Choreonoid講習会
~ シミュレータの基本操作 ~

中村啓太（会津大学）
穴澤剛士（株式会社FSK）

講習会目的

❖ Choreonoidの基本操作を学ぶ

❖ シンプルコントローラの作成方法を学ぶ

❖ 自律制御について学び、ライントレースを行う

講習会内容

- ❖ Choreonoidとは?
- ❖ Choreonoidの基本操作
- ❖ プロジェクトの作成
- ❖ ボディモデルとは?
- ❖ コントローラとは?
- ❖ TurtleBot2の自律走行
- ❖ カメラ画像を用いたTurtleBot2の自律走行

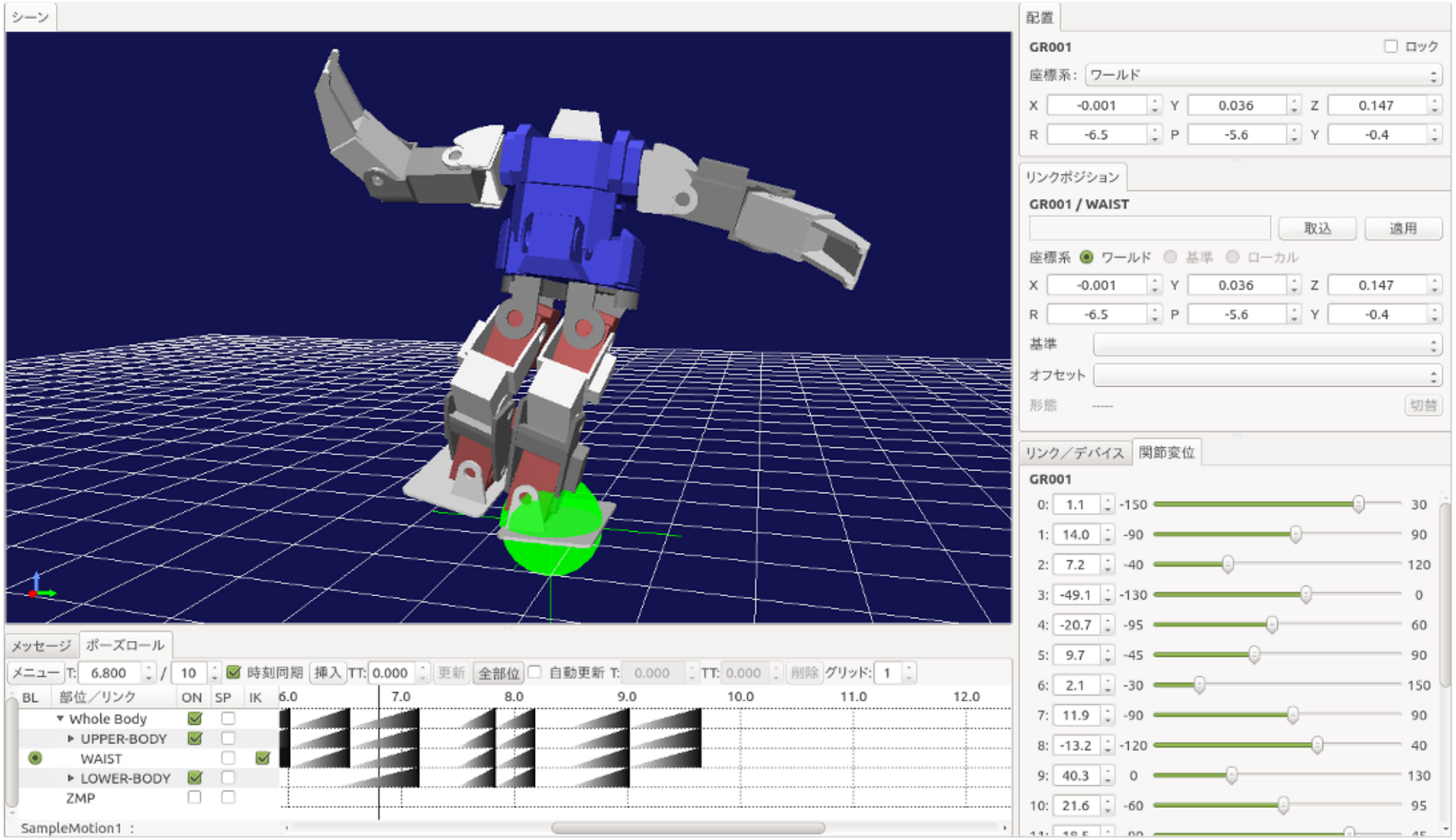
Choreonoidとは?

Choreonoid

- Choreograph（振り付けをする）+ Humanoidにより命名
- オープンソースのロボット用統合GUIソフトウェア
- 国立研究開発法人・産業技術総合研究所（産総研）が開発
  - 現在は株式会社コレオノイドが引き継いで開発
- **MITライセンス**
  - 利用や配布，改変などは無償で自由に行うことができるライセンス
- 動作振り付け機能，**動力学シミュレーション機能**
- Choreonoid関連ドキュメントは以下を参照
  - https://choreonoid.org/ja/

動作振り付け

❖ 3DCGによるロボットモデルのアニメーション表示しながら，
ロボットの多様な動作を自由に作成できる



動力学シミュレーション

- 物体の動作における力の影響をシミュレーション可能
  - ロボットが床の上を走行する
  - ロボットがバルブを回す

- 複数の物理エンジンを使用できる
  - **AISTエンジン（標準搭載）**
  - **AGX Dynamics（有償）**
  - Open Dynamics Engine

Choreonoidの基本操作

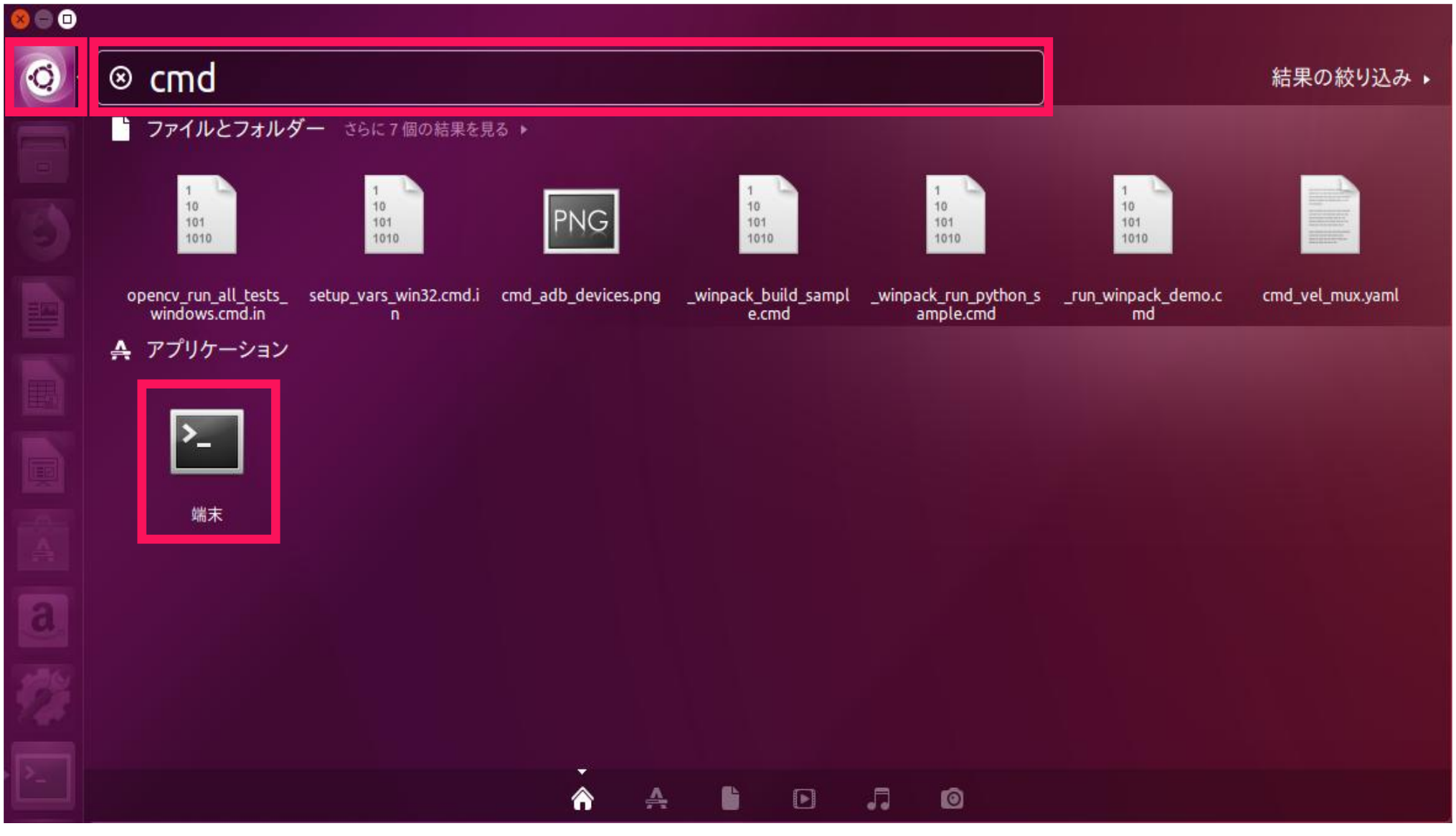
起動方法

❖ 2パターンで起動可能

1. コマンドラインからの起動
   - ❖ CUI(Character User Interface)を用いた起動方法
     - ❖ キーボードでコマンド（文字列）を入力
2. ファイルマネージャーからの起動
   - ❖ GUI(Graphical User Interface)を用いた起動方法
     - ❖ マウスでウィンドウやアイコンなどをクリック

