BodyMech 3.06.01

BodyMech Guide

Synopsis:
BodyMech is a software tool, which runs in MATLAB and offers functions for 3D human movement analysis based on cluster marker registrations.

The **BodyMechGuide** includes installation instruction, user manual, theoretical background of 3D kinematic analysis; function library and description of (LAB) initial conditions to apply BodyMech (3.06.01; $Version2006) .

Editing BodyMechGuide
Caroline Doorenbosch
Jaap Harlaar
Marre Kaandorp

Dept. of Rehabilitation Medicine VUmc
Amsterdam,
The Netherlands

12-Jul-06

More information**:** **www.BodyMech.nl**
Feedback and suggestions: BodyMech@vumc.nl

Human Movement Laboratory

VU medisch centrum

Contents

Preface

In 2001, BodyMech was created by Jaap Harlaar, in order to facilitate the analysis of 3D kinematics to relevant joint angles for (PhD-) students.

BodyMech is a software tool, which runs in MATLAB and offers functions that are needed for 3D human movement analysis based on marker registrations.

BodyMech is based on a graphical user interface (GUI) that can be used to interactively analyze and visualize experiments of human movements, recorded with a 3D-kinematics system (Optotrak, NDI). Data are stored in a structured file.

Users of BodyMech should at least have a basic knowledge of MATALB and movement analysis techniques. A theoretical introduction of these techniques can be found in Chapter IV.

The present version (3.06.01) of BodyMech is limited to the kinematics of human movement with addition of EMG recordings and ground reaction forces.

The editors gratefully acknowledge Nienke Wolterbeek for all her assistance to realize this version of BodyMech.

Jaap, Harlaar
Caroline Doorenbosch
Marre Kaandorp

June 2006, VUmc Amsterdam

More information: **www.BodyMech.nl**

Suggestions and feedback: BodyMech@vumc.nl

BodyMech
version 3.06.01

chapter I. General Introduction

Introduction

Intended audience

This guide can be used by everybody who is interested in motion analysis applications.
For students and researchers who are new in this field, it is strongly recommended to read the theoretical chapter (Chapter IV) before start using BodyMech.
More experienced persons on motion analysis can use this manual to introduce themselves to BodyMech.
For more details about the underlying codes and functions of BodyMech, the Matlab Directory Report functions (Chapter III) can give a good overview of the BodyMech functions. Also some example files are supplied in the subfolder DemoProject of BMProjects.

Prerequisites and preparations for proper use

In order to use BodyMech for human movement analysis, the following steps must be made:

1. Install the BodyMech files on your computer (p. 5), from this point on you can play around with BodyMech using the demo files as an example.
*As a default, specific **VUMC_LAB** functions are included in the initial installation (subfolder VUMC_LAB in folder LabFunctions). They are specifically written for applications in the Human Movement Lab at the VUmc and apply for calibration, data-file format of acquisition (import function) and used stylus. Depending on the lab, similar functions have to be adapted for its specific setup.*

2. Create a ProjectModel: each research project starts with the creation of a **Project model** (see Chapter II). This is a BodyModelScript which includes information related to the intended experiments (e.g. which segments, marker identification of each segment; muscles are measured, how many force-plates are used etc.). For proper use of BodyMech for kinematic analysis, clusterframes with at least 3 markers per segment are required.
Several (editable) template ProjectModel scripts are included in the subfolder BMtemplates.

*Note: Later in the analysis, for each **Session** or Subject, marker setup and calibration for a specific subject (e.g. static calibration measurement of a reference position, anatomical calibration files) are imported and stored for subsequent analysis of all trials of that subject . Finally, segment and joint kinematics can be calculated from raw 3D data from the **Trials** within each session. For each **Trial** (i.e. each performed movement) the appropriate files from the experiments (3D data and EMG/force if available) must be imported to analyse the specific movement. (see Chapter II)*
Note: it is always the responsibility of the user to make sure that analog and marker files are loaded are synchronized and that they apply to the same movement trial.

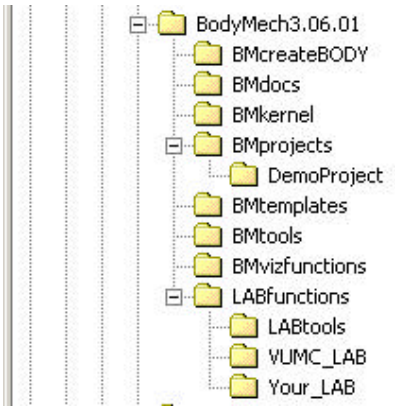
3. Design an anatomical calculation function; depending on the **Project**, an anatomical calculation function is required or has to be created beforehand (see for details Chapter II). A decision has to be made regarding the joint angle conventions, that will be used to calculate the joint kinematics. *Some templates of Anatomical Calculation functions for the lower extremity are included in the subfolder BMtemplates If the experimental protocols for these issues remain identical for each experiment the Project file has to be created only once.*

4. Calibrate your LAB set-up and requirements for data format import (Chapter II)

Initial conditions and actions for a proper installation

- **open MATLAB (version 5.3 or higher);**

- **Unzip the file BodyMech2006.zip** in a folder named BodyMech3.06.01. BodyMech software consists of a number of MATLAB m-files, stored in subfolders (see below) :



The reason for using different folders, is that each set of m-files has its own scope:

. **BMcreateBODY**: files that are required to Create a Projectmodel
. **BMdocs**: all BodyMech documents , e.g this guide
. **BMkernel** : files that are general to the approach of 3D human movement analysis.
. **BMprojects**: all files used for specific Projects. Default a subfolder DEMOproject is added. Users can add their own Project folder, which will contain any user written m-files and data that are bounded to a specific project. (see Chapter III: BodyMechFunctionLibrary for more help info for each function)
. **BM templates**: template script files of Projectmodels and anatomical calculation functions
. **BMtools:** basic helpful functions
. **BMvizfunctions**: files that are related to all Graphics User Interface items
. **LabFunctions**: all files that are specific or helpful to the laboratory environment; divided in general LABtools and specific folders (VUMC_LAB; Your_LAB)

- **Include the main and subfolders of BodyMech3.06.01 the path of MATLAB**. To extend the path setting, startup MATLAB and invoke the MATLAB Path Browser, by selecting ‘*Set Path’* from the File menu. In the MATLAB Set Path Window, click: ‘*Add with Subfolders’*. Select the BodyMech3.06.01 folder to add it to the Matlab Path (in Matlab5.3 all BodyMech folders have to be added separately). Then ‘*OK’*. Select ‘*Save Path’* from the file menu and close the Matlab Path Browser.

- **Type BodyMech on the Matlab command line**. This should startup the BodyMech graphical user interface start window (Figure 1).

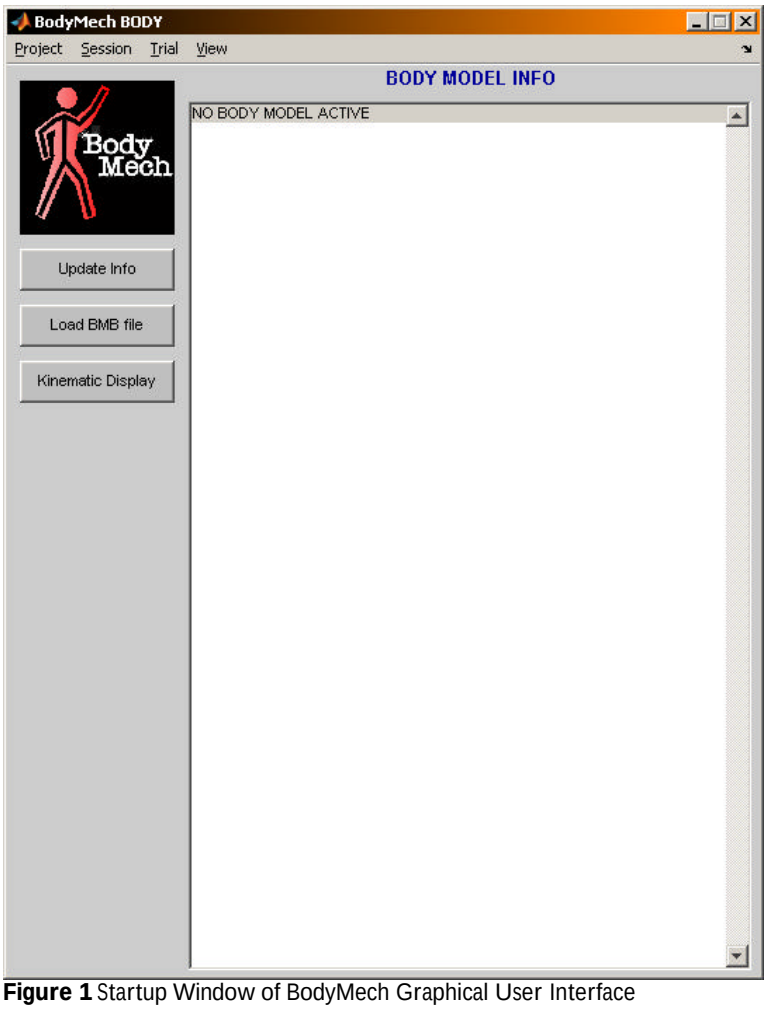


Figure 1 Startup Window of BodyMech Graphical User Interface

❑ **Go for a Quickstart**: Use the BodyMech GUI (Graphical User Interface) to visualize a DEMOtrial of human movement:

In the Demoproject folder, two complete trials are copied, which can be loaded as an example of one fully analysed movement:

→ Select: **`Load Trial file`** from the Trial menu (or Click PushButton Load BMBfile)

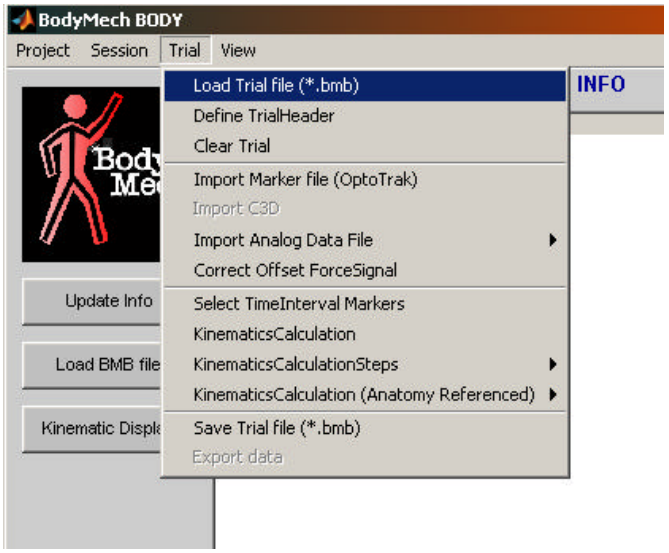


Figure 2. Load Trial

→ A window opens; browse to the folder \Bmprojects\DemoProject and select **`DEMOtrial6Gait.bmb`**

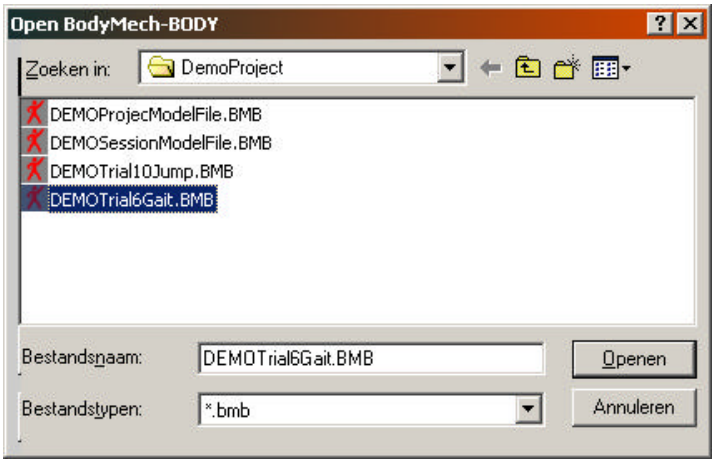


Figure 3. Select DEMO GaitTrial

The BODY MODEL INFO screen is filled with the content of the BODYfile (*note that 4 segments, 3 joints and 6 muscles are defined for this DEMO)*

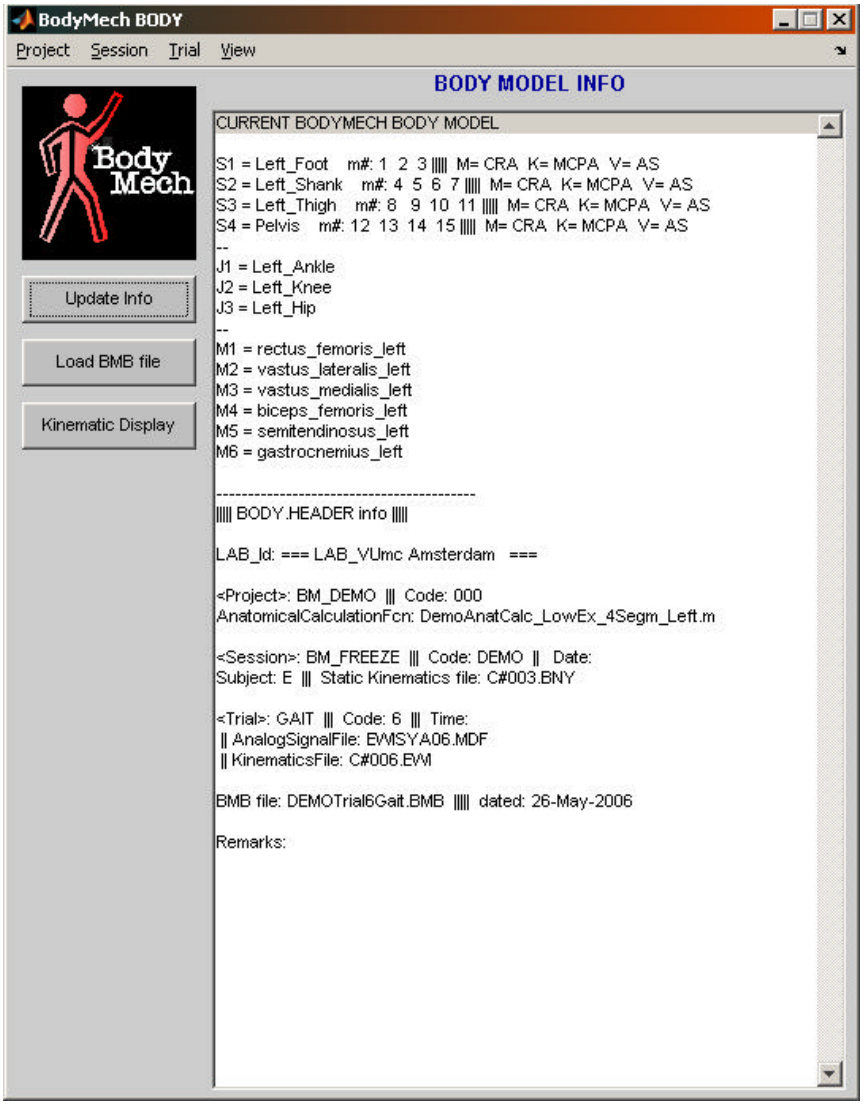


Figure 4. BODY MODEL INFO screen with ProjectModel HEADER content

To get a quick impression of the results: open de View menu and select e.g Show Orthogonal Views (or Click PushButton Kinematic Display)

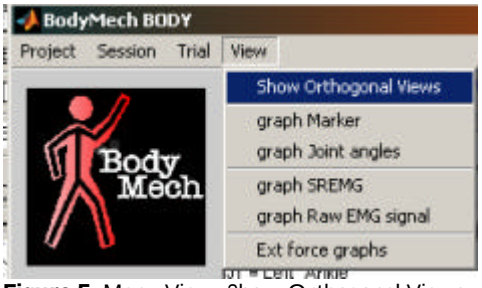


Figure 5. Menu View: Show Orthogonal Views

and the following window opens. Click on e.g. StickFigures and External force; scroll with the slider through the time frames to get a similar display as in Figure 6 below:

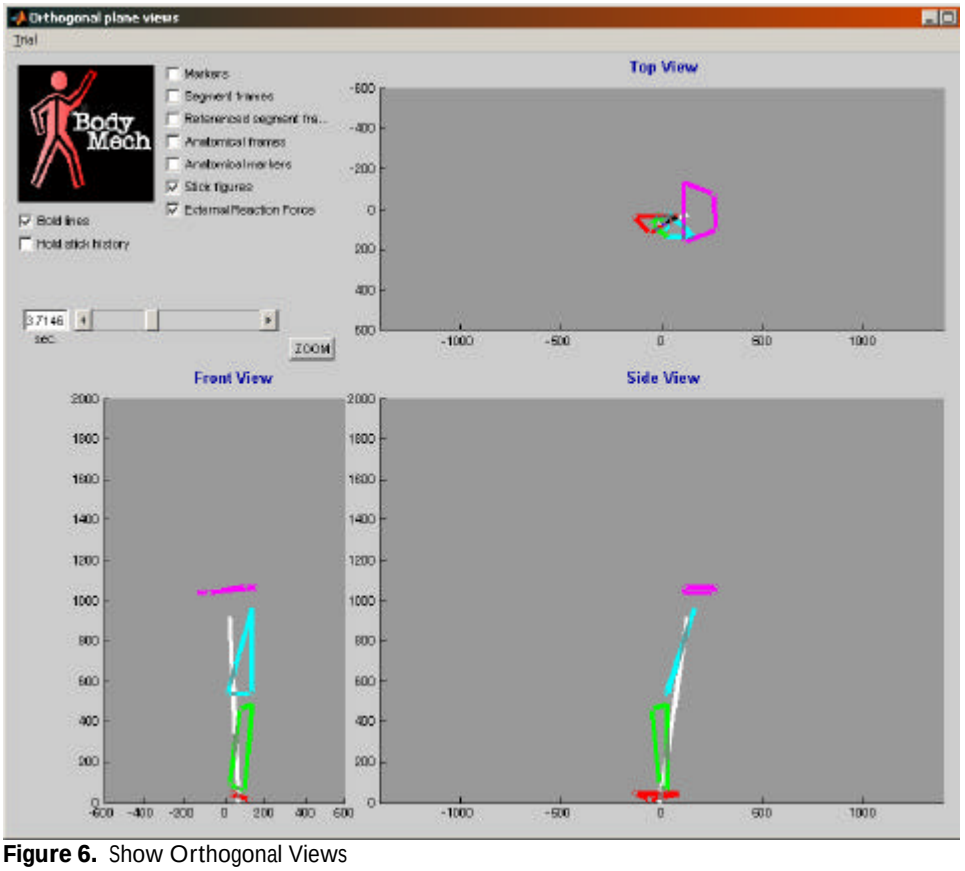


Figure 6. Show Orthogonal Views

Try all other options in this window and in the View menu of BodyMech .

In Chapter II, all steps from creating a Project and Session model and completing a Trial model will be described.

Starting from a cleared screen, these steps will finally result in the presented demo Trial file **`DEMOtrial6Gait.bmb`**

Following the described actions in Bodymech will give more insight in the possibilities of BodyMech and will facilitate its application for your own use, project, experiments and analysis.

BodyMech data structure

The basis of BodyMech data file is a structured array (Matlab variable) called BODY. This structure contains fields to store all measured and calculated data as well as additional information.

Appendix A show an overview of all possible BODY-fields and subfields.

If the structure file named BODY is made in BodyMech (i.e. a Bodymech model with corresponding data), the contents of BODY and its fields can be viewed in the Matlab command window, by typing for example:
`BODY.HEADER` *(see Figure 7). (NB: if an error message appears of the type of 'unknown variable or function; first type* `global BODY` *and then again BODY.HEADER (or some other field).*

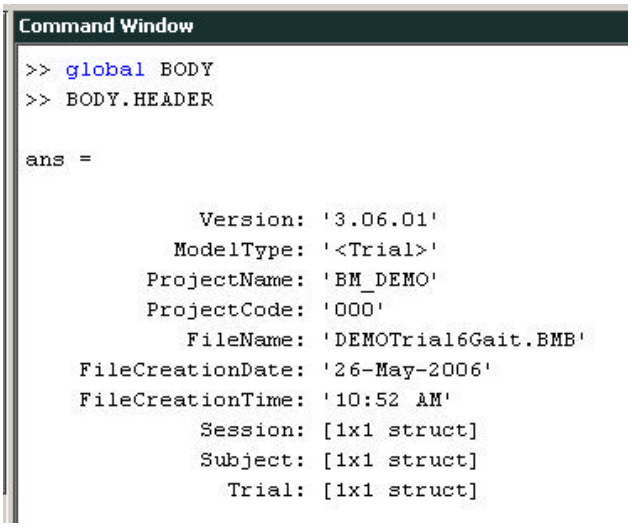


Figure 7 Content of BODY.HEADER in the Matlab Command Window.

The BODY structure will be saved as a '<Project>', '<Session>' or '<Trial>' file (all *.bmb files), depending on the contents of the structure (e.g. what is already calibrated or calculated).

Type of model is indicated in the field BODY.HEADER.ModelType ('<Project>', '<Session>' or '<Trial>', see also Appendix A). Content and use of each ModelType will be elucidated in the next Chapter.

Appendix B shows the Actions of BodyMech (i.e. a menu item in the Graphical User Interface) and the called BodyMech Functions (2nd column) and the derived Output (i.e. filled fields of the BODY structure).

BodyMech
version 3.06.01

chapter II. User Manual

In general, following all BodyMech calculation steps (structured in the BodyMech GUI), joint kinematics are derived from raw 3D marker coordinates recorded with a 3D analysis system. This version of BodyMech is based on the use of Optotrak for 3D marker recording in a set-up of clusters of at least 3 markers per segment.

In Figure 8 the steps of the different BodyMech functions to process a recorded movement data are shown. The round boxes show the input files from measurements that are required for the BodyMech processes in the grey boxes. More specific relation between BodyMech menu items and calculation steps can be found in Appendix B.

This Chapter will lead you through these steps.

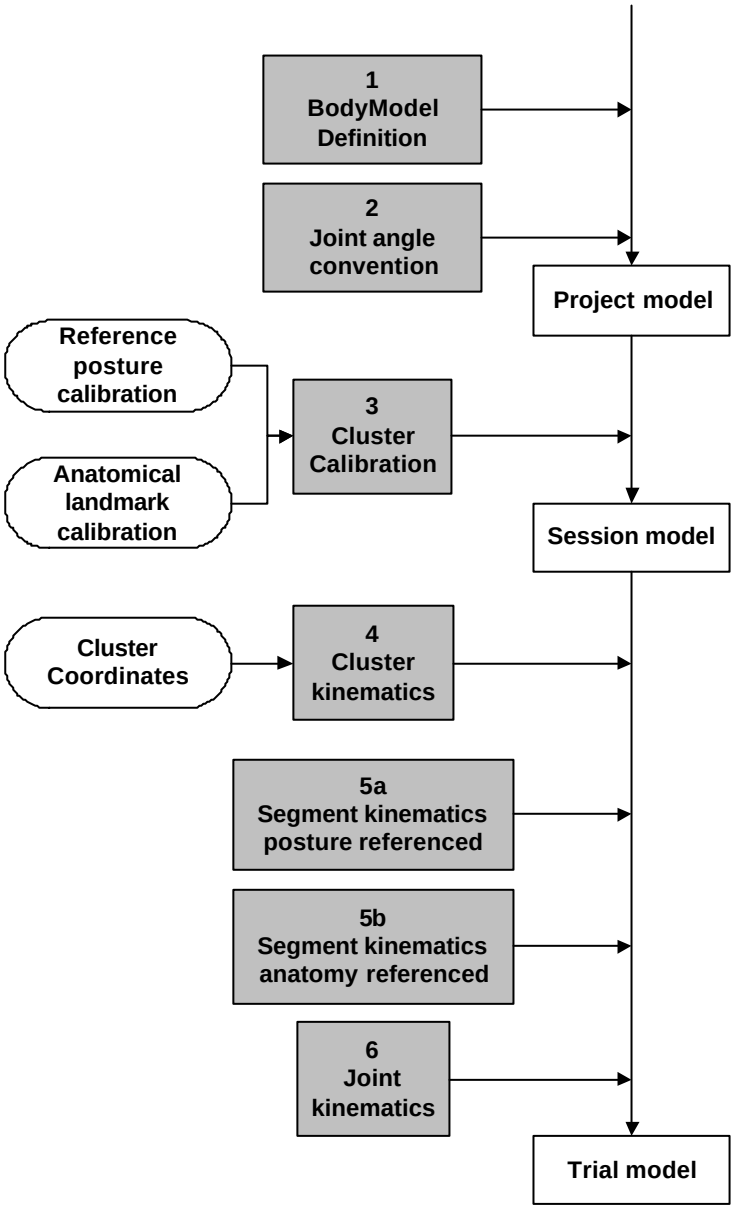


Figure 8 Human movement analysis with BodyMech

Body Model Definition

In BodyMech, the following terms Project, Session and Trial are used for different models, which reflect the different levels of analysis and include specific BodyMech information. Figure 9 shows this structure and the relation of the different modeltypes. In summary:

Project models contains all parameters of a whole research project (e.g. Project Name, Used segments; Used joints etc...).

Subsequently, a **Session model**, parameters are added to the basic Project Model, that corresponds to a specific experimental session or subject, who participates in the experiments of that project (e.g. subject names; marker setup; calibrations, etc...).

Finally, a **Trial Model** is build with a Session Model, in which all parameters are appended that fits with each measurement / movement of that specific subject; including EMG and force recordings if applicable.

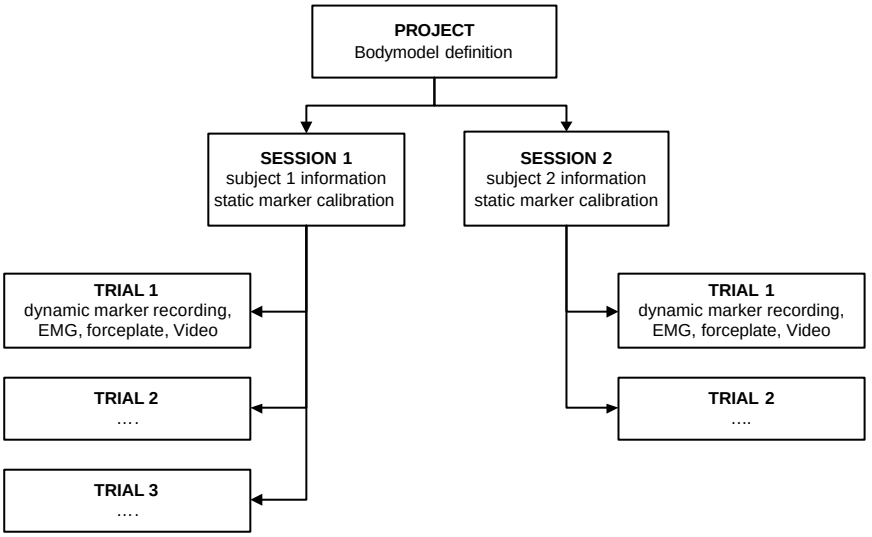


Figure 9 Project, Sessions and Trials

*Note A Project, Session or Trial file is always a *.bmb file and has the same file structure. The only difference is the contents of the field: BODY.HEADER.ModelType and the number of filled fields. The reason you can save the BODY structure as different files is that it will facilitate the orderly use of BodyMech and immediately shows the state of the file.*
*For a new research project you create a **Project** bmb file (model). For each new subject you start with the **Project**modelfile, calibrate it for the specific subject and save this as a **Session** bmb file (model). Then for each measurement you made with this subject, you open or start with the **Session** model and appends / import the measurements and save it as a **Trial** bmb file (model). This will be elucidated in more detail in the next section*

Before you begin

Before you can start with BodyMech calculations, some prerequisite project and LAB actions must be prepared:

1. Create a ProjectModel
2. Define Anatomical Calculation Function
3. Adapt functions according to your LAB specifications

ad 1. How to Create a ProjectModel

For each research project, decisions have to be made about the required information from the measurements and what body model matches to obtain this information (described in the project protocol). This chapter explains how to create a Project model.

