



Visual 3D™

Tutorial #3: The Command Processing Pipeline

Revision 3.30

C-Motion, Inc.
15821-A Crabbs Branch Way
Rockville, MD 20855 USA
(301) 840-1919 (phone)
(301) 840-0271 (fax)

support@c-motion.com

Table of Contents

How to Use the Visual3D™ Tutorials	3
To locate sample input files	3
Objectives of Tutorial #3	3
Preparing for Tutorial #3.....	3
The Command Processing Pipeline.....	4
About the Visual3D Pipeline Processor	4
About Command Editing.....	5
Executing the Pipeline.....	6
Pipeline Commands and the Active Files	7
Workspace TAGS.....	9
Pipeline GLOBAL PARAMETERS	10
Pipeline SIGNAL PROCESSING.....	11
Additional Features	14
To show processing history.....	14
Summary.....	14

Acknowledgements: The development of Visual3D software was funded in part by STTR grant (R43 HD37286-01) from the National Institute of Child Health and Human Development (NICHD). C-Motion also gratefully acknowledges and appreciates the assistance provided by the Rehabilitation Medicine Department in the Warren Grant Magnuson Clinical Center at the National Institutes of Health.

How to Use the Visual3D™ Tutorials

There are four tutorials, covering:

- (1) ***Building a Model***
- (2) ***Data Visualization***
- (3) ***The Command Processing Pipeline***
- (4) ***Signal and Event Processing***
- (5) ***Analysis and Reporting.***

Each tutorial starts where the previous one ends. The results of each tutorial are automatically saved as a .cmo file. So you can skip parts or try different features.

To locate sample input files

All the sample input files you will need for this tutorial can be downloaded from the C-Motion website:

<http://www.c-motion.com/support/tutorials.htm>

Objectives of Tutorial #3

This tutorial assumes that Visual3D has been installed and that a model has been created as described in **Tutorial #1** and movement data applied to the model as described in **Tutorial #2**.

The Objective of **Tutorial #3** is to demonstrate the **Command Processing Pipeline**

I won't be surprised if many people feel compelled to *Skim* through this tutorial. Please resist this temptation because the Pipeline is the heart and soul of Visual3D signal and event processing.

The file "Tutorial2.cmo" contains the results from Tutorial#2 and can be used as the starting material for this tutorial. You can download this file from the C-Motion website:

<http://www.c-motion.com/support/tutorials.htm>

Don't be too scared by the command language syntax used by Visual3D. This is not a scripting language or a programming language it is simply a series of commands that are run sequentially. The syntax is actually more straightforward than it looks ;-)

Preparing for Tutorial #3

1. In the **File** menu, select **Open** and then select **Tutorial2.cmo**. This is the file resulting from Tutorial #2, which you either created through Tutorial #2 or downloaded complete from the C-Motion website.
2. Click on **Signal and Event Processing** to visualize the animation of the model based on the movement data and the model that was applied to it. If the animation doesn't appear in the 3D Animation viewer, check the **active file combo box** on the toolbar. It should read 'Walking Trial 1.c3d' rather than ALL-FILES
3. If the animation is not playing, click on the PLAY button of the VCR controls at the bottom of the screen.
4. There are many viewing options for the Animation viewer available under the **View Menu** item or by clicking with the **Right Mouse Button** in the Animation Viewer itself. You should play around with these options to see the effects they have; most of the effects are intuitively obvious.

The Command Processing Pipeline

The **Command Processing Pipeline** provides access to the core of Visual3D's functionality by providing a command line interface into all of Visual3D's functions. The Pipeline is typically used to **automate processing steps**, which is useful for multiple, repeated tasks. The **Pipeline** is a set of Visual3D commands that are processed in sequence. The **Pipeline** has the ability to manage files, define events, execute signal processing computations, create and edit models, and, create and modify reports.