コマンドラインによる起動

❖ ランチャーから『コンピュータを検索』をクリックし，
検索欄に『cmd』と入力して，『端末』をクリックし起動



コマンドラインによる起動 | 引数なし

- 端末上で以下のコマンドを実行
  - make installを行った場合

```
$ choreonoid
```

  - あるいは

```
$ cd [ビルドディレクトリ]
$ bin/choreonoid
```

例: cd ~/choreonoid/build/

コマンドラインによる起動 | 引数あり

❖ コマンドライン引数によるプロジェクトファイルの起動

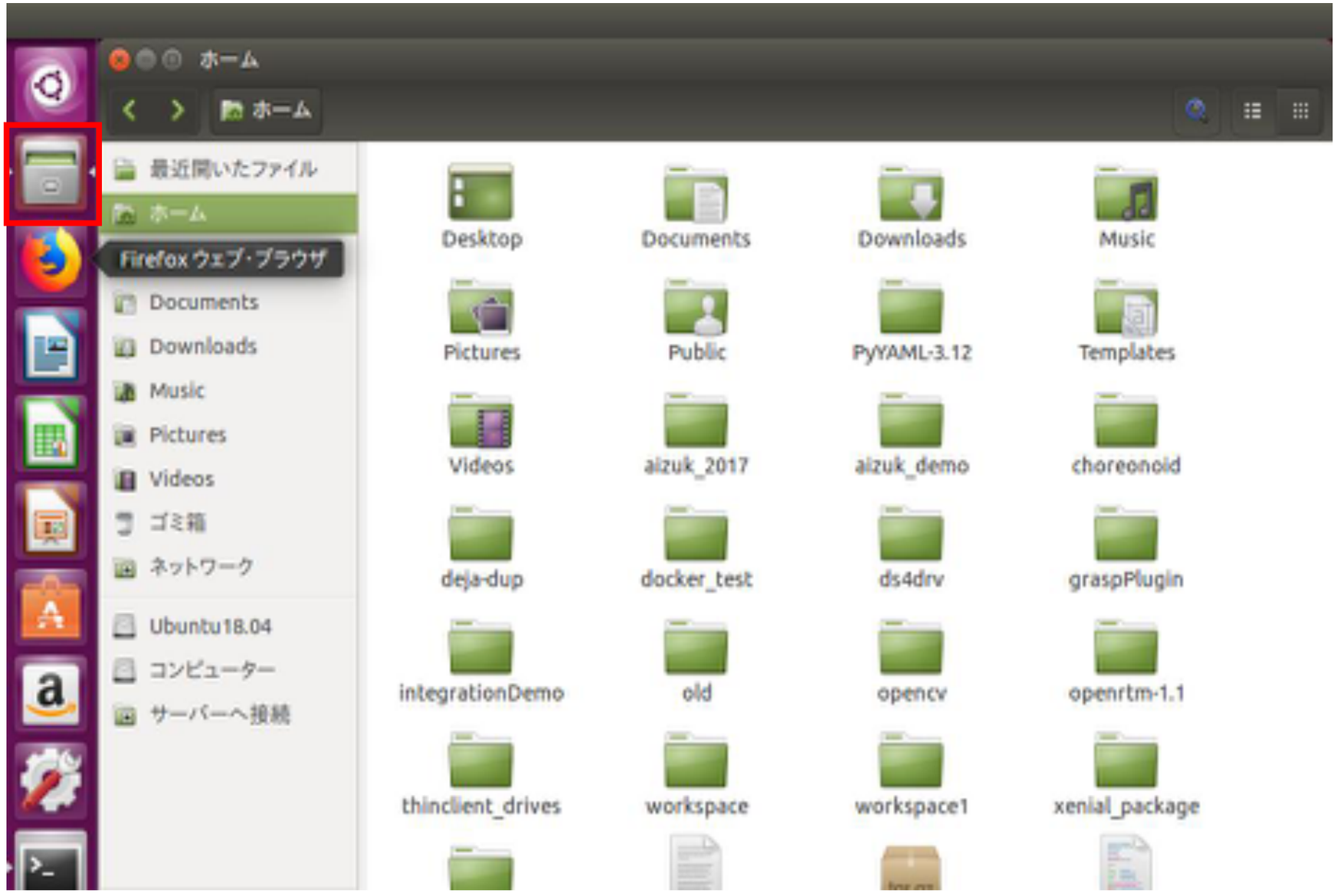
```
$ choreonoid [ プロジェクトファイルパス ]
```

❖ 例えば, ユーザ名が『user1』,
プロジェクトファイルパスが『/home/user1/choreonoid/sample/SimpleController/Tank.cnoid』の場合,

```
$ choreonoid /home/user1/choreonoid/sample/
SimpleController/Tank.cnoid
```

ファイルマネージャによる起動

❖ ランチャーから『ファイル』をクリックし起動



ファイルマネージャによる起動

- ファイルマネージャーを起動し，以下のディレクトリに移動
- 『choreonoid』をダブルクリックし起動
- Choreonoid格納場所は，
  - “CMAKE_INSTALL_PREFIX” オプションで指定したディレクトリ
    - デフォルトの場合，“**/usr/local/bin**”（非推奨）
    - ”/home/ユーザ名/choreonoid/program”に変更した場合，”**/home/ユーザ名/choreonoid/program/bin**”(推奨)
    - make installを行っていない場合，インストールディレクトリに『choreonoid』ファイルが存在しないので注意
  - “**/home/ユーザ名/choreonoid/build/bin**”（必ず存在）