Edit a new ProjectModel by yourself (skip the following section) or use one of the templates as follows:

→ select '**`Start Project'`** from the *Project menu*. *If another model was opened, all info will be cleared, resulting in a blank* **Body Model Info screen** *(as in Figure 1)*.

→ and select **`Edit ProjectModel (from template)`**

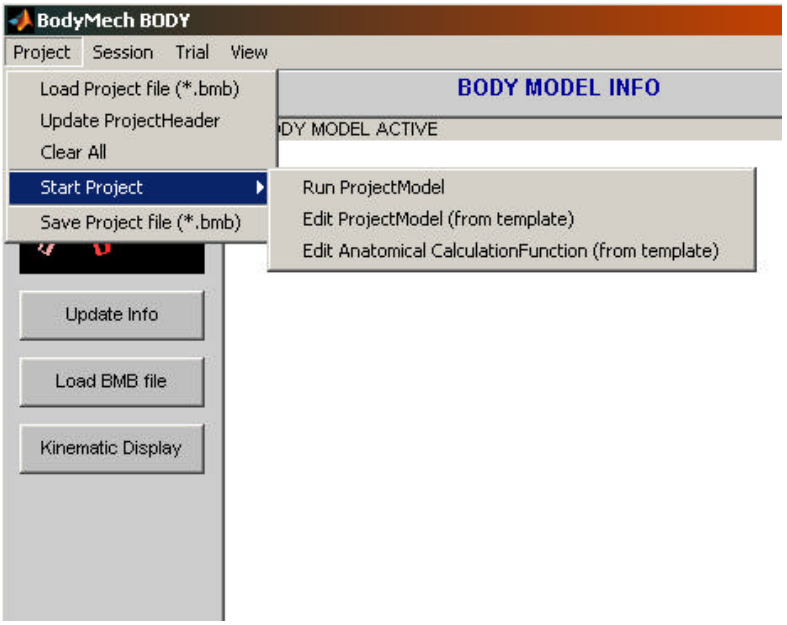


Figure 9 Start Project

→ a window will appear in which you can select one of the templates.

For the DEMO project: **`ProjectModel_LowEx_4Segm_Left.m`** is selected:

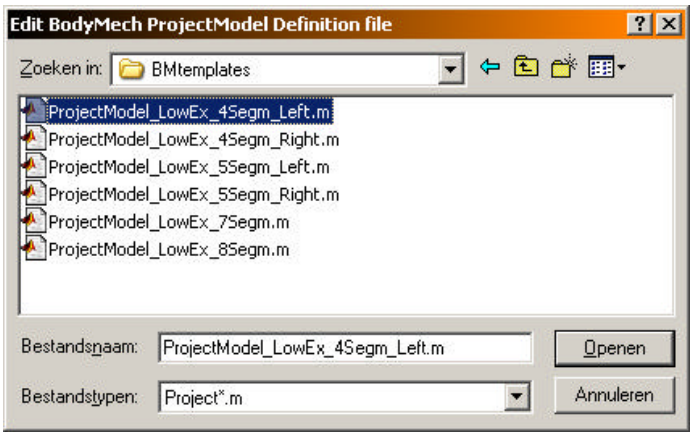


Figure 10 Select ProjectModel_LowEx_4Segm_Left.m

This will evoke the Matlab Editor and shows a scriptfile of a template of ProjectModel.

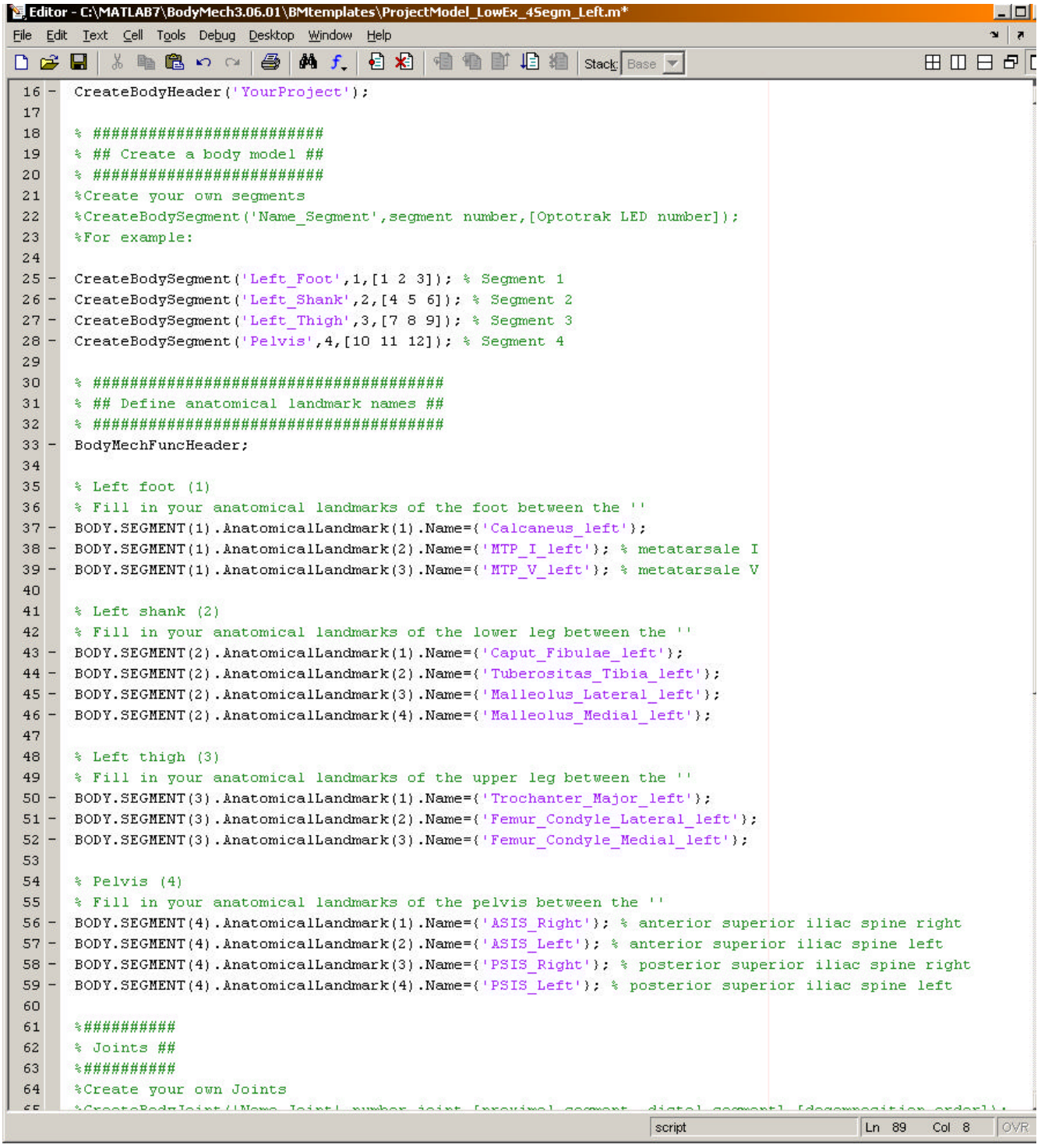


Figure 11. Upper part of script template of ProjectModel

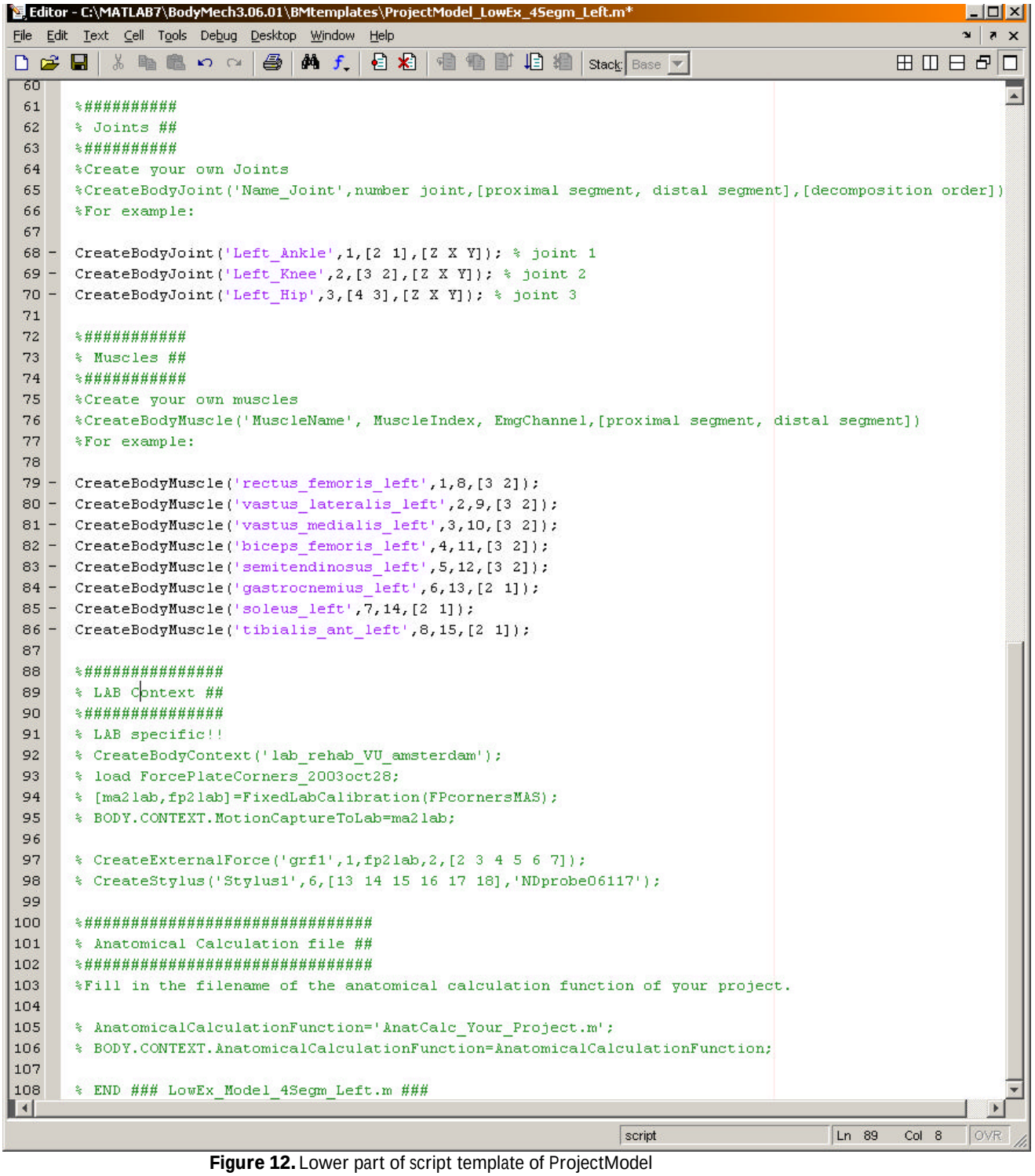


Figure 12. Lower part of script template of ProjectModel

To create your own ProjectModel:
Edit in this script and adapt all BODY field content to the requirements and protocol of you own Project (type segment names and index, corresponding marker indices, joint names + indices, muscle names and indices, fill in LABcontext information) and save the script under a different filename, preferably in your own ProjectFolder.
In the template file; the main instructions are given by the comment lines for each part. Note that all BODY.CONTEXT fields are not filled and should be adapted according to your LAB specifications and calibrations (see next section). After completion of this scriptfile; save is as a unique filename in YourProjectFolder. This ends the creation process of your ProjectModel file.

For the DEMOproject here: this CONTEXT part was 'uncommented': all fields are filled according to VUMC_LAB specifications; and saved as DEMOmodel_LowEx_4segm_Left.m inde DEMOproject folder.

ad 2. How to Define Anatomical Calculation Function

Dependent on the conventions adopted at your LAB for each joint, an anatomical calculation function must be designed for the Project.

BodyMech provides several templates of anatomical calculation functions for the joint of the lower extremity, based on CAMARC conventions (Capozzo et al, 1995) and the ISB standards (2001, 2005)
When this suits your experiments and convention, you may use one of the templates to edit and / run for your specific set-up.

In Bodymech:

→ select **`Start Project`** from the *Project menu*.

→ and select **`Edit Anatomical CalculationFunction (from template)`**

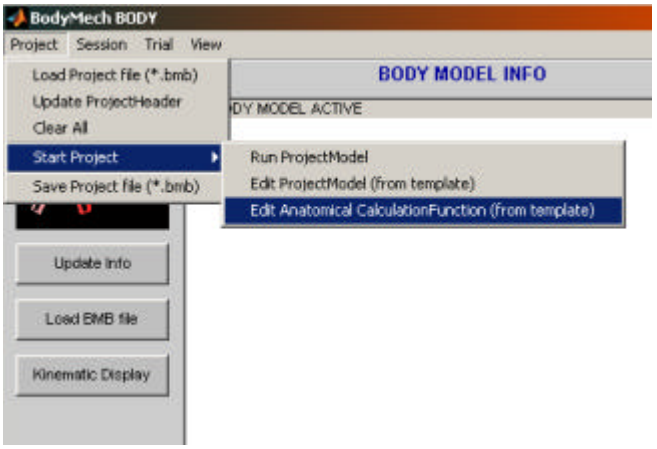


Figure 13.

Browse to the BMtemplate folder and select one of the files . For the DEMOproject the file AnatCalc_lowEx_4Seg_left.m is selected.

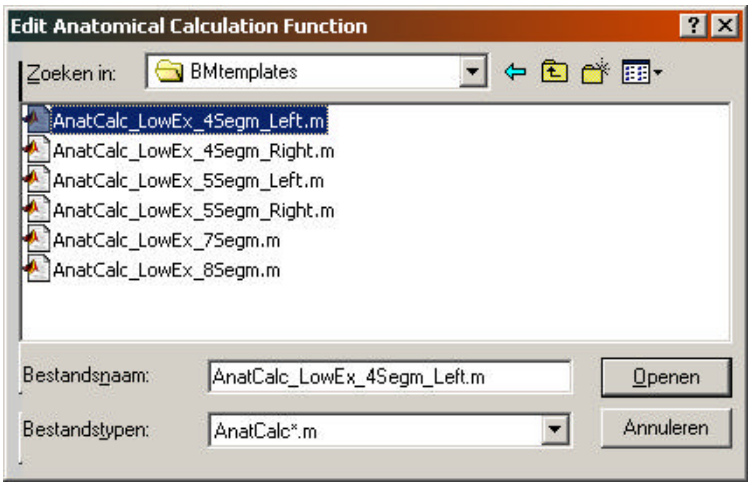


Figure 14. Select one of the AnatCalc template function to edit

This will evoke the Matlab Editor and shows a scriptfile of a template of a Anatomical CalculationFunction.

To create your project specific Anatomical Calculation function:

Check this function file for your own project and edit where necessary. When the number of segments and joint coincides with your Project set-up (segment names and index; joint names and index etc..) AND you agree with the convention of anatomical calculation for joint axes, planes and joint center calculation (i.e. CAMARC project), you do not have to change anything and may save this template under your unique projectName and in your own ProjectFolder.
Save the file with a unique filename in YourProjectfolder (also recommended when no adaptations were made).
Tip: put this filename in your ProjectModel script in the CONTEXT part

ad 3. Set your LAB specifications

The following issues must be solved to adapt your own LAB specifications to the BodyMech structure.

. **Defining LAB coordinate system**
. How to write data import functions

These will be explained below (also with aid of some functions form the folder BMtools):

Defining LAB coordinate systems

This section describes the general idea of defing coordinate systems in your lab and the principles of writing a function FixLabCalibration.
There seems no point in using another coordinate system than the one that is defined by the Motion Analysis system that you use. In the end, your results will be independent of the experimental coordinate system that was used, and reported in anatomical terms. Very often this is true, however two considerations also apply:
1. During the computational stages, you might want to inspect your data using the visualisation interface of BodyMech.
2. When you use a reference position, instead of - or next to - an anatomical calibration, you will need to align the body with the global coordinate system.

In BodyMech the convention of the International Society of Biomechanics is adopted, see Figure 15, which means that the Y axis is pointing upward, in contrast to many systems assuming Z to be upwards.

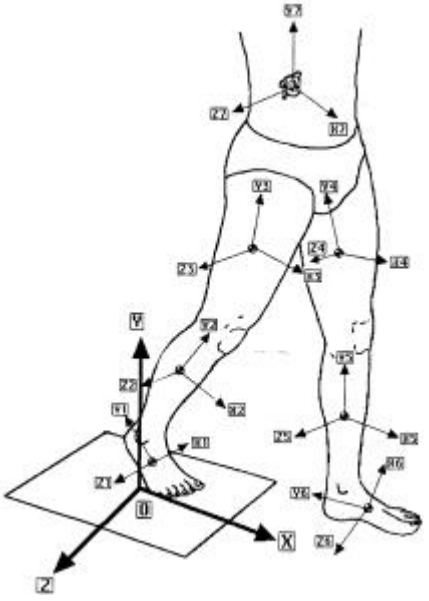


Figure 15: adapted from : Wu G, Cavanagh PR. ISB recommendations for standardization in the reporting of kinematic data. J Biomech. 1995, 10:1257-61

So in order to define a coordinate system, it is sufficient to establish an origin ${}_{mas}O$; some point in the X direction ${}_{mas}X$; and some point in the Z direction ${}_{mas}Z$. (Just stick 3 markers on the ground).

From that point on the following holds:
${}_{lab}\boldsymbol{P} = {}_{lab}R_{mas} * {}_{mas}\boldsymbol{P} - {}_{mas}O$, *with* ${}_{lab}R_{mas} = ({}_{mas}R_{lab})^{-1}$

and: ${}_{mas}R_{lab} = ({}_{mas}X_{lab} ; {}_{mas}Y_{lab} ; {}_{mas}Z_{lab})$

${}_{mas}X_{lab} = {}_{mas}X - {}_{mas}O$
${}_{mas}Y_{lab} = cross({}_{mas}Z - {}_{mas}O , {}_{mas}X_{lab})$
${}_{mas}Z_{lab} = cross({}_{mas}X_{lab} , {}_{mas}Y_{lab})$

This gives you the translation vector and (after normalisation) the rotation matrix between the two coordinate sytems. It is more convenient to define a transformation matrix:

$$ {}_{mas}T_{lab} = \left|\begin{array}{c|c} {}_{mas}\boldsymbol{R}_{lab} & {}_{mas}O_{lab} \\ \hline 0 \quad 0 \quad 0 & 1 \end{array}\right| $$

Subsequently the inverse of ${}_{mas}T_{lab}$; i.e. ${}_{mas}T^{-1}_{lab} = {}_{lab}T_{mas}$ is used and must be applied to each measured marker time-series in the Motion Analysis System.

In BodyMech this transformation matrix must be defined in the **`ProjectModelFile`** and stored in the field:
`BODY.CONTEXT.MotionCaptureToLab`
Doing so,it will be automatically applied to each marker time-series, after importing a marker file.
However, in many gait analysis studies a forceplate is also used, and therefore it is more convenient to use the corners of the force plate as ancherpoints to define the laboratory coordinate system. The origin of the lab coordinate system is in the middle on the surface of the forceplate.

Consider the following top view :

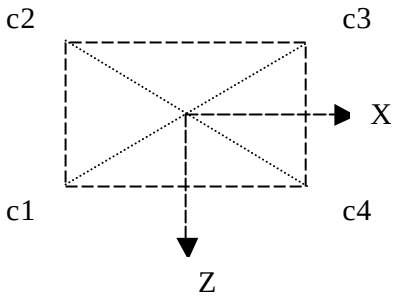


Figure 16.Top view of a surfac uise for LAB calibration

The points C1, C2, C3 and C4 are the coordinates [x;y;z] of the corners of the (rectangular) forceplate in the MAS-coordinate system. It can be probed, using a stylus, or using 4 markers that are placed on the corners and recorded in a static view.

The function **`FixedLabCalibration([C1,C2,C3,C4])`** will give you the proper transformation matrix, with the origin at the geometrical centre of the forceplate, and the directions of the lab-axis as shown.

When you want to include the force measurements in the analysis, the coordinate system of the force plate should be spatial synchronized with the lab coordinate system. To do so, the mechanical coordinate system of the force plate should be known. This can be found in the technical reference manual of the forceplate as provided by the manufacturer. Usually the mechanical coordinate system of the force plate is defined relative to the geometical centre of the surface of the force-plate and this is exactly where the lab coordinate system is defined ! This makes it rather easy to draw up a tranformation matrix.

In the gait lab of the human movement laboratory of the VU University hospital in Amsterdam, the system according to Figure 17 applies:

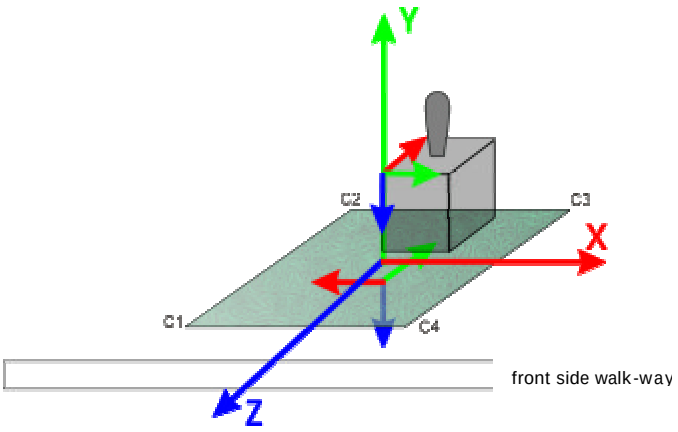


Figure 17.The dark grey cube establishes the MAS coordinate system, the large arrows represent the lab coordinate system. The mechanical coordinate system of the force has its origin some cm. below the surface and using the Z-direction for the vertical, which is quiete common in force plates.

```
% The following applies to AMTI OR6-5-1000 ; serialno. 3509

% the geometrical forceplate orientation towards laboratory:
xmech_offset=0.;
ymech_offset=-0.68/1000.;
zmech_offset=41/1000.;

% the force-plate is covered with linoleum that is 3 mm. thick
zmech_offset=zmech_offset+.003;

% transformation
fp_geometrical_to_fp_mechanical= ...
   [eye(3),[xmech_offset ymech_offset  zmech_offset]' ; 0 0 0 1];

% the geometrical forceplate orientation towards laboratory:
% X_lab=-X_fp ; Y_lab=-Z_fp Z_lab=-Y_fp :
fp_geometrical_to_lab= [ -1  0  0  0 ;...
                          0  0 -1  0 ;...
                          0 -1  0  0 ;...
                          0  0  0  1 ];

forceplate_to_lab=fp_geometrical_to_lab...
                *inv(fp_geometrical_to_fp_mechanical);
```

This 4x4 transformation matrix will be stored in:

`BODY.CONTEXT.ExternalForce(1).ForceSensorToLab`
and used by BodyMech to transform all force sensor readings into the lab coordinate system.

How to write data import functions

This section describes the general idea of a importing measurement data into the BODYMECH structures.
Assumption: all data files contain rectangular data structures, i.e. a number of channels that all are sampled on the same equidistant instances of time.

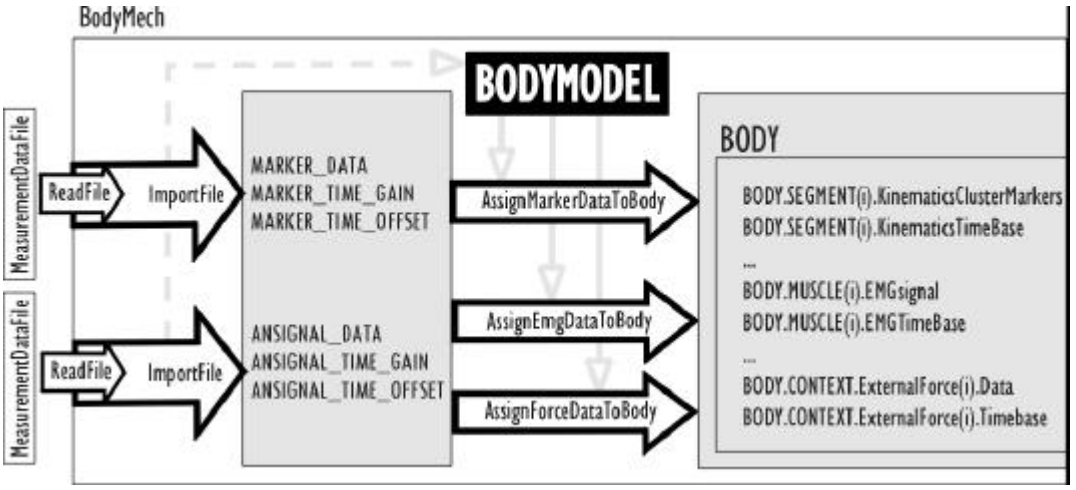


Figure 18. General illustration about import function. Grey area's are global variables within BodyMech (BODY fields indicated are not identical with current version).

Importing measurement datafiles (eg. OptoTrak NDF files, Porti TMS files, or labview DAR files) is implemented in a specific **ImportFile**

An ImportFile has to be written for BodyMech for each type of measurement data file that needs to be imported in BodyMech:

- If no filename is passed to Loadfile as a parameter, LoadFile will invoke the file browser
- For actual reading the datafile, ImportFile uses the general purpose READFILE which might use a particular dataformat.
- If necessary, Loadfile uses calibration values of the labinstrument to calculate the measurement signals into engineering units
- Importfile handels synchonisation information, resulting in time-gain and offset.
- Importfile puts the data into the Global data structures of BodyMech, preserving the channel structure
- In specific cases ImportFile might use that header information of a MeasurementDatafile to define the BodyModel (dashed line in Figure 18), this is the case when model configuration is dependent on userinput at the time of measurement. This requires that at least a name for each channel is passed in the measurement datafile, e.g. muscle names as channel idenitification in EMG equipment.

Available data file import functions in BodyMech

BodyMech can read or import the following data files (reated MATLAB function in folder: LABfunctions):

Optotrak file – BodyMech function: **`BmimportNdf.m`** ; imports data file with raw 3D coordinates of the clusters used on the subject. The extension of the file is free to choose and usually the subject code is used for this purpose. 3D files start with *'C#0000.<ext>'.*

DAR file – BodyMech function: **`BmimportDAR.m`** : imports data file with measured force plate and EMG data acquired with SYBARDAR application (DICOM format)

MDF file – BodyMech function: **`BmimportMdf.m`** : imports a Sybar measurement data file with the measured force plate and EMG data in ASCII format.

