

Choreonoid モデル作成方法マニュアル

名城大学メカトロニクス工学科

ロボットシステムデザイン研究室




2017 年 1 月 11 日

1. はじめに

本書の目的は動力学シミュレータである Choreonoid のモデルを作成する方法を解説することである．ここで解説する手法は数多くあるモデル作成方法の一つなので，他の方法があることも留意すること．

2. 使用するソフトウェア

モデル作成に使用するソフトは以下の通りである．本文書では以下のソフトウェアの詳しい使い方については解説しない．

	Autodesk Inventor 2017
	Autodesk 3dsMax 2017
	Choreonoid-1.5.0

3. 作成環境

本文書でのモデル作成環境は以下に示す通りである．パーツを結合する際はテキストエディタで VRML97 を編集することになるので，Choreonoid はソースパッケージからのダウンロードが容易な Ubuntu14.04 で行っている．

Autodesk Inventor 2017	Windows10
Autodesk 3dsMax 2017	Windows10
Choreonoid-1.5.0	Ubuntu14.04

4. 作成するモデル

本文書では，作成する例として以下の図 1 に示す 5 自由度マニピュレータである CRANE+ のモデルを作成する



図 1 モデルを作成するロボットと完成図

5. モデル作成手順

本文書での Choreonoid モデルの作成は，Autodesk Inventor 2017 モデルのパーツを作成し，STL ファイルで出力する．出力したファイルを 3dsMax で VRML97 に変換する．その後，テキストファイルでパーツの結合を行う．それぞれの手順について解説していく．

6. Autodesk Inventor 2017 での Choreonoid パーツ作成

Inventor を立ち上げたら，新規作成のパーツを選択する．（図 2）

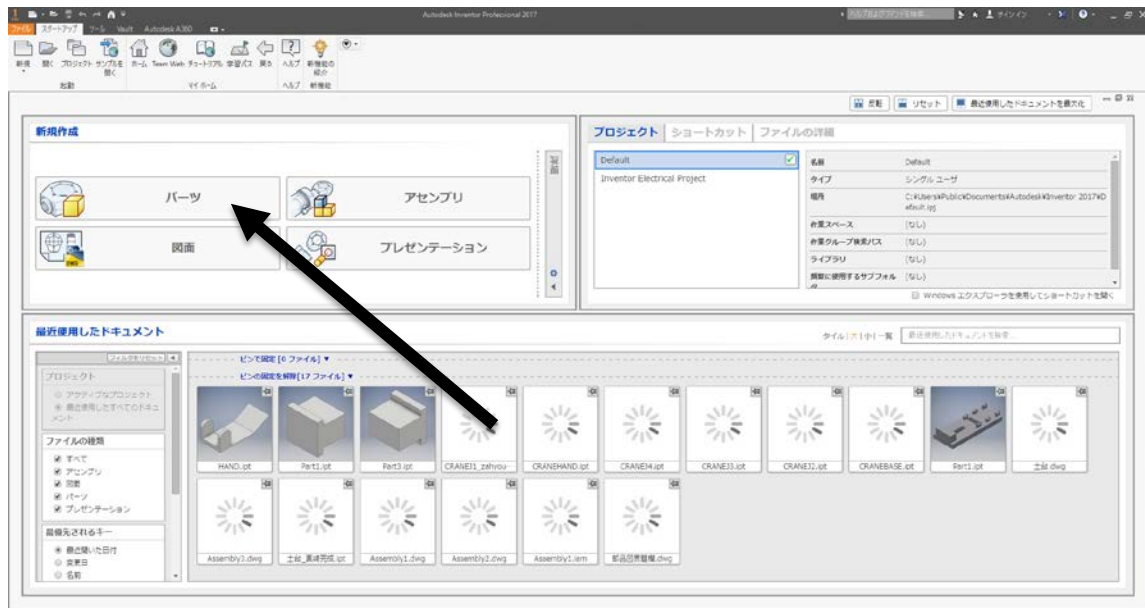


図 2 Inventor 新規作成

2D スケッチ開始をクリックする。(図 3)

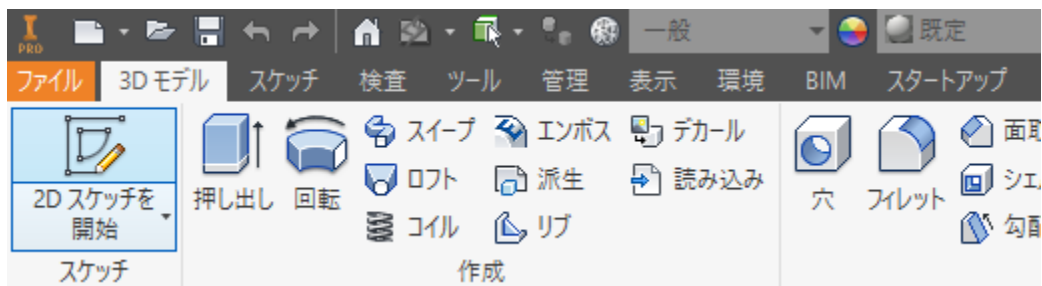


図 3 2D スケッチを開始

この 2D スケッチで Choreonoid 用のスケッチをする必要がある。

最初に図面を 2 次元で書く面を指定するが、ここで図 4 のように右上ブロックを右の反対にし、ブロックの右上をクリックする。



図 4 2D スケッチ右上ブロック

すると、左下の座標が以下の図 5 のようになる。

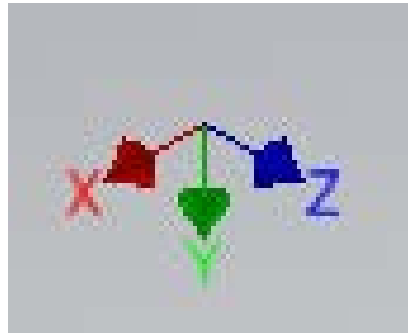


図 5 2D スケッチ座標

この状態で，図 6 のように XZPlane を選択する．

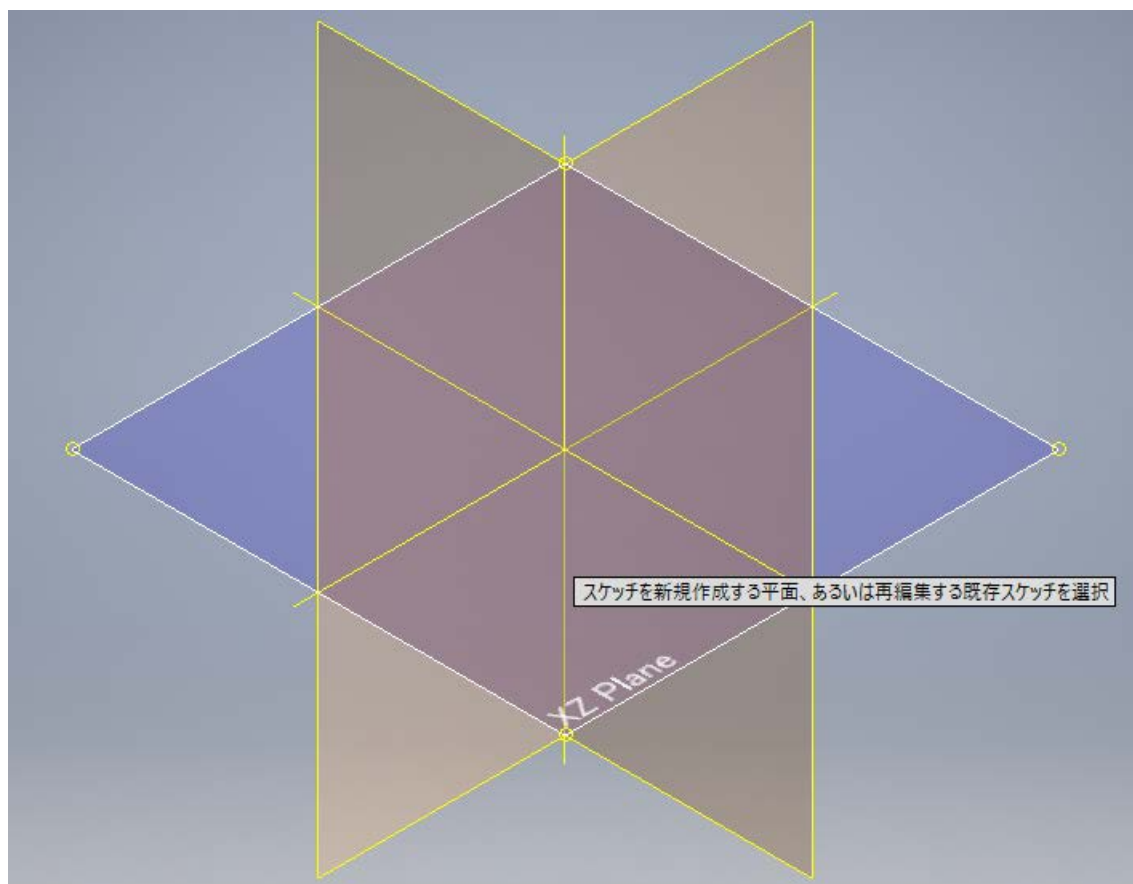


図 6 2D スケッチ XZPlane

すると，右上のブロックが図 7，左下の座標が図 8 のようになる．



図 7 2次元製図 右上ブロック

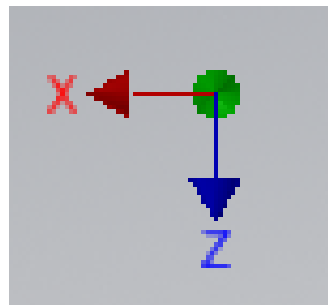


図 8 2次元製図 座標

この状態の Inventor の座標が，Choreonoid に変換すると以下の図 9 になる．
 なお，Choreonoid の座標は赤色が X 軸，緑色が Y 軸，青色が Z 軸である．



図 9 2次元製図 Choreonoid との座標比較

以上の点を留意して，Inventor で 3D モデルを製図する．完成したパーツモデルが Inventor 上で以下の図 10 の座標のとき，Choreonoid で図 10 の座標で表示される．

