# simVITRO Data File Contents for Open Knee

# User Manual

# 2013CB-031-002.B

## Table of Contents



# 1. Revision History

| Revision | Date | Name | Comment |
|---|---|---|---|
| A | 5-5-15 | R. Colbrunn | Created |
| B | 5-6-15 | R. Colbrunn | Section 8. Added notes about optimized femur coordinate systems |
| | | | |

# 2. Purpose

This document was created for the CoBi Core Open Knee testing. It explains the contents of two key data files for this specific project. These data files are only the main and main_processed files. Explanations of any additional simVITRO data or configuration files are not included in this document.

# 3. Experiment Run Data File Overview

There are two different files included in this document (main & main_processed). They are both binary data files in the National Instruments Technical Data Management Streaming (TDMS) file format. National Instruments introduced the TDMS file format as a result of the deficiencies of other data storage options commonly used in test and measurement applications. The binary TDMS file format is an easily exchangeable, inherently structured, high-speed-streaming-capable file format that becomes quickly searchable without the need for complicated and expensive database design, architecture, or maintenance. TDMS files can be read into other software packages using many freely available tools. Converters for TDMS file exist for Python, Matlab, OpenOffice calc, and Excel, but also can be written for any development environment with dll compatibility. TDMS files are organized by groups and channels. The file is a two level hierarchy. At the first level the file is organized into groups. Then, under each group is a variable number of channels for that group. Each group and channel have properties that can identify the channel name, units, absolute time at initial sample, dt between samples, etc.

## 4. Main Data File

*Filename: Experiment Run Number_Experiment Run Name* + "_Main"
*File extension:* ".tdms"
*File Format:* binary
*File Type:* NI TDMS

The main data file stores all the main data for a given experiment run. Data is organized into groups depending on the transducer, type, state, or source of the data. It contains all the data acquired by the DAQ boards as well as any data displayed to the user during the execution of the motion. Some data may have originated from an analog sensor measured with the A/D board, other data may be from a digitized sensor which gets its information over ethernet, from a dll, or other similar digitized source. The main file is considered the raw data file.

Not all data in the main file is time aligned with other data channels in the file. Not in initial time nor in dt between samples. More specifically, the initial data point for a given group/channel did not necessarily get sampled at the same time as a channel from a different group. Within a group, however, it is safe to assume that the initial time points were from the same absolute time. The only caveat with this assumption is that there may be small multiplexing delays (µs) when sampling on an A/D board. With analog signals the dt between samples will be consistent across all channels and is based on the simVITRO analog sample rate parameter. When sampling data from digitized sensors, the initial time and dt is less deterministic compared to the analog data sampling. There is temporal uncertainty from when the data is actually captured by the sensor and then communicated to, and read by, simVITRO. This uncertainty is not always knowable and can vary depending on which type of digitized sensor is being used. Also, the dt between samples of the digitized sensor data is variable and is a function of when simVITRO is able to read the data from the sensor. For this reason, the dt between when the digitized sensors are read each time by simVITRO is itself a stored data channel in the data file. This variable dt channel is used in the data processing step to resample and time align everything.

The main file, being the raw file, contains data from the sensors as well as the data from the states (relative positions and loads) that were calculated based on the sensor values. The amplitude of these channels are in engineering units and the TDMS contains the units for each group/channel so there is no ambiguity on the meaning of the amplitude. If the data is filtered during the acquisition process (per the simVITRO low-pass filter settings) then the data in the main file will be filtered.

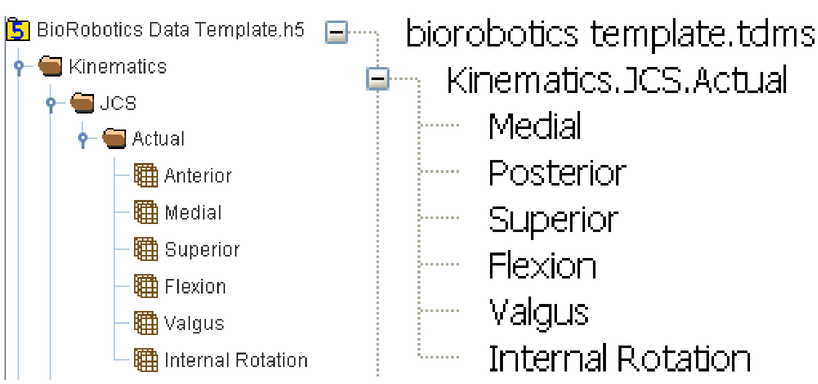
## 5. Main Processed Data File

*Filename: Experiment Run Number_Experiment Run Name* + "_Main" + "_Processed"
*File extension:* ".tdms"
*File Format:* binary
*File Type:* NI TDMS