TMS file – BodyMech function: **`BmimportTMS.m`** :imports a file with EMG data acquired with the TMS Porti software (Twente Medical Systems, Enchede, The Netherlands).

Use BodyMech & BodyModels

You can apply Bodymech for kinematic analysis when all Project and LAB requirements are met (previous section); i.e.

- ❑ Project script function file is defined (m.file)
- ❑ Anatomical Calculation function is defined (m.file)
- ❑ LAB specifications are set

Basic models for BodyMech are the Project, Session and Trial Models. The creation and final analysis will be explained and demonstrated in the next sections:

Project

You may start with BodyMech for the kinematic analysis:

All steps will be illustrated with de DEMO data set.

Run your Project scriptfile within BodyMech GUI:

→ select '**`Start Project'`** from the *Project menu*.
→ and select **`Run ProjectModel`**

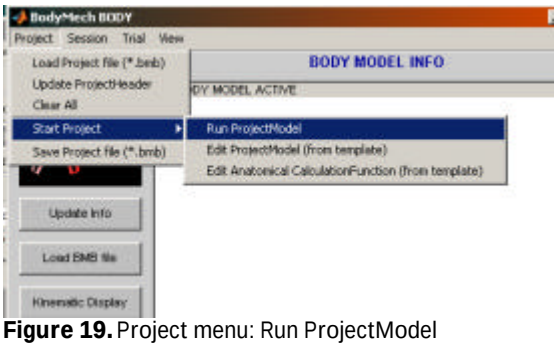


Figure 19. Project menu: Run ProjectModel

A window appears to select your ProjectScriptfile to Run:

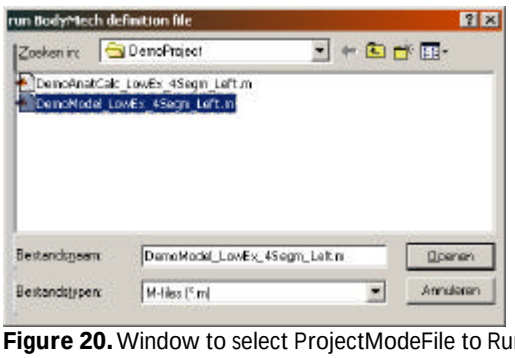


Figure 20. Window to select ProjectModeFile to Run

By running this scriptfile: you have build a BODY structure array with content in all relevant fields for a ProjectModel.
In the BODY MODEL INFO screen, all relevant information for this Project BODY is shown (Figure 15). *(compare thisProject info with the Trial info from Figure 4 to identify the differences)*

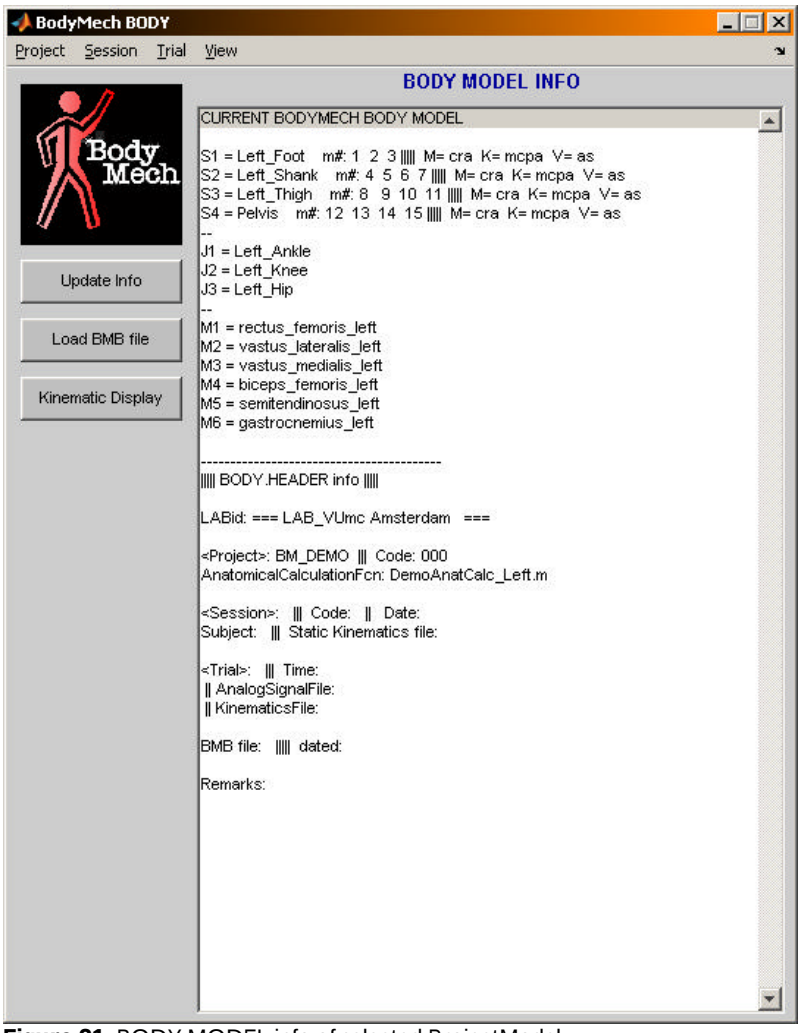


Figure 21. BODY MODEL info of selected ProjectModel

Save Project File

Save your BODY (ProjectModel) as a BMB file
→ select '**`Save Project file (*.bmb)'`** from the *Project menu*. First, a window appears in which you may to edit your filename:

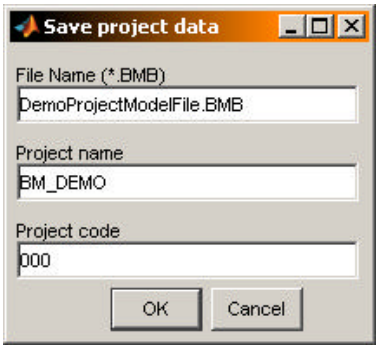


Figure 22. BODY MODEL info of selected ProjectModel

Browse to your Project folder and save the .BMB file

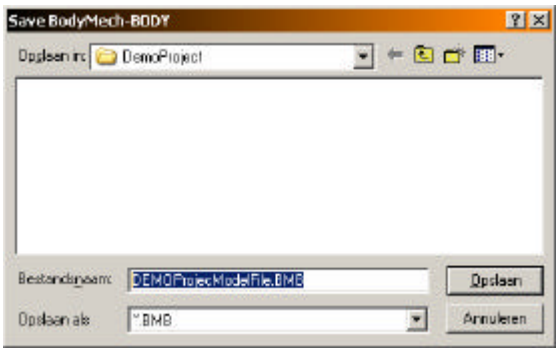


Figure 23 Save BMB file of selected ProjectModel

NOTE:
In the **BODY MODEL Info panel**, the letters **S,** J and **M** show the used segments, joints and muscles used in this body model.
Lowercase **#m** indicates the numbers of the markers on each segment (marker setup).
The letters behind the marker setup of each segment show the status of the calculations. In the presented example (Figure 15) all letters are lower case, which means that no analysis is done yet. E.g. in Figure 1, all letters are UPPER case, which means that the corresponding calculations are done. See Table1 for more details about the significance of each letter.

Model **M = CRA**	**Kinematics** **K = MCPA**	**Visualization** **V = AS**
Cluster Definition	**M**arker Kinematics	**A**natomical markers
Reference Posture	**C**luster Kinematics	**S**tick markers
Anatomical Calibration	**P**osture referenced kinematics	
	Anatomy referenced kinematics	

Table 1 Explanation of Body Model List indicators

Session

For each session within each Project, i.e. usually each subject, a '*Session model*' has to be made. For this, the previously created '*Project model'* is required as a basis, since these parameters are necessary to generate a Session model. If no Project File is opened yet (blank info screen); 'Load Project File' from the Project Menu.
In a Session model, more fields of the BODY structure are filled (see appendix B). These are specific header information, measurement data of a reference position and importing anatomical calibration files.

Define Session Header

Start Defining your Session Model:

→ Select **`Define SessionHeader`** from the *Session menu*

An input window appears and all requested information can be typed.

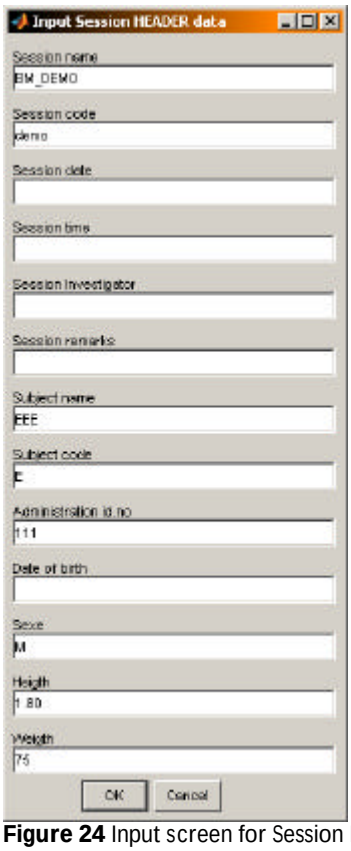


Figure 24 Input screen for Session HEADER parameters.

Import static trial

For the definition of Cluster frames (i.e. the specific cluster position relative to the segments of the subject), a marker registration file has to imported. This can be done with the following actions:

→ select **`Import Static Trial`** from the *Session menu*

For the DEMO file **`C#003.bny`** (one of the anatomical calibration trials) can be used.

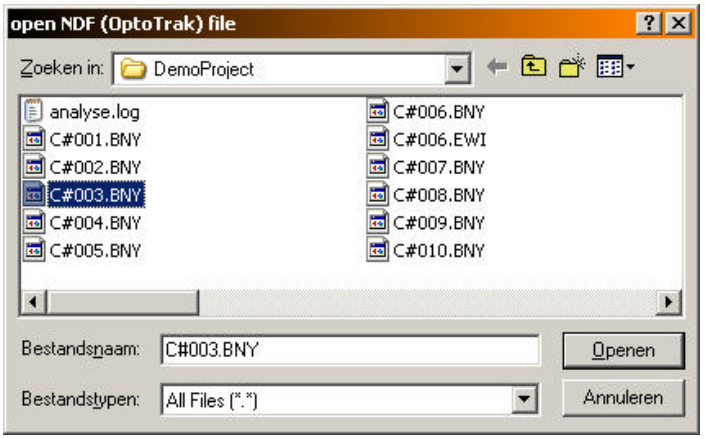


Figure 24 Input screen for Session parameters
In the BODY MODEL INFO panel, the codes K=mcpa is changed in K=Mcpa.

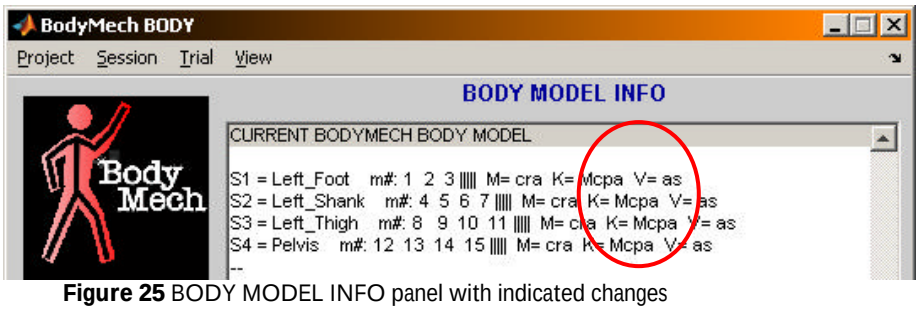


Figure 25 BODY MODEL INFO panel with indicated changes

NOTE: For a static trial to Define Cluster Markers; all cluster markers must be visible for at least one time frame. Check your experimental administration for specific trialname/number.

Define marker clusters

For ClusterDefinition; a time-instance has to be selected in which it is sure that all measured markers are visible.

→ select '**`Define Marker Clusters'`** from the *Session Menu*.
A plot appears in which the number of markers is visible over the sampled time for each defined segment. With a cross haired cursor select the time instance by a left mouse click (see Figure 26).

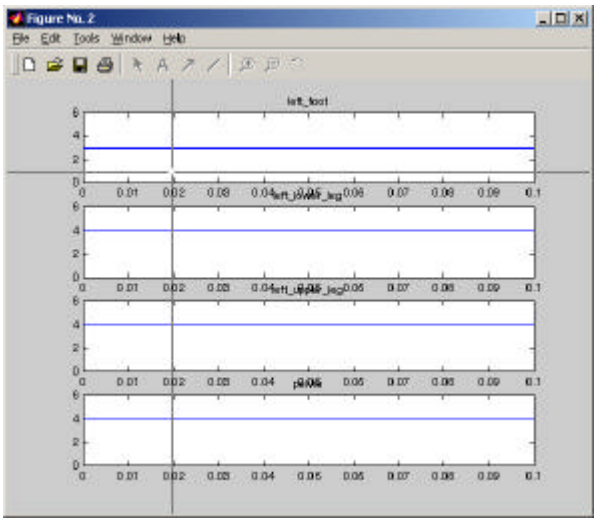


Figure 26 Selection of the time instance for each cluster frame of the segments

In the BODY MODEL INFO screen the codes M=cra is changed in M=CRa.

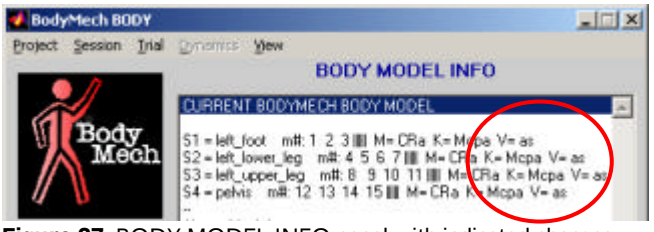


Figure 27 BODY MODEL INFO panel with indicated changes

Define Reference Posture

The next step is defining a reference position.

NOTE: Despite that the reference position still has to be defined, the r is already an 'R' (see Figure 21. This is explained by the fact that the recording of a ReferencePosture is simply a byproduct of the ClusterDefinition. When you have defined the cluster frames by selecting a time-instance, the pose of the body in that frame is also stored as the ReferencePosture. If this position can be used as a ReferencePosture for the calibration of your model ,the following action can be skipped.

However if another measurement corresponds to a standard ReferencePosture, this file has to be loaded, which is then used to define reference position.

For a standard reference position, (i.e. static calibration) the subject takes a position in which the segment axes are aligned with the laboratory axes.

In that case: load the marker file for this reference position:

→ **`Load Static Trial`**; (select file of reference position)

and subsequently

→ Select '**`Define Reference Posture'`** from the Session Menu,
a plot appears similar to the one of Figure 26 which requires the same action:

→ select a time-instance which has all markers visible

In the DEMOproject, the first static measurement (C#0003,BNY) use for ClusterDefinition suits the Reference Position. Therefore, there is no need to load a new file.

Tip: Combine these two steps by using a single static measurement in which the subject took the preferred reference position. i.e. a "static calibration".

The BODY MODEL INFO panel, will show M=CRa for each segment. You have now explicitly added a (new) ReferencePosture.

Check Appendix B to see which functions BodyMech uses to define the cluster markers and what fields are filled after this step.

Define Anatomical Markers

The second type of reference to marker registrations of rigid segments is the anatomy of the bones that are the basic rigid structure within a Body Segment. For this purpose, the position of specific superficial bony landmarks are defined by palpation. Wit a special probe the positions of these landmarks are recorded (see LABnet for more details).
In the creation of the Project model that is used in this demo case, these bony landmarks are named for each segment.

When anatomical calibration is used, the 3D coordinates of predefined *Anatomical Landmarks* have to be imported to the Session model. These files contain data which determine the position of each anatomical landmark relative to clusters on each segment.

→ select **`Define Anatomical Markers`** from the *Session menu*

This will invoke the file browser, which will prompt you for every defined anatomical mark (the name is indicated in the header of the browser) to select the appropriate file. It follows all the anatomical marks for all of the segments, as selected when creating the Project model. Please load all the files in accordance to table 2#. *(derived from the experimental administration).*

Experiment This is done by palpating the bony landmark by means of a stylus, and perform a short measurement of all the markers of the segment-cluster as well as all the markers of the stylus. (which also might be a single marker)

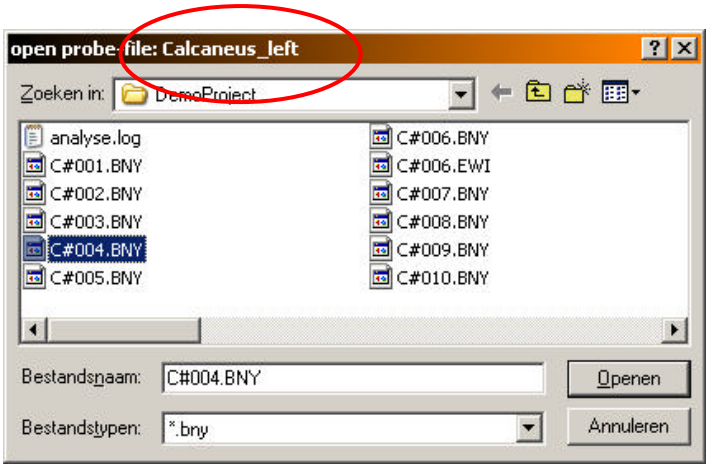


Figure 28 Selection of one of the anatomical landmark ('calcaneus_left')

NOTE: This step is critical. Loading the wrong file creates an erroneous anatomical model.

The list of filenames that contain these anatomical measurements of each bony landmark for the DEMO is shown in Table 2.

Bony Landmark	File name
MTP_V_left	C#002.bny
MTP_I_left	C#003.bny
Calcaneus_left	C#004.bny
Malleolus_Lateral_left	C#005.bny
Malleolus_Medial_left	C#006.bny
Tuberositas_Tibea_left	C#007.bny
Caput_Fibulae_left	C#008.bny
Epicondyl_Lateral_left	C#009.bny
Epicondyl_Medial_left	C#010.bny
Trochantor_Major_left	C#011.bny
ASIS_Left	C#012.bny
ASIS_Right	C#013.bny
PSIS_Left	C#014.bny
PSIS_Right	C#015.bny

Table 2 Files containing bony landmark measurements.

In the Session model, the coordinates of the bony landmarks are included in the corresponding fields (see appendix B). These positions will be the virtual anatomical marks of the BODY model. In the MATLAB command window you can also view the names of the anatomical markers:
→ *type*: `BODY.SEGMENT(1).AnatomicalLandmark.Name` *to see the names for the first segment.*

After completion of the definition of all anatomical markers, the BODY MODEL INFO panel shows M=CRA. You have added the *Anatomical Reference* so now the Session model is completely defined and is ready to be used for calculation of the joint kinematics from the trial data.

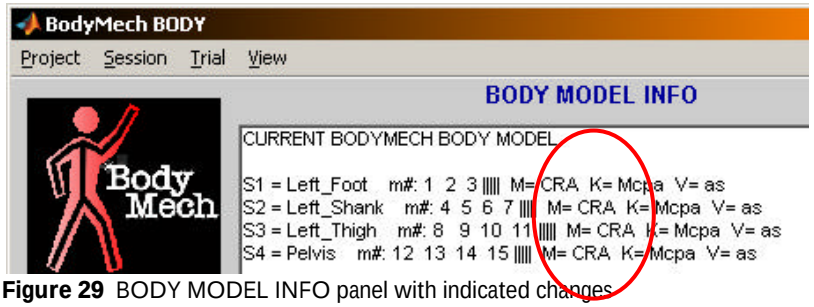


Figure 29 BODY MODEL INFO panel with indicated changes

Save Session file

Save your BODY (SessionModel) as a BMB file
→ select '**`Save Session file (*.bmb)'`** from the *Session menu.*
First, a window appears in which you may to edit your filename :

Figure 29 Update Session Header

Use a unique name for your SessionModel to store :

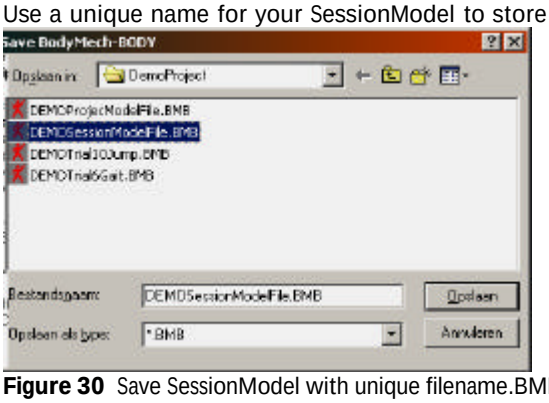


Figure 30 Save SessionModel with unique filename.BMB

About references

A reference to the human body can be added in two ways:
1. Recording a calibration posture. This reference posture is used to calibrate the local clusters to the global frame.
2. Recording an anatomical relationship

Note: The attractive option of the first approach is that it does not assume that the markers of the segments they define have an a priori relation to anatomical (bony) landmarks. So no specific anatomical model is needed. The second approach makes it possible to use an anatomical model, without the need to switch to anatomical oriented markers. However, then it is necessary to add "virtual markers" to a segment, by initial calibration (anatomical probing).

However, although the architecture of BodyMech is mainly based on this approach, also anatomical based marker configurations (Helen-Hayes or Cleveland) can be handled. Purely anatomical based marker configurations will need no calibration reference measurements for either cluster definition, or reference posture, as all follows from the description of the marker setup. In practice a mixed approach is often used, employing an additional single static calibration (with e.g. additional markers, like a knee alignment device). This is not elaborated here, but ways to work around this, for these common marker setups mentioned, is foreseen.

When the clusters and anatomical landmarks are calibrated for a subject, the subjects' measurements can be processed. This chapter shows how to import measurement files and calculate kinematics for one of the movement trials of the DEMO project.

If no BODY model is open yet: select **`Load Session file`** from the *Session menu*.
In a Trial model, more fields of the BODY structure can be filled (see appendix B). These are specific Trial header information, measurement data of the movement and extra data, e.g. EMG and reaction force.

Define Trial Header

Start your Trial Model by defining HEADER information

→ Select **`Define Trial Header`** from the *Trial menu*.

An input window appears and all requested information can be typed. Click OK after completion.

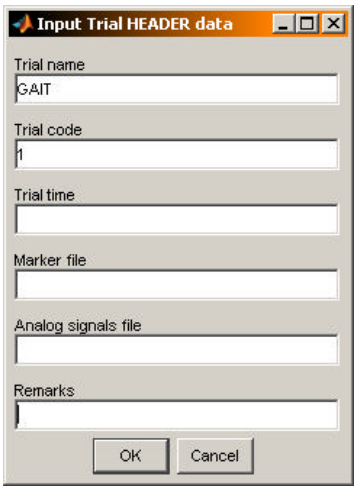


Figure 31 Input screen for Trial parameters.

Import marker file

Import the 3D position datafile corresponding to the specific movement *(check your experimental adminstration)*

→ Select **`Import Marker file`** from the *Trial Menu*.

This will invoke the file browser to select the OT-file (*.EWI)

Select and Open **`C#006.ewi.`** *(this includes the marker data of the gait trial of this subject)*

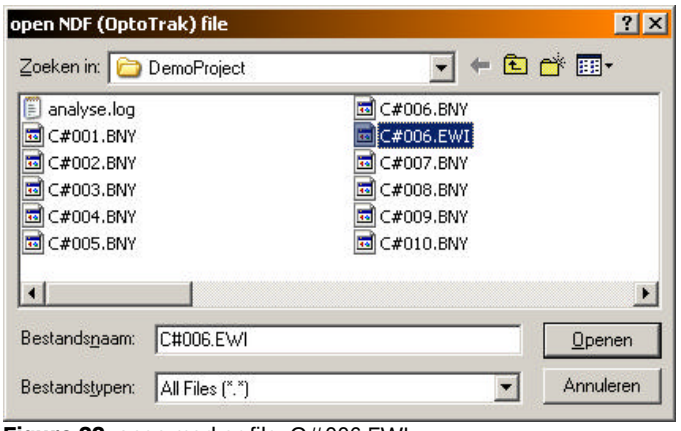


Figure 32 open marker file C#006.EWI

Import analog file

If force plate and/or EMG recordings are made during the specific Trial, this file must also be loaded.

→ Select **`Import Analog File`** from the *Trial Menu*. → **<datatype>**

Select your own data format for used analog data. For the DEMO Trial: MDF format is used for EMG and reaction force data

Note: it is always the responsibility of the user to make sure that analog and marker files are loaded are synchronzed and that apply to the same movement trial.

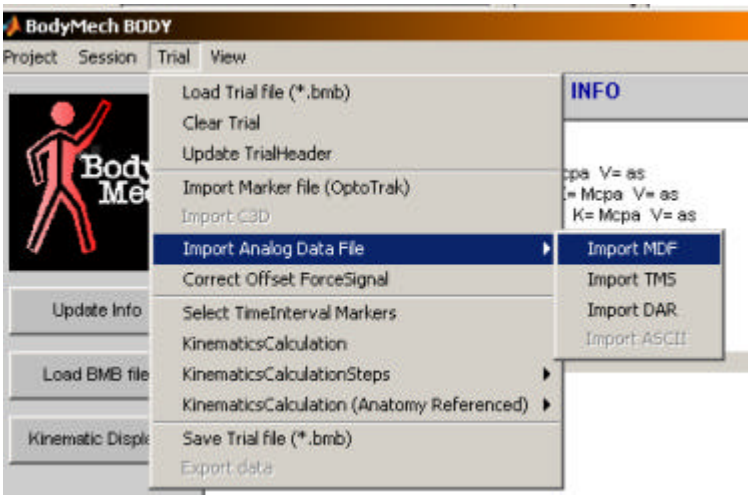


Figure 33 Selecty dataformat for analogdata for this experimental trial

For this DEMOproject select ***`EWIsya06.mdf`***

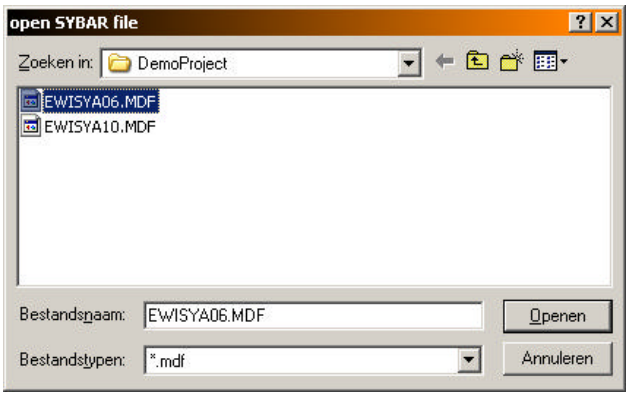
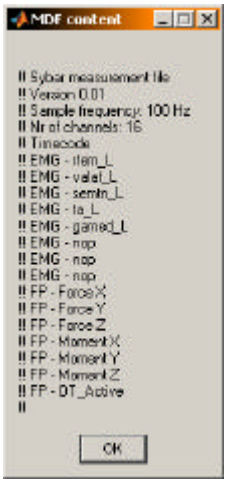


Figure 34; import analog data file

For this type of data (*.MDF); a dialog box will show the header of its content, click OK.



