Description of BodyMech new functions:

(*): useful only for UL GUI.

**AggregateAllPoints**: *

**AnatomicalFrameDefinition_VECT**: here every point is passed in as complete matrix containing 3 coordinates data for every frame. In this way, operations like "-" and "cross" can be done only at once without loop. The loop for every frame is only used in the last part by getT, where the 4 x 4 x Nframes matrix is created. Originally, there was a loop for every time frame outside AnatomicalFrameDefinition_VECT (slower).

**BMimportULC3D_3_AUTO**: here marker labels are retrieved directly from BODY.CONTEXT.MarkerLabels instead of being hard-coded in the function itself (more general). The marker labels where inserted in BODY.CONTEXT.MarkerLabels while the parsing the BodyModel files, using the new function PushLabelsToBODY.

**CalculateClusterKinematics_NO_WB_OPT2**: here 1) the loop on the number of segment (outer loop) is parametrized on the segment indices in input. 2) the time frame loop is delegated to the function RigidBodyTransformation_VECT_OPT. 3) Waitbars are commented, since in the UL GUI we have our own globsl waitbar. A good update could be add another input called 'verbose'. If 1, than all the messages to command window and waitbars are shown, otherwise no.

**CalculateDefaultStickFigure_NO_WB**: here waitbars are commented.

**CalculateFunctionalJointCenters**: this new function is used to calculate functional joint centers. A full description is placed in it.

**CalculateJointKinematics_OPT3_ADV**: here 1) the joints on which reference based calculation is perfomed are parametrized on the input iJointsAbs 2) the time frames loop is reduced: there only the input for RotationMatrixToCardanicAngles_VECT is prepared, and the calculations are done in RotationMatrixToCardanicAngles_VECT itself.

**CalculatePostureRefKinematics_NO_WB**: here waitbars are commented.

**CalculateVirtualMarkers_NO_WB**: here waitbars are commented.

**DefineLocalClusterFrames_AUTO**: here the first frame in which all the markers are visible is automatically found ('automatic' mode). The previously implemented mode ('static') is still available by the way.

**FindSegmentIndex**: *

**GetMarkerNamesFromClusters**: *

**GetRelevantAngles**: *

**ghestnew_gert_<left/right>_VECT**: the loop for every time frame is pushed here instead of having it outside ghestnew_gert_<left/right>_VECT. The general idea is to push long loops (e.g time frame loops) in the function that is originally in the inner loop. This reduces the computational time, since appearently calling N times a function doing something is worse than calling that function once and letting it do N times something.

**InterpolateMarkerKinematics_ADV**: here a new method ('Cubic') for gap filling is implemented. This method uses a cubic spline to interpolate the gaps, and it is independent from the GCVSPL.dll library, that is valid only for Windows XP 32 bit.

**ProbeAnatomy_AUTO**: here 1) it is possible to use two different type of input data: the absolute path in which all the calibration files are placed, or a structure already containing the data for marker of calibration files. If you choose the first modality, then a prefix needs to be provided; the function will search for every calibration file whose name is "<prefix> + <anatomical landmark as specified in the BodyModel file>".

**RecordReferencePose_AUTO**: here the same new functionality of DefineLocalClusterFrames_AUTO is introduced.

**RigidBodyTransformation_VECT_OPT: this function is used by** CalculateClusterKinematics_NO_WB_OPT2. Here, an attempt to substitute the time frame loop with for any computation is performed. The idea is to put data (of any kind) for a time frame in a cell of a cell-array. The final cell-array will have as many cells as the number of time frames.

Then the idea is to call:

[cell_array_out1, cell_array_out2, ...] = cellfun(@function_to_be_performed_every_timeframe, cell_array_in_1, cell_array_in_2, ... 'UniformOutput', false);

and, with a second time frame loop, put back data from the output cell-arrays to the wanted structure. It has to be tested how much (and in some cases, if) this method is faster than the classical time frame. RigidBodyTransformation_VECT_OPT2 and RigidBodyTransformation_VECT_OPT3 are alternatives than seems to be slower than RigidBodyTransformation_VECT_OPT by the way.

**RotationMatrixToCardanicAngles_VECT**: since in this function the only operations are trigonometric functions that can be performed at once on vectors, the time frame loop (originally outside this function) can be deleted

**SetAnglesMeaning**: *