７．ダイナミックウォーキングチャレンジ

このページはDynamic Walking Challenge : Go Distanceに対応しています。
Dynamic Walking Challengeファイルのダウンロード

概要

　このページでは動的歩行のシミュレーション実行とモデル拡張にオープンシムソフトウェアを用います。初めに4セグメントのダイナミックウォーカーモデルと歩行路を作ります。パラメータの調節や新しい要素の追加によってモデルを変えることで坂道歩行距離の延長を目指します。オープンシムGUIやMatlabスクリプトコマンドによりモデル要素の追加、プロパティーの調節、動的シミュレーションの視覚化、シミュレーション結果のプロットによってモデル作成およびパラメータの調整方法を学びます。

＊このチュートリアルでは途中からMatlabが必要になります。

動画 はHPを参照

I. はじめに

Scripting with Matlabページからスクリプトを使うための準備をしてください。32、64ビット共にサポートされていますが、64ビットのセットアップは別の操作が必要になります。

＊簡単にMatlab環境の設定方法を記載します。詳細はHPで確認してください。Automated Setupではうまくいかなかったので、manual Setupで実施することをお勧めします。

1. オープンシムとMatlabのビット数が同じバージョンをインストールしてください。（32ビットまたは64ビット）
2. 管理者として実行からMatlabを立ち上げてください。
3. Matlabのコマンドウィンドーでedit classpath.txtと入力し、エンターを押してください。
4. クラスパスの最後の行にOPENSIM_INSTALL_DIRECTORY(C:/OpenSim 3.3など)/opensim/modules/org-opensim-modeling.jarを追加してファイルを保存してください。
5. コマンドウィンドーでedit librarypath.txtと入力し、エンターを押してください。
6. ファイルの最後にOPENSIM_INSTALL_DIRECTORY(C:/OpenSim 3.3など)/bin/と入力し、ファイルを保存してください。
7. Matlabのパスにオープンシムライブラリーを追加します。コマンドウィンドーでpathtoolと入力してエンターを押してください。
8. フォルダー追加先から先ほど入力したbinフォルダーを探して、フォルダーの選択を押すとパスファイルが追加されます。Matlabからオープンシムの関数を使うには、一度Matlabを閉じてから再度立ち上げてください。
9. コマンドウィンドーでimport org.opensim.modeling.* と入力してエラーが表示されなければうまくセットアップができています。

II. オープンシムGUIとモデル

このセクションではオープンシムGUIを操作して、ウォーカーと歩行環境のプロパティーの作成・変更方法を学びます。まずGUIでモデルを構成するセグメント、関節、接触点を調べます。あらかじめ用意されているモデルはシンプルなモデルです。モデルは膝関節、2つの脚、球状の足部でできています。この後のセクションでセグメントや関節の追加、モデルプロパティーの調整方法を学んで、自身のウォーカーを作ります。ウォーカーのモデルと歩行環境は次の要素から構成されています。：

- 障害物が埋め込まれた傾斜角10度の長いプラットフォーム
- プラットフォーム傾斜角度に応じて動く（2つのスライドジョイント）骨盤
- ピンジョイントで骨盤と連結する2つの大腿セグメント
- ピンジョイントで大腿に連結する2つの下腿セグメント
- 反力が生じる球面の接触点
- 接触点で体に生じる反力
- 関節の動作範囲を規定する制限力

A. オープンシムの起動

ウィンドーズメニューからオープンシムを起動してください。Viewウィンドーに何も表示されていない状態から始めます。GUIを詳しく知りたい方はオープンシムのユーザーズガイドを見てください。

B. モデル要素の確認

1. オープンシムのGUIで**File>Open Model...**を選択し、**DynamicWalkerStarter/Model**ファイルからWalkerModelTerrain.osimを開いてください。あるいは、モデルファイルをviewウィンドーにドラッグして開くことも可能です。
2. **View**パネルはモデルの動きを視覚的に確認するために使います。マウスをドラッグして視点を回転、移動、ズームすることができます。
3. 左側の**Navigator**パネルで**+アイコン**を押すとモデル要素が確認できます。Navigatorパネルでモデルの要素を確認してみましょう。
4. 要素名をクリックするとNavigatorパネルの下にある**Property**エディターにプロパティーが表示されます。
5. **Coordinates**パネルはモデルの現在の関節角度や位置が表示されます。
    - Coordinatesの値はスライダーを用いるか、左横のテキストボックスに値を入力することで変更できます。値はメートルまたは角度で表示されています。
    - 速度（m/sまたはradian/s表記）は順動力学の開始状態で用います。この値は右のテキストボックに入力することで設定できます。
    - Coordinateの値は鍵アイコンでロックのオンオフができます。
    - モデルをデフォルト姿勢に戻すには**Poses>Default**を選択してください。Posesメニューを使ってデフォルト姿勢の変更や新たな姿勢を追加できます。

