = 0;
        for(i=0; i<11; ++i)    im.mths[fr][i]=deg2rad(mths_bow[i][fr]);
        for(i=0; i<6; ++i)     im.mphis[fr][i]=deg2rad(mphis_bow[i][fr]);
         for(i=0; i<3; ++i)     im.mpsis[fr][i]=0;
    }
    SmoothMotionTransition(im, selected_motion);
    im.SerialFrameMotion(cm730, 0, 0.9);
                        ⋮
}
```

첫 번째 컷 모션의 네 가지 모션 프리미티브([그림 3-7])의 절대관절변수 벡터들을 MotionDB.h 파일에 형식에 맞춰 저장하고 모션의 안정성을 확인한다. 만약 OP2가 넘어지거나 동작이 부자연스러우면 경유점의 관절각이나 시간을 조절하여 안정된 모션으로 만들고, 더 이상 모션에 문제가 발생하지 않으면 모션 제작이 완료된 것이다.

```
void SelfMotion::SerialFrameMotion(CM730 &cm730, int mno, double sratio)
{
                        ⋮
    if(mno == 0){      // mno: 정상적인 모션이면 0, 모션 간 연결모션이면 1
        nf =  no_frame;
        t0 = times[0], tf = times[nf];
    }
                        ⋮
    for(int cp=0; cp<ns; ++cp){
        tStart = clock();
        if(mno == 0)    ct = cp*st + times[0];
                        ⋮
        while(i<nf){
            if(mno == 0){
                if(ct<times[i+1]){
                    it0=i; itf=i+1;        // it0: 모션 시작 프레임, itf: 모션 종료 프레임
                    t0 = times[it0], tf = times[itf];
                    break;
                }
                else ++i;
            }
                        ⋮
        for(k=0; k<11; ++k){
            if(mno == 0)    CalcInterpolateAng(ct, t0, tf, mths[it0][k], mths[itf][k], res);
            mphi[k] = res[0];
        }
        for(k=0; k<6; ++k){
             if(mno == 0)    CalcInterpolateAng(ct, t0, tf, mphis[it0][k], mphis[itf][k], res);
             mphi[k] = res[0];
        }
        for(k=0; k<3; ++k){
             if(mno == 0)    CalcInterpolateAng(ct, t0, tf, mpsis[it0][k], mpsis[itf][k], res);
             mpsi[k] = res[0];
        }
                        ⋮
        for(k=0; k<NUM_ACTUATOR; ++k){    // 다이나믹셀에 회전속도 값 전송
             cm730.WriteWord(k+1, MX28::P_MOVING_SPEED_L, GoalVel[k], 0);
        }
        TransMotorAng2MX28Position(mth, mphi, mpsi, GoalPos, "rad");
        SendPositionValue2AllDynamixel(cm730, cp);    // 다이나믹셀에 위치 값 전송하여 OP동작
                        ⋮
}
```

2. Soar를 이용한 로봇 콘텐츠 메타모델 제작

(1) 초기화

현재 상태를 조사하기 위해 Soar의 state인 s 식별자(identifier)의 ^read 속성(attribute)의 값(value)인 r 식별자로부터 시간(^time), 모션(^read), 소리(^sound), 직전모션(^last motion) 속성 정보를 읽어온다.

그리고 Soar의 시맨틱 메모리(smem)에 메타모델의 정보를 저장한다. 아래 코드에서는 시간이 5초 경과하면 motion 1(팔 흔들기) 수행, 10초 경과하면 motion 2(가슴 치기) 수행, 15초 경과하면 motion 3(전화 걸기) 수행 및 sound 1(벨소리) 재생, 25초 경과하면 동일한 모션 수행에 sound 2("배대리, 당장 내 방으로 오게") 재생의 메타모델을 시맨틱 메모리에 구현한 예를 보인다.

```
smem --set learning on
sp {propose*initialize-robot1_drama
   (state <s> ^type state
              -^name)
-->
   (<s> ^operator <o> +)
   (<o> ^name initialize-robot1_drama)
}
sp {apply*initialize-robot1_drama
   (state <s> ^operator.name initialize-robot1_drama
              ^smem.command <cmd>
              ^io.output-link <i3>)
-->
   (<s> ^name robot1_drama
        ^io-link-status changed
        ^read <r>)
   (<r> ^time 0 ^motion 0 ^sound 0 ^last_motion 0)
   (<i3> ^motion 0 ^sound 0)
   (<cmd> ^store <a> <aa> <b> <bb> <c> <cc> <d> <dd>
          ^retrieve <lti>)
   (<a> ^time 5 ^connect <aa>)
   (<aa> ^motion 1 ^sound 0)    # motion 1: swing arms ; sound 0: no sound.
   (<b> ^time 10 ^connect <bb>)
   (<bb> ^motion 2 ^sound 0)    # motion 2: hit chest ; sound 0: no sound.
   (<c> ^time 15 ^connect <cc>)
   (<cc> ^motion 3 ^sound 1)    # motion 3: call robot 2; sound 1: Bell ringing.
   (<d> ^time 25 ^connect <dd>)
   (<dd> ^motion 3 ^sound 2)    # motion 3: call robot 2; sound 2:  "Mr. Bae. Come here
      right now."
}
```

(2) 시맨틱 메모리 읽어오기

다음 코드는 현재 경과 시간(^time)을 측정해서 로봇 콘텐츠 메타모델이 저장되어 있는 시맨틱 메모리에 접근하여 모션과 소리를 불러오는(retrieve) 과정을 Soar 소스코드로 나타낸 것이다.

```
sp {robot1_drama*propose*ncbRetrieval
   (state <s> ^name robot1_drama
              ^io-link-status read
              ^read <r>
              ^smem.command <cmd>)
   (<r> ^time <v>)
   (<cmd> ^store <st>)
   (<st> ^time <v>
         ^connect <con>)
   (<con> ^motion <motion_id>
          ^sound <sound_id>)
-->
   (<s> ^operator <op> + =)
   (<op> ^name ncbRetrieval
         ^connect <con>)
}
sp {robot1_drama*apply*ncbRetrieval
   (state <s> ^name robot1_drama
              ^operator <op>
              ^read <r>
              ^io-link-status read)
   (<op> ^name ncbRetrieval
         ^connect <con>)
   (<con> ^motion <v1>
          ^sound <v2>)
   (<r> ^motion <v11>
        ^sound <v22>)
-->
   (<r> ^motion <v11> - <v1>
        ^sound <v22> - <v2>)
   (<s> ^io-link-status read - retrieved)
}
```

(3) 로봇 콘텐츠 구현

다음 코드는 로봇 콘텐츠 컷의 경과 시간에 따라 시맨틱 메모리에서 읽어온 모션 식별자(<motion_id>)와 음성 파일 식별자(<sound_id>)를 출력 링크(^io.output-link)로 보내서 OP로 동작과 목소리를 구현하는 Soar의 소스코드 부분이다.

