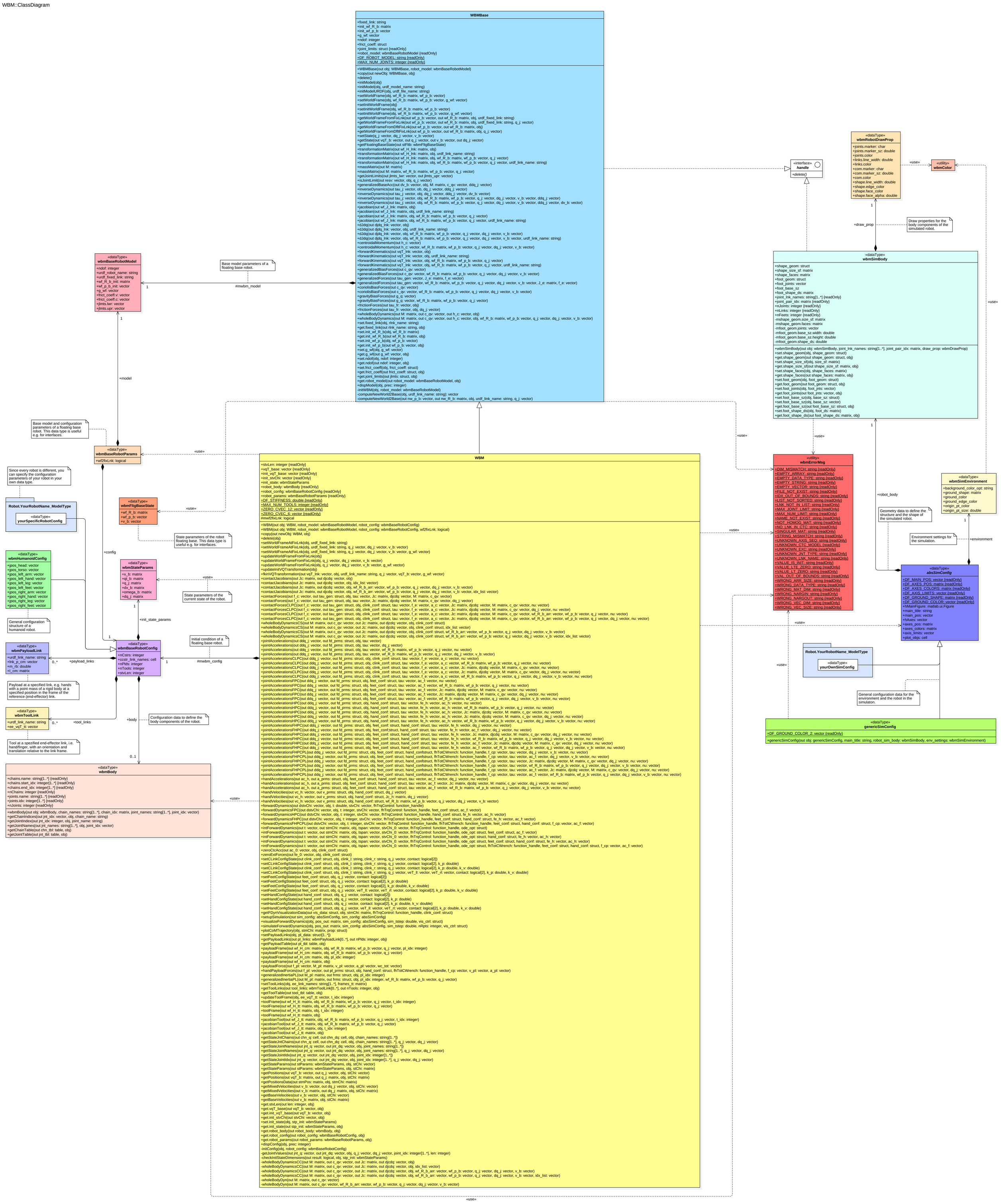
WBM::ClassDiagram


WBMBase
+fixed_link: string
+init_wf_R_b: matrix
+init_wf_p_b: vector
+g_wf: vector
+ndof: integer
+frict_coeff: struct
+joint_limits: struct {readOnly}
+robot_model: wbmBaseRobotModel {readOnly}
+DF_ROBOT_MODEL: string {readOnly}
+MAX_NUM_JOINTS: integer {readOnly}
+WBMBase(out obj: WBMBase, robot_model: wbmBaseRobotModel)
+copy(out newObj: WBMBase, obj)
+delete()
+initModel(obj)
+initModel(obj, urdf_model_name: string)
+initModelURDF(obj, urdf_file_name: string)
+setWorldFrame(obj, wf_R_b: matrix, wf_p_b: vector)
+setWorldFrame(obj, wf_R_b: matrix, wf_p_b: vector, g_wf: vector)
+setInitWorldFrame(obj)
+setInitWorldFrame(obj, wf_R_b: matrix, wf_p_b: vector)
+setInitWorldFrame(obj, wf_R_b: matrix, wf_p_b: vector, g_wf: vector)
+getWorldFrameFromFixLnk(out wf_p_b: vector, out wf_R_b: matrix, obj, urdf_fixed_link: string)
+getWorldFrameFromFixLnk(out wf_p_b: vector, out wf_R_b: matrix, obj, urdf_fixed_link: string, q_j: vector)
+getWorldFrameFromDfltFixLnk(out wf_p_b: vector, out wf_R_b: matrix, obj)
+getWorldFrameFromDfltFixLnk(out wf_p_b: vector, out wf_R_b: matrix, obj, q_j: vector)
+setState(q_j: vector, dq_j: vector, v_b: vector)
+getState(out vqT_b: vector, out q_j: vector, out v_b: vector, out dq_j: vector)
+getFloatingBaseState(out stFltb: wbmFltgBaseState)
+transformationMatrix(out wf_H_lnk: matrix, obj)
+transformationMatrix(out wf_H_lnk: matrix, obj, urdf_link_name: string)
+transformationMatrix(out wf_H_lnk: matrix, obj, wf_R_b: matrix, wf_p_b: vector, q_j: vector)
+transformationMatrix(out wf_H_lnk: matrix, obj, wf_R_b: matrix, wf_p_b: vector, q_j: vector, urdf_link_name: string)
+massMatrix(out M: matrix)
+massMatrix(out M: matrix, wf_R_b: matrix, wf_p_b: vector, q_j: vector)
+getJointLimits(out jlmts_lwr: vector, out jlmts_upr: vector)
+isJointLimit(out resv: vector, obj, q_j: vector)
+generalizedBaseAcc(out dv_b: vector, obj, M: matrix, c_qv: vector, ddq_j: vector)
+inverseDynamics(out tau_j: vector, ob, dq_j: vector, ddq_j: vector)
+inverseDynamics(out tau_j: vector, obj, dq_j: vector, ddq_j: vector, dv_b: vector)