The main_processed file stores all the processed main data for a given experiment run. Data is organized into groups depending on the transducer, type, state, or source of the data. This file is considered the processed data file. It contains all the post-process low-pass filtered (per the simVITRO settings) and converted versions of the pertinent data from the run. In the time domain, this file starts from a point in time where the desired trajectory begins execution. The file ends at the end of the desired trajectory. All the data in between those points have been time aligned and resampled, if necessary, to achieve the same initial time, dt, and final time for all groups and channels of data. The main_processed file is the one that should be used for any post processing of simVITRO data. The names of the groups/channels in the data files can vary depending on how simVITRO is configured. Presented in section 8 are those specific to the Open Knee project and any considerations with using them.

## 6. Data File Hierarchy

For the amount and type of data being created the TDMS format is limited in its ability to create and organize data in a hierarchical format. It only has two levels, group and channel. However, simVITRO data can be thought of as a deeper hierarchy and the file is structured so that other post processing programs for analyzing the data can convert the simVITRO data to a deeper format if desired. To overcome the TDMS hierarchical limitation the data is organized via the following data structure. ***Category.DataSource.DataState.Channels*** and the TDMS files will just have a large group name encompassing ***Category.DataSource.DataState*** and use "." in between all the various subgroups. The figures below show the comparison of the TDMS and deeper hierarchy techniques.




## 7. Data File Channel Attributes

Within the TDMS file each channel has properties (attributes) that allow the user to know how to properly interpret the temporal and magnitude information. Custom attributes can be assigned to waveform data, but the following list contains the standard set of attributes that should be in each channel. For more details see http://zone.ni.com/reference/en-XX/help/371361J-01/glang/tdms_set_properties/

| **Property Name** | **Data Type** | **Description** |
|---|---|---|
| name | String | Specifies the object name, such as the root name, group name, or channel name, in a .tdms file. |
| NI_ChannelLength | 64-bitunsigned integer numeric | Represents the number of values in a channel of a.tdms file. |
| NI_DataType | 16-bit unsigned integer numeric | Represents the data type of a channel. The value is an integer that corresponds to a LabVIEW type code. |
| unit_string | String | Specifies the unit of the channel data in a .tdmsfile. |
| wf_increment | Double | Represents the increment between two consecutive samples on the *x*-axis. |
| wf_samples | 32-bit signed integer numeric | Represents the number of samples in the first data chunk of the waveform you write to a .tdmsfile. The value must be greater than zero. |
| wf_start_offset | Double | Frequency-domain data and histogram results use this value as the first value on the *x*-axis. |
| wf_start_time | Timestamp | Represents the time at which the waveform was acquired or generated. This property can be zero if the time information is relative or the waveform is not in time domain. |
| NI_ChannelName | String | Specifies the name of the waveform that was written to the channel in a .tdms file. |
| NI_UnitDescription | String | Specifies the units of the waveform that was written to the channel in |

