

# Open Jig Ware

작성자 : 온진욱  
현 소속 : (주)DST 로봇 – 구 (주)동부로봇  
Mail : ojw5014@hanmail.net

# Open Jig Ware 소개

- 목적
  - 개발자
    - 단일 로봇의 제어를 손쉽게 할 수 있게 하는 것
  - 교육
    - 교사
      -
    - 학생
      -

# 장점과 문제점

- 장점
  - 모델링 파일이 있는 경우 하나의 모델링 파일 안에 모든 로봇의 세부적인 정보가 수식을 포함하여 전부 들어있다.
  - 로봇의 종류에 상관없이 모든 로봇에 제어가 가능(바퀴, 회전/직동 관절, 체인, 병렬제어 등)
- 문제점
  - 현재 개발되어 있는 모터 제어 함수가 한정되어 있다.
  - 3D 모델링이 컴퓨터의 환경을 타는 경우가 있다.

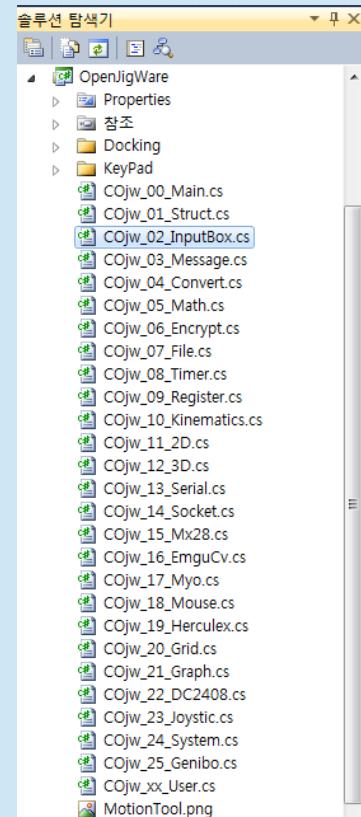
# 나아가고 싶은 방향

- 3D 모델링을 그리게 되면 이게 STL 파일로 저장되거나 아님 바로 Gcode로 slice를 해서 3D 프린팅과 연동이 되었으면...
- Myo, Leap Motion, Arduino 등의 디바이스연계가 되었으면...
- 리눅스(라즈베리파이), 안드로이드, Unity에 포팅이 되었으면...
- 수식 컴파일러가 프로그래밍 스타일도 가능했으면...
-

# OpenJigWare 란?

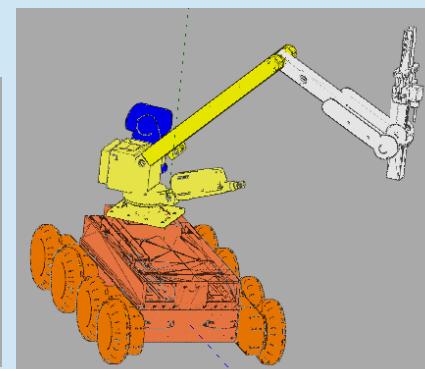
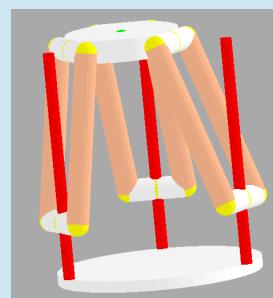
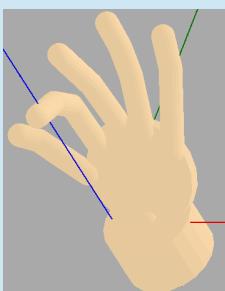
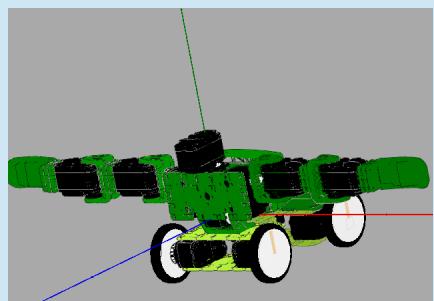
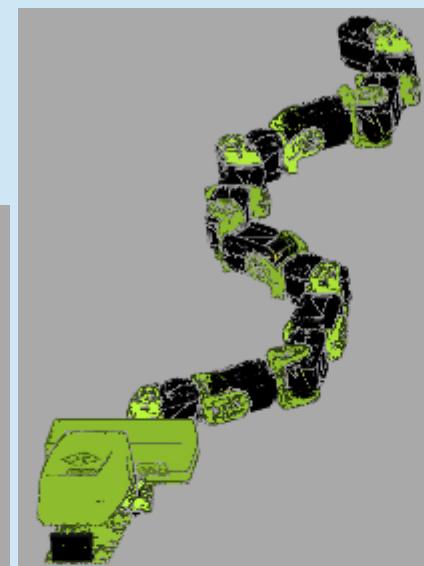
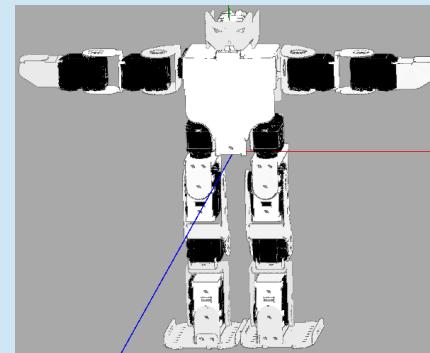
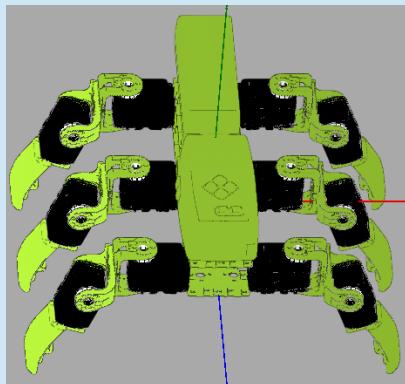
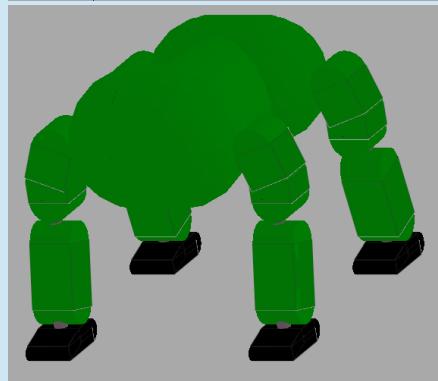
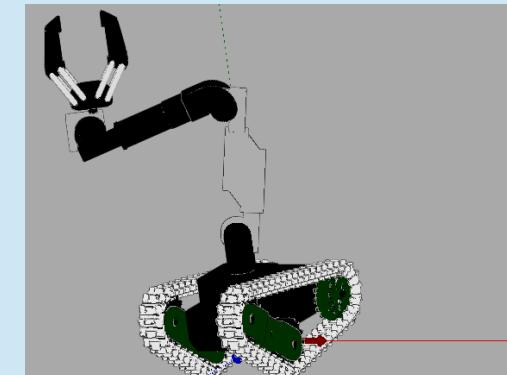
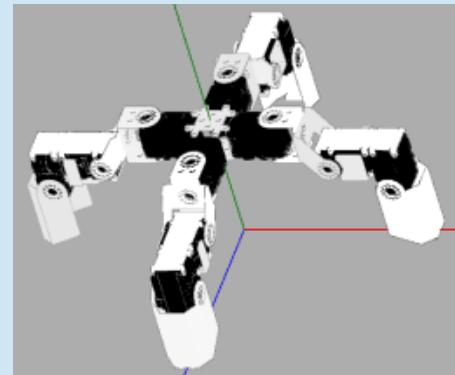
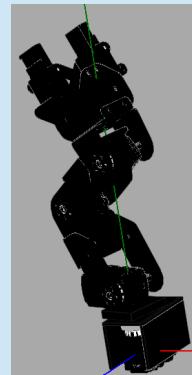
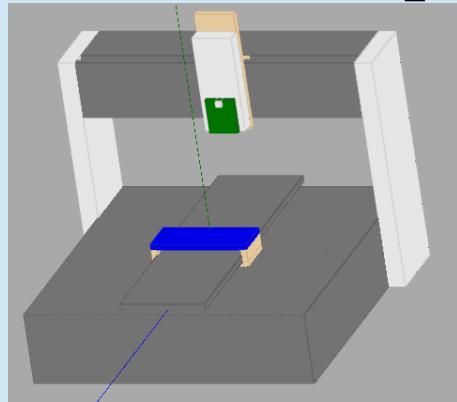
- 교육

- 선생님과 학생이 구분해서 사용 가능
  - 선생님
    - 3D로 로봇 만들기, 수식 입력, 모터 정의 및 상관관계입력
    - 다양한 로봇을 학생들에게 접하게 할 수 있고 로봇의 수식을 학생들에게 실제적으로 적용해서 보일 수 있다.
    - 클릭하는 로봇의 부품에 따라 회전, 직교로 동작할지 Forward/Inverse 등의 수식이 적용될지, 혹은 클릭이 되지 않게 적용할지를 선택할 수 있다.



# 어떤 로봇들이 가능한가?

< 실제 OpenJigWare에서 모델링한 로봇들 >



# 개발 동기

- 어느 날 던져진 화두
  - 모든 로봇에 적용될 수 있는 무언가가 없다.
    - 휴머노이드건 바퀴형 로봇이건 직교좌표로봇이건 모든 로봇에 적용될 수 있는 것이 있다면?
    - 로봇은 수식으로 이루어진 수식 덩어리. 그렇다면 그 수식만 바꿔주면 될거 아닌가?
    - Kinematics 를 교육하기란 너무 어렵고 또한 이걸 위해 실물을 가져다 놓기엔 비용이 너무 많이 듈다.
  - 아이언 맨

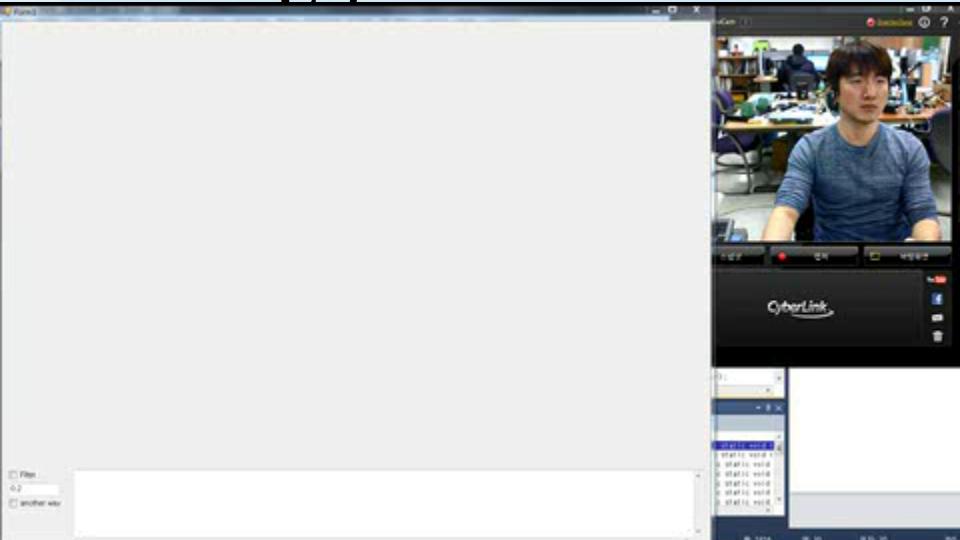
# 2008. Iron man

- Making the suit

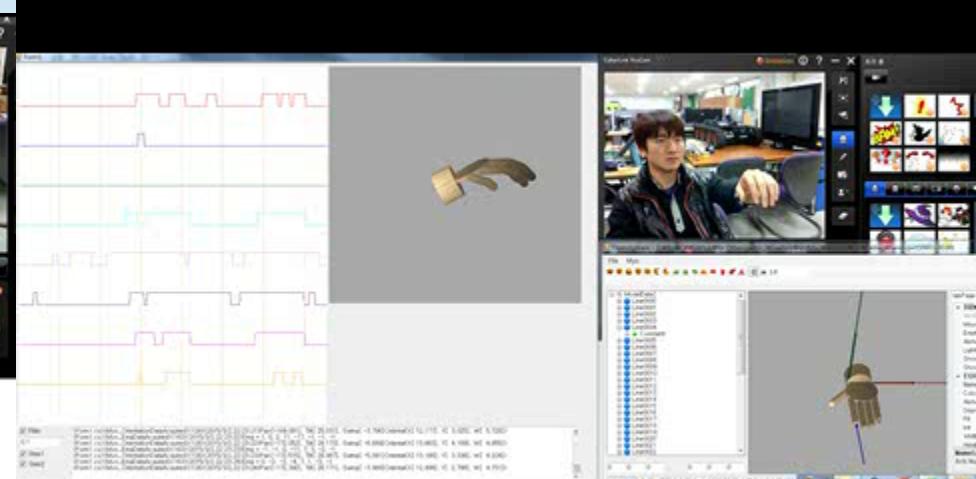


# 사용환경

- Microsoft Visual Studio 2010
  - C# 개발환경 ( with Tao frame work : OpenGL )