The pipeline processor can be launched from the Pipeline menu option or from the Visual3D toolbar

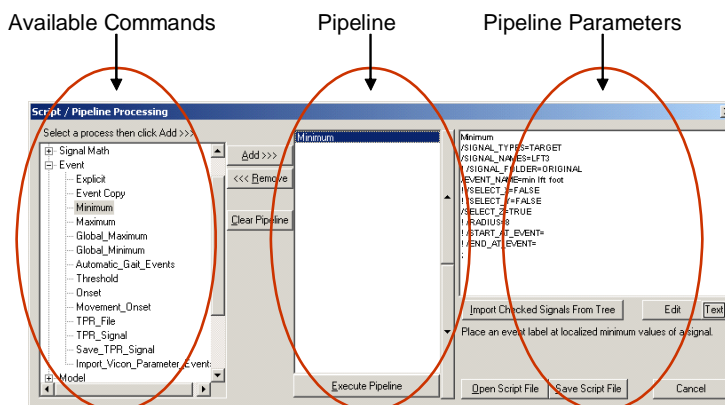


Any pipeline command that can be run interactively through the pipeline processor may be saved to a text file. This file may be reloaded later, or combined with other command files to build a more complex pipeline. The Pipeline dialog (or any text processor) can be used to create the pipeline.

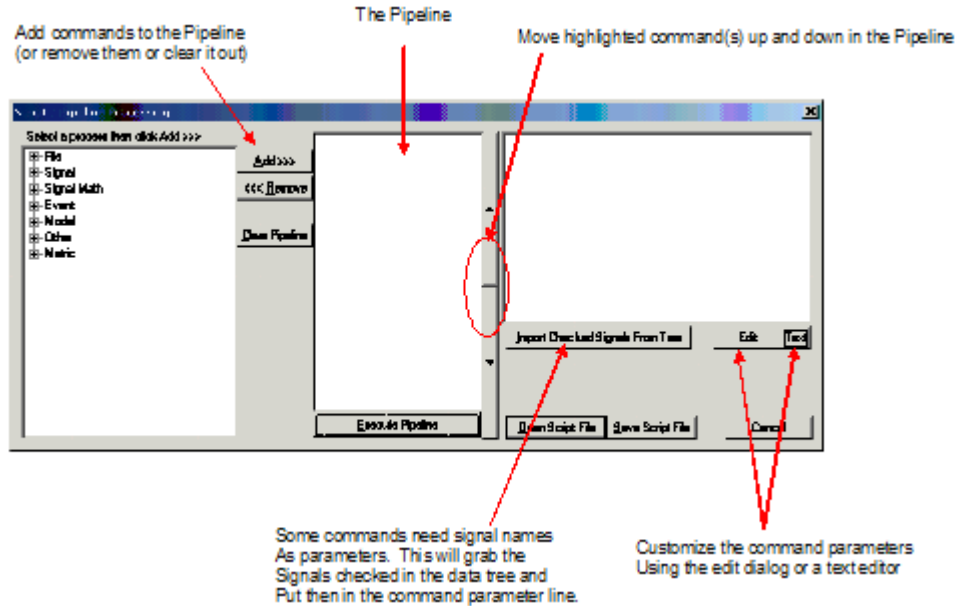
NOTE: In special cases, C-Motion creates custom pipeline commands for users. These commands are fully functional but hidden from the interface. Users requiring these special commands will need to remember the command syntax.

About the Visual3D Pipeline Processor

Pipeline commands are divided into various categories: File, Signal, Event, etc. Commands under these folders are added to the pipeline by expanding the folder for a category, and selecting the function to be added by clicking the "Add" button (or by double clicking on the item). The selected function will be added to the pipeline immediately after the command that is currently highlighted in the pipeline. Commands can be repositioned in the pipeline using the arrow keys to the right of the list box.



The **Pipeline Dialog** has three parts: a list of available commands; a list of commands selected to run; and the details (custom parameters) for the selected command in the pipeline.