+inverseDynamics(out tau_j: vector, obj, wf_R_b: matrix, wf_p_b: vector, q_j: vector, dq_j: vector, v_b: vector, ddq_j: vector)
+inverseDynamics(out tau_j: vector, obj, wf_R_b: matrix, wf_p_b: vector, q_j: vector, dq_j: vector, v_b: vector, ddq_j: vector, dv_b: vector)
+jacobian(out wf_J_lnk: matrix, obj)
+jacobian(out wf_J_lnk: matrix, obj, urdf_link_name: string)
+jacobian(out wf_J_lnk: matrix, obj, wf_R_b: matrix, wf_p_b: vector, q_j: vector)
+jacobian(out wf_J_lnk: matrix, obj, wf_R_b: matrix, wf_p_b: vector, q_j: vector, urdf_link_name: string)
+dJdq(out djdq_lnk: vector, obj)
+dJdq(out djdq_lnk: vector, obj, urdf_link_name: string)
+dJdq(out djdq_lnk: vector, obj, wf_R_b: matrix, wf_p_b: vector, q_j: vector, dq_j: vector, v_b: vector)
+dJdq(out djdq_lnk: vector, obj, wf_R_b: matrix, wf_p_b: vector, q_j: vector, dq_j: vector, v_b: vector, urdf_link_name: string)
+centroidalMomentum(out h_c: vector)
+centroidalMomentum(out h_c: vector, wf_R_b: matrix, wf_p_b: vector, q_j: vector, dq_j: vector, v_b: vector)
+forwardKinematics(out vqT_lnk: vector, obj)
+forwardKinematics(out vqT_lnk: vector, obj, urdf_link_name: string)
+forwardKinematics(out vqT_lnk: vector, obj, wf_R_b: matrix, wf_p_b: vector, q_j: vector)
+forwardKinematics(out vqT_lnk: vector, obj, wf_R_b: matrix, wf_p_b: vector, q_j: vector, urdf_link_name: string)
+generalizedBiasForces(out c_qv: vector)
+generalizedBiasForces(out c_qv: vector, wf_R_b: matrix, wf_p_b: vector, q_j: vector, dq_j: vector, v_b: vector)
+generalizedForces(out tau_gen: vector, J_e: matrix, f_e: vector)
+generalizedForces(out tau_gen: vector, wf_R_b: matrix, wf_p_b: vector, q_j: vector, dq_j: vector, v_b: vector, J_e: matrix, f_e: vector)
+coriolisBiasForces(out c_qv: vector)
+coriolisBiasForces(out c_qv: vector, wf_R_b: matrix, wf_p_b: vector, q_j: vector, dq_j: vector, v_b: vector)
+gravityBiasForces(out g_q: vector)
+gravityBiasForces(out g_q: vector, wf_R_b: matrix, wf_p_b: vector, q_j: vector)
+frictionForces(out tau_fr: vector, obj)
+frictionForces(out tau_fr: vector, obj, dq_j: vector)
+wholeBodyDynamics(out M: matrix, out c_qv: vector, out h_c: vector, obj)
