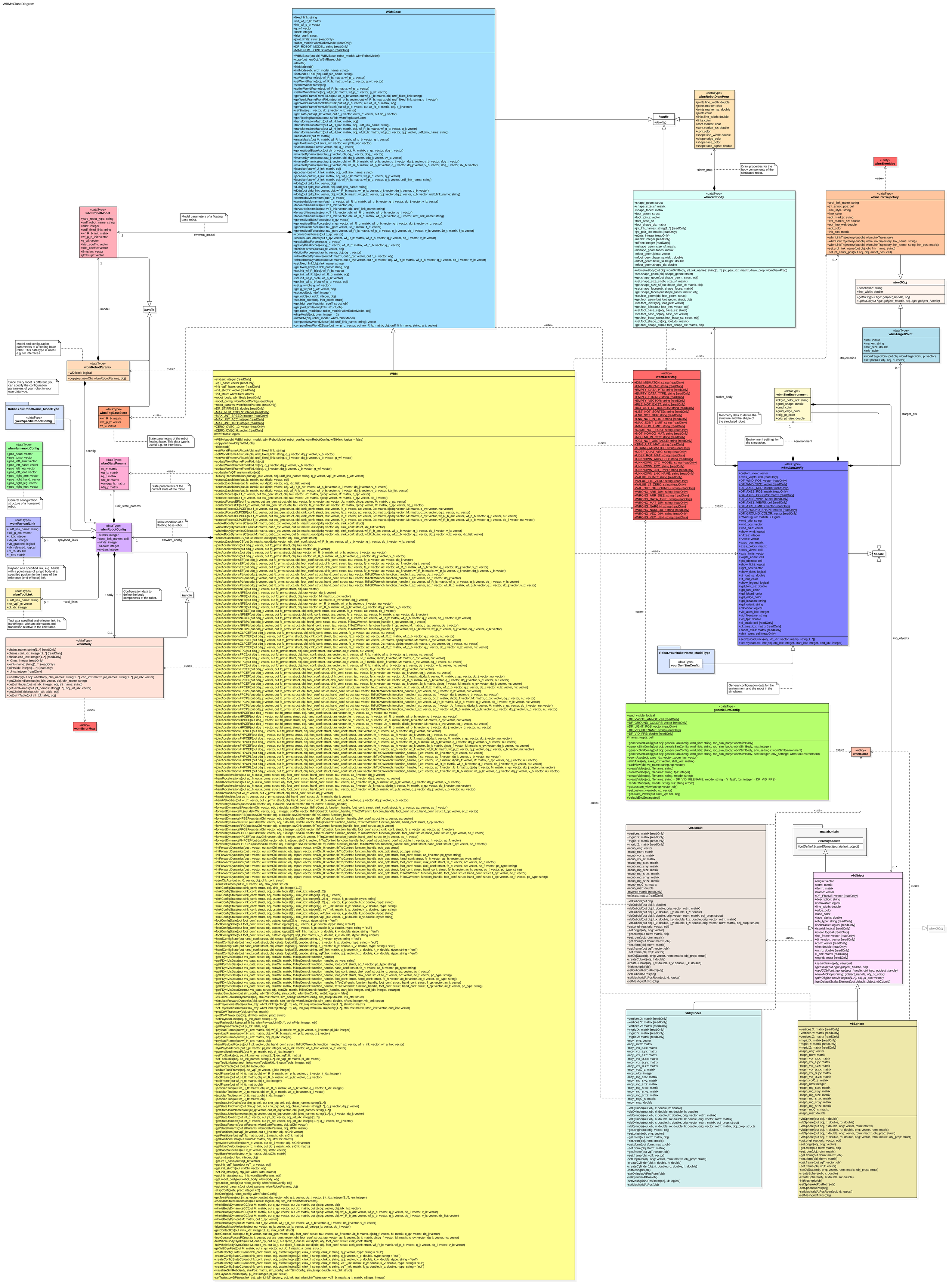
WBM::ClassDiagram


WBMBase
handle
+delete()
«dataType»
wbmRobotDrawProp
+joints.line_width: double
+joints.marker: char
+joints.marker_sz: double
+joints.color
+links.line_width: double
+links.color
+com.marker: char
+com.marker_sz: double
+com.color
+shape.line_width: double
+shape.edge_color
+shape.face_color
+shape.face_alpha: double
+draw_prop
Draw properties for the body components of the simulated robot.