```
...
sp {robot1_drama*propose*motion
   (state <s> ^name robot1_drama
              ^read <r>
              ^io-link-status retrieved)
   (<r> ^time << 5 15 20 25 >>
        ^motion <motion_id>
        ^sound <sound_id>)
-->
   (<s> ^operator <op> + =)
   (<op> ^name read_input
          ^motion <motion_id>
          ^sound <sound_id>)
}
sp {robot1_drama*apply*motion
   (state <s> ^name robot1_drama
              ^operator <o>
              ^io.output-link <i3>
              ^io-link-status retrieved)
   (<o> ^name read_input
        ^motion <v1>
        ^sound <v2>)
   (<i3> ^motion <v11>
        ^sound <v22>)
-->
   (<i3> ^motion <v11> - <v1>
        ^sound <v22> - <v2>)
   (<s> ^io-link-status retrieved - changed)
}
```

학습 1	로봇 콘텐츠 설계서 작성하기
학습 2	로봇 콘텐츠 개발하기
학습 3	로봇 콘텐츠 실행 소프트웨어 개발하기

학습 4 로봇 콘텐츠 저작 도구 개발하기

4-1. 로봇 콘텐츠 저작 도구 개발

학습 목표

- 설계된 로봇 콘텐츠를 가상 플랫폼에서 실행할 수 있는 저작 도구를 설계할 수 있다.
- 로봇 콘텐츠 저작 도구를 개발하기 위해 로봇의 하드웨어 소프트웨어 플랫폼 개발 환경을 활용할 수 있다.
- 로봇 동작과 멀티미디어를 동기화하기 위해 로봇 콘텐츠를 편집할 수 있다.
- 로봇 동작과 멀티미디어 콘텐츠가 동시에 실행되는지 시험평가할 수 있다.

필요 지식 /

[1] 로봇 콘텐츠 가상 플랫폼

1. Gazebo를 이용한 로봇 콘텐츠 실행

Gazebo는 복잡한 실내와 실외 환경에 대한 멀티로봇 시뮬레이터이다(http://gazebosim.org). Gazebo는 3차원 세계에서 다양한 로봇과 센서, 물체들을 그룹 형태로도 시뮬레이션 할 수 있으며, 실제와 유사한 센서 데이터를 얻고 물체간 물리적 상호 작용을 모사할 수 있다. Gazebo는 물리엔진(ODE, Bullet, Simbody, DART 등)을 사용하여 현실적인 동역학 시뮬레이션을 가능케 하며, 빛·그림자·텍스쳐를 실감나게 표현하는 3차원 그래픽 지원, LRF·Kinect·2D/3D 카메라 등의 센서 지원(노이즈 포함 가능), 로봇을 제어하기 위한 플러그인(plug-in) 추가, 다양한 로봇 모델 지원, TCP/IP 데이터 전송, 클라우드 시뮬레이션 지원 등의 기능으로 전 세계 로봇 개발자에게 각광을 받고 있다(표윤석, 2015).

Gazebo를 로봇 콘텐츠 가상 플랫폼으로 선정해서 로봇 콘텐츠를 실행하기 위해서는 [그림 4-1]과 같은 단계를 거쳐야 한다.

[그림 4-1] Gazebo를 이용한 로봇 콘텐츠 실행 순서

(1) ROS 패키지 설치 및 Gazebo와 인터페이스

ROS 커뮤니티에서는 다양한 로봇과 센서를 위해 우수한 기능성을 갖춘 패키지들이 지속적으로 공유 및 개발되고 있기 때문에 ROS는 인간과의 상호 작용 기능이 풍부한 실감형 로봇 콘텐츠를 구현하는 데에 적합하다.

ROS와 Gazebo를 연결하기 위해 다음의 순서를 따르도록 하자(ROS Indigo 버전).

- "$ gazebo" 명령을 입력하여 ROS에 Gazebo가 설치되었는지를 확인한다.
- "$ sudo apt-get install ros-indigo-gazebo-ros-pkgs ros-indigo-gazebo-roscontrol" 명령을 입력하여 ROS 패키지를 설치한다.
- "$ roscore & rosrun gazebo_ros gazebo" 명령을 입력하여 Gazebo와 ROS가 잘 연동하는지를 확인한다.

(2) 로봇 모델링

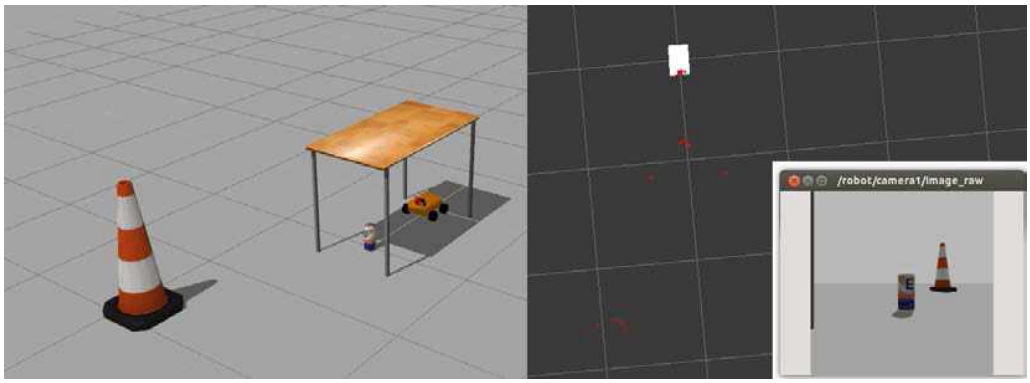
ROS는 로봇, 부품, 관절, 규격 등을 설명하는 XML 포맷인 URDF(unified robot description format) 파일을 이용해서 로봇이나 부품의 3차원 모델을 표현한다. Gazebo는 로봇 시뮬레이터와 시각화, 제어를 위해 물체와 환경을 설명하는 XML 포맷인 SDF(simulation description format)를 사용하지만, URDF의 <link>, <joint>, <robot> 요소에 <gazebo> 요소를 추가함으로써 URDF 파일도 변환하여 Gazebo에서 사용할 수 있다(http://gazebosim.org/tutorials/?tut=ros_urdf).

다음은 URDF의 이해를 돕기 위해 로봇 베이스에 네 개의 바퀴를 붙이는 예제 코드의 일부를 보인 것이다(Fernandez, Crespo and Mahtani et al., 2015). 이 형식을 따르면 로봇 콘텐츠에 사용되는 임의의 로봇을 ROS와 Gazebo에서 사용할 수 있도록 모델링할 수 있다.