[https://youtu.be/l\\_QerQFRaZk](https://youtu.be/l_QerQFRaZk)



<https://youtu.be/GY4MUXbbW2o>

# ROS 가 있는데 이건 무슨 필요?

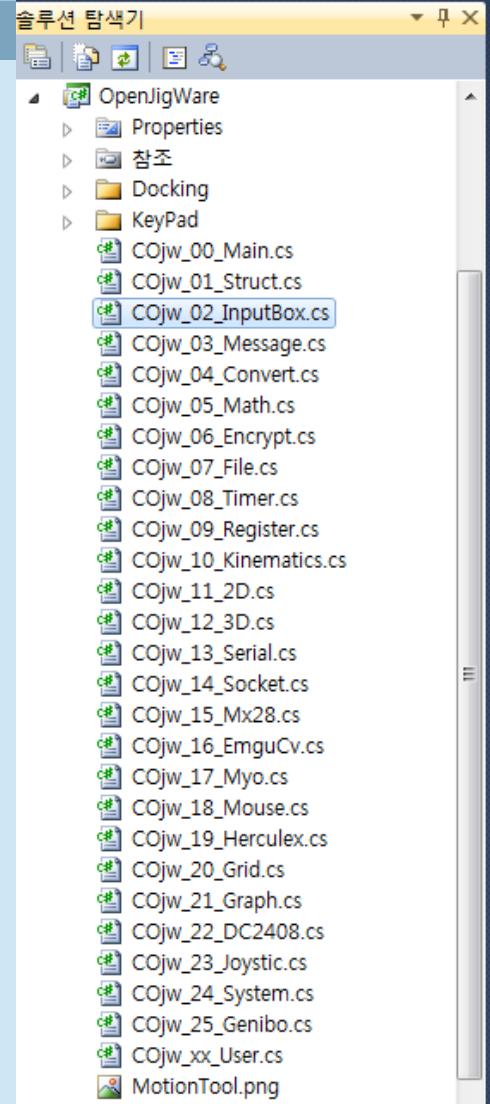
- 최초 개발은 모션 편집용 프로그램.
- Jig 개발에 최적화 되어 있다.
- 모션 제작 및 튜닝에 최적화
- 3D 모델링의 빠른 구현
  - 복잡한 형태도 숙련자의 경우 2시간 정도면 구현 완료 – 가상에서 보면서 조립하면 끝
  - 기타 세부적인 것을 정의하는데 시간이 걸림
    - 수식 / 실제 사용할 모터 기어비 / 자신이 넣고 싶은 정보들...

# 어디에서 구할 수 있는가?

- Source & DLL
  - OpenJigWare 로 인터넷 검색 하거나
  - 아래의 링크에서 소스 포함 다운로드 가능
    - <https://github.com/ojw5014/OpenJigWare>. ~~~
- Manual
  - 현재 Document 로는 없고 동영상 강의로 조금 씩 올리고 있습니다.
    - [www.youtube.com](http://www.youtube.com)
    - ojw5014 아이디를 찾으시거나 OpenJigWare 로

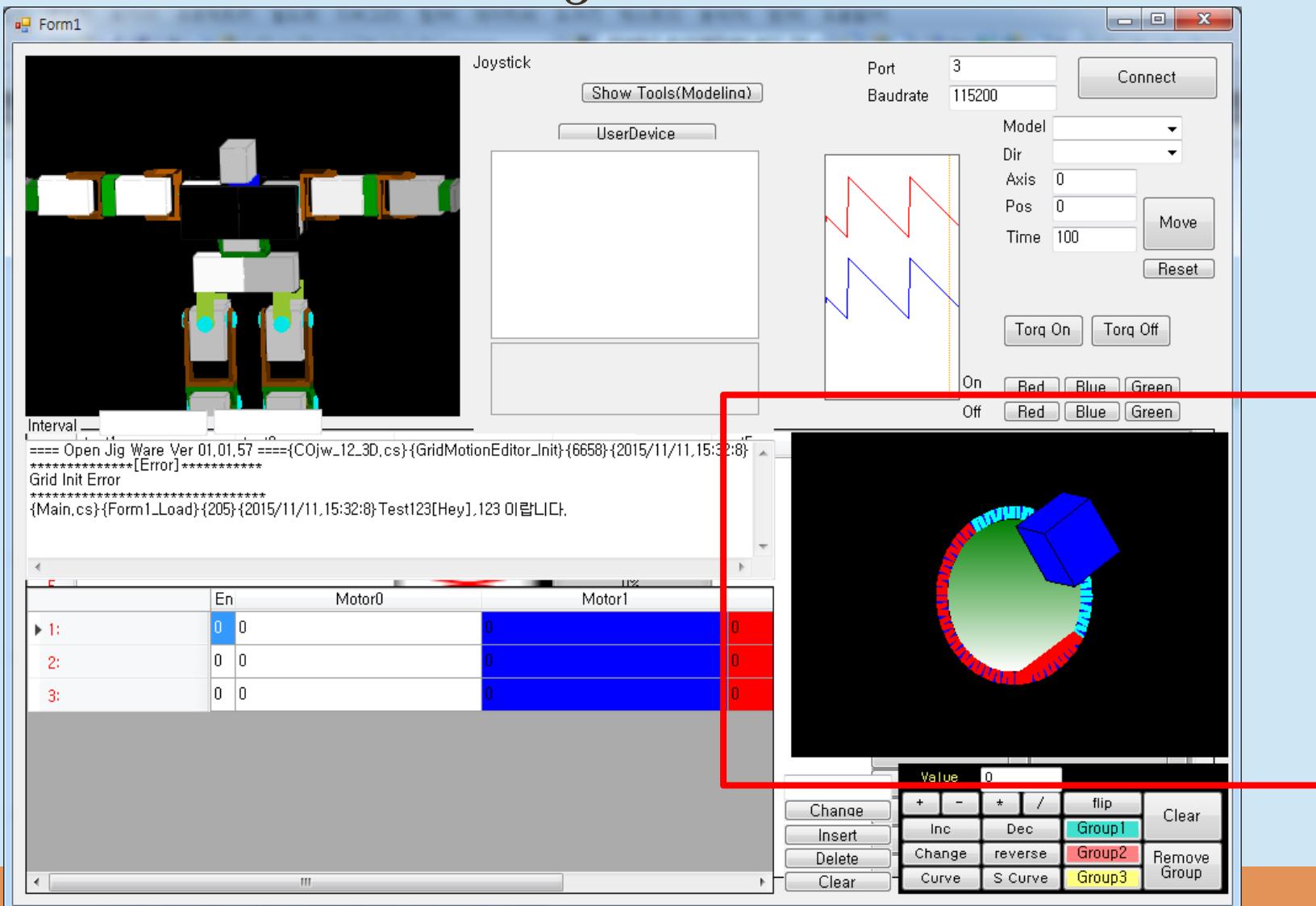
# 중요기능

- Message History
  - 내가 원하는 메시지를 MessageBox 와 연결하여 History Message 를 남길 수 있고 원하는 경우 이를 파일로 에러와 구분해서 저장이 가능(날짜별 구분 생성)
- Convert 가 직관적으로 쉽다.
  - 문자변환 등 형변환 관련
- Parameter 파일 생성 및 Loading 이 쉽다.



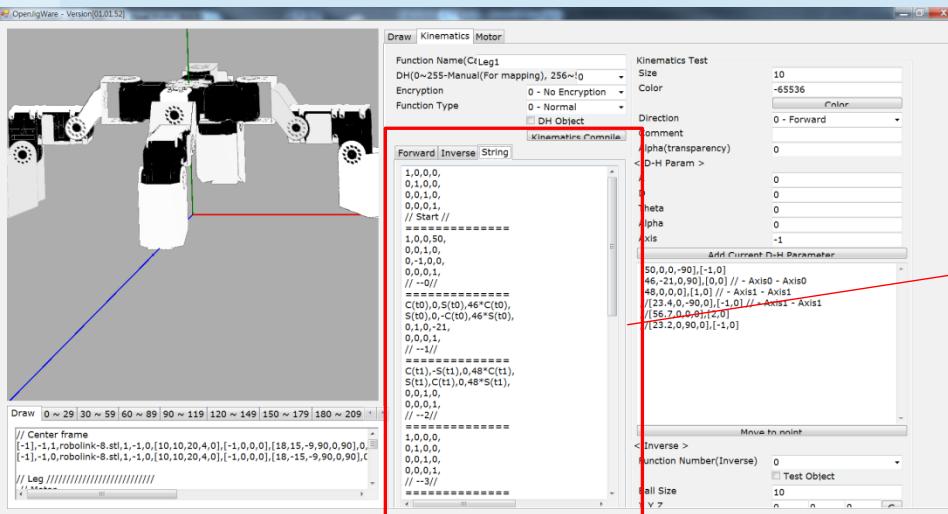
# 중요기능(추가)

- 2D 그래픽을 이용해 3D 그래픽을 구현한 모습



# [ 지원되는 기능 ]

- 초보자에게 쉬워야 한다.
  - 코딩라인 5줄 이내로 3D 모델링이 가능(참조구문, 변수선언 포함)
- 전문가에 의한 코딩의 다양화가 가능해야 한다.
  - 내부 OpenGL을 활용한 세부적 접근 가능