Choreonoidの画面構成

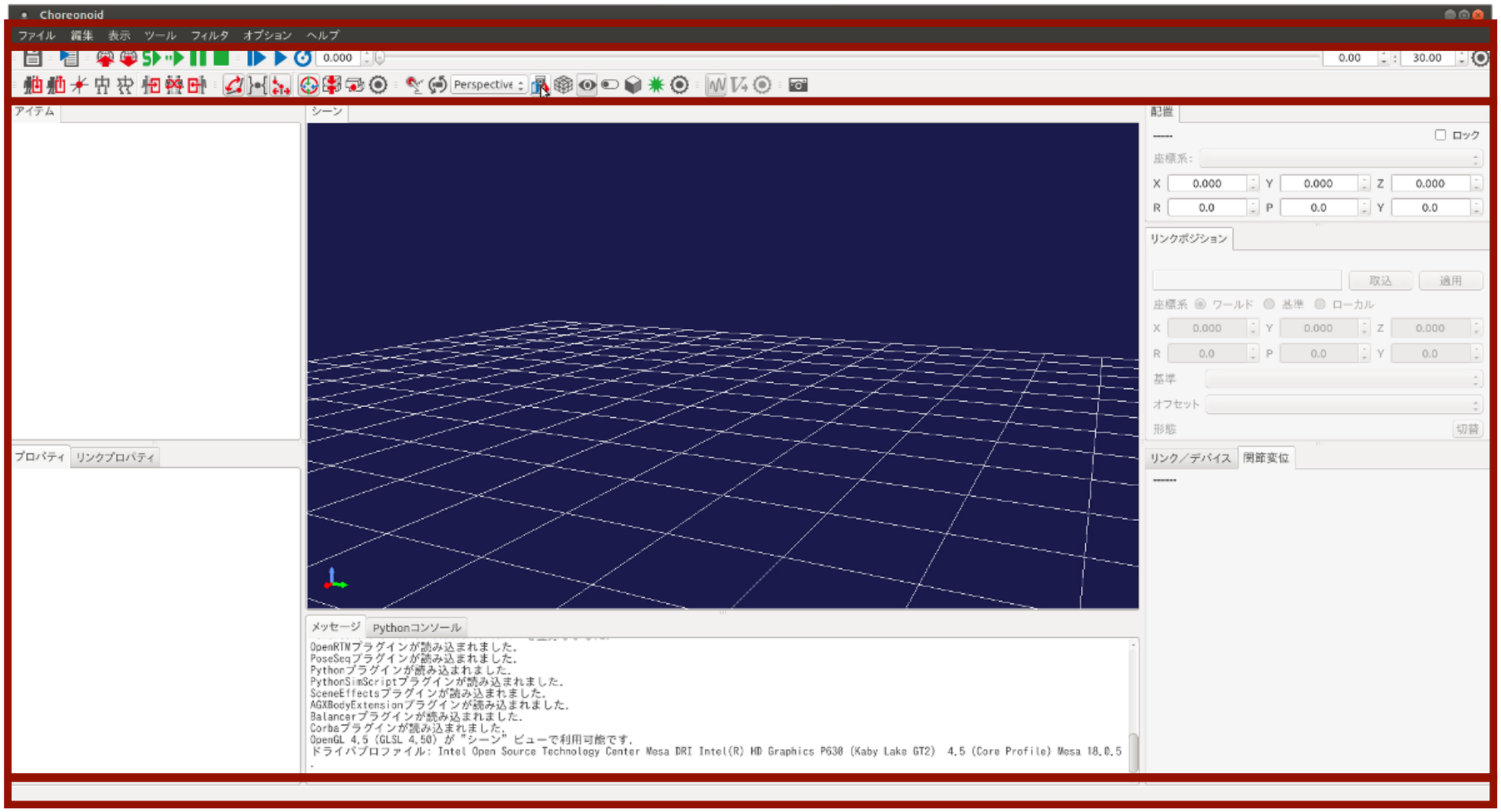
❖ 『メイン画面』は以下の構成となる

メインメニュー

ツールバー領域

ビュー領域

ステータスバー



メインメニュー

❖ メインメニューは各種操作や設定を一覧で表示

❖ ワールド（仮想世界）の生成やモデルの追加

❖ 作成した仮想環境（プロジェクト）の読み込み・保存

❖ ビューの生成・削除

❖ シミュレーションの録画

ツールバー

❖ ツールバーは頻繁に使用するボタン等を並べたもので,
ワンクリックするだけで実行可能

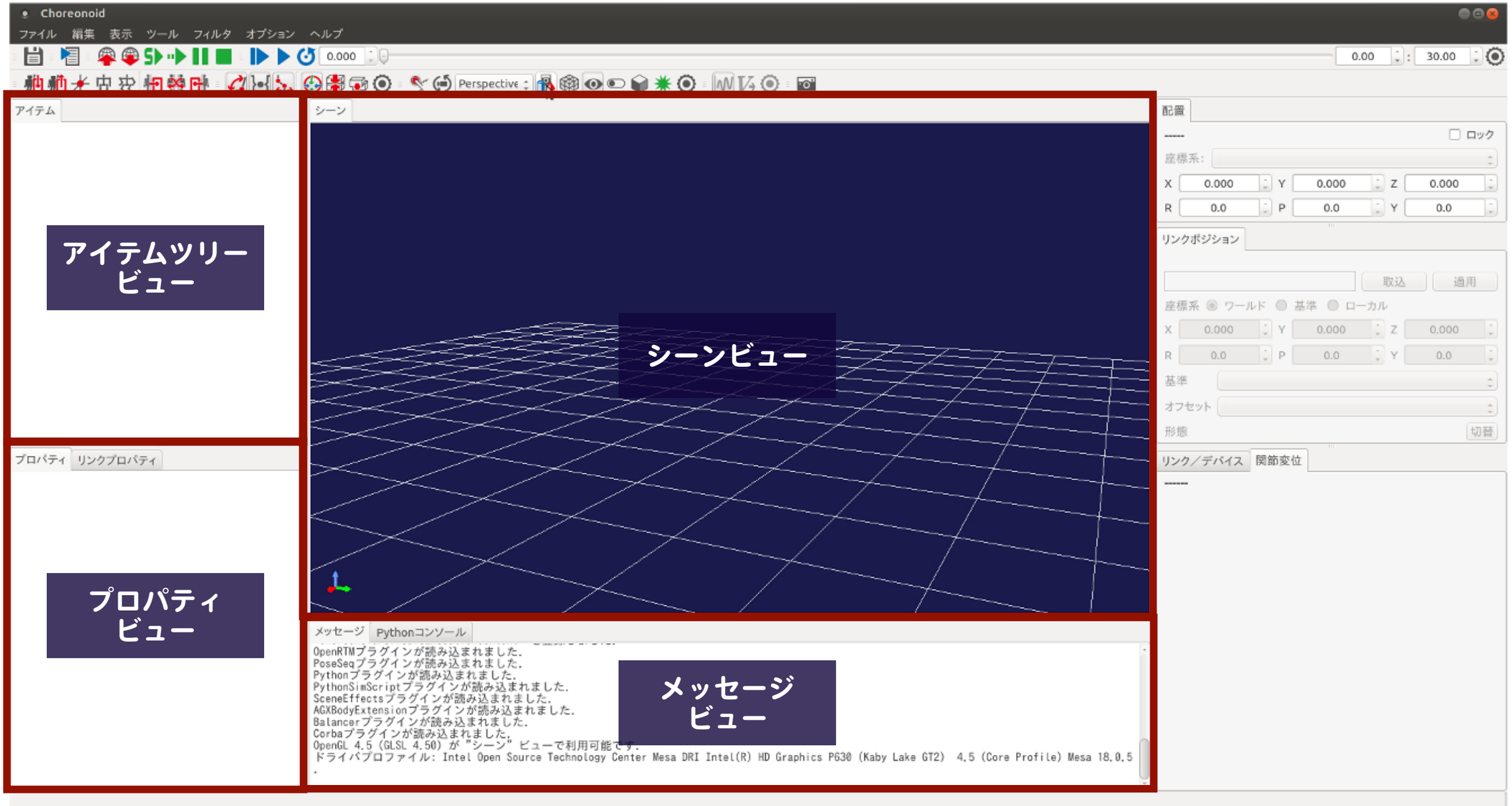
種類	説明
ファイルバー 	プロジェクトファイルを上書き保存
タイムバー 	シミュレーション時間の操作
シーンバー 	シーンビューの設定変更やカメラ表示切替
シミュレーションバー 	シミュレーションの実行・停止

ビュー

❖ ビューはパネル状の領域となっており，
各種データの表示・編集を行える

種類	説明
アイテムツリービュー	プロジェクトを構成するアイテムを 木構造で表示するビュー
アイテムプロパティビュー	アイテムのプロパティを閲覧・編集するためのビュー
メッセージビュー	Choreonoidからのメッセージが出力されるビュー
シーンビュー	各種データを3次元コンピュータグラフィックス (3D-CG)によって表示するビュー

各ビューの配置


Choreonoid
ファイル 編集 表示 ツール フィルタ オプション ヘルプ
0.000
0.00
30.00
Perspective
アイテム
アイテムツリービュー
シーン
シーンビュー
配置
ロック
座標系:
X 0.000 Y 0.000 Z 0.000
R 0.0 P 0.0 Y 0.0
リンクポジション
取込
適用
座標系 ワールド 基準 ローカル
X 0.000 Y 0.000 Z 0.000
R 0.0 P 0.0 Y 0.0
基準
オフセット
形態
切替
リンク／デバイス
関節変位
プロパティ
リンクプロパティ
プロパティビュー
メッセージ
Pythonコンソール
OpenRTMプラグインが読み込まれました.
PoseSeqプラグインが読み込まれました.
Pythonプラグインが読み込まれました.
PythonSimScriptプラグインが読み込まれました.
SceneEffectsプラグインが読み込まれました.
AGXBodyExtensionプラグインが読み込まれました.
Balancerプラグインが読み込まれました.
Corbaプラグインが読み込まれました.
OpenGL 4.5 (GLSL 4.50) が "シーン" ビューで利用可能です.
ドライバプロファイル: Intel Open Source Technology Center Mesa DRI Intel(R) HD Graphics P630 (Kaby Lake GT2) 4.5 (Core Profile) Mesa 18.0.5
.
メッセージビュー

プロジェクトとアイテム

- プロジェクト
  - シミュレーションに必要なデーター式をChoreonoid上で扱えるようにまとめたもの

- アイテム
  - プロジェクトを構成する各々のデータのうち,
    ユーザの操作対象となるもの

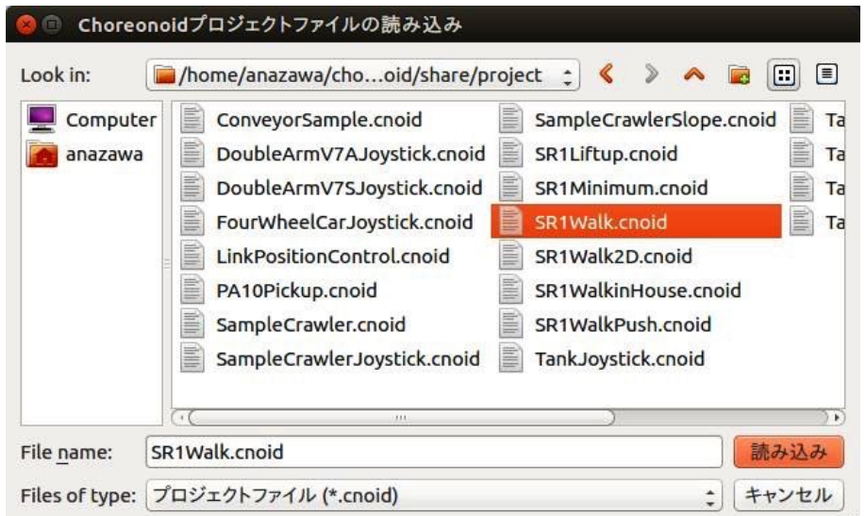
- Choreonoidのプロジェクトファイルは,
  拡張子が『cnoid』形式

プロジェクトとアイテム

- プロジェクト: **複数のアイテムで構成される**
  - 右上の例では，4種類のアイテムで構成
    - ワールドアイテム (World)
    - ボディアイテム (SR1，Floor)
      - SR1：ロボット，Floor：環境
    - シンプルコントローラアイテム (SR1WalkControllrer)
    - シミュレータアイテム (AISTSimulator)

プロジェクトの読み込み

❖ メインメニューから『ファイル』→
『プロジェクトの読み込み』を選択



❖ **『Choreonoidプロジェクト**
ファイルの読み込み』ダイアログが表示

❖ ダイアログ上で読み込みたいプロジェクト(.cnoid)を指定

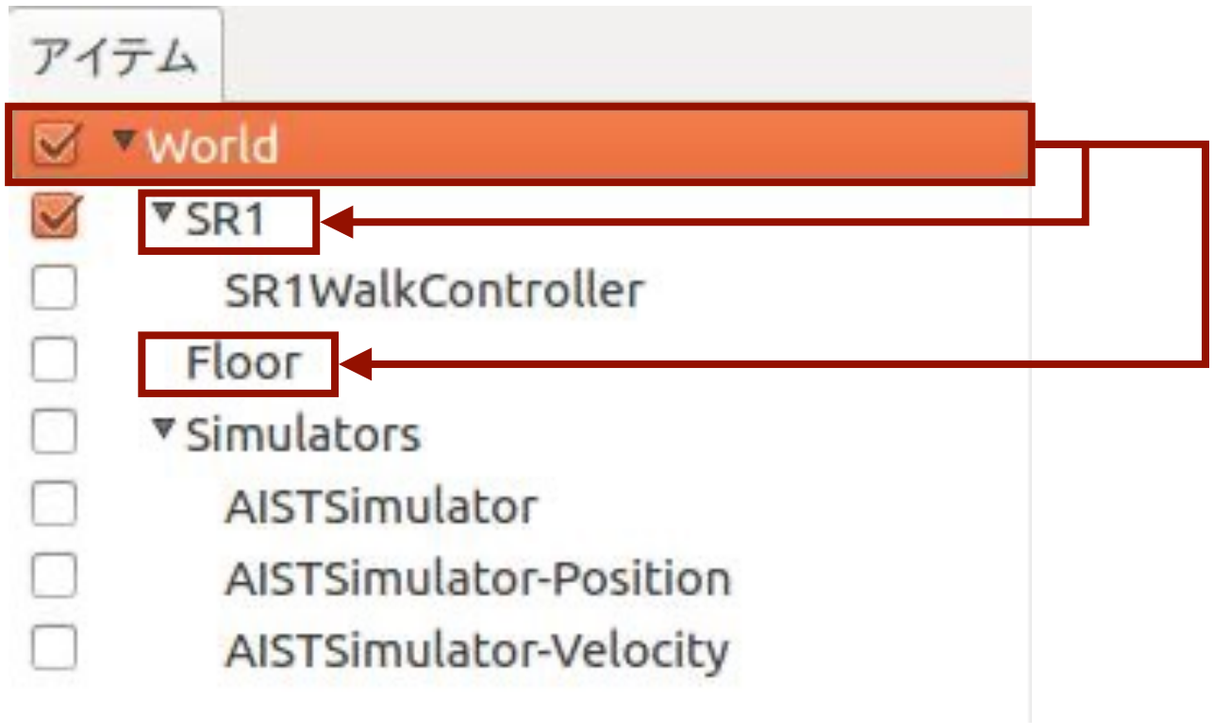
❖ 例: sample/Simplecontroller/SR1Walk.cnoidを選択

❖ サンプルプロジェクト格納場所

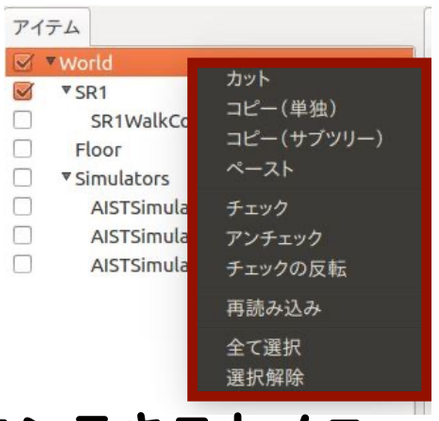
❖ ~/choreonoid/sample/SimpleController

アイテムツリービュー

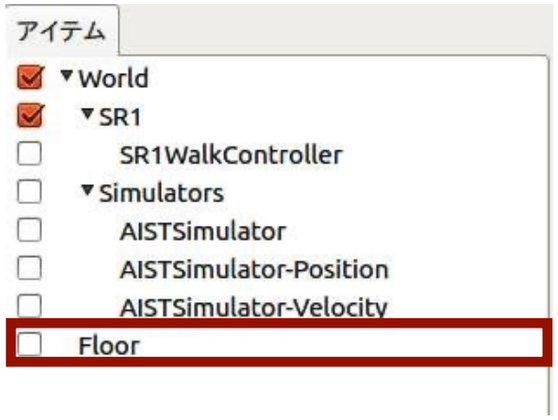
- ワールド，ロボットモデル，シミュレータアイテムなどを木構造で表示するビュー
  - 木構造では，**どのアイテムの小アイテムとするかが重要**
  - ロボット (SR1)，環境モデル (Floor) は，ワールド (World) の小アイテムとして配置