```
<?xml version="1.0"?>
  <robot name="Robot1">
    <link name="base_link">
      <visual>
        <geometry>
          <box size="0.2 .3 .1"/>
        </geometry>
        <origin rpy="0 0 0" xyz="0 0 0.05"/>
        <material name="white">
          <color rgba="1 1 1 1"/>
        </material>
      </visual>
    </link>
    <link name="wheel_1">
      <visual>
        <geometry>
          <cylinder length="0.05" radius="0.05"/>
        </geometry>
        <origin rpy="0 1.5 0" xyz="0.1 0.1 0"/>
          <material name="black">
            <color rgba="0 0 0 1"/>
          </material>
        </visual>
      </link>
              ⋮
    <joint name="base_to_wheel1" type="fixed">
      <parent link="base_link"/>
        <child link="wheel_1"/>
        <origin xyz="0 0 0"/>
    </joint>
              ⋮
  </robot>
```

(3) Gazebo에 센서 추가

Gazebo에 센서의 동작 특성이 구현되어 있다면 센서 또한 Gazebo로 시뮬레이션할 수 있다. [그림 4-2]에서 Xacro(XML Macros) 파일을 수정하여 모바일 로봇에 Hokuyo 레이저 거리 측정 센서와 카메라를 부착한 후 "rostopic echo" 명령어를 입력해서 센서 데이터를 실제와 유사하게 Gazebo 시뮬레이터에서 획득하는 상황을 보였다.



출처: Fernandez, Crespo and Mahtani et al. (2015). Learning ROS for Robotics Programming. 2nd Ed. Packt Publishing. p. 296.
[그림 4-2] 모바일 로봇에 센서를 추가하여 센서 데이터를 Gazebo에서 시뮬레이션하는 장면

(4) Gazebo에 맵 불러오기

gazebo_worlds 패키지 안에 다양한 맵 데이터들이 있으므로 이를 설치한다. Gazebo는 기본 환경에서 Menu - Save as 기능을 이용하여 테이블과 의자 등의 가구들의 모델을 맵에 추가함으로써 로봇 콘텐츠에 필요한 인테리어와 소품들을 손쉽게 가상 환경에서 구현할 수 있다. 특히 http://gazebosim.org/models 사이트에 접속함으로써 액추에이터, 센서, 로봇에 대한 풍부한 모델을 얻을 수 있다.

(5) 플러그인 기능으로 로봇 움직이기

Gazebo에서 플러그인 기능을 사용하면 앞에서 얻은 다양한 모션 프리미티브들을 기반으로 한 연속 동작을 로봇으로 시뮬레이션할 수 있다.

(6) 로봇 콘텐츠 메타모델 불러오기

HRI지능 소프트웨어 개발하기(LM1903080308_14v1.4) 학습모듈의 soarwrapper 패키지(ROS와 Soar를 연결)와 Gazebo를 연결한다. 그리고 학습모듈 3-1의 수행에서 보았듯이 Soar의 시맨틱 메모리에 씬별 메타모델 정보를 기록하고 soarwrapper 패키지를 실행함으로써 메타모델을 불러온다. 특히, soarwrapper 패키지를 사용하면 ROS 기반으로 구현된 다양한 HRI(human-robot interation, 인간로봇상호작용) 패키지를 사용할 수 있다.

(7) 멀티미디어 파일 불러오기

Gazebo는 자체적으로 멀티미디어 파일을 재생하는 기능이 없으므로 HRI지능 소프

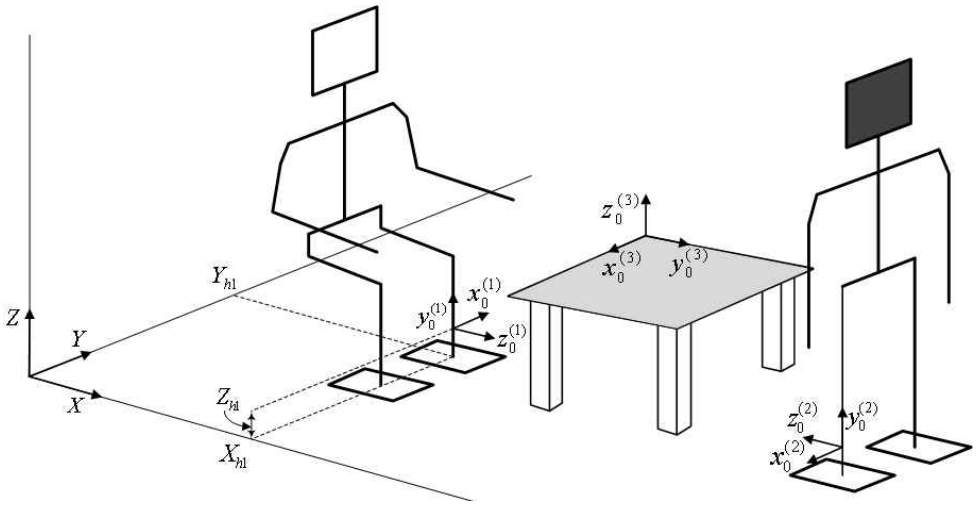
트웨어 개발하기(LM1903080308_14v1.4) 학습모듈에서 소개한 sound_play 패키지를 다운로드하여 ROS에 연결하여 사용하면 된다.

2. MATLAB을 이용한 로봇 콘텐츠 실행

(1) 전역좌표계 변환

3-1절에서는 [그림 3-3], [그림 3-4], [그림 3-5], [그림 3-7]에 보인 것처럼 하나의 로봇에 대해 모션을 만들고 MATLAB으로 시뮬레이션하는 방법을 설명했다.

MATLAB을 로봇 콘텐츠의 가상 플랫폼으로 만들기 위해서는 각 컷에 등장하는 모든 로봇과 가구, 물체 등을 한 화면에 넣을 수 있어야 한다. 이때 각 로봇과 가구, 물체들은 각자의 지역좌표계를 가지고 있으므로 현재 공간에서 표준이 되는 전역좌표계를 설정하고 각 지역좌표계와의 변환 관계를 설정해 주어야 한다.



[그림 4-3] 두 대의 로봇(왼쪽: 최과장 로봇, 오른쪽: 배대리 로봇)과 그 사이에 테이블이 있는 사무실 공간의 지역좌표계들과 전역좌표계

[그림 4-3]은 Scene 1에서 최과장 로봇(앉아 있는 로봇)과 배대리 로봇(서있는 로봇)이 함께 있는 상황을 그림으로 표현한 것이다. [그림 3-2]에 보인 휴머노이드 로봇 모델의 지역기준좌표계(local reference coordinate frame)는 지지하는 다리(양측 지지의 경우 왼쪽 다리)의 발목 관절에 위치하며 왼쪽을 향하는 방향이 x_0축, 위쪽을 향하는 방향이 y_0축, 앞쪽을 향하는 방향이 z_0축으로 설정되어 있다(김종욱, 2015a). 한편 [그림 4-3]의 공간의 전역좌표계인 XYZ좌표계의 원점은 맨 왼쪽 아래의 구석으로 설정되어 있다.

최과장 로봇의 지역기준좌표계의 원점이 전역좌표계상에서 (X_{h1}, Y_{h1}, Z_{h1})에 있다면 전역좌표계와 최과장 로봇의 지역좌표계 간의 동차변환행렬(homogeneous transformation matrix) 표현식은 다음과 같다(김종욱, 2015a).

$$W_{h1}^{0}=\begin{bmatrix} \bar{x}_0^{(1)}\cdot\bar{X} & \bar{y}_0^{(1)}\cdot\bar{X} & \bar{z}_0^{(1)}\cdot\bar{X} & X_{h1} \\ \bar{x}_0^{(1)}\cdot\bar{Y} & \bar{y}_0^{(1)}\cdot\bar{Y} & \bar{z}_0^{(1)}\cdot\bar{Y} & Y_{h1} \\ \bar{x}_0^{(1)}\cdot\bar{Z} & \bar{y}_0^{(1)}\cdot\bar{Z} & \bar{z}_0^{(1)}\cdot\bar{Z} & Z_{h1} \\ 0 & 0 & 0 & 1 \end{bmatrix}=\begin{bmatrix} 0 & 0 & 1 & X_{h1} \\ 1 & 0 & 0 & Y_{h1} \\ 0 & 1 & 0 & Z_{h1} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{식 (9)}$$

여기서 $\bar{x}_0^{(1)}$, $\bar{y}_0^{(1)}$, $\bar{z}_0^{(1)}$은 최과장 로봇의 지역기준좌표계의 세 축의 단위벡터를 나타내며, $\bar{X},\bar{Y},\bar{Z}$는 전역좌표계의 세 축의 단위벡터를 의미한다.

마찬가지로 배대리 로봇의 지역기준좌표계의 원점이 전역좌표계상에서 (X_{h2}, Y_{h2}, Z_{h2})에 있다면 전역좌표계와 배대리 로봇의 지역좌표계간의 동차변환행렬식은 다음과 같다.

$$W_{h2}^{0}=\begin{bmatrix} \bar{x}_0^{(2)}\cdot\bar{X} & \bar{y}_0^{(2)}\cdot\bar{X} & \bar{z}_0^{(2)}\cdot\bar{X} & X_{h2} \\ \bar{x}_0^{(2)}\cdot\bar{Y} & \bar{y}_0^{(2)}\cdot\bar{Y} & \bar{z}_0^{(2)}\cdot\bar{Y} & Y_{h2} \\ \bar{x}_0^{(2)}\cdot\bar{Z} & \bar{y}_0^{(2)}\cdot\bar{Z} & \bar{z}_0^{(2)}\cdot\bar{Z} & Z_{h2} \\ 0 & 0 & 0 & 1 \end{bmatrix}=\begin{bmatrix} 0 & 0 & -1 & X_{h2} \\ -1 & 0 & 0 & Y_{h2} \\ 0 & 1 & 0 & Z_{h2} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{식 (11)}$$

여기서 $\bar{x}_0^{(2)}$, $\bar{y}_0^{(2)}$, $\bar{z}_0^{(2)}$는 배대리 로봇의 지역기준좌표계의 세 축의 단위벡터를 나타낸다. [그림 4-3]에 있는 책상의 지역기준좌표계인 $x_0^{(3)}y_0^{(3)}z_0^{(3)}$좌표계에 대해서도 동일한 관계식을 적용하면 된다.

(2) 멀티미디어 파일 재생

MATLAB에서는 다음의 함수로 wav 파일이나 mp3 파일을 불러와서 음성, 배경 음악, 효과음을 재생할 수 있다(https://kr.mathworks.com/help/matlab/ref/audioread.html).