CorrectForcePlate signal for possible offset :

→ Select **`Correct Offset ForceSignal`** from the *Trial Menu*.

Click one time frame where the force has to be zero (unloaded). Force is corrected for a time periode of .4 sec (± 0.2) sec. around the selected time frame.

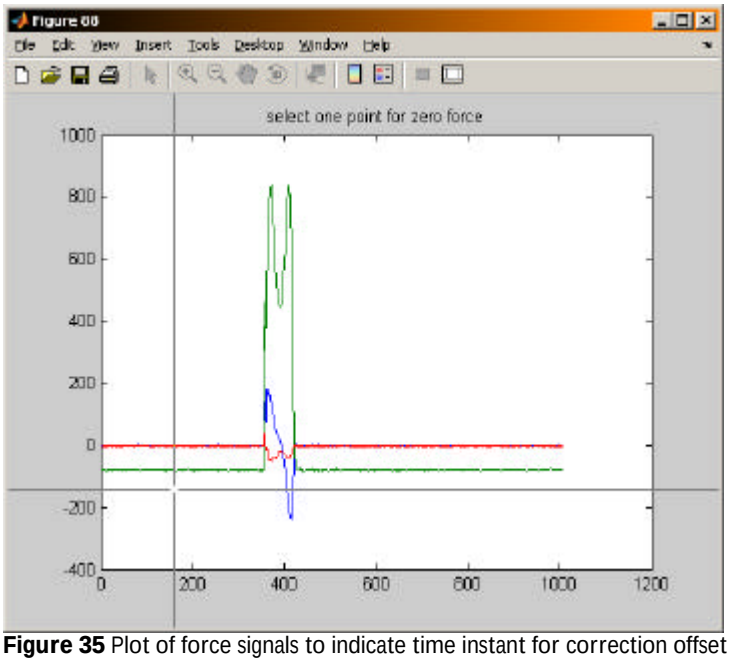


Figure 35 Plot of force signals to indicate time instant for correction offset

Select time interval markers

Since all relevant measurement files are loaded, it is important to make a selection of the interesting part of the measurement first.

→ select '**`Select Time Interval`**' from the *Trial menu*. This will pop up a selection window (see Figure 36).

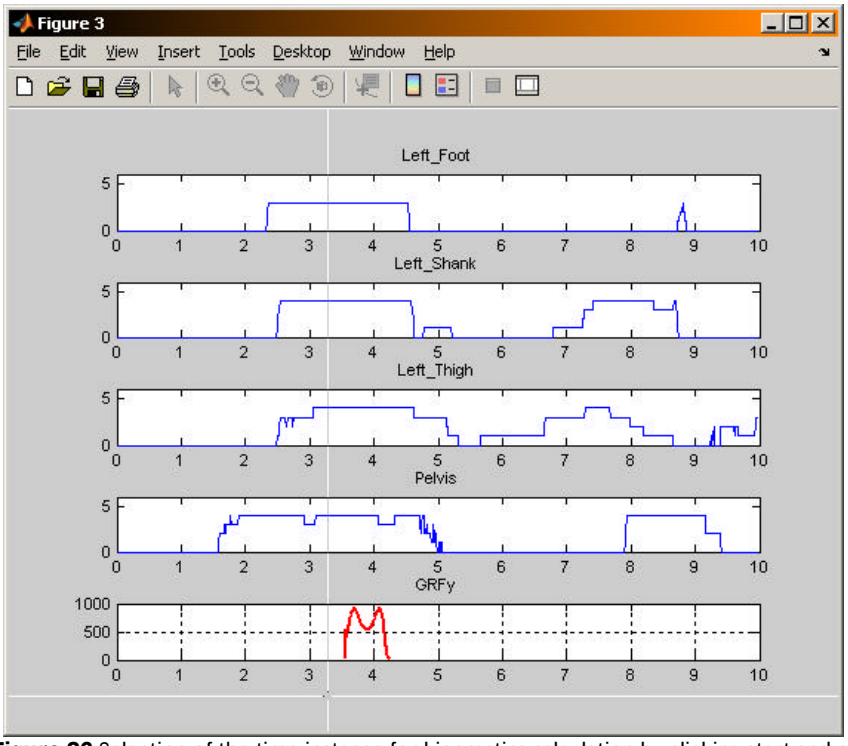


Figure 36 Selection of the time instance for kinematics calculation by clicking start and end point.

The first upper axes (# 4: corresponding to the number of SEGMENTS for the DEMO) show the number of visible markers per segment. This should be at least three for proper identification of the kinematics of each segment cluster (See Chapter IV for theoretical background). However missing markers can be reconstructed by interpolation (see next paragraph). The lowest axis shows the vertical component of the GRF for reference (red lines). A time interval selection takes effect on all marker and all analogue signals.

→ move the cross hair in any axis and left click at start and stop time-instance
→ after two clicks the message appears:

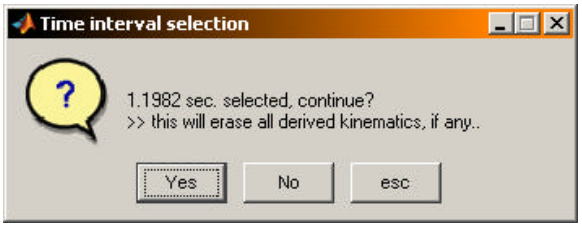


Figure 37. message after time selection. Choose Yes to continue with selected time interval

Kinematics Calculation

To calculate segment and joint kinematics
→ **`KinematicsCalculation`** in the Trial Menu. Appendix B shows which fields that will be filled in the BODY structure.

To get a better understanding of the calculation process, the intermediate steps taken in this calculation can be chosen separately. In this case use the items in the *'Kinematics Calculation Steps'* submenu.
Each step is described below.

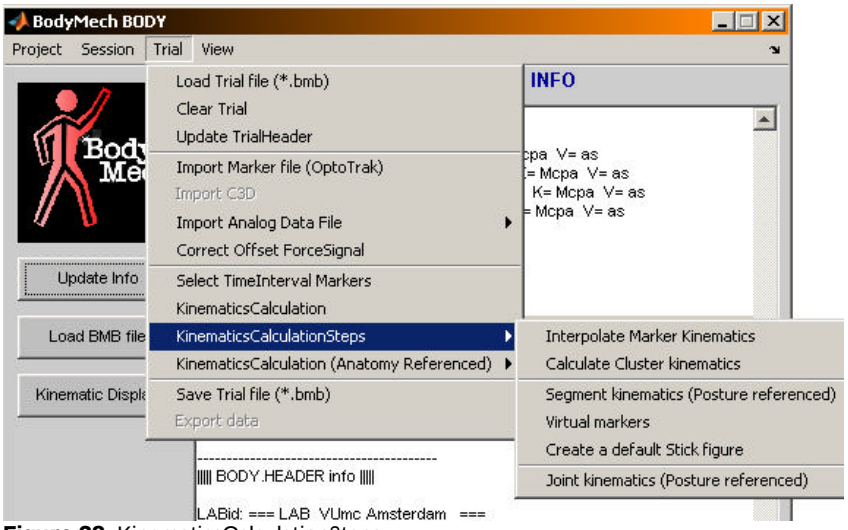


Figure 38 KinematicsCalculationSteps

InterPolate Marker Kinematics

→select **`Interpolate Marker Kinematics'`** for the interpolation of missing markers.

→ use **`Graph Marker`** from the View menu to inspect the results *(interpolated values in red).*

→ select **`show Orthogonal View`** from the view menu;
click Markers to visualize the marker trajectories in space in each plane.

Calculate Cluster Kinematics

With known marker kinematics, next step is to Calculate Cluster Kinematics:
→ select '**`Calculate Cluster Kinematics'`** from the *'Kinematics Calculation Steps'* submenu.

→ select **`'show Orthogonal View'`** from the *View* menu;
click Segment frame in the graphical display windows.

To get from the arbitrary segments towards well-referenced segments, reconstruction of segment kinematics relative to a reference posture is required *(the reference position is already stored in the SessionModel)*.

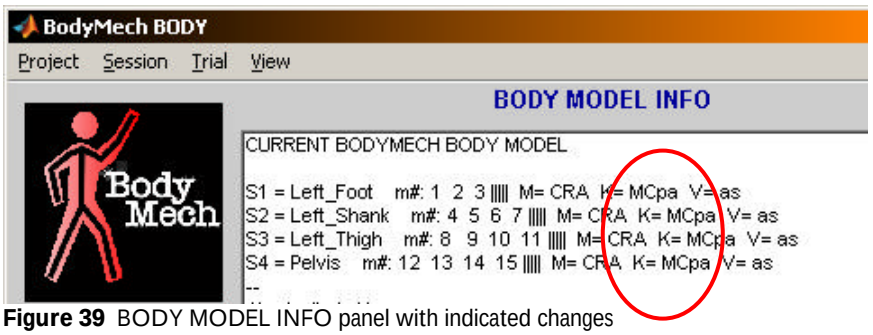


Figure 39 BODY MODEL INFO panel with indicated changes

Segment Kinematics (posture referenced)

→ Select '**`Segment Kinematics (posture referenced)`**' from the kinematics menu.

→ select **`'show Orthogonal View'`** from the view menu;
click ReferencedSegment frame in the graphical display windows.

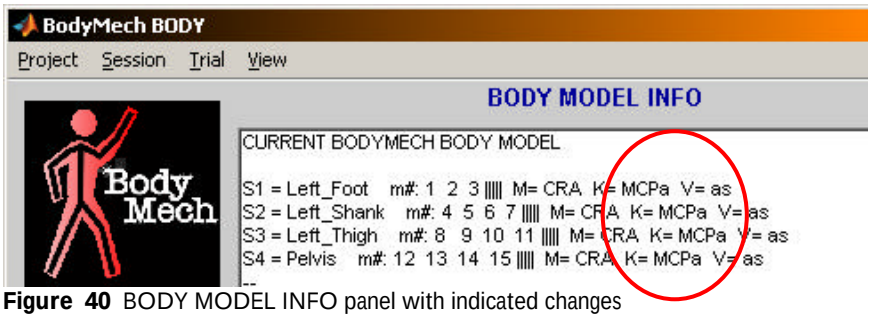


Figure 40 BODY MODEL INFO panel with indicated changes

Virtual Markers

→ Select '**`Virtual Markers`**' from the kinematics menu.

Create a Default StickFigure

→ Select '**`Create a default StickFigure`**' from the kinematics menu.

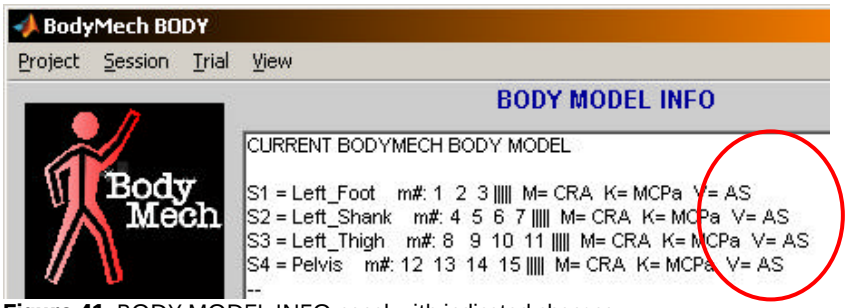


Figure 41 BODY MODEL INFO panel with indicated changes

Joint Kinematics (Posture Referenced)

To calculate joint rotation angles of the joint involved, relative to the reference position:

→ select **`Joint Kinematics (Posture referenced)`**

Kinematics Calculation (Anatomy Referenced)

Alternatively, segment kinematics might be reconstructed relative to bony marks (these anatomical markers are known in the Sessionmodel also).

→ Select' **`Kinematics Calculation (Anatomy Referenced)`**

→ Select '**`Segment kinematics`**

A windows appears to browse and select the pre-defined anatomical calculation Function, as indicated in the ProjectModel (CONTEXT part).
For the DEMO: **`DemoAnatCalc_Low_Ex_4segm_Left.m`** is selected.

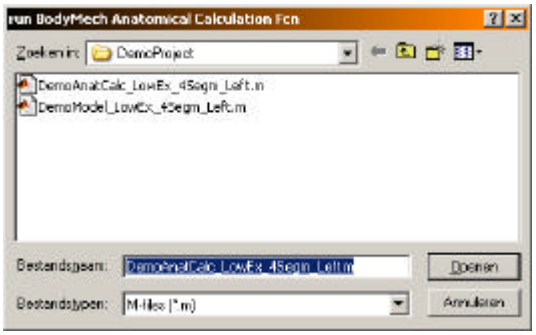


Figure 42 Run selected AnatomicalCalculationFunction

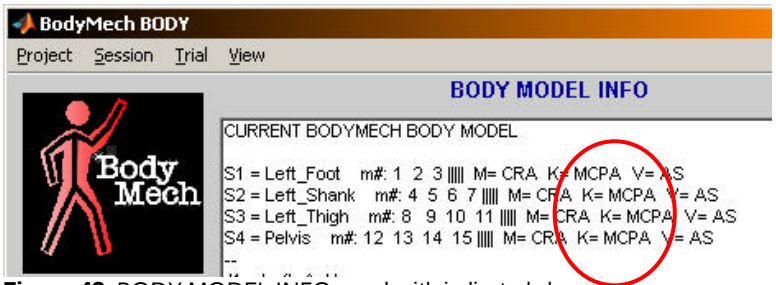


Figure 43 BODY MODEL INFO panel with indicated changes

→ select **`show Orthogonal View'`** from the view menu;
click Anatomical Frames, AnatomicalMarkers and/or StickFigures in the graphical display windows.

Note: This is a project specific file, because its use (definition of anatomical frames) is project specific, and very closely related to the model definition.

Running this file will result in the anatomy referenced kinematics of each segment, and also in the kinematics of the anatomical markers and the stick markers, the latter two will be used for visualization.

Joint Kinematics (Anatomy Referenced)

To calculate joint rotation angles of the joint involved, relative to the anatomical calibrations:
Alternatively, segment kinematics might be reconstructed relative to bony marks (these anatomical marks should be known in the model also).

→ Select **`Kinematics Calculation (Anatomy Referenced)`**

→ Select **`Joint kinematics`**

Save Trial file

Finally, save Trial model with a unique filename

Figure 44 Update Trial header

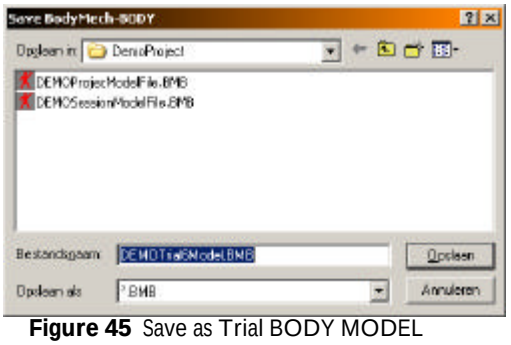


Figure 45 Save as Trial BODY MODEL

From this point all kinematics of the walking trial (#006) of subject EWI in Project DEMO are calculated and can be inspected or visualized by the items in the *View* menu.

View menu

To view the final results of the DEMO trial, use the view menu.
E.g.
→ Select **`Show Orthogonal Views`**

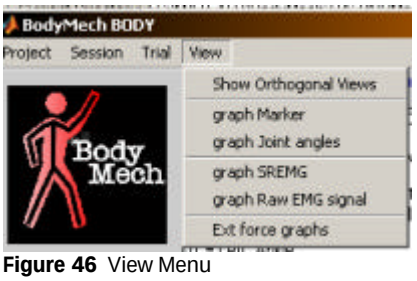


Figure 46 View Menu

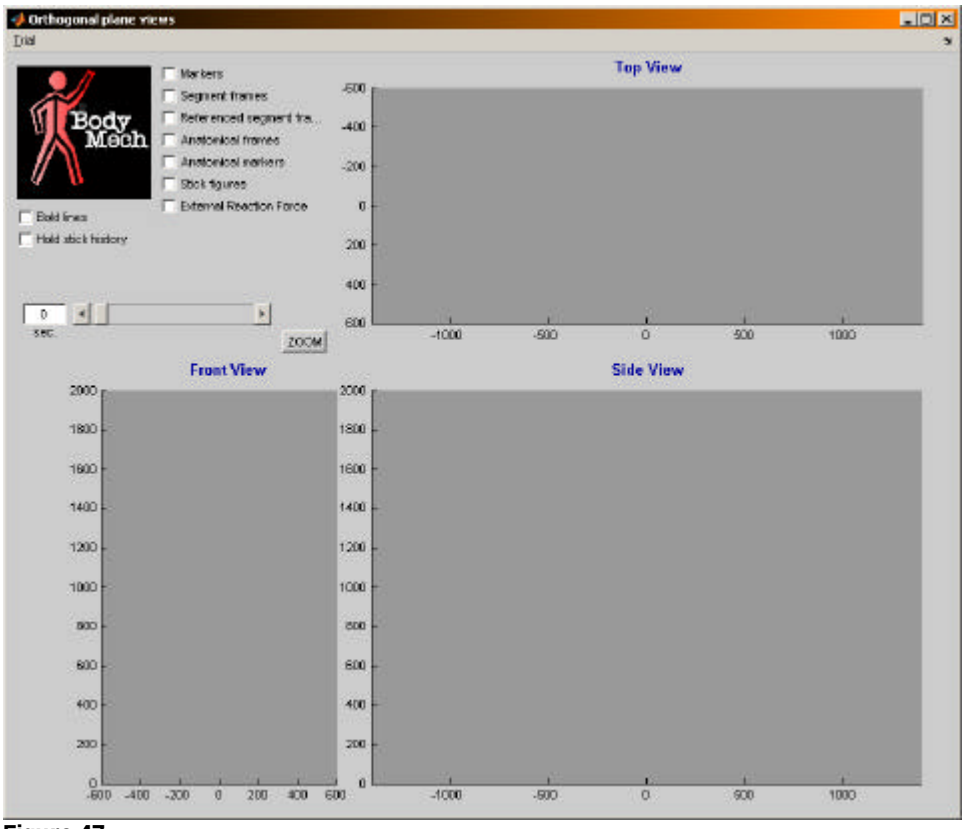


Figure 47

Click on Stickfigures and External Force →

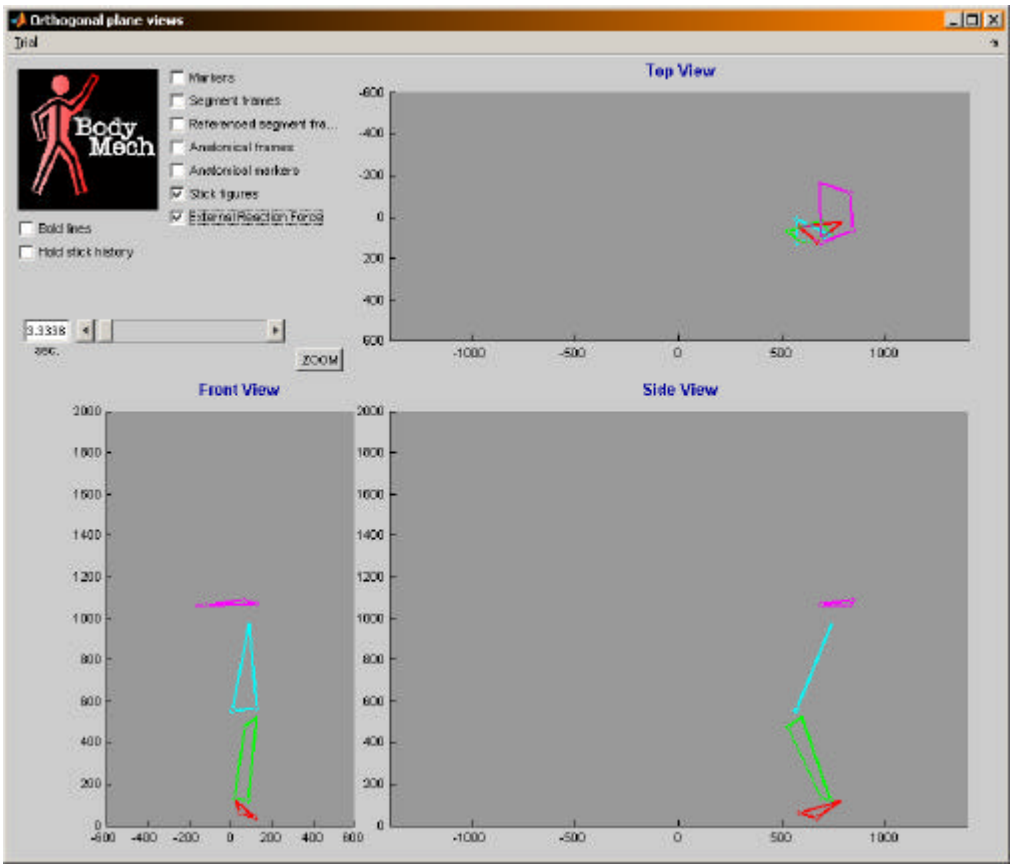


Figure 48

Use the slider to view the stick figures in time of recording →

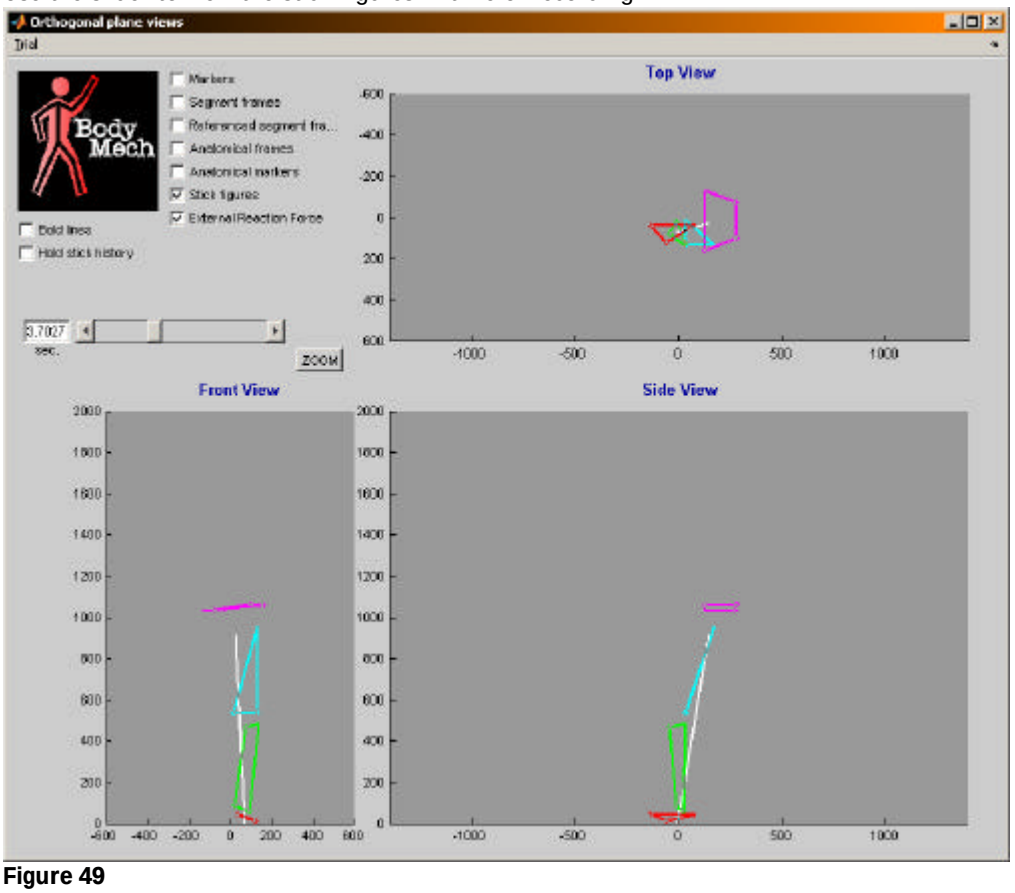


Figure 49

or hold stick history →

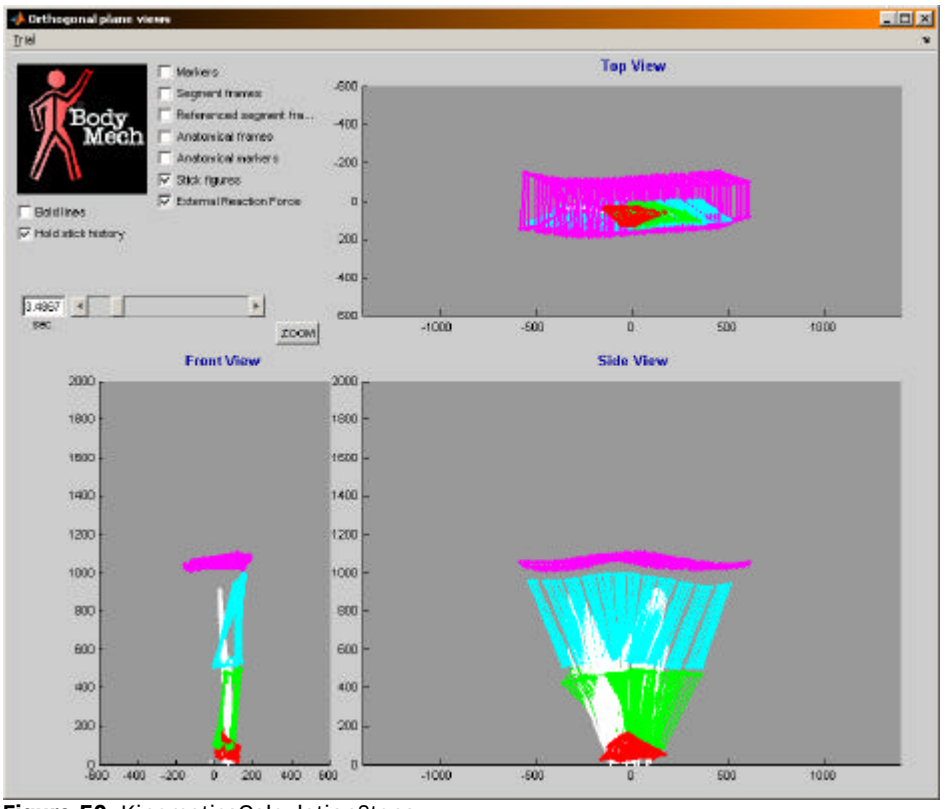


Figure 50 KinematicsCalculationSteps

Try all features in this orthogonal window.