Questions

1. モデルの自由度を調べてください。
2. 大腿と下腿の長軸の長さを調べてください。（ヒント：下肢は関節で連結しています。大腿と骨盤、下腿と大腿のピンジョイントの位置は大腿を基準としています。）
3. モデルの大腿と下腿の質量比を調べてください。

III. ウォーカーのシミュレーション

このセクションではオープンシムGUIでモデルのシミュレーション、動作結果の再生、プロット、動画の保存を行います。

A. 順動力学ツールの実行

1. 順動力学ツールを用いる前に、左の**Coordinates**タブで任意のシミュレーション初期角度と速度を入力してください。
2. GUIの上のメニューから**Tools>Forward Dynamics**を選択してください。
3. **Time range to process**は0秒と２秒を入力してください。
4. OutputのDirectoryにはDynamicWalkingStarter\Results\FWDを選択してください。これ以外の項目は入力不要です。
5. 次に同様の順動力学ツールを用いるために、セッティングファイルを保存しておきます。**Save**からフォルダ（Resultsなど）を選択し、名前（Setup_Forward.xmlなど）を入力して保存してください。
6. **Run**ボタンを押してシミュレーションを開始してください。
7. **Close**を押してください。

B. 動作の再生・視覚化

順動力学ツールを使うと、control signalの時間経過とモデルの状態が出力ディレクトリに3つのstoファイルで保存されます。Coordinateの値や速度などモデル状態が**Navigator**パネルの**Motions**の下に表示され、動きを確認することができます。

1. 異なる開始状態で複数の動きを作成したい場合は、**Navigator** の**Results**を右クリックし、Resultsの名前を変更して下さい。

GUI画面の中央上にある青色の動画コントロールボタンで再生および結果の表示ができます。上下の矢印アイコンをクリックするかテキストボックスに値を入力することで動画速度を調整できます。中央のコントロールボタンで動画の再生や逆再生、フレーム送りができます。

1. 青の再生ボタンを押して順動力学シミュレーションの結果を確認してください。

C. GUI動画の作成

1. **Motions**の下の**Navigator**パネルで動作名をダブルクリックしてカレント動作にしてください。（**太字**で表示させる）
2. Viewウィンドー右側のビデオカメラアイコン（下から3番目）をクリックしてください。これでカメラアイコンがオレンジ色に変わり、保存するファイルの選択画面が表示されます。
3. 動画ファイル名を入力し、**Open**を押してください。＊チュートリアルでは任意としていますが、保存フォルダーはResults、保存ファイル名はWalkerMovieとして説明します。
4. Viewコントロールを使って時間や動作スピードの設定をしてください。＊0-2秒、動作速度0.5として説明します。
5. 動作を再生してください。（特定の部分を保存したい場合には、動作がすべて終了する前に停止ボタンをクリックしましょう）動作の途中で停止、視点の回転、再生をすることで異なる視点でモデル動作を保存できます。
6. 動作終了後にビデオカメラアイコンを再度押してください。これでキャプチャーが終了し、カメラアイコンが青に戻ります。

Note: 動画サイズはviewスクリーンの大きさによって変わります。再生速度によって決まる時間フレームで動きが保存されています。小さな容量に動画ファイルをしたい場合はウィンドーサイズと再生速度を調整してください。

＊OpenSim3.3の64bit（OpenSim3.2も同様？）は動画ファイルを保存・再生することができません。詳細は**Recording a Movie**を参照してください。EditでPreferences...を選択し、PreferencesのSave Movie FramesにOffと入力して動画キャプチャーをすると複数の静止画を保存できます。この静止画を別ソフトで組み合わせれば動画に変換することができます。また、オープンシムフォーラムでOpenSim3.3のMATLABコードも参考になります。