```
[y,Fs] = audioread('handel.wav');
% [y,Fs] = audioread('handel.mp3');
sound(y,Fs);
```

로봇 콘텐츠 메타모델 파일에서 해당 시점에 이 코드를 배치하면 모션 실행과 멀티미디어 파일의 동기화가 이루어진다.

수행 내용 / 로봇 콘텐츠 저작 도구 개발하기

재료 · 자료

- 로봇 하드웨어 사양서
- 로봇 운영체제 사양서
- 로봇 소프트웨어 플랫폼의 설명서
- 로봇 소프트웨어 개발 매뉴얼
- 오픈소스 지적재산권 지침서

기기(장비 · 공구)

- 컴퓨터, 프린터, 인터넷
- 로봇 소프트웨어 플랫폼의 소프트웨어 개발 환경툴(SDK)
- 프로그램 개발용 소프트웨어
- 컴파일러
- 로봇 콘텐츠를 실행할 로봇
- 문서 작성용 소프트웨어

안전 · 유의 사항

- 오픈 소스의 지적재산권 규정을 준수해야 한다.

수행 순서

1 설계 된 로봇 콘텐츠를 가상플랫폼에서 실행할 수 있는 저작 도구를 설계할 수 있다.

1. Gazebo를 이용한 저작 도구 설계

(1) OP2의 Gazebo 시뮬레이터 패키지 설치

로봇 콘텐츠를 구현할 로봇인 OP2를 Gazebo로 시뮬레이션 하기 위해 Github의 전용 사이트(https://github.com/ROBOTIS-OP)에 접속해서 다운로드하거나, 휴머노이드파크 웹사이트의 '로봇기술 자료실'에서 'Darwin_OP(ROS)' 폴더를 다운로드한다. 이 폴더에는 darwinOP.urdf 파일이 있어서 OP의 링크와 관절 정보들이 담겨 있다.

다음의 단계를 밟아서 Gazebo 시뮬레이션을 실행한다.

패키지를 다운로드한 디렉토리로 이동한다.

```
cd home/user/Darwin_OP/Darwin_Optimal_Preview_Control_Walking
```

아래 명령어로 프로젝트를 컴파일한다.

```
cd build
cmake ../
make
cd ..
```

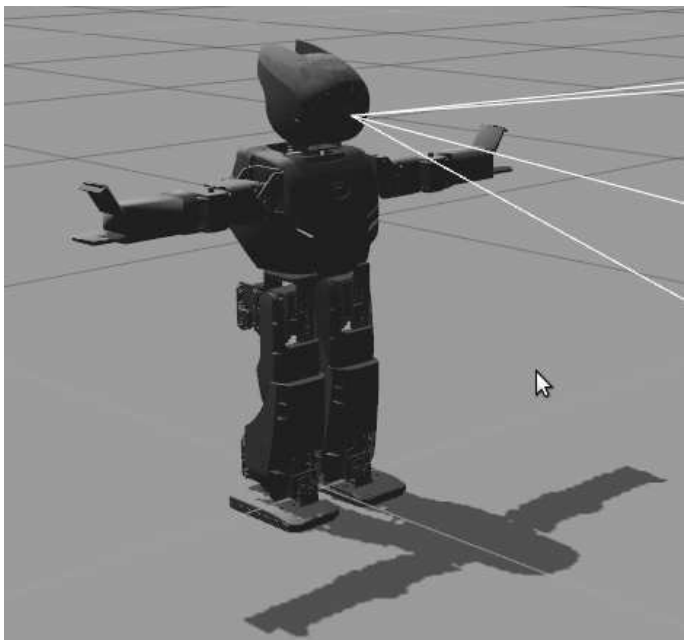
Gazebo 플러그인을 export한다.