More view functions: e.g.
→ Select **`Graph Joint Angles`**

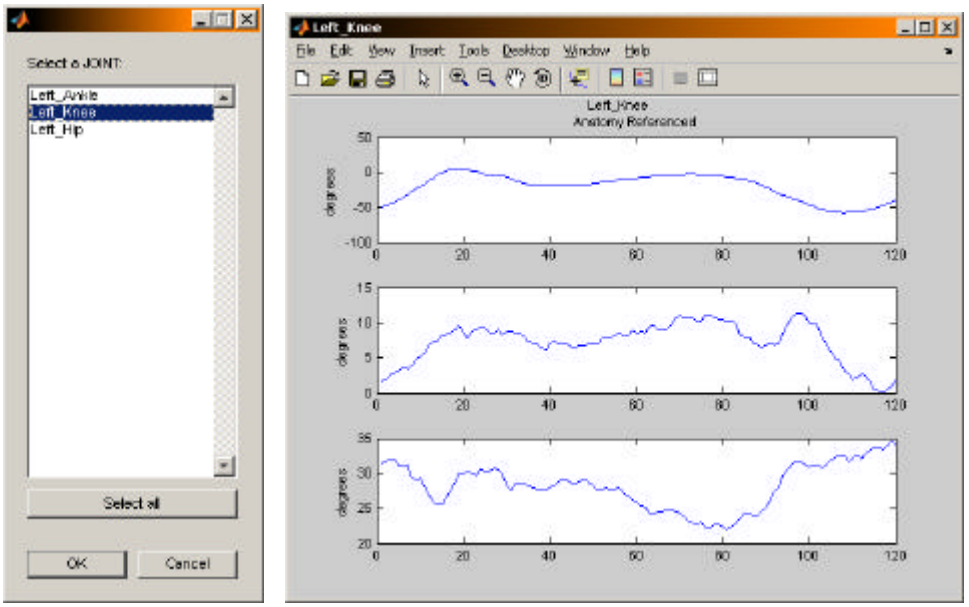


Figure 51 Select Joint to view in graph

or. → Select **Graph SREMG**

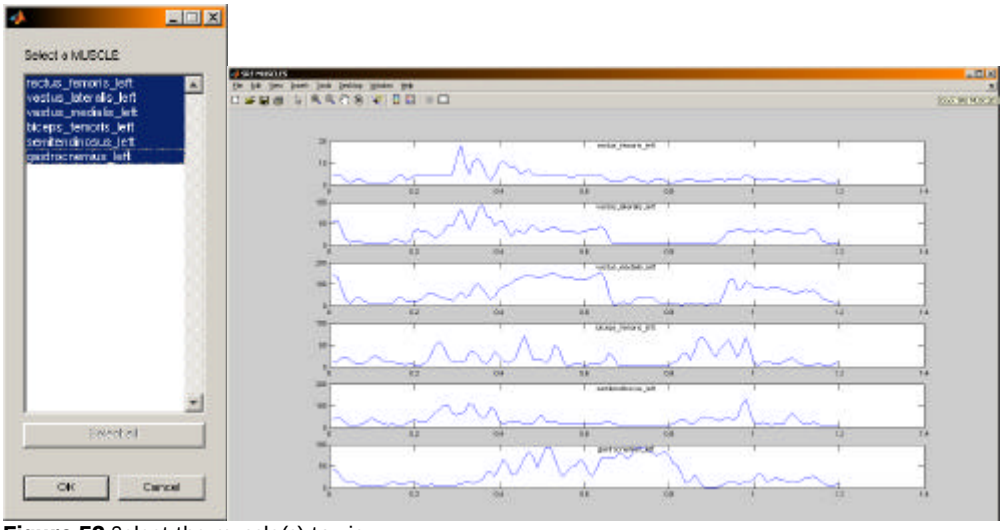


Figure 52 Select the muscle(s) to view

Jump ahead

Now try to analyse another Trial (vertical jump) of this DEMO with Bodymech, using the following steps :

1. Load SessionModel of DEMO
2. import MarkerFile (C#010.EWI)
3. import analog data file (EWISYA10.MDF)
4. correct force signal
5. select time interval of interesting part
6. run kinematics calculation
7. run anatomical referenced calculation
8. view results

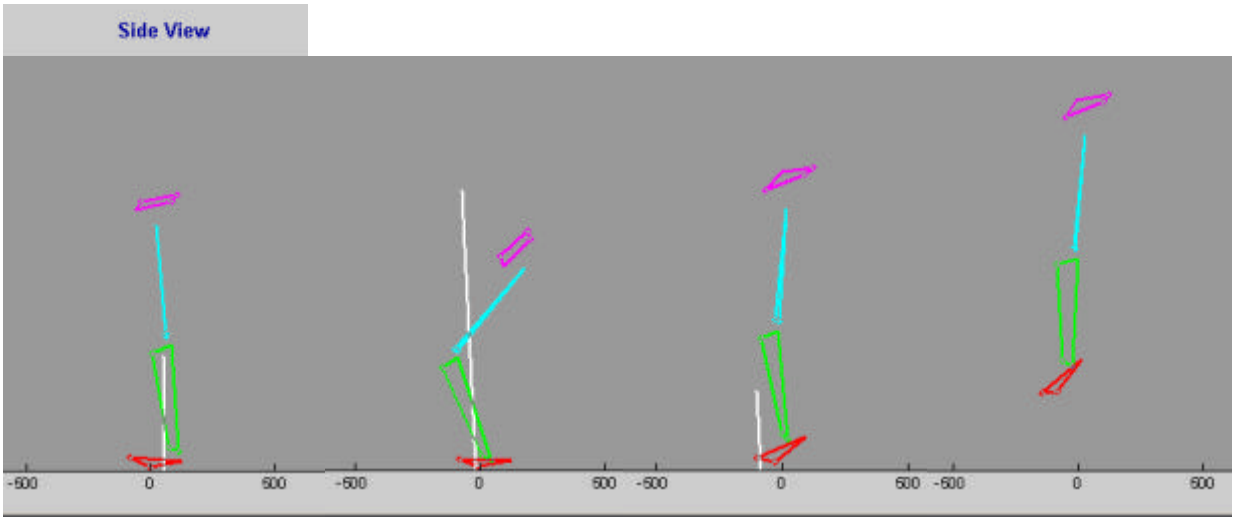


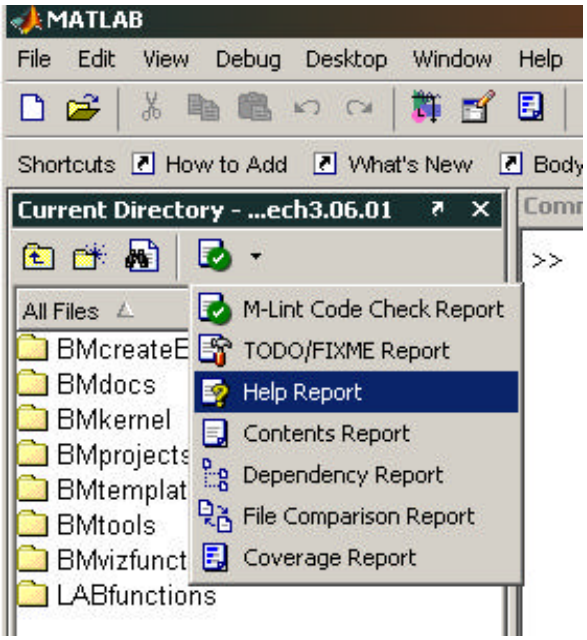
Figure 53 Part of result after final analysis of Trial of vertical jump

BodyMech
version 3.06.01

chapter III. Function Library

Use Matlab directory reports

When you use Matlab 7 or v2006a it is possible to generate a 'Help Report' in the Current Directory Browser. This report gives an overview of all the functions in the current folder. If you wish more information about the BodyMech functions, make sure this folder is selected and click '*Help Report*' as shown below.



All used BodyMech functions are listed per forlder below as a Matlab Help Report:

BMcreateBODY

Scope: Files that are required to Create a Projectmodel

CreateBodyContext	CREATEBODYCONTEXT [BodyMech 3.06.01]: declares the CONTEXT variables of BODY INPUT ContextName : name of the experimental location (usually a lab) PROCESS Generation of a substructure to the global variable BODY that houses contextual information to a movement study. OUTPUT GLOBAL: BODY.CONTEXT % Copyright 2000-2006 This program is free software under the terms of the
CreateBodyHeader	CREATEBODYHEADER [BodyMech 3.06.01]: declares a header to BODY INPUT project_name : name of the project PROCESS Generation of variable that represent a rigid body of the human body OUTPUT GLOBAL: BODY.HEADER % Copyright 2000-2006 This program is free software under the terms of the
CreateBodyJoint	CREATEBODYJOINT [BodyMech 3.06.01]: declares a new bodyjoint to BODY.JOINT INPUT JointName : name of the joint SegmentLinks: [proximal_segment_no distal_segment_no] JointIndex: a priori joint number DecompositionFormat: eulerian; cardanian decomposition format PROCESS Generation of variable that represent a joint of the human body OUTPUT GLOBAL: BODY.JOINT: new cell in the array of joint names % Copyright 2000-2006 This program is free software under the terms of the
CreateBodyMuscle	CREATEBODYMUSCLE [BodyMech 3.06.01]: declares a new bodymuscle to BODY.MUSCLE INPUT MuscleName : name of the muscle MuscleIndex : number of channel EMG signal EmgChannel : number of the column in which the EMGdata can be retrieved from the original datafile (e.g. MDF format) SegmentLinks: [segment of Origin (prox); segment of insertion (dist)] PROCESS Generation of variable that represent a muscle of the human body OUTPUT GLOBAL: BODY.MUSCLE: next cell in the array of muscle names % Copyright 2000-2006 This program is free software under the terms of the
CreateBodySegment	CREATEBODYSEGMENT [BodyMech 3.06.01]: declares a new bodysegment to BODY.SEGMENT INPUT SegmentName : name of the segment SegmentId : number of the segment (optional) MarkerList : list of marker labels assigned to the bodysegment (optional) PROCESS Generation of variable that represent a rigid body of the human body OUTPUT GLOBAL: BODY.SEGMENT: (first free) cell in the array of segment names is assigned

	% Copyright 2000-2006 This program is free software under the terms of the
CreateExternalForce	CREATEEXTERNALFORCE [BodyMech 3.06.01]: declares a new external force to BODY.CONTEXT INPUT ExtForceName : name of the external force ExtForceIndex : indexnumber of the external forces ForceSensorToLab : transformation matrix from forceplate coordinate system(s) to lab coordinate system. Usually FP corner measurements are used to construct this ForceSensorType : either 1,2,3 or 4; 4 is identical 2 except that a sensitivity matrix is used (definition follows C3D conventions) TYPE 1 TYPE 2,4 TYPE 3 CHANNEL (1,i) Force X Force X Force X1,2 CHANNEL (2,i) Force Y Force Y Force X3,4 CHANNEL (3,i) Force Z Force Z Force Y1,4 CHANNEL (4,i) CoP X Moment X Force Y2,3 CHANNEL (5,i) CoP Y Moment Y Force Z1 CHANNEL (6,i) Free Moment Z Moment Z Force Z2 CHANNEL (7,i) n/a n/a Force Z3 CHANNEL (8,i) n/a n/a Force Z4 typical brand -- AMTI;BERTEC KISTLER ForceSensorChannels: channel numbers that correspond to the analog input file, SensMatrix: optional,only required for type 4 forcesensors PROCESS Generation of variable that represent a external force to the body OUTPUT GLOBAL: BODY.CONTEXT.ExternalForce : an element is added to the array % Copyright 2000-2006 This program is free software under the terms of the
CreateStylus	CREATESTYLUS [BodyMech 3.06.01]: declares a new stylus to BODY.CONTEXT INPUT StylusName : name of stylus StylusType : tracking sensor(0) or #markers(>=1) StylusToTipFunction : .m function StylusIndex : StylusType==1+: list of marker labels assigned to the Stylus : StylusType==0 : tracking sensor no. StylusData : PROCESS Generation of variable that represent a stylus, to be used in anatomical calibration OUTPUT GLOBAL: BODY.CONTEXT.Stylus % Copyright 2000-2006 This program is free software under the terms of the

BMdocs

Scope: all BodyMech documents , e.g this BodyMechGuide

BMkernel

files that are general to the approach of 3D human movement analysis.

AnatomyRefSegmentKinematics	ANATOMYREFSEGMENTKINEMATICS [BodyMech 3.06.01]: Calls an m-file with specific calculations INPUT PROCESS Calls an m-file with specific calculations OUTPUT % Copyright 2000-2006 This program is free software under the terms of the
AssignEmgDataToBody	ASSIGNEMGDATATOBODY [BodyMech 3.06.01]: assigns measured data in the BODY.MUSCLE-fields INPUT ForceThreshold (optional) in Newton GLOBAL : ANSIGNAL_DATA ANSIGNAL_TIME_GAIN ANSIGNAL_TIME_OFFSET BODY.MUSCLE(..).Emg.InputFileIndices PROCESS Reads the AnalogSignals into BODY.MUSCLE-fields OUTPUT GLOBAL : BODY.MUSCLE(..).Emg.Signal BODY.MUSCLE(iMuscleIndex).Emg.TimeGain BODY.MUSCLE(iMuscleIndex).Emg.TimeOffset % Copyright 2000-2006 This program is free software under the terms of the
AssignEmgEnvelopeDataToBody	ASSIGNEMGENVELOPEDATATOBODY [BodyMech 3.06.01]: Assigns measured data in the appropriate BODY fields INPUT ForceThreshold (optional) in Newton GLOBAL : ANSIGNAL_DATA ANSIGNAL_TIME_GAIN ANSIGNAL_TIME_OFFSET BODY.MUSCLE(..).Emg.InputFileIndices PROCESS Reads the AnalogSignals into BODY.MUSCLE-fields OUTPUT GLOBAL : BODY.MUSCLE(..).Emg.Envelope BODY.MUSCLE(iMuscleIndex).Emg.EnvelopeTimeGain BODY.MUSCLE(iMuscleIndex).Emg.EnvelopeTimeOffset % Copyright 2000-2006 This program is free software under the terms of the
AssignEmgRawAndEnvelopeToBody	ASSIGNEMGRAWANDENVELOPETOBODY [BodyMech 3.06.01]: assigns measured data in the BODY.MUSCLE-fields INPUT ForceThreshold (optional) in Newton GLOBAL : ANSIGNAL_DATA ANSIGNAL_TIME_GAIN ANSIGNAL_TIME_OFFSET BODY.MUSCLE(..).Emg.InputFileIndices PROCESS Reads the AnalogSignals into BODY.MUSCLE-fields OUTPUT GLOBAL : BODY.MUSCLE(..).Emg.Signal BODY.MUSCLE(iMuscleIndex).Emg.TimeGain BODY.MUSCLE(iMuscleIndex).Emg.TimeOffset % Copyright 2000-2006 This program is free software under the terms of the
AssignForceDataToBody	ASSIGNFORCEDATATOBODY [BodyMech 3.06.01]: Assigns measured data in the BODY.CONTEXT-fields INPUT ForceThreshold (optional) in Newton Global: ANSIGNAL_DATA ANSIGNAL_TIME_GAIN ANSIGNAL_TIME_OFFSET BODY.CONTEXT.ExternalForce(1).Type BODY.CONTEXT.ExternalForce(1).InputFileIndices

	BODY.CONTEXT.ExternalForce(1).ForceSensorToLab PROCESS breaks up external force data and assigns it according to current BODYdefinition Transforms all data to the laboratory frame OUTPUT Global: BODY.CONTEXT.ExternalForce(1).Signals BODY.CONTEXT.ExternalForce(1).TimeGain BODY.CONTEXT.ExternalForce(1).TimeOffset % Copyright 2000-2006 This program is free software under the terms of the
AssignMarkerDataToBody	ASSIGNMARKERDATATOBODY [BodyMech 3.06.01]: assigns measured data in the appropiate BODY-fields INPUT GLOBAL : MARKER_DATA MARKER_TIME_BASE PROCESS breaks up to data and assigns it according to current BODYdefinition Transforms all data to the laboratory frame OUTPUT GLOBAL : BODY.SEGMENT(..).Cluster.KinematicsMarkers BODY.SEGMENT(..).Cluster.RecordedMarkers BODY.SEGMENT(..).Cluster.AvailableMarkers % Copyright 2000-2006 This program is free software under the terms of the
AssignMarkerDataToStylus	ASSIGNMARKERDATATOSTYLUS [BodyMech 3.06.01]: assigns measured data the the current stylus INPUT GLOBAL : MARKER_DATA : MARKER_TIME_BASE PROCESS Breaks up to data and assigns it to the stylus Transforms all data to the laboratory frame OUTPUT GLOBAL : BODY.CONTEXT.Stylus.KinematicsMarkers % Copyright 2000-2006 This program is free software under the terms of the
BodyMech	BODYMECH [BodyMech 3.06.01]: Executes Human Movement Analysis Software BodyMech INPUT no input required PROCESS Creates Global parameters and initializes BodyMech Opens the Graphical User Interface of Bodymech OUTPUT GUI of BodyMech % Copyright 2000-2006 This program is free software under the terms of the
BodyMechFuncFooter	BODYMECHFUNCFOOTER [BodyMech 3.06.01]: calls at the end of each BodyMech Function INPUT no input PROCESS nil OUTPUT no output % Copyright 2000-2006 This program is free software under the terms of the
BodyMechFuncHeader	BODYMECHFUNCHEADER [BodyMech 3.06.01]: is called in each BodyMech Function to initialize global variables INPUT none PROCESS makes BodyMech global varables available in BodyMech Functions OUTPUT none

	% Copyright 2000-2006 This program is free software under the terms of the
CalculateAnatomicalKinematics	CALCULATEANATOMICALKINEMATICS [BodyMech 3.06.01]: Calls an m-file with specific calculations INPUT none PROCESS Loads and calls an m-file with specific anatomical calculations OUTPUT GLOBAL BODY.CONTEXT.AnatomicalCalculationFunction % Copyright 2000-2006 This program is free software under the terms of the
CalculateClusterKinematics	CALCULATECLUSTERKINEMATICS [BodyMech 3.06.01]: Estimation of the transformation matrix INPUT segment_no : segment number of actual BODY definition global: BODY.SEGMENT(segment_no).Cluster.MarkerCoordinates BODY.SEGMENT(segment_no).Cluster.KinematicsMarkers BODY.SEGMENT(segment_no).Cluster.AvailableMarkers PROCESS Application of rigid_body transformation for each time-instance only valid markers are selected OUTPUT GLOBAL BODY.SEGMENT(segment_no).Cluster.KinematicsPose % Copyright 2000-2006 This program is free software under the terms of the
CalculateDefaultStickFigure	CALCULATEDEFAULTSTICKFIGURE [BodyMech 3.06.01]: generates default stickfigure from anatomical landmarks INPUT Input : none PROCESS generates default stickdiagram from anatomical landmarks for visualisation OUTPUT GLOBAL BODY.SEGMENT(iSegm).StickFigure(iStickmarker).Kinematics % Copyright 2000-2006 This program is free software under the terms of the
CalculateJointKinematics	CALCULATEJOINTKINEMATICS [BodyMech 3.06.01]: calculates Euler-joint angles INPUT Method : string, either 'AnatomyBased' or 'ReferenceBased' PROCESS Calculates the rotationmatrix between two segments Decomposition of the matrix into sequentional rotation angles (Euler, Cardanic) OUTPUT Alfa, Beta & Gamma for each joint BODY.JOINT(jnt_id).AnatomyRefKinematics.RotationAngles BODY.JOINT(jnt_id).PostureRefKinematics.RotationAngles % Copyright 2000-2006 This program is free software under the terms of the
CalculatePostureRefKinematics	CALCULATEPOSTUREREFKINEMATICS [BodyMech 3.06.01]: calculates all segment kinematics INPUT GLOBAL :BODY.SEGMENT(..).Cluster.KinematicsPose, GLOBAL :BODY.SEGMENT(..).Cluster.PosturePose) PROCESS Calculates segment kinematics relative to a reference posture OUTPUT GLOBAL : BODY.SEGMENT(..).Cluster.PostureRefKinematicsPose % Copyright 2000-2006 This program is free software under the terms of the
CalculateVirtualMarkers	CALCULATEVIRTUALMARKERS [BodyMech 3.06.01]: generates virtual marker coordinates from anatomical markers INPUT Input : none

	PROCESS generates virtual marker coordinates from anatomical markers for visualisation OUTPUT GLOBAL BODY.SEGMENT(iSegm).AnatomicalLandmark(iAnatMarker).Kinematics BODY.SEGMENT(iSegm).AnatomicalLandmark(iAnatMarker).TimeGain BODY.SEGMENT(iSegm).AnatomicalLandmark(iAnatMarker).TimeOffset % Copyright 2000-2006 This program is free software under the terms of the
ClearAllButProjectModel	CLEARALLBUTPROJECTMODEL [BodyMech 3.06.01]: Clears (global) BODY content of all Session and Trial fields except PROJECTMODEL INPUT none PROCESS Clears all BODY Session & Trial fields: BODY content of PROJECTmodel remains OUTPUT GLOBAL BODY content for ProjectModel % Copyright 2000-2006 This program is free software under the terms of the
ClearBody	CLEARBODY [BodyMech 3.06.01]: Clears (global) BODY content of All ModelType Fields form BODY INPUT none PROCESS Clears all BODY fields contents OUTPUT GLOBAL BODY (empty fields) % Copyright 2000-2006 This program is free software under the terms of the
ClearBodyProjectModel	CLEARBODYPROJECTMODEL (BodyMech 3.06.01): Clears all (global) BODY content INPUT none PROCESS Clears all BODY fields contents OUTPUT GLOBAL BODY (empty fields) % Copyright 2000-2006 This program is free software under the terms of the
ClearKinematics	CLEARKINEMATICS [BodyMech 3.06.01]: Clear relevant fields in BODY.SEGMENT and BODY.JOINT INPUT scope : 'markers';'segments';'jonts' PROCESS clears kinematics fields in BODY.SEGMENT and BODY.JOINT OUTPUT GLOBAL: BODY.SEGMENT (all segments); BODY.JOINT (all segments) : kinematic fields cleared % Copyright 2000-2006 This program is free software under the terms of the
ClearSessionModel	CLEARSESSIONMODEL [BodyMech 3.06.01] (is identical to ClearAllButprojectmodel.m) Clears (global) BODY content of SessionModel and TrialFields from BODY INPUT PROCESS Clears all BODY Session fields: BODYcontent of PROJECTmodel remains OUTPUT % Copyright 2000-2006 This program is free software under the terms of the

ClearTrialModel	CLEARTRIALMODEL [BodyMech 3.06.01]: Clears (global) BODY content of TrialModel Fields from BODY INPUT PROCESS Clears all BODY Trial fields: BODYcontent of PROJECT and SESSION model remains OUTPUT % Copyright 2000-2006 This program is free software under the terms of the
CorrectOffsetForceSignals	CORRECTOFFSETFORCESIGNALS [BodyMech 3.06.01]: Correct for bias on all ForceSignals from BODY.CONTEXT.ExtrenalForce INPUT Forceindex = 1 (when only one FP is used) Global: BODY.CONTEXT.ExternalForce(1).Signals BODY.CONTEXT.ExternalForce(1).TimeGain BODY.CONTEXT.ExternalForce(1).TimeOffset BODY.CONTEXT.ExternalForce(1).Type PROCESS plots the external force signals and asks for indication of zero-force level (= and - 0.5 seconds will used to calculate bias) includes: correct for bias and elimination of too low forces (vertical only) OUTPUT Global: corrected: BODY.CONTEXT.ExternalForce(1).Signals % Copyright 2000-2006 This program is free software under the terms of the
DefineLocalClusterFrames	DEFINELOCALCLUSTERFRAMES [BodyMech 3.06.01]: Defines an adhoc cluster frame for each segment INPUT mode : either 'static' or 'dynamic' PROCESS 'static' uses LocalReferenceFrame for each BODY.SEGFMENT applies the first frame of ClusterKinematics OUTPUT GLOBAL: BODY.SEGMENT(i).Cluster.MarkerCoordinates BODY.SEGMENT(i).Cluster.PosturePose % Copyright 2000-2006 This program is free software under the terms of the
Dehomogenize	DEHOMOGENIZE [BodyMech 3.06.01]: Transformation to non homogeneous matrices INPUT Transformation matrix: size(HomogeneousMatrix)=[4,4,Nsamples] Homogeneous coordinates: size(HomogeneousMatrix)= [4,Nsamples] PROCESS Transformation matrix: splits into a translation vector and a rotation matrix Homogeneous coordinates: revomes the 4th (hogenoues) coordinate in th first dim. OUTPUT Transformation matrix: vector [3,Nsamples] and matrix [3,3,Nsamples] Homogeneous coordinates: vector [3,Nsamples] % Copyright 2000-2006 This program is free software under the terms of the
EditNewAnatCalcFunction	EDITNEWANATCALCFUNCTION [BodyMech 3.06.01]: opens a template Anatomical Calculation Function to edit INPUT Input : none PROCESS Loads an (template) anatomical calculation function (m.file) and opens Editor OUTPUT none % Copyright 2000-2006 This program is free software under the terms of the
EditNewProjectModel	EDITNEWPROJECTMODEL [BodyMech 3.06.01]: opens a template to create ProjectModel to edit INPUT Input : none

	PROCESS opens a template (m.file)to create ProjectModel and evokes the editor OUTPUT none % Copyright 2000-2006 This program is free software under the terms of the
Homogenize	HOMOGENIZE [BodyMech 3.06.01]: transformation to homogeneous coordinates INPUT if N_input_argument =1 : vector [3,Nsamples] if N_input_argument =2 : vector [3,Nsamples],matrix [3,3,Nsamples] PROCESS if N_input_argument =1 : adds ones as the 4th coordinate in the first dimension if N_input_argument =2 : combines translation and rotation into a single transformation (4X4) matrix OUTPUT vector [4,Nsamples} or matrix [4,4,Nsamples] % Copyright 2000-2006 This program is free software under the terms of the
HomogenizeCoordinates	HOMOGENIZECOORDINATES [BodyMech 3.06.01]: transformation to homogeneous coordinates INPUT coordinate_array DIM1=3 PROCESS adds ones as the 4th coordinate in the first dimension OUTPUT h_coordinate_array % Copyright 2000-2006 This program is free software under the terms of the
HomogenizeMatrix	HOMOGENIZEMATRIX [BodyMech 3.06.01]: transformation to homogeneous coordinates INPUT TranslationVector [3,Nsamples] RotationMatrix [3,3,Nsamples] PROCESS Combines rotation and translation into a single transformation (4X4) matrix OUTPUT Transformation_matrix [4,4,Nsamples] % Copyright 2000-2006 This program is free software under the terms of the
InitBodyMech	INITBODYMECH [BodyMech 3.06.01]: initializes BodyMech status of variables PROCESS Initialization of BodyMech global variables OUTPUT GLOBAL: MARKER_DATA, MARKER_TIME_GAIN, MARKER_TIME_OFFSET ANSIGNAL_DATA, ANSIGNAL_TIME_GAIN, ANSIGNAL_TIME_OFFSET X=1 Y=2 Z=3 U=4 VIZ.ORTHOMODE, VIZ.ORTHOHANDLE VIZ.DDDMODE, VIZ.DDDHANDLE BODYSTATUS % Copyright 2000-2006 This program is free software under the terms of the
InterpolateMarkerKinematics	INTERPOLATEMARKERKINEMATICS [BodyMech 3.06.01]: interpolates all marker-signalsarray INPUT GLOBAL: BODY.SEGMENT(..).Cluster.KinematicsMarkers GLOBAL: BODY.SEGMENT(..).Cluster.RecordedMarkers CriticalHole : warninglevel PROCESS Interpolates "holes" in the signal and smoothes the signal uses GCVSPL OUTPUT GLOBAL: BODY.SEGMENT(..).Cluster.KinematicsMarkers GLOBAL: BODY.SEGMENT(..).Cluster.AvailableMarkers % Copyright 2000-2006 This program is free software under the terms of the
LoadBodyFile	LOADBODYFILE [BodyMech 3.06.01]: Opens BodyFile (*.BMB) selected by user INPUT none