«utility»
wbmErrorMsg
«use»
«dataType»
wbmSimBody
+shape_geom: struct
+shape_size_sf: matrix
+shape_faces: matrix
+foot_geom: struct
+foot_joints: vector
+foot_base_sz
+foot_shape_ds: matrix
+jnt_lnk_names: string[1..*] {readOnly}
+jnt_pair_idx: matrix {readOnly}
+nJnts: integer {readOnly}
+nLnks: integer {readOnly}
+nFeet: integer {readOnly}
-mshape_geom.size_sf: matrix
-mshape_geom.faces: matrix
-mfoot_geom.joints: vector
-mfoot_geom.base_sz.width: double
-mfoot_geom.base_sz.height: double
-mfoot_geom.shape_ds: double
+wbmSimBody(out obj: wbmSimBody, jnt_lnk_names: string[1..*], jnt_pair_idx: matrix, draw_prop: wbmDrawProp)
+set.shape_geom(obj, shape_geom: struct)
+get.shape_geom(out shape_geom: struct, obj)
+set.shape_size_sf(obj, size_sf: matrix)
+get.shape_size_sf(out shape_size_sf: matrix, obj)
+set.shape_faces(obj, shape_faces: matrix)
+get.shape_faces(out shape_faces: matrix, obj)
+set.foot_geom(obj, foot_geom: struct)
+get.foot_geom(out foot_geom: struct, obj)
+set.foot_joints(obj, foot_jnts: vector)
+get.foot_joints(out foot_jnts: vector, obj)
+set.foot_base_sz(obj, base_sz: struct)
+get.foot_base_sz(obj, base_sz: vector)
+get.foot_base_sz(out foot_base_sz: struct, obj)
+set.foot_shape_ds(obj, foot_ds: matrix)
+get.foot_shape_ds(out foot_shape_ds: matrix, obj)
«dataType»
wbmLinkTrajectory
+urdf_link_name: string
+jnt_annot_pos: cell
+line_style: string
+line_color
+ept_marker: string
+ept_marker_sz: double
+ept_line_wid: double
+ept_color
+lnk_pos: matrix
+wbmLinkTrajectory(out obj: wbmLinkTrajectory)
+wbmLinkTrajectory(out obj: wbmLinkTrajectory, lnk_name: string)
+wbmLinkTrajectory(out obj: wbmLinkTrajectory, lnk_name: string, lnk_pos: matrix)
+set.urdf_link_name(out obj, obj, lnk_name: string)
+set.jnt_annot_pos(out obj, obj, annot_pos: cell)
wbmGObj
+description: string
+line_width: double
+getGObj(out hgo: gobject_handle, obj)
+updGObj(out hgo: gobject_handle, obj, hgo: gobject_handle)
«dataType»
wbmTargetPoint
+pos: vector
+marker: string
+mkr_size: double
+mkr_color
+wbmTargetPoint(out obj: wbmTargetPoint, p: vector)
+set.pos(out obj, obj, p: vector)
+trajectories
0..*
«dataType»
wbmRobotModel
+yarp_robot_type: string
+urdf_robot_name: string
+ndof: integer
+urdf_fixed_link: string
+wf_R_b_init: matrix
+wf_p_b_init: vector
+q_wf: vector
+frict_coeff.v: vector
+frict_coeff.c: vector
+jlmts.lwr: vector
+jlmts.upr: vector
Model parameters of a floating base robot.
#mwbm_model
+model
Model and configuration parameters of a floating base robot. This data type is useful e.g. for interfaces.
«dataType»
wbmRobotParams
+wf2fixlnk: logical
+copy(out newObj: wbmRobotParams, obj)
Since every robot is different, you can specify the configuration parameters of your robot in your own data type.
Robot.YourRobotName_ModelType
«dataType»
yourSpecificRobotConfig
«dataType»
wbmFltgBaseState
+wf_R_b: matrix
+wf_p_b: vector
+v_b: vector
State parameters of the robot floating base. This data type is useful e.g. for interfaces.
+config
«dataType»
wbmHumanoidConfig
+jpos_head: vector
+jpos_torso: vector
+jpos_left_arm: vector
+jpos_left_hand: vector
+jpos_left_leg: vector
+jpos_left_foot: vector
+jpos_right_arm: vector
+jpos_right_hand: vector
+jpos_right_leg: vector
+jpos_right_foot: vector
General configuration structure of a humanoid robot.
«dataType»
wbmStateParams
+x_b: matrix
+qt_b: matrix
+q_j: matrix
+dx_b: matrix
+omega_b: matrix
+dq_j: matrix
State parameters of the current state of the robot.
+init_state_params
«dataType»
wbmPayloadLink
+urdf_link_name: string
+link_p_cm: vector
+t_idx: integer
+vb_idx: integer
+vb_grabbed: logical
+vb_released: logical
+m_rb: double
+I_cm: matrix
+payload_links
Payload at a specified link, e.g. hands with a point mass of a rigid body at a specified position in the frame of the reference (end-effector) link.
«dataType»
wbmToolLink
+urdf_link_name: string
+ee_vqT_tt: vector
+pl_idx: integer
+tool_links
Tool at a spezified end-effector link, i.e. hand/finger, with an orientation and translation relative to the link frame.
+body
Configuration data to define the body components of the robot.