アイテムツリービュー

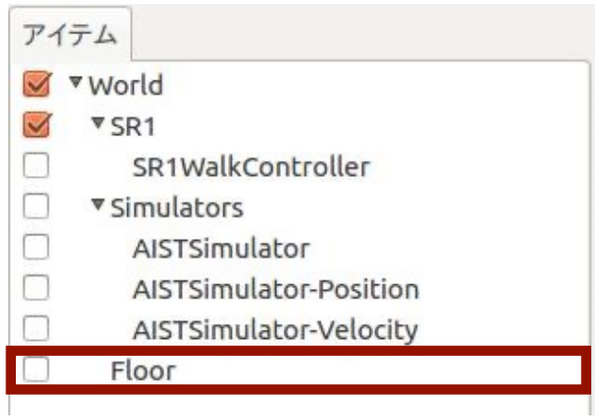


- アイテムツリービュー上で右クリックすることで，**コンテキストメニュー**カット，コピー，ペーストなどを行える
- アイテムツリービュー上でアイテムをドラッグすることで，**アイテムの位置を移動できる**
- アイテムの横のチェックボックスで，**表示・非表示の切り替えが可能**



FloorをWorldの子アイテムとなる位置に移動

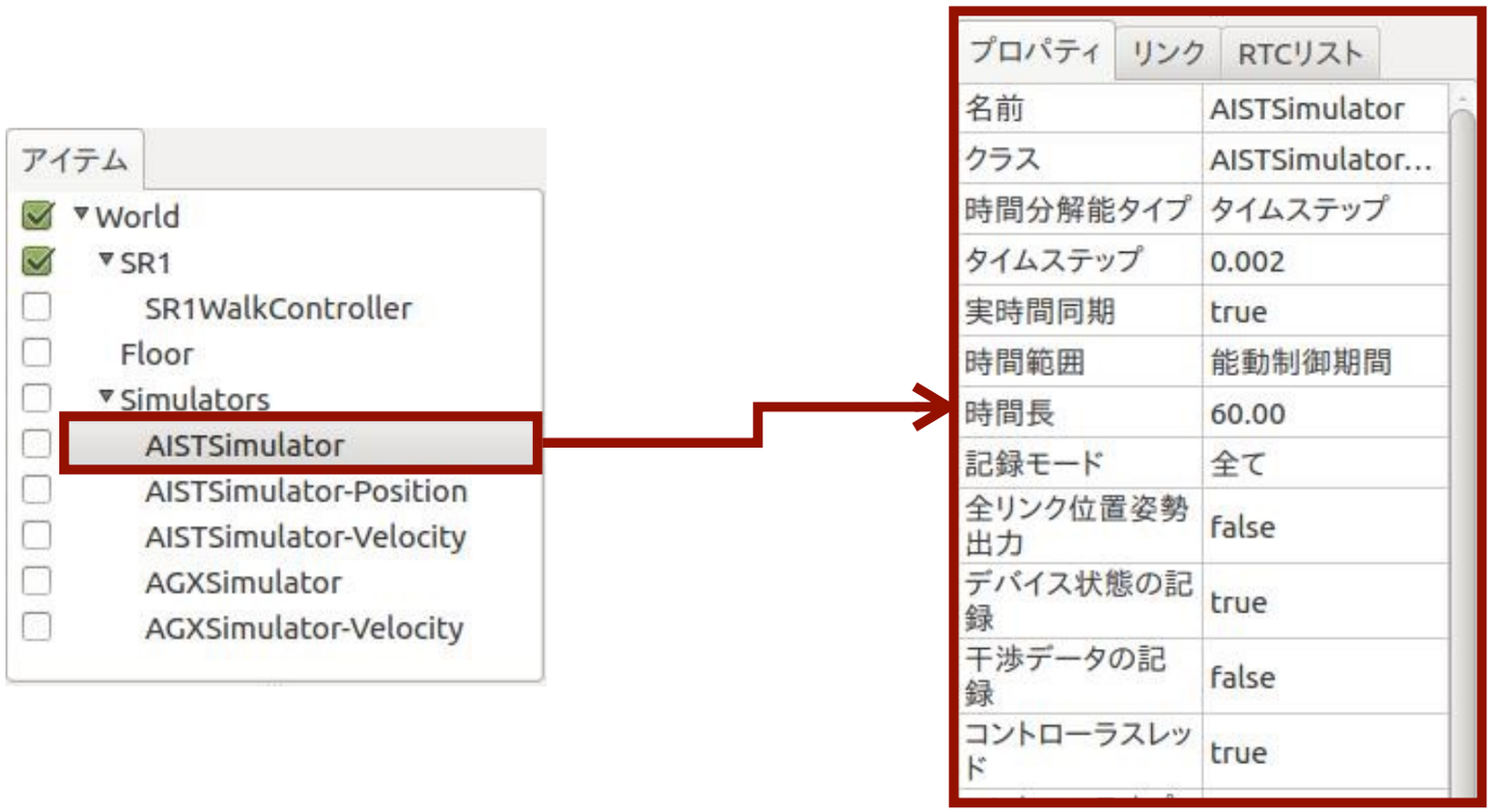
Floorをドラッグし，Worldの位置に持っていく



FloorがWorldの小アイテムとなっている

アイテムプロパティビュー

- アイテムツリービュー上で,
  選択しているアイテムの情報を閲覧・編集できる
- アイテムプロパティビュー上でプロパティ値を
  ダブルクリックすると編集が行える
- **プロパティ値の編集を行えない項目あり**



メッセージビュー

❖ モデルファイル等の読み込み状況や
シミュレーションの実行時間の確認が行える

メッセージ

```
FolderItem "Simulators" を読み込み中
AISTSimulatorItem "AISTSimulator" を読み込み中
AISTSimulatorItem "AISTSimulator-Position" を読み込み中
AISTSimulatorItem "AISTSimulator-Velocity" を読み込み中
8 / 8 のアイテムが読みこまれました.
プロジェクト "/home/anazawa/choreonoid/share/project/SR1Walk.cnoid" を完全に読み込みました.
SR1WalkControllerのコントローラモジュール"/usr/local/lib/choreonoid-1.7/simplecontroller/SR1WalkPa
tternController"をロード中･･･OK!
コントローラインスタンスを生成しました.
SR1WalkPatternController: torque control mode.
AISTSimulatorによるシミュレーションを開始しました.
SR1WalkControllerのコントローラモジュール"/usr/local/lib/choreonoid-1.7/simplecontroller/SR1WalkPa
tternController"をアンロードしました.
AISTSimulatorによるシミュレーションが6.982秒時点で終了しました.
計算時間: 6.981 [s], 計算時間 / シミュレーション時間 = 0.999857
```