```
export GAZEBO_PLUGIN_PATH=${GAZEBO_PLUGIN_PATH}:~/Darwin_OP/Darwin_Optimal
Preview_Control_Walking/build
```

프로젝트를 실행한다.

```
-u Darwin_OPC_walking
```

[그림 4-4]는 Gazebo에서 OP 패키지를 실행한 모습이다.

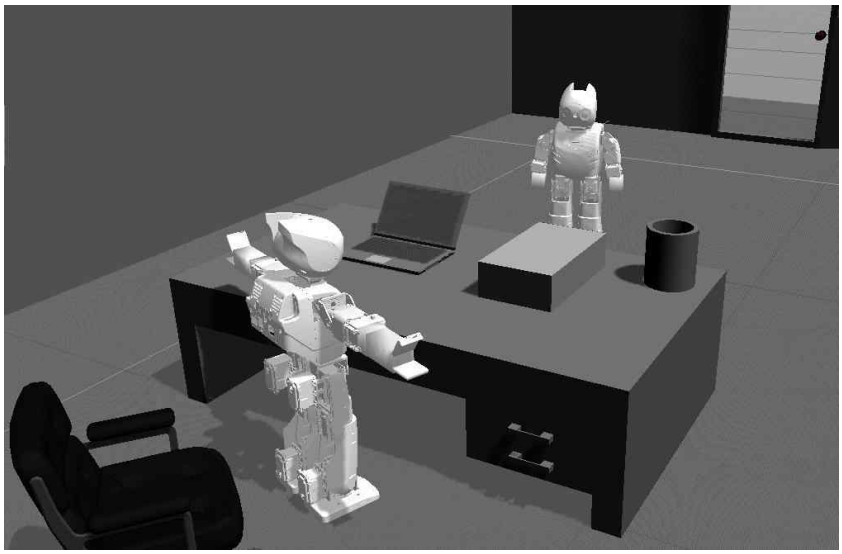


출처: https://github.com/HumaRobotics/darwin_gazebo. 2016. 10. 09. 스크린샷.
[그림 4-4] OP를 Gazebo에서 시뮬레이션 한 화면

(2) Gazebo에서 로봇 드라마 환경 구현하기

Gazebo로 로봇 드라마가 구현되는 환경을 gazebo_worlds 패키지의 모델들을 이용해서 실감 있게 구현한다. [그림 4-5]는 <표 2-3> 메타모델의 첫 번째 씬의 배경이 되는 사

무실 공간을 Gazebo로 구현한 화면이다. 화면에서 등을 보이고 서 있는 로봇이 최과장 로봇이며, 책상 너머에 서 있는 로봇이 배대리 로봇이다.



출처: Gazebo. 스크린샷.
[그림 4-5] OP를 Gazebo에서 시뮬레이션한 화면

(3) 로봇 모션을 위한 플러그인 만들기

'Darwin_OP(ROS)' 폴더의 'Darwin_Optimal_Preview_Control_Walking' 디렉토리에 가면 darwin_plugin_motion.cc라는 파일이 있다. 이 파일을 열면 다음 페이지와 같은 플러그인 소스코드를 볼 수 있다.

코드에서 xp는 각 모션 프리미티브의 시간을 나타내고, R_Sho_Pitch_Loc에서 Head_Tilt_Loc까지의 20개의 배열은 각 관절의 모션 프리미티브별 회전각을 모터의 위치값으로 표현한 것이다. 이 값들은 액션에디터를 사용하여 각 자세를 만들면서 얻을 수 있다.

첫 번째 씬의 첫 번째 컷에 대한 네 가지 모션 프리미티브([그림 3-7])에 대한 각 관절 모터의 절대관절변수를 변환 공식을 이용해서 모터의 위치값으로 변환하여 R_Sho_Pitch_Loc에서 Head_Tilt_Loc까지의 값에 부여하면 된다.

Gazebo로 로봇 모션을 시뮬레이션할 때 주의할 점은 각 관절 모터의 PID 제어 이득들을 적절한 값으로 정해 줘야 한다는 것이다. 코드에서 UpdatePID1~UpdatePID9는 플러그인 코드로 구현하려는 9가지 동작마다 적절한 PID 제어 이득값을 조절하기 위해 작성된 함수이다. 이 이득값들은 실제 로봇에 적용하는 PID 제어 이득과는 별개임을 유의하자.

```
#include “gazebo/common/common.hh“
#include “gazebo/physics/physics.hh“
#include “gazebo/common/Events.hh“
#include “gazebo/common/PID.hh“
#include <iostream>
#include <fstream>

float xp [91] = {0,1,2,3,4,5,6,7,8,9,10, ...,  85,86,87,88,89,90};
float R_Sho_Pitch_Loc [] = {2047,2047,2047, ... ,1160,1160,1160}; // ID 1
float L_Sho_Pitch_Loc [] = {2047,2047,2047, ... ,2932,2932,2932}; // ID 2
float R_Sho_Roll_Loc [] = {1547,1547,1547, ... ,2262,1845,2262}; // ID 3
float L_Sho_Roll_Loc [] = {2540,2540,2540,... ,1669,2248,1669}; // ID 4
float R_Elbow_Loc [] = {970,970,970, ... ,2446,2446,2446}; // ID 5
float L_Elbow_Loc [] = {3122,3122,3122, ... ,1887,1887,1887}; // ID 6
float R_Hip_Yaw_Loc [] = {2047,2047,2047, ... , 2048,2048,2048}; // ID 7
float L_Hip_Yaw_Loc [] = {2047,2047,2047, ... , 2048,2048,2048}; // ID 8
float R_Hip_Roll_Loc [] = {2047,2047,2047, ... , 2052,2052,2052}; // ID 9
                          ⋮
float R_Ankle_Roll_Loc [] = {2047,2047,2047, ... ,2057,2057,2057}; // ID 17
float L_Ankle_Roll_Loc [] = {2047,2047,2047, ..., 2039,2039,2039}; // ID 18
float Head_Pan_Loc [] = {2047,2047,2047, ..., 2048,2048,2048}; // ID 19
float Head_Tilt_Loc [] = {2170,2170,2170, ... ,2100,2300,2100}; // ID 20