«dataType»
wbmRobotConfig
+nCstrs: integer
+cstr_link_names: cell
+nPlds: integer
+nTools: integer
+stvLen: integer
#mwbm_config
«dataType»
wbmBody
+chains.name: string[1..*] {readOnly}
+chains.start_idx: integer[1..*] {readOnly}
+chains.end_idx: integer[1..*] {readOnly}
+nChns: integer {readOnly}
+joints.name: string[1..*] {readOnly}
+joints.idx: integer[1..*] {readOnly}
+nJnts: integer {readOnly}
+wbmBody(out obj: wbmBody, chn_names: string[1..*], chn_idx: matrix, jnt_names: string[1..*], jnt_idx: vector)
+getChainIndices(out jnt_idx: vector, obj, chn_name: string)
+getJointIndex(out jnt_idx: integer, obj, jnt_name: string)
+getJointNames(out jnt_names: string[1..*], obj, jnt_idx: vector)
+getChainTable(out chn_tbl: table, obj)
+getJointTable(out jnt_tbl: table, obj)
«utility»
wbmErrorMsg
WBM
+stvLen: integer {readOnly}
+vqT_base: vector {readOnly}
+init_vqT_base: vector {readOnly}
+init_stvChi: vector {readOnly}
+init_state: wbmStateParams
+robot_body: wbmBody {readOnly}
+robot_config: wbmRobotConfig {readOnly}
+robot_params: wbmRobotParams {readOnly}
+DF_STIFFNESS: double {readOnly}
+MAX_NUM_TOOLS: integer {readOnly}
+MAX_JNT_SPEED: integer {readOnly}
+MAX_JNT_ACC: integer {readOnly}
+MAX_JNT_TRQ: integer {readOnly}
+ZERO_CVEC_12: vector {readOnly}
+ZERO_CVEC_6: vector {readOnly}
#mwf2fixlnk: logical
+WBM(out obj: WBM, robot_model: wbmRobotModel, robot_config: wbmRobotConfig, wf2fixlnk: logical = false)
+copy(out newObj: WBM, obj)
+delete(obj)
+setWorldFrameAtFixLnk(obj, urdf_fixed_link: string)
+setWorldFrameAtFixLnk(obj, urdf_fixed_link: string, q_j: vector, dq_j: vector, v_b: vector)
+setWorldFrameAtFixLnk(obj, urdf_fixed_link: string, q_j: vector, dq_j: vector, v_b: vector, g_wf: vector)
+updateWorldFrameFromFixLnk(obj)
+updateWorldFrameFromFixLnk(obj, q_j: vector, dq_j: vector, v_b: vector)
+updateWorldFrameFromFixLnk(obj, q_j: vector, dq_j: vector, v_b: vector, g_wf: vector)
+updateInitVQTransformation(obj)
+fkInVQTransformation(out vqT_lnk: vector, obj, urdf_link_name: string, q_j: vector, vqT_b: vector, g_wf: vector)
+contactJacobians(out Jc: matrix, out djcdq: vector, obj)
+contactJacobians(out Jc: matrix, out djcdq: vector, obj, idx_list: vector)
+contactJacobians(out Jc: matrix, out djcdq: vector, obj, wf_R_b_arr: vector, wf_p_b: vector, q_j: vector, dq_j: vector, v_b: vector)
+contactJacobians(out Jc: matrix, out djcdq: vector, obj, wf_R_b_arr: vector, wf_p_b: vector, q_j: vector, dq_j: vector, v_b: vector, idx_list: vector)
+contactForces(out f_c: vector, out tau_gen: struct, obj, tau: vector, Jc: matrix, djcdq: vector, M: matrix, c_qv: vector)
+contactForces(out f_c: vector, out tau_gen: struct, obj, tau: vector, Jc: matrix, djcdq: vector, M: matrix, c_qv: vector, dq_j: vector)
+contactForcesEF(out f_c: vector, out tau_gen: struct, obj, tau: vector, fe_c: vector, ac: vector, Jc: matrix, djcdq: vector, M: matrix, c_qv: vector)
+contactForcesEF(out f_c: vector, out tau_gen: struct, obj, tau: vector, fe_c: vector, ac: vector, Jc: matrix, djcdq: vector, M: matrix, c_qv: vector, dq_j: vector)
+contactForcesCLPCEF(out f_c: vector, out tau_gen: struct, obj, clnk_conf: struct, tau: vector, fe_c: vector, ac: vector, Jc: matrix, djcdq: vector, M: matrix, c_qv: vector, nu: vector)
+contactForcesCLPCEF(out f_c: vector, out tau_gen: struct, obj, clnk_conf: struct, tau: vector, fe_c: vector, ac: vector, Jc: matrix, djcdq: vector, M: matrix, c_qv: vector, dq_j: vector, nu: vector)
+contactForcesCLPCEF(out f_c: vector, out tau_gen: struct, obj, clnk_conf: struct, tau: vector, fe_c: vector, ac: vector, Jc: matrix, djcdq: vector, M: matrix, c_qv: vector, wf_R_b_arr: vector, wf_p_b: vector, q_j: vector, nu: vector)
+contactForcesCLPCEF(out f_c: vector, out tau_gen: struct, obj, clnk_conf: struct, tau: vector, fe_c: vector, ac: vector, Jc: matrix, djcdq: vector, M: matrix, c_qv: vector, wf_R_b_arr: vector, wf_p_b: vector, q_j: vector, dq_j: vector, nu: vector)
+wholeBodyDynamicsCS(out M: matrix, out c_qv: vector, out Jc: matrix, out djcdq: vector, obj, clnk_conf: struct)
+wholeBodyDynamicsCS(out M: matrix, out c_qv: vector, out Jc: matrix, out djcdq: vector, obj, clnk_conf: struct, idx_list: vector)
