Contents

Chapter 1

ROBOOP, A Robotics Object Oriented Package in C++ Hierarchical Index

1.1 ROBOOP, A Robotics Object Oriented Package in C++ Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Chapter 4

ROBOOP, A Robotics Object Oriented Package in C++ Class

- Clik (const mRobot_min_para &mrobot_min_para_, const DiagonalMatrix &Kp_, const DiagonalMatrix &Ko_, const Real eps_=0.04, const Real lambda_-max_=0.04, const Real dt=1.0)

  Constructor.

- Clik (const Clik &x)

  Copy constructor.

- ~Clik ()
- Clik & operator=

- mRobot mrobot

  mRobot instance.

  - mRobot_min_para mrobot_min_para

    mRobot_min_para instance.

  - DiagonalMatrix Kp

pd,: Desired eff position in base frame.

pdd,: Desired eff velocity in base frame.

wd,: Desired eff angular velocity in base frame.

q_,: Output joint position.

qp_,: Output joint velocity.

Definition at line 269 of file clik.cpp.

References CLICK_DH, CLICK_mDH, CLICK_mDH_min_para, dt, endeff_pos_-ori_err(), eps, Integ_Trap(), Robot_basic::jacobian_DLS_inv(), Koe0Quat, Kpep, lambda_max, mrobot, mrobot_min_para, q, qp, qp_prev, robot, robot_type, Robot_-basic::set_q(), and v.

4.1.2.2 int Clik::endeff_pos_ori_err (const ColumnVector & *pd*, constColumnVector & *pdd*, const Quaternion &

4.2 Computed_torque_method Class Reference

```
#include <controller.h>
```

4.2.1 Detailed Description

Computer torque method controller class.

The dynamic model of a robot manipulator can be expressed in joint space as

$$B(q)\ddot{q} + C(q, \dot{q})\dot{q} +$$

Definition at line 588 of file controller.cpp.

References dof, Kp, and WRONG_SIZE.

Referenced by Computed_torque_method().

- bool bPrintErrorMessages

4.4 Control_Select Class Reference

`#include <control_select.h>`

4.4.1 Detailed Description

Select controller class.

This class contains an instance of each controller class. The active controller will be selected when reading a controller file. "type" value correspond to the active controller, ex:

- `type = NONE` : no controller selected
- `type = PD` : Proportional Derivative
- `type = CTM` : Computer Torque Method
-

typeCTM atiDetailRatorgAccr

4.6 Dynamics Class Reference

```
#include <dynamics_sim.h>
```

Inheritance diagram for Dynamics::

- double time

  Time during simulation.

- double to

  Initial simulation time.

- double tf

  Final time used in Runge_Kutta4_Real_time.

- double

- ColumnVector qd

Static Public Attributes

- short set_Dp (const DiagonalMatrix &Dp_)

  Assign the translational impedance damping matrix D_p.

- short set_Dp (const Real Dp_i, const short i)

  Assign the translational impedance damping term $D_p(i, i)$.

- short set_Kp (const DiagonalMatrix &Kp_)

  Assign the translational impedance stifness matrix K_p.

- short set_Kp (const Real Kp_i, const short i)

  Assign the translational impedance stifness term $K_p(i, i)$.

- short set_Mo (const DiagonalMatrix &Mo_)

  Assign the rotational impedance inertia matrix M_o.

- short set_Mo (const Real Mo_i, const short i)

  Assign the rotational impedance inertia term $M_o(i, i)$.

- short set_Do (const DiagonalMatrix &Do_)

  Assign the rotational impedance damping matrix D [illegible]

- ColumnVector wcd

  Difference between wc and desired angular velocity.

Private Attributes

- DiagonalMatrix Mp

  Translational impedance inertia matrix.

- DiagonalMatrix Dp

  Translational impedance damping matrix.

- DiagonalMatrix Kp

  Translational impedance stifness matrix.

Definition at line 166 of file controller.cpp.

References Kp, and WRONG_SIZE.

Referenced by Impedance().

4.8.2.6 short Impedance::set_Kp (const Real *Kp_i*, const short *i*)

Assign the translational impedance stifness term $K_p(i, i)$.