namespace gazebo
{
  class PID1Joints : public ModelPlugin
  {
    public: void Load(physics::ModelPtr _model, sdf::ElementPtr /*_sdf*/)
    {
                          ⋮
    }
    void UpdatePID1(){   }
                          ⋮
    void UpdatePID9(){   }
                          ⋮
}
```

(4) 로봇 콘텐츠 메타모델 구현

ROS와 Soar를 연결한 soarwrapper 패키지와 sound_play 패키지를 다운로드해서 Gazebo와 연결하고, 필요한 음성·음향 효과 멀티미디어 파일을 Darwin_OP(ROS) 폴더에 저장 후 불러온다.

각 컷의 모션과 멀티미디어 파일 재생을 동기화 하는 것은 Soar의 메타모델 시맨틱 메모리 값을 조절하면서 구현한다.

2. MATLAB을 이용한 저작 도구 설계

(1) MATLAB 패키지

휴머노이드파크 웹사이트의 '로봇기술 자료실'에 가서 'robot_drama(matlab)' 패키지를 다운로드한다.

robot_drama(matlab) 패키지에는 로봇 드라마와 관련하여 다음의 파일들이 있다. 그 이외의 파일들은 김종욱(2015a)의 책을 참고하기 바란다.

- sim_roborama_scene1.m: 메인코드로서 Scene1의 일부 상황을 구현
- sim_pose.m: 휴머노이드 로봇의 모션 프리미티브 생성을 위한 자세 시뮬레이션
- CalRelativeJointAngles.m: 각 경유점별 모션 프리미티브에 해당되는 시상면·관상면·횡평면 절대관절변수 벡터를 인자로 받아서 혼합다항식을 이용하여 경유점 사이 임의 시간에서의 절대관절변수 벡터 계산
- GenMotionWithDH2: 주어진 상대관절변수 벡터를 이용하여 휴머노이드 로봇의 순간 자세를 3차원에서 그려줌
- WalkManager2: 로봇이 보행할 경우 초기보행, 주기보행, 최종보행의 세 가지 상태 중의 하나를 선택해서 전신의 각 관절에 적절한 시작각과 마침각, 최적의 혼합다항함수 변수들을 할당
- GenRoboticWalkingWithCost2: 로봇이 보행할 경우 전신의 샘플 시간별 절대관절변수 값을 계산하고 이를 상대관절변수로 변환하여 GenMotionWithDH2 함수에 넘겨줌으로써 부드럽게 걸어가는 휴머노이드 로봇 모션 생성

(2) 메타모델 및 실행 코드

다음 페이지의 코드는 robot_drama(matlab) 패키지의 메인코드인 sim_roborama_scene1.m 파일의 메타모델 부분을 나타낸 것이다.

- times_M1_R1_S1은 Scene1(S1)의 첫 번째 로봇(R1, 최과장 로봇)의 첫 번째 모션(M1)을 위한 시간 경유점 벡터를 나타낸다.
- mths_M1_R1_S1, mphis_M1_R1_S1, mpsis_M1_R1_S1 행렬은 첫 번째 로봇의 M1 모션을 위한 모션 프리미티브의 각 절대관절변수의 각도값 행렬들이다.
- times_M1_R2_S1은 Scene1(S1)의 두 번째 로봇(R2, 배대리 로봇)의 첫 번째 모션(M1)을 위한 시간 경유점 벡터를 나타낸다.
- mths_M1_R2_S1, mphis_M1_R2_S1, mpsis_M1_R2_S1 행렬은 두 번째 로봇의 M1 모션을 위한 모션 프리미티브의 각 절대관절변수의 각도값 행렬들이다.
- times_W1_R2_S1은 두 번째 로봇의 첫 번째 보행모션(W1)을 위한 시간 경유점 벡터를 나타낸다. 보폭은 OP의 경우 5cm로 정해져 있으며, 시간 간격을 좁혀서 설정하면 빠른 걸음이 되고, 넓혀서 설정하면 느린 걸음이 된다. walk_stat 벡터는

각 스텝이 최초보행인지, 주기보행인지, 최종보행인지에 대한 정보를 담는데, 보통 초기값은 1, 마지막값은 3으로 설정해야 직립 상태에서 걷기 시작해서 직립 상태로 멈춘다.

- times_M2_R2_S1은 Scene1의 두 번째 로봇의 보행 후 두 번째 모션(M2)을 위한 시간 경유점 벡터를 나타낸다.
- mths_M2_R2_S1, mphis_M2_R2_S1, mpsis_M2_R2_S1 행렬은 두 번째 로봇의 M2 모션을 위한 모션 프리미티브의 각 절대관절변수의 각도값 행렬들이다.

```
% 최과장 로봇이 앉아서 화를 내다가 전화를 검
times_M1_R1_S1 = [1 3 4 5 6 7 10 19 20];
mths_M1_R1_S1 = [mth_stand_up;    mth_sit_down;   mth_hit_chest_right_arm;
    mth_hit_chest_left_arm; mth_hit_chest_right_arm; mth_hit_chest_left_arm;
    mth_phone_call_left_hand; mth_phone_call_left_hand; mth_sit_down];
mphis_M1_R1_S1 = [mphi_stand_up; mphi_sit_down; mphi_hit_chest_right_arm;
    mphi_hit_chest_left_arm; mphi_hit_chest_right_arm; mphi_hit_chest_left_arm;
    mphi_phone_call_left_hand; mphi_phone_call_left_hand; mphi_sit_down];
mpsis_M1_R1_S1 = [mpsi_stand_up; mpsi_sit_down; mpsi_hit_chest_right_arm;
    mpsi_hit_chest_left_arm; mpsi_hit_chest_right_arm; mpsi_hit_chest_left_arm;
    mpsi_phone_call_left_hand; mpsi_phone_call_left_hand; mpsi_sit_down];

% 배대리 로봇이 서 있다가 전화를 받고 걸어감
times_M1_R2_S1 = [0 12 15 18 19];
mths_M1_R2_S1 = [mth_stand_up;    mth_stand_up;   mth_phone_call_left_hand_stand;
    mth_phone_call_left_hand_stand; mth_stand_up];
mphis_M1_R2_S1 = [mphi_stand_up; mphi_stand_up; mphi_phone_call_left_hand_stand;
    mphi_phone_call_left_hand_stand; mphi_stand_up];
mpsis_M1_R2_S1 = [mpsi_stand_up; mpsi_stand_up; mpsi_phone_call_left_hand_stand;
    mpsi_phone_call_left_hand_stand; mpsi_stand_up];

times_W1_R2_S1 = [19 20 21 22 23];
walk_stat = [1 2 2 3];        % 보행상태, 1:최초보행, 2:주기보행, 3:최종보행

% 배대리 로봇이 최과장 로봇에게 인사를 함
times_M2_R2_S1 = [23 25 27];
mths_M2_R2_S1 = [mth_stand_up;    mth_bow;   mth_stand_up];
mphis_M2_R2_S1 = [mphi_stand_up;   mphi_bow;   mphi_stand_up];
mpsis_M2_R2_S1 = [mpsi_stand_up;   mpsi_bow;   mpsi_stand_up];

% 멀티미디어 재생
times_media_play = [12 15 17];
names_file_play = ['bell_sound_1.mp3'; 'come here.mp3'; 'Yes sir.mp3'];
```

아래의 코드는 앞에서 정한 Scene1의 메타코드를 기반으로 매 샘플 시간마다 로봇1과 로봇2의 자세를 동적으로 계산하여 로봇 콘텐츠를 실행하는 부분이다.

