URDF

ROS မှာ Robot Model တစ်ခုကို Represent လုပ်ဖို့အတွက် XML Format နဲ့ URDF (Unified Robot Description Format) ဆိုတဲ့ Package တစ်ခုရှိတယ် ၊ သူ့ကို အသုံးပြုဖို့အတွက် urdf extension နဲ့ဆုံးတဲ့ File လေး တစ်ခုတည်ဆောက်ပေးရမယ် ၊ အဲ့ဒီ File လေးထဲမှာ Robot Model တစ်ခုအတွက်လိုအပ်တဲ့ အချက်အလက်မှန်သမျှပါပါတယ် ။

URDF ထဲမှာသုံးမယ့် အရေးကြီးတဲ့ Tag ၆ ခုရှိပါတယ် ။

1. robot tag (<robot>........................</robot>)
2. link tag (<link>.........................</link>)
3. joint tag (<joint>........................</joint>)
4. sensor tag (<sensor>.......................</sensor>)
5. transmission (<transmission>...............</transmission>)
6. gazebo (<gazebo>.......................</gazebo>)

အခုပြောပြမှာကတော့ simulation လုပ်တဲ့အခါ urdf file ထဲမှာ robot model တစ်ခုလုံးအတွက် လိုအပ်မယ့် tag လေးခုကို tag တစ်ခုချင်းစီ အသေးစိတ်ရှင်းပြပေးသွားမှာ ဖြစ်ပါတယ် ၊ သို့ပေမယ့် ဒီ pdf ထဲမှာ အရေးကြီးတဲ့ tag နှစ်ခုတော့ မပါဝင်ပါဘူး ၊ transmission tag နဲ့ gazebo tag ပါ ။ အဲ့ဒီ tag နှစ်ခုက မိမိတို့ robot model ကို simulation လုပ်တဲ့အခါ control လုပ်မယ့် tag တွေ ဖြစ်တာကြောင့် မထည့်ထားပါဘူး (ကျနော်လည်း သေချာမလေ့လာရသေးလို့) ။

ဒီ pdf ရဲ့ ရည်ရွယ်ချက်က robot model ကို control လုပ်တာမပါဘဲ robot model သီးသန့်ပါ ။

tag တွေအကြောင်း စမပြောခင် ကြိုသိထားရမှာက Tag တစ်ခုမှာ attribute နဲ့ element ဆိုပီး အချက်နှစ်ချက် ရှိတယ်ဆိုတာ သိထားပေးပါ ။

Attribute ကတော့ ဒီ tag ရဲ့ characteristic ဖြစ်ပါတယ် ၊ Elements ဆိုတာကတော့ ဒီ Tag ထဲမှာရှိနိုင်တဲ့ တခြားသော Tag တွေကိုဆိုလိုခြင်းဖြစ်ပါတယ် ။

1 - robot tag

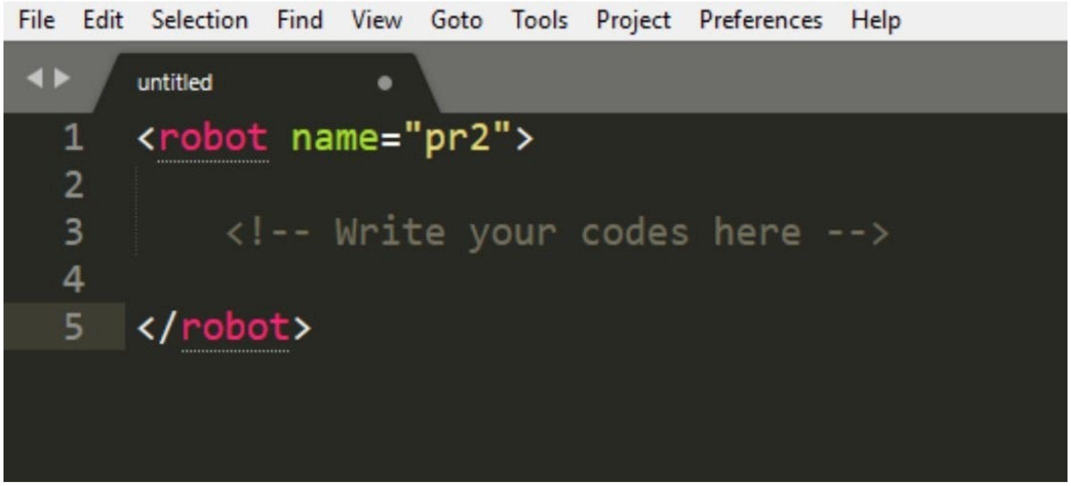
Syntax - <robot>................</robot>

ဒီ Tag နှစ်ခုအတွင်းမှာ robot အတွက် လိုအပ်တဲ့ properties တွေအကုန်ဖော်ပြရပါတယ် ။ သူကတော့ root tag ကြီးပါ ၊ တခြားသော tag တွေလည်း သူ့ထဲမှာ ရေးရပါတယ် ။

Attributes of robot tag

1. name

name ဆိုတာကတော့ မိမိ robot ရဲ့ နာမည်ပါ - ဥပမာအနေနဲ့ အောက်ကပုံလေးမှာ ကြည့်ပါ ။



အထက်ပါပုံမှာဆိုရင် robot name က pr2 ဖြစ်ပါတယ်

Elements of robot tag

link tag (<link>........................</link>)

joint tag (<joint>.......................</joint>)

transmission tag (<transmission>................</transmission>)

gazebo tag (<gazebo>......................</gazebo>)

robot tag ရဲ့ elements တွေကိုတော့ အောက်က tag တစ်ခုချင်းစီမှာ အသေးစိတ်ကြည့်ပါ

2 - link tag

Syntax - <link>................</link>

ဒီ tag နှစ်ခုအတွင်းမှာတော့ robot link တစ်ခုရဲ့ kinematic နဲ့ dynamic properties တွေကို ဖော်ပြရာမှာ သုံးပါတယ် ၊ အောက်မှာဆက်ကြည့်ရင် ပိုရှင်းသွားပါလိမ့်မယ် ။