+wholeBodyDynamicsCS(out M: matrix, out c_qv: vector, out Jc: matrix, out djcdq: vector, obj, clnk_conf: struct, wf_R_b_arr: vector, wf_p_b: vector, q_j: vector, dq_j: vector, v_b: vector)
+wholeBodyDynamicsCS(out M: matrix, out c_qv: vector, out Jc: matrix, out djcdq: vector, obj, clnk_conf: struct, wf_R_b_arr: vector, wf_p_b: vector, q_j: vector, dq_j: vector, v_b: vector, idx_list: vector)
+contactJacobiansCS(out Jc: matrix, out djcdq: vector, obj, clnk_conf: struct)
+contactJacobiansCS(out Jc: matrix, out djcdq: vector, obj, clnk_conf: struct, wf_R_b_arr: vector, wf_p_b: vector, q_j: vector, dq_j: vector, v_b: vector)
+pointAccelerations(out ddq_j: vector, out fd_prms: struct, obj, tau: vector)
+pointAccelerations(out ddq_j: vector, out fd_prms: struct, obj, tau: vector, dq_j: vector)
+pointAccelerations(out ddq_j: vector, out fd_prms: struct, obj, tau: vector, wf_R_b: matrix, wf_p_b: vector, q_j: vector, nu: vector)
+pointAccelerations(out ddq_j: vector, out fd_prms: struct, obj, tau: vector, wf_R_b: matrix, wf_p_b: vector, q_j: vector, dq_j: vector, v_b: vector)
+pointAccelerationsEF(out ddq_j: vector, out fd_prms: struct, obj, foot_conf: struct, clnk_conf: struct, tau: vector, fe_c: vector, ac: vector, dq_j: vector)
+pointAccelerationsEF(out ddq_j: vector, out fd_prms: struct, obj, foot_conf: struct, clnk_conf: struct, tau: vector, fe_c: vector, ac: vector, ac_f: vector, dq_j: vector)
+pointAccelerationsEF(out ddq_j: vector, out fd_prms: struct, obj, foot_conf: struct, clnk_conf: struct, tau: vector, fe_c: vector, ac: vector, wf_R_b: matrix, wf_p_b: vector, q_j: vector, dq_j: vector, v_b: vector)
+pointAccelerationsEF(out ddq_j: vector, out fd_prms: struct, obj, foot_conf: struct, clnk_conf: struct, tau: vector, fe_c: vector, ac: vector, ac_f: vector, wf_R_b: matrix, wf_p_b: vector, q_j: vector, dq_j: vector, v_b: vector)
+pointAccelerationsPL(out ddq_j: vector, out fd_prms: struct, obj, foot_conf: struct, hand_conf: struct, tau: vector, fhTotCWrench: function_handle, f_cp: vector, dq_j: vector)
+pointAccelerationsPL(out ddq_j: vector, out fd_prms: struct, obj, foot_conf: struct, hand_conf: struct, tau: vector, fhTotCWrench: function_handle, f_cp: vector, ac_f: vector, dq_j: vector)
+pointAccelerationsPL(out ddq_j: vector, out fd_prms: struct, obj, foot_conf: struct, hand_conf: struct, tau: vector, fhTotCWrench: function_handle, f_cp: vector, wf_R_b: matrix, wf_p_b: vector, q_j: vector, dq_j: vector, v_b: vector)
+pointAccelerationsPL(out ddq_j: vector, out fd_prms: struct, obj, foot_conf: struct, hand_conf: struct, tau: vector, fhTotCWrench: function_handle, f_cp: vector, ac_f: vector, wf_R_b: matrix, wf_p_b: vector, q_j: vector, dq_j: vector, v_b: vector)
+pointAccelerationsNFB(out ddq_j: vector, out fd_prms: struct, obj, tau: vector)
+pointAccelerationsNFB(out ddq_j: vector, out fd_prms: struct, obj, tau: vector, dq_j: vector)
+pointAccelerationsNFB(out ddq_j: vector, out fd_prms: struct, obj, tau: vector, M: matrix, c_qv: vector)
+pointAccelerationsNFB(out ddq_j: vector, out fd_prms: struct, obj, tau: vector, M: matrix, c_qv: vector, dq_j: vector)
+pointAccelerationsNFB(out ddq_j: vector, out fd_prms: struct, obj, tau: vector, wf_R_b: matrix, wf_p_b: vector, q_j: vector, nu: vector)
+pointAccelerationsNFB(out ddq_j: vector, out fd_prms: struct, obj, tau: vector, wf_R_b: matrix, wf_p_b: vector, q_j: vector, v_b: vector)
+pointAccelerationsNFBEF(out ddq_j: vector, out fd_prms: struct, obj, clnk_conf: struct, tau: vector, fe_c: vector, ac: vector, dq_j: vector)
+pointAccelerationsNFBEF(out ddq_j: vector, out fd_prms: struct, obj, clnk_conf: struct, tau: vector, fe_c: vector, ac: vector, M: matrix, c_qv: vector, dq_j: vector)
+pointAccelerationsNFBEF(out ddq_j: vector, out fd_prms: struct, obj, clnk_conf: struct, tau: vector, fe_c: vector, ac: vector, wf_R_b: matrix, wf_p_b: vector, q_j: vector, dq_j: vector, v_b: vector)
+pointAccelerationsNFBPL(out ddq_j: vector, out fd_prms: struct, obj, hand_conf: struct, tau: vector, fhTotCWrench: function_handle, f_cp: vector, dq_j: vector)
+pointAccelerationsNFBPL(out ddq_j: vector, out fd_prms: struct, obj, hand_conf: struct, tau: vector, fhTotCWrench: function_handle, f_cp: vector, M: matrix, c_qv: vector, dq_j: vector)