```
Ti = 0;  Tf = 27;  ns = floor((Tf-Ti)/st)+1;
istep = 1;  % 보행 스텝 인덱스
iplay = 1;  % 파일재생 인덱스

for i=1:ns
    t = (i-1)*st +Ti
    if iplay <= length(times_media_play)
        if t == times_media_play(iplay)
            if iplay==1, file_name = 'bell_sound_1.mp3';
            elseif iplay==2, file_name = 'come here.mp3';
            elseif iplay==3, file_name =  'Yes sir.mp3';
            end
            [y,Fs] = audioread(file_name);
            sound(y,Fs);
            iplay = iplay+1;
        end
    end
    %%% 최고장의 모션 %%%
    Ang = CalRelativeJointAngles(sf1, t, times_M1_R1_S1, mths_M1_R1_S1, mphis_M1_R1_S1,
       mpsis_M1_R1_S1, n_pos);
    th = Ang(1:11,1);     th_1d = Ang(1:11,2);    th_2d = Ang(1:11,3);
    phi = Ang(12:17,1);  phi_1d = Ang(12:17,2);  phi_2d = Ang(12:17,3);
    psi = Ang(18:20,1);  psi_1d = Ang(18:20,2);  psi_2d = Ang(18:20,3);
    res1 = GenMotionsWithDH2(sf1, W1, th, th_1d, th_2d, phi, phi_1d, phi_2d, psi, psi_1d, psi_2d, 1);
    %%% 배대리의 모션 %%%
    if (t >= times_M1_R2_S1(1)) && (t < times_M1_R2_S1(length(times_M1_R2_S1)))
        Ang = CalRelativeJointAngles(sf2,  t, times_M1_R2_S1, mths_M1_R2_S1, mphis_M1_R2_S1,
            mpsis_M1_R2_S1);
        th = Ang(1:11,1);     th_1d = Ang(1:11,2);    th_2d = Ang(1:11,3);
        phi = Ang(12:17,1);  phi_1d = Ang(12:17,2);  phi_2d = Ang(12:17,3);
        psi = Ang(18:20,1);  psi_1d = Ang(18:20,2);  psi_2d = Ang(18:20,3);
        res2 = GenMotionsWithDH2(sf2, W2, th, th_1d, th_2d, phi, phi_1d, phi_2d, psi, psi_1d,
            psi_2d, 1);
    elseif t <= times_W1_R2_S1(length(times_W1_R2_S1))
        swf = WalkManager2(t, times_W1_R2_S1(istep), times_W1_R2_S1(istep+1), sf2, W2,
            walk_cond2, fig_no, 1);
```