D. GUIのプロット

結果を表示するためGUIにはPlotterツールがあります。モデル右足のCoordinate速度をプロットします。

1. GUIのトップメニューから**Tools>Plot**を選択してください。
2. **Y-Quantity...**は順動力学ツールで作成した**Results(deg.)**を選択してください。
3. 右股関節の速度（RHip_rz_u）と右膝の速度（RKnee_rz_u）のチェックボックスをオンにしてOKを押してください。
4. **X-Quantity...**はtimeを選択してください。
5. **Add**をクリックしてデータのプロットを表示させます。

このplotツールに関して詳しく知りたい場合はHelpボタンを押してヘルプページを参照してください。

E. 歩行距離が最大となるモデルへの調節

調節できるパラメータの一部を下にリストアップしています。

１．質量、慣性モーメント

- セグメントが持つ質量と関節モーメントはNavigatorパネルで変更できます。＋アイコンをクリックしてBodyグループを開き、リストから特定のセグメントを選択してください。Propertiesウィンドーで質量、質量中心、慣性行列の6要素を選択できます。
- 質量中心はセグメント開始点からの場所をセグメント参照フレームで表しています。また慣性モーメントの値はセグメント参照フレームにおける質量中心周りの慣性モーメントを表します。

２．セグメントの長さ

- **Navigator**パネルで**Joints**セットプロパティを開きます。セグメントの長さは隣り合う関節の位置によって規定されます。（右大腿セグメントを変える場合は、RThighToPelvisとRShankToRThighのパラメータを変えます。オープンシムの関節に関してはオープンシムモデルページまたはオープンシムDoxygenを読んでください。）

３．**Coordinate**パネルから変更できる開始姿勢と開始スピード

Notes:

- **Forward tool**ではCoordinatesパネルで開始条件を決めます。
- 関節位置の変更によるセグメント長の変化はGUI画面では表示されない場合が多いです。**Body**セットから**Properties**パネルでセグメントを開いて確認してください。

Questions

１．モデルが多くのステップを行えない理由は何ですか？歩行を安定させるには何を変える必要がありましたか？

IV. Matlabスクリプトインタフェースによるモデル拡張

元のモデルでは立脚時の膝伸展保持が難しく倒れてしまいます。そこで、2つのモデルの拡張例を行い、Matlabスクリプトインターフェースの使い方を学びます。膝伸展位を保持するためには、膝関節にマグネットモデルを作って、膝伸展力を追加します。また、次にCAD.objファイルから足部モデルを追加します。

始める前に：

1. **Scripting with Matlab** を参照してオープンシムとMatlabのインターフェースを設定してください。
2. スクリプトはMatlabのワーキングディレクトリー（カレントディレクトリ）がDynamicWalkingConference/MatlabScriptであるとして書かれています。＊ダウンロードしたフォルダーは別の名前となっていました。このためDynamicWakingStarter\UserFunctionsに保存したとして説明します。別のフォルダーに保存する場合はフォルダーのプログラム内のフォルダー名やフォルダー位置を書き直してください。

A. Matlabを用いた膝関節マグネットフォースの追加

ダイナミックウォーキングモデルでは立脚期の膝が容易に曲がってしまいます。立脚期で膝伸展位を保持する方法として、伸展力を発生するマグネットフォースを膝関節に作成します。オープンシムでは筋モデルを作成できるよう、ライブラリーにスプリング要素を持つアクチュエータが用意されています。今回ExpressionBasedPointToPointForceを用いてマグネットフォースを作成します。ExpressionBasedPointToPointForceは2点間の距離（d）とその時間微分（ddot）が計算されるため、**d**と**ddot**の変数を用いた数式（発生する力）を設定することができます。力要素はストリングで書かれており、パーサは+, -, *, exp, pow, sqrt, sin, cos, tanなどC言語で書かれた演算式を読み込むことができます。

フォルダーにはプログラムの一部が書かれたスクリプトAddExpressionPointToPointForceMagnets.mが用意されています。（ダウンロードファイルには見つかりませんでした。このため、後に出てくるプログラムコードをコピーしてファイルを作成しました。保存するフォルダーはDynamicWakingStarter\UserFunctionsとしてください。）

