Expression Items

Contents

Data_Tree

Signals should be placed into the expression in the form:

SIGNAL_TYPE::SIGNAL_FOLDER::SIGNAL_NAME To define a specific element of a signal (e.g. the X Component)

SIGNAL_TYPE::SIGNAL_FOLDER::SIGNAL_NAME::X To define a specific frame of a data of signal (e.g. Frame 2)

SIGNAL_TYPE::SIGNAL_FOLDER::SIGNAL_NAME[2] Data stored in the GLOBAL Workspace should be expressed as follows:

GLOBAL::SIGNAL_TYPE::SIGNAL_FOLDER::SIGNAL_NAME NOTE: Global signals can be accessed regardless of the ACTIVE FILES.

Examples for Specifying a Data Tree Signal

TARGET::ORIGINAL::RFT1 = Signal RFT1 in the TARGET type and ORIGINAL folder
ANALOG::PROCESSED::FX1 = Signal FX1 in the ANALOG type and PROCESSED folder
PARAMETER::ANALOG::RATE= Parameter RATE in the ANALOG group of the C3D PARAMETERS

Model_Builder

Model metrics have a simpler syntax. The Signal Type and Signal Folder need not be specified because there is only one version of a signal.

Creating a Landmark at the center of a ball

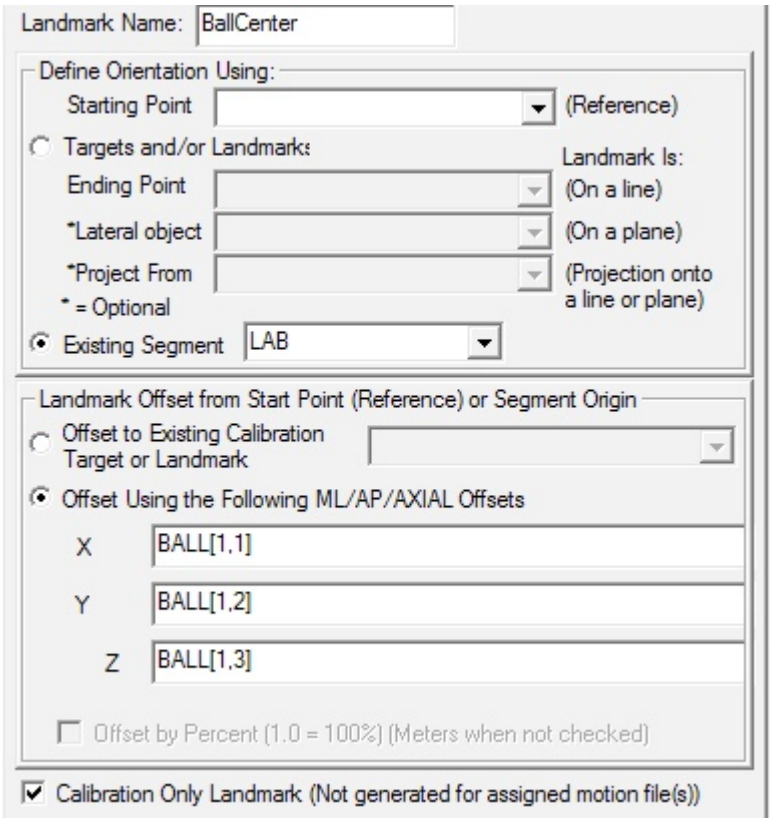
```
Given 6 markers placed on the surface of a round ball

BALL1, BALL2, BALL3, BALL4, BALL5, BALL6

Create a model metric at the center of a best fit sphere to the ball.
Metric Name= BALL
Metric Expression = Best_Fit_Sphere(List(BALL1, BALL2, BALL3, BALL4, BALL5,
BALL6))

The resulting metric will have 3 values separated by a comma (e.g. the 3
components of the center)
As an example a center value of (0.5, 0.6, 0.7) would appear as
0.5,0.6,0.7
```

Create a landmark using this metric data



Note the syntax for the offsets
BALL[1,1]
The syntax may seem a little strange, but [1,1] refers to the first element of the first frame

Pipeline_Parameters

The syntax for using a pipeline parameter as part of an expression is a bit unusual and takes an understanding of how Visual3D parses parameters and pre-processes commands.

The ampersand & is used in pipeline commands to concatenate strings together, and thus is a separator for the parser to find the pieces that need to be parsed separately.

For example, to use a pipeline parameter LP_FREQ:

```
/EXPRESSION=2*pi()*&::LP_FREQ
The ampersand tells the parser to take the 2*pi()* and the ::LP_FREQ
separately through the pre-parser
The first part is just taken as is since it doesn't have a prefix of ::.
The value after the & does have a :: prefix, so it is substituted with the
pipeline parameter.

If you have more complex expressions, you might need to surround each
pipeline parameter with an ampersand

&::LP_FREQ&*&::MULTIPLIER&-&::CONSTANT which may evaluate to something like:
60*1.4-90.0 once all the pipeline parameters are substituted.
The general rule is to surround the pipeline parameter with ampersands.
```

Using a pipeline parameter as part of a signal definition

The syntax is a little funky when it comes to using pipeline parameters in the middle of a signal definition because of the order in which equations are parsed.

The following subtracts two signals.

```
Evaluate_Expression
/EXPRESSION=ANALOG::FILTERED&:&:&::INDEX&:&:&X&-
METRIC::PROCESSED&:&:&::INDEX&_zero
/RESULT_NAME=::INDEX
/RESULT_TYPE=ANALOG
/RESULT_FOLDER=OFFSET
;
```

Tags

TAGS can be expressed as follows:

TAG::TAG_NAME

C3D_Parameters

C3D Parameters can be expressed as follows:

PARAMETER::GROUP::PARAMETER_NAME

Model_Metrics

Specifying an item from a model

The following Model data is available.

```
MODEL::METRIC::NAME
MODEL::SEGMENT::segname::LENGTH
MODEL::SEGMENT::segname::DEPTH
MODEL::SEGMENT::segname::CENTER_OF_MASS
MODEL::SEGMENT::segname::IXX
MODEL::SEGMENT::segname::IYY
MODEL::SEGMENT::segname::IZZ
MODEL::SEGMENT::segname::MASS
MODEL::SEGMENT::segname::PROXIMAL_RADIUS
MODEL::SEGMENT::segname::DISTAL_RADIUS
MODEL::SEGMENT::segname::ORIGIN
```

To refer to a marker

MODEL::TARGET::MARKER_NAME
If you want to refer to a specific component of a signal

MODEL::TARGET::MARKER_NAME::X
To refer to a landmark

MODEL::LANDMARK::LANDMARK_NAME
To refer to a force. **NOTE** This will be available in version 5.0

MODEL::FORCE::FP1
The following command will create a global signal containing the average value of the signal FORCE::FP1 over the range of frames specified for the model (e.g. if a range is not specified, all frames will be used).

The command is executed once for each active file, so the resulting signal will continually be overwritten and the last file processed will correspond to the resulting signal.

```
Evaluate_Expression
/EXPRESSION=MODEL::FORCE::FP1
/RESULT_NAME=GLOBAL::SCOTT
/RESULT_TYPE=DERIVED
! /RESULT_FOLDER=PROCESSED
;
```

If you edit commands in the text editor, which is the only way to use evaluate_expression at the moment, you need to use the 3 letter acronyms for the names.

Default (Internal) Segment Names