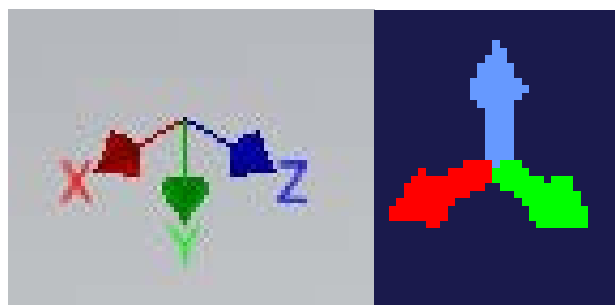
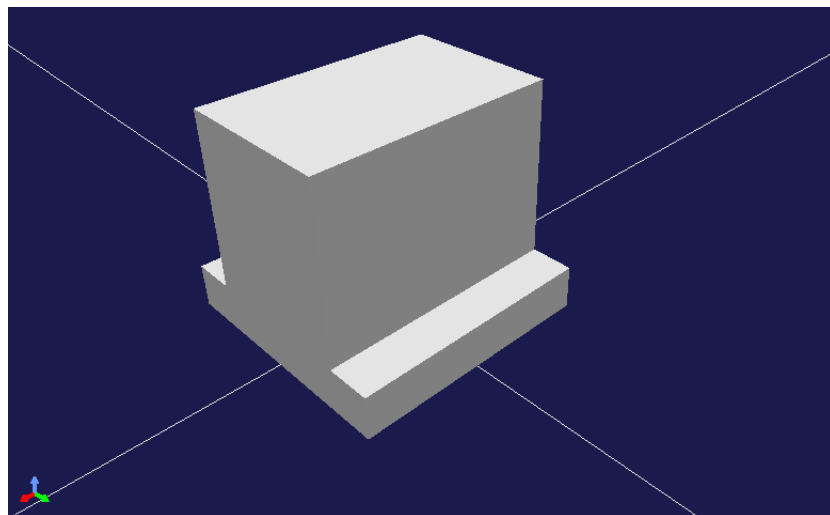
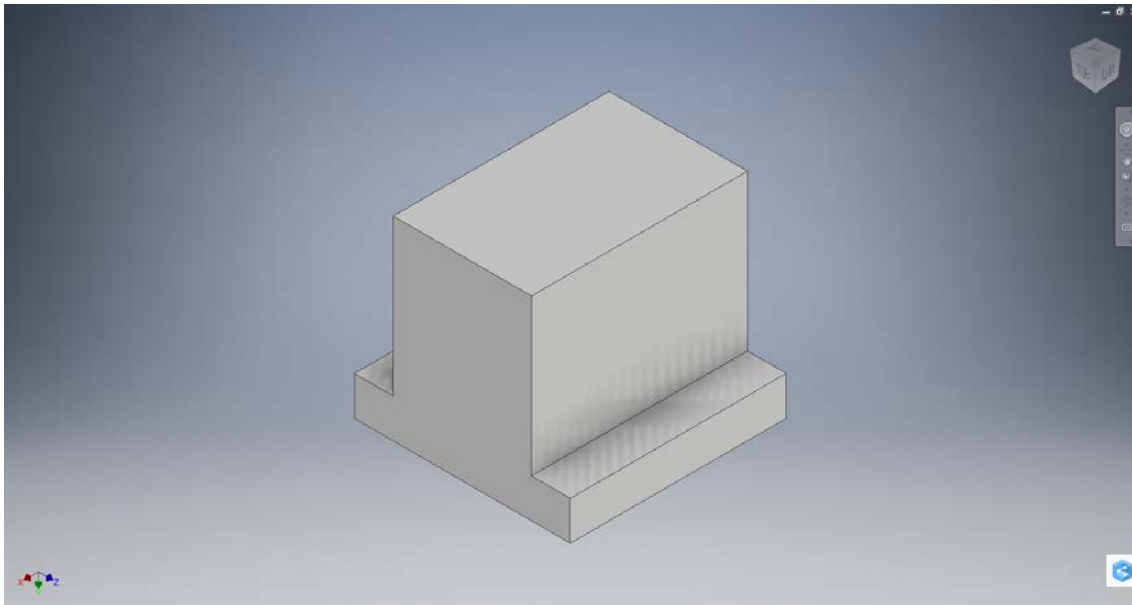


図 10 Inventor と Choreonoid の座標比較

7. VRML97 へのデータ変換

Choreonoid-1.5.0 で表示できるモデルファイルは VRML97 である。そのため、Inventor で 3D モデルを作成したら、そのモデルを VRML97 で保存する必要がある。しかし Autodesk Inventor 2017 では VRML97 で出力する機能はないので、STL ファイルで出力した後に Autodesk 3dsMax 2017 で STL ファイルを VRML97 に変換する必要がある。

まず、Inventor で作成したモデルを STL ファイルで出力する。ファイル→書き出し→CAD 形式を選択する。(図 11)

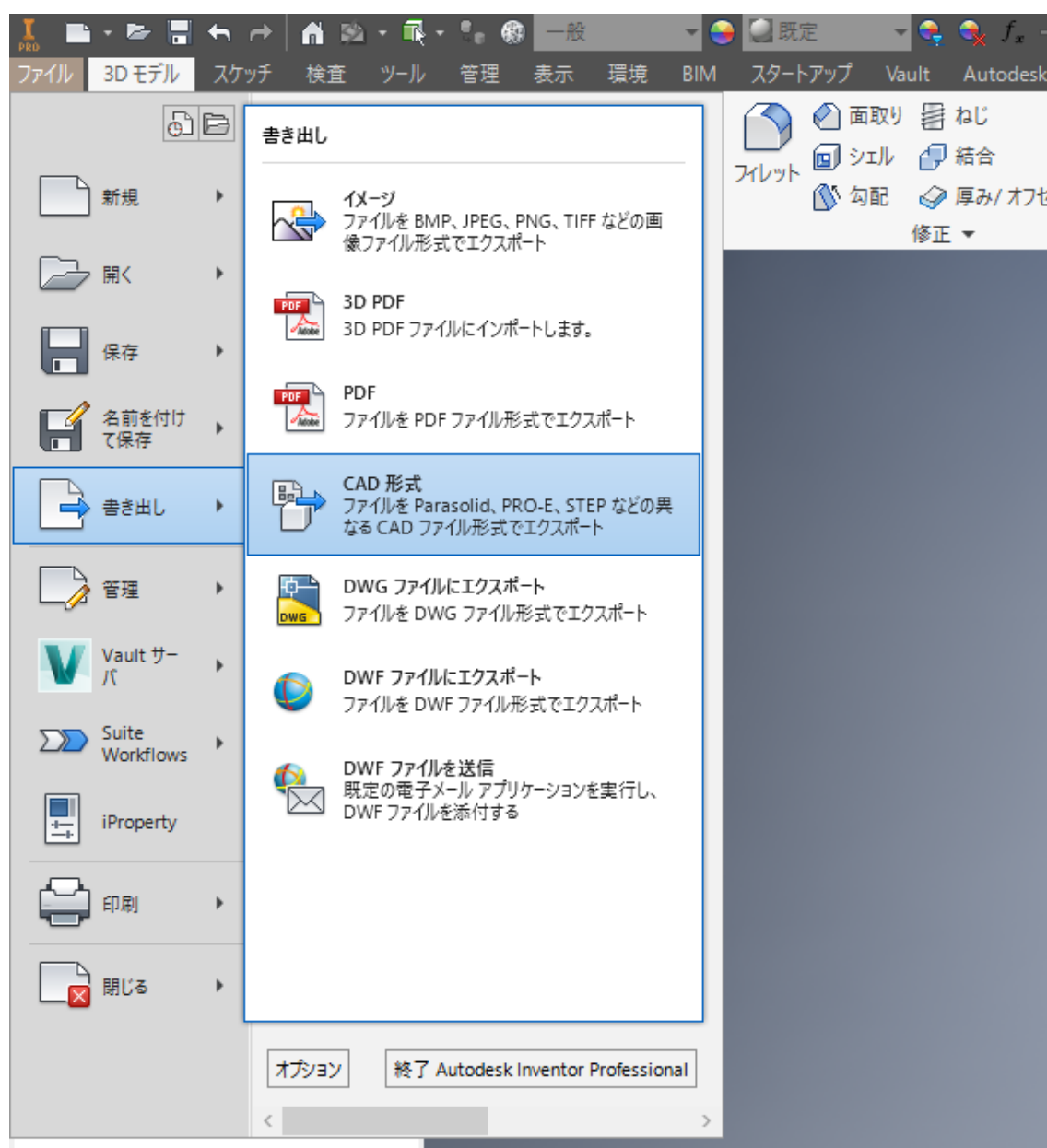


図 11 Inventor 書き出し

ファイル名，保存場所を指定して，ファイルの種類を **STL** ファイルにして保存する．（図 12）

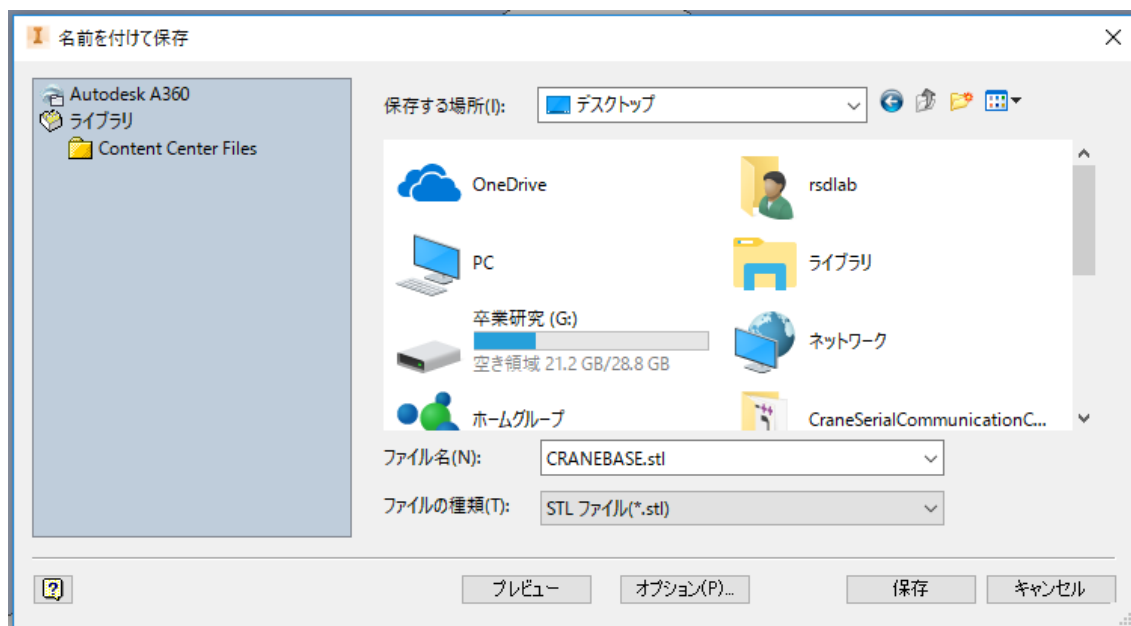


図 12 STL ファイルの保存

STL ファイルの保存ができれば，3dsMax を起動する．3dsMax を起動したら，左上のファイル→インポートを選択し，先ほど保存した STL ファイルをインポートする（図 13～図 17）．図 16 に関しては，そのまま OK を押せばよい．