モデルファイルの読み込み状況

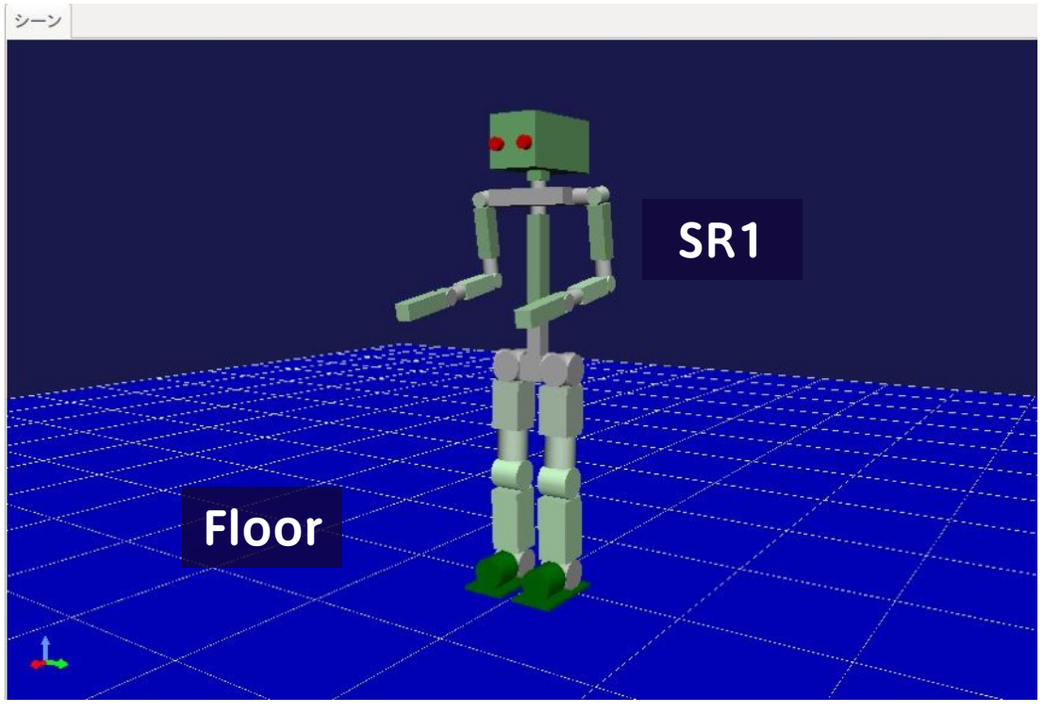
シミュレーションの実行時間

シーンビュー | 座標系

❖ 各種データを**三次元コンピュータグラフィックス (3DCG) によって表示する**ビュー

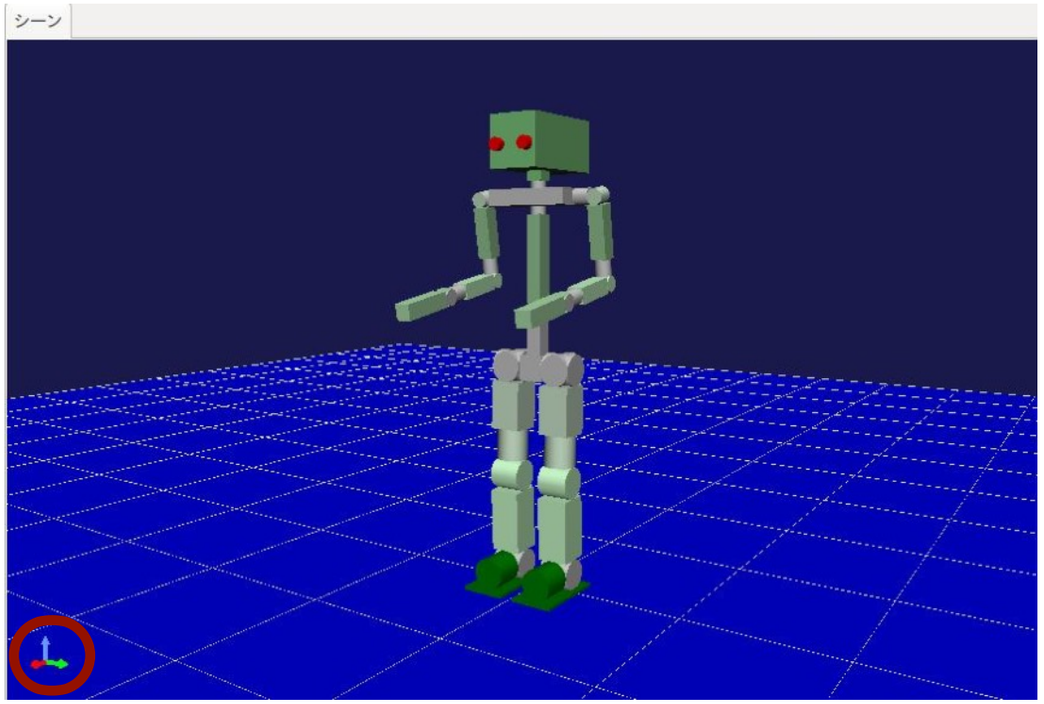
❖ ロボットや環境モデルを描画

❖ 例：ロボット(SR1)と環境モデル(Floor)を表示



シーンビュー | 座標系

- 左下にある3つの矢印の組は，現在の視点から見た3次元空間の座標系
  - （赤色軸，緑色軸，青色軸） ⇒（X軸，Y軸，Z軸）に対応
- Choreonoidでは, 鉛直上向きがZ軸の正方向
- 格子状のグリッド線は，Z=0 の面を表す



シーンビュー | 閲覧モード

❖ 表示されているロボットや環境モデルをドラッグすることで，
視点変更が可能

❖ マウスホイールをスクロールすると，モデルの**拡大・縮小可能**

❖ シーンビュー上で『右クリック』，
または『Esc』キーを押すことで，
閲覧モードから編集モードに切り替えが可能

シーンビュー | 編集モード

- 編集モードでロボットモデルを
  左クリックすると**姿勢や位置の変更を行える**
  - **静的モデルの場合，位置の変更はできない**
- 静的モデルの確認方法
  - アイテムツリービューで
    確認したいモデルを選択
  - アイテムプロパティビューの
    静的モデルを確認
  - **true: 静的モデル**
  - **false: 動的モデル**

名前	Tank
クラス	BodyItem
モデル名	Tank
リンク数	5
関節数	2
デバイス数	6
ルートリンク	CHASSIS
ベースリンク	none
質量	30.000
静的モデル	false
干渉検出	true
自己干渉検出	false
編集可能	true
ファイル	Tank.body

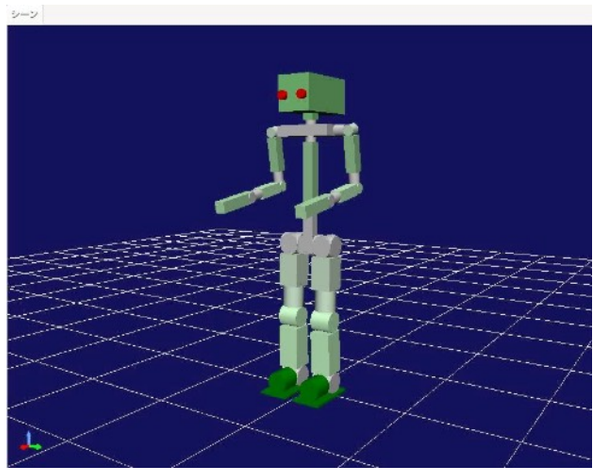
シーンビュー | シーンバーの機能

❖ シーンバーの機能を使用することで，以下のことが可能

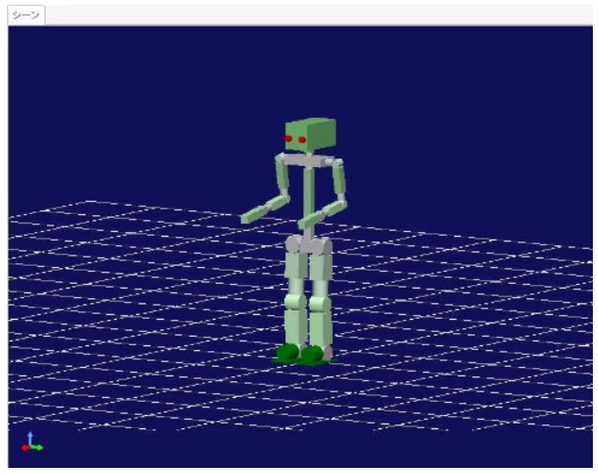
機能	説明
編集モードの切り替え 	閲覧・編集モードの切り替え
視点操作モードの切り替え 	物体を中心とした操作から 視点を中心とした操作への切り替え
描画用カメラ選択 	ロボットに搭載されているカメラ視点切り替え Perspective（透視投影） Orthographic（正射影）
視点回復 	シーン上の全ての物体が見えるように調節
干渉線の表示 	物体同士の干渉を黄緑色の線で表示
ワイヤフレーム表示 	シーンがワイヤフレームで描画される
設定ダイアログ 	シーンの描画や挙動をダイアログで設定

シーンビュー | 描画用カメラ選択

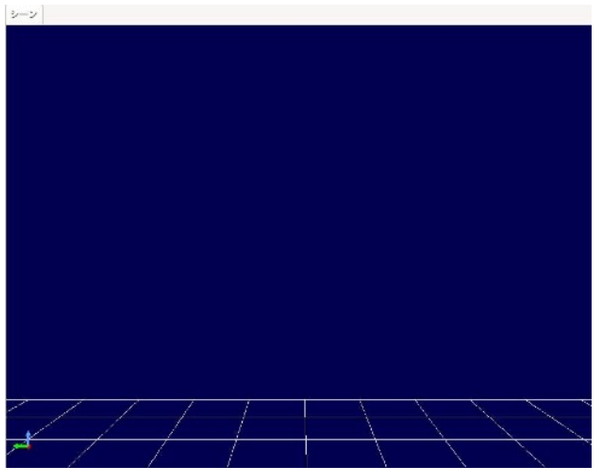
- Perspective (透視投影) カメラ：
  - **遠近感のついた画像を表示**
- Orthographic (正射影) カメラ：
  - **遠近感を排除した正射影の画像を表示**
- ロボット搭載カメラ：
  - **ロボットに付けているカメラ視点の画像を表示**



Perspective（透視投影）



Orthographic（正射影）



ロボットの左目に付けているカメラ

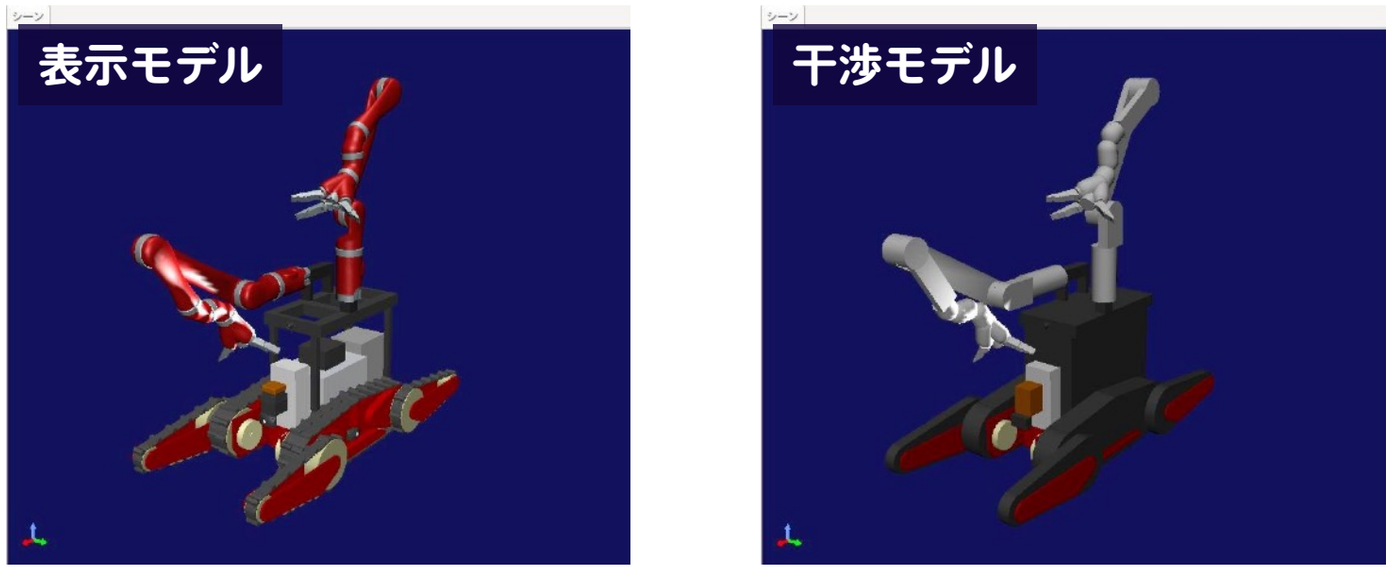
シーンビュー | 設定ダイアログ

❖ シーンバーの設定ダイアログで，以下の項目を設定可能

機能	説明
ヘッドライト	常時視点位置から照射されるライトをONにする
ワールドライト	シーン上に固定されたライト (通常上方から照射) のON/OFFを切り替える
追加のライト	シーン上に読み込まれたモデルにライトがついている場合，そのON/OFFを切り替える
背景色	シーン上で何も物体がない領域の色を設定
床グリッド線の表示	床グリッド線の表示切替，グリッドの大きさ，色を設定

シーンビュー | 表示モデルと干渉モデル

- 表示モデル
  - 3Dモデリングツールを用いて作成した**複雑な形状 (多角形，穴のあるものなど) のモデルを表示**
- 干渉モデル
  - **四角形や円柱などを組み合わせて作成**
  - 物体同士の干渉など物理計算に使用