- Use **ADD** to add selected commands from the left most list box to the Pipeline and **REMOVE** to remove them.
- To **move a command up or down in the Pipeline**, highlight it and use the arrow bars along the side of the Pipeline window.
- Use **Import Check Signals from Tree** for commands that require signals as parameters. This button will take signals from the Data Tree and add them to the Pipeline parameters. This simplifies the task of entering the **signal type, name and folder** manually.
- Use **Edit** or double click on a command in the middle list box to customize the command parameters. For many commands an edit dialog will appear allowing you to enter the commands more intuitively. If for some reason you need to edit the command in a text editor, select the **Text** button.

About Command Editing

Each Pipeline command consists of two parts - the command and its parameters.

```

Command_Nam
e
/Parameter1= something
/Parameter2= this_value+another_value
/Parameter3= 3.14
!/Parameter4= default_value_for_Parameter4;
    
```

Note that each parameter begins with a slash and that the command is terminated with a semicolon. The generalized command above passes one set of three parameters to the command "Command_Name". To pass more than one set of parameters to the command, list the values of each parameter, separated by "+" signs (e.g. Parameter 2 above). Optional parameters are commented out - thus they start with an exclamation mark.

If the Parameter is not listed in the command or if the "!" character is at the beginning of a line, the parameter value is ignored, and the default values for the parameter are used in the processing.

When a command is created, all parameters are listed, so the user doesn't have to continually check the documentation for the syntax. The script text is provided so that these optional parameters may be un-commented and additional parameter values added.

In cases where the following three parameters are included:

Signal_Name, Signal_Type, and Signal_Folder

Signals that are checked in the data tree can be imported into the command at the push of the **Import Checked Signals From Tree** button.

Executing the Pipeline

Selecting the Execute Pipeline button will cause all of the commands in the middle list box to be executed in sequence. When the pipeline has finished a status screen will pop-up listing the commands that were processed, and any WARNINGS or ERRORS that were generated. NOTE that the pipeline will continue to process even if there were ERRORS in any of the commands, so it is important that you check the status at the end of the processing for any ERRORS.

If you would like the Pipeline processing to halt after the first error. Select the **Halt Script on First Error** button located beside the **Execute Pipeline** button.

Example 1:

Select the Open_File command from the list of command within the File Folder. The following command will be added to the pipeline.

```
Open_File
!   Ask for the Movement data file(s).
!   Multiple files can be selected using CTRL-Click
! /FILE_NAME=
;
```

The default value for FILE_NAME is a blank, which indicates that the user should select the file when the pipeline is executing.

Select **Execute pipeline**.

A dialog will appear to allow you to browse and select a c3d file. The c3d file will be opened in the Workspace.

Example 2:

Modify the previous pipeline by adding another command in the first line.

```
File_New
;
Open_File
;
```

The **File_New** command clears the Visual3D Workspace.

Note: the removal of the **FILE_NAME** parameter in the **Open_File** command had no effect on the command because it was commented out in the first pipeline.

Pipeline Commands and the Active Files

Pipeline commands usually process the "Active Files"; e.g. those files selected in the combo box on the Visual3D toolbar. If only one file is selected in the file selection box, the pipeline, when executed, will perform actions only on the data from that file. If **ALL_FILES** is selected, the script will perform actions on every file individually, as if the script had been run sequentially for each file. If a **TAG** (see next section) is selected, the script will perform actions on every file with that TAG individually, as if the script had been run sequentially for each file.

NOTE: Commands that include the parameter FILE_NAME explicitly act on that file rather the Active Files.

The user can control the Active Files in the Pipeline by adding the command **Select_Active_File** to the Pipeline:

Example 3: Add the **Select_Active_File** command to the previous pipeline.

```
File_New
;
Open_File
;
Select_Active_File
/FILE_NAME=
;
```

Execute the pipeline.: The status dialog should display.

```
-----
File_New
;
-----
Open_File
;
-----
Select_Active_File
/FILE_NAME=
!/QUERY=
;
ERROR! - Parameter FILE_NAME is required for Select_Active_File processing!
ERROR! - Selected file not found!
```

Note: the ERROR message after the Select_Active_File command indicates that the command was not executed properly.