→

## Inverse Kinematics (3/3)

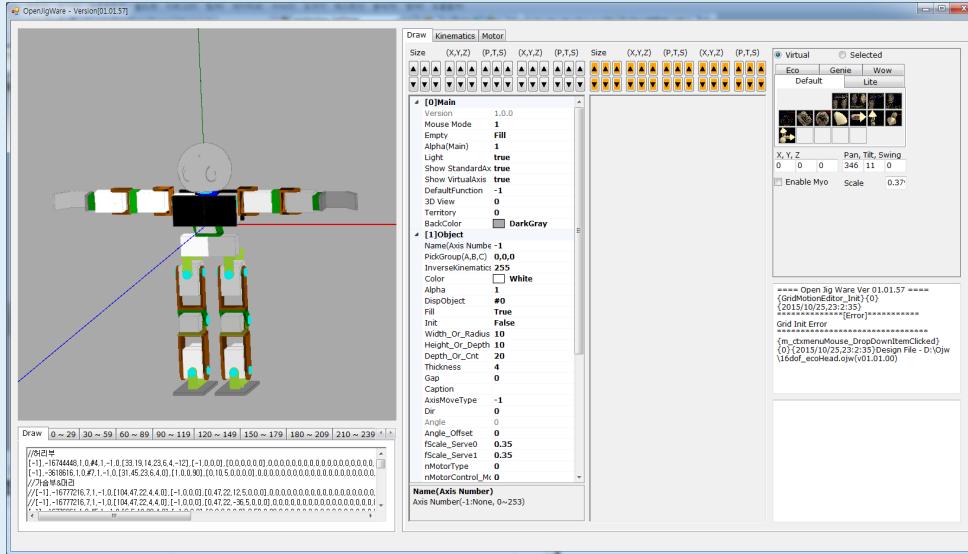
Axis X	Axis Y	Axis Z	Result
$-S(t2)*C(t3+90)*C(t4)-S(t2)*S(t3+90)*S(t4)*C(t5)$ $-S(t2)*C(t3+90)*S(t4)-S(t2)*S(t3+90)*C(t4)*S(t5)$  $S(t+90)*C(t4)+C(t3+90)*S(t4)*S(t5)+S(t3+90)*C(t4)+C(t3+90)*C(t4)*C(t5)$ $-1*C(t2)*C(t3+90)*C(t4)+C(t2)*S(t3+90)*S(t4)*C(t5)$ $-1*C(t2)*C(t3+90)*S(t4)-1*C(t2)*S(t3+90)*C(t4)*S(t5)$	$-S(t2)*C(t3+90)*S(t4)*C(t5)$ $-S(t2)*C(t3+90)*S(t4)+C(t3+90)*C(t4)*C(t5)$  $S(t3+90)*C(t4)+C(t3+90)*S(t4)*S(t5)+S(t3+90)*C(t4)+C(t3+90)*C(t4)*C(t5)$ $-1*C(t2)*C(t3+90)*C(t4)-1*C(t2)*S(t3+90)*S(t4)*C(t5)$ $-1*C(t2)*C(t3+90)*S(t4)-1*C(t2)*S(t3+90)*C(t4)*S(t5)$	$C(t2)$  $0$	$-S(t2)*C(t3+90)*482*C(t4)-S(t2)*S(t3+90)*482*S(t4)-S(t2)*625*C(t3+90)$ $S(t3+90)*482*C(t4)+C(t3+90)*482*S(t4)+625*S(t3+90)+200+190$  $-1*C(t2)*C(t3+90)*482*C(t4)-1*C(t2)*S(t3+90)*482*S(t4)-1*C(t2)*625*(t3+90)-246$  $1$

$$X = -\sin(t_2) * \cos(t_3 + 90) * 482 * \cos(t_4) - \sin(t_2) * -\sin(t_3 + 90) * 482 * \sin(t_4) - \sin(t_2) * 625 * \cos(t_3 + 90)$$

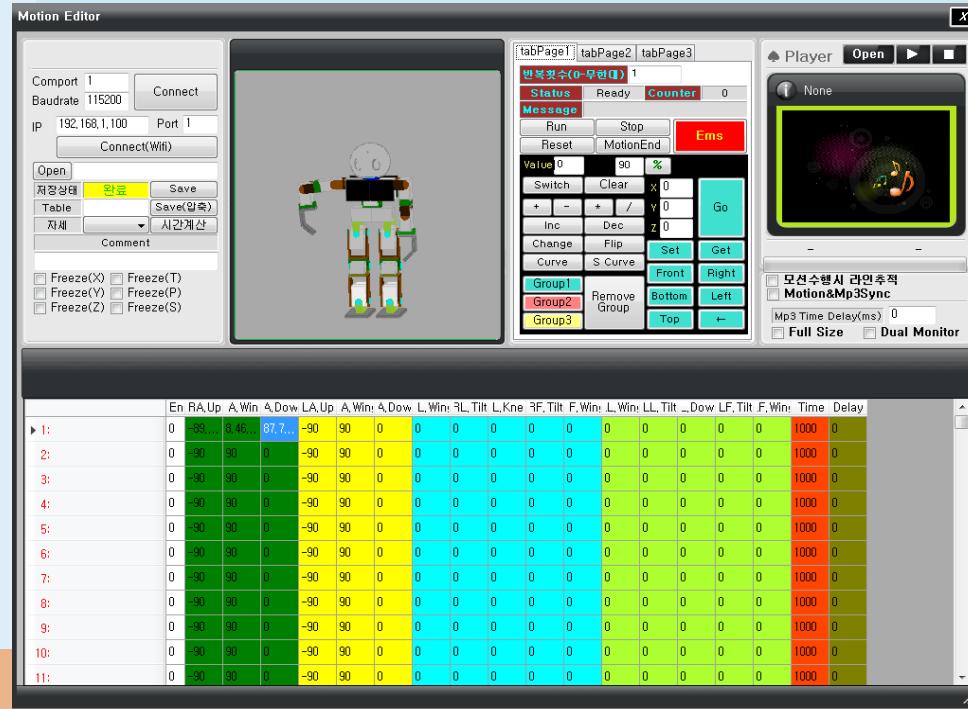
$$Y = \sin(t_3 + 90) * 482 * \cos(t_4) + \cos(t_3 + 90) * 482 * \sin(t_4) + 625 * \sin(t_3 + 90) + 200 + 190$$

$$Z = -1 * \cos(t_2) * \cos(t_3 + 90) * 482 * \cos(t_4) - 1 * \cos(t_2) * -\sin(t_3 + 90) * 482 * \sin(t_4) - 1 * \cos(t_2) * 625 * \cos(t_3 + 90) - 246$$

# Modeling tool



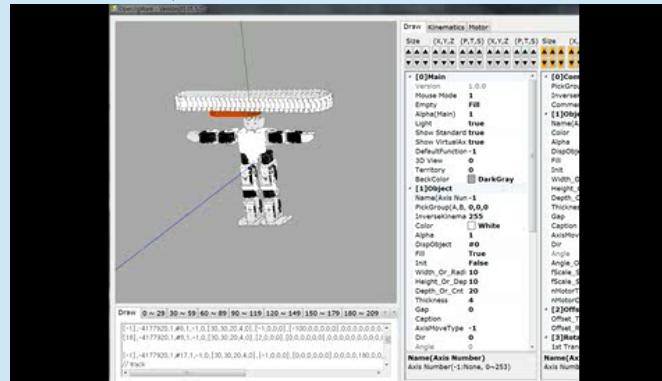
# Motion tool



# 모델링 툴, 구현

## • 모델링 툴

<https://youtu.be/srUZRiMfd8k>

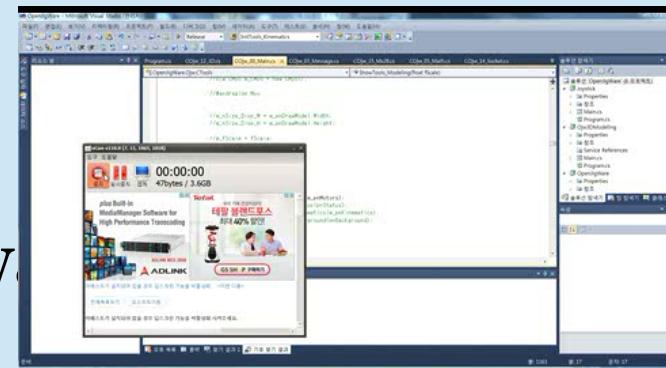


## 수식구현

## 교육대상자의 프로그램 실습

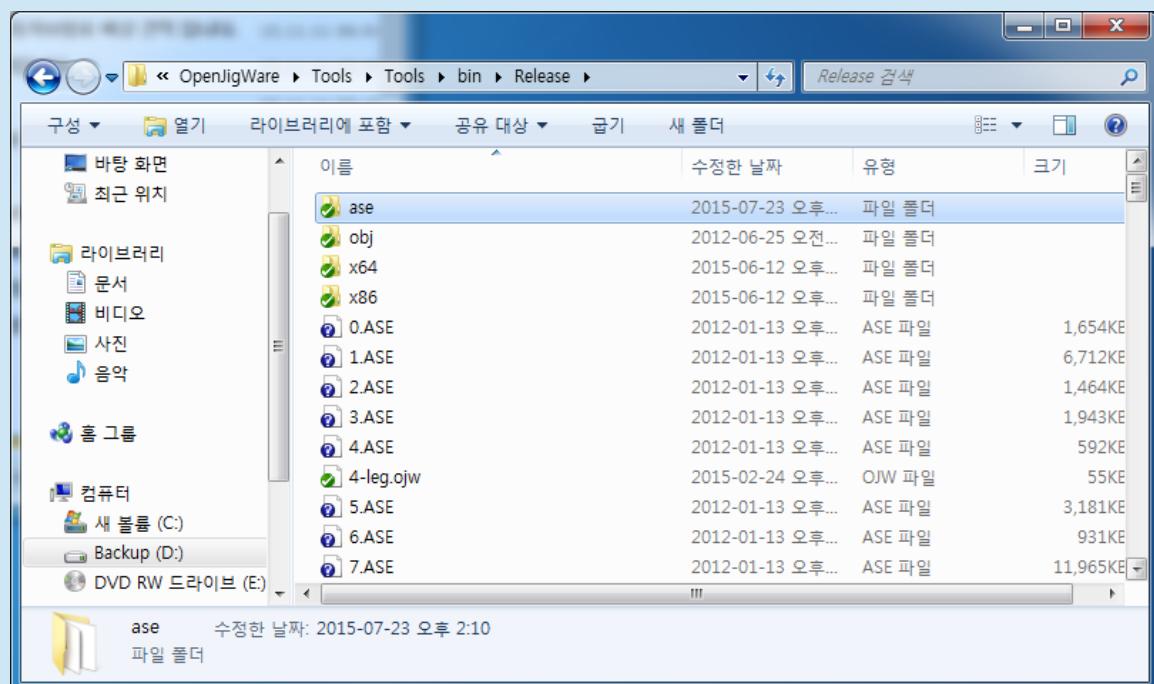
[https://youtu.be/BER\\_5CZqk-w](https://youtu.be/BER_5CZqk-w)

- [https://www.youtube.com/watch?v=\\_c2tNN-tayA](https://www.youtube.com/watch?v=_c2tNN-tayA)
- 모션제작 툴을 만들어 보기
- 완성까지 프로그래밍 총 200라인
- [https://www.youtube.com/watch?v=\\_c2tNN-tayA](https://www.youtube.com/watch?v=_c2tNN-tayA)