```
        if t == times_W1_R2_S1(istep+1)
            % 배대리 로봇의 지역좌표계 동차변환행렬
            W2 = [0 0 -1 swf(1); -1 0 0 swf(2); 0 1 0 l0; 0 0 0 1];
            sf2 = SwitchSupportLeg(sf2);
            istep = istep+1;
            if istep <= length(walk_stat), walk_cond2(6) = walk_stat(istep);  end
        end
    elseif t <= times_M2_R2_S1(length(times_M2_R2_S1))
        % n_pos = length(times_M2_R2_S1);
        Ang = CalRelativeJointAngles(sf2,  t, times_M2_R2_S1, mths_M2_R2_S1, mphis_M2_R2_S1,
            mpsis_M2_R2_S1);
        th = Ang(1:11,1);     th_1d = Ang(1:11,2);    th_2d = Ang(1:11,3);
        phi = Ang(12:17,1);  phi_1d = Ang(12:17,2);  phi_2d = Ang(12:17,3);
        psi = Ang(18:20,1);  psi_1d = Ang(18:20,2);  psi_2d = Ang(18:20,3);
        res2 = GenMotionsWithDH2(sf2, W2, th, th_1d, th_2d, phi, phi_1d, phi_2d, psi, psi_1d,
            psi_2d, 1);
    end
                ⋮
end
```

(3) 로봇 콘텐츠 실행

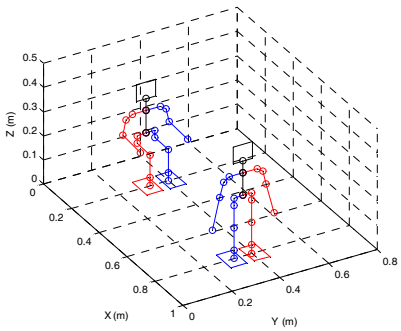
다음 페이지의 [그림 4-6]은 robot_drama(matlab) 패키지의 sim_roborama_scene1.m 파일을 실행했을 때 각 시간별로 시뮬레이션을 수행하는 모습이다. 중간 중간에 mp3 파일로 만든 음성 파일이 적절히 동기화되어 재생되는 것을 알 수 있다.

- t=3초: 최과장 로봇은 사무실에 앉아 있고, 배대리 로봇은 바깥에 서 있다.
- t=5초: 최과장 로봇이 화가 나서 가슴을 치고 있다.
- t=10초: 최과장 로봇이 배대리 로봇에게 전화를 한다.
- t=14초: 배대리 로봇이 최과장 로봇과 통화를 한다.
- t=19초~23초: 배대리 로봇이 전화를 끊고 최과장에게 걸어간다.
- t=24초: 배대리 로봇이 최과장에게 인사한다.

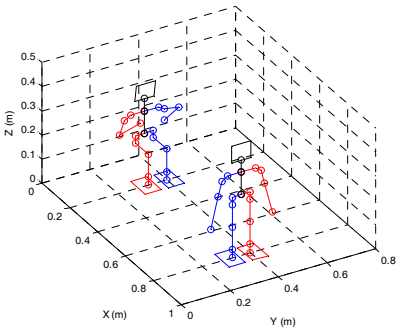
(4) 로봇 콘텐츠 편집

로봇 콘텐츠를 실행하면서 로봇의 동작과 멀티미디어가 동기화되지 않을 경우 앞의 sim_roborama_scene1.m 코드에서 다음 페이지의 시간 경유점 들의 간격을 적절히 조절하면 된다.

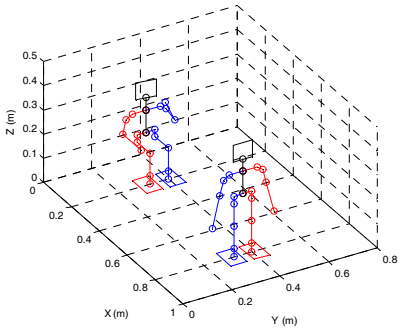
지금까지 실시한 로봇 드라마 시뮬레이션은 '아버지의 이름으로' 로봇 드라마의 Scene1의 일부분이며, 나머지 부분들을 실습으로 구현하기 바란다.



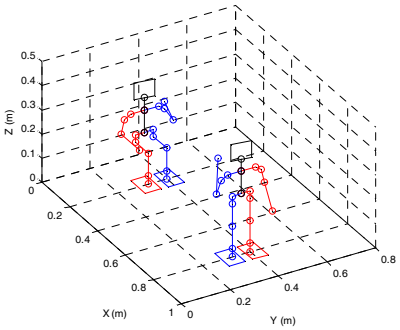
t=3초



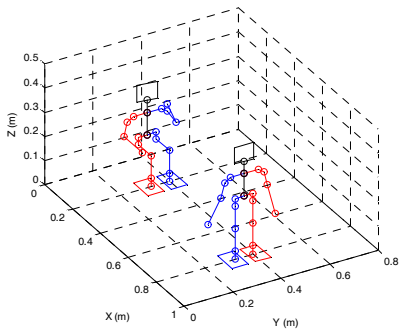
t=5초



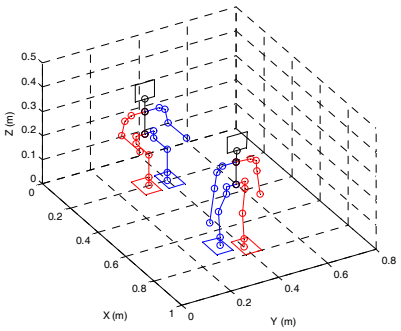
t=10초



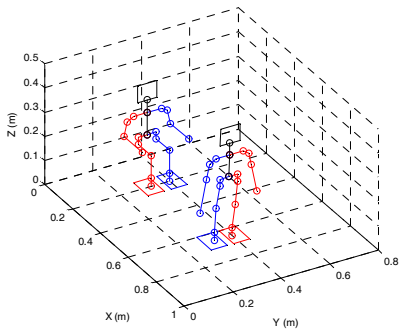
t=14초



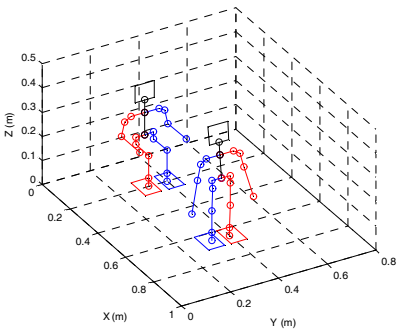
t=19초



t=21초



t=23초



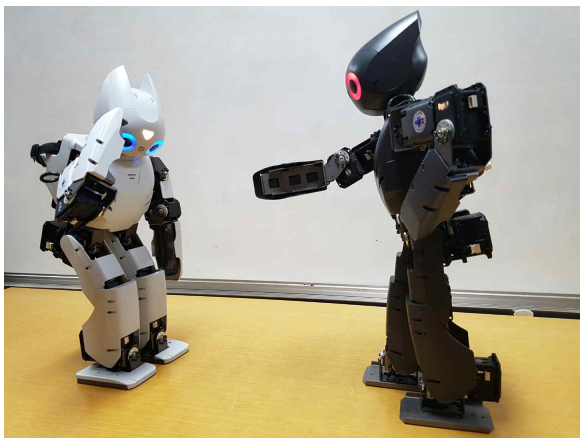
t=24초

[그림 4-6] Scene1의 일부 모션을 MATLAB으로 시뮬레이션할 때의 시간별 화면

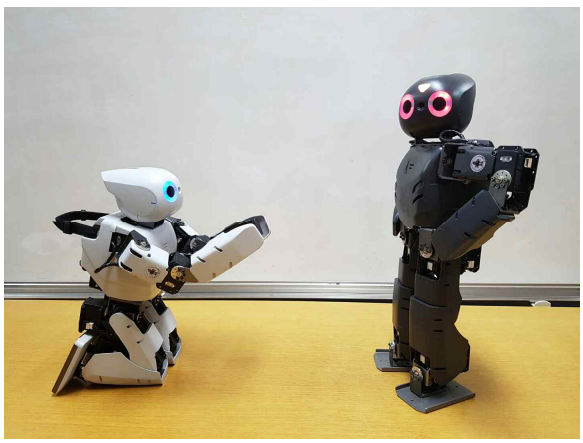
```
                    ⋮
times_R1_S1 = [1 3 4 5 6 7 10 19 20];
                    ⋮
times_M1_R2_S1 = [0 12 15 18 19];
                    ⋮
times_W1_R2_S1 = [19 20 21 22 23];
                    ⋮
times_M2_R2_S1 = [23 25 27];
                    ⋮
times_media_play = [12 15 17];
                    ⋮
```

(5) 로봇 콘텐츠 실행 및 피드백

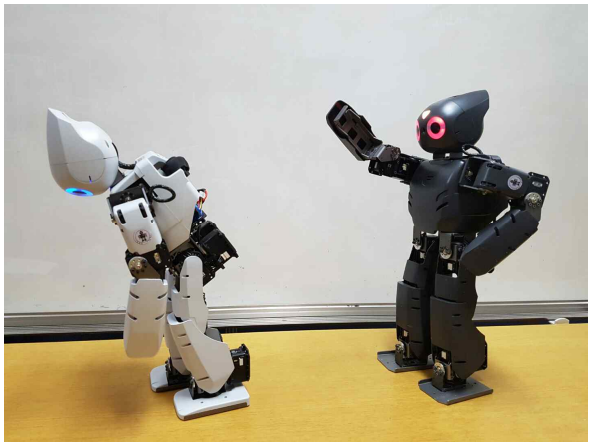
제작한 로봇 콘텐츠를 관객 앞에서 시연 후 관객의 반응을 피드백 받는다. [그림 4-7]~[그림 4-9]는 Scene 1의 주요 장면을 나타낸다. 이처럼 휴머노이드 로봇은 자세와 모션을 정교하게 구현하면 큰 흥미와 감동을 불러일으킬 수 있다.



[그림 4-7] 최과장 로봇이 배대리 로봇을 야단치는 장면



[그림 4-8] 배대리 로봇이 최과장 로봇에게 용서를 구하는 장면



[그림 4-9] 배대리 로봇이 쓸쓸히 돌아서는 장면