# *BodyBuilder session 7*

Kinetic Functions

## *Course Objectives*

1. *Using the REACTION function*
2. *Using the CONNECT function*
3. *Managing force platforms in BodyBuilder*
    1. *The ForceVector Macro*
    2. *The Auto connect feature*
4. *Manually assigning forces to the corresponding foot*

# Kinetic Reactions

## *Kinetic Objects in BodyLanguage*

- FORCE
  - Force objects have identical properties to points
  - SYNTAX
    - Force = {ForceX, ForceY, ForceZ}
    - ForceX = 1(Force)

- MOMENT
  - Moment objects have identical properties to points
  - SYNTAX
    - Moment = {MomentX, MomentY, MomentZ}
    - MomentX = 1(Moment)

- REACTION
  - A Reaction is a combination of force, moment and point of application
  - A Reaction is expressed as a 3 x 3 matrix
  - Force = 1(Reaction)
  - Moment = 2(Reaction)
  - Point = 3(Reaction)
  - Force, Moment, Point all are three components vectors
  - SYNTAX
    - Reaction = |Force, Moment, Point|

# Introduction to inverse dynamics

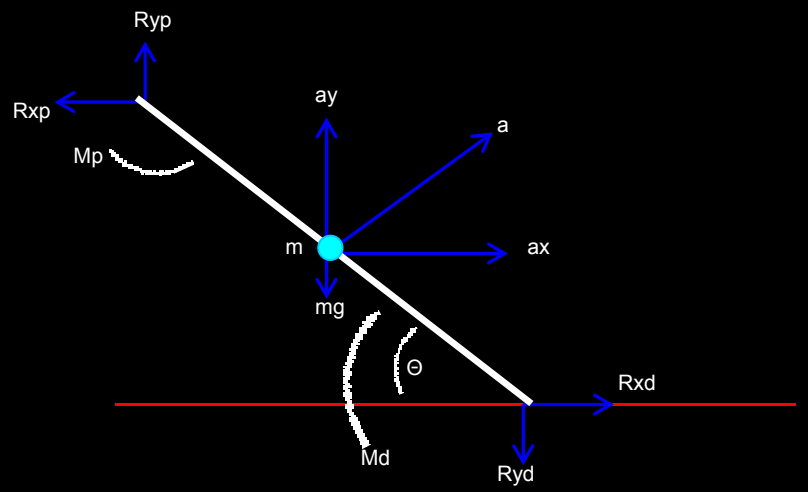
*Assuming that the Reaction at the distal end of the segment is known, it is possible to calculate the Reaction at the proximal end of the segment.*

*Known Quantities:*

- *ax, ay: acceleration of the segment's CoM*
- *ά: angular acceleration of the segment in the plane of motion*
- *$I_0$: Moment of inertia of the segment*
- *Rxd, Ryd: Reaction Forces acting at the distal end of the segment (usually determined as the proximal reaction acting on a distal segment)*
- *Md: net Moment acting at the distal end of the segment*

*Unknown Quantities:*

- *Rxp, Ryp: Reaction Forces acting at the proximal end of the segment*
- *Mp: net Moment acting at the distal end of the segment*




$$\Sigma F_x = ma_x \Longrightarrow Rxp$$

$$\Sigma F_y = ma_y \Longrightarrow Ryp$$

$$\Sigma M = I_0 ά \Longrightarrow Mp$$

## *The REACTION function*

- In BodyLanguage, the reaction is calculated with a single command and solves the equations of motion of a segment, incorporating all components due to
    - the action of child segments
    - segment mass distribution
    - gravity and inertia.

- Syntax:

```
ReactionR = Reaction (Segment1)
```

- Output is 3 x 3 matrix  |ForceF,MomentM,PointP|
    - Force (Fx,Fy,Fz)
    - Moment (Mx,My,Mz)
    - Point of Application of the reaction on the specified segment (Px,Py,Pz)

## *Using the REACTION function – Decomposition and re-composition*

The output of the REACTION function can be decomposed into its individual components

- ForceF = 1(ReactionR) or ReactionR(1)
- MomentM = 2(ReactionR) or ReactionR(2)
- PointP = 3(ReactionR) or ReactionR(3)

A reaction can then be re-composed (i.e. in the case that one of the components was changed) using straight brackets | …. |

CustomReaction = |ForceF,MomentM,PointP|

## *Using the REACTION function – adjusting the sign and re-composing a reaction*

As with other objects, the sign of any components of the REACTION function can be changed by calling this function with a (-) sign

ReactionR = Reaction (Segment1)

Forces = ReactionR(1) which has 3 components (Fx, Fy, Fz)

NegativeForces = -ReactionR(1)  which is the same as –(Fx, Fy, FZ) = (-Fx, -Fy, -Fz)

negativeForcex = -1(Forces) = -Fx

The reaction can then be Recomposed with the negated forces

ReactionR = |NegativeForces, ReactionR(2), ReactionR(3)|

## *Using the REACTION function – Normalizing for Bodymass*

Normalising for Bodymass

- The reaction must first be decomposed into its components
- Only forces and moments can be normalised