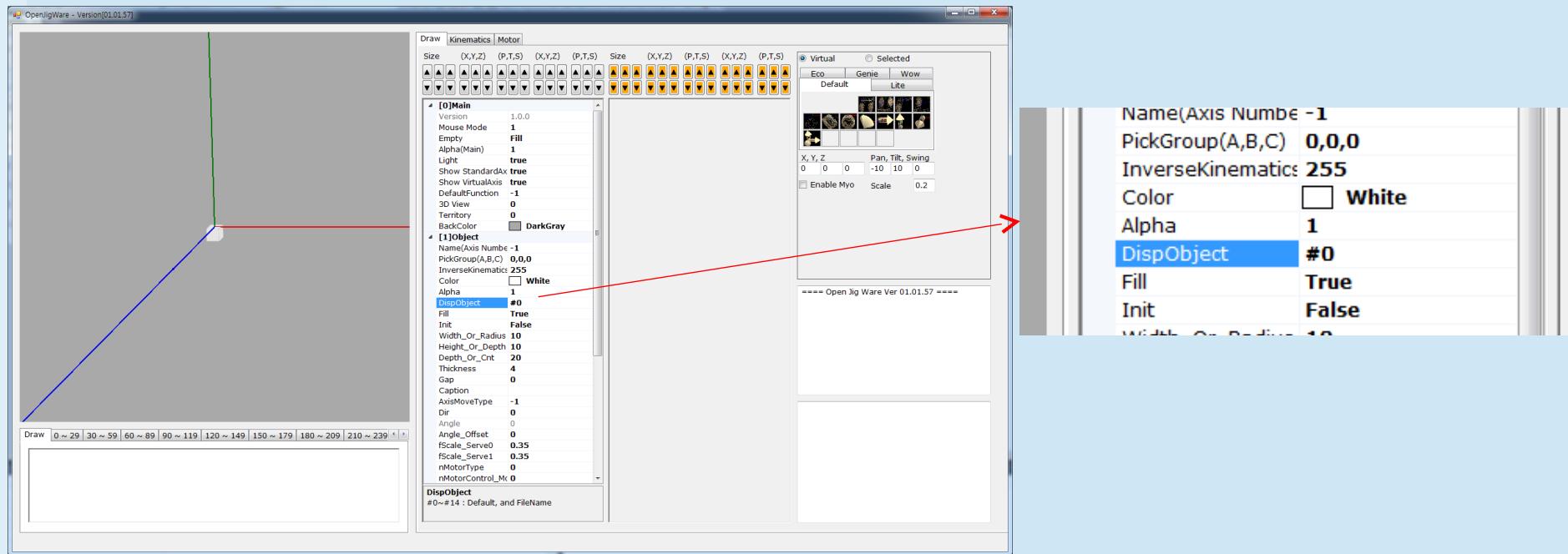
# 모델링 툴

- Cad 를 넣어보자. – Step 1
  - 가지고 있는 Cad 파일(obj, ase, stl 파일을 실행 폴더에 넣어준다. – 구 버전에서는 ase 폴더에 넣어준다.)



# 모델링 툴

- Cad 를 넣어보자. – Step 2
  - 모델링 툴 안에서 DispObject 항목의 이름을 가지고 있는 파일의 이름으로 바꿔준다.
    - # 으로 시작하는 이름은 내부에 정의된 17가지 패턴을 의미
    - Ase 파일은 확장자 없이 적어도 상관없다. 다른 파일들은 확



# 모델링 툴

- Cad 를 넣어보자. – Step 3
  - 들어간 모델을 확인 후 원하는 위치로 이동 및 회전 후 Add 한다.

The screenshot shows a CAD application window with a 3D view on the left displaying a white rectangular model with a ribbed base. To the right of the view is a property manager containing several parameters:

DefaultFunction	-1
3D View	0
Territory	0
BackColor	DarkGray
[1]Object	
Name(Axis Numbe	-1
PickGroup(A,B,C)	0,0,0
InverseKinematic	255
Color	<input type="checkbox"/> White
Alpha	1
DispObject	test.stl
Fill	True
Text	

Below the main view, there are two orange callout boxes with arrows pointing to specific parameters in the property manager:

- 1. Offset 으로 100 이동 후 45도 회전  
2. Y축으로 50 이동, Y축 45도 회전
- 1. Offset 은 실 회전과 상관없이 현재의 자세를 결정  
2. Trans / Rotation 은 실제적이 이동 및 회전을 의미(일반적인 OpenGL 3D 모델링에서의 회전 및 이동의 이미와 동일)

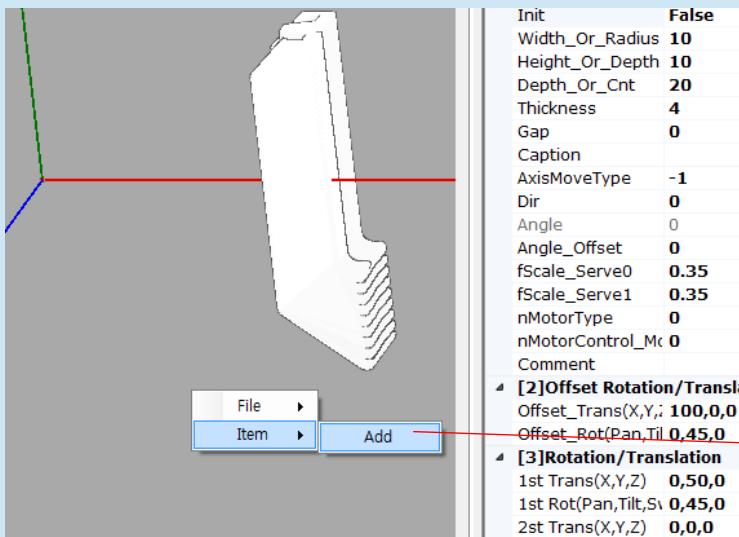
On the right side of the interface, there is another panel titled "Init" with the value set to "False". It contains the following parameters:

Init	False
Width_Or_Radius	10
Height_Or_Depth	10
Depth_Or_Cnt	20
Thickness	4
Gap	0
Caption	
AxisMoveType	-1
Dir	0
Angle	0
Angle_Offset	0
fScale_Serve0	0.35
fScale_Serve1	0.35
nMotorType	0
nMotorControl_Mc	0
Comment	

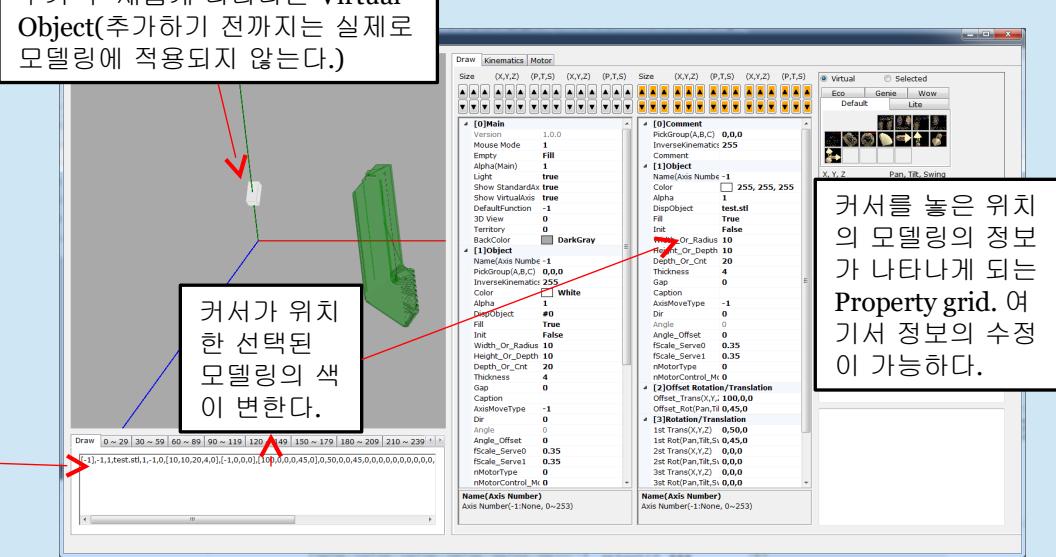
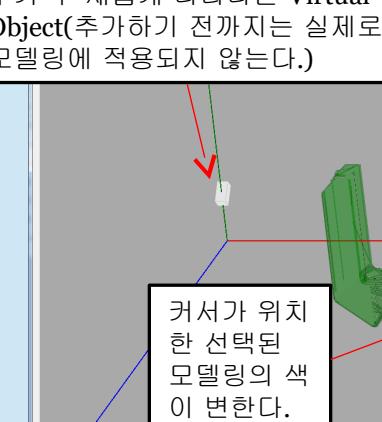
Below this panel, there are two additional sections:

- [2]Offset Rotation/Translation
  - Offset\_Trans(X,Y,Z) 100,0,0
  - Offset\_Rot(Pan,Tilt,Sl) 0,45,0
- [3]Rotation/Translation
  - 1st Trans(X,Y,Z) 0,50,0
  - 1st Rot(Pan,Tilt,Sl) 0,45,0
  - 2nd Trans(X,Y,Z) 0,0,0

- Cad 를 넣어보자. – Step 4
- 장치를 Add 한다 (마우스 오른 클릭)

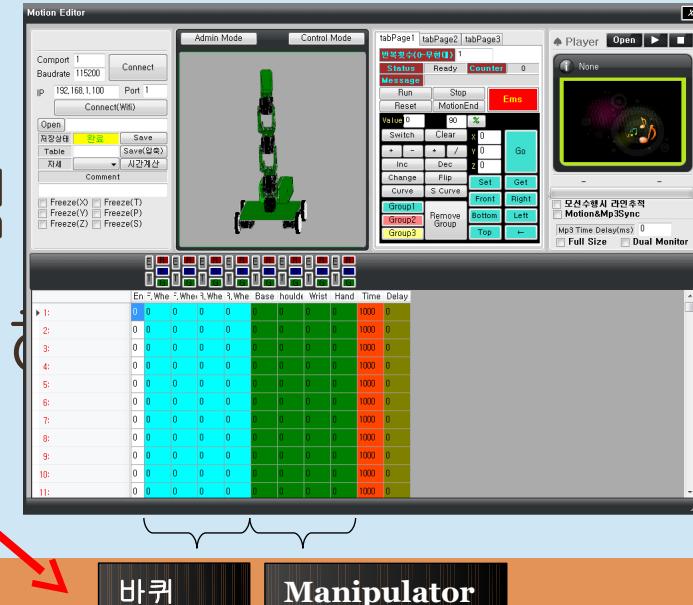


추가 후 새롭게 나타나는 Virtual Object(추가하기 전까지는 실제로 모델링에 적용되지 않는다.)



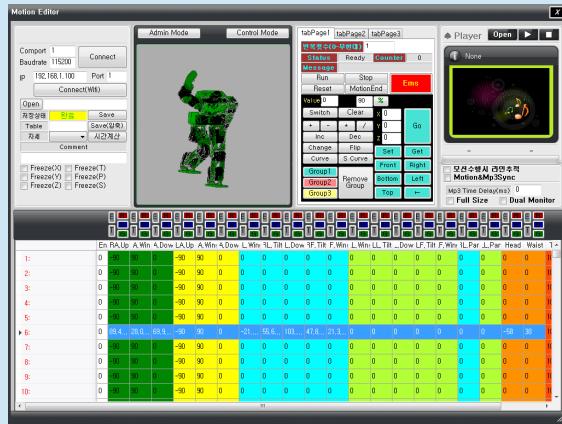
# 모션제작 툴

- 현재 제작 중(60%)
- 불러들이는 모델의 종류에 따라 모션툴의 기능이 다양하게 변화한다.
- 관절의 역할에 따라 색을 달리 정하는 것 이 가능하다.
- 클릭된 위치에 따라 어떤식 의 동작을 할 것인지 정의하는 것이 가능

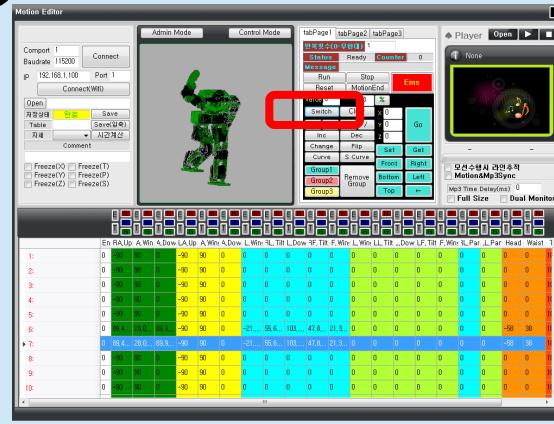


# 모션제작 툴(휴머노이드 모델링의 경

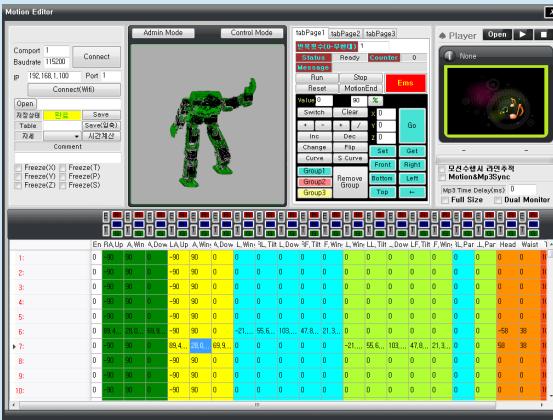
모델별 변화되는 기능 예 : Switch



테스트를 위해 모션 한 프레임을 만들어 본다.  
보는 바와 같이 우측 팔과 우측 다리를 들고 머리를 우측으로 돌리며 허리를 숙인것을 볼 수 있다.