Modify the pipeline as follows:

```
File_New
;
Open_File
;
Select_Active_File
/FILE_NAME= Walking Trial 1.c3d
;
```

Execute the pipeline and load the example file **Tutorial2.cmo**

The Select Active File command will still produce an ERROR message even though it appears that the correct filename has been requested.

This is a potentially frustrating issue related to the Select_Active_File command (and many other commands), which is worth discussing before moving on because it confuses many users. The Filename must be the complete Filename as seen in the Visual3D Workspace. In Visual3D the complete Filename includes the path to the file.

Visual3D uses the path to the file to determine the uniqueness of a file that is loaded in the Workspace. This allows users to use the same filenames for every data collection session (e.g. static, trial1, trial2, etc), distinguishing the files by the folder in which the files are stored on disk. For many manufacturers this is the default method for saving trials that the user can not control.

Modify the pipeline as follows:

```
File_New
;
Open_File
;
Select_Active_File
/FILE_NAME= C:\demo files\tutorials\Walking Trial 1.c3d
;
Switch_to_Event_Processing_Mode
;
```

Execute the pipeline

Select Tutorial2.cmo (you may need to change the type of file to *.* (or *.cmo) in the browse dialog in order to see the cmo files).

Visual3D should make Walking Trial 1.c3d the active file. For convenience, it will also switch to Event processing mode (Command 4 above).

Workspace TAGS

Files can be associated with other files opened in the Visual3D workspace by assigning a TAG to the file. A file can have multiple TAGS associated with it. A TAG can have multiple files associated with it.

For example, all files associated with trials in which the subjects walked barefoot can have a TAG labeled "Barefoot" assigned to them. Those trials that are "Bare Foot" and are only women subjects could have another TAG labeled "Female" to that file. If the TAG is selected as the Active File, all Files of the TAG are selected; Pipeline Commands will act on all of these files.

To assign a TAG to a file, select the **Add New File Tag** button in the middle of the Visual3D Workspace Tab. Type the name of the TAG into the Edit Box that appears. This TAG will then be added to the grid. Check the box beside the files that you want assigned to the Tag.

The Workspace Status page presents a “picture” of the current Workspace contents, in the form of a table having two or more columns as shown below:

Models/Calibration Files	Motion Files	Barefoot	Shoes
s0104Standing.c3d	s0113.c3d	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Lower Body Static Trial.c3d	Walking Trial 2.c3d	<input type="checkbox"/>	<input checked="" type="checkbox"/>
	Walking Trial 3.c3d	<input type="checkbox"/>	<input checked="" type="checkbox"/>
	Walking Trial 4.c3d	<input checked="" type="checkbox"/>	<input type="checkbox"/>
	Walking Trial 1.c3d	<input checked="" type="checkbox"/>	<input type="checkbox"/>

The illustration above shows a workspace in which two models have been defined, one according to standing trial s0104Standing.c3d and the other according to a second standing trial Lower Body Static Trial.c3d. The workspace also contains five movement trials; the first one, s0113.c3d, is associated with the first model, and the other four are associated with the second model. The relationship between movement trials and models is indicated graphically by the coloring of the rows in the table.

The first two columns of the table, headed “Models/Calibration Files” and “Motion Files” are always present. The other two columns in the illustration were added when the Visual3D user defined two file tags called “Barefoot” and “Shoes”. The file tagging mechanism allows you to define specific subsets of your C3D files, to facilitate processing these as a group. In the illustration, the user has defined tags to distinguish movement trials in which the subject walked barefoot from those in which the subject wore shoes. The checkboxes respond to mouse clicks in the expected way, to identify which files are tagged and which are not.

The mechanism for deleting file tags is not always intuitive: if no files are checked in a tag’s column, the tag will be deleted as soon as you switch to a

different page. If you accidentally delete a tag in this way, you can just create it again using the Add New File Tag button on the Workspace Status page

Pipeline GLOBAL PARAMETERS

An important feature of the pipeline is the ability to create and use Global Parameters. A GLOBAL PARAMETER is a way to store a text string for use in Pipeline commands. In one sense it is similar to specifying a global variable in a *Scripting Language*, such as body weight, that could be used in computations. It is actually much more flexible than that in the pipeline. The Visual3D pipeline commands permit multiple entries on a single line, and since the entire line can be represented as a string, a single GLOBAL PARAMETER can represent multiple entries.