+wholeBodyDynamics(out M: matrix, out c_qv: vector, out h_c: vector, obj, wf_R_b: matrix, wf_p_b: vector, q_j: vector, dq_j: vector, v_b: vector)
+set.fixed_link(obj, rlnk_name: string)
+get.fixed_link(out rlnk_name: string, obj)
+set.init_wf_R_b(obj, wf_R_b: matrix)
+get.init_wf_R_b(out wf_R_b: matrix, obj)
+set.init_wf_p_b(obj, wf_p_b: vector)
+get.init_wf_p_b(out wf_p_b: vector, obj)
+set.g_wf(obj, g_wf: vector)
+get.g_wf(out g_wf: vector, obj)
+set.ndof(obj, ndof: integer)
+get.ndof(out ndof: integer, obj)
+set.frict_coeff(obj, frict_coeff: struct)
+get.frict_coeff(out frict_coeff: struct, obj)
+get.joint_limits(out jlmts: struct, obj)
+get.robot_model(out robot_model: wbmBaseRobotModel, obj)
+dispModel(obj, prec: integer)
-initWBM(obj, robot_model: wbmBaseRobotModel)
-computeNewWorld2Base(obj, urdf_link_name: string): vector
-computeNewWorld2Base(out nw_p_b: vector, out nw_R_b: matrix, obj, urdf_link_name: string, q_j: vector)
«interface»
handle
+delete()
«dataType»
wbmRobotDrawProp
+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
«use»
«utility»
wbmColor
1
+draw_prop
Draw properties for the body components of the simulated robot.
«dataType»
wbmBaseRobotModel
+ndof: integer
+urdf_robot_name: string
+urdf_fixed_link: string
+wf_R_b_init: matrix
+wf_p_b_init: vector
+g_wf: vector
+frict_coeff.v: vector
+frict_coeff.c: vector
+jlmts.lwr: vector
+jlmts.upr: vector
Base model parameters of a floating base robot.
#mwbm_model
1
+model
«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
+joint_lnk_names: string[1..*] {readOnly}
+joint_pair_idx: matrix {readOnly}
+nJoints: integer {readOnly}
+nLinks: integer {readOnly}
+nFeets: 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, joint_lnk_names: string[1..*], joint_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)
+set.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)
«use»
Base model and configuration parameters of a floating base robot. This data type is useful e.g. for interfaces.
«dataType»
wbmBaseRobotParams
+wf2fixLnk: logical
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.