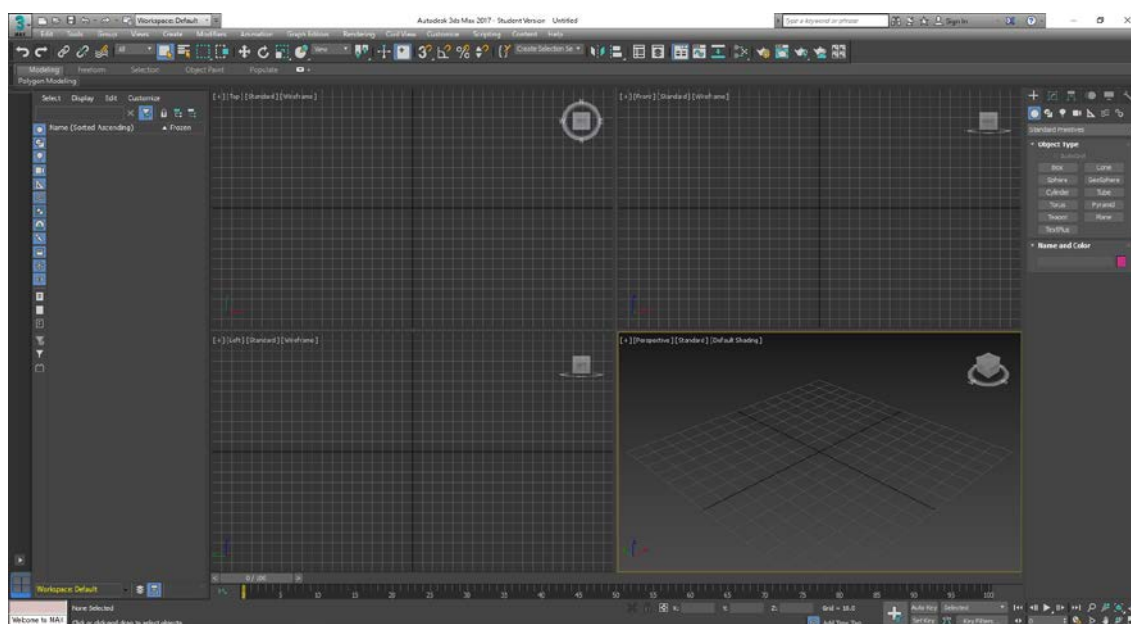


図 13 3dsMax 起動画面

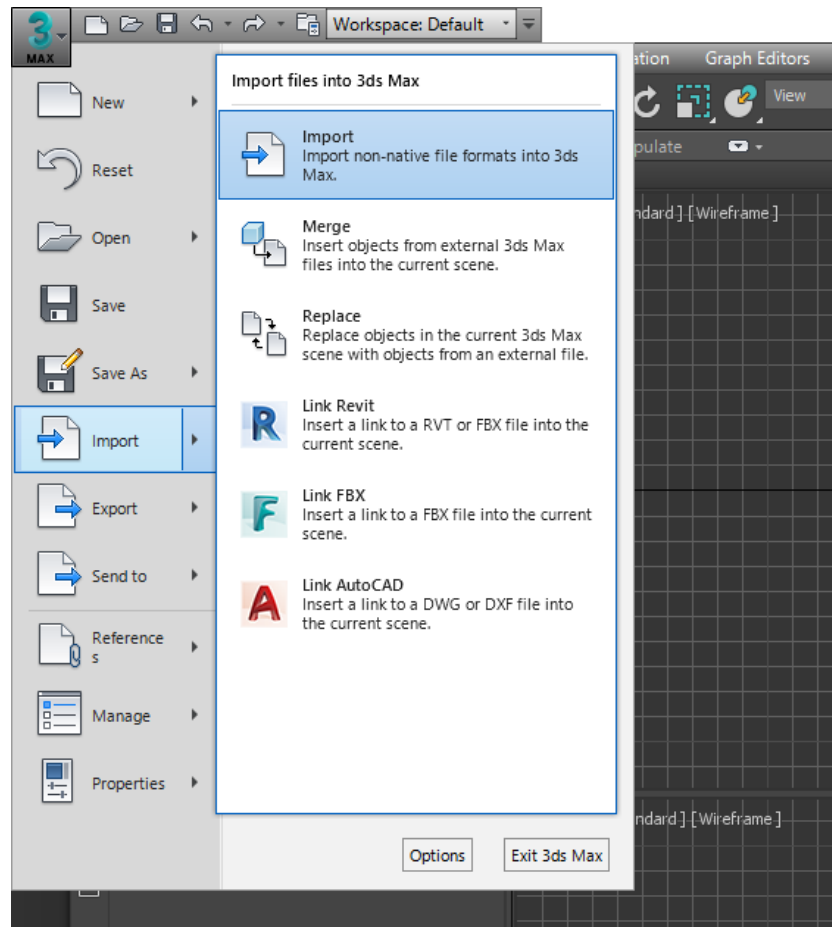


図 14 3dsMax インポート

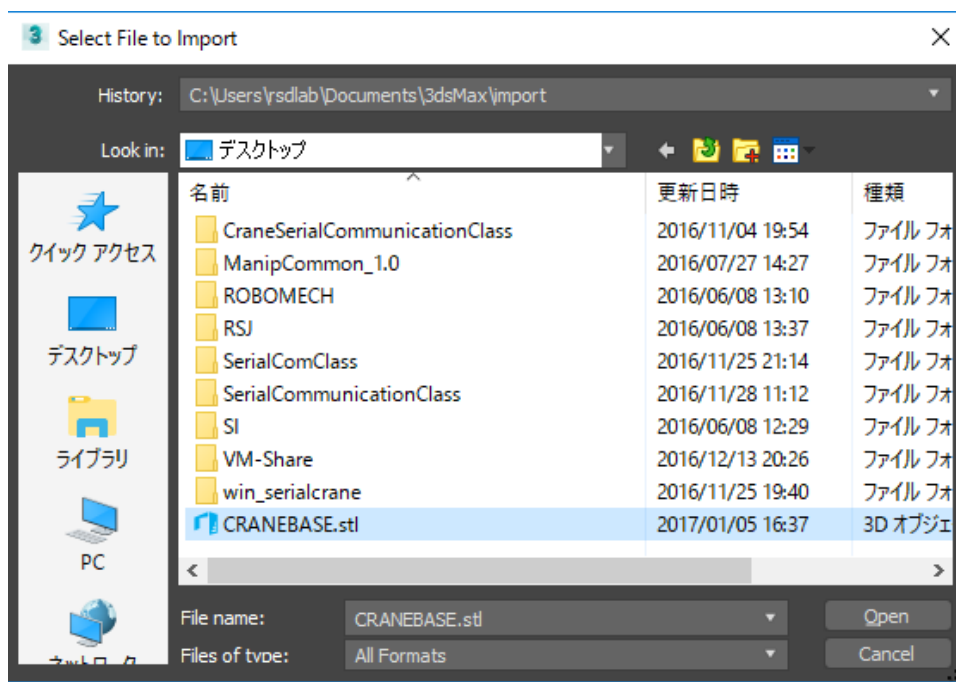


図 15 3dsMax インポートする STL ファイルの選択

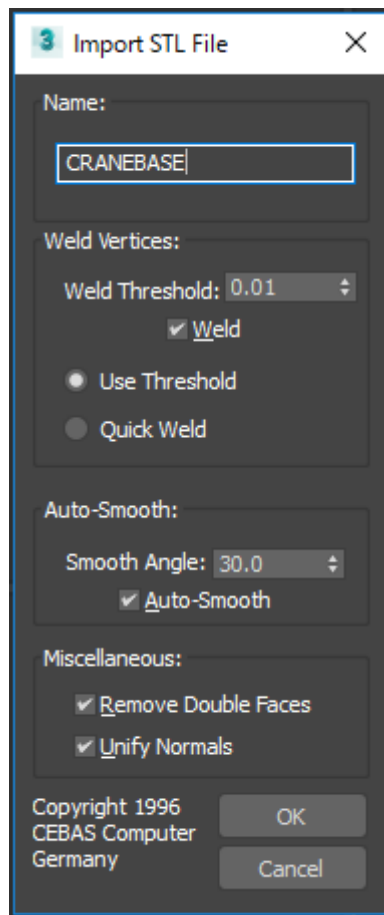


図 16 インポートの設定画面

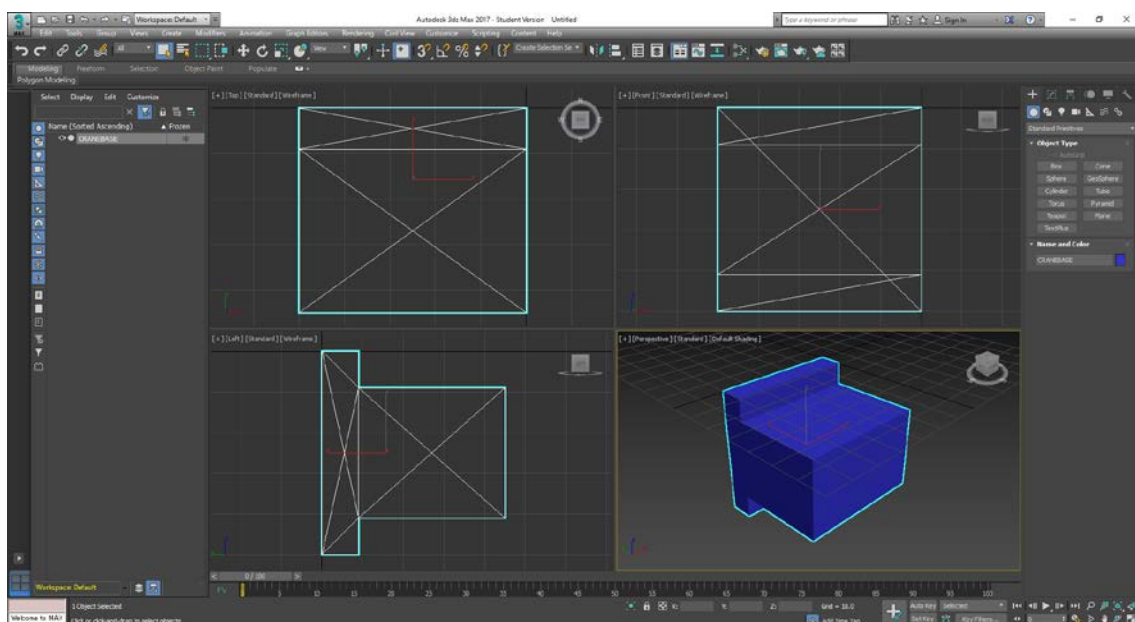


図 17 3dsMax インポート成功

STL ファイルのインポートができれば、そのまま VRML97 にエクスポートする。

左上のファイルからエクスポートを選択する（図 18）。

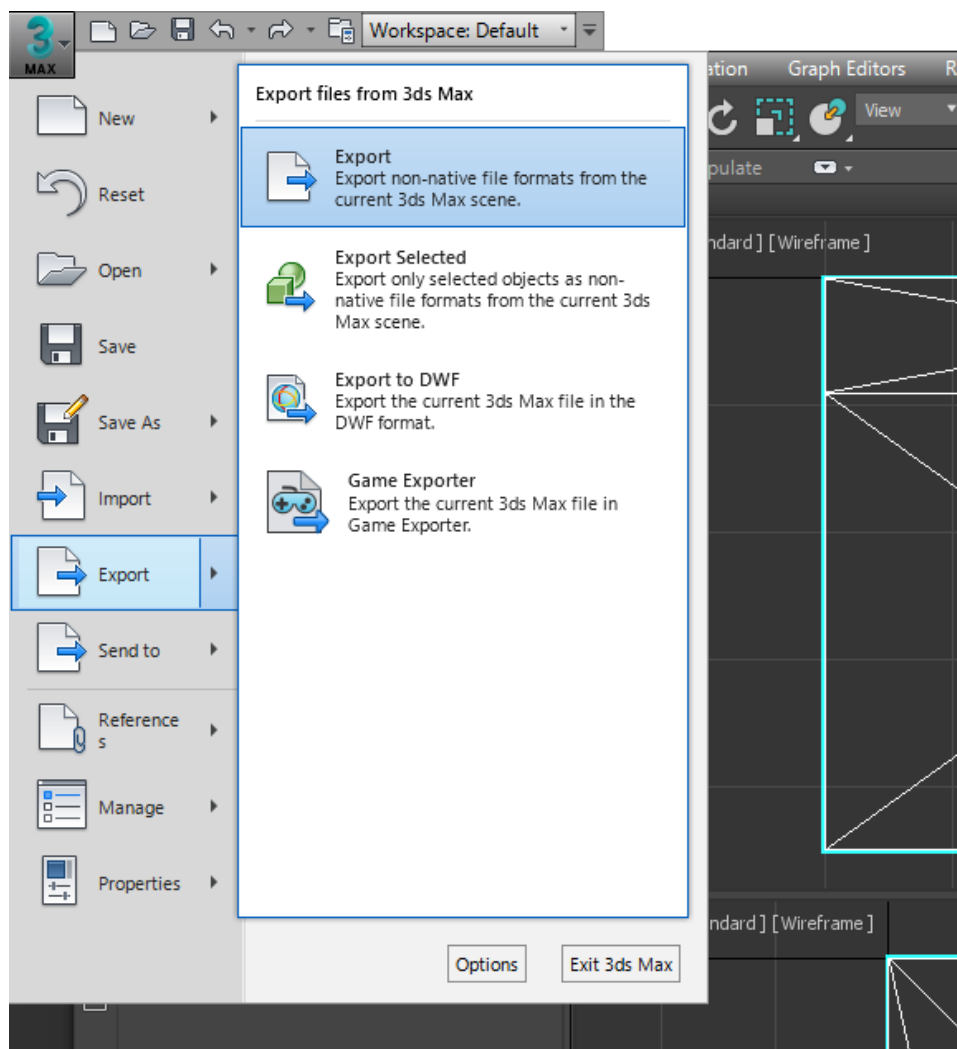


図 18 3dsMax エクスポート

エクスポートを選択したら、ファイル名、保存場所を選択して **Save as Type** を VRML97 にして保存する（図 19, 20）。図 20 に関してはそのまま OK を押せばよい。