The use of GLOBAL PARAMETERS will be described by Example.

The following command uses a Wildcard to load all of the c3d file in a folder.

```
File_New
;
Open_File
/FILE_NAME= C:\demo files\tutorials\*.c3d
;
Switch_to_Event_Processing_Mode
;
```

Executing this pipeline will cause Visual3D to open the two c3d files that we used in Tutorial 1 and Tutorial 2, provided of course that the path name makes sense on your disk. To make the pipeline more general it is convenient to separate the FOLDER containing your data files from the command because in many cases it is only the name of the FOLDER that changes from subject to subject.

The following commands will create a Global Parameter that will contain the Folder, and the Open_File command will use this parameter.

```
File_New
;
Create_Global_Parameter
/PARAMETER_NAME=FOLDER
/PARAMETER_VALUE= C:\demo files\tutorials\
;
Open_File
/FILE_NAME= ::FOLDER&*.c3d
;
Switch_to_Event_Processing_Mode
;
```

This will produce the same results as the previous pipeline. Note that to use a GLOBAL PARAMETER in another command the PARAMETER_NAME should be preceded by two colons (::). The ampersand (&) indicates that the strings “::FOLDER” and “*.c3d” should be concatenated to produce

C:\demo files\tutorials*.c3d

A folder is really a special case of a GLOBAL_PARAMETER because it is used so often. Another command GLOBAL_FOLDER_PARAMETER exists that acts exactly like the GLOBAL_PARAMETER command with ONE EXCEPTION; if the PARAMETER_VALUE is left blank, on execution a BROWSE DIALOG will appear to allow you to browse and select the folder. This eliminates the need to type in the FOLDER parameter for every subject.

```
File_New
;
Create_Global_Folder_Parameter
/PARAMETER_NAME=FOLDER
/PARAMETER_VALUE=
;
Open_File
/FILE_NAME= ::FOLDER&*.c3d
;
Switch_to_Event_Processing_Mode
;
```

Through judicious use of parameter and wildcards very general pipelines can be developed that can be used as batch processing scripts for every day processing of data.