Returns:

short: 0 or WRONG_SIZE if the matrix is not 3×3.

Definition at line 183 of file controller.cpp.

References Kp, and WRONG_SIZE.

4.8.2.7 short Impedance::set_Mo (const DiagonalMatrix & *Mo_*)

Assign the rotational impedance inertia matrix M_o.

Returns:

short: 0 or WRONG_SIZE if the matrix is not 3×3.

Definition at line 199 of file controller.cpp.

References Mo, and WRONG_SIZE.

Referenced by Impedance().

4.8.2.8 short Impedance::set_Mo (const Real *Mo_i*, const short *i*)

Assign the rotational impedance inertia term $M_o(i, i)$.

Returns:

short: 0 or WRONG_SIZE if the matrix is not 3×3.

Definition at line 216 of file controller.cpp.

References Mo, and WRONG_SIZE.

4.8.2.9 short Impedance::set_Do (const DiagonalMatrix & *Do_*)

Assign the rotational impedance damping matrix D_o.

- std::n25tngfilename

  File name.

4.10 Link Class Reference

```
#include <robot.h>
```

4.10.1 Detailed Description

Link definitions.

A n degree of freedom (dof) serial manipulator is composed of n links. This class describe the property of a link. A n dof robot has n instance of the class Link.

Definition at line 137 of file robot.h.

Public Member Functions

- Link

Return a.

- Real get_alpha (void) const

  Return alpha.

- Real get_q (void) const

  Return joint position (theta if joint type is rotoide, d otherwise).

- Real get_theta_min (void) const

  Return theta_min.

- Real get_theta_max (void) const

  Return theta_max.

- Real get_joint_offset (void) const

  Return joint_offset.

- ReturnMatrix get_mc (void)

  Return mc.

- ReturnMatrix get_r (void)

  Return r.

- ReturnMatrix get_p (void) const

  Return p.

- Real get_m (void) const

  Return m.

- Real get_Im (void) const

  Return Im.

- Real get_Gr (void) const

  Return Gr.

- Real get_B (void) const

  Return B.

- Real get_Cf (void) const

  Return Cf.

- ReturnMatrix get_I (void) const

- Real

•

Return the unit vector of the universal joint along the third axis of the fixed revolute joint.

- ReturnMatrix Find_AngularKin (const Real dl, const Real ddl)

  Return the angular speed (Column 1) and angular acceleration (Column 2) of the link.

- ReturnMatrix

- Real I1nn

  Inertia along the tangent axis for part 1.

4.11.3 Member Function Documentation

4.11.3.1 const LinkStewart & LinkStewart::operator= (const LinkStewart &

Parameters:

ddq,: Acceleration of the platform.

Omega,:

4.11.3.7 ReturnMatrix LinkStewart::Find_a (const Matrix *wRp*, const ColumnVector *q*)

Return the position of the attachment point on the platform.

Parameters:

wRp,: Rotation matrix.

q,: Position of the platform.

The position of the attachment point on the platform is equal to the position of the center of the platform plus the position of the attach (in the local referencial) multiplicated by the rotation matrix:

$a = (x, y, z)_q + wRp \cdot a_l$

where*))*

Definition at line 486 of file stewart.cpp.

References aPos, and b.

Referenced by LinkStewart(), and LTransform().

4.11.3.11 ReturnMatrix LinkStewart::Find_VctU ()

Return the unit vector of the universal joint along the first axis of the fixed revolute joint.

This vector is equal to the unitary projection of the link unit vector on the X-Z plane:

$$u_x = \frac{n_x}{\sqrt{n_x^2 +}}$$

4.11.3.13 ReturnMatrix LinkStewart::Find_VctC ()

Return the unit vector of the universal joint along the third axis of the fixed revolute joint.

Eq:

$c = u$

•

4.11.3.19 Real LinkStewart::ActuationForce (const Matrix *J1*, const ColumnVector *C*, const int *Index*, const Real *Gravity* = `GRAVITY`)

Return the actuation force that power the prismatic joint.