`sample/SimpleController/AizuSpiderDS.cnoid`

シーンビュー | 表示モデルと干渉モデル

- シーンバーにより，表示モデルと干渉モデルの切り替えが可能
  -  : 表示モデルを表示
  -  : 表示中のモデル形式の切り替え
    - 切り替えパターン：
      - 表示 ⇒ 干渉
      - 干渉 ⇒ 表示
      - (表示 + 干渉) ⇒ 非表示
      - 非表示 ⇒ (表示 + 干渉)
  -  : 干渉モデルを表示

ステータスバー

- Choreonoidで現在進行中の処理内容を簡潔なメッセージで表示

- シーンビュー上のグローバル座標位置を表示

- ステータスバーは『メインメニュー』→『表示』で，表示・非表示の切り替えが可能

プロジェクトの作成

プロジェクトの作成 | ワールドの生成

❖ メインメニューから『ファイル』→
『新規』→『ワールド』を選択

❖ **『新しいワールドアイテムの生成』
ダイアログが表示**

❖ ダイアログ上でワールド名を入力し，
『生成』ボタンを押す

プロジェクトの作成 | モデル読込

❖ アイテムツリービュー上の『World』を選択

❖ メインメニューから『ファイル』→『読み込み』→
『ボディ』を選択

❖ 『ボディの読み込み』ダイアログが表示

❖ ダイアログ上でロボットモデルを選択し,
『読み込み』ボタンを押す

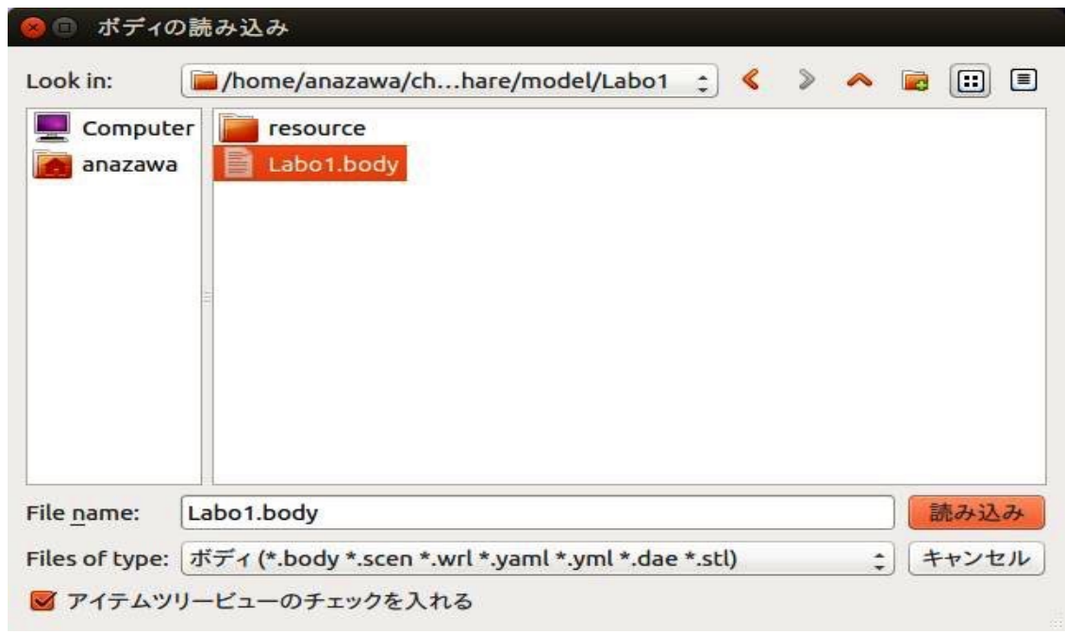
プロジェクトの作成 | モデル読込

- 読み込めるモデルファイルの形式は以下となる
  - **.body（Choreonoidモデル形式）**
  - .wrl（VRML形式）
  - .dae（COLLADA形式）
  - .stl

share/model/Tank/Tank.bodyを選択

プロジェクトの作成 | 環境モデル読込

- アイテムツリービュー上の『World』を選択
- メインメニューから『ファイル』→
  『読み込み』→『ボディ』を選択
- 『ボディの読み込み』ダイアログが表示
- ダイアログ上で環境モデルを選択し，
  『読み込み』ボタンを押す



share/model/Labo1/Labo1.bodyを選択

プロジェクトの作成 | コントローラ読込

- アイテムツリービュー上のロボットモデルを選択
- メインメニューから『ファイル』→『新規作成』→『シンプルコントローラ』を選択
- 『**新しいシンプルコントローラアイテムの生成**』ダイアログが表示
- ダイアログ上で名前を入力し『生成』ボタンを押す
  - **コントローラ名は任意の名前で良いが，基本的には英数字，”-”（ハイフン），”_”（アンダーライン）で命名を推奨**

プロジェクトの作成 | コントローラ読込

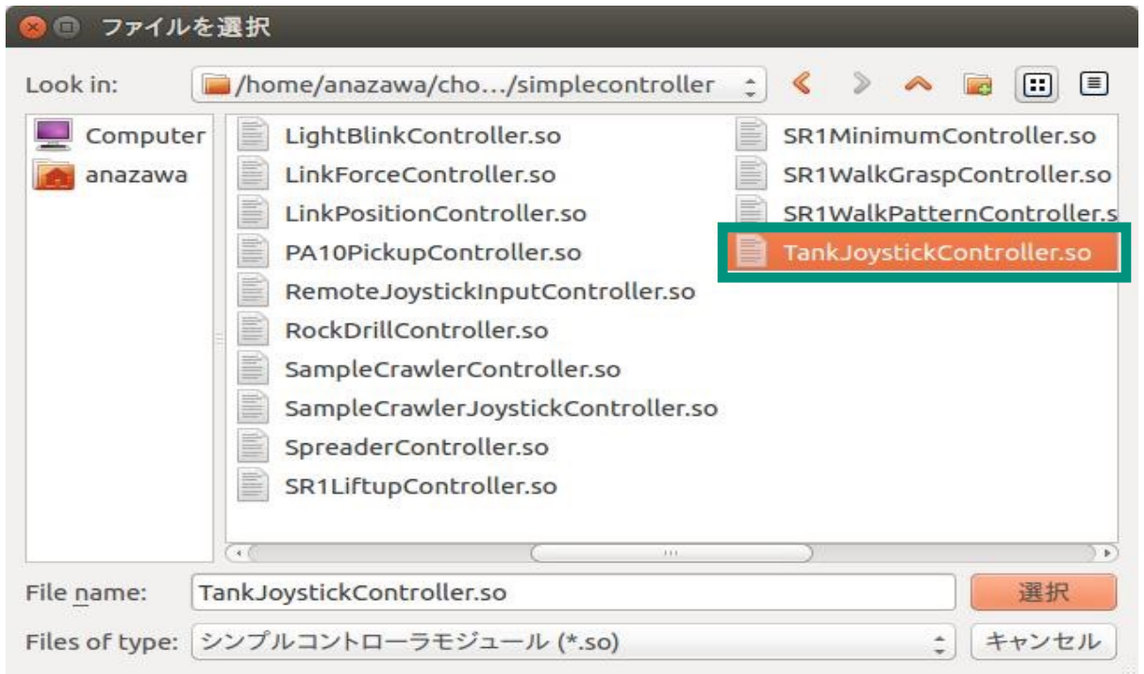
- アイテムツリービューでコントローラを選択し，
  アイテムプロパティビューの『コントローラモジュール』を
  ダブルクリック
- 『コントローラモジュール』の右端のフォルダマーク を
  選択

**コントローラモジュールを
ダブルクリック**

フォルダマークを選択

プロジェクトの作成 | コントローラ読込

- 対象のコントローラ（拡張子：**.so**）を選択
  - **コントローラはロボット毎に異なるため，**
    別のロボットのコントローラを選択しないように注意する
- コントローラモジュールは以下のディレクトリに格納
  - choreonoid/build/lib/choreonoid-1.8/simplecontroller



TankJoystickController.soを選択

モデルとコントローラの対応表

モデル	コントローラ
AizuSpiderNS.body	AizuSpiderController.so
AizuWheel.body	AizuWheelController.so
DoubleArmV7S.body	DoubleArmV7Controller.so
JACO2.body	JACO2Controller.so
Quadcopter.body	QuadcopterController.so

- AizuSpiderモデルは，AizuSpderNS.body以外に，以下の2モデル存在
  - AizuSpiderSS.body（単腕付き）
  - AizuSpiderDS.body（双腕付き）

プロジェクトの作成 | シミュレータ読込

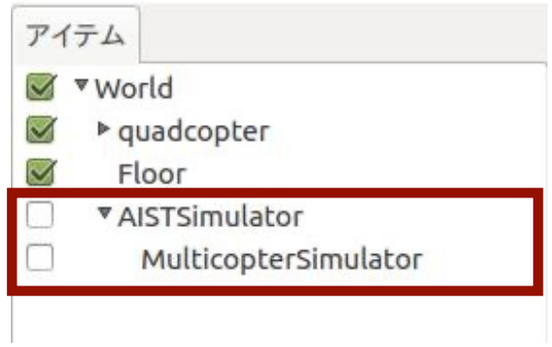
- アイテムツリービュー上の『World』を選択
- メインメニューから『ファイル』→『新規』→『AISTシミュレータ』を選択
- 『**新しいAISTシミュレータアイテムの生成**』ダイアログが表示
- ダイアログ上で名前を入力し，『生成』ボタン押す

プロジェクトの作成 | シミュレータ設定

- 以下のプロパティを変更
  - 静止摩擦係数：1.0 → 0.5
    - 静止摩擦力が最大の時の摩擦係数
  - 動摩擦係数：1.0 → 0.5
    - 物体が動く時の摩擦係数
  - 接触間引き距離：0.005 → 0.01
    - 指定した距離内に複数接触点がある場合に，接触点を1つに間引く
  - 接触補正速度係数：1 → 30
    - 物体同士が接触している時の，めり込みを補正する速度係数

プロジェクトの作成 | シミュレータ設定

- MulticopterSimulator
  - ドローンモデルを使用する場合，
    AISTシミュレータだけでは動作しない
  - AISTシミュレータの子アイテムとして，
    MulticopterSimulatorの追加が必要
- アイテムツリービュー上で『AISTSimulator』を選択
- 『ファイル』→『新規』→『MulticopterSimulator』を選択
- 『**新しいMulticopterSimulatorItemの生成**』
  ダイアログが表示
- ダイアログ上で名前を入力し，『生成』ボタンを押す