スクリプトを完成させるには、膝マグネットフォースの起始停止のセグメントおよび位置を決める必要があります。マグネットフォースの数式を書きましょう。コードを入力する場所は**(***** WRITE YOUR CODE HERE: ... *****)**で示しています。

プログラム作成の参考として、リファレンスフォルダーに完成したファイル(AddExpressionPointToPointForceMagnets.m)があります。（ファイルは見つかりませんでした。AddKneeMagnets.mが該当するものと思われます。）

マグネットフォースは$f=C/d^2$で計算されます。；定数Cは$0.01(Nm)^2$です。dはマグネットの起始と停止の距離を表しています。

1. 膝のマグネットフォースを加えるスクリプトを完成させましょう。オープンシムモデルの**doxygen**（セグメントを作る方法など）とExpressionBasedPointToPointForceを参考にしてください。＊すでに完成しているプログラムAddKneeMagnets.mと作成中のAddExpressionPointToPointForceMagnets.m を見比べて、マグネット作成に必要な要素を確認しましょう。その内容はExpressionBasedPointToPointForceで調べることができます。
2. セクションVでオープンシムGUIやそのスクリプトのヘルプを使いながら視覚化とシミュレーションを行います。ウォーカーが進むステップ数は増えるでしょうか？

AddExpressionPointToPointForceMagnets.mのコード

```
% -------------------------------------------------------------------------- %
% The OpenSim API is a toolkit for musculoskeletal modeling and           %
% simulation. See http://opensim.stanford.edu and the NOTICE file         %
% for more information. OpenSim is developed at Stanford University       %
% and supported by the US National Institutes of Health (U54 GM072970,    %
% R24 HD065690) and by DARPA through the Warrior Web program.             %
%                                                                         %
% Copyright (c) 2005-2013 Stanford University and the Authors             %
% Author(s): Daniel A. Jacobs                                             %
%                                                                         %
% Licensed under the Apache License, Version 2.0 (the "License");         %
% you may not use this file except in compliance with the License.        %
% You may obtain a copy of the License at                                 %
% http://www.apache.org/licenses/LICENSE-2.0.                             %
%                                                                         %
% Unless required by applicable law or agreed to in writing, software     %
% distributed under the License is distributed on an "AS IS" BASIS,       %
% WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or         %
% implied. See the License for the specific language governing            %
% permissions and limitations under the License.                          %
% -------------------------------------------------------------------------- %
% This script uses a ExpressionBasedPointToPointForce to realize a magnet
% force model.  The ExpressionPointToPointForce calculates the relative
% translation and relative velocity between two points in the global frame
% and allows the user to specify a force based on a string of the symbols
% d and ddot.

% Import Java Library
import org.opensim.modeling.*

% Open the model
walkerModel = Model('../Model/WalkerModelTerrain.osim');

% Change the name
walkerModel.setName('DW2013_WalkerModelTerrainAddMagnet');
```

```
% Define the body and points location in each body's coordinate frame
locBody1 = Vec3(0.05,-0.2, 0);
locBody2 = Vec3(0.05, 0.2, 0);

% Create an ExpressionBasedBushingForce for the magnets
leftKneeMagnet = ExpressionBasedPointToPointForce();
leftKneeMagnet.setName('LeftKneeMagnet');
rightKneeMagnet = ExpressionBasedPointToPointForce();
rightKneeMagnet.setName('RightKneeMagnet')

% Set the first body and the reference frame properties
% ***** WRITE YOUR CODE HERE:  *****
% Task: Navigate to your local doxygen and find the methods to set the name
% of Body1 and set the location of Point1.

% Set the second body and the location of the reference point
% ***** WRITE YOUR CODE HERE:  *****
% Task: Set the name of Body2 and the location of Point2.

% Set the expression to represent a magnet (1/d^2) force
% ***** WRITE YOUR CODE HERE:  *****
% Task: Use the setExpression method for each magnet and pass an char type
% argument of the desired function (0.01/d^2).

% Add the force to the model
walkerModel.addForce(leftKneeMagnet);
walkerModel.addForce(rightKneeMagnet);

% Print a new model file
walkerModel.print('../Model/DW2013_WalkerModelTerrainAddMagnet.osim');
```

B. 足部の追加

立脚期に膝伸展モーメントを加える別の方法として、足部を大きな半径の半球状にする方法があります。立脚期中の床からの反力は膝伸展など下肢のモーメントを生み出します。接触点を前足部に移動させると立脚期の膝屈曲を抑制することになります。