Attributes of link tag

1 – name (required)

name ဆိုတာကတော့ မိမိ robot ရဲ့ link နာမည်ပါ - ဥပမာအနေနဲ့ အောက်ကပုံလေးမှာ ကြည့်ပါ ။

အထက်ပါပုံမှာဆိုရင် link name က first_link ဖြစ်ပါတယ်

Elements of link tag

(i) - <visual>..................</visual> tag

(ii) - <collision>................</collision> tag

(iii) - <inertial>................</inertial> tag

Tag တစ်ခုချင်းစီကို ဆက်ရှင်းပါမယ်

(i) <visual>.............</visual>

ဒီ Tag မှာတော့ robot ရဲ့ real link ကို ဖော်ပြပါတယ် ၊ ကျနော်တို့ Gazebo ထဲမှာ simulation လုပ်တဲ့အခါ Rviz ထဲမှာ visualization လုပ်တဲ့အခါမှာ ဒီ tag ထဲမှာရေးထားတဲ့အတိုင်းမြင်ရမှာပါ။သူ့မှာလည်း sub-tag လေးတွေရှိပါတယ်

1- <origin> - ဒီ sub-tag မှာတော့ ကိုယ်ဖန်တီးတဲ့ robot link ကို initial pose (position+orientation) ဘယ်နေရာမှာထားမယ်ဆိုတာတွေကို ဒီ Tag အတွင်းမှာရေးရပါတယ် ၊ တနည်းအားဖြင့် link`s reference frame ရဲ့ pose ပေါ့ ။ position အတွက် xyz ဆိုတဲ့ parameter နဲ့ orientation အတွက် rpy ဆိုတဲ့ parameter လေးတွေရှိပါတယ် ။ ဒီ origin sub-tag လေးကတော့ optional ပါ ၊ သဘောကတော့ ထည့်ထည့်မထည့်ထည့် ရတယ်ပေါ့ဗျာ ၊ မထည့်ဖူးဆိုရင် default က identity matrix ပါပဲ ။

2 - <geometry>...............</geometry> - ဒီ sub-tag မှာတော့ ကိုယ်ဖန်တီးချင်တဲ့ link ရဲ့ shape(box,cylinder,sphere,etc...) ကို ရေးပေးရပါတယ် ၊ mesh file လည်း ထည့်ပေးလို့ရပါတယ် ၊ ထည့်ပေးဖို့ အတွက် tag လေး သေးသေးကလေး ၄ ခုရှိပါတယ် (<box>,<cylinder>,<sphere>,<mesh>) ။ ဒီ Tag က required ပါ ၊ geometry sub-tag ထဲမှာ မဖြစ်မနေ ထည့်ပေးရပါတယ် ။

3 - <material>..............</material> - ဒီ sub-tag မှာတော့ ကိုယ်က ဒီ robot link ကို ဘယ်လို material နဲ့ ဖန်တီးချင်လဲဆိုတာ ရေးပေးရပါတယ် ၊ material ရဲ့ နာမည်ကိုဖော်ပြဖို့ name parameter ၊ material ရဲ့ အရောင်ကိုဖော်ပြဖို့အတွက် color ဆိုတဲ့ tag လေး သေးသေး ကလေးထဲမှာ rgba ဆိုတဲ့ parameter နဲ့ material ရဲ့ texture ကို ဖော်ပြဖို့အတွက် texture ဆိုတဲ့ tag လေး သေးသေးလေးထဲမှာ texture file ကို ထည့်ပေးလို့ ရပါတယ် ၊ ဒီ material sub-tag လေးကတော့ optional ပါ ။

ပြီးတော့ visual tag မှာ name ဆိုတဲ့ parameter လေးလည်းရှိပါတယ် ၊ link ရဲ့ geometry name ပါ ၊ သုံးတာတော့ သိပ်မတွေ့ဖူးဘူးဗျ ။ သူလည်း optional ပါ ။

အောက်က ပုံလေးမှာကြည့်ပါ ။

untitled • - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

untitled

```
<link name="first_link">

    <visual>

        <origin xyz="0 0 0" rpy="0 0 0 " />

        <geometry>
            <box size="1 1 3" />
        </geometry>

        <material name="Red">
            <color rgba="1.0 0 0 1.0" />
        </material>

    </visual>

</link>
```

အထက်ပါပုံမှာ အလျား 1m ၊ အနံ 1m အမြင့် 3m ရှိတဲ့ စတုရန်းပုံစံ link လေးတစ်ခုကို initial position(x=0,y=0,z=0) နဲ့ initial orientation(row=0,pitch=0,angle=0) ရှိတဲ့ reference frame တစ်ခုမှာ နေရာချထားပါတယ် ။ material name ကတော့ Red လို့ပေးထားပြီး color ကတော့ အနီရောင် ပါ ။

(ii) - <collision>.................</collision>

ဒီ Tag ကတော့ collision checking အတွက် အသုံးပြုပါတယ် ၊ သဘောကတော့ visual tag ကနေ ဖန်တီးထားတဲ့ real link ကို တခြား object တစ်ခုခုနဲ့ မတိုက်မိအောင် သို့ link အချင်းချင်း self-collision မဖြစ်အောင် detect လုပ်ပေးတာပါ ။ မြင်အောင်ပြောရရင် အကာအုပ်ပေးတဲ့သဘောပါပဲ - ကြက်ဥပြုတ်နဲ့ ဥပမာပေးရမယ်ဆို ပိုသိသာမယ်ထင်တယ် ၊ visual tag က ဖန်တီးထားတဲ့ real link က ကြက်ဥပြုတ်ရဲ့ အနှစ်ဖြစ်ပြီး collision tag က ဖန်တီးထားတဲ့ link က ကြက်ဥပြုတ်ရဲ့ အကာပေါ့ ။ သူ့မှာလည်း sub-tag လေးတွေရှိပါတယ်။