+pointAccelerationsNFBPL(out ddq_j: vector, out fd_prms: struct, obj, hand_conf: struct, tau: vector, fhTotCWrench: function_handle, f_cp: vector, wf_R_b: matrix, wf_p_b: vector, q_j: vector, dq_j: vector, v_b: vector)
+pointAccelerationsCLPCEF(out ddq_j: vector, out fd_prms: struct, obj, clnk_conf: struct, tau: vector, fe_c: vector, ac: vector, nu: vector)
+pointAccelerationsCLPCEF(out ddq_j: vector, out fd_prms: struct, obj, clnk_conf: struct, tau: vector, fe_c: vector, ac: vector, wf_R_b: matrix, wf_p_b: vector, q_j: vector, nu: vector)
+pointAccelerationsCLPCEF(out ddq_j: vector, out fd_prms: struct, obj, clnk_conf: struct, tau: vector, fe_c: vector, ac: vector, Jc: matrix, djcdq: vector, M: matrix, c_qv: vector, nu: vector)
+pointAccelerationsCLPCEF(out ddq_j: vector, out fd_prms: struct, obj, clnk_conf: struct, tau: vector, fe_c: vector, ac: vector, Jc: matrix, djcdq: vector, M: matrix, c_qv: vector, dq_j: vector, nu: vector)
+pointAccelerationsCLPCEF(out ddq_j: vector, out fd_prms: struct, obj, clnk_conf: struct, tau: vector, fe_c: vector, ac: vector, wf_R_b: matrix, wf_p_b: vector, q_j: vector, dq_j: vector, v_b: vector, nu: vector)
+pointAccelerationsFPC(out ddq_j: vector, out fd_prms: struct, obj, foot_conf: struct, tau: vector, ac_f: vector, nu: vector)
+pointAccelerationsFPC(out ddq_j: vector, out fd_prms: struct, obj, foot_conf: struct, tau: vector, ac_f: vector, wf_R_b: matrix, wf_p_b: vector, q_j: vector, nu: vector)
+pointAccelerationsFPC(out ddq_j: vector, out fd_prms: struct, obj, foot_conf: struct, tau: vector, ac_f: vector, Jc_f: matrix, djcdq_f: vector, M: matrix, c_qv: vector, nu: vector)
+pointAccelerationsFPC(out ddq_j: vector, out fd_prms: struct, obj, foot_conf: struct, tau: vector, ac_f: vector, Jc_f: matrix, djcdq_f: vector, M: matrix, c_qv: vector, dq_j: vector, nu: vector)
+pointAccelerationsFPCEF(out ddq_j: vector, out fd_prms: struct, obj, foot_conf: struct, hand_conf: struct, tau: vector, fe_c: vector, ac: vector, ac_f: vector, nu: vector)
+pointAccelerationsFPCEF(out ddq_j: vector, out fd_prms: struct, obj, foot_conf: struct, hand_conf: struct, tau: vector, fe_c: vector, ac: vector, ac_f: vector, dq_j: vector, nu: vector)
+pointAccelerationsFPCEF(out ddq_j: vector, out fd_prms: struct, obj, foot_conf: struct, hand_conf: struct, tau: vector, fe_c: vector, ac: vector, wf_R_b: matrix, wf_p_b: vector, q_j: vector, dq_j: vector, v_b: vector, nu: vector)
+pointAccelerationsFPCEF(out ddq_j: vector, out fd_prms: struct, obj, foot_conf: struct, hand_conf: struct, tau: vector, fe_c: vector, ac: vector, Jc_f: matrix, djcdq_f: vector, M: matrix, c_qv: vector, dq_j: vector, nu: vector)
+pointAccelerationsFPCEF(out ddq_j: vector, out fd_prms: struct, obj, foot_conf: struct, hand_conf: struct, tau: vector, fe_c: vector, ac: vector, Jc_f: matrix, djcdq_f: vector, M: matrix, c_qv: vector, dq_j: vector, nu: vector)
+pointAccelerationsFPCEF(out ddq_j: vector, out fd_prms: struct, obj, foot_conf: struct, hand_conf: struct, tau: vector, fe_c: vector, ac: vector, wf_R_b: matrix, wf_p_b: vector, q_j: vector, dq_j: vector, v_b: vector, nu: vector)
+pointAccelerationsFHPCEF(out ddq_j: vector, out fd_prms: struct, obj, foot_conf: struct, hand_conf: struct, tau: vector, fhTotCWrench: function_handle, f_cp: vector, dq_j: vector, v_b: vector, nu: vector)
+pointAccelerationsFHPCEF(out ddq_j: vector, out fd_prms: struct, obj, foot_conf: struct, hand_conf: struct, tau: vector, fhTotCWrench: function_handle, f_cp: vector, ac_f: vector, dq_j: vector, nu: vector)
+pointAccelerationsFHPCEF(out ddq_j: vector, out fd_prms: struct, obj, foot_conf: struct, hand_conf: struct, tau: vector, fhTotCWrench: function_handle, f_cp: vector, ac_f: vector, wf_R_b: matrix, wf_p_b: vector, q_j: vector, dq_j: vector, v_b: vector, nu: vector)
+pointAccelerationsFHPCEF(out ddq_j: vector, out fd_prms: struct, obj, foot_conf: struct, hand_conf: struct, tau: vector, fhTotCWrench: function_handle, f_cp: vector, Jc_f: matrix, djcdq_f: vector, M: matrix, c_qv: vector, dq_j: vector, nu: vector)
+pointAccelerationsFHPCEF(out ddq_j: vector, out fd_prms: struct, obj, foot_conf: struct, hand_conf: struct, tau: vector, fhTotCWrench: function_handle, f_cp: vector, ac_f: vector, Jc_f: matrix, djcdq_f: vector, M: matrix, c_qv: vector, dq_j: vector, nu: vector)