## 8. Open Knee Data File Groups/Channels

| Group | Channel(s) | Notes |
|---|---|---|
| Attributes | Version | This is the simVITRO version of the data file and can be used for proper version control and data interpretation for automated post-processing of the file contents. |
| Kinematics.JCS.Desired | Medial Translation - Desired<br>Posterior Translation - Desired<br>Superior Translation - Desired<br>Flexion Angle - Desired<br>Valgus Angle - Desired<br>Internal Rotation Angle - Desired | This is the desired kinematic state of the tibiofemoral joint. More specifically, it is the kinematic setpoint that the controller was being given, at the time instant this data was recorded to the file, as an input to the controller. |
| Kinematics.JCS.Actual | Medial Translation<br>Posterior Translation<br>Superior Translation<br>Flexion Angle<br>Valgus Angle<br>Internal Rotation Angle | This is the kinematic state of the tibiofemoral joint. More specifically, it is the position that the joint was commanded to be at for the time point this data was recorded in the file. Depending on the speed of the system at the time this may or may not be a perfect representation of the actual joint position. These kinematics are relative JCS kinematics, not absolute JCS kinematics. The offset between the two can be found in the State configuration file in the 'JCS' state section. In addition, if the femur coordinate system was optimized as a result of the experiment setup, this state will reflect the kinematics given the new, optimized, coordinate system. |
| State.JCS | JCS Medial<br>JCS Posterior<br>JCS Superior<br>JCS Flexion<br>JCS Valgus<br>JCS Internal Rotation | This is the kinematic state of the tibiofemoral joint. More specifically, it is the position that the joint is calculated to be given the most recently measured robot position. Compared to the Kinematics.JCS.Actual group, this group is likely to be a better representation of the actual joint position. However, it is subject to some temporal uncertainty of measuring the robot position. These kinematics are relative JCS kinematics, not absolute JCS kinematics. The offset between the two can be found in the State configuration file in the 'JCS' state section. In addition, if the femur coordinate system was optimized as a result of the experiment setup, this state will reflect the kinematics given the new, optimized, coordinate system. |
| State.Knee JCS | Knee JCS Medial<br>Knee JCS Posterior<br>Knee JCS Superior<br>Knee JCS Flexion<br>Knee JCS Valgus<br>Knee JCS Internal Rotation | This is the kinematic state of the tibiofemoral joint. More specifically, it is the position that the joint is calculated to be given the most recently measured positions of the motion capture sensors. Compared to the Kinematics.JCS.Actual group, this group is likely to be a better representation of the actual |

| | | |
|---|---|---|
| | | joint position. However, it is subject to some temporal uncertainty of measuring the motion tracking marker position. Compared to the State.JCS group, this group is likely to be a better representation of the actual joint position in that it can account for bone bending artifacts (depending on where the sensors are mounted). However, it is likely to have a larger noise floor compared to the JCS calculations based on robot position. These kinematics are relative JCS kinematics, not absolute JCS kinematics. The offset between the two can be found in the State configuration file in the 'Knee JCS' state section. In addition, if the femur coordinate system was optimized as a result of the experiment setup, this state will NOT reflect the kinematics given the new, optimized, coordinate system. It will reflect only the original digitized coordinate system. |
| State.Knee PTFJ<br>(Patellofemoral testing only) | Knee PTFJ Medial<br>Knee PTFJ Posterior<br>Knee PTFJ Superior<br>Knee PTFJ Flexion<br>Knee PTFJ Valgus<br>Knee PTFJ Internal Rotation | This is the kinematic state of the patellofemoral joint. More specifically, it is the position that the joint is calculated to be given the most recently measured positions of the motion capture sensors. These kinematics are relative PTFJ kinematics, not absolute PTFJ kinematics. The offset between the two can be found in the State configuration file in the 'Knee PTFJ' state section. In addition, if the femur coordinate system was optimized as a result of the experiment setup, this state will NOT reflect the kinematics given the new, optimized, coordinate system. It will reflect only the original digitized coordinate system. |
| Kinetics.JCS.Desired | Lateral Drawer - Desired<br>Anterior Drawer - Desired<br>Distraction - Desired<br>Extension Torque - Desired<br>Varus Torque - Desired<br>External Rotation Torque - Desired | This is the desired kinetic state of the joint (i.e. tibia loads). More specifically, it is the kinetic setpoint that the controller was being given, at the time instant this data was recorded to the file, as an input to the controller. |