1 - <geometry>...............</geometry> ဒီ sub-tag မှာတော့ အပေါ်က visual tag မှာရှိတဲ့ geometry sub-tag လေးနဲ့ တူတူပါပဲ - value တွေလည်း တူတူပေးလို့လည်းရပါတယ် ။

2 - <origin> ဒီ sub-tag မှာလည်း အပေါ်က visual tag မှာရှိတဲ့ origin sub-tag လေးနဲ့ တူတူပါပဲ - value တွေလည်း တူတူပေးလို့လည်းရပါတယ် ။

ပြီးတော့ collision tag မှာလည်း name ဆိုတဲ့ parameter လေးရှိပါတယ် ၊ link ရဲ့ geometry name ပါ ၊ သုံးတာတော့ သိပ်မတွေ့ဖူးဘူးဗျ ။ သူလည်း optional ပါ ။

အောက်က ပုံလေးမှာကြည့်ပါ ။

```
<link name="first_link">

    <collision>

        <origin xyz="0 0 0" rpy="0 0 0 " />

        <geometry>
            <box size="1 1 3" />
        </geometry>

    </collision>

</link>
```

အထက်ပါပုံမှာ အလျား 1m ၊ အနံ 1m အမြင့် 3m ရှိတဲ့ စတုရန်းပုံစံ collision object လေးတစ်ခုကို initial position(x=0,y=0,z=0) နဲ့ initial orientation(row=0,pitch=0,angle=0) ရှိတဲ့ reference frame တစ်ခုမှာ နေရာချထားပါတယ် ။

(iii) - <inertial>................</inertial>

ဒီ Tag ကတော့ robot link ရဲ့ dynamic properties တွေထဲက တစ်ခုဖြစ်တဲ့ inertial properties အတွက် အသုံးပြုပါတယ် ၊ dynamics ကိုသေချာ မလေ့လာဖူးသေးလို့ မပြောပြနိုင်ပေမယ့် code ထဲမှာ ရေးထားသလောက်ကိုတော့ပြောပြပါမယ် ၊ သူ့ sub-tag လေးတွေအကြောင်း ပြောကြရအောင်

1 - <origin> ဒီ sub-tag မှာတော့ အပေါ်က visual tag နဲ့ collision tag မှာရှိတဲ့ origin sub-tag လေးနဲ့ တူတူပါပဲ - value တွေလည်း တူတူပေးလို့လည်းရပါတယ် ။

2 - <mass> ဒီ sub-tag မှာတော့ မိမိ robot link ရဲ့ mass ကို value တစ်ခု ရေးပေးရပါတယ်။ ဒီမှာတော့ mass က inertia အတွက် ဘယ်လိုအရေးပါကြောင်း အသေးစိတ်မရှင်းတော့ပါဘူး ၊ သေချာလေးသိချင်တယ်ဆို ဒီ link ထဲသွားကြည့်လိုက်ပါ (https://www.youtube.com/watch?v=1XSyyjcEHo0)

3 - <inertia> ဒီ sub-tag မှာတော့ link ရဲ့ shape ကို inertia frame ထဲမှာ 3x3 rotational inertia matrix အနေနဲ့ ဖော်ပြတာပါ ၊ အောက်က matrix ကို ကြည့်နိုင်ပါတယ် ၊ 3x3 matrix ဖြစ်လို့ Elements ကိုးခုပေမယ့်

ကိုးခုလုံးမသုံးဘဲ Diagonal elements ရဲ့ အပေါ်ခြမ်းကိုပဲ သုံးပါတယ် ဘာကြောင့်လဲဆိုတော့ rotational inertia matrix က symmetric ဖြစ်လို့ပါ ။

ixx	ixy	ixz
ixy	iyy	iyz
ixz	iyz	izz

ဒီ matrix အတိုင်းဆို ixx,ixy,ixz,iyy,iyz,izz - ဒီ ခြောက်ခုပေါ့ ၊ ဒီ matrix ခြောက်ခုကို value သတ်မှတ်တာနဲ့ ပက်သက်ပြီး shape က ပြောင်းလဲ သွားတာပါ ၊ ဒီ link မှာ matrix အလိုက် shape အမျိုးမျိုးသွားကြည့်လို့ရပါတယ်

(https://en.wikipedia.org/wiki/List_of_moments_of_inertia#List_of_3D_inertia_tensors)

အောက်က ပုံလေးမှာကြည့်ပါ ။

```
<link name="first_link">

    <inertial>

        <origin xyz="0 0 0.5" rpy="0 0 0"/>

        <mass value="1"/>

        <inertia ixx="1.0" ixy="0.0" ixz="0.0"
                           iyy="0.5" iyz="0.0"
                                     izz="1.0"/>

    </inertial>

</link>
```

ပြီးတော့ inertial tag ကို သေချာလေး ရှင်းမပြနိုင်တာ Sorry ပါဗျာ - ကျနော် လောလောဆယ် ဒီလောက်ပဲ နားလည်လို့ပါ

အပေါ်မှာ ရှင်းပြခဲ့တဲ့ **link tag** တစ်ခုလုံးကို ရှင်းရှင်းလင်းလင်း ဒီ အောက်က example မှာ ကြည့်နိုင်ပါတယ် ။

untitled • - Sublime Text (UNREGISTERED)

```
<link name="first_link">

    <visual>
     <origin xyz="0 0 0" rpy="0 0 0" />
     <geometry>
       <box size="1 1 3" />
     </geometry>
     <material name="Red">
       <color rgba="1.0 0 0 1.0"/>
     </material>
   </visual>

   <collision>
     <origin xyz="0 0 0" rpy="0 0 0"/>
     <geometry>
       <box size="1 1 3" />
     </geometry>
   </collision>

   <inertial>
     <origin xyz="0 0 0.5" rpy="0 0 0"/>
     <mass value="1"/>
     <inertia ixx="1.0" ixy="0.0" ixz="0.0"
                        iyy="0.5" iyz="0.0"
                                  izz="1.0" />
   </inertial>

 </link>
```