BM = Bodymass
ReactionR = Reaction (Segment1)
Forces = ReactionR(1)
NormalizedForces = Forces/BM
NormalizedMoments = ReactionR(2)/BM
Point = ReactionR(3)

ReactionR = | NormalizedForces, NormalizedMoments, Point|

## *Using the REACTION function – Final example from Golem.mod*

```
        {* Define BodyMass for the subject by calling the parameter from the MP file *}
NN = $BodyMass
        {* Normalize the forces for the LFemur segment *}
LHF = 1(REACTION(LFemur))/NN
        {* negate the third component of the LFemur forces *}
LHF = {1(LHF),2(LHF),-3(LHF)}
        {* Normalize the moments for the Lfemur segment *}
LHM = 2(REACTION(LFemur))/NN
        {* Recompose the REACTION for the Lfemur segment *}
LFemurR = |LHF,LHM,3(REACTION(LFemur))|
        {* Define the LHipForces as the updated forces from the LFEMUR segment *}
LHipForce = 1(LFemurR)
        {* Define the LHipMoments as the updated moments from the LFEMUR segment *}
LHipMoment = 2(LFemurR)

OUTPUT(LHipForce, LHipMoment)
```

NOTE: in order for the forces and moments as calculated using the Reaction function to appear in the correct section of the Nexus Model Outputs section or the Polygon ‘Kinetics’ section, it is necessary to first recompose the reaction and then to extract the forces and moments from there.

## *Using the REACTION function – Final example from Golem.mod*

```
        {* Define BodyMass for the subject by calling the parameter from the MP file *}
NN = $BodyMass
        {* Normalize the forces for the LFemur segment *}
LHF = 1(REACTION(LFemur))/NN
        {* negate the third component of the LFemur forces *}
LHF = {1(LHF),2(LHF),-3(LHF)}
        {* Normalize the moments for the Lfemur segment *}
LHM = 2(REACTION(LFemur))/NN
OUTPUT(LHF, LHM)
        {* Recompose the REACTION for the Lfemur segment *}
LFemurR = |LHF,LHM,3(REACTION(LFemur))|
        {* Define the LHipForces as the updated forces from the LFEMUR segment *}
LHipForce = 1(LFemurR)
        {* Define the LHipMoments as the updated moments from the LFEMUR segment *}
LHipMoment = 2(LFemurR)

OUTPUT(LHipForce, LHipMoment)
```

NOTE: If the part of the code in dark is missing from the .mod file, the model will still execute without giving errors, but the LHF and LHM variables will be treated as 3D points and not as forces and moments.
It is necessary, after any manual modification made to the output of the Reaction function, to recompose the Reaction using the modified variables and then extract those again to make sure those variables will be correctly identified as forces and/or moments

## *The POWER function*

- SYNTAX
  - PowerI = Power (SegmentA,SegmentB)
  - SegmentA is the segment **to** which power flows
  - SegmentB is the segment **from** which power flows

- Scalar output
  - Power is a scalar value and is output as a single component

- Normalization
  - Powers can be normalized for BodyMass by diving the Power output by the subject's BodyMass

EXAMPLE:
NN = BodyMass
LHipPower = POWER( Pelvis, LFemur )/NN
RHipPower = POWER( Pelvis, RFemur )/NN
LKneePower = POWER( LFemur, LTibia )/NN
RKneePower = POWER( RFemur, RTibia )/NN
LAnklePower = POWER( LTibia, LFoot )/NN
RAnklePower = POWER( RTibia, RFoot )/NN

OUTPUT(LHipPower,RHipPower,LKneePower,RKneePower,LAnklePower,RAnklePower)

# The CONNECT function

## *The CONNECT function*

The CONNECT function allows the application of a given Reaction to a given Segment. This means for example that a force being measured at the hand can be applied to the hand.

NOTE - Force plates are automatically connected to the foot segment without the use of a specific CONNECT function.

CONNECT(Segment,Reaction,ScalarTest)

Segment – The segment the specified reaction will be connected to

Reaction – the full definition of the reaction |force, moment, assertion point| that is applied to the segmnet

Scalar test – a condition statement to determine if the reaction is acting on the segment for any given frame

## *The CONNECT function – Additional information*

Specifying the reaction

- Once this function has been called, the reaction function is held internally with the segment name
- The connected reaction is applied to the segment when a REACTION is called for the segment

The Scalar Test

- Allows selective connection of the reaction
- If ScalarTest evaluates to zero for a particular frame, then the Reaction is not applied to the Segment, otherwise it is.
- You can use an IF statement and a measure of the absolute force on the external device to set up the ScalarTest condition

## *The CONNECT function – An example*

```
IF ABS(Force)>20
        ScalarTest=1        {*results in ScalarTest being equal to 1 for |Force|>20 *}
Else
        ScalarTest=0        {* ScalarTest = 0 for |Force|<20 *}
Endif
```