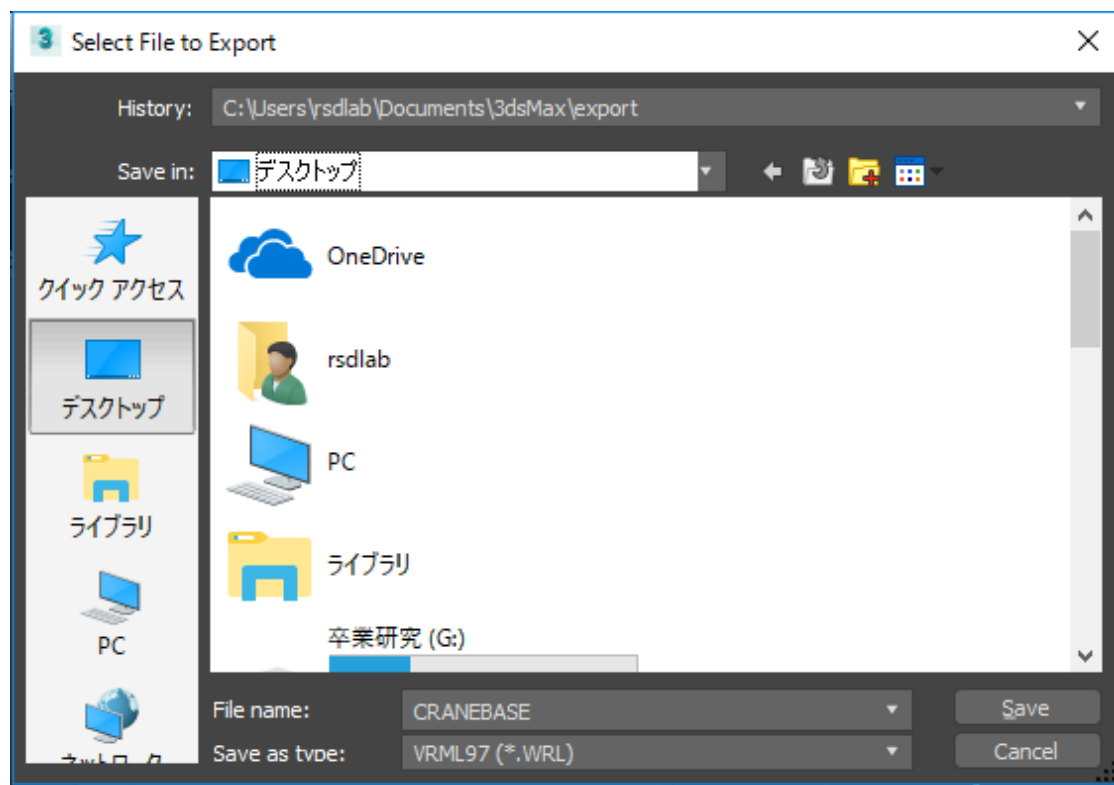


図 19 保存場所，ファイル名

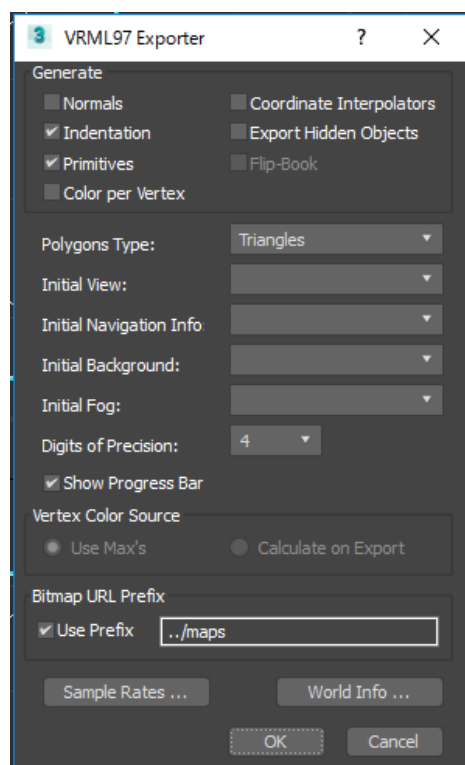


図 20 エクスポートの設定画面

8. Choreonoid パーツモデル表示

パーツモデルを VRML97 で保存したら，Choreonoid で表示する．しかし，3dsMax で出力される VRML97 の拡張子は.WRL で，Choreonoid では拡張子.wrl しか表示することができないと思われる．そのため，テキストエディタで作成した VRML97 と同名の拡張子が.wrl のファイルを作成し，.WRL の中身をそのままコピー&ペーストする(図 21， 22)．