3 – Joint Tag

Syntax - <joint>...........................</joint>

ပထမဦးစား ကြိုသိရမှာက Joint ဆိုတာက link နှစ်ခုကို ဆက်ပေးတဲ့အရာပါ ၊ ပိုရှင်းအောင်ပြောရရင် parent link နဲ့ child link ကိုဆက်ပေးတဲ့အရာပေါ့ ။ ဒီလိုပဲ စစချင်း အလွယ်မှတ်နိုင်ပါတယ် ။

ဒီ tag နှစ်ခုအတွင်းမှာတော့ robot joint တစ်ခုရဲ့ kinematic နဲ့ dynamic properties တွေကို ဖော်ပြရာမှာ သုံးပါတယ် ၊

Joint တစ်ခုရဲ့ safety limits ကိုလည်း သတ်မှတ်ပေးလို့ရပါတယ် ။ အောက်မှာဆက်ကြည့်ရင် ပိုရှင်းသွားပါလိမ့်မယ် ။

Attributes of joint tag

1 – name (required)

name ဆိုတာကတော့ မိမိ robot ရဲ့ Joint နာမည်ပါ - ဥပမာအနေနဲ့ အောက်ကပုံလေးမှာ ကြည့်ပါ ။

2 – type (required)

type ဆိုတာကတော့ မိမိ robot ရဲ့ Joint အမျိုးအစားပါ ။ Joint အမျိုးအစားတွေကတော့ revolute , continuous, prismatic, fixed , floating , planar တို့ဖြစ်ပါတယ် ။ ဥပမာအနေနဲ့ အောက်ကပုံလေးမှာ ကြည့်ပါ ။

အထက်ပါပုံမှာဆိုရင် joint name က first_joint ဖြစ်ပါတယ် ။ Joint type က revolute ပါ ။

Elements of Joint tag

(i) - <origin> tag

(ii) - <parent> tag

(iii) - <child> tag

(iv) - <axis> tag

(v) - <limit> tag

(vi) - <dynamics> tag

(vii) - < mimic > tag

(viii) - < safety_controller > tag

(ix) - < calibration > tag

Tag တစ်ခုချင်းစီကို ဆက်ရှင်းပါမယ်

(i) - <origin> Tag (optional)

ဒီ Tag မှာတော့ ကိုယ်ဖန်တီးတဲ့ robot joint ကို initial pose (position+orientation) ဘယ်နေရာမှာထားမယ်ဆိုတာတွေကို ဒီ Tag အတွင်းမှာရေးရပါတယ် ၊ တနည်းအားဖြင့် joint`s reference frame ရဲ့ pose ပေါ့ ။ position အတွက် xyz ဆိုတဲ့ parameter နဲ့ orientation အတွက် rpy ဆိုတဲ့ parameter လေးတွေရှိပါတယ် ။

(ii) - <parent> Tag (required)

ဒီ Tag မှာတော့ Joint မတိုင်ခင် လာမယ့် parent link ရဲ့ နာမည်ကို ရေးရပါတယ် ၊ name ဆိုတဲ့ parameter တစ်ခုရှိပါတယ် ။

(iii) - <child> Tag (required)

ဒီ Tag မှာတော့ Joint ပြီးရင်လာမယ့် child link ရဲ့ နာမည်ကို ရေးရပါတယ် ၊ name ဆိုတဲ့ parameter တစ်ခုရှိပါတယ် ။

(iv) - <axis> Tag (optional)

ဒီ Tag မှာတော့ မိမိဖန်တီးထားတဲ့ Joint ရဲ့ axis ကို သတ်မှတ်တာပါ ။ xyz ဆိုတဲ့ parameter လေးတစ်ခုရှိပါတယ် ၊ မိမိ actuate လုပ်စေချင်တဲ့ axis နေရာမှာ 1 လို့ရေးပေးပြီး ကျန် axis နေရမှာ 0 လို့ ရေးပေးရမှာပါ ။Default အနေနဲ့ကတော့ (1,0,0) ပါ ၊ x axis မှာ rotation or translation ဖြစ်မယ်ပေါ့ ။

ရှင်းအောင်ထပ်ပြောရရင်

- revolute joint type အတွက်ဆို axis of rotation ပေါ့
- prismatic joint type အတွက်ဆို axis of translation ဖြစ်ပြီး
- planar joint type အတွက်ဆိုရင်တော့ surface normal ပါ ၊
- တစ်ခု သတိထားရမှာက fixed နဲ့ floating joint types တွေမှာတော့ ဒီ axis tag ကို မသုံးပါဘူး ။

အကယ်၍ ကိုယ်က revolute , prismatic နဲ့ planar joint types တွေ အတွက် axis tag ကို သုံးမယ်ဆိုရင်တော့ ဒီ xyz parameter လေးက required ဆိုတာ သဘောပေါက်မယ်ထင်ပါတယ် ။

(v) - <limit> Tag (required for prismatic and revolute)

ဒီ tag ကတော့ joint ရဲ့ limit values , force နဲ့ velocity ကို ဖော်ပြချင်တဲ့အခါ ရေးရပါတယ် ၊ အဲ့လို ဖော်ပြဖို့အတွက် parameter လေးခု ရှိပါတယ် ၊ အောက်မှ ကြည့်ရင် ပိုရှင်းပါလိမ့်မယ် ။ lower ၊ upper ၊ effort နဲ့ velocity တို့ ဖြစ်ပါတယ် ။ သူ့ကို revolute နဲ့ prismatic joint types တွေအတွက်သာ သုံးပါတယ် ၊ တခြား joint types တွေအတွက် မသုံးပါ ။