	PROCESS Opens BodyFile selected by user OUTPUT filename and ptahname of selected BMB file % Copyright 2000-2006 This program is free software under the terms of the
LocalReferenceFrame	LOCALREFERENCEFRAME [BodyMech 3.06.01]: creates an ad-hoc local coordinate system INPUT ClusterMarkers: matrix [3,M] of a rigid cluster of M marker positions in the GRF (global reference frame) Marker positions are column vectors: i.e comply to the [x;y;z] format with N=> 3 : at least three non-colinear markers are needed PROCESS calculates the coordinates of the markers in an orthonormal, righthanded local coordinate system origin: centroid of all N points marker1-x: direction of local Xaxis x,marker1,marker2: local XY-plane OUTPUT M = (non-homogeneus) markercoordinates in the local frame T = [R,t; 0 0 0 1] : the homogeneous transformation matrix R = [Ex Ey Ez]; Ez=Ex X Ey ; local coordinate vectors in global axis, i.e. the rotation matrix t = origin of the local coordinate system (in the GRF) % Copyright 2000-2006 This program is free software under the terms of the
ProbeAnatomy	PROBEANATOMY [BodyMech 3.06.01]: Calculates coordinates of bony landmarks in CRF INPUT Global: BODY.SEGMENT.Cluster-fields PROCESS Calculates coordinates of bony landmarks in cluster frame coordinates and a transformation matrix of the global to the local (cluster) frame. OUTPUT Global: BODY.SEGMENT(iSegment).AnatomicalLandmark.ClusterFrameCoordinates BODY.SEGMENT(iSegment).AnatomicalLandmark.ProbingPose BODY.CONTEXT.Stylus.TipPosition 17: % Copyright 2000-2006 This program is free software under the terms of the
RecordReferencePose	RECORDREFERENCEPOSE [BodyMech 3.06.01]: interactive selection from recoderded marker kinematics INPUT Global: BODY.SEGMENT(iSegment).Cluster-fields PROCESS Interactive selection from recoderded marker kinematics OUTPUT Global: BODY.SEGMENT(iSegment).Cluster.PosturePose % Copyright 2000-2006 This program is free software under the terms of the
RigidBodyTransformation	RIGIDBODYTRANSFORMATION [BodyMech 3.06.01]: calculates a transformation matrix INPUT x,y : matrices of a cluster of N markers on the rigid body in [x;y;z] format; N=>3 x : first position y : second position PROCESS Estimation of rotation and translation between x and y solving y = Rx + t in a least square optimal sense. OUTPUT R : rotation matrix t : translation vector e : error % Copyright 2000-2006 This program is free software under the terms of the
RotationMatrixToCardanicAngles	ROTATIONMATRIXTOCARDANICANGLES [BodyMech 3.06.01] : Rotation matrix to Cardan or Eulerian angles. INPUT

	R: BODY.JOINT(jnt_id).AnatomyRefKinematics.Pose DecompositionFormat: BODY.JOINT(jnt_id).AnatomyRefKinematics.DecompositionFormat PROCESS Extracts the Cardan (or Euler) angles from a rotation matrix. The parameters i, j, k specify the sequence of the rotation axes (their value must be the constant (X,Y or Z). j must be different from i and k, k could be equal to i. OUTPUT The two solutions are stored in the three-element vectors a and b ORIGINAL FUNCTION: RTOCARDA (Spacelib) (c) G.Legnani, C. Moiola 1998; adapted from: G.Legnani and R.Adamini 1993 % Copyright 2000-2006 This program is free software under the terms of the
RunBodyDefinition	RUNBODYDEFINITION [BodyMech 3.06.01]: INPUT Input : PROCESS OUTPUT % Copyright 2000-2006 This program is free software under the terms of the
RunProjectModel	RUNPROJECTMODEL [BodyMech 3.06.01]: opens and calls a function to create a New ProjectModel INPUT Input : none PROCESS Loads and calls an m.file for creation of a new ProjectModel OUTPUT GLOBAL BODY with projectModel content % Copyright 2000-2006 This program is free software under the terms of the
SaveAsBodyFile	SAVEASBODYFILE [BodyMech 3.06.01]: saves BODY model as *.bmb file INPUT filename, pathname PROCESS saves BODY model as *.BMB file OUTPUT GLOBAL BODY.HEADER.FileName % Copyright 2000-2006 This program is free software under the terms of the
SelectMarkerKinematics	SELECTMARKERKINEMATICS [BodyMech 3.06.01]: user interactive selection of measured marker kinematics INPUT Global: BODY.SEGMENT(i).Cluster.KinematicsMarkers PROCESS User selects either time-instance or time-interval OUTPUT Global: BODY.SEGMENT(iSegment).Cluster.KinematicsPose % Copyright 2000-2006 This program is free software under the terms of the
ShowSelectedMarkerKinematics	SHOWSELECTEDMARKERKINEMATICS [BodyMech 3.06.01]: user interactive selection of measured marker kinematics INPUT Global: BODY.SEGMENT(i).Cluster.KinematicsMarkers PROCESS User selects either time-instance or time-interval OUTPUT Global: BODY.SEGMENT(iSegment).Cluster.KinematicsPose % Copyright 2000-2006 This program is free software under the terms of the
Transform	TRANSFORM [BodyMech 3.06.01]: Transform uses a 4x4 TransformationMatrix on all elements of the Invector INPUT M : TransformationMatrix p : InVector PROCESS Transform uses a 4x4 TransformationMatrix on all elements of the Invector

```
    An OutVector (as well as its homogeneous value) is calculated.
  OUTPUT
    q : OutVector
    b : HomOutVector

% Copyright 2000-2006 This program is free software under the terms of
the
```

BMprojects

all files used for specific Projects. Default a subfolder DEMOproject is added. Users can add their own Project folder, which will contain any user written m-files and data that are bounded to a specific project. (see Chapter III: BodyMechFunctionLibrary for more help info for each function)

DEMOproject

Content of all relevant datafiles and functions for DEMO

YourProject

Scope: to be used for your own project data files and functions. Rename in unique folder name according to your project

BM templates

template script files of Projectmodels and anatomical calculation functions

AnatCalc_LowEx_4Segm_Left

```
  ANATCALC_LOWEX_4SEGM_LEFT [ BodyMech 3.06.01 ]: anatomical calculation file (t
  INPUT
     Input : none
  PROCESS
    Anatomical calculation of lower leg kinematics, based on CAMARC convention (
al. 1995).
    Calls function
     AnatomicalFrameDefinition with specific input
  OUTPUT
    GLOBAL  BODY.SEGMENT(iSegm).AnatomicalFrame.KinematicsPose
            BODY.SEGMENT(iSegm).StickFigure(iStick).Kinematics
            BODY.SEGMENT(iSegm).StickFigure(iStick).TimeOffset
            BODY.SEGMENT(iSegm).StickFigure(iStick).TimeGain


% Copyright 2000-2006 This program is free software under the terms of the
```

AnatCalc_LowEx_4Segm_Right

```
  ANATCALC_LOWEX_4SEGM_RIGHT [ BodyMech 3.06.01 ]: anatomical calculation file (
  INPUT
     Input : none
  PROCESS
    Anatomical calculation of lower leg kinematics, based on CAMARC convention (
al. 1995).
    Calls function
     AnatomicalFrameDefinition with specific input
  OUTPUT
    GLOBAL  BODY.SEGMENT(iSegm).AnatomicalFrame.KinematicsPose
            BODY.SEGMENT(iSegm).StickFigure(iStick).Kinematics
            BODY.SEGMENT(iSegm).StickFigure(iStick).TimeOffset
            BODY.SEGMENT(iSegm).StickFigure(iStick).TimeGain


% Copyright 2000-2006 This program is free software under the terms of the
```

AnatCalc_LowEx_5Segm_Left

```
  ANATCALC_LOWEX_5SEGM_LEFT [ BodyMech 3.06.01 ]: anatomical calculation file (t
  INPUT
     Input : none
  PROCESS
    Anatomical calculation of lower leg kinematics, based on CAMARC convention (
al. 1995).
    Calls function
     AnatomicalFrameDefinition with specific input
  OUTPUT
    GLOBAL  BODY.SEGMENT(iSegm).AnatomicalFrame.KinematicsPose
            BODY.SEGMENT(iSegm).StickFigure(iStick).Kinematics
```

	BODY.SEGMENT(iSegm).StickFigure(iStick).TimeOffset BODY.SEGMENT(iSegm).StickFigure(iStick).TimeGain % Copyright 2000-2006 This program is free software under the terms of the
AnatCalc_LowEx_5Segm_Right	ANATCALC_LOWEX_5SEGM_RIGHT [BodyMech 3.06.01]: anatomical calculation file (INPUT Input : none PROCESS Anatomical calculation of lower leg kinematics, based on CAMARC convention (al. 1995). Calls function AnatomicalFrameDefinition with specific input OUTPUT GLOBAL BODY.SEGMENT(iSegm).AnatomicalFrame.KinematicsPose BODY.SEGMENT(iSegm).StickFigure(iStick).Kinematics BODY.SEGMENT(iSegm).StickFigure(iStick).TimeOffset BODY.SEGMENT(iSegm).StickFigure(iStick).TimeGain % Copyright 2000-2006 This program is free software under the terms of the
AnatCalc_LowEx_7Segm	ANATCALC_LOWEX_7SEGM [BodyMech 3.06.01]: anatomical calculation file (templa INPUT Input : none PROCESS Anatomical calculation of lower leg kinematics, based on CAMARC convention (al. 1995). Calls function AnatomicalFrameDefinition with specific input OUTPUT GLOBAL BODY.SEGMENT(iSegm).AnatomicalFrame.KinematicsPose BODY.SEGMENT(iSegm).StickFigure(iStick).Kinematics BODY.SEGMENT(iSegm).StickFigure(iStick).TimeOffset BODY.SEGMENT(iSegm).StickFigure(iStick).TimeGain % Copyright 2000-2006 This program is free software under the terms of the
AnatCalc_LowEx_8Segm	ANATCALC_LOWEX_8SEGM [BodyMech 3.06.01]: anatomical calculation file (templa INPUT Input : none PROCESS Anatomical calculation of lower leg kinematics, based on CAMARC convention (al. 1995). Calls function AnatomicalFrameDefinition with specific input OUTPUT GLOBAL BODY.SEGMENT(iSegm).AnatomicalFrame.KinematicsPose BODY.SEGMENT(iSegm).StickFigure(iStick).Kinematics BODY.SEGMENT(iSegm).StickFigure(iStick).TimeOffset BODY.SEGMENT(iSegm).StickFigure(iStick).TimeGain % Copyright 2000-2006 This program is free software under the terms of the
AnatomicalFrameDefinition	ANATOMICALFRAMEDEFINITION [BodyMech 3.06.01]: calculates anatomical frames INPUT SegmentName : name of the segment: 'Pelvis';'Right_Thigh';'Right_Shank';'Rig ;'Left_Thigh';'Left_Shank';'Left_Foot'; 'Right_Hand';'Right_Forearm';'Right_Humerus';'Right_Scapula';'Trunk' AnatomicalMarkers : coordinates of anatomical landmarks : NB. segment name, number and order is pre-defined in yo AnatomicalCalculationFunction!!! PROCESS Generation of anatomical frames according to the Camarc protocol Capozzo et al. (1995) Wu et al. (2005) OUTPUT T=anatomical frame (hom. coordinates) % Copyright 2000-2006 This program is free software under the terms of the
ProjectModel_LowEx_4Segm_Left	PROJECTMODEL_LOWEX_4SEGM_LEFT [BodyMech 3.06.01]: Template to create New Pro for 4 segments: foot, lower leg, upper leg, pelvis INPUT Input : none PROCESS will run to create a BODY with ProjectModel content OUTPUT

	GLOBAL BODY with projectModel content % Copyright 2000-2006 This program is free software under the terms of the
ProjectModel_LowEx_4Segm_Right	PROJECTMODEL_LOWEX_4SEGM_RIGHT [BodyMech 3.06.01]: Template to create New Pr for 4 segments: foot, lower leg, upper leg and pelvis INPUT Input : none PROCESS will run to create a BODY with ProjectModel content OUTPUT GLOBAL BODY with projectModel content % Copyright 2000-2006 This program is free software under the terms of the
ProjectModel_LowEx_5Segm_Left	PROJECTMODEL_LOWEX_5SEGM_LEFT [BodyMech 3.06.01]: Template to create New Pro for 5 segments: foot, lower leg, upper leg, pelvis and trunk INPUT Input : none PROCESS will run to create a BODY with ProjectModel content OUTPUT GLOBAL BODY with projectModel content % Copyright 2000-2006 This program is free software under the terms of the
ProjectModel_LowEx_5Segm_Right	PROJECTMODEL_LOWEX_5SEGM_RIGHT [BodyMech 3.06.01]: Template to create New Pr for 5 segments: foot, lower leg, upper leg, pelvis and trunk INPUT Input : none PROCESS will run to create a BODY with ProjectModel content OUTPUT GLOBAL BODY with projectModel content % Copyright 2000-2006 This program is free software under the terms of the
ProjectModel_LowEx_7Segm	PROJECTMODEL_LOWEX_7SEGM [BodyMech 3.06.01]: Template to create New ProjectM for 7 segments: feet, lower legs, upper legs, pelvis INPUT Input : none PROCESS will run to create a BODY with ProjectModel content OUTPUT GLOBAL BODY with projectModel content % Copyright 2000-2006 This program is free software under the terms of the
ProjectModel_LowEx_8Segm	PROJECTMODEL_LOWEX_8SEGM [BodyMech 3.06.01]: Template to create New ProjectM for 8 segments: feet, lower legs, upper legs, pelvis and trunk INPUT Input : none PROCESS will run to create a BODY with ProjectModel content OUTPUT GLOBAL BODY with projectModel content % Copyright 2000-2006 This program is free software under the terms of the

BMtools

basic helpful functions

OverviewAllBodyFields	OVERVIEWALLBODYFIELDS [BodyMech 3.06.01]: list of all possible BODY fields % Copyright 2000-2006 This program is free software under the terms of the
ReadDar4BM	READDAR4BM : Function to read data from DAR files (which are DICOM structured files) INPUT DarFileName datapath PROCESS

	Function to read data from DAR files (which are DICOM structured files) especially for kinetic and electrophysiological data to be assigned in a "Sybar" environment uses internal functions: ReadDicomAttribute FindDicomTag ReadNextDicomTag OUTPUT FileData: structured array % Copyright 2000-2006 This program is free software under the terms of the
ccc	CCC PROCESS; clears whole worksapce, closes all graphics and clears screem
clearallbut	CLEARALLBUT: Clear all variables except these. CLEARALLBUT VAR1 removes all variables from the workspace except VAR1. CLEARALLBUT VAR1 VAR2 VAR3 ... CLEARALLBUT('VAR1','VAR2','VAR3',...) CLEARALLBUT({'VAR1','VAR2','VAR3',...}) do the same for various variable names. Copyright 2000-2006 This program is free software under the terms of the
loadMDF	LOADMDF : loads SYBAR data files (*.MDF) from disk INPUT Input : PROCESS OUTPUT % Copyright 2000-2006 This program is free software under the terms of the
loadNDF	LOADNDF : loads an 3D Optotrak File INPUT Input : FileExtension,WindowHeader PROCESS loads an 3D Optotrak File OUTPUT % Copyright 2000-2006 This program is free software under the terms of the
loadTMS	LOADTMS : loads PORTI data files (*.S??) from disk INPUT: PROCESS OUTPUT % Copyright 2000-2006 This program is free software under the terms of the
normalizetimebase	NORMALIZETIMEBASE : calculates a 0-100% timebase, ensemble-averages cyclic signals [Cycle,TimeGain]=NormalizeTimeBase(signal,trigind) INPUT : signal: any one-dimensional array trigind : array of indices, default: [1 length(signal)] should increase monotoneusly PROCESS: calculates new points based on a 0-100% time base by spline interpolation for each time interval OUTPUT: if length(trigind)=2: Cycle [101 1] if length(trigind)>2: Cycle [101 Ncycles+2] Ncyles=length(trigind)-1, Ncycles+1: mean signal per point, i.e. ensemble averaged Ncycles+2: stand.dev ensemble averaged points TimeGain: (average) amplification/reduction of time-axis (i.e. 100/(samples/cycle)) WARNING user should be aware of information loss in case of excessive downsampling % Copyright 2000-2006 This program is free software under the terms of the

readMDF	READMDF : reads a Sybar measurement file
	INPUT
	Input : data: numerical matrix
	header: string matrix
	get path and name of MDF file to be loaded
	PROCESS
	for keyboard-input
	datapath=input('geef het path van de data :','s');
	datafile=input('geef de datafile (8 chars) :','s');
	datafile=[datafile, '.mdf'];
	invoke the windows-file-browser
	[datafile,datapath] = uigetfile('*.mdf', 'open SYBAR file', 40, 40);
	OUTPUT
	
	% Copyright 2000-2006 This program is free software under the terms of the

readNDF	READNDF :reads NDI-optotrak data files (*.NDF) from disk INPUT: datafile, *.NDF format (Northern-digital Data File) see OPTOTRAK documentation for more information) including extention datapath: computers' pathname (ending on a \) PROCESS reads file strips header information reads measurementdata recodes missing markers to NaN OUTPUT data: 3D-matrix of measurement data (coordinates, markers, time) maesures in MILLIMETERS Sfrequency : sample frequency [1/sec] CollDate : Collection date [dd/mm/yy] CollTime : Collection time [hh:mm:ss] % Copyright 2000-2006 This program is free software under the terms of the
readTMS	READTMS : reads a file which holds a measurement from the TMS porti system dicrimates between: Poly5 format 9(ee TMS POLY 5 technical reference guide, page 179) The later TMS32 format, necessary to read large signals. INPUT name = name of the file (including path and extension) OUTPUT Signal [Nsamples, Nchannels] : matrix of measured data (TMS32: floatingpoint; poly5: integer) Sfreq : 1/intersample_interval (integer number) SignalName [Nchannels] : cell array of channel names % Copyright 2000-2006 This program is free software under the terms of the
removeartefact	REMOVEARTEFACT : reduces low frequency artefacts (high-pass filter) INPUT : Signal: any one dimensional array Fc : Low pass cut-off frequency (default: 10 Hz) Fs : sample freqency of Signal (default: 1000 Hz) PROCESS: 3rd order high pass butterworth filter OUTPUT: signal with reduced lowfreq.artefacts % Copyright 2000-2006 This program is free software under the terms of the
smooth	SMOOTH : smoothes a signal (lowpass filter) INPUT Input : Signal: any one dimensional array Fc : Low pass cut-off frequency Fs : sample freqency of Signal (default: 1000 Hz) symmetric: 0 = one pass filter (default) 1 = twopass (bidirectional) PROCESS 2nd order low pass butterworth filter (symmetric=0) or 4th order low pass butterworth filter, no phase lag (symmetric=1) OUTPUT the smoothed signal % Copyright 2000-2006 This program is free software under the terms of the
subsample	SUBSAMPLE : resamples an equidistant timeseries in case of downsampling, Nyquist' criterion is preserved INPUT

	Signal : Original timeseries SourceFrq : sample frequency of original timeseries DestFrq : target sample frequency Bandwith : Forced bandwith of the target signal relative to DestFrq (default=.3, i.e the Nyquist criterion (.5) at the safe side) PROCESS resamples an equidistant timeseries lowpass filtering will prevent undersampling OUTPUT SubSampledSignal : New time series, at 1/DestFrq time intervals % Copyright 2000-2006 This program is free software under the terms of the

BMvizfunctions

files that are related to all Graphics User Interface items

BodyMechControlWindow	BODYMECHCONTROLWINDOW [BodyMech 3.06.01]: creates main user i/o window INPUT Input : none PROCESS creates main BodyMech Graphical User Interface features OUTPUT BodyMech GUI % Copyright 2000-2006 This program is free software under the terms of the GNU General Public License (www.gnu.org) declaration
BodyMechMenuFcn	BODYMECHMENUFCN [BodyMech 3.06.01]: calling interface to common BodyMech functions from the menu system INPUT FunctionDescription: character string this string must be (case sensitive!) identical to the used callback string-variable in the menu system (via BodyMechControlWindow.m) PROCESS calls the appropiate BodyMech function OUTPUT % Copyright 2000-2006 This program is free software under the terms of the
GraphForceSignals	GRAPHFORCESIGNALS [BodyMech 3.06.01]: plots ForceSignals (only availbale when importing Dar Files) INPUT GLOBAL :BODY.CONTEXT.ExternalForce.Signals PROCESS plots GroundReactionForce-signals OUTPUT % Copyright 2000-2006 This program is free software under the terms of the
GraphJointAngles	GRAPHJOINTANGLES [BodyMech 3.06.01]: Graphs angular decomposition angles of joints INPUT Global: BODY.JOINT.PostureRefKinematics.RotationAngles BODY.JOINT.AnatomyRefKinematics.RotationAngles PROCESS Generation of joint-angle graphs OUTPUT % Copyright 2000-2006 This program is free software under the terms of the
GraphMarkers	GRAPHMARKERS [BodyMech 3.06.01]: Plots marker trajectories INPUT GLOBAL :BODY.SEGMENT(..).Cluster.KinematicsMarkers GLOBAL :BODY.SEGMENT(..).Cluster.RecordedMarkers USER: selection of segments PROCESS plots marker trajectories: a figure for each selected segment

	a plot-axis for each marker interpolated markers in red % Copyright 2000-2006 This program is free software under the terms of the
GraphNetMoments	GRAPHNETMOMENTS [BodyMech 3.06.01]: Graphs angular decomposition angles of joints INPUT Global: BODY.JOINT.NetMoment PROCESS Generation of moment-angle graphs OUTPUT % Copyright 2000-2006 This program is free software under the terms of the
GraphRawEmg	GRAPHRAWEMG [BodyMech 3.06.01]: plots Raw Emg Signals (only availbale when importing Dar Files) INPUT GLOBAL :BODY.MUSCLE(..).Emg.Signal USER: selection of muscles PROCESS plots SRE-signals: a plot-axis for each selected muscle OUTPUT % Copyright 2000-2006 This program is free software under the terms of the
GraphSRE	GRAPHSRE [BodyMech 3.06.01]: Plots SRE INPUT GLOBAL :BODY.MUSCLE(..).Emg.Envelope USER: selection of muscles PROCESS plots SRE-signals: a plot-axis for each selected muscle % Copyright 2000-2006 This program is free software under the terms of the
ListBodyHeader	LISTBODYHEADER [BodyMech 3.06.01]: lists the BODYheader to the screen INPUT global: BODY PROCESS creates char list of all header-info on the screen % Copyright 2000-2006 This program is free software under the terms of the
ListBodyModel	LISTBODYMODEL [BodyMech 3.06.01]: lists the BodyMech BODY model to the screen INPUT global: BODY PROCESS lists all current active segments, joints and muscles on the screen % Copyright 2000-2006 This program is free software under the terms of the
OrthoView	ORTHOVIEW [BodyMech 3.06.01]: Opens orthogonal display window INPUT lab_area: (xlow, xhigh, ylow, yhigh, zlow, zhigh] PROCESS Creates window with kinematics menu and orthogonal plane views % Copyright 2000-2006 This program is free software under the terms of the
ShowTip	SHOWTIP [BodyMech 3.06.01]: Shows tipposition in display INPUT Input : global BODY.CONTEXT.Stylus.TipPosition PROCESS Shows tipposition of anatomical prober in display % Copyright 2000-2006 This program is free software under the terms of the
TimeControl	TIMECONTROL [BodyMech 3.06.01]: invoked when GUI timecontrol is manipulated INPUT Input : relative or absolute time-instance from GUI-control PROCESS calculates time and calls updating functions

	% Copyright 2000-2006 This program is free software under the terms of the
UpdateBodyHeader	UPDATEBODYHEADER [BodyMech 3.06.01]: user i/o to a header to BODY INPUT Scope ['All';'Project';'Session';'Trial'] user i/o PROCESS Generation of variables OUTPUT GLOBAL: BODY.HEADER % Copyright 2000-2006 This program is free software under the terms of the
UpdateOrthoView	UPDATEORTHOVIEW [BodyMech 3.06.01]: Updates orthogonal display window INPUT Input : global VIZ.ORTHOMODE VIZ.ORTHOHANDLE BODY.SEGMENT(i).Cluster.KinematicsMarkers BODY.SEGMENT(i).Cluster.KinematicsPose BODY.SEGMENT(i).Cluster.PostureRefKinematicsPose BODY.SEGMENT(i).AnatomicalFrame.KinematicsPose BODY.SEGMENT(i).AnatomicalLandmark.Kinematics BODY.SEGMENT(i).StickFigure.Kinematics BODY.SEGMENT(1).Cluster.TimeGain BODY.SEGMENT(1).Cluster.TimeOffset BODY.CONTEXT.ExternalForce PROCESS Updates orthogonal display window dependent on slider time % Copyright 2000-2006 This program is free software under the terms of the
UpdateVizOrthoMode	UPDATEVIZORTHOMODE [BodyMech 3.06.01]: Updates orthogonal visualization modes INPUT Input :none PROCESS Updates orthogonal visualization modes OUTPUT GLOBAL VIZ.ORTHOMODE % Copyright 2000-2006 This program is free software under the terms of the
UpdateVizTime	UPDATEVIZTIME [BodyMech 3.06.01]: Updates orthogonal display window INPUT Input : global VIZ.ORTHOMODE VIZ.ORTHOHANDLE VizTime BODY.SEGMENT(i).Cluster.KinematicsMarkers BODY.SEGMENT(i).Cluster.KinematicsPose BODY.SEGMENT(i).Cluster.PostureRefKinematicsPose BODY.SEGMENT(i).AnatomicalFrame.KinematicsPose BODY.SEGMENT(i).AnatomicalLandmark.Kinematics BODY.SEGMENT(i).StickFigure.Kinematics BODY.SEGMENT(1).Cluster.TimeGain BODY.SEGMENT(1).Cluster.TimeOffset BODY.CONTEXT.ExternalForce PROCESS Updates orthogonal display window dependent on slider time % Copyright 2000-2006 This program is free software under the terms of the

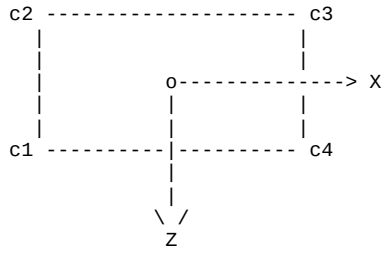
LabFunctions

all files that are specific or helpful to the laboratory environment; divided in general LABtools and specific folders (VUMC_LAB; Your_LAB)

LABtools

Basic handy tools and template functions to design yourown LAB specifications (see Chapter I and II for more details)