*{* Specify the reaction on the segment as force = CableTension, Moment = Cable Moment and Assertion point = CableApplication. *}*

```
ReactionForce=|CableTension,CableMoment,CableApplication|
```

*{* Use the connect function to then "connect" the reaction from the cable to the RightHand segment with the ScalarTest condition of the force needing to be above 20 N for the reaction to actually apply. *}*

```
CONNECT(RightHand,ReactionForce,ScalarTest) -
```

# Managing Force Platforms in BodyBuilder

## *Managing Force Platforms in BodyBuilder – The ForceVector Macro*

BodyBuilder refers to each of the force platforms installed in the system with the following syntax

ForcePlate1
ForcePlate2
…
ForcePlateN

The order of the force platforms is the same as the one used to set them up in Nexus

Each force platform represents a Reaction in BodyLanguage

EXAMPLE

```
FP1 = ForcePlate1          {* Now FP1 is a 3 x 3 reaction *}
Force1 = 1(FP1)            {* Force1 = {Force1X, Force1Y, Force1Z} *}
Moment1 = 2(FP1)           {* Moment1 = {Moment1X, Moment1Y, Moment1Z} *}
Point1 = 3(FP1)            {* Point1 = {Point1X, Point1Y, Point1Z} *}
```

## *Managing Force Platforms in BodyBuilder – The ForceVector Macro*

For each force plaform, the code below will allow the user to retrieve the GRF, GRM and CoP variables.

In the case below, as soon as the absolute value of the GRF vector goes above 10 (N) the CoP location gets calculated using a typical formula to determine the CoP location from forces and moments measured by the force platform.

```
IF EXIST( ForcePlate1 )

    Force1 = ForcePlate1(1)
    Moment1 = ForcePlate1(2)
    Centre1 = ForcePlate1(3)

    IF ( ABS ( Force1 ) > 10 )
        Point1 = Centre1 + { -Moment1(2)/Force1(3), Moment1(1)/Force1(3), 0 }
    ELSE
        Point1 = Centre1
    ENDIF

    OUTPUT (Point1, Moment1)
    Force1_BB = Force1 + Point1 {* For visualization *}
    OUTPUT(Force1_BB)

ENDIF
```

## *Managing Force Platforms in BodyBuilder – The ForceVector Macro*

The fragment of code in the previous slide can be also grouped in a macro as follows:

```
macro FORCEVECTOR(FP)
If ExistAtAll( FP )
      F_#FP = FP(1)
      M_#FP = FP(2)
      C_#FP = FP(3)
      if ( ABS ( F_#FP ) > 10 )
            P_#FP = C_#FP + { -M_#FP(2)/F_#FP(3), M_#FP(1)/F_#FP(3), -C_#FP(3) }
      else
            P_#FP = C_#FP
      endif
      F_#FP = F_#FP + P_#FP
      OUTPUT ( P_#FP, F_#FP )
EndIf
endmacro
```

## *Managing Force Platforms in BodyBuilder – The autoconnect function*

- The autoconnect feature applies to force plates which have a set position in the global reference frame.

- The Force Plates are set up in Vicon Nexus to be static in relation to the laboratory

- A force plate is automatically identified as a REACTION with |Force, Moment, Centre of Pressure|

- An automatic algorithm uses the following parameters to determine which segments defined in the script are connected to each forceplate.

    - The ForceThreshold must exceed a specified value, default is 10N
    - A segment origin or attachment point must be perpendicularly above the plate, at a distance less then the DistanceThreshold, default is 200mm
    - The closest end of the segment to the plate must also have a velocity less then the VelocityThreshold. The default VelocityThreshold is set to 2000 mm/s
        - The segment deemed to be in contact with the plate is the one which has the shortest perpendicular distance from origin or attachment point to the plate, and the lowest velocity
        - Where two segments have the same shortest distance, the segment with the closest second end (either origin or attachment point) is taken, preferring those ends above the plate.

Manually assigning forces to the corresponding foot

## *Managing Force Platforms in BodyBuilder – The autoconnect function*

If the autoconnect feature needs to be deactivated, one of the three threshold values needs to be given a value always below or above a realistic threshold, so that the condition is never met.

EXAMPLE
$BodyMass = 80
{* turn off autoconnect *}
ForceThreshold = 100*$BodyMass

{* In the case above, the autoconnect feature would only work for those frames where the vertical force measured by the force plate is above 100 times the body mass of the subject, which is basically never going to happen.
Once the autoconnect feature has been deactivated, then the user is free to use the CONNECT function in order to connect the GRV to the subject following some special conditions defined by the user. *}

## *Assignment – session 7*

1. Calculate the reaction for all lower body segments in your code thus far.
2. Normalize the force and moment components of the reactions by bodymass
3. Recreate the reactions with the new force and moment values, keeping the point of application the same.
4. Use the force vector mocro given in this session to find the GRF, GRM, and COP for any force plates in the trial.