Real_Avatar 최종보고서

IT응용시스템공학과
1694056 김정호

1. 프로젝트 개발 목표 및 내용

가. 프로젝트 개발 요약

1) 목표 : 소셜 로봇의 안전하고 자연스러운 동작을 생성하고 목적에 맞도록 수정하는 솔루션 개발.

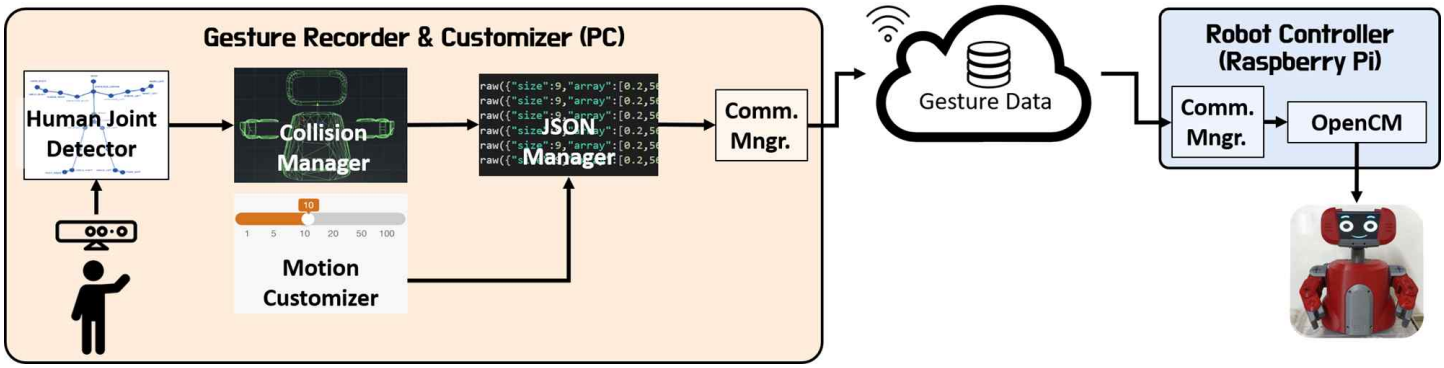
2) 솔루션 구성

- 키넥트(Kinect) : 모션 캡쳐를 위한 카메라
- 충돌체크 : 하드웨어의 충돌을 방지하기 위한 절차.
- 모션 기록 및 편집기 : 모션을 기록하고 기록된 모션의 속도 또는 구동 범위를 수정.
- 실물로봇 제어 : PC가 실물로봇과 통신하여 로봇 제어.

3) 솔루션 진행 절차

- 로봇의 동작을 생성할 사람의 모션 캡쳐.
- 캡쳐된 모션의 데이터로 PC(Unity)내 3D로봇모델을 구동.
- 안전한 실물로봇 구동을 위해 PC(Unity)에서의 충돌 검사.
- 충돌 검사 후 얻은 안전한 데이터로 실물 로봇 제어.
- 수정이 필요한 경우 속도 또는 구동 범위를 수정

4) 구조도



2. 프로젝트 진행 일정

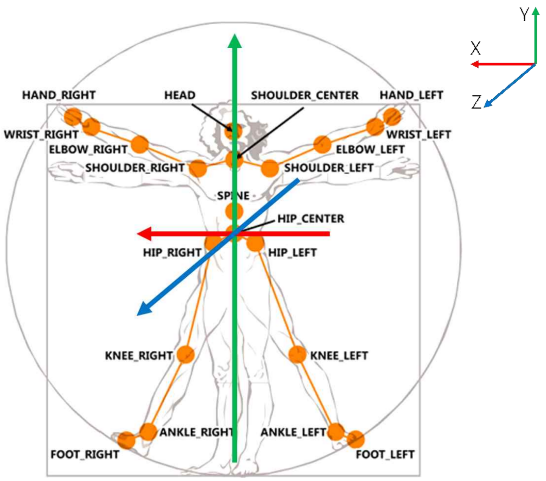
가. 진행 일정

Work	월	2월				3월				4월				5월				
	주차	-3	-2	-1	0	1	2	3	4	5	6	7	8	9	10	11	12	13
프로젝트 설계																		
키넥트 설치 및 분석																		
유니티 내부 3D로봇모델 구동																		
유니티 내부에서 충돌 검증 절차구현																		
충돌 검증 된 데이터를 기반으로 실물로봇 구동																		
모션 저장 및 편집 기능 개발																		
전체 통합 테스트																		