+pointAccelerationsFHPCEF(out ddq_j: vector, out fd_prms: struct, obj, foot_conf: struct, hand_conf: struct, tau: vector, fhTotCWrench: function_handle, f_cp: vector, ac_f: vector, wf_R_b: matrix, wf_p_b: vector, q_j: vector, dq_j: vector, v_b: vector, nu: vector)
+pointAccelerationsFHPCL(out ddq_j: vector, out fd_prms: struct, obj, foot_conf: struct, hand_conf: struct, tau: vector, fhTotCWrench: function_handle, f_cp: vector, ac_f: vector, dq_j: vector, v_b: vector, nu: vector)
+pointAccelerationsFHPCL(out ddq_j: vector, out fd_prms: struct, obj, foot_conf: struct, hand_conf: struct, tau: vector, fhTotCWrench: function_handle, f_cp: vector, Jc_f: matrix, djcdq_f: vector, M: matrix, c_qv: vector, dq_j: vector, nu: vector)
+pointAccelerationsFHPCL(out ddq_j: vector, out fd_prms: struct, obj, foot_conf: struct, hand_conf: struct, tau: vector, fhTotCWrench: function_handle, f_cp: vector, wf_R_b: matrix, wf_p_b: vector, q_j: vector, dq_j: vector, v_b: vector, nu: vector)
+pointAccelerationsFHPCL(out ddq_j: vector, out fd_prms: struct, obj, foot_conf: struct, hand_conf: struct, tau: vector, fhTotCWrench: function_handle, f_cp: vector, ac_f: vector, dq_j: vector, nu: vector)
+pointAccelerationsFHPCL(out ddq_j: vector, out fd_prms: struct, obj, foot_conf: struct, hand_conf: struct, tau: vector, fhTotCWrench: function_handle, f_cp: vector, ac_f: vector, Jc_f: matrix, djcdq_f: vector, M: matrix, c_qv: vector, dq_j: vector, nu: vector)
+pointAccelerationsFHPCL(out ddq_j: vector, out fd_prms: struct, obj, foot_conf: struct, hand_conf: struct, tau: vector, fhTotCWrench: function_handle, f_cp: vector, ac_f: vector, wf_R_b: matrix, wf_p_b: vector, q_j: vector, dq_j: vector, v_b: vector, nu: vector)
+handAccelerations(out ac_h: out a_prms: struct, obj, foot_conf: struct, hand_conf: struct, tau: vector, ac_f: vector, dq_j: vector)
+handAccelerations(out ac_h: out a_prms: struct, obj, foot_conf: struct, hand_conf: struct, tau: vector, ac_f: vector, dq_j: vector, nu: vector)
+handAccelerations(out ac_h: out a_prms: struct, obj, foot_conf: struct, hand_conf: struct, tau: vector, ac_f: vector, wf_R_b: matrix, wf_p_b: vector, q_j: vector, dq_j: vector, v_b: vector)
+handAccelerations(out ac_h: out a_prms: struct, obj, foot_conf: struct, hand_conf: struct, tau: vector, ac_f: vector, Jc_f: matrix, djcdq_f: vector, M: matrix, c_qv: vector, dq_j: vector, nu: vector)
+handAccelerations(out ac_h: out a_prms: struct, obj, foot_conf: struct, hand_conf: struct, tau: vector, ac_f: vector, wf_R_b: matrix, wf_p_b: vector, q_j: vector, dq_j: vector, v_b: vector, nu: vector)
+handVelocities(out vc_h: vector, out v_prms: struct, obj, hand_conf: struct, Jc_h: matrix, dq_j: vector)
+handVelocities(out vc_h: vector, out v_prms: struct, obj, hand_conf: struct, wf_R_b: matrix, wf_p_b: vector, q_j: vector, dq_j: vector, v_b: vector)
+forwardDynamics(out dstvChi: vector, obj, t: double, stvChi: vector, fhTrqControl: function_handle)
+forwardDynamicsEF(out dstvChi: vector, obj, t: double, stvChi: vector, fhTrqControl: function_handle, foot_conf: struct, clnk_conf: struct, fe_c: vector, ac: vector, ac_f: vector)
+forwardDynamicsPL(out dstvChi: vector, obj, t: integer, stvChi: vector, fhTrqControl: function_handle, fhTotCWrench: function_handle, foot_conf: struct, hand_conf: struct, f_cp: vector, ac_f: vector)
+forwardDynamicsNFB(out dstvChi: vector, obj, t: double, stvChi: vector, fhTrqControl: function_handle)
+forwardDynamicsNFBEF(out dstvChi: vector, obj, t: double, stvChi: vector, fhTrqControl: function_handle, clnk_conf: struct, fe_c: vector, ac: vector)
+forwardDynamicsNFBPL(out dstvChi: vector, obj, t: double, stvChi: vector, fhTrqControl: function_handle, fhTotCWrench: function_handle, hand_conf: struct, f_cp: vector)
+forwardDynamicsFPC(out dstvChi: vector, obj, t: integer, stvChi: vector, fhTrqControl: function_handle, foot_conf: struct, ac_f: vector)
+forwardDynamicsFPCEF(out dstvChi: vector, obj, t: integer, stvChi: vector, fhTrqControl: function_handle, foot_conf: struct, clnk_conf: struct, fe_c: vector, ac: vector, ac_f: vector)
+forwardDynamicsFPCPL(out dstvChi: vector, obj, t: integer, stvChi: vector, fhTrqControl: function_handle, fhTotCWrench: function_handle, foot_conf: struct, hand_conf: struct, f_cp: vector, ac_f: vector)