| | | |
|---|---|---|
| State.JCS Load | JCS Load Lateral Drawer<br>JCS Load Anterior Drawer<br>JCS Load Distraction<br>JCS Load Extension Torque<br>JCS Load Varus Torque<br>JCS Load External Rotation Torque | This is the kinetic state of the joint (i.e. tibia loads). More specifically, it is the joint (tibia) loads that are calculated given the most recently measured load cell output and the relative position of the tibia and load cell from the previous loop. Depending on the mounting configuration, and the speed of the system at the time, the temporal uncertainty in the relative position may or may not affect the computed tibia loads. For example, if the tibia is statically mounted to the load cell, then it has no influence on the measurement. If the load cell is mounted to the femur and the system is moving slowly, then again it will have little to no influence. It is only in the case where the femur is mounted to the load cell, and the system is moving quickly, that a chance for the temporal uncertainty to have any affect is possible. |
| Kinematics.Robot.Actual | x<br>y<br>z<br>roll<br>pitch<br>yaw | This is the robot position. More specifically, it is the position that the robot was commanded to be at for the time point this data was recorded in the file. Depending on the speed of the robot at the time this may or may not be a perfect representation of the actual robot position. |
| Sensor.Robot Position | Robot Position_x<br>Robot Position_y<br>Robot Position_z<br>Robot Position_roll<br>Robot Position_pitch<br>Robot Position_yaw | This is the most recently measured position of the robot in the native robot (WORLD2) reference frame. |
| Sensor.Femur Sensor | Femur Sensor_Femur.x<br>Femur Sensor_Femur.y<br>Femur Sensor_Femur.z<br>Femur Sensor_Femur.r<br>Femur Sensor_Femur.p<br>Femur Sensor_Femur.w | This is the most recently measured position of the motion tracking sensor in the femur, in the native (WORLD1) reference frame. |
| Sensor.Tibia Sensor | Tibia Sensor_Tibia.x<br>Tibia Sensor_Tibia.y<br>Tibia Sensor_Tibia.z<br>Tibia Sensor_Tibia.r<br>Tibia Sensor_Tibia.p<br>Tibia Sensor_Tibia.w | This is the most recently measured position of the motion tracking sensor in the tibia, in the native (WORLD1) reference frame. |
| Sensor.Patella Sensor (Patellofemoral testing only) | Patella Sensor_Patella.x<br>Patella Sensor_Patella.y<br>Patella Sensor_Patella.z<br>Patella Sensor_Patella.r<br>Patella Sensor_Patella.p<br>Patella Sensor_Patella.w | This is the most recently measured position of the motion tracking sensor in the patella in the native (WORLD1) reference frame. |
| Sensor.Static Load Cell (Tibiofemoral testing only) | Static Load Cell_Fx<br>Static Load Cell_Fy<br>Static Load Cell_Fz<br>Static Load Cell_Mx<br>Static Load Cell_My<br>Static Load Cell_Mz | This is the most recently measured loads from the load cell in the native load cell reference frame. |

| | | |
|---|---|---|
| Sensor.Control Load Cell (Patellofemoral testing only) | Control Load Cell_Fx (Filtered)<br>Control Load Cell_Fy (Filtered)<br>Control Load Cell_Fz (Filtered)<br>Control Load Cell_Mx (Filtered)<br>Control Load Cell_My (Filtered)<br>Control Load Cell_Mz (Filtered) | This is the most recently measured loads from the load cell in the native load cell reference frame. |
| Kinematics.Nominal JCS Position | Untitled | The joint nominal position values are a 6x1 array of numbers that are the Nominal Position values on the simVITRO main screen. These values get written to the data file at the end of the experiment run. At any point throughout the experiment run the user can press the Define Nominal Position button to define that particular kinematic position as the Nominal Position. This value would be useful in the data file in the case where the user wants to store information about a reference position interesting to that particular experiment run. For example, if the kinematic values of the joint are non-zero under some nominal load state (or at a fixed flexion angle) and they want to measure the change in position, from this nominal position, when applying loads, then storing the nominal position can make quick work of post processing and calculating relative positions. |
| Timing.Control Loop Actual dt | Actual dt | This is the time between the the current and previous data acquisition loop. This loop performs, among other things, 1) data aqcuisition of digitized sensors, 2) state calculations, and 3) writing all data to disk. Note that this group name is a bit of a misnomer in that the control loop is separate from the data acquisition loop, and this value refers to the data acquisition loop. This data is used for resampling and time aligning the digitized sensor, and state data to have a constant dt in the main_processed file. |
| Timing.Sync Trigger | Setpoint Time | This is the time that the setpoint is passing to the controller to define trajectory execution timing. This data is used for time aligning the digitized sensor and state data to properly truncate the data before and after the actual trajectory execution. Note that in the main file this channel will have values of -1 before and after trajectory execution. In the main_processed file all values will be greater than or equal to zero. |

| | | |
|---|---|---|
| Timing.Velocity Multiplier | Velocity Multiplier | This is the value of the master gain (a.k.a. blue slider that is treated as the system Sensitivity or Velocity Multiplier) at the time instant this data was recorded to the file.  This data is used for iterative learning of trajectory gains by scaling the stored trajectory gains by the time history of this user controlled master gain.  These values are in units of percentage where 100 represents an unmodified gain in the stored trajectory gain values.  A value of 50 represents a reduction in the gain by one half. |