3dsMaxで作成される
VRML97の拡張子

Choreonoidで表示できる
VRML97の拡張子

.WRL → .wrl

テキストエディタで.wrlを作って.WRLの内容を
コピー&ペーストする

図 21 WRL から wrl への変換

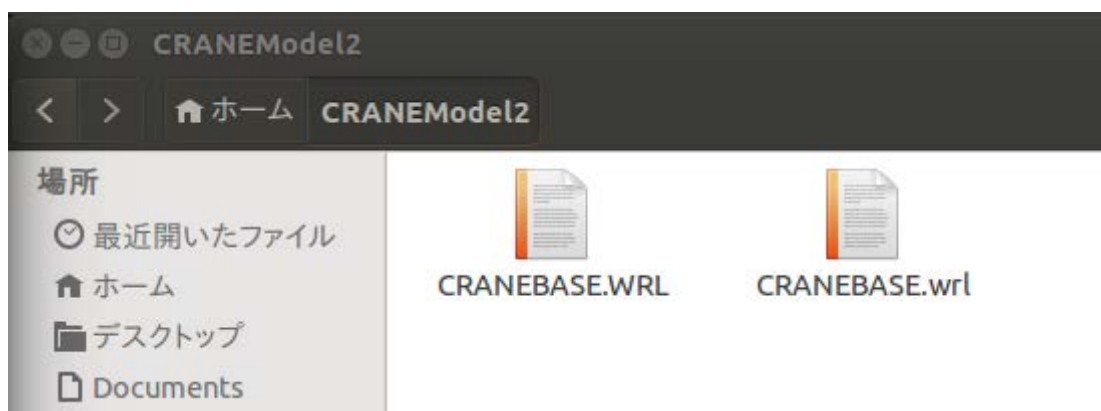


図 22 2つの VRML97 ファイル

.wrl ファイルができたなら，Choreonoid を起動し，左上のファイル→読み込み→OpenHRP モデルファイルを選択する（図 23）．

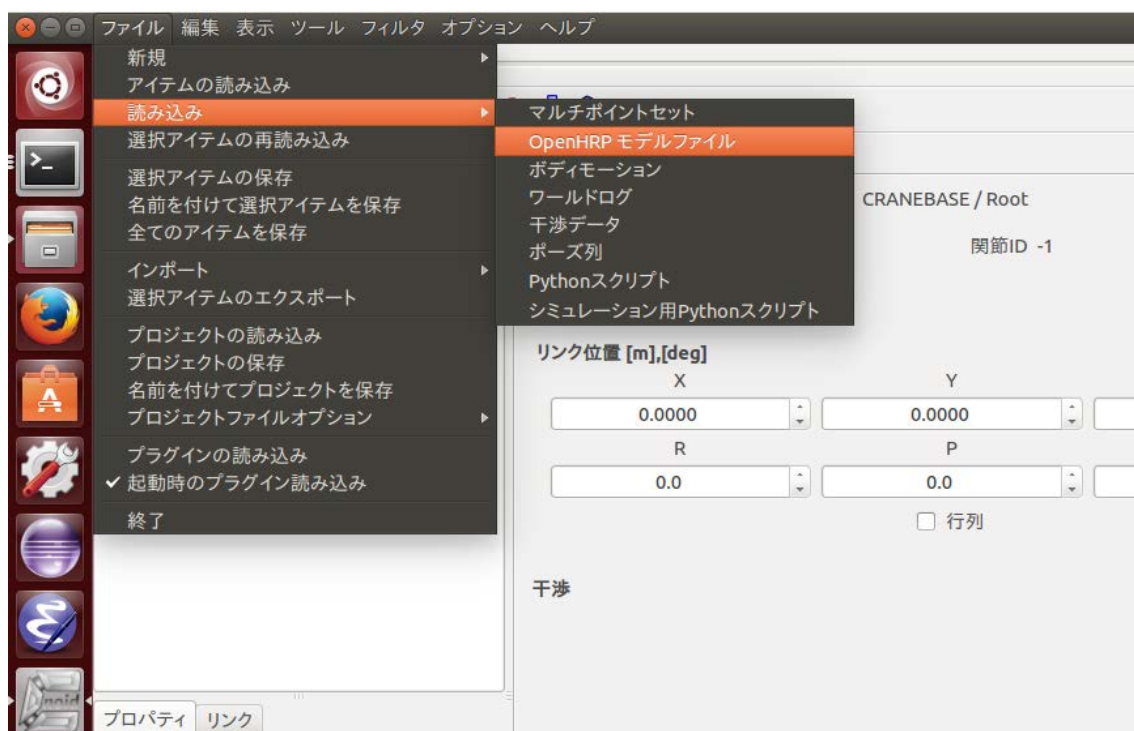


図 23 Choreonoid モデルファイルの読み込み

そこで、先ほど.wrlに変換した VRML97 を読み込みと、以下の図 24 のように Inventor で作成したモデルの 1000 倍の大きさで表示される。

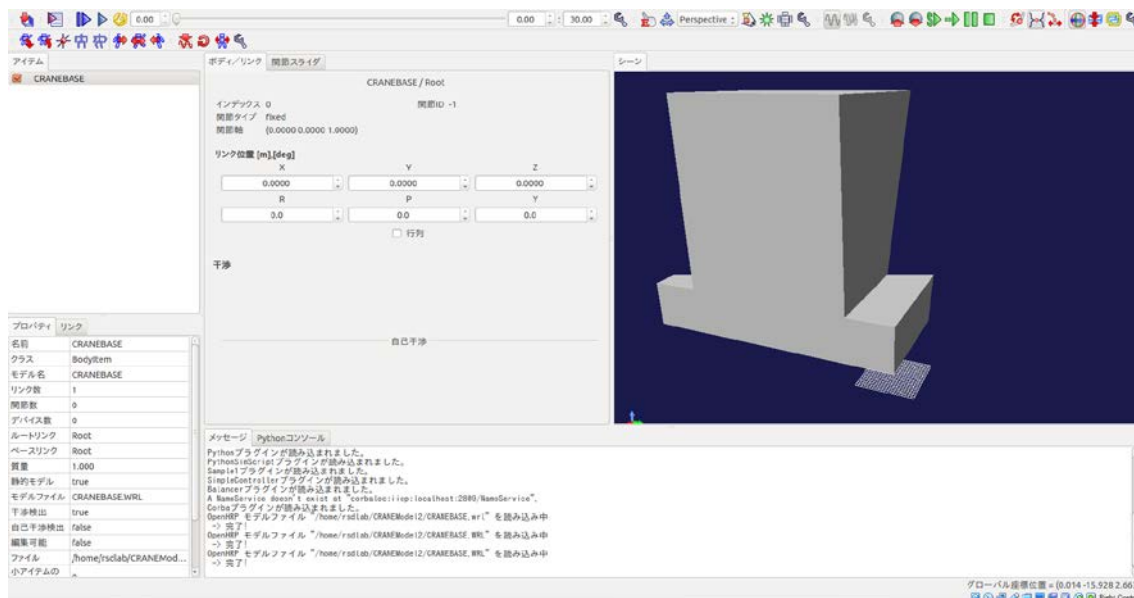
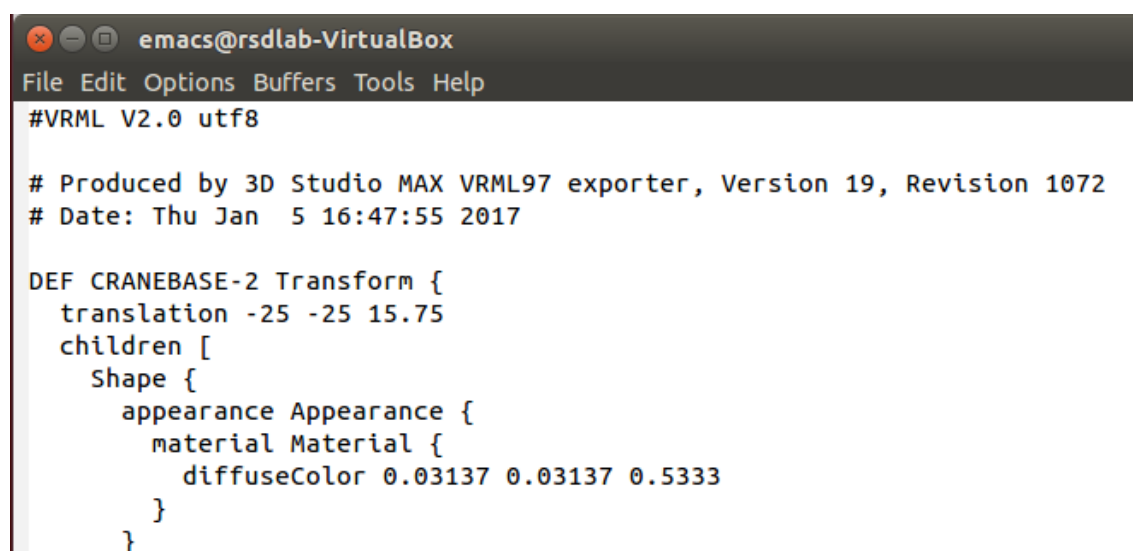


図 24 Choreonoid 1000 倍モデル

そのため、.wrl ファイルを開いてモデルの大きさを 1/1000 にする必要がある。以下の編集前の図 25 から、図 26 のように、wrl ファイルに scale という 1 行を追加する。また、モデルの大きさが 1/1000 になるため、translation の値も 1/1000 にする。モデルを座標の中心に置きたい場合は、translation の前 2 つの数字を 0 にする。(x, y 座標の移動を 0 にする)




```
#VRML V2.0 utf8

# Produced by 3D Studio MAX VRML97 exporter, Version 19, Revision 1072
# Date: Thu Jan 5 16:47:55 2017

DEF CRANEBASE-2 Transform {
  translation -25 -25 15.75
  children [
    Shape {
      appearance Appearance {
        material Material {
          diffuseColor 0.03137 0.03137 0.5333
        }
      }
    }
  ]
}
```

図 25 .wrl ファイル スケール変更前



```
#VRML V2.0 utf8

# Produced by 3D Studio MAX VRML97 exporter, Version 19, Revision 1072
# Date: Thu Jan 5 16:47:55 2017

DEF CRANEBASE-2 Transform {
  scale 0.001 0.001 0.001
  translation 0 0 0.01575
  children [
    Shape {
      appearance Appearance {
        material Material {
          diffuseColor 0.03137 0.03137 0.5333
        }
      }
    }
  ]
}
```

図 26 .wrl ファイル スケール変更後

スケールを 1/1000 に変更すると、以下の図 27 のように実寸大の大きさで Choreonoid に表示される。

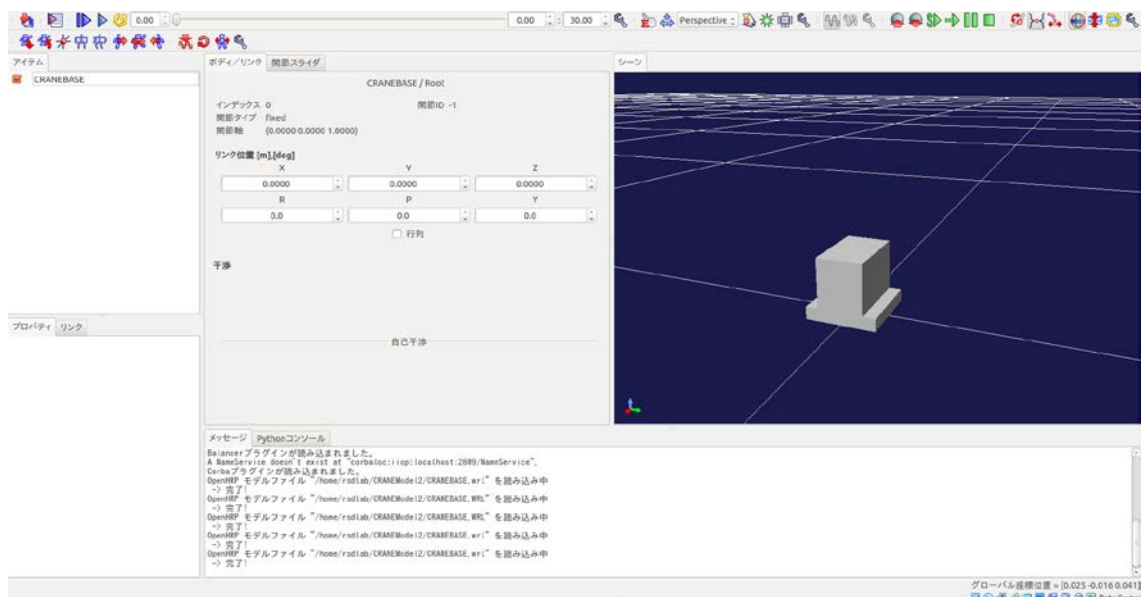


図 27 Choreonoid 実寸大モデル表示

同じように、作成したいモデルのパーツを図 28 のように Inventor で作成する。

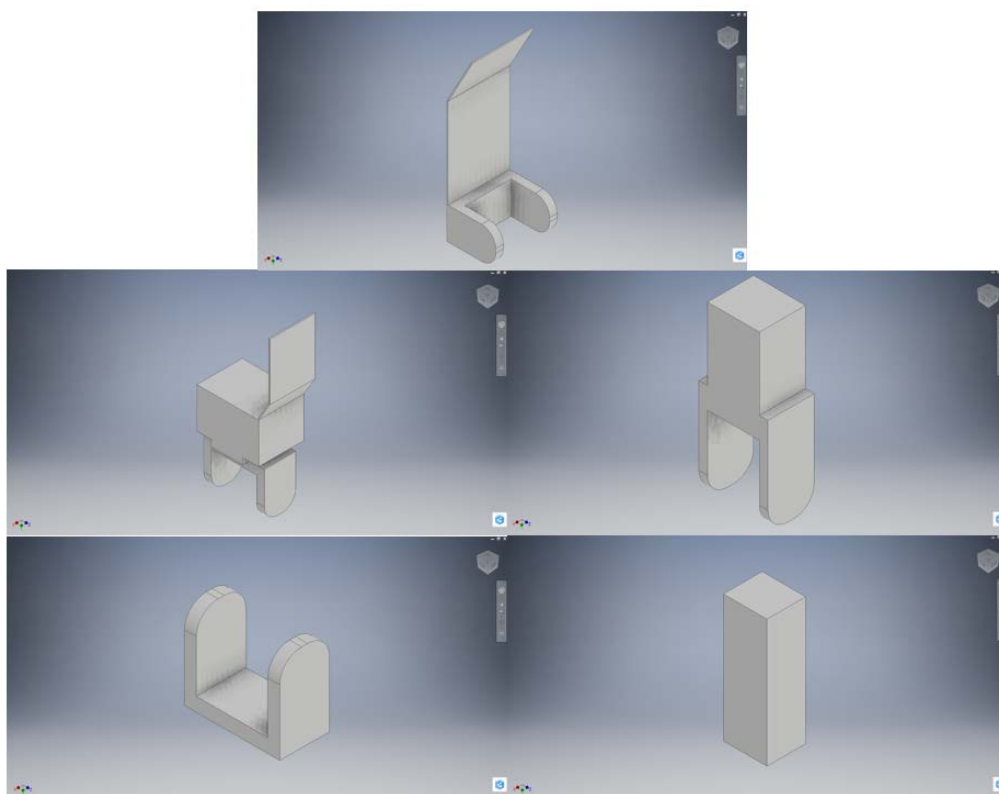


図 28 その他のパーツモデル

9. Choreonoid のパーツの回転軸の設定

Inventor でモデルのパーツをすべて作成し, VRML97 に変換して Choreonoid でモデルの表示が確認できたら, すべてのパーツを結合して一つのモデルファイルを作る.

まず, パーツの回転軸の位置を設定する. Z 軸方向に回転する関節は, 図 29 のように Choreonoid に表示したときの座標の中心の Z 軸方向が回転軸となる.

(translation の(x, y)を(0, 0) に設定している場合のモデルの中心) そのため, 図 29 のようにモデルの中心が Z 軸回転軸の場合回転軸の位置を設定する必要はない.

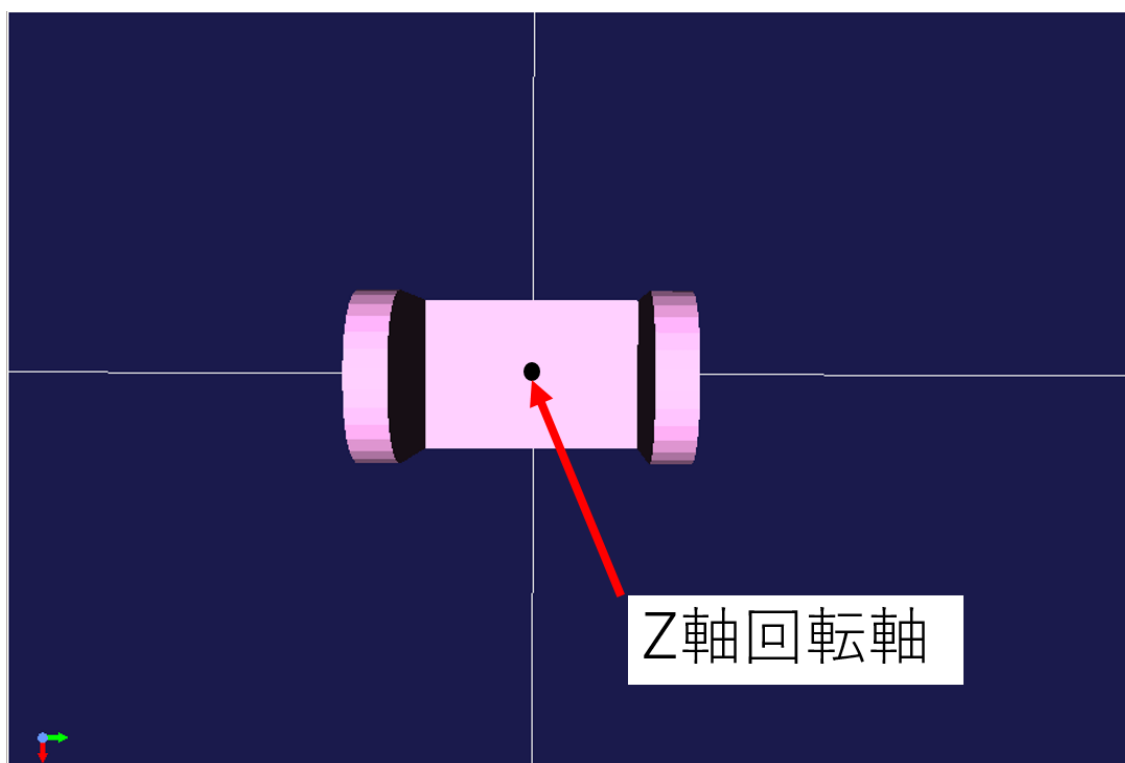


図 29 Z 軸回転軸の位置

次に, Y 軸回転軸を指定する. Y 軸回転軸は, 図 30 のように Choreonoid に表示したときの座標の中心が Y 軸方向の回転軸となる.