オープンシムはヒトの骨格要素を視覚化するために球、円柱、四角柱などモデルジオメトリーファイルがあります。ライブラリーにモデルがなければ、ジオメトリーファイルに任意の形状を追加することができます。このセクションではMatlabスクリプトインターフェースを使って足部セグメントを追加する方法を説明します。オープンシムは.vtp, .stl .obj.拡張子のモデルファイルをインポートできます。 * Modelsフォルダーに ThinHalfCylinder100mmby50mmファイルが保存されています。

始めはシリンダー状の足部モデルデザイン.objファイルでできています。オブジェクトファイルは3dCADプログラムBlenderのオープンソースで作られています。AddCustomFeet.mスクリプトはobjファイルからContactMeshを追加し、足部と床面の接触点ElasticFoundationForceを作ります。

AddCustomFeet.mスクリプトファイルはプログラムの一部が書かれています。（後に示すプログラムコードです。なお、UserFunctionsフォルダーにあるAddCustomFeet.mは完成したプログラムです。）足部をモデルに追加するにはペアレントセグメントである下腿とチャイルドセグメントである足部の間のWeldジョイントの場所と軸角度を表す4つのVec3オブジェクトを定義する必要があります。コードを入力する場所は**(***** WRITE YOUR CODE HERE: ... *****)**で示しています。プログラム作成の参考に、完成したファイル(AddCustomFootComplete.m)がリファレンスフォルダーにあります。（＊ありませんでした。）

Weldジョイントは下記パラメータを使います。

- ペアレントフレーム位置（下腿フレーム原点からの位置）: 0.05, -0.2, 0 m
- ペアレントフレーム角度（下腿フレーム角度に対するWeldジョイントフレーム角度）: 0, 0, 0 rad
- チャイルドフレーム位置（足部フレーム原点からの位置）: 0, 0, 0 m
- チャイルドフレーム角度（足部フレーム角度に対するWeldジョイントフレーム角度）: 0, 0, 0 rad

Weldジョイントコンストラクターファンクションの関数名(locationInParent, orientationInParent, locationInChild, and orientationInChild)を使用してください。

1. Tneウォーカーモデルに足部を追加するスクリプトを作成します。Doxygenの**Body**, **WeldJoint**, **ContactMesh**, **ElasticFoundationForce**セクションを参考にしましょう。
2. オープンシムのGUIやセクションVでスクリプトのヘルプを使ってモデルを確認し、シミュレーションを行いましょう。ウォーカーのステップ数は増えるでしょうか？