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: wbmBaseRobotConfig {readOnly}
+robot_params: wbmBaseRobotParams {readOnly}
+DF_STIFFNESS: double {readOnly}
+MAX_NUM_TOOLS: integer {readOnly}
+ZERO_CVEC_12: vector {readOnly}
+ZERO_CVEC_6: vector {readOnly}
#mwf2fixLnk: logical
+WBM(out obj: WBM, robot_model: wbmBaseRobotModel, robot_config: wbmBaseRobotConfig)
+WBM(out obj: WBM, robot_model: wbmBaseRobotModel, robot_config: wbmBaseRobotConfig, wf2fixLnk: logical)
+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)
+contactForcesCLPC(out f_c: vector, out tau_gen: struct, obj, clink_conf: struct, tau: vector, f_e: vector, a_c: vector, Jc: matrix, djcdq: vector, M: matrix, c_qv: vector, nu: vector)
+contactForcesCLPC(out f_c: vector, out tau_gen: struct, obj, clink_conf: struct, tau: vector, f_e: vector, a_c: vector, Jc: matrix, djcdq: vector, M: matrix, c_qv: vector, dq_j: vector, nu: vector)
+contactForcesCLPC(out f_c: vector, out tau_gen: struct, obj, clink_conf: struct, tau: vector, f_e: vector, a_c: vector, Jc: matrix, djcdq: vector, M: matrix, c_qv: vector, wf_R_b_arr: vector, wf_p_b: vector, q_j: vector, nu: vector)
+contactForcesCLPC(out f_c: vector, out tau_gen: struct, obj, clink_conf: struct, tau: vector, f_e: vector, a_c: 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, clink_conf: struct)
+wholeBodyDynamicsCS(out M: matrix, out c_qv: vector, out Jc: matrix, out djcdq: vector, obj, clink_conf: struct, idx_list: vector)
+wholeBodyDynamicsCS(out M: matrix, out c_qv: vector, out Jc: matrix, out djcdq: vector, obj, clink_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, clink_conf: struct, wf_R_b_arr: vector, wf_p_b: vector, q_j: vector, dq_j: vector, v_b: vector, idx_list: vector)
+jointAccelerations(out ddq_j: vector, out fd_prms: struct, obj, tau: vector)
+jointAccelerations(out ddq_j: vector, out fd_prms: struct, obj, tau: vector, dq_j: vector)
+jointAccelerations(out ddq_j: vector, out fd_prms: struct, obj, tau: vector, wf_R_b: matrix, wf_p_b: vector, q_j: vector, nu: vector)
+jointAccelerations(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)
+jointAccelerationsCLPC(out ddq_j: vector, out fd_prms: struct, obj, clink_conf: struct, tau: vector, f_e: vector, a_c: vector, nu: vector)
+jointAccelerationsCLPC(out ddq_j: vector, out fd_prms: struct, obj, clink_conf: struct, tau: vector, f_e: vector, a_c: vector, wf_R_b: matrix, wf_p_b: vector, q_j: vector, nu: vector)
+jointAccelerationsCLPC(out ddq_j: vector, out fd_prms: struct, obj, clink_conf: struct, tau: vector, f_e: vector, a_c: vector, Jc: matrix, djcdq: vector, M: matrix, c_qv: vector, dq_j: vector, nu: vector)
+jointAccelerationsCLPC(out ddq_j: vector, out fd_prms: struct, obj, clink_conf: struct, tau: vector, f_e: vector, a_c: vector, Jc: matrix, djcdq: vector, M: matrix, c_qv: vector, dq_j: vector, nu: vector)
+jointAccelerationsCLPC(out ddq_j: vector, out fd_prms: struct, obj, clink_conf: struct, tau: vector, f_e: vector, a_c: vector, wf_R_b: matrix, wf_p_b: vector, q_j: vector, dq_j: vector, v_b: vector, nu: vector)
+jointAccelerationsFPC(out ddq_j: vector, out fd_prms: struct, obj, feet_conf: struct, tau: vector, ac_f: vector, nu: vector)
+jointAccelerationsFPC(out ddq_j: vector, out fd_prms: struct, obj, feet_conf: struct, tau: vector, ac_f: vector, wf_R_b: matrix, wf_p_b: vector, q_j: vector, nu: vector)
+jointAccelerationsFPC(out ddq_j: vector, out fd_prms: struct, obj, feet_conf: struct, tau: vector, ac_f: vector, Jc: matrix, djcdq: vector, M: matrix, c_qv: vector, nu: vector)