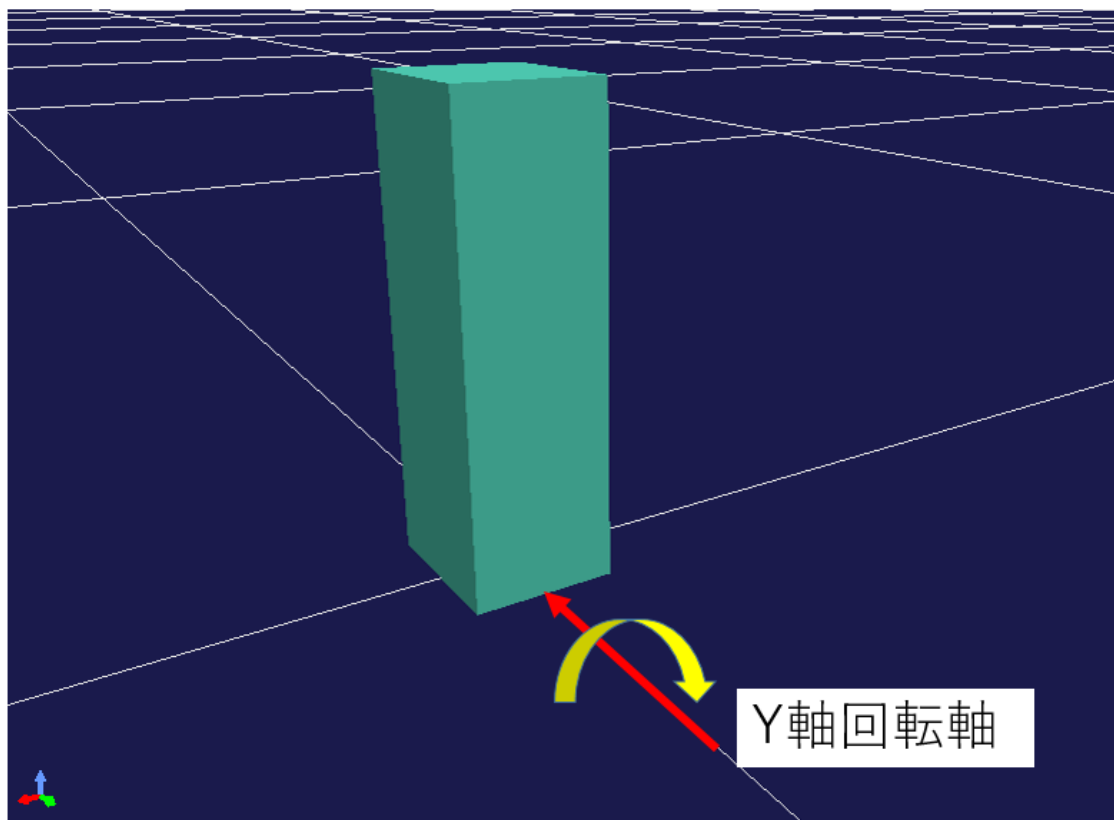


図 30 Y 軸回転の位置

今回は、下から 11mm の位置を Y 軸回転軸にしたいので、wrl ファイルを編集して Z 方向にモデルを移動する．図 31 のように、translation の Z 軸の値を -11mm(-0.011)にする．

```
DEF CRANEJ2-1 Transform {
  scale 0.001 0.001 0.001
  translation 0 0 0.0415 #編集後
  #translation 0 0 0.0525 #編集前
```

図 31 Y 軸設定のために Z 方向にモデル移動

図 31 のように編集すると、モデルが Z 軸方向に -11mm される(図 32)．

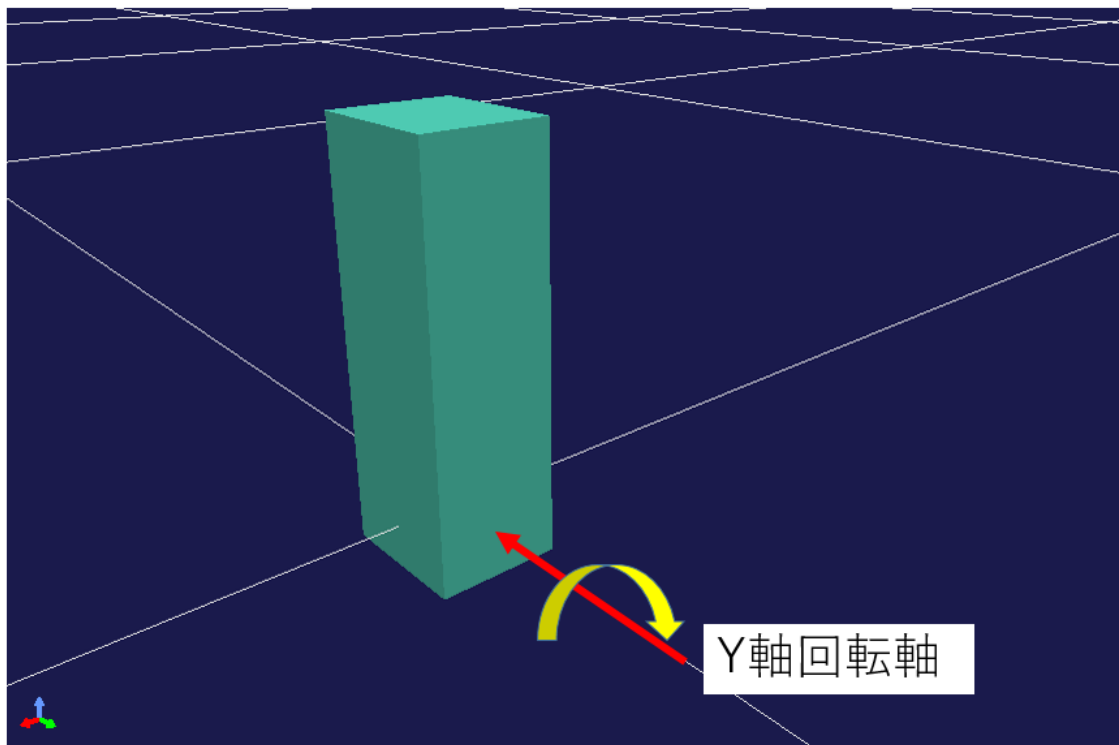


図 32 Z 軸方向移動後 Y 軸回転軸

X 軸回転するパーツも同様に，図 33 の位置が回転軸となるので，図 34 のように `wrl` ファイルを編集して回転軸にしたい位置に移動させる（図 35）．

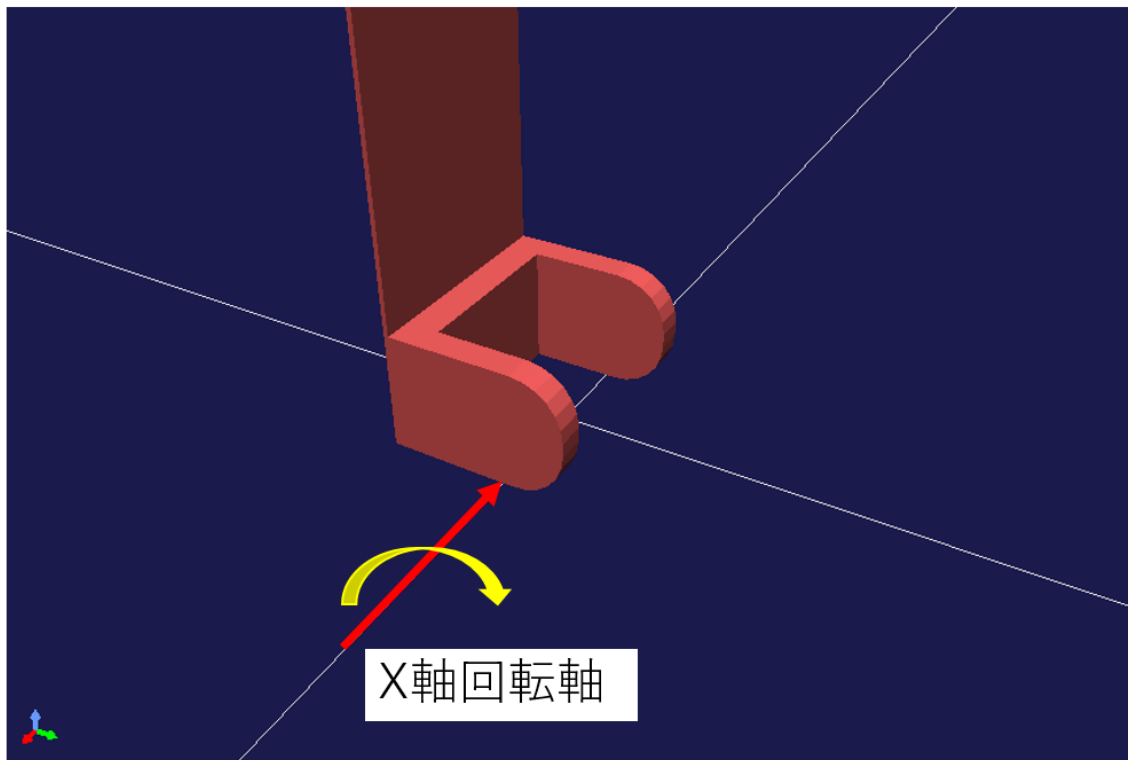


図 33 X 軸回転の位置

```
DEF newCRANEHAND3-2 Transform {
  scale 0.001 0.001 0.001
  translation 0 -0.00193 0.01475 #編集後
  #translation 0 0 0.02725 #編集前
```