- lower parameter (optional , default = 0) - ဒီ parameter ကတော့ joint ရဲ့ အနိမ့်ဆုံးရှိနိုင်တဲ့ joint limit value ပါ (revolute type အတွက်ဆို radians unit နဲ့ယူရတာဖြစ်ပြီး prismatic type အတွက်ဆို meter unit နဲ့ယူရပါတယ်) ။
- upper parameter(optional , default = 0) - ဒီ parameter ကတော့ joint ရဲ့ အမြင့်ဆုံးရှိနိုင်တဲ့ joint limit value ပါ (revolute type အတွက်ဆို radians unit နဲ့ယူရတာဖြစ်ပြီး prismatic type အတွက်ဆို meter unit နဲ့ယူရပါတယ်) ။
- effort parameter(required) - ဒီ parameter ကတော့ joint ရဲ့ force or torque values ပါ ၊ ဆိုလိုချင်တာက prismatic type အတွက်ဆို force value ဖြစ်ပြီး revolute type အတွက်ဆို torque value ပါ ။ အကယ်၍ ကိုယ်က limit tag ကို သုံးမယ်ဆိုရင်တော့ ဒီ effort parameter လေးက *required* ဆိုတာ သဘောပေါက်မယ်ထင်ပါတယ် ။
- velocity parameter (required) - ဒီ parameter ကတော့ joint ရဲ့ အမြင့်ဆုံး ရှိနိုင်တဲ့ velocity ပေါ့ ။ ဆိုလိုချင်တာက prismatic type အတွက်ဆို maximum linear velocity ဖြစ်ပြီး revolute type အတွက်ဆို maximum angular velocity ပေါ့ ။ ။ အကယ်၍ ကိုယ်က limit tag ကို သုံးမယ်ဆိုရင်တော့ ဒီ velocity parameter လေးက *required* ဆိုတာ သဘောပေါက်မယ်ထင်ပါတယ် ။

(vi) - <dynamics> Tag (optional)

ဒီ Tag ကတော့ joint တစ်ခုရဲ့ dynamics နဲ့ဆိုင်တဲ့ physical properties တွေကို သုံးချင်တဲ့အခါ ရေးရပါတယ် ၊ သူ့မှာ parameter နှစ်ခုရှိပါတယ် ၊ damping parameter နဲ့ friction parameter ပါ ။ အောက်မှ အသေးစိတ် ဆက်ရှင်းပါမယ်

- **damping parameter (optional , default = 0)** - ဒီ parameter ကတော့ Joint ရဲ့ physical damping value ကို သုံးချင်တဲ့အခါ ရေးရပါတယ် ၊ damping value ဆိုတာ တနည်းအားဖြင့် velocity ပေါ်မူတည်ပြီး ဖြစ်ပေါ်နေတဲ့ resistive force တစ်ခုပါပဲ ၊ velocity က resistive force နဲ့ opposite direction ပါ ။ ပြောချင်တာက joint တစ်ခု မြန်မြန်ရွေ့လျားရင် damping value နည်းနေခြင်းဖြစ်ပြီး ၊ joint တစ်ခု နှေးနှေးလေး ရွေ့လျားနေရင် damping value များနေတယ်လို့ ဆိုလိုတာပါ (heavy damping လို့ခေါ်ပါတယ်) ။ အလွယ်မှတ်ချင်ရင်တော့ joint တစ်ခု မြန်မြန် ရွေ့လျားလား မရွေ့လျားလားဆိုတာ damping value အပေါ် မူတည်တယ်ပေါ့ဗျာ ။ ကျနော်လည်း ဒီလိုနားလည်ထားလို့ ဒီလောက်ပဲ ရှင်းအောင် ပြောပြနိုင်ပါတယ်ဗျာ - ဒီ link နှစ်ခုကြောင့် နားလည်ခဲ့တာပါ (နားလည်တာ မှားရင်လည်း မှားမှာပေါ့)
(https://robotics.stackexchange.com/questions/10029/damping-vs-friction)
(https://www.youtube.com/watch?v=y0YFw9ZzSyM)

- **friction parameter (optional , default = 0)** - ဒီ parameter ကတော့ Joint ရဲ့ physical static friction value ကို သုံးချင်တဲ့အခါ ရေးရပါတယ်၊ Static friction ဆိုတာကတော့ motion တစ်ခုစတင်ဖြစ်ပေါ်မှုကို resist လုပ်ထားတဲ့ force တစ်ခုပါပဲ ၊ ကျနော်တို့ ကိုးတန်း ဆယ်တန်းမှာ နားလည်ပြီးသားတွေပါ ။ ဆက်ရှင်းမပြတော့ဖူးနော် ၊ အကယ်၍ မေ့နေတယ်ဆို ဒီ link မှာ သွားနွေးလို့ရပါတယ် (https://www.youtube.com/watch?v=3EbUa5ZDybg&fbclid=IwAR2wHMogVK-gLbrunX0FGblClcQKdPTek2eEJ4LsRhhxfbTT0KESN0QOKQI)

(vii) - <mimic> Tag (optional)

ဒီ tag ကတော့ မိမိ အခုအသစ်ဖန်တီးမယ့် joint ကို ယခင်ရှိပြီးသား existing joint ရဲ့ value အတိုင်းဖြစ်အောင် သတ်မှတ်ချင်ရင် သုံးရပါတယ် ။ joint ရဲ့ value ကို အောက်ပါ formula နဲ့ တွက်နိုင်ပါတယ် ။

*value = multiplier * other_joint_value + offset*

joint value ကိုအထက်ပါ formula အတိုင်းတွက်ချက်ဖို့အတွက် parameter သုံးခု ရှိပြီးသားပါ ။ အောက်မှာ ဆက်ကြည့်ပါ