AddCustomFoot.mコード

```
% --------------------------------------------------------------------------
% The OpenSim API is a toolkit for musculoskeletal modeling and
% simulation. See http://opensim.stanford.edu and the NOTICE file
% for more information. OpenSim is developed at Stanford University
% and supported by the US National Institutes of Health (U54 GM072970,
% R24 HD065690) and by DARPA through the Warrior Web program.
%
% Copyright (c) 2005-2013 Stanford University and the Authors
% Author(s): Daniel A. Jacobs
%
% Licensed under the Apache License, Version 2.0 (the "License");
% you may not use this file except in compliance with the License.
% You may obtain a copy of the License at
% http://www.apache.org/licenses/LICENSE-2.0.
%
% Unless required by applicable law or agreed to in writing, software
% distributed under the License is distributed on an "AS IS" BASIS,
% WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
% implied. See the License for the specific language governing
```

```
% permissions and limitations under the License.
% ----------------------------------------------------------------------
% This file demonstrates how to add a foot to the model using a custom
% object.
%
% The model will look for .object files in two locations
% 1) The model's local directory
% 2) The Models directory in the OpenSim Installation Directory
% e.g. (<OpenSim_Home>\Models)
%
% The allowed object extensions are .vtp, .stl, .obj
% ----------------------------------------------------------------------
% Import Java Library
import org.opensim.modeling.*

% Open the model
walkerModel = Model('../Model/WalkerModelTerrain.osim');

% Change the name
walkerModel.setName('DW2013_WalkerModelTerrainAddFoot');

% Disable the current foot forces
walkerModel.updForceSet().get('LFootForce').set_isDisabled(true);
walkerModel.updForceSet().get('RFootForce').set_isDisabled(true);
walkerModel.updContactGeometrySet().get('LFootContact').setDisplayPreference(0);
walkerModel.updContactGeometrySet().get('RFootContact').setDisplayPreference(0);
% ----------------------------------------------------------------------
% Setup the new foot paramters
footMass    = 0.0001;
footInertia = Inertia(1,1,.0001,0,0,0);

% Create the leftFoot and rightFoot bodies
% ***** WRITE YOUR CODE HERE:  *****
% Task: Declare a leftFoot and a rightFoot variable and assign them a
% object with the constructor Body().  Use the "set" functions to set the
% name, the mass and the inertia.


% Set up the parameters for the Weld Joint connecting the feet to the
% shanks
% ***** WRITE YOUR CODE HERE:  *****
```

```
% Task: Create the Vec3s for the location and orientation of the parent and
% child using the constructor Vec3(x,y,z)

% Get a reference to each shank
leftShankBody = walkerModel.updBodySet().get('LeftShank');
rightShankBody = walkerModel.updBodySet().get('RightShank');

% Create Weld Joint
lFootToLShank    = WeldJoint('LFootToLShank', leftShankBody, locationInParent, ...
   orientationInParent, leftFoot, locationInChild, orientationInChild, false);
rFootToRShank    = WeldJoint('RFootToRShank', rightShankBody, locationInParent, ...
   orientationInParent, rightFoot, locationInChild, orientationInChild, false);

% Add the Visual Object
leftFoot.addDisplayGeometry('ThinHalfCylinder100mmby50mm.obj');
rightFoot.addDisplayGeometry('ThinHalfCylinder100mmby50mm.obj');

% Add the body to the Model
walkerModel.addBody(leftFoot);
walkerModel.addBody(rightFoot);
% ----------------------------------------------------------------------
% Create a ContactMesh for each foot
footMeshLocation    = Vec3(0,0,0);
footMeshOrientation = Vec3(0,0,0);
leftFootContact = ContactMesh('ThinHalfCylinder100mmby50mm.obj', ...
   footMeshLocation, footMeshOrientation, leftFoot, 'LFootElasticContact');
rightFootContact = ContactMesh('ThinHalfCylinder100mmby50mm.obj', ...
   footMeshLocation, footMeshOrientation, rightFoot, 'RFootElasticContact');

% Add ContactGeometry
walkerModel.addContactGeometry(leftFootContact);
walkerModel.addContactGeometry(rightFootContact);
% ----------------------------------------------------------------------
% Create an elastic foundation force for both feet
leftElasticFootForce = ElasticFoundationForce();
rightElasticFootForce = ElasticFoundationForce();

% Set Names
leftElasticFootForce.setName('LeftElasticFootForce');
rightElasticFootForce.setName('RightElasticFootForce');
```

```
% Set transition velocity
leftElasticFootForce.setTransitionVelocity(0.1);
rightElasticFootForce.setTransitionVelocity(0.1);

% Define Contact Parameters
stiffness          = 1.0E6;
dissipation        = 2.0;
staticFriction     = 0.8;
dynamicFriction    = 0.4;
viscousFriction    = 0.4;

% Set the Contact Parameters for the forces
leftElasticFootForce.addGeometry('LFootElasticContact');
leftElasticFootForce.addGeometry('PlatformContact');
leftElasticFootForce.setStiffness(stiffness);
leftElasticFootForce.setDissipation(dissipation);
leftElasticFootForce.setStaticFriction(staticFriction);
leftElasticFootForce.setDynamicFriction(dynamicFriction);
leftElasticFootForce.setViscousFriction(viscousFriction);

rightElasticFootForce.addGeometry('RFootElasticContact');
rightElasticFootForce.addGeometry('PlatformContact');
rightElasticFootForce.setStiffness(stiffness);
rightElasticFootForce.setDissipation(dissipation);
rightElasticFootForce.setStaticFriction(staticFriction);
rightElasticFootForce.setDynamicFriction(dynamicFriction);
rightElasticFootForce.setViscousFriction(viscousFriction);
% ----------------------------------------------------------------------
% Add Forces
walkerModel.addForce(leftElasticFootForce);
walkerModel.addForce(rightElasticFootForce);
% ----------------------------------------------------------------------
% Save the new model
walkerModel.print('../Model/DW2013_WalkerModelTerrainAddFoot.osim');
```