+forwardDynamicsHPCEF(out dstvChi: vector, obj, t: integer, stvChi: vector, fhTrqControl: function_handle, hand_conf: struct, fe_h: vector, ac_h: vector, ac_f: vector)
+forwardDynamicsFHPCEF(out dstvChi: vector, obj, t: integer, stvChi: vector, fhTrqControl: function_handle, foot_conf: struct, hand_conf: struct, fe_h: vector, ac_h: vector, ac_f: vector)
+forwardDynamicsFHPCPL(out dstvChi: vector, obj, t: integer, stvChi: vector, fhTrqControl: function_handle, fhTotCWrench: function_handle, foot_conf: struct, hand_conf: struct, f_cp: vector, ac_f: vector)
+initForwardDynamics(out t: vector, out stmChi: matrix, obj, tspan: vector, stvChi_0: vector, fhTrqControl: function_handle, ode_opt: struct)
+initForwardDynamics(out t: vector, out stmChi: matrix, obj, tspan: vector, stvChi_0: vector, fhTrqControl: function_handle, ode_opt: struct, pc_type: string)
+initForwardDynamics(out t: vector, out stmChi: matrix, obj, tspan: vector, stvChi_0: vector, fhTrqControl: function_handle, ode_opt: struct, foot_conf: struct, ac_f: vector, pc_type: string)
+initForwardDynamics(out t: vector, out stmChi: matrix, obj, tspan: vector, stvChi_0: vector, fhTrqControl: function_handle, ode_opt: struct, hand_conf: struct, fe_h: vector, ac_h: vector, pc_type: string)
+initForwardDynamics(out t: vector, out stmChi: matrix, obj, tspan: vector, stvChi_0: vector, fhTrqControl: function_handle, ode_opt: struct, foot_conf: struct, clnk_conf: struct, fe_c: vector, ac: vector, ac_f: vector)
+initForwardDynamics(out t: vector, out stmChi: matrix, obj, tspan: vector, stvChi_0: vector, fhTrqControl: function_handle, ode_opt: struct, foot_conf: struct, clnk_conf: struct, fe_c: vector, ac: vector, ac_f: vector, pc_type: string)
+initForwardDynamics(out t: vector, out stmChi: matrix, obj, tspan: vector, stvChi_0: vector, fhTrqControl: function_handle, ode_opt: struct, foot_conf: struct, hand_conf: struct, fe_h: vector, ac_h: vector, ac_f: vector, pc_type: string)
+initForwardDynamics(out t: vector, out stmChi: matrix, obj, tspan: vector, stvChi_0: vector, fhTrqControl: function_handle, ode_opt: struct, foot_conf: struct, hand_conf: struct, f_cp: vector, ac_f: vector)
+initForwardDynamics(out t: vector, out stmChi: matrix, obj, tspan: vector, stvChi_0: vector, fhTrqControl: function_handle, ode_opt: struct, fhTotCWrench: function_handle, foot_conf: struct, hand_conf: struct, f_cp: vector, ac_f: vector, pc_type: string)
+zeroCtcAcc(out ac_0: vector, obj, clnk_conf: struct)
+zeroExtForces(out fe_0: vector, obj, clnk_conf: struct)
+clnkConfigState(out clnk_conf: struct, obj, clnk_idx: integer[1..2])
+clnkConfigState(out clnk_conf: struct, obj, cstate: logical[2], clnk_idx: integer[1..2])
+clnkConfigState(out clnk_conf: struct, obj, cstate: logical[2], clnk_idx: integer[1..2], q_j: vector)
+clnkConfigState(out clnk_conf: struct, obj, cstate: logical[2], clnk_idx: integer[1..2], q_j: vector, k_p: double, rtype: string)
+clnkConfigState(out clnk_conf: struct, obj, cstate: logical[2], clnk_idx: integer[1..2], q_j: vector, k_p: double, k_v: double, rtype: string)
+clnkConfigState(out clnk_conf: struct, obj, cstate: logical[2], clnk_idx: integer[2], veT_lnk: matrix, k_p: double, k_v: double, rtype: string)
+clnkConfigState(out clnk_conf: struct, obj, cstate: logical[2], clnk_idx: integer[2], vqT_lnk: matrix, k_p: double, k_v: double, rtype: string)
+clnkConfigState(out clnk_conf: struct, obj, cstate: logical[2], clnk_idx: integer, veT_lnk: vector, k_p: double, k_v: double, rtype: string)
+clnkConfigState(out clnk_conf: struct, obj, cstate: logical[2], clnk_idx: integer, vqT_lnk: vector, k_p: double, k_v: double, rtype: string)
+footConfigState(out foot_conf: struct, obj, cstate: logical[2], q_j: vector, rtype: string = "eul")
+footConfigState(out foot_conf: struct, obj, cstate: logical[2], q_j: vector, k_p: double, rtype: string = "eul")
+footConfigState(out foot_conf: struct, obj, cstate: logical[2], q_j: vector, k_p: double, k_v: double, rtype: string = "eul")
+footConfigState(out foot_conf: struct, obj, cstate: logical[2], veT_lnk: matrix, k_p: double, k_v: double, rtype: string = "eul")
+footConfigState(out foot_conf: struct, obj, cstate: logical[2], vqT_lnk: matrix, k_p: double, k_v: double, rtype: string = "eul")