+jointAccelerationsFPC(out ddq_j: vector, out fd_prms: struct, obj, feet_conf: struct, tau: vector, ac_f: vector, Jc: matrix, djcdq: vector, M: matrix, c_qv: vector, dq_j: vector, nu: vector)
+jointAccelerationsFPC(out ddq_j: vector, out fd_prms: struct, obj, feet_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)
+jointAccelerationsHPC(out ddq_j: vector, out fd_prms: struct, obj, hand_conf: struct, tau: vector, fe_h: vector, ac_h: vector, nu: vector)
+jointAccelerationsHPC(out ddq_j: vector, out fd_prms: struct, obj, hand_conf: struct, tau: vector, fe_h: vector, ac_h: vector, wf_R_b: matrix, wf_p_b: vector, q_j: vector, nu: vector)
+jointAccelerationsHPC(out ddq_j: vector, out fd_prms: struct, obj, hand_conf: struct, tau: vector, fe_h: vector, ac_h: vector, Jc: matrix, djcdq: vector, M: matrix, c_qv: vector, nu: vector)
+jointAccelerationsHPC(out ddq_j: vector, out fd_prms: struct, obj, hand_conf: struct, tau: vector, fe_h: vector, ac_h: vector, Jc: matrix, djcdq: vector, M: matrix, c_qv: vector, dq_j: vector, nu: vector)
+jointAccelerationsHPC(out ddq_j: vector, out fd_prms: struct, obj, hand_conf: struct, tau: vector, fe_h: vector, ac_h: vector, wf_R_b: matrix, wf_p_b: vector, q_j: vector, dq_j: vector, v_b: vector, nu: vector)
+jointAccelerationsHPC(out ddq_j: vector, out fd_prms: struct, obj, feet_conf: struct, hand_conf: struct, tau: vector, fe_h: vector, dq_j: vector, nu: vector)
+jointAccelerationsHPC(out ddq_j: vector, out fd_prms: struct, obj, feet_conf:struct, hand_conf: struct, tau: vector, fe_h: vector, ac_f: vector, dq_j: vector, nu: vector)
+jointAccelerationsHPC(out ddq_j: vector, out fd_prms: struct, obj, feet_conf: struct, hand_conf: struct, tau: vector, fe_h: vector, Jc: matrix, djcdq: vector, M: matrix, c_qv: vector, dq_j: vector, nu: vector)
+jointAccelerationsHPC(out ddq_j: vector, out fd_prms: struct, obj, feet_conf: struct, hand_conf: struct, tau: vector, fe_h: vector, wf_R_b: matrix, wf_p_b: vector, q_j: vector, dq_j: vector, v_b: vector, nu: vector)
+jointAccelerationsHPC(out ddq_j: vector, out fd_prms: struct, obj, feet_conf: struct, hand_conf: struct, tau: vector, fe_h: vector, ac_f: vector, Jc: matrix, djcdq: vector, M: matrix, c_qv: vector, dq_j: vector, nu: vector)
+jointAccelerationsHPCL(out ddq_j: vector, out fd_prms: struct, obj, feet_conf: struct, hand_conf: struct, tau: vector, fe_h: vector, ac_f: vector, wf_R_b: matrix, wf_p_b: vector, q_j: vector, dq_j: vector, v_b: vector, nu: vector)
+jointAccelerationsHPCL(out ddq_j: vector, out fd_prms: struct, obj, feet_conf: struct, hand_conf: struct, tau: vector, fe_h: vector, ac_f: vector, Jc: matrix, djcdq: vector, M: matrix, c_qv: vector, dq_j: vector, nu: vector)
+jointAccelerationsHPCL(out ddq_j: vector, out fd_prms: struct, obj, feet_conf: struct, hand_conf: struct, tau: vector, fe_h: vector, ac_f: vector, Jc: matrix, djcdq: vector, M: matrix, c_qv: vector, dq_j: vector, nu: vector)
+jointAccelerationsHPCL(out ddq_j: vector, out fd_prms: struct, obj, feet_conf: struct, hand_conf: struct, tau: vector, fe_h: vector, ac_f: vector, wf_R_b: matrix, wf_p_b: vector, q_j: vector, dq_j: vector, v_b: vector, nu: vector)
+jointAccelerationsHPCL(out ddq_j: vector, out fd_prms: struct, obj, feet_conf: struct, hand_conf: struct, tau: vector, fe_h: vector, ac_f: vector, Jc: matrix, djcdq: vector, M: matrix, c_qv: vector, dq_j: vector, nu: vector)
+jointAccelerationsFHPCL(out ddq_j: vector, out fd_prms: struct, obj, feet_conf: struct, hand_conf: struct, tau: vector, fe_h: vector, ac_f: vector, wf_R_b: matrix, wf_p_b: vector, q_j: vector, dq_j: vector, v_b: vector, nu: vector)
+jointAccelerationsF