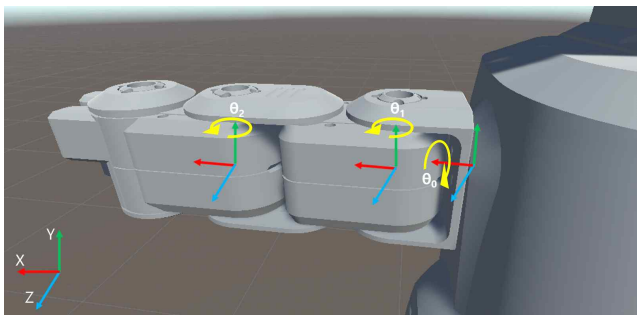
3. 상세 기능 설명

상세 기능 설명

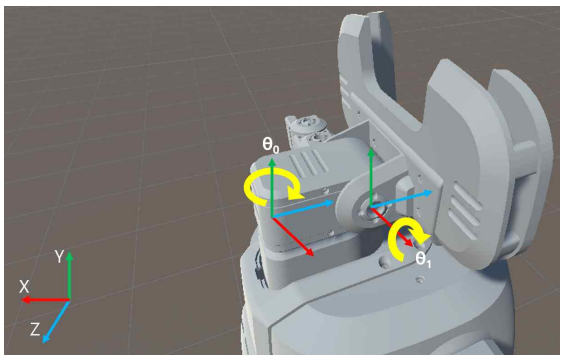
1. 조인트의 좌표계 얻는 방법.
   a. 오른쪽의 그림과 같이 기본자세는 차려 상태에서 팔을 들어 T 자 모양을 한 자세이다.
   b, 손, 발 그리고 머리가 이루는 평면은 XY평면 위에 있다.
   c. 항상 머리 위 방향을 Y축으로 잡는다.
   d. 임시 X축을 구한다. (현재 X축은 Y축과 수직인 보장이 없음)
      1) Y축에 포함된 조인트들의 X축 : 왼쪽 어깨에서 오른쪽 어깨로의 방향 또는 왼쪽 골반에서 오른쪽 골반으로의 방향.
      2) 나머지 조인트들의 X축 : 조인트가 속해있는 팔 또는 다리에서 끝으로 가는 방향 중 가장 가까운 조인트로의 방향.
   e. Y축과 임시 X축을 외적 하여 Z축을 구한다.
   f. Y축과 Z축을 외적 하여 X축을 구한다.



2. 3축 팔 3D로봇모델 구동하기
   a. 어깨 2축, 팔꿈치 1축.
   b. 조인트 회전시키기. (1주차 그림 참고)
      1) θ_0 회전 : Hip_Center의 Y축과 Shoulder_Left에서 Elbow_Left 방향으로의 벡터를 내적 하여 얻은 값만큼 회전.
      2) θ_1 회전 : Hip_Center의 X축과 Shoulder_Left에서 Elbow_Left 방향으로의 벡터를 내적 하여 얻은 값만큼 회전.
      3) θ_2 회전 : Shoulder_Left에서 Elbow_Left 방향으로의 벡터와 Elbow_Left에서 Wrist_Left 방향으로의 벡터를 내적 하여 얻은 값만큼 회전.
      4) 각 조인트마다 위치와 회전해야할 방향이 다름.
         => 두 벡터의 내적 값에 각 조인트별로 오프셋과 회전 방향을 지정하여 해결.



3. 얼굴인식을 활용한 로봇 목 제어.
   a. 얼굴인식을 이용하면 Head의 좌표를 얻을 수 있다.
   b. θ_0(pan) 회전 : Head의 Y축 회전
   c. θ_1(tilt) 회전 : Head의 X축 회전



4. 충돌 검사 시뮬레이터 만들기
   a. 각 조인트의 각도 제한하기.
      1) 각도를 제한함으로써 충돌체의 수를 줄이고 비이상적인 모션을 제한.
      2) 조인트별 제한된 각도 범위
         - 왼쪽 어깨1 : -90 ~ 90
         - 왼쪽 어깨2 : -90 ~ 0
         - 왼쪽 팔꿈치 : -90 ~ 0
         - 오른쪽 어깨1 : -90 ~ 90
         - 오른쪽 어깨2 : 0 ~ 90

- 오른쪽 팔꿈치 : 0 ~ 90
- 목(좌우) : -90 ~ 90
- 목(상하) : -30 ~ 15

b. 충돌 검증 3D모델과 충돌 검증된 데이터로 안전하게 동작하는 3D모델 나누기.

1) 충돌 검증 3D모델, 검증 완료 3D모델 오브젝트 추가.

c. 충돌 검증 모델에서 목표 모델로 데이터 넘기기.

1) 데이터는 조인트의 정보를 가지고 있는 클래스의 내용임.

2) 충돌 검증 모델의 조인트는 CDJoint.cs에서, 목표 모델의 조인트는 Joint.cs에서 관리.

3) AngleMessenger.cs가 CDJoint에서 데이터를 가져오고 Joint에 데이터 전달.

d. 충돌 검증

1) 충돌 검증 모델의 실제 충돌이 일어나는 부분(머리, 팔, 몸통)에 충돌체를 추가.

2) 충돌체에서 충돌이 일어나면 충돌체가 속하는 부분의 데이터를 목표 모델로 전달하지 않음.

3) 비교사진

충돌 검증 모델에서 충돌이 일어나면 목표 모델은 충돌 직전의 자세를 취함.

5. 모션 데이터 저장하기

a. 실물 로봇을 구동하기 위해서는 시간 값 1개, 모터 각도 값 8개 까지 총 8개의 값이 필요하다.

b. 검증 완료 모델의 각도 값을 관리하는 Joint에 접근하여 a의 값을 배열에 저장함.

c. b에서 저장한 배열들을 지정된 프레임 단위로 또 다른 배열에 저장한다.

b. c에서 저장한 배열을 JSON형태로 변환하여 파일로 저장.

6. 모션 데이터 편집하기

a. 저장된 JSON 파일을 다시 저장하기 직전의 배열로 변환한다.

b. 속도 편집

1) 배열의 속도 값을 수정.

c. 구동 범위 편집

2) 배열의 모터 각도 값을 수정.

7. 실물 로봇 구동

a. PC에서 SSH 프로토콜을 이용하여 서버로 JSON파일의 값을 5프레임 단위로 보낸다.

b. 실물 로봇에 연결된 라즈베리파이가 wifi로 서버와 통신하여 PC에서 넘어온 값들을 받는다.

c. b에서 받은 값들을 openCM으로 시리얼 통신을 이용하여 전달하고 그 값들로 로봇을 구동한다.

8. 최종 UI 화면