- **joint parameter (required)** - ဒီ parameter ကတော့ ကိုယ် mimic လုပ်မယ့် existing joint ရဲ့ နာမည်ကို ရေးရပါတယ် ။
- **multiplier parameter (optional)** - ဒီ parameter ကတော့ အထက်ပါ formula အတိုင်း ကိုယ်လိုချင်တဲ့ joint value ဖြစ်အောင် multiplicative factor သတ်မှတ်တာပါ ။
- **offset parameter(optional)** - ဒီ parameter ကတော့ အထက်ပါ formula အတိုင်း joint offset ကို သတ်မှတ်တာပါ ၊ မိမိ အသစ်ဖန်တီးမယ့် joint နဲ့ မူလရှိပြီးသား joint ကြားက offset ပါ ။

(viii) - <safety controller> Tag (optional)

ဒီ Tag မှာတော့ safety controller ကနေ joint ရဲ့ position နဲ့ velocity ကို safety ဖြစ်ချင်ရင် သတ်မှတ်ပေးရပါတယ် ။ သဘောကတော့ joint က သူသတ်မှတ်တဲ့ limit value ထဲမှာရှိရင် error မရှိနိုင်ဖူးပေါ့ ၊ safe ဖြစ်အောင် boundary သတ်မှတ်တဲ့သဘောပါ ။ parameter လေးခုရှိပါတယ် ၊ အောက်မှာဆက်ကြည့်ပါ ။

- **soft_lower_limit parameter (optional) –** ဒီ parameter ကတော့ joint position ကို safe ဖြစ်စေမယ့် lower joint boundary သတ်မှတ်တာပါ ၊ သူက limit tag ကနေ သတ်မှတ်ထားတဲ့ lower parameter ရဲ့ value ထက် ပိုကြီးပေးရပါမယ် ၊ မရှုပ်သွားပါနဲ့ ။ limit tag ကနေ သတ်မှတ်တဲ့ lower parameter က joint အနိမ့်ဆုံးရှိနိုင်တဲ့ limit ပါ ၊ သူ့ထက် နည်းတာနဲ့ error ဖြစ်ပါပြီ ၊ အခု tag ရဲ့ parameter က safe ဖြစ်စေမယ့် boundary ပါ ။

- **soft_upper_limit parameter (optional) –** ဒီ parameter ကတော့ joint position ကို safe ဖြစ်စေမယ့် upper joint boundary သတ်မှတ်တာပါ ၊ သူက limit tag ကနေ သတ်မှတ်ထားတဲ့ upper parameter ရဲ့ value ထက် ပိုငယ် ပေးရပါမယ် ၊ မရှုပ်သွားပါနဲ့ ။ limit tag ကနေ သတ်မှတ်တဲ့ upper parameter က joint အမြင့်ဆုံးရှိနိုင်တဲ့ limit ပါ ၊ သူ့ထက် ကြီးတာနဲ့ error ဖြစ်ပါပြီ ၊ အခု tag ရဲ့ parameter က safe ဖြစ်စေမယ့် boundary ပါ ။

- **k_position parameter (optional) –** ဒီ parameter ကတော့ position ကို safety limit ကနေ ကျော်မသွားအောင် boundary ဝင်အောင် velocity limit ကို scale လုပ်ပေးတဲ့အခါ ရေးရပါတယ်။

- **k_velocity parameter (optional) –** ဒီ parameter ကတော့ velocity ကို safety limit ကနေ ကျော်မသွားအောင် boundary ဝင်အောင် effort limit ကို scale လုပ်ပေးတဲ့အခါ ရေးရပါတယ်။

ဒီ Parameter လေးခုကို ပိုရှင်းသွားအောင် အောက်က ပုံနှစ်ပုံမှာ ကြည့်နိုင်ပါတယ် ။

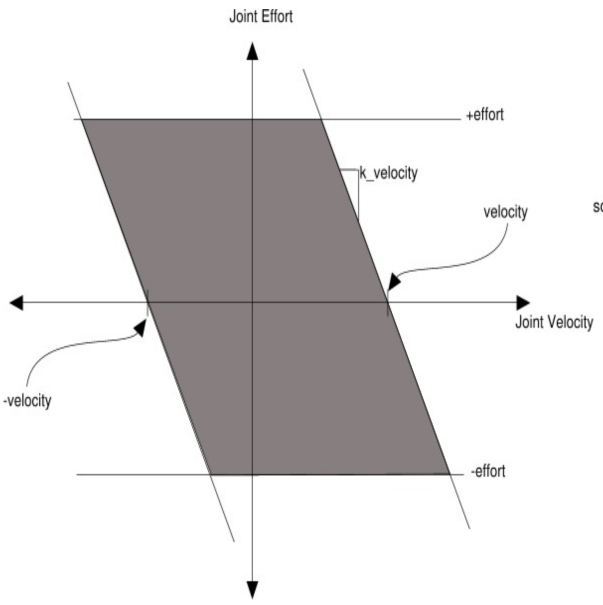


Fig : k_velocity

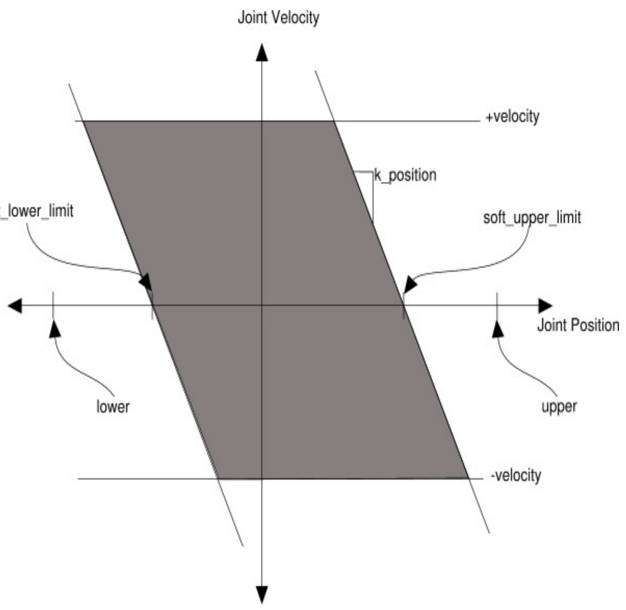


Fig : k_position

(viii) - <calibration> Tag (optional)