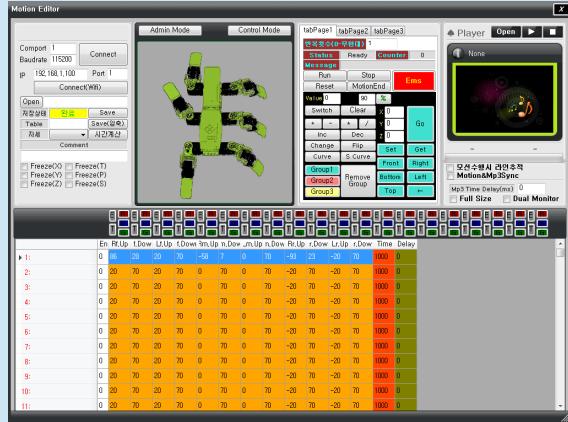
비교를 위해 아래에 Ctrl^C, Ctrl^V 하여 프레임  
복사를 한다. 이후 Switch 버튼 클릭



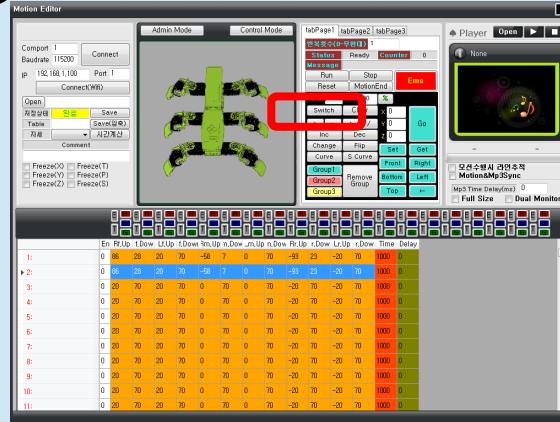
우측 팔/다리의 모션이 좌측 팔/다리로 바뀌었고, 머리는 반대방향으로, 허리는 변화 없는 것을 볼 수 있다.

# 모션제작 툴(HexaPod 모델링의 경)

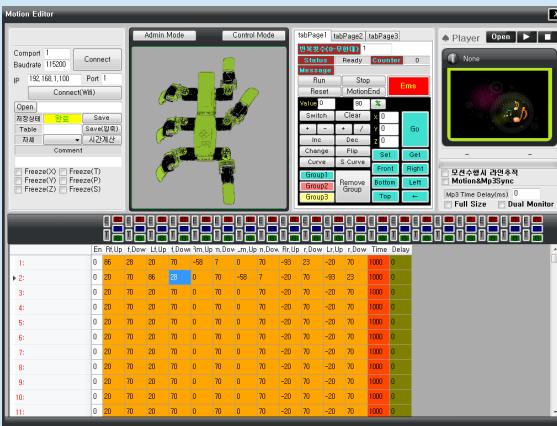
모델별 변화되는 기능 예 : Switch



프레임 복사



비교를 위해 아래에 Ctrl^C, Ctrl^V 하여 프레임  
복사를 한다. 이후 Switch 버튼 클릭



Switch 클릭

우측 관절 전체의 모션이 좌측관절로 바뀐것을 알 수 있다.

참고 : 이 모델링 파일은 그룹 구분이 되어 있지 않아 색 구분이 없이 주황색으로 통일 된 상태

# 모션제작툴 향후…

- Genibo 연동이 가능하도록 기능추가
- Hovis series 와 연동가능하도록 DR-Sim 호환 가능하도록 기능 추가
- 파일 저장 및 변환이 가능하도록 기능 추가
- 음악과 Sync 동작이 가능하게끔 기능 추가

# 혼자 OpenSource 개발 시 문제점

- 자료를 올려놓을 곳이 마땅치 않다.
  - 현재 Github 사용
    - <https://github.com/ojw5014/OpenJigWare>
  - 다른 컴퓨터에서는?

# 사용 방법

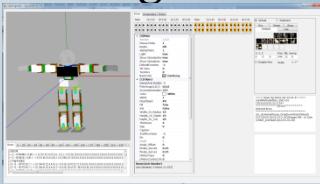
Using OpenJigware.dll

Teacher

Student

Engineer

Modeling Tool 실행



Drawing(3D)

Motor 정의

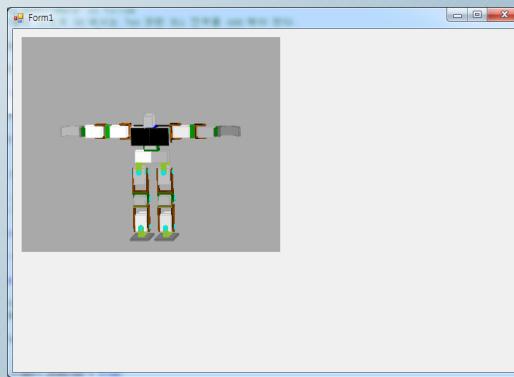
(Limit, 기어비 클릭 이벤트 등)

교육하고자 하는 수식 입력

만들어진 파일 학생에게 전달

Visual Studio C# 실행

프로그래밍 따라하기  
(최소 코딩라인 4줄)



Modeling Tool 실행

제어할 타입(델타, 직교,  
다관절, 바퀴형 외) 결정  
후 Drawing(3D)

Motor 정의  
(Limit, 기어비 클릭 이벤  
트 등)

사용하고자 하는 수식 입  
력

프로그램 제작

# 실제 소스를 확인해 보자

- 실행프로그램

Ex5\_3D - Microsoft Visual Studio (관리자)

Form1.cs x Form1.cs [디자인] Ex5\_3D.Form1 Form1

```
// For Use
// 1. 참조 - 참조추가 - 찾아보기 - DLL 선택(OpenJigWare.dll)
// 2. add "using OpenJigWare;" as follow
// 3. 다른 예제들과 다르게 3d에서는 Tao 관련 DLL 전부를 Add 해야 한다.
// 4. freeglut.dll 파일은 실행 폴더에 같이 복사해 두어야 한다.
using OpenJigWare;

namespace Ex5_3D
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        #region 변수 선언
        // 변수 선언
        private Ojw.C3d.m_C3d = new Ojw.C3d();
        #endregion 변수 선언 Ojw.C3d.C3d0

        private void Form1_Load(object sender, EventArgs e)
        {
            #region 3D 그림
            // 이것만 선언하면 기본 선언은 끝.
            m_C3d.Init(picDisp);

            if (m_C3d.FileOpen(@"test.dhf") == true) // 모델링 파일이 잘 로드 되었
            {
                //m_C3d.OjwDraw(); // 3D 모델을 화면에 출력한다.
                timer1.Enabled = true;
            }
            #endregion 3D 그림
        }

        private void timer1_Tick(object sender, EventArgs e)
        {
            #region 그리자
            m_C3d.OjwDraw();
            #endregion 그리자
        }
    }
}
```

솔루션 탐색기 Properties 참조 Microsoft.CSharp OpenJigWare System System.Core System.Data System.Data.DataSetExtensions System.Deployment System.Drawing System.Windows.Forms System.Xml System.Xml.Linq Tao.FreeGlut Tao.Ode Tao.OpenGL Tao.Platform.Windows Form1.cs Form1.Designer.cs

한 줄 더 추가하면...

Version: 1.0.0  
Mouse Mode: Fill  
Empty: Fill  
Alpha(Main): 1  
Light: true  
Show StandardAxis: false  
Show VirtualAxis: false  
DefaultFunction: 1  
3D View: 0  
Territory: 0  
BackColor: DarkGray  
[1]Object  
Name(Axis Number): -1  
Data(Signed & C)  
Name(Axis Number): Axis Number(-1:None, 0-255)

출석 출력 보기 선택(S):

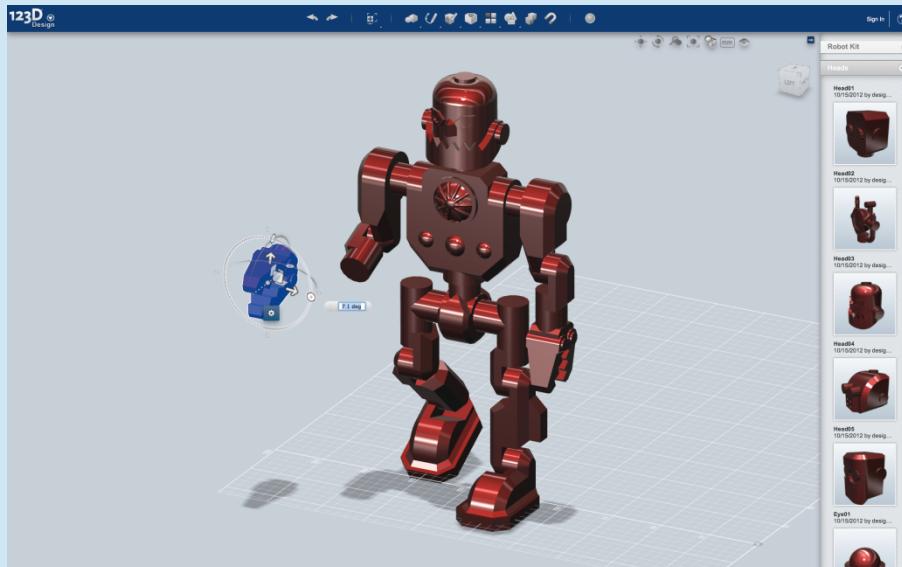
준비 출처 찾기 결과 1

오류 목록 출처 찾기 결과 1

줄: 17 열: 1 문자: 1 INS

# 모델링 데이터는 어디서 구할까? [1/2]

- 직접 그린다.
  - 솔리드 웍스, Maya, 3D Max 및 기타 3D 모델링 툴 이용(파일 저장은 **obj, ase, stl[Binary or Text]**로 한다.)
  - Autodesk 123Design 같은 툴을 사용한다.(무



는데 시간이 걸리므로  
과가 나타난다.

# 모델링 데이터는 어디서 구할까? [2/2]

- 난 그릴줄 몰라!!!
  - 다운로드한다.
    - 3D 프린터와 동일, Thingiverse, Yeggi 등의 사이트에서

The screenshot shows a 3D model of a robotic arm on a green background. To the right is a sidebar with social sharing options: Like (58), Collect (64), Comment (1), I Made One (0), Remix It (0), and Share. Below this is a blue button labeled "Download This Thing!". At the bottom of the page, there are links for "Thing Details", "Thing Files", "Comment (1)", "Mode (0)", "Collections (64)", and "Remixes (0)". A "Summary" section at the very bottom indicates "Work in progress!" with statistics: 4587 Views and 1319 Downloads.