V. カスタムウォーカーモデルの作成

UserFunctionsフォルダーの中にはスクリプト例が多くあり、ウォーカーモデル作成の参考になるでしょう。これらのスクリプトは簡単にモデルが作成できるようにMatlabスクリプトインターフェース機能の重要な要素を選んでいます。Matlabのヘルプインターフェースを使って各機能の説明にアクセスしてください（例：Matlabのコマンドウィンドウでhelp RunForwardToolと入力すると説明を表示できます）。 スクリプトで使われているオープンシムの重要なクラスの詳細を示すdoxygenページにリンクします。（オープンシムはC++を使ってプログラムが書かれていますクラスがわからない場合はC++言語を学んでください。）

＊スクリプト説明の一部を記載しています。すべて見る場合はHPを参考にしてください。

スクリプト名	説明	関連するFunction/Class
CompareTwoModels.m	二つのモデルを同じ初期状態にセット、プロットで結果の比較	IntegrateOpenSimPlant, PlotOpenSimData
DesignMainStarter.m	モデル読み込み、初期状態のセット、順動力学シミュレーションの実行、結果のプロット	IntegrateOpenSimPlant, PlotOpenSimData
DesignMainStarterWithControls.m	アクチュエータの活動量をDesignMainStarterに追加	OpenSimPlantControlsFunction

スクリプト名	説明	関連するFunction/Class
IntergateOpenSimPlant.m	Matlabのインテグレータを使って実行、モデルや状態パラメータを計算して返す機能の作成	
OpenSimPlantControlsFunction.m	筋やアクチュエータに与えるcontrol値を計算	DesignMainStarterWithControls
OpenSimPlantFunction.m	オープンシムモデルオブジェクトなどの状態を計算するインターフェースの作成	
PlotOpenSimData.m	Matlabからプロットを作成	
ReadOpenSimData.m	オープンシムのストレージファイル（.sto）やモーションファイル（.mot）からMatlabフォーマット文を作成	
RunForwardTool.m	オープンシムの順動力学シミュレーションを実行	ForceReporter, ForwardTool

スクリプト名	説明	関連するFunction/Class

AddClutchedPathSpring.m	クラッチスプリングの作成	ClutchedPathSpring
AddCoordinateActuator.m	Coordinateアクチュエータの作成	CoordinateActuator
AddCustomFoot.m	メッシュオブジェクトの作成	WeldJoint, ContactMesh, ElasticFoundationForce
AddExpressionPointToPointForceMagnets.m	マグネットフォースの作成	ExpressionBasedPointToPointForce
AddMillardMuscle.m	Hillタイプモデル筋の作成(Millard 2013)	Millard2012EquilibriumMuscle
AddPathActuator.m	パスアクチュエータの作成	PathActuator
AddPathSpring.m	パススプリングの作成	PathSpring
AddSpringGeneralizedForce.m	スプリングジェネラライズフォースの作成	SpringGeneralizedForce
CreateWalkingModelAndEnvironment.m	歩行環境の作成	Body, WeldJoint, SliderJoint, PinJoint, ContactSphere, ContactHalfSpace, HuntCrossleyForce, CoordinateLimitForce

ヒント:

- 自由にモデルを作成するため、多くのモデル要素が用意されています。モデル要素とその情報は3.1 Beta Doxygenのオンラインドキュメンテーションを参照してください。
- Matlabスクリプトとのインターフェースに関してはこちらのページを参照してください。
- Matlabではmethodsview('classname')やmethodsview('OpenSimObject')を使って使用可能な機能のリストを表示できます。
- オープンシムGUIのviewウィンドーでドラッグとドロップで出力ファイルからモデルと動きを読み込むことができます。
- 床面の障害物で生じる力を取り除きたい場合には、NavigatorパネルのForceセットでContact forceのObstacleForcesを右クリックしてDisableを選択してください。
- このページで示すMatlabスクリプトは、MatlabGUIでhelp<function-name>をタイプすることができます。

参考文献:

- Millard, M., Uchida, T., Seth, A., Delp, S.L. (2013) Flexing computational muscle: modeling and simulation of musculotendon dynamics. ASME Journal of Biomechanical Engineering, 135(2):021005.