FixedLabCalibration	FIXEDLABCALIBRATION [BodyMech 3.06.01]: Calculates calibration of laboratory equipment setup INPUT FPcornersMAS=[c1_mas c2_mas c3_mas c4_mas]; (To be generated with eg: ProbeForcePlateCorners.m) default is the 1999 calibration of VUmc laboratory PROCESS calculates transformation based on factory calibrations OUTPUT motionanalysis_To_lab = transformation matrix forceplate_To_lab = transformation matrix % Copyright 2000-2006 This program is free software under the terms of the
ForceplateTransformation	FORCEPLATETRANSFORMATION [BodyMech 3.06.01]: Calculates calibration of laboratory equipment setup INPUT FPcornersMAS=[c1_mas c2_mas c3_mas c4_mas]; (To be generated with eg: ProbeForcePlateCorners.m) default is the 1999 calibration of VUmc laboratory PROCESS calculates transformation based on factory calibrations OUTPUT motionanalysis_To_lab = transformation matrix forceplate_To_lab = transformation matrix % Copyright 2000-2006 This program is free software under the terms of the
LoadProbing	LOADPROBING [BodyMech 3.06.01]: Import probing data from Optotrakfile (*.NDF) INPUT Data from probing file PROCESS Calculates StylusTip coordinates OUTPUT StylusTip % Copyright 2000-2006 This program is free software under the terms of the
ProbeForcePlateCorners	PROBEFORCEPLATECORNERS [BodyMech 3.06.01]: probing the forceplate corners INPUT Input : PROCESS OUTPUT % Copyright 2000-2006 This program is free software under the terms of the
ProbeRectangleCorners	PROBERECTANGLECORNERS [BodyMech 3.06.01]: load probing of the corners of a rectangle defining a standard lab frame relative to the motion analysis frame. See RectangleToLabFrame.m for details. % Copyright 2000-2006 This program is free software under the terms of the
RectangleToLabFrame	RECTANGLETOLABFRAME [BodyMech 3.06.01]: Calculates calibration of laboratory frame by equipment setup INPUT RectCornersMAS = [3 x 4] matrix Coordinates of the 4 corners of the rectangle - that is assumed to be level - given in the coordinate system of the Movement analysis system NB the sequence of the corners c1--c4 (see below) is critical Tip: Taking the corners of the forceplate is quite convenient PROCESS Calibratin of the motion analysis coordinate system towards a standard labsystem: ========== the geometrical centre of the rectangle is definied as the origin of the laboratory coordinate system adapting the ISB recommendations for standardization in the reporting of kinematic data, the Y axis points upward and parallel with the field of gravity which is perpendicular to the laboratory-floor, the X axis points in the direction of the walkway (assumed from

	left to right) Zaxis follows as a consequence (in a right handed coordinate system), TOP VIEW  OUTPUT ma2lab= (4x4) transformation matrix % Copyright 2000-2006 This program is free software under the terms of the

VUMC_LAB

NDprobe06117	NDPROBE06117 [BodyMech 3.06.01]: calculates the tip coordinates of the stylus INPUT Input: actual global stylus marker coordinates PROCESS uses: a stylus specific calculation OUTPUT StylusTip=global coordinates of the stylus tip % Copyright 2000-2006 This program is free software under the terms of the
NDprobe06138	NDPROBE06138 [BodyMech 3.06.01]: calculates the tip coordinates of the stylus INPUT InputData : actual global stylus marker coordinates PROCESS uses: a stylus specific calculation OUTPUT StylusTip=global coordinates of the stylus tip % Copyright 2000-2006 This program is free software under the terms of the
VUmcMASCalibration1999	VUMCMASCALIBRATION1999 [BodyMech 3.06.01]: provides forceplatecorner-coordinates used in 1999 at the LAB VUmc INPUT no variables the 1999 calibration of Forceplate in the VUmc laboratory is documented here PROCESS calculates corners OUTPUT ForcePlateCornersMAS = [3 x 4] matrix of the MAS coordinates of each corner % Copyright 2000-2006 This program is free software under the terms of the

Your_LAB

Scope: to be used for your own LAB specific calibration and import files anf function. Rename in unique folder name according to your

BodyMech
version 3.06.01

chapter IV. Theory and Background

Reconstruction of joint kinematics from 3D marker registration

---

version: May 2006

Author: J. Harlaar

Vumc, Amsterdam

Introduction

Kinematics is a qualitative description of human movement. For most purposes, the human body is modeled as a mechanical linked system of rigid bodies that move with a limited number of degrees of freedom. (*linked segment model*). This way of description is mainly suitable when movements have to be reduced to the level of anatomical structures.

This Chapter of the BodyMechGuide presents all technical and mathematical steps that are required to get from 3D marker registration to joint kinematics, which are implemented as algorithms in the software and data structure of BODYMECH in Matlab. All equations are referred to in the other Chapters of the BodyMechGuide

Several MatLab functions described here, were derived from the Faculty of Human Movement Sciences, VU University Amsterdam (DirkJan Veeger and Idsart Kingma) and from the software library Spacelib of Giovanni Legnani (http://bsing.ing.unibs.it/~glegnani/#spacelib)

Global set up

The sequence and steps to reconstruct joint kinematics from 3D marker positions are as follows:

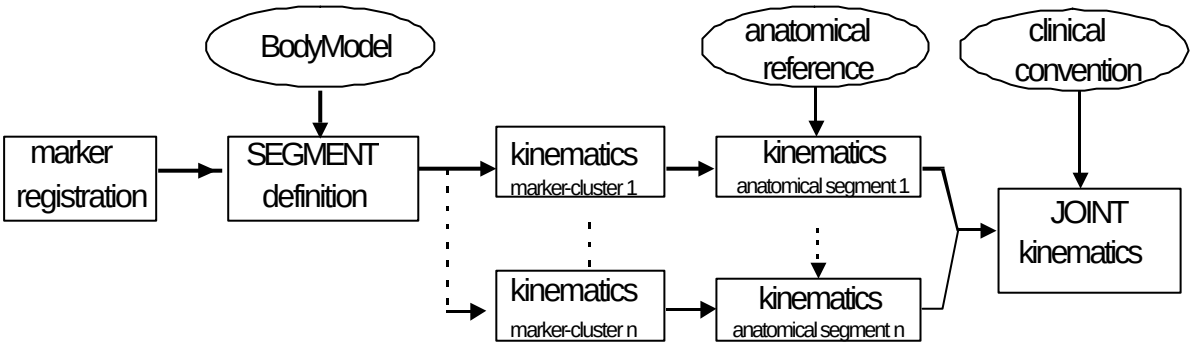


Figure 54. Schema of all theoretical steps to reconstruct joint kinematics from 3D marker positions.

Marker registration

3D positions of all markers recorded during the experiments. Each body segment is represented by a cluster of at least three markers.

Model definition of the body

The degrees of freedom of a joint is considered as the relative position and orientation of two adjacent body segments. Each body segment is judged as a rigid body and has to be defined dependent on each specific project or experiments. This is called a Body Model.
An example of a BodyModel is a template of 17 body segments , describing the whole body (Figure 2)

.

Cluster kinematics

At each body segment, a cluster of at least three markers is coupled. A coordinate system is assigned to this cluster (= CRF: Cluster Reference frame). For each recorded time frame, six degrees of freedom, position and orientation (= pose) of the CRF relative

to the glbal reference frame (GRF) can be calculated; i.e. three degrees of freedom of rotation (orientation) and three degrees of freedom of translation (position). This is described by a 4x4 transformation matrix (a rotation matrix eand a translation vector) per body segment per recorded time frame.

Body segment calibration

Body calibration can be obtained in several ways:

A. with a reference posture of the body: for example by recording the body in a standard (static) reference position; for example in the anatomical posture in which the anatomical axes ahve a known orientation relative to the GRF. In this way, axes of rotation and positions of the body segments can be defined. For each subsequent recording (of a movement), cluster kinematics is *compared* with this reference trial.

B. with an anatomical reference coordinate calibration. In this way, on each segment a defined anatomical coordinate system is appended (ARF: Anatomical Reference Frame). This ARF system is defined by at least three anatomical reference points, ususally by means of palpable bony landmarks. The relation between the CRF and the ARF is fixed with the assumption that the position of the clustermarkers do not shift relative to the identified anatomical landmarks. With these ARF per segment, the cluster kinematics can be *converted*.

C. some mixed forms from the two calibration methods above can be applied, e.g. when it is not possible to palpate the three anatomical points. Or with some additional *a priori* information and assumptions (e.g. an ideal spherical joint), the location of some anatomical locations can be determined.

Joint kinematics.

A joint is defined as the coupling between two adjacent body segments. Joint kinematics can be judged as the differences in orientation and position between two body segments as a function of time. One of the segments is used as reference segment (e.g. the proximal segment). This is unambiguously possible by means of three angles, which describe the orientation relative to the three orthogonal anatomical axes of the reference segment. However, usually, joint position is judged as a sequence of angle rotations. This is possible to decompensate the rotation matrices to a specific sequence. Clinical conventions play an important role in the choice of the decompensation sequence.

Assumptions

Assumption 1.
The human movement system is considered as a set of linked segments (rigid bodies).Type of links between these segments are the joints.

Assumption 2
The type of data, i.e. the 3D positions of the markers are available as time dependent signals, usually obtained with some type of 3D registration equipment.

Body Model

Body Segment Definition

A template of a 17 segment Body Model is shown in Figure 55.

1. pelvis
2. right_thigh
3. right_shank
4. right_foot
5. left_thigh
6. left_shank
7. left_foot
8. trunk
9. head
10. right_scapula
11. right_upper_arm
12. right_lower_arm
13. right_hand
14. left_scapula
15. left_upper_arm
16. left_lower_arm
17. left_hand

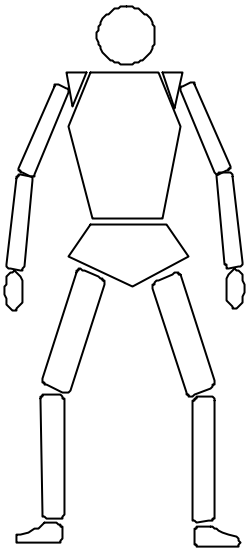


Figure 55. A template of a 17 segment Body Model

Each segment has the following properties which are related on different levels to the human body:

[1] **Identification:** the name of the segment; the indices of the markers attached
[2] **Calibration:** the rigid position of the markers during an experimental session
[3] **Kinematics** : the description of the movement during each trial.
[4] **Anthropometrics:** fixed information which is not relevant for the recontruction of the kinematics.

For each segment, a procedure has to be available, which links the anatomical coordinate system to the segment [e.g. see Capozzo, 1995, ISB standard protocols]

The assumption of a rigid body segment is difficult to apply, due to intern dynamics of movement and the wobbling mass of body soft tissue and also for largely defromable segments, such as the trunk, foot or hand. If necessary for a project, a more accurate model must be used for these types of segments or include more segments (fore- and midfoot; toes; fingers; more trunk segments)

Joint Definition

[1] **Identification:** the name of the joint and the adjacent segments
[2] **Calibration:** the definition of the anatomical coordinate system (dependent on chosen procedure)
[3] **Kinematics:** the description of the joint angles during each movement trial.

Marker kinematics

Marker kinematics are the experimental data i.e. the 3D marker coordinates of all markers as a function of time of some 3D registration system in a pre-defined format.

Cluster Reference Frame

To define a coordinate system to a cluster of markers (CRF: cluster reference frame), a calibration (static) recording is required in which all markers are valid at a same time frame in the GRF. See Figure 56.

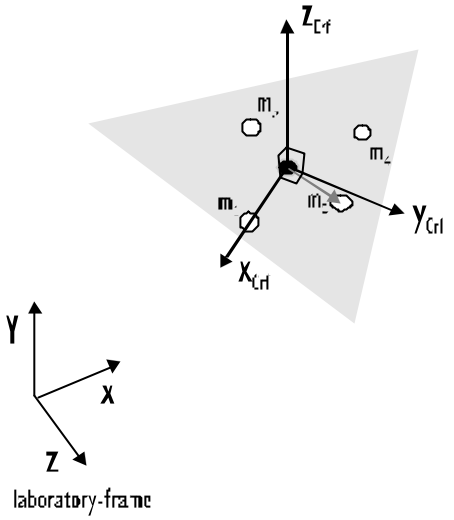


Figure 56. A rigid body (triangle) with a cluster of 4 markers (m1..m4) eand its corresponding CRF

First step is to determine an **origin**, a **plane** and **an axis** in this plane in order to construct an orthogonal reference frame to the cluster.
One of the possible solutions:
origin: the centroid of the marker coordinate is determined, which is used as the origin for the CRF:

$$\vec{t} = \frac{1}{n}\sum_{i=1}^{n}\vec{m}_i \qquad \{1\}$$

Use (arbitrary) two markers to define the X-as.(in Figure 56 ; through origin and marker 1) and a plane (XY; in Figure 56 throiugh origin, marker 1 and marker 2); this results in a coordinate system {eq. 2}. To divide these three orthogonal vectors by their length, an orthonormal (righthanded) coordinate system {eq. 2}.

$$\begin{aligned} \vec{x} &= (\vec{m}_1 - \vec{t}) & \vec{e}_x &= \vec{x}/|\vec{x}| \\ \vec{y} &= \vec{z} \times \vec{x} & \vec{e}_y &= \vec{y}/|\vec{y}| \\ z &= \vec{x} \times (\vec{m}_2 - \vec{t}) & \vec{e}_z &= \vec{z}/|\vec{z}| \end{aligned} \qquad \{2\}$$

The coordinates of the markers in this local system are obtained according to eq 3.: [1].

$$crf\ \vec{m}_i = |\ grf\ \vec{m}_i \cdot grf\ \vec{e}_x \quad grf\ \vec{m}_i \cdot \vec{e}_y \quad grf\ \vec{m}_i \cdot \vec{e}_z\ | \qquad \{3\}$$

[1] *remarks for notation: Matrices are indicated by CAPITALS; vectors with lower case and vector arrow; subscript left indicateds the coordinate system in which the corresponding vector of the column vectors of the matrix are given. Right subscripts indicates identificatione. Superscript right: possible arithmetrisch operation (eg square, inverse or transpose).*

By definition; this yields the column vectors of the rotation matrix of the local system to the global system {eq 4}.

$$ {}_{grf}R_{crf} = (\vec{e}_x \quad \vec{e}_y \quad \vec{e}_z) \qquad \{4\} $$

With vector $\vec{t}$ of the origin of the system {eq.1}, the relation between point m in the GRF and the same point in the CRF is given by equation 5:

$$ {}_{grf}\vec{m} = {}_{grf}R_{crf} * {}_{crf}\vec{m} + {}_{grf}\vec{t}_{crf} \qquad \{5\} $$

Point *m* has (normal) x,y,z coordinates in the local and global system. To describe the transformation , matrix multiplication with a rotation matrix and addition of a vector is required. Such a (non-homogeneousous) process can also be done in one step when using homogeneousous coordinates.

Homogeneous coordinates

Both the rotation matrix R and vector $\vec{t}$ of the origin of CRF define the transformation, i.e. the 4x4 transformation matrix of the system to transform a point in coordinates of the local reference system to coordinates in the global system. {eq. 6}:

$$ {}_{grf}T_{crf} = \left[\begin{array}{ccc|c} & {}_{grf}\boldsymbol{R}_{crf} & & {}_{grf}t_{crf} \\ \hline 0 & 0 & 0 & 1 \end{array}\right] = \left[\begin{array}{cccc} e_{x_x} & e_{y_x} & e_{z_x} & t_x \\ e_{x_y} & e_{y_y} & e_{z_y} & t_y \\ e_{x_z} & e_{y_z} & e_{z_z} & t_z \\ \hline 0 & 0 & 0 & 1 \end{array}\right] = \left[\begin{array}{cccc} x_x & y_x & z_x & t_x \\ x_y & y_y & z_y & t_y \\ x_z & y_z & z_z & t_z \\ \hline 0 & 0 & 0 & 1 \end{array}\right], \qquad \{6\} $$

With {eq. 6}, the pose (= position ánd orientation) of the local reference system (here: the CRF) relative to the global system is described. Using the *homogeneous* coordinates of point P are given by:

$P = \begin{vmatrix} x & y & z & w \end{vmatrix}^t$. The homogeneous coordinate w is zero voor infinite points and usually 1 for a point in space. This generalisation is clear from the last part of equation 6.

When point P (see Figure 57) has the (homogeneous) coordinates

$P_0 = \begin{vmatrix} x_0 & y_0 & z_0 & 1 \end{vmatrix}^t$ in the absolute reference frame (0) and in the local, body fixed reference frame (1), the coordinates $P_1 = \begin{vmatrix} x_1 & y_1 & z_1 & 1 \end{vmatrix}^t$, the (homogeneous) transformation is described by equation 7:

$$ P_0 = \mathbf{M}_{0,1} \cdot P_1 \qquad \{7\} $$

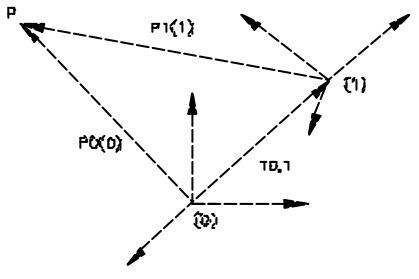


Figure 57 Illustration of homogeneous coordinates

With this, each point of a rigid body can be described in a reference system that is linked to the human body (CRF) . In other words: the pose of a rigid body (segment) is fully described by the transformation matrix. Section *'from joint rotation matices to joint angles'* will show how this information can be translated to orientation and position of a rigid body.

The determination the coordinates of the markers in the local system, is possible in two ways.
Equation 3 show the most direct way: the vector-inproduct of the coordinate vectors of the cluster markers relative to the origin of the local system (in the global system) with the determined unity vectors of the local system - in global coordinates- yields the local coordinates of the markers.
An other way is an inverse transformation on homogeneous global coordinates of the markers can be applied:

$$_{crf}\vec{m}_i = {}_{crf}T_{grf} \cdot {}_{grf}\vec{m}_i = {}_{grf}T_{crf}^{-1} \cdot {}_{grf}\vec{m}_i \quad \{8\}$$

Kinematics of the cluster

Section 1 demonstrated that the pose of the rigid body is determined by the transformation matrix with which the coordinates of each point on the rigid body can be calculated (see eq. 5). In practice, the problem is reversed: the laboratorium coordinates of (some of the) markers op the rigid body will be recorded during the movement (one set for each time frame). Subsequently, the transformation matrix T (which describes the kinematics of the rigid body) must be estiemated from the measured laboratorium coordinates ${}_{grf}m_i$ (with known local coordinates ${}_{crf}m_i$); equation 9:

$$\min_{T} \sum_{i=1}^{n} \left\| T_{GRF,CRF} \cdot m_{i,CRF} - m_{i,GRF} \right\|^2 \quad \{9\}$$

Idem, in non-homogeneous coordinates:

$$\min_{R,d} \sum_{i=1}^{n} \left\| R_{GRF,CRF} \cdot \vec{m}_{i,CRF} + \vec{d} - \vec{m}_{i,GRF} \right\|^2 \quad \{10\}$$

This problem is solved in the literatuur by Spoor and Veldpaus (1980) and adapted by Veldpaus ea. (1988). Later it is shown that the problem can alos be solved with the les complex *Singular Value Decomposition (SVD)* method (Arun ea. (1987), a technique to solve an overdetermined system and is superior in difficult conditioned marker geometries (e.g when markers ar almost aligned (Söderkvist and Wedin, 1993)).

The reference pose

For a reference pose, the transformation between the cluster reference frame *n* and the laboratorium frame determined by ${}_{GRF}T_{CRF_n}(ref)$. When at time frame *i* during recording, the transformation matrix will be determined by ${}_{grf}T_{crf_n}(i)$, the transformation relative to the reference position is (eq. 11):

$${}_{GRF}T_{CRF_n(i)} * {}_{CRF_n(ref)}T_{GRF} = {}_{GRF}T_{CRF_n(i)} * {}_{GRF}T_{CRF_n(ref)}^{-1} \quad \{11\}$$

Anatomical Reference Frame (ARF)

The use of a anatomical reference frame, means that a reference frame is used, that is linked to the anatomy; i.e. the body segment. The bony structurs in this segment will be used as starting point (pap[able and rigid). Figure 58 shows an ARF for the right thigh, linked to bony patrs of the femur. By recording the position of each bony point (landmark) simultaneously with the cluster markers on the segment, the position of the anatomical location can be expressed in the local coordinates of the cluster:

$$crf\ \vec{m}_{anat\ i} = grf\ T_{crf}^{-1} * grf\ \vec{m}_{anat\ i} \qquad \{12\}$$

In Figure 58, the anatomical points ME (mediale epicondyle) LE (laterale epicondyle) and TM (Trochanter major) are shown.. For the right thigh, the following ARF can be defined based on these three bony landmarks:

$$\vec{t} = 1/2 * (\vec{m}_{LE} + \vec{m}_{ME}) \qquad \{13\}$$

$$\begin{aligned} \vec{x} &= (\vec{m}_{TM} - \vec{t}) \times \vec{z} & \vec{e}_x &= \vec{x} / |\vec{x}| \\ \vec{y} &= \vec{z} \times \vec{x} & \vec{e}_y &= \vec{y} / |\vec{y}| \\ \vec{z} &= (\vec{m}_{LE} - \vec{t}) & \vec{e}_z &= \vec{z} / |\vec{z}| \end{aligned} \qquad \{14\}$$

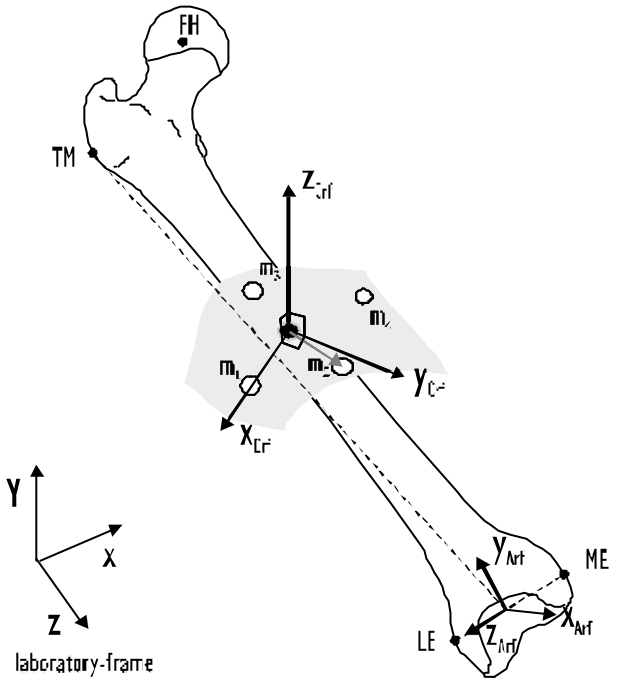


Figure 58 Illustration of the three reference frames: laboratory (GRF), cluster (CRF) and anatomical (ARF). CRF and ARF are indicated on the femur; anatomical bonty landmarks: medial epicondyle (ME), lateral epicondyle(LE), trochanter major (TM) and hip joint center (FH) are shown.

Comparable to the unity vectors and the translation vector of the cluster referencesystem, ($_{grf}T_{crf}$) (see equation 6), the three unity vectiors and the vector of the origin are defined by a transformation matrix: $_{crf}T_{anat}$.

The position of the body segment can be calculated for each time frame relative to GRF:

$$ {}_{grf}T_{anat}(i) = {}_{grf}T_{crf}(i) * {}_{crf}T_{anat} \quad \textbf{\{15\}} $$

Equation 15 expresses the cluster function as an intermediate in the process of calculation of the ARF. Also note the uniformity with {eq. 11}.

An alternative is to use the recordings of the anatomical points {eq.12} and the actual cluster transformation matrix, to determine the position of the anatomical points in the global system (as virtual markers). Further, the local system can be constructed (apply equations 13 and 14, but now in the GRF) which yields directly the transformation matrix (see eq. 6). Disadvantage is more calculation time; advantage; for each position the ARF can be constructed which allows *ad hoc* adaptations.

Specific anatomical coordinate systems

Conventions for anatomical coordinate systems (examples in Figure 59) can be fiund e.g. in Capozzo et al. (1995) and the ISB standards.

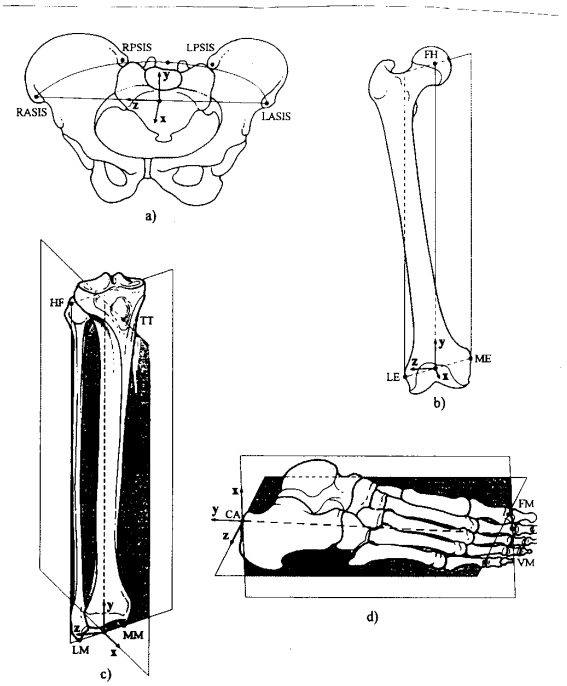


Figure 59. Example of specific anatomical coordinate systems (from Capozzo et al., 1995)

From segments to joints

A joint is defined as the link between two adjacent segments, e.g. a proximal and a distal segment. When the position of both segments in the GRF is known, the position of the distal segment relative to the proximal segment follows from :

$$
\begin{aligned}
{}_{prox}T_{dist}(i) &= {}_{prox}T_{grf}(i) * {}_{grf}T_{dist}(i) \\
&\Leftrightarrow \\
{}_{prox}T_{dist}(i) &= {}_{grf}T^{-1}_{prox}(i) * {}_{grf}T_{dist}(i)
\end{aligned}
\qquad \{16\}
$$

The rotation part of the transformation matrix yields the interrelated orientation of the two segmenten, and with that the joint.

From joint rotatiematrix to joint angles

The orientation or degrees of freedom can be expressed by three joint angles; most common used is the definition of sequence of joint rotations. After the first rotation, the next axis can be chosen in the new coordinate system in the adapted sequence. Many choices are possible, but hese are mainly driven by clinical conventions. For most joints, the standaardisation, proposed by Grood and Suntay (1983) is the most accepted (e.g. see Figure 60).

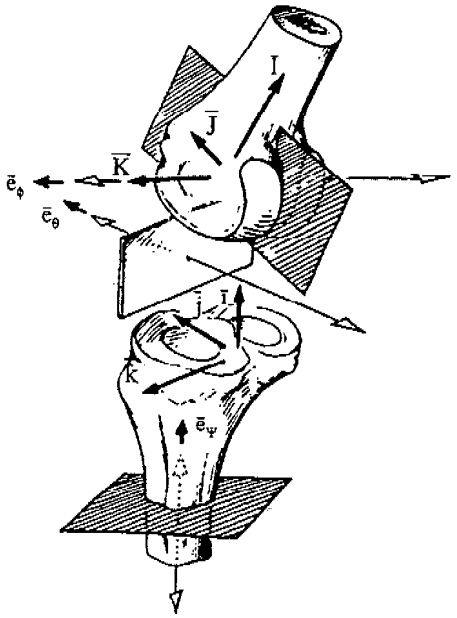


Figure 60 Illustration of the standaardised sequenace , according to Grood and Suntay (1983)

With a standardized decomposition of 3D-oriententatie in a sequence of plenary angles, the appearance of singular points, or "gimbal lock", is possible. For example at the shoulder, which requires a special treatment.