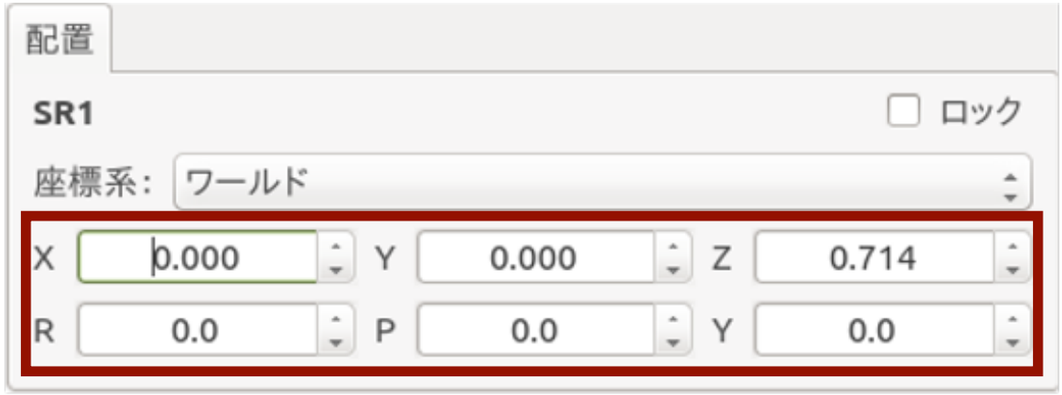


MulticopterSimulatorの追加

プロジェクトの作成 | モデルの移動

❖ アイテムツリービュー上で
移動するモデルの選択を行う

❖ **ボディ/リンクビュー**に
表示されている（X，Y，Z）に
任意の値（単位：m）を
入力することでモデルが移動

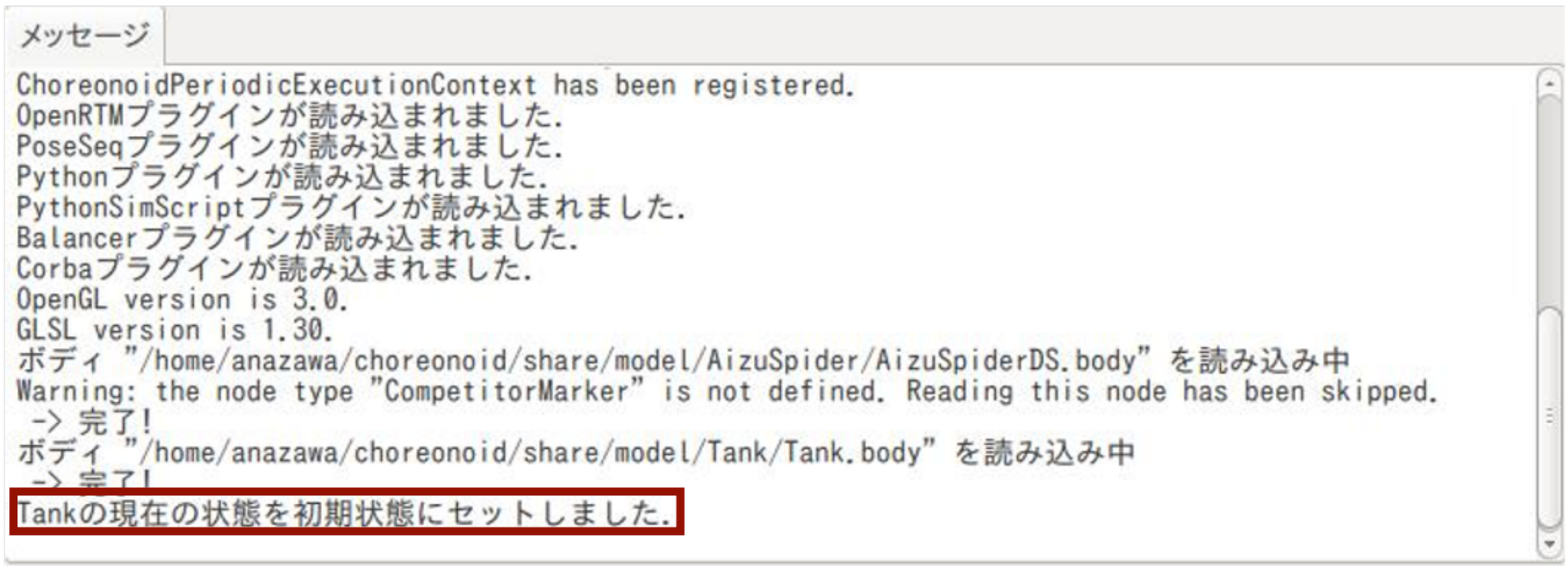


❖ （R，P，Y）に角度（単位：度) を
入力するとモデルが回転

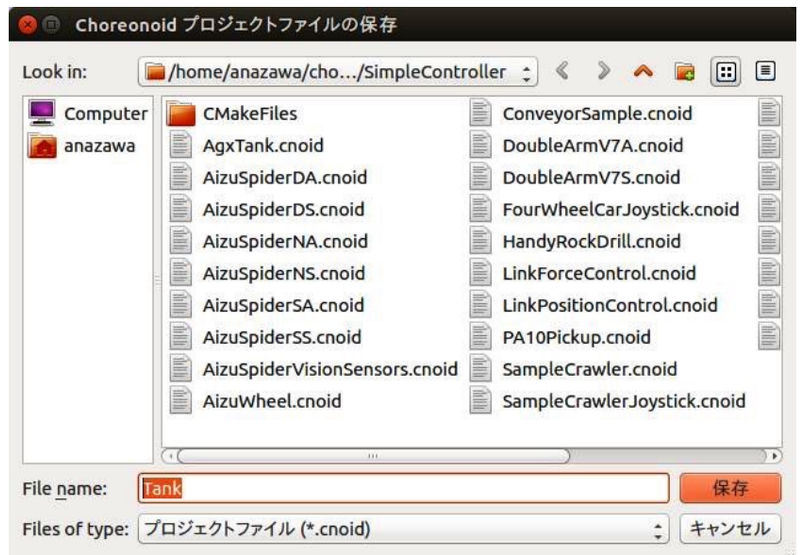
❖ **微調整を行う場合などに使用**

プロジェクトの作成 | モデルの初期設定

- ❖ 移動を行ったモデルをアイテムツリービュー上で選択
- ❖ シミュレーションバーの『ワールド初期状態設定』ボタン  をクリックすることで，**モデルの初期位置を登録**
- ❖ 登録に成功すると，その旨がメッセージビューに表示



プロジェクトの作成 | 保存



- プロジェクトを保存することで，
  次回からプロジェクト作成の
  操作をせずに，シミュレーションが可能
- 『メインメニュー』→
  『名前を付けてプロジェクトを保存』を選択
- 『**Choreonoidプロジェクトファイルの保存**』
  ダイアログが表示
- ファイル名を入力し，『保存』ボタンを押す
- 今回ファイル名を『TankSample.cnoid』とする

シミュレーションの実行・停止

❖ シミュレーションバー内の以下のボタンで,
シミュレーションの実行・停止が可能

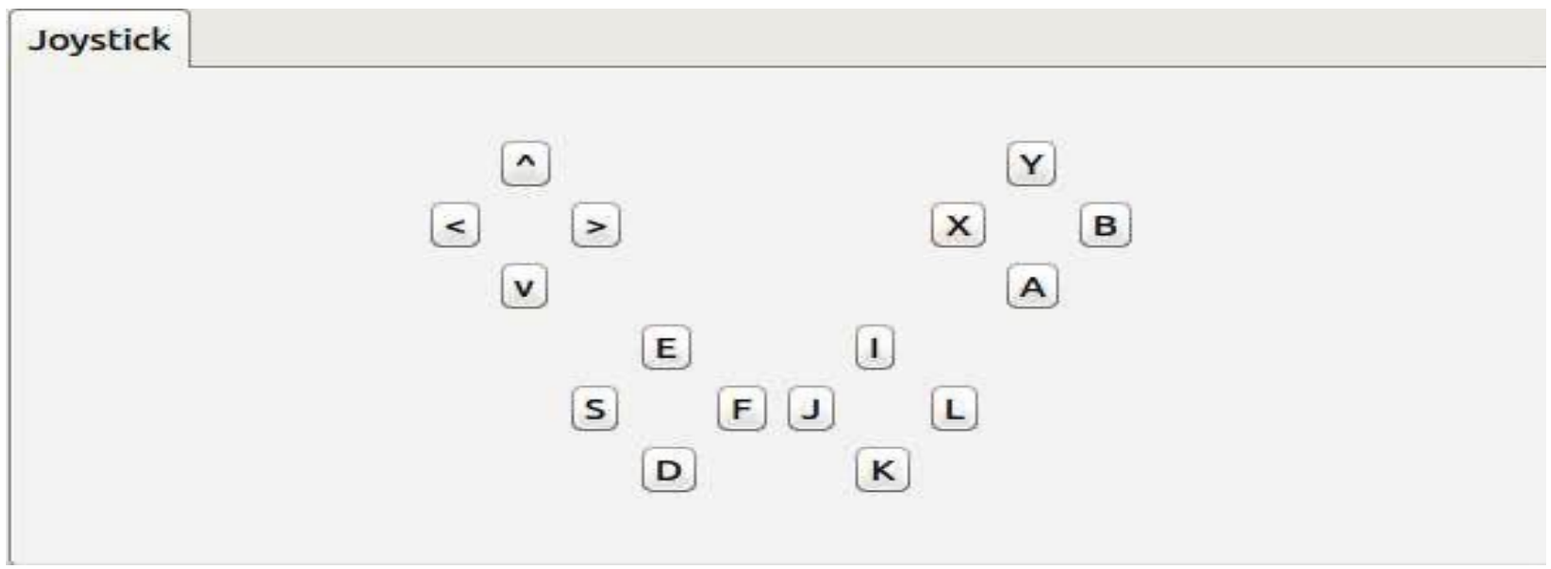
ボタン名	説明
初期位置からの シミュレーション 	初期位置からシミュレーションを開始する
現在位置からの シミュレーション 	現在位置からシミュレーションを開始する
一時停止 	シミュレーションを一時停止する 「現在位置からのシミュレーション」ボタンで シミュレーションを再開できる
停止 	シミュレーションを停止する

モデルの操作 | 仮想ジョイスティック

❖ ゲームパッドを持っていない場合，
仮想ジョイスティックビューで代用可能

❖ 『メインメニュー』→『表示』→『ビューの表示』→
『Joystick』を選択することで，Joystickビューを表示可能

❖ 仮想ジョイスティックでの操作は，シミュレーション実行後に
Joystickビューをクリックし，**アクティブ状態にする**



仮想ジョイスティック操作例

- メインメニューから『ファイル』→
  『プロジェクトの読み込み』を選択
- 今回，作成した『**TankSample.cnoid**』を読み込む
- Joystickビューが表示されている状態で，
  シミュレーションを開始する
- Tankモデル操作方法
  - E，D：前進後退
  - I，K：砲塔の上下移動
  - S，F：左右の旋回
  - J，K：砲塔の左右移動
  - A：ライトの点灯