<http://www.thingiverse.com>

The screenshot shows a search results page for "yeggi". The top navigation bar includes links for "printable 3d models", "search now!", and "search". Below the search bar, there are several advertisements for 3D printing services like Try 3DPrinterOS, Astroprint, Trijexx, and All3DP. The main content area is titled "Search Engine for 3D printable Models" and features a grid of 3D model thumbnails. The first row includes a Plus Vase, a Simple Lamp, a Voronoi lamp 2, a Tiny hurricane lamp, and a Lamp "Jellyfish". The second row includes a Triangles iPhone 6 case, a STARS iPhone 6 Case, a phone 6, an iPhone 6 Plus case, and a Model of iPhone 6 Case. A section titled "3D Model Selections - most searched and clicked" is also visible.

<http://www.yeggi.com/>

# 수식부 컴파일러

**수식 간략화(컴파일 전단계) & 컴파일**

length = 5  
 $t0 = v0 + length * \sin(30)$  // Input Const  
 $t1 = -t0$   
 $v1 = \text{pow}(v0, 2)$  // Input another external var(v1)

**1단계**

```
length=5
t0=v0+length*sin(30)
t1=-t0
v1=pow(v0,(2))
```

**2단계**

```
_V0+=30
_W0+=length*sin_V0
_T0+=_K0+_W0
_T1=-_T0
_V2+=2
_V1+=_K0,_UP0,_V2
_K1+=pow_V1
```

**2단계**

```
VAR_M0
CLB_M0
ADD_5
VAR_M1
CLR_M1
ADD_30
VAR_M2
CLR_M2
ADD_M0
MUL_sin_M1
VAR_T0
CLR_T0
ADD_T0
ADD_K0
ADD_M2
VAR_T1
CLR_T1
SUB_T0
VAR_M3
CLR_M3
ADD_2
VAR_M4
```

**3단계**

```
VAR_length
CLB_length
ADD_5
VAR_V0
CLR_V0
ADD_30
VAR_W0
CLR_W0
ADD_length
MUL_sin_V0
VAR_T0
CLR_T0
ADD_T0
ADD_K0
ADD_W0
VAR_T1
CLR_T1
SUB_T0
VAR_V2
CLR_V2
ADD_2
VAR_V1
```

**4단계**

```
0000000001_M0
000000000e_M0
0000000002_5
0000000001_M1
000000000e_M1
0000000002_30
0000000001_M2
000000000e_M2
0000000002_M0
0000000004_M1
0000000001_T0
000000000e_T0
0000000002_K0
0000000001_T1
000000000e_T1
0000000003_T0
0000000001_M3
000000000e_M3
0000000002_2
0000000001_M4
```

**사용된 모터의 개수**  
2

**사용된 모터의 ID들...**  
Motor-0-1

**사용된 변수의 개수**  
2

**사용된 변수 번호들...**  
변수(V)-0-1

**실제 사용될 바이너리 코드 생성**

t(Index) 0 t(Value) 0

x 0 y 0 z 0

0	Ovr	12.5	v0	10
1	Ovr	-12.5	v1	0
2	Ovr	0	v2	0
3	Ovr	0	v3	0
4	Ovr	0	v4	0
5	Ovr	0	v5	0
6	Ovr	0	v6	0
7	Ovr	0	v7	0
8	Ovr	0	v8	0
9	Ovr	0	v9	0
10	Ovr	0	v10	0
11	Ovr	0	v11	0
12	Ovr	0	v12	0
13	Ovr	0	v13	0
14	Ovr	0	v14	0
15	Ovr	0	v15	0
16	Ovr	0	v16	0
17	Ovr	0	v17	0

x=0  
y=0  
z=0  
Mot0=12.5  
Mot1=-12.5  
Mot2=0  
V0=10  
V1=100  
V2=0  
V3=0  
V4=0  
V5=0  
V6=0  
V7=0  
V8=0  
V9=0  
V10=0  
V11=0  
V12=0  
V13=0  
V14=0  
V15=0  
V16=0  
V17=0  
V18=0  
V19=0

암호화 >

암호화 <

# 수식부 컴파일러

- 사전 정의된 변수(Non-Case Sensitive)  
=> 모든 변수는 컴파일러 외부에서 연산 전에 넣거나 연산 후 결과값을 되돌려 받을 수 있는 변수
  - X, Y, Z( 대소문자 구분 없음)
  - To ~ T<sub>255</sub> : 모터 변수
  - Vo ~ V<sub>252</sub> : External 지정 변수 (프로그램 외부에서 변수값을 사전에 미리 정의하거나 컴파일 과정에서 연산된 값을 사용자가 임의로 알 수 있는 변수)
- 사용된 문법

# 수식 간략화 단계

- 컴파일 전단계
- 정렬 및 주석제거 역할
  - String separate – 스트링 데이터를 조각조각 쪼개 놓는다.
  - 쪼개진 데이터를 순서에 입각해서 변수 정의한다.
  - \_V(변수넘버)를 순서에 맞게 정렬한다.

```
length = 5           // Input Const
t0 = v0 + length * sin(30) // Calc with v0[external var]
t1 = -t0
v1 = pow(v0, 2)      // Input another external var(v1)
```

정렬

```
length=5
t0=v0+length*sin(30)
t1=-t0
v1=pow(v0,(2))
```

# 컴파일 단계

- 암호화 및 암호화 해제, v 변수, T 모터변수 및 내부변수 정의, 불필요 문자 제거, 내부 Basic 규정 문법으로 변환, 수식의 단순화 전처리 과정

```
length=5  
t0=v0+length*sin(30)  
t1=-t0  
v1=pow(v0,(2))
```

```
length=+5  
  
_V0=+30  
_W0=+length*sin_V0  
_T0=+_K0+_W0  
  
_T1=-_T0  
  
_V2=+2  
_V1=+_K0,_UP0__V2  
_K1=+pow_V1
```

# 컴파일 2 단계

- Basic 문자를 내부 규정 Assembler 문자로 변환

The screenshot shows a software interface for program compilation. On the left, there is a list of Basic statements. On the right, there is a corresponding list of assembly-like instructions. A red arrow points from the left column to the right column, indicating the transformation process.

Basic Statements (Left)	Assembly Instructions (Right)
length=+5	VAR,_M0 CLR,_M0 ADD,5
_V0=+30	VAR,_M1 CLR,_M1 ADD,30
_W0=+length*sin_V0	VAR,_M2 CLR,_M2 ADD,_M0
_T0=+_K0+_W0	MUL,sin_M1 VAR,_T0 CLR,_T0 ADD,_K0
 	ADD,_M2
_T1=-_T0	VAR,_T1 CLR,_T1 SUB,_T0
 	VAR,_T1 CLR,_T1 SUB,_T0
_V2=+2	ADD,_M3 CLR,_M3 ADD,2
 	VAR,_M4
_V1=+_K0,_UP0-_V2	
_K1=+pow_V1	

# 컴파일 3 단계

- Assembler 문자를 최종 점검을 위한 Code 문자로 변환

VAR,_M0	VAR,length
CLR,_M0	CLR,length
ADD,5	ADD,5
VAR,_M1	VAR,_V0
CLR,_M1	CLR,_V0
ADD,30	ADD,30
VAR,_M2	VAR,_W0
CLR,_M2	CLR,_W0
ADD,_M0	ADD,length
MUL,sin,_M1	MUL,sin,_V0
VAR,_T0	VAR,_T0
CLR,_T0	CLR,_T0
ADD,_K0	ADD,_K0
ADD,_M2	ADD,_W0
VAR,_T1	VAR,_T1
CLR,_T1	CLR,_T1
SUB,_T0	SUB,_T0
VAR,_M3	VAR,_V2
CLR,_M3	CLR,_V2
ADD,2	ADD,2
VAR,_M4	VAR,_V1

00000000001,_M0	2
0000000000e,_M0	Motor-0-1
00000000002,5	
00000000001,_M1	
0000000000e,_M1	
00000000002,30	
00000000001,_M2	
0000000000e,_M2	
00000000002,_M0	
00000000604,_M1	
00000000001,_T0	
0000000000e,_T0	
00000000002,_K0	
00000000002,_M2	
00000000001,_T1	
0000000000e,_T1	
00000000003,_T0	
00000000001,_M3	
0000000000e,_M3	
00000000002,2	
00000000001,_M4	

# 컴파일 4 단계

- Code Generation
  - Code 문자를 Binary Code 로 변환
  - 실 바이너리 코드의 생성(고속 연산처리 가능)
    - 이후의 연산은 엔진에서 바이너리 코드를 읽어서 실행한다.

# 수식에도 조건문을 넣고 싶다…

- 현재 수식컴파일러에는 조건문을 넣지 않았다… 그럼 관절이 Over 된 경우와 그렇지 않은 경우는 어떻게 구분하지? 조건문이 필요해…
- 수식만으로 조건문 생성이 가능하다?

# 수식으로 조건문 만들기

```
//////////  
// 수식참고  
// 짹수면 (-1), 홀수면 (1) - 변수값에 따라 부호를 반대로 주어야 할 경우 사용  
//v1=pow(-1,(vo + 1)) // 0(-1), 1(1), 2(-1). 3(1). 4(-1). 5(1). 6(-1). 7(1). 8(-1), 9(1), 10(-1)  
  
// 같으면 0, 적으면 -1, 많으면 1 - 비교문  
//a=5  
//v1=(vo-a)/abs(vo-a) // 0(-1), 1(-1), 2(-1). 3(-1). 4(-1). 5(0). 6(1). 7(1). 8(1), 9(1), 10(1)  
  
// 같으면 0, 다르면 1  
//a=5  
//v1=abs((vo-a)/abs(vo-a)) // 0(1), 1(1), 2(1). 3(1). 4(1). 5(0). 6(1). 7(1). 8(1), 9(1), 10(1)  
  
// 같으면 1, 다르면 0  
//a=5  
//v1=abs(abs((vo-a)/abs(vo-a))-1) // 0(1), 1(1), 2(1). 3(1). 4(1). 5(0). 6(1). 7(1). 8(1), 9(1), 10(1)  
//////////  
  
//vo = 0 인 경우 우측다리 = (중력방향 + v1(tilt각도) + v2(swing각도))  
//vo = 1 인 경우 좌측다리 = (중력방향 + v1(tilt각도) + v2(swing각도))  
//vo = 2 인 경우 양쪽다리 = (중력방향 + v1(tilt각도) + v2(swing각도))  
//vo = 3 인 경우 우측다리에 좌측다리 일치화(좌측다리 변화)  
//vo = 4 인 경우 좌측다리에 우측다리 일치화(우측다리 변화)  
//v1 = offset 각도(Tilt)  
//v2 = offset 각도(Swing)
```

# Kinematics

- 왜 Kinematics 를 풀까?

- About Kinematics...

- Forward kinematics

- Joint Angle → Cartesian space(x, y, z)
- Simple and Unique solution

- Inverse kinematics

- Cartesian space(x, y, z) → Joint Angle
- It is not a unique solution, difficult(but delta system)
- Sometimes we have singular position or some position we cannot go

- Forward Kinematics(1/2)

- Translation

$$\begin{aligned}X &= x + dx \\Y &= y + dy \\Z &= z + dz\end{aligned}$$

- to Matrix

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{vmatrix} 1 & 0 & dx \\ 0 & 1 & dy \\ 0 & 0 & dz \end{vmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

- Forward Kinematics(2/2)

- Rotation : R

$$X\text{축 회전} : R_x : \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix}$$

$$Y\text{축 회전} : R_y : \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}$$

$$Z\text{축 회전} : R_z : \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- Forward Kinematics (D-H Notation)

- T(Transformation Matrix)

- It has rotation and translation in their  $4 \times 4$  matrix

$$\begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} R & d_x \\ 0 & d_y \\ 0 & d_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

- Parameter

$a_i$ : 길이 (length)	$d_i$ : 오프셋 (offset)
$\alpha_i$ : 비틀림 (twist)	$\theta_i$ : 각도 (angle)

- Define

- $X(i+1)$  and  $Z(i)$  is orthogonal
- $X(i+1)$  and  $Z(i)$  has a matching point

(DH1)  $x_{i+1}$  축은  $z_i$  축과 수직이다.