図 34 X 軸設定のために ZY 方向に移動

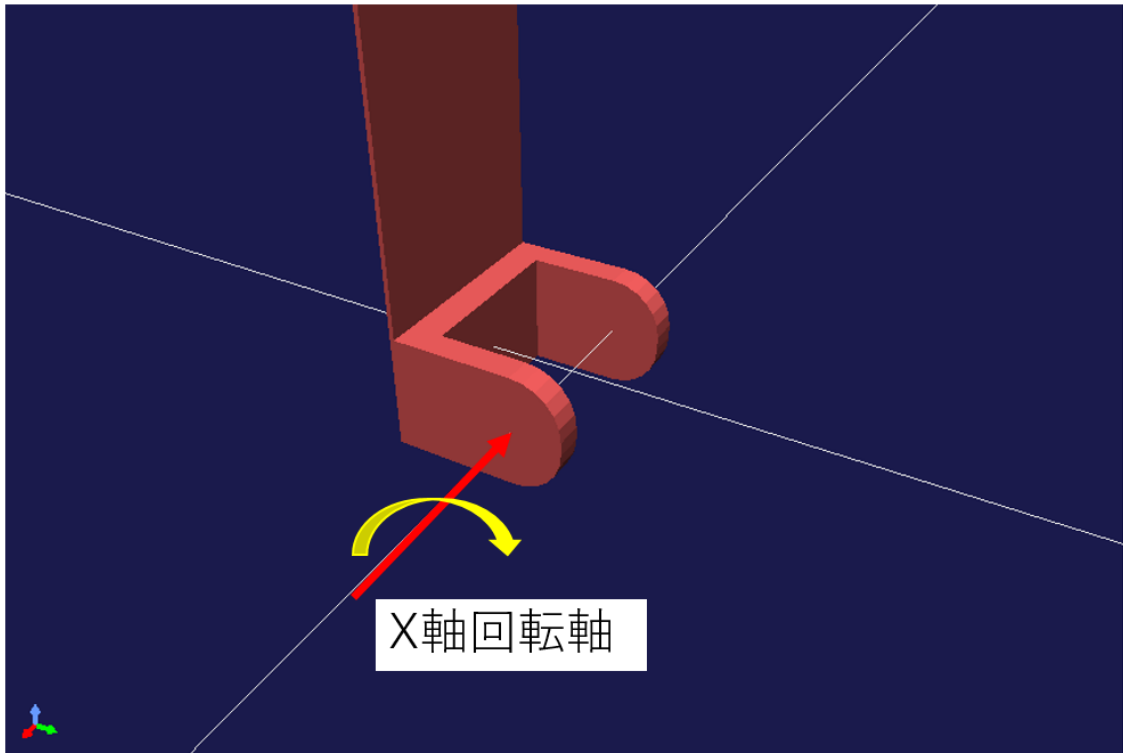


図 35 ZY 軸方向移動後 X 軸回転軸

10. Choreonoid パーツ結合

Choreonoid パーツの回転軸が設定出来たら、パーツを結合したモデルファイルを作る。今回は、サンプルとして提供されている PA10 のモデルファイルをコピーし、それを編集して作成する。Choreonoid のモデルはリンク構造、動力学/機構パラメータを定義するノードのインスタンスを組み合わせで階層構造を作ることによって、モデルを作成していく。詳細は Choreonoid ホームページの OpenHRP モデルファイルを参照してほしい。

Choreonoid ホームページ OpenHRP モデルファイル

<http://choreonoid.org/ja/manuals/1.5/handling-models/modelfile/modelfile-openhrp.html>

まず、PA10 のモデルファイルを開くと PROTO ノードが最初に宣言されているので、すべてコピーする (図 36)。

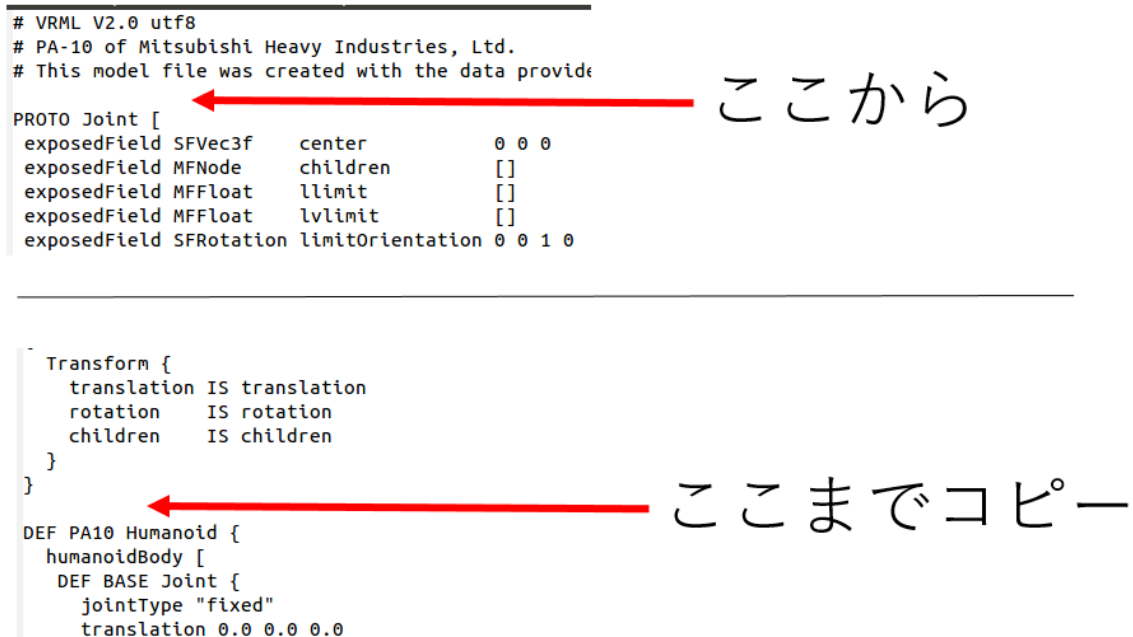


図 36 PA10 サンプルモデルをコピー

コピーしたところ以下の Humanoid ノードが、モデルのリンク構造を決定する部分となる。PA10 の Humanoid ノードを確認すると、リンクツリーが以下の図 37 のようになっていることが確認できる。

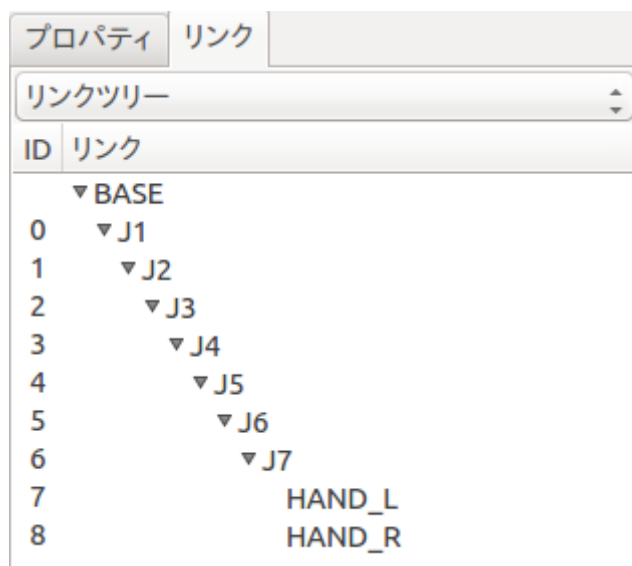


図 37 PA10 リンクツリー

今回作成したいモデルは 5 自由度であるため、以下の図 38 のようなリンクツリ

ーを目指す.



図 38 CRANE+リンクツリー

今回作成したいモデルはリンク構造が PA10 と似ているので, PA10 のモデルファイルの Humanoid 以下をすべてコピーし, J6, J7, HAND_L, HAND_R を削除する. J6, J7, HAND_L, HAND_R を削除するとは, PA10.wrl のコピーが以下の図 39 のようになればよい.

```
children [  
    DEF J5_LINK Segment {  
        mass 3.45  
        centerOfMass 0 0 1.09875  
        momentsOfInertia [1 0 0 0 1 0 0 0 1]  
        children [  
            Inline {  
                url "parts/J5.wrl"  
            }  
        ]  
    }  
] # end of joint J5  
] # end of joint J4  
] # end of joint J3  
] # end of joint J2  
] # end of joint J1  
] # end of joint BASE  
  
joints [  
    USE BASE,  
    USE J1,  
    USE J2,  
    USE J3,  
    USE J4,  
    USE J5  
]  
  
segments [  
    USE BASE_LINK,  
    USE J1_LINK,  
    USE J2_LINK,  
    USE J3_LINK,  
    USE J4_LINK,  
    USE J5_LINK  
]  
]
```

図 39 J6, J7, HAND_L, HAND_R 削除後 wrl ファイル

その後、BASE～J5 を先ほど Autodesk Inventor で作ったモデルに置き換えていく．以下の図 40 のように，各値を作成したいモデルの値に変えていく．

<pre> DEF PA10 Humanoid { humanoidBody [DEF BASE Joint { jointType "fixed" translation 0.0 0.0 0.0 rotation 0 0 1 0 children [DEF BASE_LINK Segment { mass 3.04 centerOfMass 0 0 0.075 momentsOfInertia [1 0 0 0 1 0 0 0 1] children [Inline { url "parts/BASE.wrl" }] }] }] DEF J1 Joint { jointType "rotate" jointAxis 0 0 1 jointId 0 translation 0 0 0.2 rotation 0 0 1 0 ulimit 3.08923278 llimit -3.08923278 uvlimit 3.141592653589793 lvlimit -3.141592653589793 rotorInertia 3.0E-4 children [DEF J1_LINK Segment { mass 9.78 centerOfMass 0 0 0.14818 momentsOfInertia [1 0 0 0 1 0 0 0 1] children [Inline { url "parts/J1.wrl" }] }] } } </pre>	<p>編集</p> 	<pre> DEF CRANE Humanoid { humanoidBody [DEF BASE Joint { jointType "fixed" translation 0.0 0.0 0.0 rotation 0 0 1 0 children [DEF BASE_LINK Segment { mass 0.065 centerOfMass 0 0 0.02 momentsOfInertia [1 0 0 0 1 0 0 0 1] children [Inline { url "parts/CRANEBASE.wrl" }] }] }] DEF J1 Joint { jointType "rotate" jointAxis 0 0 1 jointId 0 translation 0.0125 0 0.045 rotation 0 0 1 0 ulimit 2.617993878 llimit -2.617993878 uvlimit 3.141592653589793 lvlimit -3.141592653589793 rotorInertia 0 children [DEF J1_LINK Segment { mass 0.01 centerOfMass 0 0 0.015 momentsOfInertia [1 0 0 0 1 0 0 0 1] children [Inline { url "parts/CRANEJ1.wrl" }] }] } } </pre>
----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	---------------------------------------------------------------------------------------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

図 40 PA10 モデルファイルを編集して自分のモデルへ

ここで，それぞれの値が何を示しているのか以下の図 41，42 に示す．図 41，42 の表は，Choreonoid ホームページ OpenHRP モデルファイルから参照したものである．参照するパーツのモデルファイルは Segment ノードの children 以下の url に記述する．図 43 のように，結合するモデルファイルと，パーツファイルをディレクトリに分けるとわかりやすい．