ဒီ Tag ကို ကျနော် သေချာကြီး နားမလည်ဘူးဗျ ၊ ခု ကျနော်နားလည်ထားသလောက်ပြောရရင် joint accuracy ကို improve ဖြစ်အောင် ရေးချင်တဲ့အခါ သုံးတယ်လို့ပဲ သိပါတယ် ၊ သူ့မှာ rising နဲ့ falling ဆိုပြီး parameter နှစ်ခုရှိပါတယ် ။ ကျနော်လည်း နားမလည်တော့ ရမ်းမရှိးချင်ပါဘူး ၊ ros wiki မှာ တော့ ဒီလိုလေးရေးထားပါတယ် ။

<calibration> *(optional)*

The reference positions of the joint, used to calibrate the absolute position of the joint.

rising *(optional)*

When the joint moves in a positive direction, this reference position will trigger a rising edge.

falling *(optional)*

When the joint moves in a positive direction, this reference position will trigger a falling edge.

အပေါ်မှာ ရှင်းပြခဲ့တဲ့ **joint tag** တစ်ခုလုံးကို ရှင်းရှင်းလင်းလင်း ဒီ အောက်က example မှာ ကြည့်နိုင်ပါတယ် ။

untitled • - Sublime Text (UNREGISTERED)

File Edit Selection Find View Goto Tools Project Preferences Help

untitled

```
<joint name="first_joint" type="revolute">

    <origin xyz="0 0 1" rpy="0 0 3.1416"/>
    <parent link="first_link"/>
    <child link="second_link"/>
    <axis xyz="0 0 1" />

    <calibration rising="0.0"/>
    <dynamics damping="0.0" friction="0.0"/>
    <limit effort="30" velocity="1.0" lower="-2.2" upper="0.7" />
    <safety_controller k_velocity="10" k_position="15" soft_lower_limit="-2.0" soft_upper_limit="0.5" />

 </joint>
```

4 – sensor tag

Syntax - <sensor>......................</sensor>

ဒီ Tag နှစ်ခုအတွင်းမှာတော့ မိမိသုံးမယ့် sensor နဲ့ပက်သက်တဲ့ properties တွေ information တွေကို ရေးရပါတယ် ။ အောက်မှာဆက်ကြည့်ရင် ပိုရှင်းသွားပါလိမ့်မယ် ။ မိမိသုံးမယ့် sensor အပေါ်မူတည်ပြီး element တွေ မတူညီပါဘူး ။ ဒီ link မှာသွားကြည့်နိုင်ပါတယ် (http://wiki.ros.org/urdf/XML/sensor/proposals) ၊ ကျနော်ကတော့ camera sensor ကို ဥပမာပေးပြီး ပြောမှာဖြစ်တာကြောင့် camera tag ကိုပဲ သုံးထားပါတယ် ။

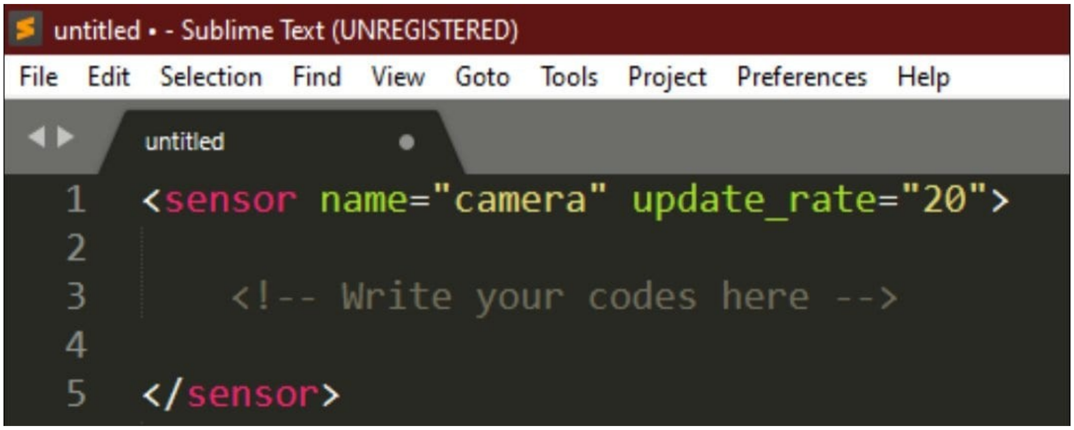
Attribute of sensor tag

1 – name (required)

name ဆိုတာကတော့ မိမိ robot မှာ တပ်ဆင်မယ့် sensor နာမည်ပါ

2 – update_rate (optional)

update_rate ဆိုတာကတော့ မိမိ sensor က ဘယ်လောက် frequency မှာ sensor data တွေကို generate လုပ်မလဲဆိုတာ သတ်မှတ်ပေးရတာပါ ။ frequency ဖြစ်လို့ unit က hz ပါ ။ ဒါကြောင့် ဘယ်နှစက္ကန့်တစ်ခါ generate လုပ်မလဲဆိုတာကိုသိချင်ရင် ဒီ formula (t = 1/f) ထဲထည့်ပြီး တွက်နိုင်ပါတယ် ။



အထက်ပါ ပုံအရဆို မိမိသုံးမယ့် sensor က camera ဖြစ်တာကြောင့် နာမည်ကို camera လို့ပေးထားပြီး 20hz တစ်ခါ data တွေကို generate လုပ်ပေးနေမှာပါ ။