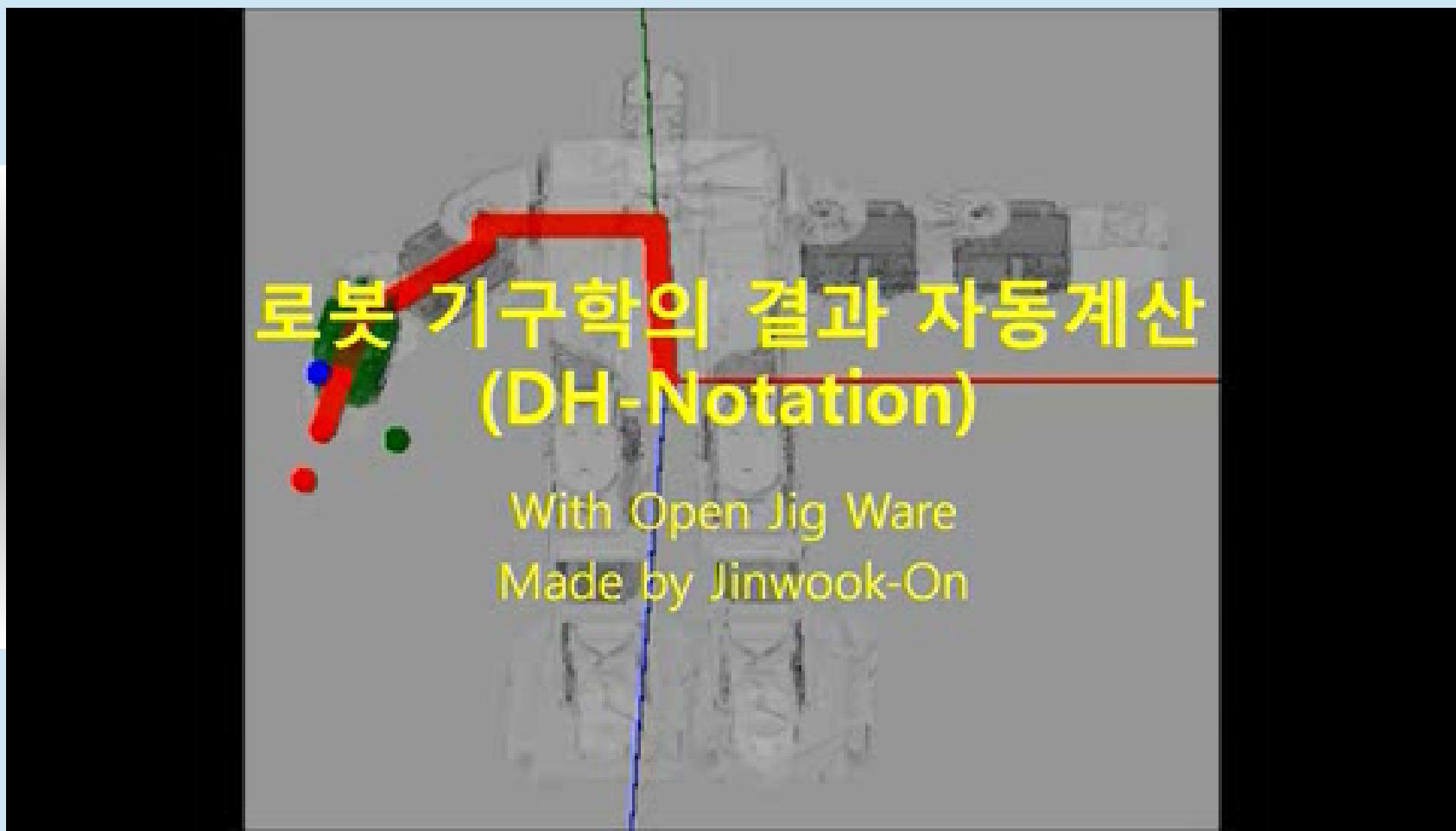
(DH2)  $x_{i+1}$  축은  $z_i$  축과 만난다.

## • D-H Notation Matrix

$$\begin{aligned} A_i &= \text{Rot}_{z,\theta_i} \text{Trans}_{z,d_i} \text{Trans}_{x,a_i} \text{Rot}_{x,\alpha_i} \\ &= \begin{bmatrix} C_{\theta_i} & -S_{\theta_i} & 0 & 0 \\ S_{\theta_i} & C_{\theta_i} & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & a_i \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & C_{\alpha_i} & -S_{\alpha_i} & 0 \\ 0 & S_{\alpha_i} & C_{\alpha_i} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} C_{\theta_i} & -S_{\theta_i}C_{\alpha_i} & S_{\theta_i}S_{\alpha_i} & a_iC_{\theta_i} \\ S_{\theta_i} & C_{\theta_i}C_{\alpha_i} & -C_{\theta_i}S_{\alpha_i} & a_iS_{\theta_i} \\ 0 & S_{\alpha_i} & C_{\alpha_i} & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned}$$

- 어렵다… 헷갈린다…

- 쉽게 가 보자
  - D-H Notation 눈으로 확인해 보기
    - <https://youtu.be/7lqSJNfkxe8>



- 성과…

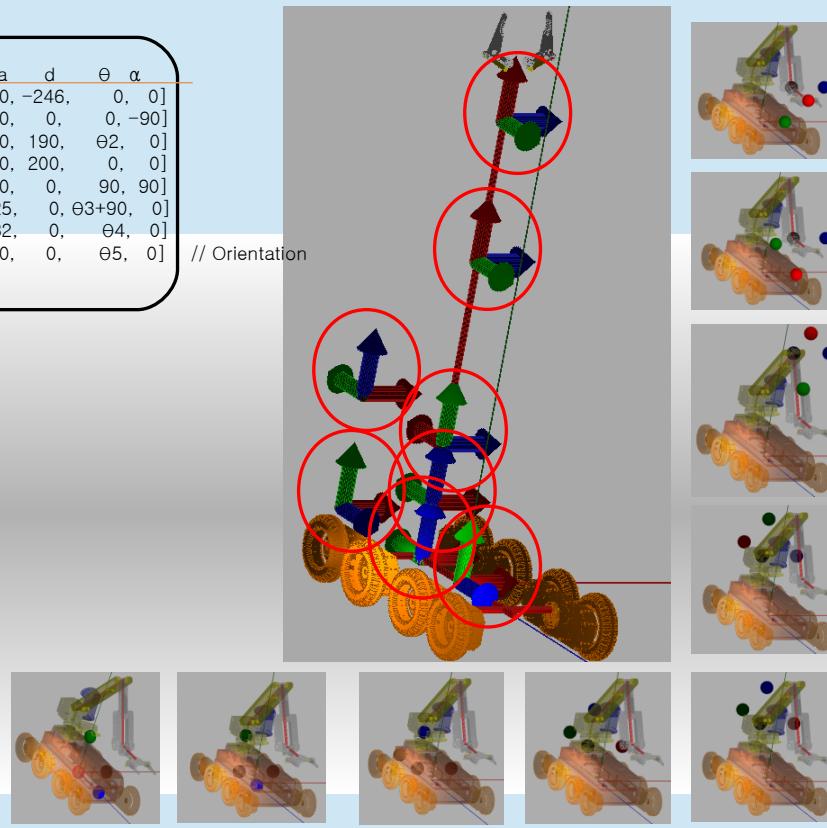
## ●kinematics

- D-H Notation 교육 2시간 만에 상당수 학생들이 D-H 파라미터를 이용한 3D 매니퓰레이터 제작 성공
- 자신이 스스로 3D 매니퓰레이터를 제작한 것에 흥미를 가짐
- D-H Notation 에 대한 이해를 쉽게 가짐.
  - 헷갈리면 그려보면 된다…o

# • Forward Kinematics & Program(1/2)

## D-H Notation

a	d	$\theta$	$\alpha$
0	[ 0, -246, 0, 0]		
1	[ 0, 0, 0, -90]		
2	[ 0, 190, $\theta_2$ , 0]		
3	[ 0, 200, 0, 0]		
4	[ 0, 0, 90, 90]		
5	[625, 0, $\theta_3+90$ , 0]		
6	[482, 0, $\theta_4$ , 0]		
7	[ 0, 0, 0, 0]	// Orientation	



1 0 0 0	1 0 0 0	$C(t2) -S(t2) 0 0$	1 0 0 0
0 1 0 0	0 0 1 0	$S(t2) C(t2) 0 0$	0 0 0 0
0 0 1 -246	0 -1 0 0	0 0 1 190	0 0 1 200
0 0 0 1	0 0 0 1	0 0 0 1	0 0 0 1

Add Rotation( $\theta_5$ )

$C(t4) -S(t4) 0 482*C(t4)$   
 $S(t4) C(t4) 0 482*S(t4)$

0	0	1	0
0	0	0	1

$C(t3+90) -S(t3+90) 0 625*C(t3+90)$   
 $S(t3+90) C(t3+90) 0 625*S(t3+90)$

0	0	1	0
0	0	0	1

0 0 1 0
1 0 0 0
0 1 0 0
0 0 0 1

- Forward Kinematics & Program(2/2)

<https://youtu.be/detjVR1dGo8>

# Kinematics 확인

[Axis X]	[Axis Y]	[Axis Z]	[Result]
$-S(t2)*C(t3+90)*C(t4)-S(t2)*-S(t3+90)*S(t4)*C(t5)$ $-S(t2)*C(t3+90)*-S(t4)-S(t2)*-S(t3+90)*C(t4)*S(t5)$  $S(t3+90)*C(t4)+C(t3+90)*S(t4)*C(t5)+$ $S(t3+90)*-S(t4)+C(t3+90)*C(t4)*S(t5)$  $-1*C(t2)*C(t3+90)*C(t4)-1*C(t2)*-S(t3+90)*S(t4)*C(t5)$ $246$ $-1*C(t2)*C(t3+90)*-S(t4)-1*C(t2)*-S(t3+90)*C(t4)*S(t5)$	$-S(t2)*C(t3+90)*C(t4)-S(t2)*-S(t3+90)*S(t4)*-S(t5)$ $-S(t2)*C(t3+90)*-S(t4)-S(t2)*-S(t3+90)*C(t4)*C(t5)$  $S(t3+90)*C(t4)+C(t3+90)*S(t4)*-S(t5)+S(t3+90)*$ $-S(t4)+C(t3+90)*C(t4)*C(t5)$  $-1*C(t2)*C(t3+90)*C(t4)-1*C(t2)*-S(t3+90)*S(t4)*-S(t5)$  $-1*C(t2)*C(t3+90)*-S(t4)-1*C(t2)*-S(t3+90)*C(t4)*C(t5)$	$C(t2)$  $S(t3+90)*482*C(t4)+C(t3+90)*482*S(t4)+625*S(t3+90)+200+190$  $-1*S(t2)$	$-S(t2)*C(t3+90)*482*C(t4)-S(t2)*-S(t3+90)*482*S(t4)-S(t2)*625*C(t3+90)$  $-1*C(t2)*C(t3+90)*482*C(t4)-1*C(t2)*-S(t3+90)*482*S(t4)-1*C(t2)*625*C(t3+90)-$ $246$
0	0	1	

$$X = -\sin(t2) * \cos(t3+90) * 482 * \cos(t4) - \sin(t2) * -\sin(t3+90) * 482 * \sin(t4) - \sin(t2) * 625 * \cos(t3+90)$$

$$Y = \sin(t3+90) * 482 * \cos(t4) + \cos(t3+90) * 482 * \sin(t4) + 625 * \sin(t3+90) + 200 + 190$$

$$Z = -1 * \cos(t2) * \cos(t3+90) * 482 * \cos(t4) - 1 * \cos(t2) * -\sin(t3+90) * 482 * \sin(t4) - 1 * \cos(t2) * 625 * \cos(t3+90) - 246$$

