Virtual Lab Direction of Progression Method 1

This script is explained in the Virtual Lab Direction of Progression tutorial. This script outlines Method 1.

Before running this script:

- A static trial must be loaded into the workspace
- A model template (with modified virtual lab) must be loaded into the workspace
- A dynamic trial must be loaded into the workspace

NOTE: A modified virtual lab follows the same outline as the Dynamic Virtual Lab, except instead of using the Lab_X landmark, the Dir_Prog_X landmark is used. An example of this model template can be found here.

```
**__Add_a_Comment__**
! /COMMENT= 1.  Find the rotation between the lab and the pelvis segment
(Trunk_Rot)
;

Compute_Model_Based_Data
/RESULT_NAME=RPV_LAB_ROT
/FUNCTION=JOINT_ROTATION
/SEGMENT=RPV
! /REFERENCE_SEGMENT=LAB
/RESOLUTION_COORDINATE_SYSTEM=
! /USE_CARDAN_SEQUENCE=FALSE
! /NORMALIZATION=FALSE
! /NORMALIZATION_METHOD=
! /NORMALIZATION_METRIC=
! /NEGATEX=FALSE
! /NEGATEY=FALSE
! /NEGATEZ=FALSE
! /AXIS1=X
! /AXIS2=Y
! /AXIS3=Z
;

**__Add_a_Comment__**
! /COMMENT= 2. Project the pelvis' coordinate system onto the floor:
;

! Project the Y Axis of the pelvis' coordinate system on the floor, then
find  the magnitude to calculate the unit vector

Evaluate_Expression
/EXPRESSION=VECTOR (LINK_MODEL_BASED::ORIGINAL::RPV_LAB_ROT::4 ,
```

```
LINK_MODEL_BASED::ORIGINAL::RPV_LAB_ROT::5 , 0 )
/RESULT_NAME=RPV_LAB_Y_PROJ
! /RESULT_TYPE=DERIVED
/RESULT_FOLDER=PROGRESSION
;

Signal_Magnitude
/SIGNAL_TYPES=DERIVED
/SIGNAL_FOLDER=PROGRESSION
/SIGNAL_NAMES=RPV_LAB_Y_PROJ
! /RESULT_TYPES=
/RESULT_FOLDER=PROGRESSION
! /RESULT_NAMES=
/RESULT_SUFFIX=_MAG
;

Divide_Signals
/SIGNAL_TYPES=DERIVED+DERIVED
/SIGNAL_FOLDER=PROGRESSION+PROGRESSION
/SIGNAL_NAMES=RPV_LAB_Y_PROJ+RPV_LAB_Y_PROJ_MAG
/COMPONENT_SEQUENCE=ALL
/RESULT_NAME=RPV_LAB_Y_PROJ_UNIT
/RESULT_FOLDER=PROGRESSION
;

! Create the Z Unit vector (0,0,1)

Evaluate_Expression
/EXPRESSION=VECTOR (0,0,1) * ( ( FRAME_NUMBERS::ORIGINAL::FRAMES * 0 + 1 ) )
/RESULT_NAME=RPV_LAB_Z_PROJ_UNIT
/RESULT_TYPE=DERIVED
/RESULT_FOLDER=PROGRESSION
;

! Create the X Unit vector by crossing the Y and Z unit vectors

Evaluate_Expression
/EXPRESSION=CROSS ( DERIVED::PROGRESSION::RPV_LAB_Y_PROJ_UNIT ,
DERIVED::PROGRESSION::RPV_LAB_Z_PROJ_UNIT )
/RESULT_NAME=RPV_LAB_X_PROJ_UNIT
/RESULT_TYPE=DERIVED
/RESULT_FOLDER=PROGRESSION
;

**__Add_a_Comment__**
! /COMMENT= Calculate the metric direction of progression from the X Unit
vector
;

Metric_Mean
```

```
/RESULT_METRIC_NAME=RPV_LAB_X_PROJ_UNIT
! /APPLY_AS_SUFFIX_TO_SIGNAL_NAME=FALSE
/RESULT_METRIC_FOLDER=PROGRESSION
/SIGNAL_TYPES=DERIVED
/SIGNAL_NAMES=RPV_LAB_X_PROJ_UNIT
/SIGNAL_FOLDER=PROGRESSION
/SIGNAL_COMPONENTS=
/COMPONENT_SEQUENCE=ALL
/EVENT_SEQUENCE=
/EXCLUDE_EVENTS=
! /GENERATE_MEAN_AND_STDDEV=TRUE
! /APPEND_TO_EXISTING_VALUES=FALSE
;

**__Add_a_Comment__**
! /COMMENT=Create the virtual lab using X Unit Vector of the direction of
progression
;

Set_Pipeline_Parameter_From_Expression
/PARAMETER_NAME=RPV_PROG_ML
/EXPRESSION=0.1 * ( METRIC::PROGRESSION::RPV_LAB_X_PROJ_UNIT_MEAN::X )
/AS_INTEGER=FALSE
;

Set_Pipeline_Parameter_From_Expression
/PARAMETER_NAME=RPV_PROG_AP
/EXPRESSION=0.1 * ( METRIC::PROGRESSION::RPV_LAB_X_PROJ_UNIT_MEAN::Y )
/AS_INTEGER=FALSE
;

Add_Landmark
/LANDMARK_NAME=Dir_Prog_X
! /CALIBRATION_FILE=
! /USER_GENERATED=TRUE
! /USE_PERCENTAGE=FALSE
! /CALIBRATION_ONLY=FALSE
! /USE_TARGETS=FALSE
/SEGMENT_NAME=LAB
! /TARGET_TYPES=
! /TARGET_NAMES=
/MCS_ML=::RPV_PROG_ML
/MCS_AP=::RPV_PROG_AP
/MCS_AXIAL=0
! /LANDMARK_LOCATION=
! /REFERENCE_LOCATION_TYPE=
! /REFERENCE_LOCATION_NAME=
! /USE_REFERENCE_LOCATION=FALSE
;

Build_Model
```

```
! /CALIBRATION_FILE=
! /REBUILD_ALL_MODELS=FALSE
! /DISPLAY_RESULTS=TRUE
;


Recalc

;
```