モデルの操作 | ゲームパッド

❖ 基本的にTankモデルなどは,
ゲームパッド(DUALSHOCK 4) をPCに接続することで,
操作可能

❖ ゲームパッドは**シミュレーション実行前にPCに接続**

❖ シミュレーションを実行すると,
ゲームパッドでモデルの操作を行える

ゲームパッドの接続確認

- jstestコマンドを使用するために，
  joystickパッケージをインストール

```
sudo apt install joystick
```

- ゲームパッドが反応することを確認するため，
  次のコマンドを実行

```
jstest /dev/input/js0 # 接続した順番で採番js0，js1，…
```

```
Driver version is 2.1.0.
Joystick (Sony Interactive Entertainment Wireless Controller) has 8 axes (X, Y, Z, Rx, Ry, Rz, Hat0X, Hat0Y)
and 14 buttons (BtnX, BtnY, BtnZ, BtnTL, BtnTR, BtnTL2, BtnTR2, BtnSelect, BtnStart, BtnMode, BtnThumbL, BtnThumbR, ?, ?).
Testing ... (interrupt to exit)
Axes:  0:     0  1:     0  2:     0  3:-32767  4:-32767  5:     0  6:     0  7:     0 Buttons:  0:off  1:off  2:off  3:off  4:off  5:off  6:off  7:off  8:off  9:off 10:off 11:off 12:off 13:off
```

ゲームパッドの操作例

- メインメニューから『ファイル』→
  『プロジェクトの読み込み』を選択
- 今回，作成した『TankSample.cnoid』を読み込む
- ゲームパッド(DUALSHOCK 4)をPCに接続し，
  シミュレーションを開始
- Tankモデル操作方法
  - 左ジョイスティック 上 / 下：前進後退
  - 左ジョイスティック 左 / 右：左右の旋回
  - 右ジョイスティック 上 / 下：砲塔の上下移動
  - 右ジョイスティック 左 / 右：砲塔の左右移動
  - xボタン：ライトの点灯

課題 | プロジェクト作成

- プロジェクト作成で行った手順を参考に，
  『ロボットとコントローラの対応表』に記載されている
  ロボットのプロジェクトを作成してください
- 車体とアームの操作があるモデルは，車体操作コントローラの
  子アイテムとしてアーム操作コントローラを配置
- JACO2コントローラを追加した場合，
  コントローラオプションの編集が必要
  - AizuWheel.body：ARM_
  - AizuSpiderSS.body：なし
  - AizuSpiderDS.body：1つ目：ARM1_，2つ目：ARM2_

課題 | プロジェクト作成

❖ すべて作成できたら，実際にシミュレーションを実行し，
ロボットを操作してみてください

❖ 操作方法は，『サンプルロボット操作マニュアル.docx』参照

❖ **AizuWheel.bodyにJACO2が付いているので，
JACO2のプロジェクトは作成不要**

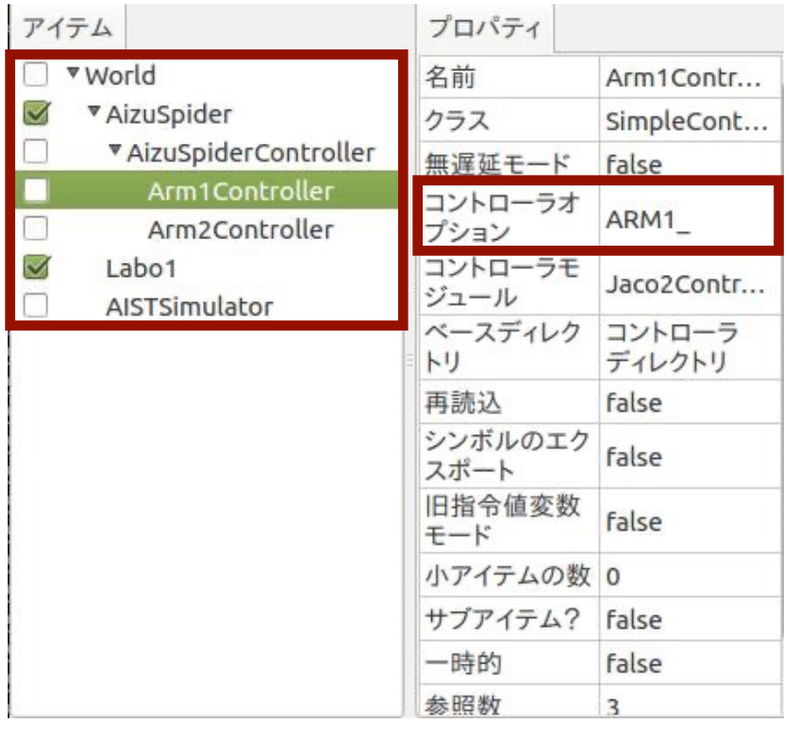
モデルとコントローラの対応表

モデル	コントローラ
AizuSpiderNS.body	AizuSpiderController.so
AizuWheel.body	AizuWheelController.so
DoubleArmV7S.body	DoubleArmV7Controller.so
JACO2.body	JACO2Controller.so
Quadcopter.body	QuadcopterController.so

- AizuSpiderモデルは，AizuSpderNS.body以外に，以下の2モデル存在
  - AizuSpiderSS.body（単腕付き）
  - AizuSpiderDS.body（双腕付き）

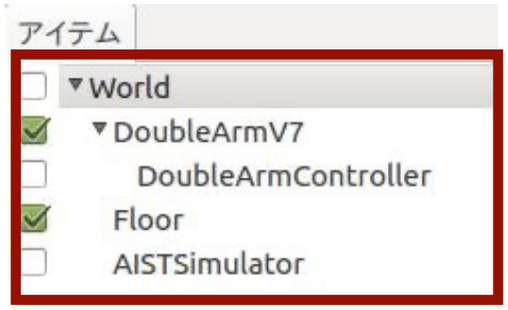
課題の解答

❖ アイテムツリービューの構成とコントローラオプション

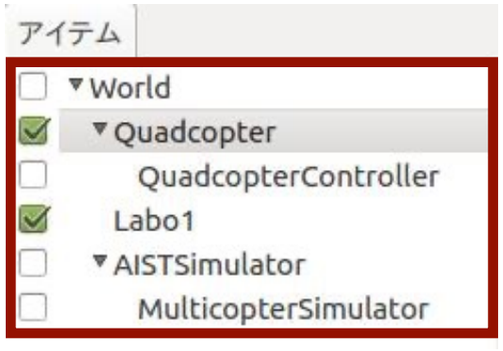


AizuSpiderDS

AizuWheel



DoubleArmV7



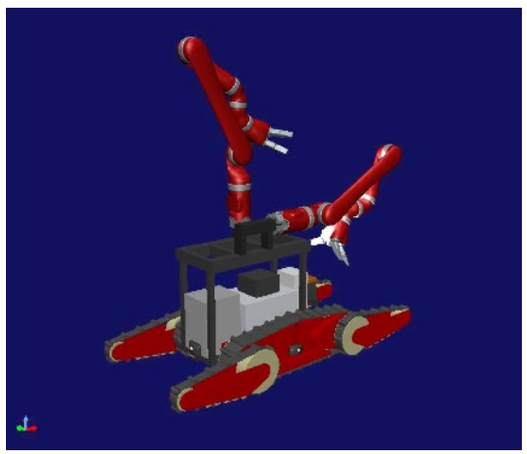
Quadcopter

サンプルモデルの操作方法

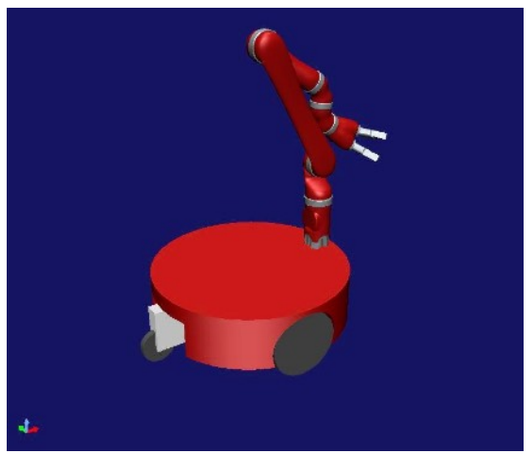
❖ サンプルロボット操作マニュアルを参照

ゲームパッド操作プロジェクト

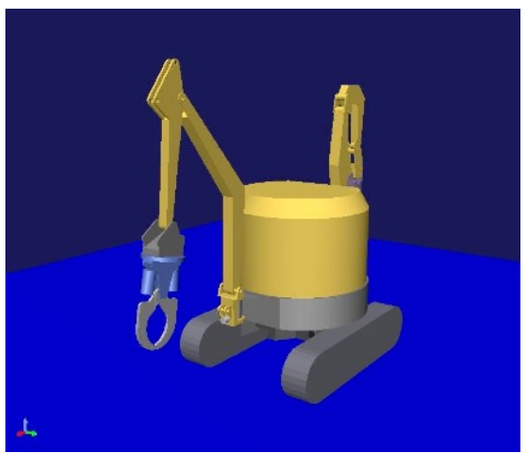
- sample/SimpleController/AizuSpiderDS.cnoid（Aizu Spider）
- sample/SimpleController/AizuWheel.cnoid（4輪モデル）
- sample/SimpleController/DoubleArmV7S.cnoid（双腕建機）
- sample/Multicopter/QuadcopterJoystick.cnoid（ドローン）



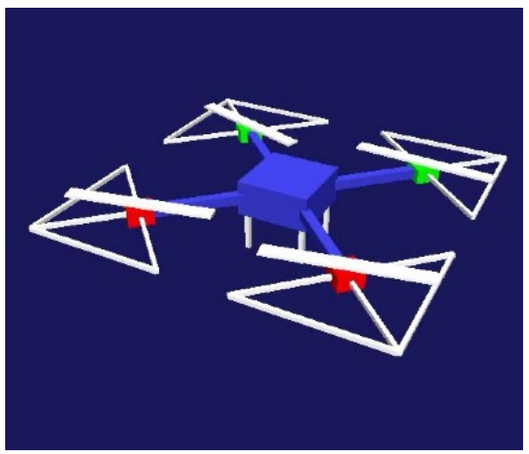
双腕付きAizuSpider



単腕付き4輪モデル



双腕建機モデル



ドローンモデル