Jointノードのフィールド

フィールド	内容
name	Joint名を指定するフィールドです。
translation	ローカル座標系の位置を設定するフィールドです。親ノードからのオフセット値を指定します。
rotation	ローカル座標系の姿勢を設定するフィールドです。親ノードからのオフセットを指定します。
center	関節回転中心の位置を指定するフィールドです。ローカル座標系原点からのオフセットで指定します。
children	子ノードをぶら下げるフィールドです。0個以上のJointノード、0または1個のSegmentノードをぶら下げます。
jointType	関節タイプを設定するためのフィールドです。free, slide, rotate, fixed, crawler のうちのいずれかを指定します。"free" は任意軸方向への並進・任意軸回りの回転が可能で、rootリンクが固定されないモデルのrootリンクに設定します(6自由度)。“rotate” はjointAxisで指定する軸回りの回転のみ可能です(1自由度)。“slide” は jointAxisで指定する軸方向への並進直動関節となります(1自由度)。“fixed” は関節を固定します(自由度なし)。“crawler”を指定すると、付随するリンクが簡易的なクローラとして機能するようになります。この詳細はクローラのシミュレーションを参照してください。
jointId	関節番号を指定するためのフィールドです。jointIdは関節角度等の属性値を配列形式に並べて格納する際に何番目の要素に入れるかを指定するために利用されます。多くの場合、ロボットのコントローラ開発において関節角度を読み取ったり、指定したりできるのは制御可能な関節のみですから、そのような関節にjointIdを付ける、と考えていただければよろしいかと思います(必ずそうでなければならないということではありません)。以下、Idのつけ方に関するルールを示します。jointIdは0から始まる。jointIdには連続した整数値を用いる(間が空いたり、重複したりしていないこと)。
jointAxis	関節の軸を指定するためのフィールドです。OpenHRPのバージョン2までは文字列の“X”、“Y”、“Z”のいずれかで軸を指定していましたが、OpenHRP3以降ではベクトルを用いて任意方向への軸を指定可能となっています。旧バージョンの指定法もサポートはされますが、今後は新しい指定法をお使いください。
ulimit	関節回転角度の上限値[rad]を指定するフィールドです。(デフォルト値: “+∞”)
llimit	関節回転角度の下限値[rad]を指定するフィールドです。(デフォルト値: “-∞”)
uvelimit	関節回転角速度の上限値[rad/s]を指定するフィールドです。(デフォルト値: “+∞”)
lvelimit	関節回転角速度の下限値[rad/s]を指定するフィールドです。(デフォルト値: “-∞”)
gearRatio	ギヤ比: モータから関節までの減速比が1/100であれば、100と記述します
gearEfficiency	減速器の効率。効率が60%であれば0.6と記述します。このフィールドがなければ、効率100%の減速器を想定します。
rotorInertia	モータ回転子の慣性モーメント[kg ^m 2]

注釈: ulimit, llimit, uvelimit, lvelimit については、シミュレーションでは通常使用されません。コントローラがこれらの値を読み込んで限界値を超えないように制御するために定義されているパラメータとなっています。

図 41 Joint ノードのフィールド

Segmentノードのフィールド

フィールド	内容
bboxCenter	OpenHRPでは使用しません。
bboxSize	OpenHRPでは使用しません。
centerOfMass	重心位置を指定するフィールドです。
children	子ノードをぶら下げるフィールドです。ここに、形状を定義するノードを追加します。
coord	OpenHRPでは使用しません。
displacers	OpenHRPでは使用しません。
mass	質量を指定するフィールドです。
momentsOfInertia	重心位置回りの慣性テンソルを指定するフィールドです。
name	Segment名を指定するフィールドです。
addChildren	OpenHRPでは使用しません。
removeChildren	OpenHRPでは使用しません。

リンク形状は、Segmentノードに定義します。Segmentノードは、Jointノードの子ノードとして複数個設定でき、Transformノードの子ノードとして記述することも可能です。

図 42 Segment ノードのフィールド

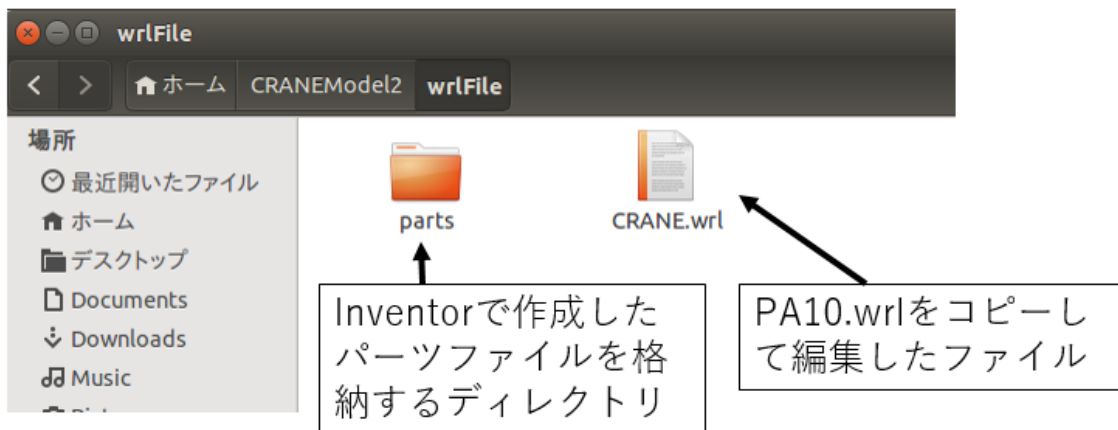


図 43 パーツファイルを分けるとわかりやすい

モデルファイルができたなら、Choreonoid を再起動して、結合したモデルファイルの読み込みを行う（図 44）.

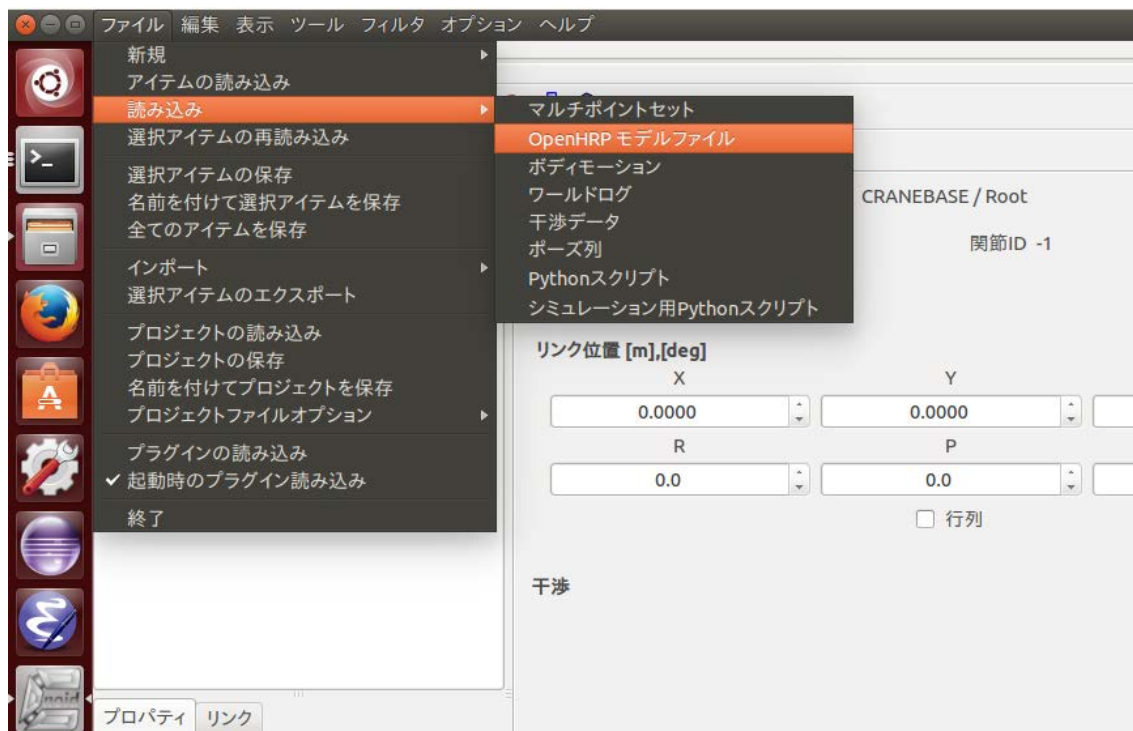


図 44 Choreonoid モデルファイルの読み込み

モデルファイルの読み込みができたなら、図 45 のように表示される。

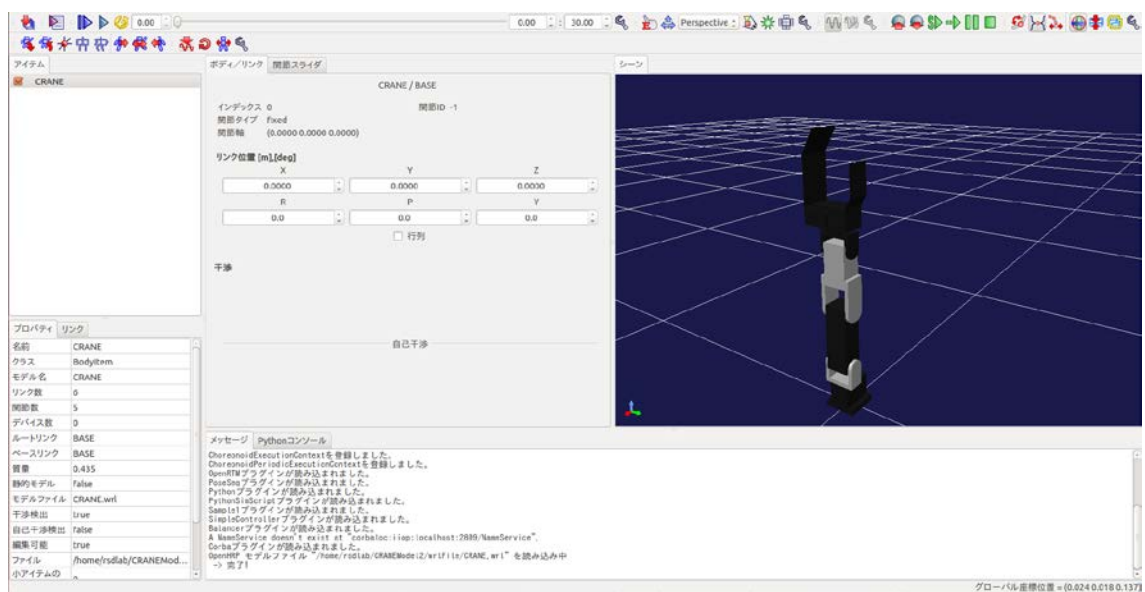


図 45 Choreonoid モデル読み込み成功

アイテムのモデルファイルを選択して、関節スライダを動かして、ジョイントが動くのを確認する（図 46）。

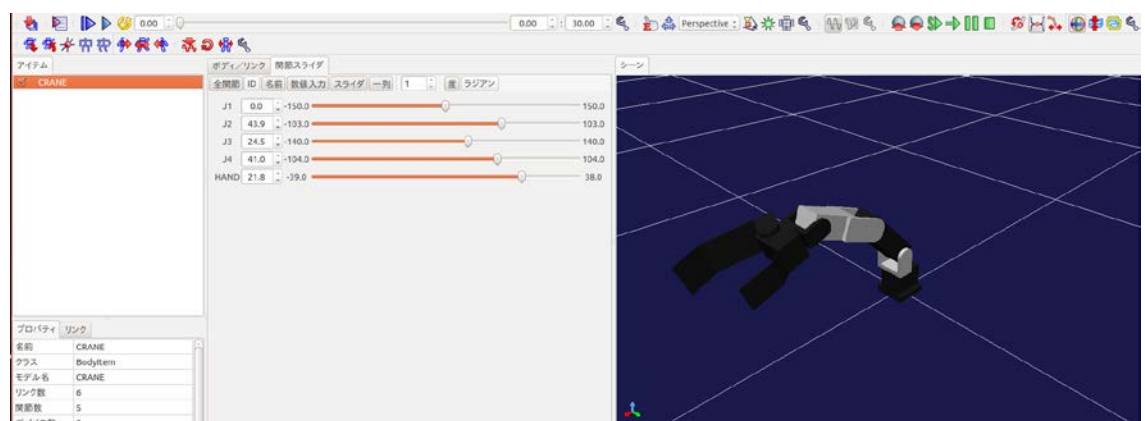


図 46 Choreonoid 関節スライダテスト