When the standard ISB convention of coordinate systems are adopted, the 'normal' Grood and Suntay decomposition will be ZY X.

Other conventions are possible, e.g. according to Legnani (SpaceLib; : http://bsing.ing.unibs.it/~glegnani/#spacelib)

There are several types of coordinates which can express the relative angular position of two moving bodies in 3D space; generally two frames are fixed on the two bodies.

In the "*neutral*" position, these frames are parallel to each other. One of the two reference frames is called *absolute*, while the other *moving*. Their relative angular position is expressed by the position of the moving frame with respect to the fixed one. The relative orientation of two frames can be imagined as obtained from the neutral position by three subsequent rotations α, β, γ of the moving frame around three axes: i, j, k. Rotations are generally performed about the axes of the fixed or of the moving frame.

It is possible to show that a group of three rotations α, β, γ around axes i, j, k of the fixed frame are equivalent to a sequence of rotations γ, β, α, around axes k, j, i (reverse order) of the moving frame.

Value of I,j,k for rotations around axes of fixed frame	***Cardan (Tait-Brian) systems*** *i ¹ j ¹ k ¹ i*	***Euler systems*** *i = k ¹ j*
Cyclic	x, y, z y, z, x z, x, y — Cardan angles	x, y, x y, z, y z, x, z — Euler angles
Anti-cyclic	z, y, x x, z, y y, x, z — Nautic angles	z, y, z x, z, x y, x, y

With the rotation axes are given, the angular position can be expressed by the three rotation angle values. Rotations about fixed axes multiply to left, while about moving axes to right.

References

Arun, K.S., Huang, T.S. and Blodstein, S.D., (1987). Least-squares fitting of two 3-d point sets. IEEE Trans. Pattern Anal. Machine Intell. 9, pp. 698–700. IEEE Trans. Pattern Anal. Machine Intell, 9, 698-700

Cappozzo et al. (1995). Position and orientation in space of bones during movement: anatomical frame definition and determination. Clinical Biomechanics, 10, 171-178 Spacelib: http://bsing.ing.unibs.it/~glegnani/#spacelib

Grood ES, Suntay WJ. (1983) A joint coordinate system for the clinical description of three-dimensional motions: application to the knee. J Biomech Eng. 1983 May;105(2):136-44.

Legnani G. software package SpaceLib: http://bsing.ing.unibs.it/~glegnani/#spacelib

Soderkvist I, Wedin PA. (1993) Determining the movements of the skeleton using well-configured markers. J Biomech. 1993 Dec;26(12):1473-7.

Spoor CW, Veldpaus FE. (1980) Rigid body motion calculated from spatial co-ordinates of markers. J Biomech. 1980;13(4):391-3.

Veldpaus FE, Woltring HJ, Dortmans LJ. (1988) A least-squares algorithm for the equiform transformation from spatial marker co-ordinates. J Biomech. 1988;21(1):45-54.

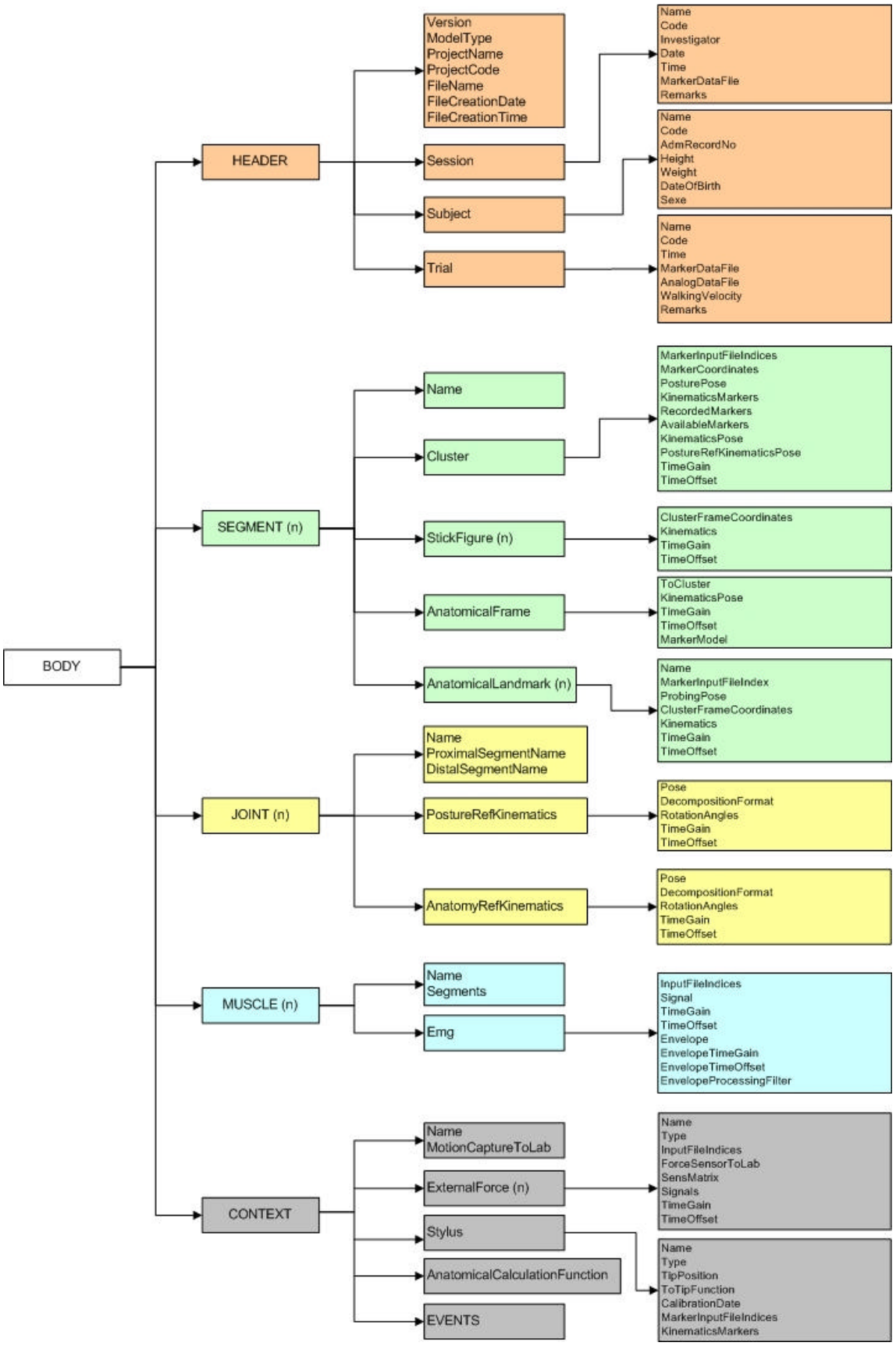
Wu G, Siegler S, Allard P, Kirtley C, Leardini A, Rosenbaum D, Whittle M, D'Lima DD, Cristofolini L, Witte H, Schmid O, Stokes I (2002) Standardization and Terminology Committee of the International Society of Biomechanics. ISB recommendation on definitions of joint coordinate system of various joints for the reporting of human joint motion--part I: ankle, hip, and spine. International Society of Biomechanics. J Biomech. 2002 Apr;35(4):543-8.

Wu G, van der Helm FC, Veeger HE, Makhsous M, Van Roy P, Anglin C, Nagels J, Karduna AR, McQuade K, Wang X, Werner FW, Buchholz B (2005) International Society of Biomechanics. ISB recommendation on definitions of joint coordinate systems of various joints for the reporting of human joint motion--Part II: shoulder, elbow, wrist and hand. J Biomech. 2005 May;38(5):981-992. Review.

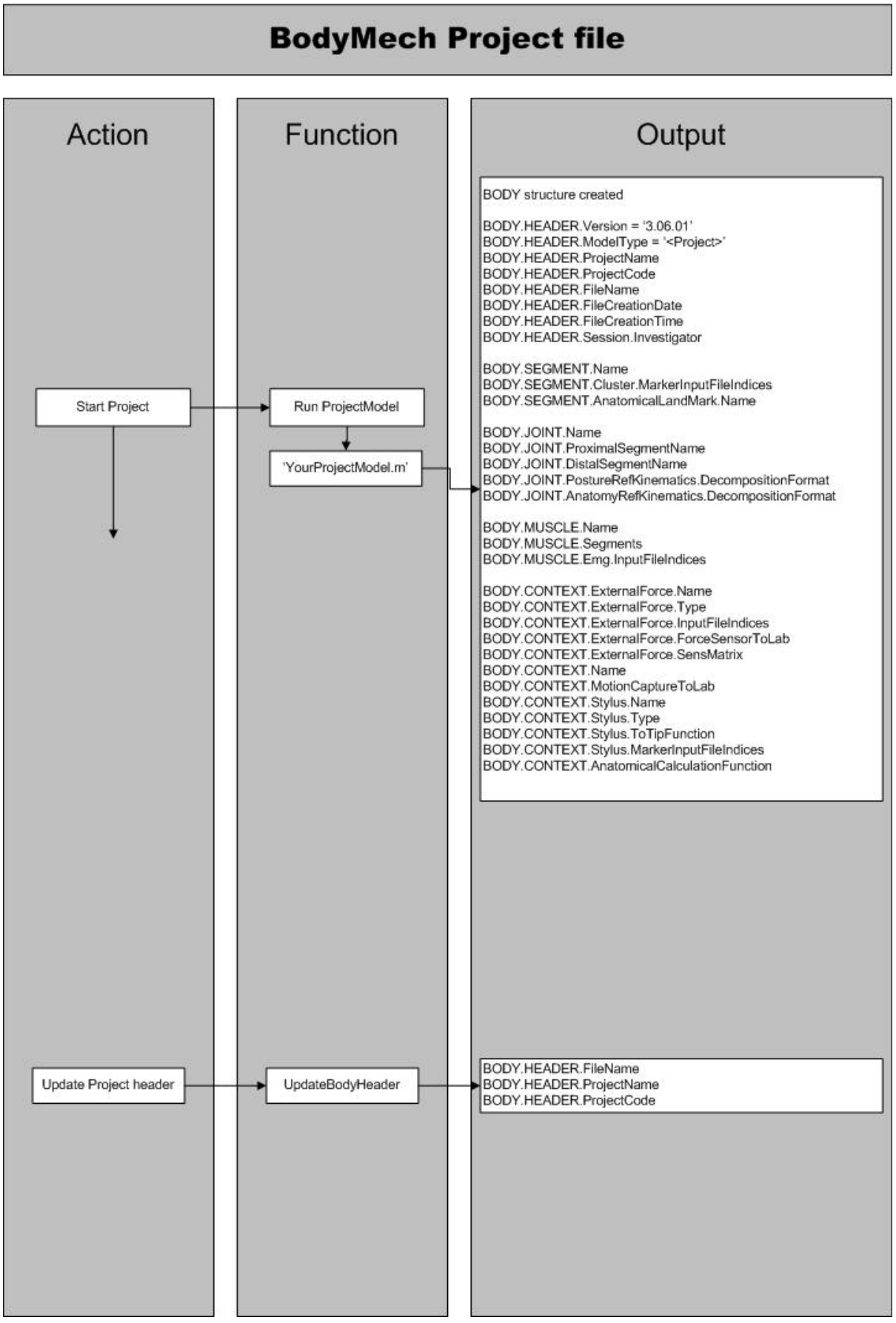
BodyMech
version 3.06.01

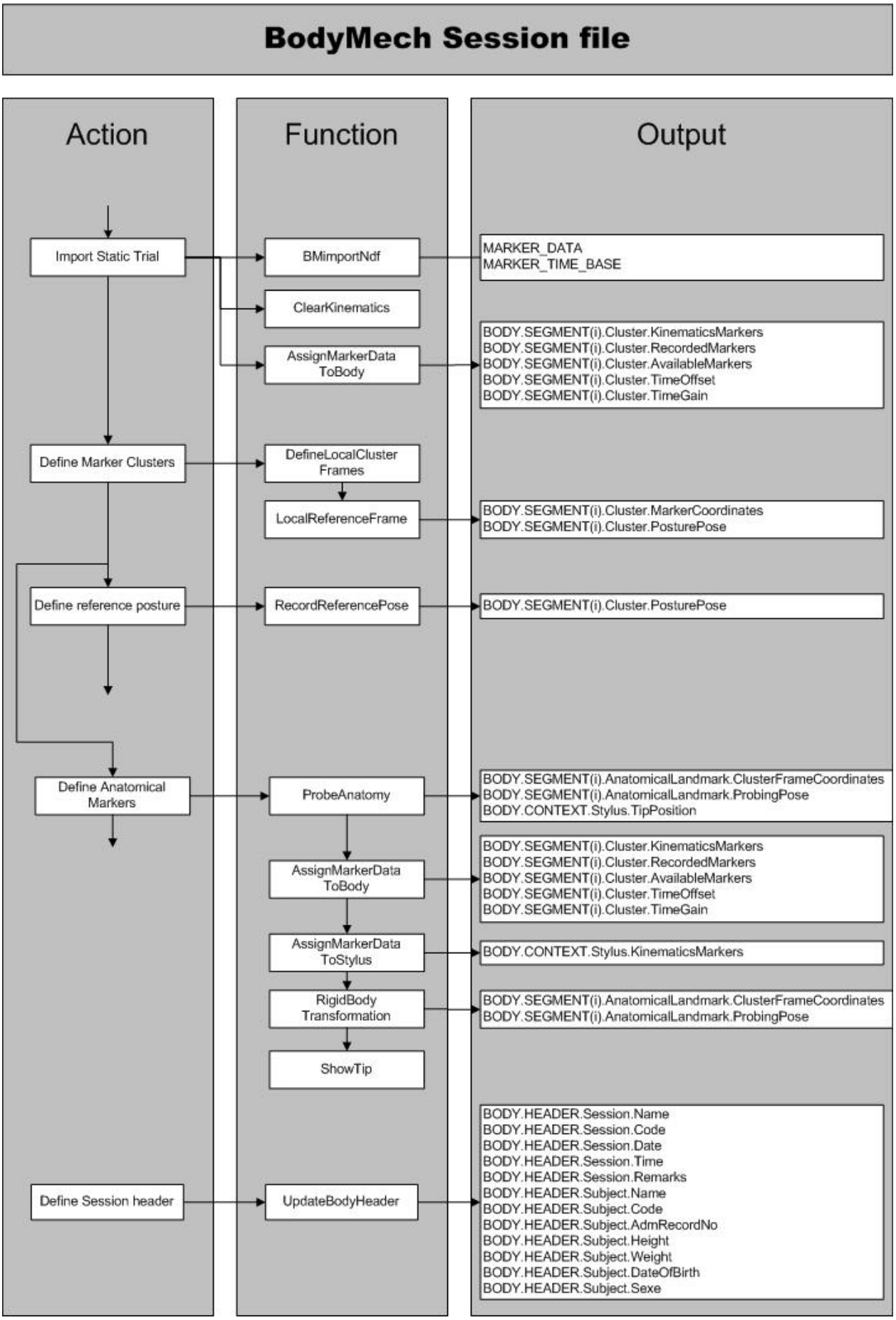
Appendices

Appendix A Body Structure

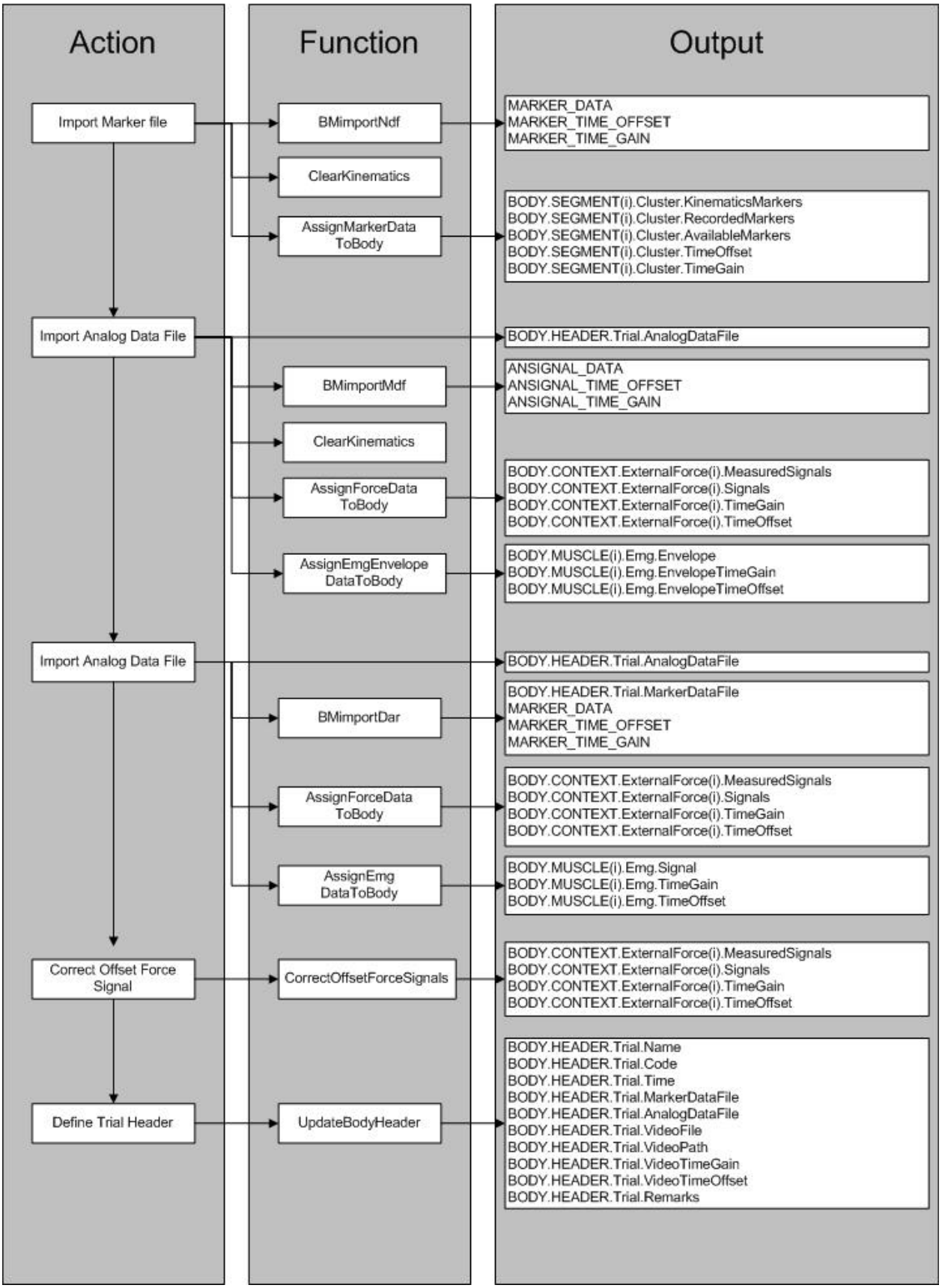


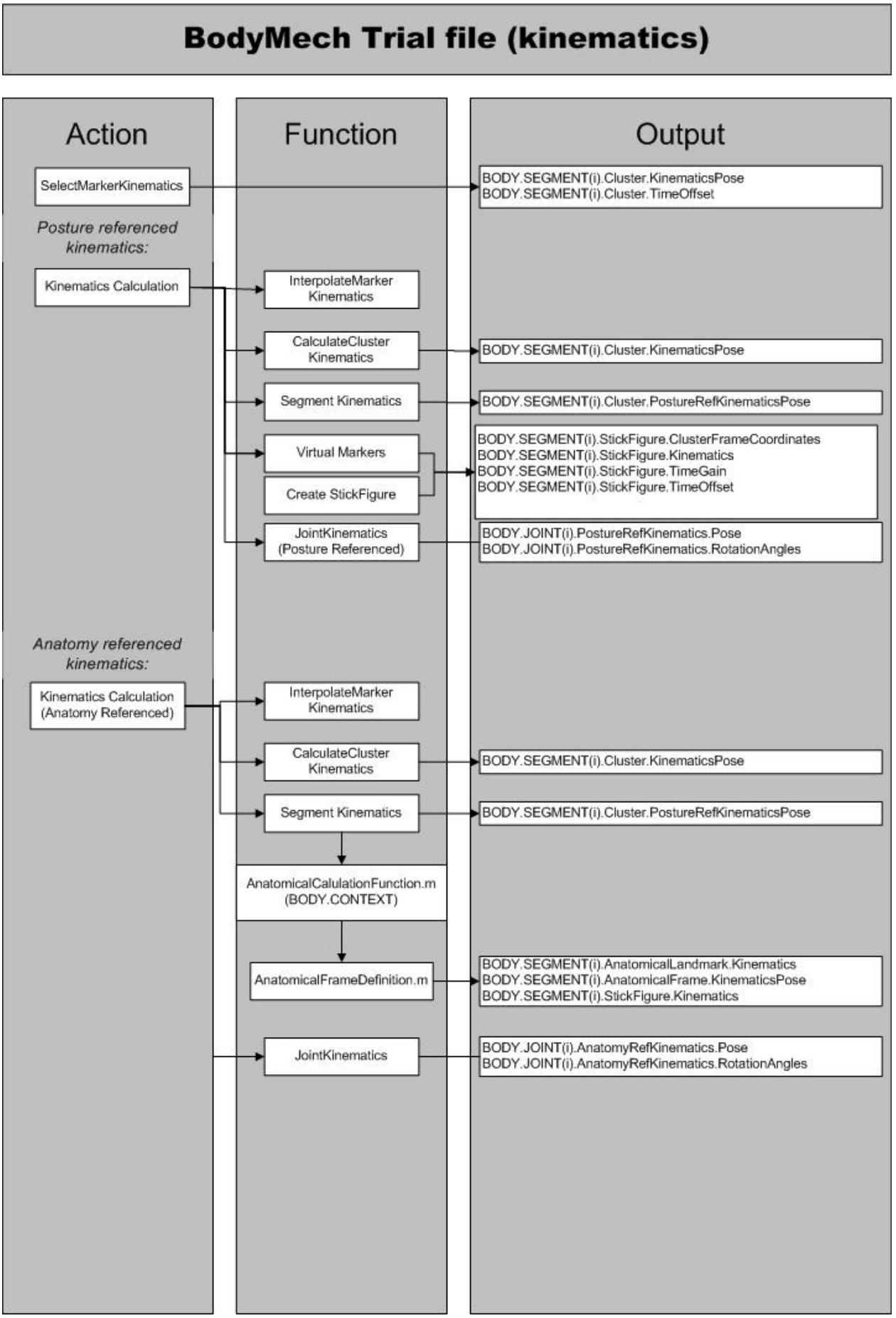
Appendix B BodyMech functions




BodyMech Session file
Action
Function
Output
Import Static Trial
BMimportNdf
MARKER_DATA
MARKER_TIME_BASE
ClearKinematics
AssignMarkerData ToBody
BODY.SEGMENT(i).Cluster.KinematicsMarkers
BODY.SEGMENT(i).Cluster.RecordedMarkers
BODY.SEGMENT(i).Cluster.AvailableMarkers
BODY.SEGMENT(i).Cluster.TimeOffset
BODY.SEGMENT(i).Cluster.TimeGain
Define Marker Clusters
DefineLocalCluster Frames
LocalReferenceFrame
BODY.SEGMENT(i).Cluster.MarkerCoordinates
BODY.SEGMENT(i).Cluster.PosturePose
Define reference posture
RecordReferencePose
BODY.SEGMENT(i).Cluster.PosturePose
Define Anatomical Markers
ProbeAnatomy
BODY.SEGMENT(i).AnatomicalLandmark.ClusterFrameCoordinates
BODY.SEGMENT(i).AnatomicalLandmark.ProbingPose
BODY.CONTEXT.Stylus.TipPosition
AssignMarkerData ToBody
BODY.SEGMENT(i).Cluster.KinematicsMarkers
BODY.SEGMENT(i).Cluster.RecordedMarkers
BODY.SEGMENT(i).Cluster.AvailableMarkers
BODY.SEGMENT(i).Cluster.TimeOffset
BODY.SEGMENT(i).Cluster.TimeGain
AssignMarkerData ToStylus
BODY.CONTEXT.Stylus.KinematicsMarkers
RigidBody Transformation
BODY.SEGMENT(i).AnatomicalLandmark.ClusterFrameCoordinates
BODY.SEGMENT(i).AnatomicalLandmark.ProbingPose
ShowTip
Define Session header
UpdateBodyHeader
BODY.HEADER.Session.Name
BODY.HEADER.Session.Code
BODY.HEADER.Session.Date
BODY.HEADER.Session.Time
BODY.HEADER.Session.Remarks
BODY.HEADER.Subject.Name
BODY.HEADER.Subject.Code
BODY.HEADER.Subject.AdmRecordNo
BODY.HEADER.Subject.Height
BODY.HEADER.Subject.Weight
BODY.HEADER.Subject.DateOfBirth
BODY.HEADER.Subject.Sexe

BodyMech Trial file (data import)




BodyMech Trial file (kinematics)
Action
Function
Output
SelectMarkerKinematics
BODY.SEGMENT(i).Cluster.KinematicsPose
BODY.SEGMENT(i).Cluster.TimeOffset
Posture referenced kinematics:
Kinematics Calculation
InterpolateMarker Kinematics
CalculateCluster Kinematics
BODY.SEGMENT(i).Cluster.KinematicsPose
Segment Kinematics
BODY.SEGMENT(i).Cluster.PostureRefKinematicsPose
Virtual Markers
Create StickFigure
BODY.SEGMENT(i).StickFigure.ClusterFrameCoordinates
BODY.SEGMENT(i).StickFigure.Kinematics
BODY.SEGMENT(i).StickFigure.TimeGain
BODY.SEGMENT(i).StickFigure.TimeOffset
JointKinematics (Posture Referenced)
BODY.JOINT(i).PostureRefKinematics.Pose
BODY.JOINT(i).PostureRefKinematics.RotationAngles
Anatomy referenced kinematics:
Kinematics Calculation (Anatomy Referenced)
InterpolateMarker Kinematics
CalculateCluster Kinematics
BODY.SEGMENT(i).Cluster.KinematicsPose
Segment Kinematics
BODY.SEGMENT(i).Cluster.PostureRefKinematicsPose
AnatomicalCalulationFunction.m (BODY.CONTEXT)
AnatomicalFrameDefinition.m
BODY.SEGMENT(i).AnatomicalLandmark.Kinematics
BODY.SEGMENT(i).AnatomicalFrame.KinematicsPose
BODY.SEGMENT(i).StickFigure.Kinematics
JointKinematics
BODY.JOINT(i).AnatomyRefKinematics.Pose
BODY.JOINT(i).AnatomyRefKinematics.RotationAngles