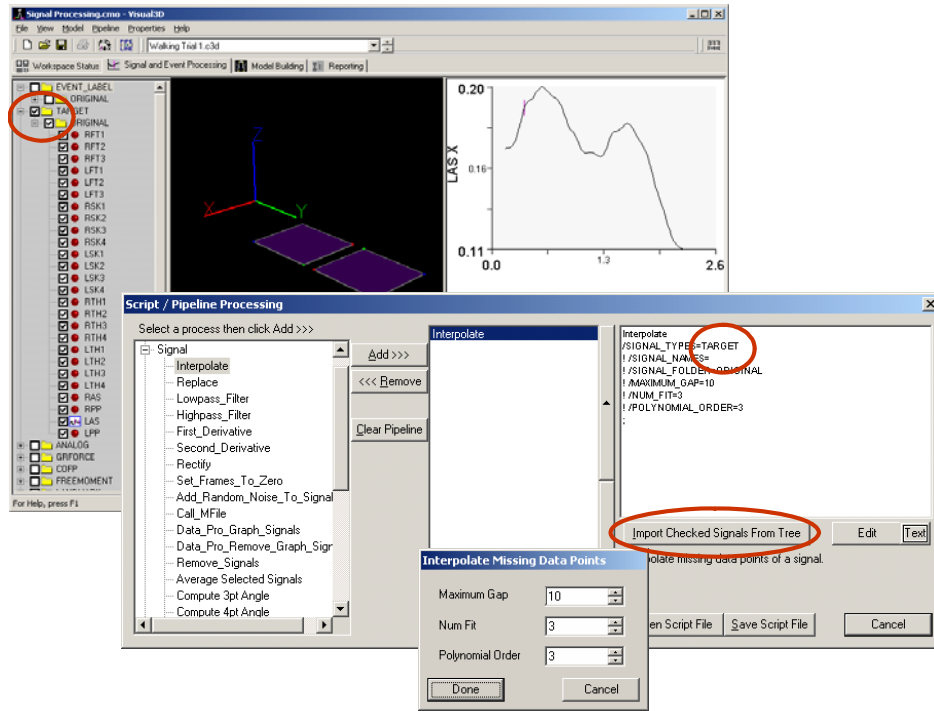
Pipeline SIGNAL PROCESSING

Before ending this tutorial I will jump ahead to something in the next tutorial simply because most Signal and Event Processing in Visual3D is done using the pipeline.

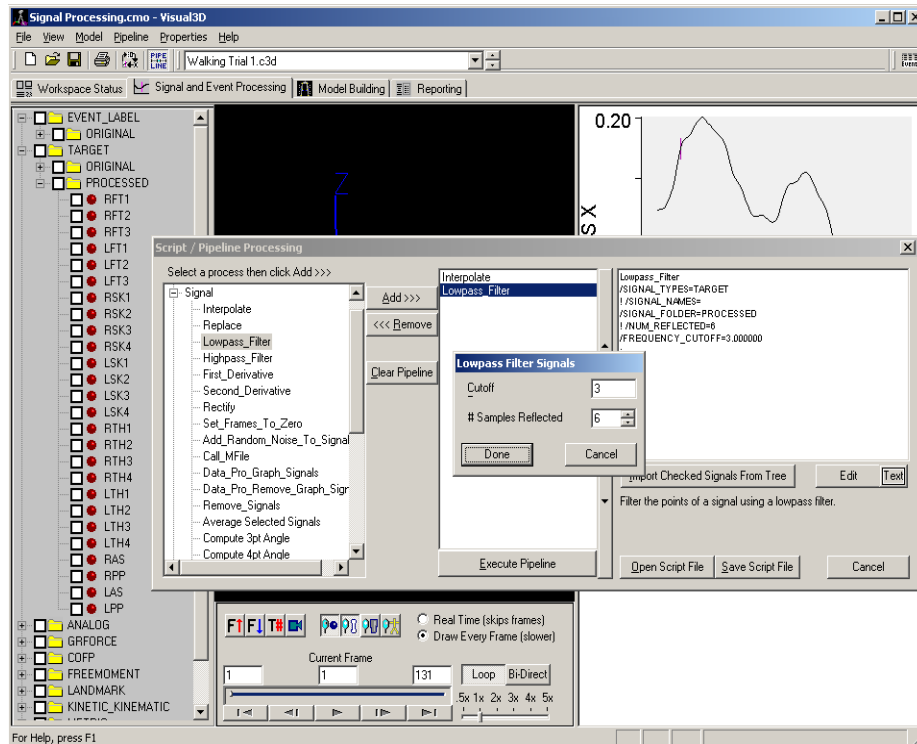
Example: A very common process in treating Motion Capture data is to Interpolate missing frames of data, then smoothing the resulting signal.

For this exercise, you will need to graph the original X signal component of LAS before beginning so that you can compare the original data to the revised data.

1. In the Data Tree, click on the **TARGET** folder, then the **ORIGINAL** folder.
2. *Right-click* **LAS**, select **Graph X** and **New Graph** to create a LAS X graph on the right side of the screen.
3. On the main toolbar, click on the Pipeline button to open the Pipeline Processor. This will open the **Pipeline Processing Dialog**.
4. In the **Select a process then click Add>>** box, click on **Signal** to open the signal options. Select **Interpolate** and click **Add>>** to add it to the pipeline box.