[For example... ]

$$t2 = 30$$

$$t3 = 60$$

$$t4 = 110$$

$$t5 = -60$$

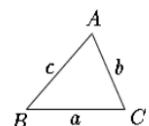
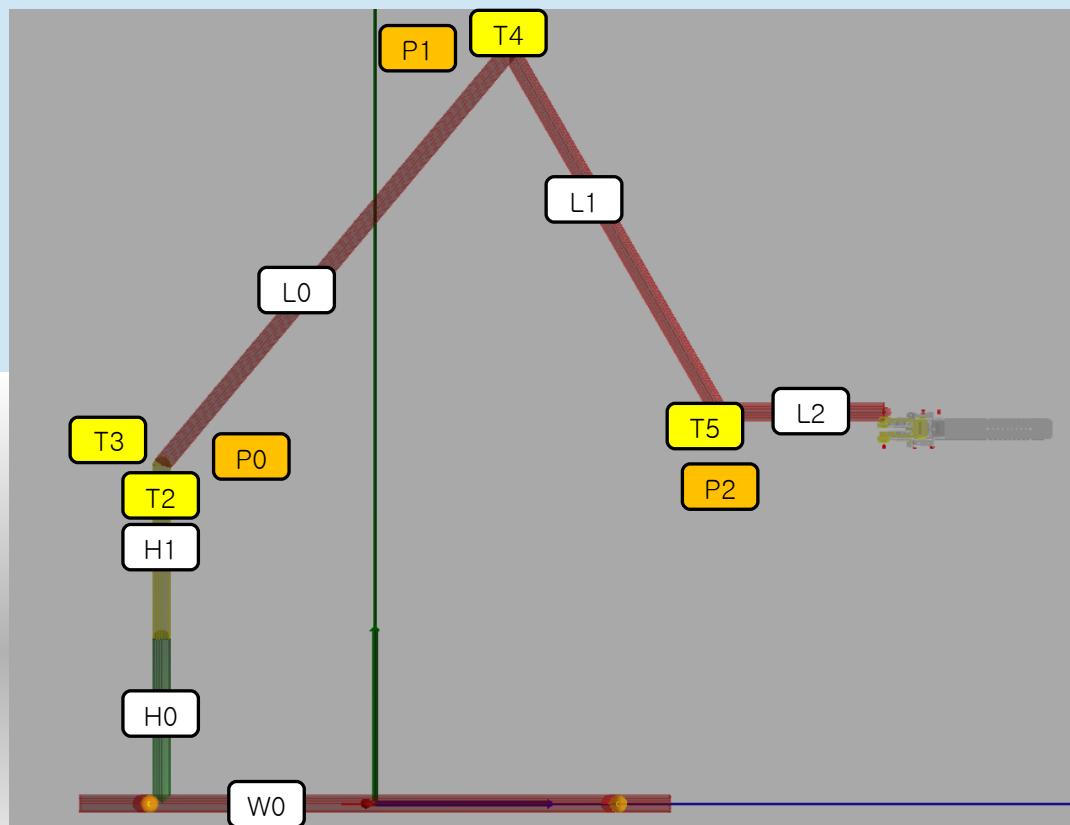
=====

$$X = -S(t2)*C(t3+90)*482*C(t4)-S(t2)*-S(t3+90)*482*S(t4)-S(t2)*625*C(t3+90) = 312.4821495$$

$$Y = S(t3+90)*482*C(t4)+C(t3+90)*482*S(t4)+625*S(t3+90)+200+190 = 227.822663$$

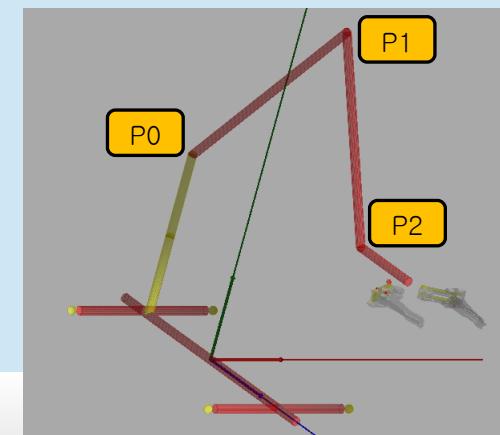
$$Z = -1*C(t2)*C(t3+90)*482*C(t4)-1*C(t2)*-S(t3+90)*482*S(t4)-1*C(t2)*625*C(t3+90)-246 = 295.2349594$$

# Inverse Kinematics (1/2)



$$\begin{aligned} a^2 &= b^2 + c^2 - 2bc \cos A \\ b^2 &= c^2 + a^2 - 2ca \cos B \\ c^2 &= a^2 + b^2 - 2ab \cos C \end{aligned}$$

$$\cos A = \frac{b^2 + c^2 - a^2}{2bc}$$



$$P0(0, H0 + H1, -W0)$$

$$P2(x, y, z)$$

$$|P0 \rightarrow P2| = \sqrt{x^2 + (y - H0 - H1)^2 + (z - W0)^2} \Rightarrow$$

$$a$$

$$b = L1$$

$$c = L0$$

$$f\Thetaeta4 = \arccos((b^2 + c^2 - a^2) / (2 * b * c))$$

$$T4 = 180 - f\Thetaeta4$$

# Inverse Kinematics (2/2)

$fT34=t3+t4$   
 $fT5=t5$

$H_0=190$   
 $H_1=200$   
 $H=H_0+H_1$   
 $W_0=-246$   
 $L_0=625$   
 $L_1=482$

$\text{depth}=z-W_0$   
 $\text{width}=x$   
 $t2=\text{atan}2(\text{depth},\text{width})$

$b=L_1$   
 $c=L_0$

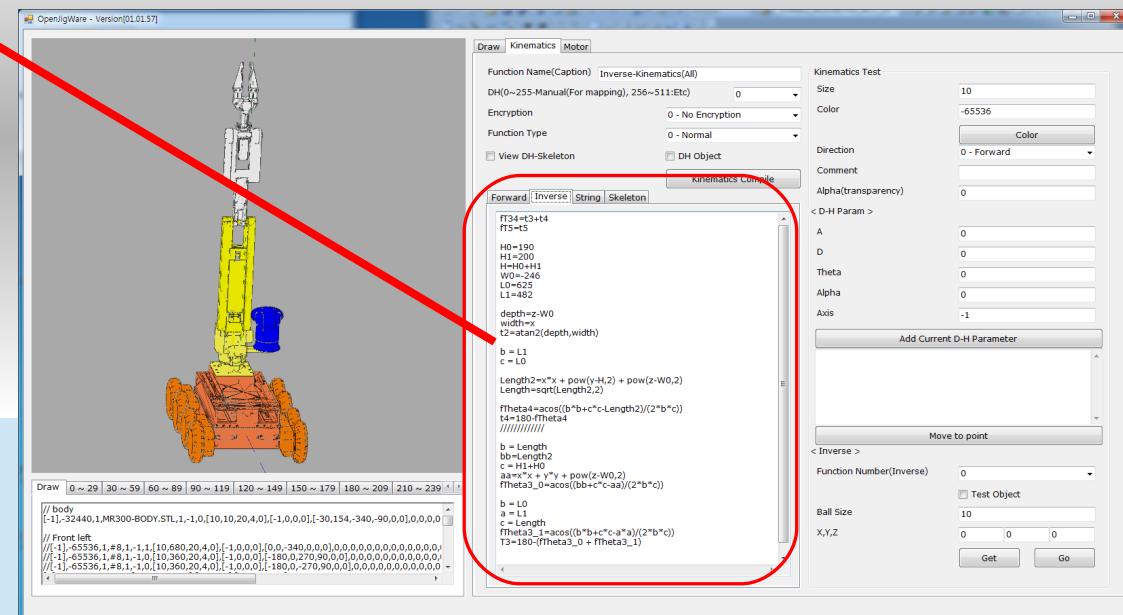
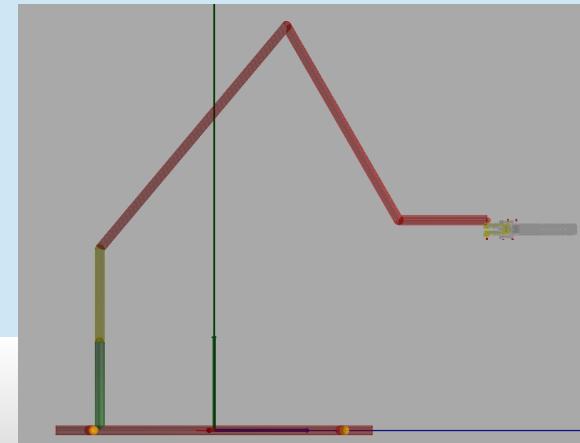
$\text{Length}_2=x^*x + \text{pow}(y-H,2) + \text{pow}(z-W_0,2)$   
 $\text{Length}=\sqrt{\text{Length}_2,2}$

$f\Theta_4=\text{acos}((b*c-a*a)/(2*b*c))$   
 $t4=180-f\Theta_4$   
 $//////////$

$b=\text{Length}$   
 $bb=\text{Length}_2$   
 $c=H_1+H_0$   
 $aa=x^*x + y^*y + \text{pow}(z-W_0,2)$   
 $f\Theta_3\_o=\text{acos}((bb+c*c-aa)/(2*b*c))$

$b=L_0$   
 $a=L_1$   
 $c=\text{Length}$   
 $f\Theta_{3\_1}=\text{acos}((b*c-a*a)/(2*b*c))$   
 $T3=180-(f\Theta_3\_o + f\Theta_{3\_1})$

$t5=fT5 + fT34 - t3 - t4$



- Result

<https://youtu.be/CWA8UQtodf8>

# 보행의 제작

- Gate / Sway / COG 외…

- 보행의 궤적 생성

- <https://youtu.be/zlp9k4NL1qo>

- 2족

- 휴머노이드 보행 자동 생성문서

- [http://www.homerobot.co.kr/kaist\\_soc/motis](http://www.homerobot.co.kr/kaist_soc/motis)

- <https://youtu.be/F8UgMhH9TXY>



- Enjoy this Walking Motion



<https://youtu.be/YIM-lIvMcaw>



# 모션제작 팁

- 캘리브레이션
  - 프로그램 다운로드

& 사용 동영상

<http://www.homerobot.co.k>



ation.zi

# 추가함수(Parallel Delta)

- Made by Donghyeon-Lee(0|동현) in DST Robot(@dstr  
obot.com)

- 오랫동안 꿈을 그리는 사람은 마침내 그 꿈을 닮아간다.
  - – 앙드레 말로
- A person longing for any dream for a long time resembles that dream at last.
  - -Andre Georges Malraux

**감사합니다.**

Dance

<https://youtu.be/DIboVjYI-Pg>



**Dongbu Robot**

Dance

[https://youtu.be/EzZl6u\\_HEfQ](https://youtu.be/EzZl6u_HEfQ)



**Dongbu Robot**

# Dance

<https://youtu.be/xfXejDcTBM4>



# Dance

<https://youtu.be/4Az1QcfQdhM>



# Dance

<https://youtu.be/w5jqQ5LecDM>



Dance

<https://youtu.be/bjH7cu9Tuxs>



**Dongbu Robot**

# 인도 무술

<https://youtu.be/rYg4peMuhJE>