Elements of sensor tag

(i) - <parent> tag

(ii) - <origin> tag

(iii) - <camera>.............</camera> tag

Tag တစ်ခုချင်းစီကို ဆက်ရှင်းပါမယ် ။

(i) - <parent> Tag (required)

ဒီ tag မှာတော့ မိမိ သုံးမယ့် sensor ကို ဘယ် link မှာ တပ်ဆင်မယ်ဆိုတာကို link name လေးရေးပေးရမှာပါ ။ link parameter လေးတစ်ခုရှိပါတယ် ။ parameter လေးက required ပါ ။

(ii) - <origin> Tag (optional)

ဒီ Tag မှာတော့ မိမိ သုံးမယ့် sensor ကို initial pose (position+orientation) ဘယ်နေရာမှာ ထားမယ်ဆိုတာတွေကို ဒီ Tag အတွင်းမှာရေးရပါတယ် ၊ တနည်းအားဖြင့် sensor`s reference frame ရဲ့ pose ပေါ့ ။ position အတွက် xyz ဆိုတဲ့ parameter နဲ့ orientation အတွက် rpy ဆိုတဲ့ parameter လေးတွေရှိပါတယ် ။ frame ယူတဲ့နေရာမှာ သတိထားရမှာက sensor ဖြစ်တဲ့အတွက်ကြောင့် z axis ကို forward ၊ x axis ကို right ဖြစ်ပြီး y axis ကတော့ down ဆိုပြီး ယူပေးရမှာပါ ။

(ii) - <camera>...........</camera> Tag (optional)

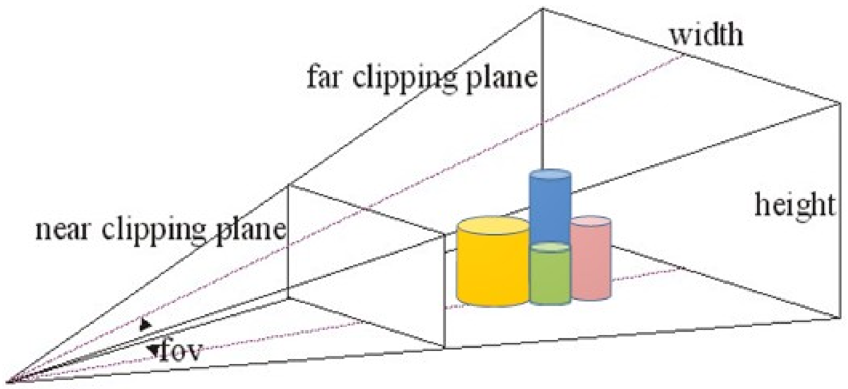
အကယ်၍ မိမိက camera sensor ကိုသုံးမယ်ဆိုရင် ဒီ Tag လေးအတွင်းမှာ image နဲ့ ဆိုင်တဲ့ information တွေကို image tag လေးအတွင်းမှာရေးပေးရမှာပါ ။ parameter အနေနဲ့ ၆ ခုရှိပါတယ် ။ အောက်မှ အသေးစိတ်ဆက်ကြည့်ပါ ။

မိမိဖန်တီးထားတဲ့ camera sensor ကို တကယ့် real sensor လိုမျိုး အသုံးပြုနိုင်ဖို့ plugin file တစ်ခုထည့်ဖို့လိုတယ်ဆိုတာ သတိချပ်စေချင်ပါတယ် ။

- **width parameter (required) –** ဒီ parameter ကတော့ image တစ်ခုရဲ့ pixels width ကိုရေးပေးရမှာပါ ။
- **height parameter (required) –** ဒီ parameter ကတော့ image တစ်ခုရဲ့ pixels height ကိုရေးပေးရမှာပါ ။
- **format parameter (required) –** ဒီ parameter ကတော့ camera ရဲ့ image format ကိုရေးပေးရမှာပါ ။ image encodings တွေပေါ့ ။ rgb လား bgr လား စသဖြင့် ။

- **hfov parameter (required) –** ဒီ parameter ကတော့ camera ရဲ့ horizontal field of view ပါ ။ မြင်အောင်ပြောရရင် camera ကနေ မြင်နိုင်မယ့် data ဖတ်နိုင်မယ့် horizontal range ပေါ့ ။ unit က randians ပါ ။
- **near parameter (required) –** ဒီ parameter ကတော့ camera ရဲ့ near clip distance ပါ ။ စာနဲ့ ပြောပြဖို့တော့ ခက်တယ်ဗျ - အောက်က ပုံမှာကြည့်ရင် သိသွားပါလိမ့်မယ် ။ near clipping plane ပါ ။
- **far parameter (required) –** ဒီ parameter ကတော့ camera ရဲ့ far clip distance ပါ ။ စာနဲ့ ပြောပြဖို့တော့ ခက်တယ်ဗျ - အောက်က ပုံမှာကြည့်ရင် သိသွားပါလိမ့်မယ် ။ far clipping plane ပါ ။
-

အပေါ်က parameter အကုန်လုံးကို လက်တွေ့မြင်နိုင်ဖို့ ဒီအောက်ကပုံလေးက ကူညီပေးပါလိမ့်မယ် ။



Source - google

အပေါ်မှာ ရှင်းပြခဲ့တဲ့ **sensor tag** တစ်ခုလုံးကို ရှင်းရှင်းလင်းလင်း ဒီ အောက်က example မှာ ကြည့်နိုင်ပါတယ် ။

```
<sensor name="camera" update_rate="20">

  <parent link="attached_link"/>

  <origin xyz="0 0 0" rpy="0 0 0"/>

  <camera>
    <image width="640" height="480" hfov="1.5708" format="RGB8" near="0.01" far="50.0"/>
  </camera>

</sensor>
```