5. In the Select a process then click Add box, select **Signal**, then **Lowpass_Filter**, then click **Add>>>**.

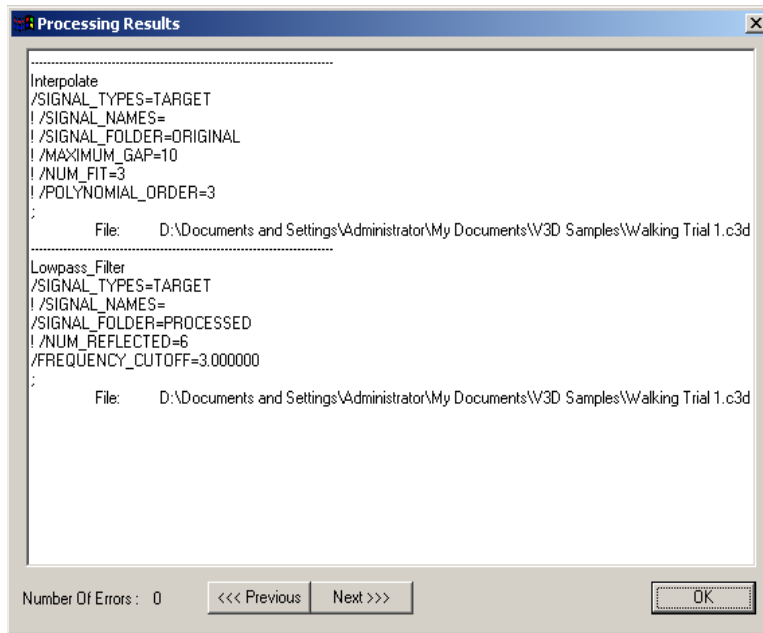


6. Highlight both the Interpolate and Filter commands in the pipeline.

7.. Select which signals to interpolate—in this case, all the targets. In the Data Tree, click the check box of the **ORIGINAL** folder to highlight all target names in that folder. All items should automatically be checked with a checkmark.

6. Click the **Import Checked Signals from Tree** button at the bottom of the Pipeline parameters box. This will automatically import the checked signals.

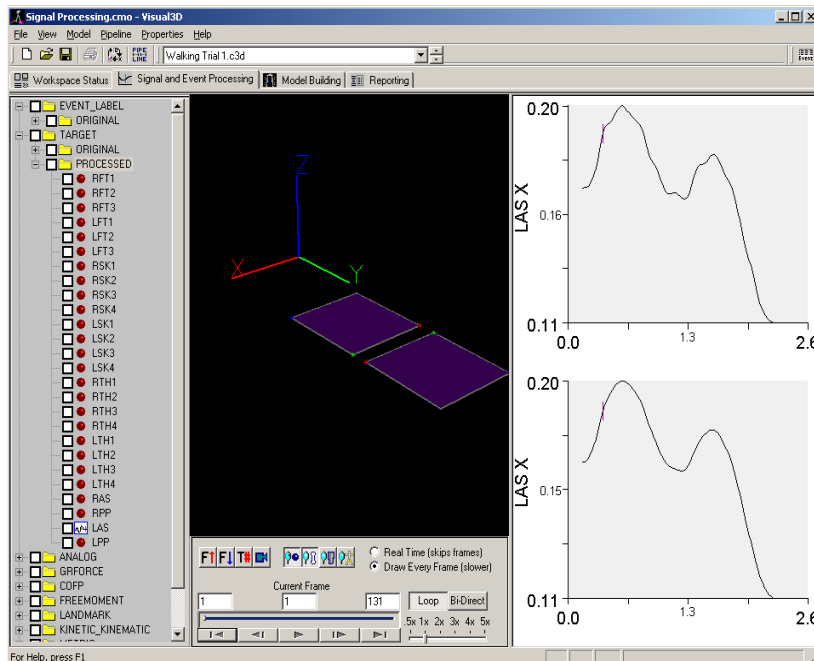
8. Click **Execute Pipeline** to interpolate and filter these signals.



A processing results box will appear, listing both the Interpolate and Lowpass Filter results. The **Number of Errors** is zero in this case.

Click **OK** to close.

At the same time, a new folder, **PROCESSED**, appears in the Data Tree. For subsequent signal processing, you must use signals from this processed list. Otherwise, you will simply overwrite the data you have already processed.



Additional Features

To show processing history