+handConfigState(out hand_conf: struct, obj, cstate: logical[2], cmode: string, q_j: vector, rtype: string = "eul")
+handConfigState(out hand_conf: struct, obj, cstate: logical[2], cmode: string, q_j: vector, k_p: double, rtype: string = "eul")
+handConfigState(out hand_conf: struct, obj, cstate: logical[2], cmode: string, q_j: vector, k_p: double, k_v: double, rtype: string = "eul")
+handConfigState(out hand_conf: struct, obj, cstate: logical[2], cmode: string, veT_lnk: matrix, q_j: vector, k_p: double, k_v: double, rtype: string = "eul")
+handConfigState(out hand_conf: struct, obj, cstate: logical[2], cmode: string, vqT_lnk: matrix, q_j: vector, k_p: double, k_v: double, rtype: string = "eul")
+getFDynVisData(out vis_data: struct, obj, stmChi: matrix, fhTrqControl: function_handle)
+getFDynVisData(out vis_data: struct, obj, stmChi: matrix, fhTrqControl: function_handle, pc_type: string)
+getFDynVisData(out vis_data: struct, obj, stmChi: matrix, fhTrqControl: function_handle, foot_conf: struct, ac_f: vector, pc_type: string)
+getFDynVisData(out vis_data: struct, obj, stmChi: matrix, fhTrqControl: function_handle, hand_conf: struct, fe_h: vector, ac_h: vector, pc_type: string)
+getFDynVisData(out vis_data: struct, obj, stmChi: matrix, fhTrqControl: function_handle, foot_conf: struct, clnk_conf: struct, fe_c: vector, ac: vector, ac_f: vector)
+getFDynVisData(out vis_data: struct, obj, stmChi: matrix, fhTrqControl: function_handle, foot_conf: struct, clnk_conf: struct, fe_c: vector, ac: vector, ac_f: vector, pc_type: string)
+getFDynVisData(out vis_data: struct, obj, stmChi: matrix, fhTrqControl: function_handle, fhTotCWrench: function_handle, foot_conf: struct, hand_conf: struct, f_cp: vector, ac_f: vector)
+getFDynVisData(out vis_data: struct, obj, stmChi: matrix, fhTrqControl: function_handle, fhTotCWrench: function_handle, foot_conf: struct, hand_conf: struct, f_cp: vector, ac_f: vector, pc_type: string)
+getFDynVisDataSect(out vis_data: struct, obj, stmChi: matrix, fhTrqControl: function_handle, start_idx: integer, end_idx: integer, varargin)
+setupSimulation(out sim_config: wbmSimConfig, sim_config: wbmSimConfig, rot3d: logical = false)
+visualizeForwardDynamics(obj, stmPos: matrix, sim_config: wbmSimConfig, sim_tstep: double, vis_ctrl: struct)
+simulateForwardDynamics(obj, stmPos: matrix, sim_config: wbmSimConfig, sim_tstep: double, nRpts: integer, vis_ctrl: struct)
+setTrajectoriesData(out lnk_traj: wbmLinkTrajectory[1..*], obj, lnk_traj: wbmLinkTrajectory[1..*], stmPos: matrix)
+setTrajectoriesData(out lnk_traj: wbmLinkTrajectory[1..*], obj, lnk_traj: wbmLinkTrajectory[1..*], stmPos: matrix, start_idx: vector, end_idx: vector)
+plotCoMTrajectory(obj, stmPos: matrix)
+plotCoMTrajectory(obj, stmPos: matrix, prop: struct)
+setPayloadLinks(obj, pl_lnk_data: struct[1..*])
+getPayloadLinks(out pl_links: wbmPayloadLink[0..*], out nPlds: integer, obj)
+getPayloadTable(out pl_tbl: table, obj)
+payloadFrame(out wf_H_cm: matrix, obj, wf_R_b: matrix, wf_p_b: vector, q_j: vector, pl_idx: integer)
+payloadFrame(out wf_H_cm: matrix, obj, wf_R_b: matrix, wf_p_b: vector, q_j: vector)
+payloadFrame(out wf_H_cm: matrix, obj, pl_idx: integer)
+payloadFrame(out wf_H_cm: matrix, obj)
+handPayloadForces(out f_pl: vector, obj, hand_conf: struct, fhTotCWrench: function_handle, f_cp: vector, wf_v_lnk: vector, wf_a_lnk: vector)
+dynPayloadForce(out f_pl: vector, pl_idx: integer, wf_v_lnk: vector, wf_a_lnk: vector, w_e: vector)
+generalizedInertiaPL(out M_pl: matrix, obj, pl_idx: integer)
+setToolLinks(obj, ee_lnk_names: string[1..*], ee_vqT_tt: matrix)
+setToolLinks(obj, ee_lnk_names: string[1..*], ee_vqT_tt: matrix, pl_idx: vector)
+getToolLinks(out tool_links: wbmToolLink[0..*], out nTools: integer, obj)
+getToolTable(out tool_tbl: table, obj)
+updateToolFrame(obj, ee_vqT_tt: vector, t_idx: integer)
+toolFrame(out wf_H_tt: matrix, obj, wf_R_b: matrix, wf_p_b: vector, q_j: vector, t_idx: integer)
+toolFrame(out wf_H_tt: matrix, obj, wf_R_b: matrix, wf_p_b: vector, q_j: vector)
+toolFrame(out wf_H_tt: matrix, obj, t_idx: integer)
+toolFrame(out wf_H_tt: matrix, obj)
+jacobianTool(out wf_J_tt: matrix, obj, wf_R_b: matrix, wf_p_b: vector, q_j: vector, t_idx: integer)
+jacobianTool(out wf_J_tt: matrix, obj, wf_R_b: matrix, wf_p_b: vector, q_j: vector)
+j