Language:	**English** • français • italiano • português • español ****

THIS PAGE IS UNDER CONSTRUCTION! So it's still being developed, and is a combination of the following wiki pages:

https://www.c-motion.com/v3dwiki/index.php?title=Tutorial:_Command_Pipeline#Pipeline_Command_Syntax:

https://www.c-motion.com/v3dwiki/index.php?title=Visual3D_Pipeline

https://www.c-motion.com/v3dwiki/index.php?title=Pipeline_Parameters

https://www.c-motion.com/v3dwiki/index.php?title=Wildcard

Introduction

Every step you perform in Visual3D through the user interface, can also be accomplished using a pipeline command. The pipeline commands are a way of automating the processing procedure.

The pipeline is a command language, not a scripting language. This essentially means you have a series of commands that act as building blocks, you just have to string them together. For example, if you want to calculate range of motion of the ankle angle, you need to define the ankle angle, calculate the maximum of the ankle angle, calculate the minimum of the ankle angle, and subtract the min from the max.

Once you have created a pipeline, it can be saved as a V3S file so the processing procedure can be repeated for the next processing session.

Pipeline Workshop

You can open the pipeline workshop by going to Pipeline → Workshop or clicking on the button on the toolbar.

The pipeline workshop has three sections:

	* **Left:** * List of available commands * **Center:** * Main Pipeline * List of commands being used * **Right:** * Preview of selected command

Commands can be moved into the Main Pipeline by using the Add » button. Command can be reordered in any sequence using the up/down arrows. Pipelines can be saved to be used again so processing is consistent between subjects.

Editing a Command

When you highlight a command in the **Main Pipeline** and click **Edit** a dialog will appear to edit the command.
When you highlight a command in the **Main Pipeline** and click **Text** a text box will appear to edit the command.

Pipeline Command

Every pipeline command has a text edit option (not all commands have GUI - or user interface). The command text is always set up as follows:

Command Name
/PARAMETER=
! /DEFAULT_PARAMETER=
;

The command name will always appear first, and the semicolon defines the end of the pipeline command. The body of the command contains the parameters.

Any line that starts with an exclamation point ! is not read, it is seen as a comment line. If a parameter is has an exclamation point in front of it, the default value will be used. To change the default value, you must delete the exclamation point and type the desired value to the right of the equals sign for that parameter.

Any line that does not start with an exclamation point defines a required parameter.

Each parameter must start with a slash.

Special Characters

- **!** - if this is the first character of a line, the line is considered a comment. If the ! is at the beginning of a variable name, the default value of the variable is used
- **/** - indicates a command variable

- **+** - is a delimeter separating entries. For example, 1+2 indicates that the variable as two values (1 and 2).
- **&** - indicates a concatenation. For example, this&that will be interpreted in the pipeline as thisthat
- ***** - indicates a wildcard (or multiplication)

When defining signal names or pipeline parameters, NEVER use a mathematical operator or a special character in the signal name. It's also best to avoid spaces (use an underscore _ instead).

Pipeline Parameters

Pipeline parameters can be used to pass data from one command to another. They can be used in many different ways.

A PIPELINE PARAMETER is a way to store a text string for use in Pipeline commands. It is similar to specifying a global variable, 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 PIPELINE PARAMETER can represent multiple entries.

A simple example is to prompt the user for a folder path, then export data to that folder path:

First the user is prompted for the folder path:

Set_Pipeline_Parameter_To_Folder_Path
/PARAMETER_NAME=EXPORT_FOLDER
/PARAMETER_VALUE=
;

The user has defined the parameter name as EXPORT_FOLDER. For this command, if the PARAMETER_VALUE is set blank, it will prompt the user for a folder location. Then everywhere it says EXPORT_FOLDER in the rest of the script, the text will be replaced with the folder path.
Export_Data_To_Ascii_File
/FILE_NAME=::EXPORT_FOLDER&LAnkleAngle.txt
…
;

When referencing a pipeline parameter in a script, you must type two colons before the parameter. If you are concatenating the pipeline parameter with a string (such as LAnkleAngle.txt), you must type an ampersand & after the pipeline parameter. Note sure if an ampersand is necessary? It's always safe to include it anyway **&::EXPORT_FOLDER&** will always work

NOTES:
* Different commands have different behavior depending on what they are intended to do. In another command, leaving a parameter blank will not necessarily bring up a prompt. * The "…" is just skipping over other pipeline parameters for the purpose of this tutorial, do not do this in an actual script.

Expressions

The Evaluate_Expression command allows you to type out an expression using mathematical operators. It's probably the most powerful pipeline command in Visual3D, but it takes time to learn the syntax. The full syntax available within Evaluate_Expression is described here.

Many commands have pipeline parameters that accept expressions using the same syntax as the Evaluate_Expression command.

Signal Names

Signal Names as Expressions

In an expression, a signal is referenced by Signal Type, Signal Folder, Signal Name, and component.

Signals are referenced by SIGNAL_TYPE::SIGNAL_FOLDER::SIGNAL_NAME

This signal would be referenced DERIVED::INVERSE_KINEMATICS::LFT_Rot1 If you look at the data tree in Signals and Events, you will see the highest level folders are the Signal Type. The Type is defined in Visual3D, and you cannot create your own Type. Example: DERIVED The sub folder is the Signal Folder. For most Types you can defined your own folder names (as opposed to the default, ex. Event_Label can only have the Folder ORIGINAL). Example: INVERSE_KINEMATICS Each folder contains signals, they're considered the Signal Name. Example: LFT_Rot1

Important: You will notice there are two colons in between the type, folder and signal name. Two colons are also used to represent a pipeline parameter, but they mean different things.

Signal Names as Parameters

Many commands have the following parameters

```
/Signal_Types= The types of the signals
/Signal_Folder= The name of the signal folder
/Signal_Names= The names of the signals
```

This means the command requires the type, folder and signal names specified in these parameters (instead of using the Expression syntax).

FAQ

Frequently Asked Questions:
You cannot create your own parameters for a command. For example, the default parameters for the Automatic Gait Events command are the following: Automatic_Gait_Events ! /FRAME_WINDOW=8 ! /USE_TPR=TRUE ; You **CANNOT** add a parameter: Automatic_Gait_Events ! /FRAME_WINDOW=8 ! /USE_TPR=TRUE **/EVENT_NAME=TEST** ;

Retrieved from “”