Parameters:

J1,: First intermidiate jacobian matrix (find with

- l is the lenght of the link
- l_1 is the distance between the center of mass of the first part of the link to the base
- ω is the angular speed of the link
- α is the angular acceleration of the link
- n is the unit vector of the link
- $\dot{l}$ is the extension rate of the link
- $\ddot{l}$ is the extension acceleration of the link

- virtual ReturnMatrix inv_kin (const Matrix &Tobj, const int mj, const int endlink, bool &converge)

4.12.2.2 ReturnMatrix mRobot::inv_kin (const Matrix & *Tobj*, const int *mj*, const int *endlink*, bool & *converge*) `[virtual]`

Inverse kinematics solutions.

The solution is based on the analytic inverse kinematics if robot type (familly) is Rhino or Puma, otherwise used the numerical algoritm defined in Robot_basic class.

Reimplemented from Robot_basic.

Definition at line 603 of file invkine.cpp.

References Robot_basic::inv_kin(), inv_kin_puma(), inv_kin_rhino(), inv_kin_-schilling(), Robot_basic::PUMA, Robot_basic::RHINO, Robot_basic::robotType, and Robot_basic::SCHILLING.

4.12.2.3 ReturnMatrix mRobot::inv_kin_rhino (const Matrix & *Tobj*, bool & *converge*) `[virtual]`

Analytic Rhino inverse kinematics.

converge will be false if the desired end effector pose is outside robot range.

Implements Robot_basic.

Definition at line 628 of file invkine.cpp.

References Robot_basic::a, Link::a, Link::d, G(), Robot_basic::get_qtsicw,(References)nv_kin_-

4.12.2.5 ReturnMatrix mRobot::inv_kin_schilling (const Matrix & *Tobj*, bool & *converge*) `[virtual]`

Analytic Schilling inverse kinematics.

converge will be false if the desired end effector pose is outside robot range.

Implements Robot_basic.

Definition at line 893 of file invkine.cpp.

References Robot_basic::a, Link::a, C(), Link::d, Robot_basic::get_q(), K, Robot_-basic::links, and M_PI.

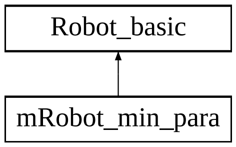
4.12.2.12 void mRobot::del

References Robot_basic::a, Robot_basic::da, Robot_basic::df, Robot_basic::dF,

4.13 mRobot_min_para Class Reference

`#include <robot.h>`

Inheritance diagram for mRobot_min_para::



4.13.1 Detailed Description

Modified DH notation and minimal inertial parameters robot class.

Definition at line 437 of file robot.h.

Public Member Functions

- mRobot_min_para

- virtual ReturnMatrix

4.13.2.9 void mRobot_min_para::dTdqi (Matrix & *dRot*, ColumnVector & *dp*, const int *i*) [virtual]

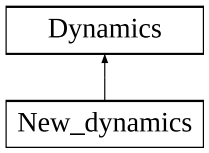
Partial derivative of the robot position (homogeneous transf.).

This function computes the partial derivatives:

$$\frac{\partial {}^0T_n}{\partial q}$$

4.14 New_dynamics Class Reference

Inheritance diagram for New_dynamics::



4.14.1 Detailed Description

4.14.3.5 int New_dynamics::i

Temporary index.

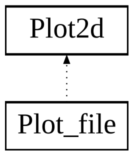
Definition at line 73 of file demo_2dof_pd.cpp.

Referenced by New_dynamics(), and plot().

4.15 Plot2d Class Reference

`#include <gnugraph.h>`

Inheritance diagram for Plot2d::



- Plot2d

  Constructor.

- void dump (void)

  Method to dump the content of Plot2d to stdout.

- void

4.15.1 Detailed Description

2d plot object.

Definition at line 149 of file gnugraph.h.

4.18 Proportional_Derivative Class Reference

```
#include <controller.h>
```

4.18.1 Detailed Description

Proportional derivative controller class.

The driving torques can be expressed as

- ColumnVector

Definition at line 674 of file controller.cpp.

References dof, Kp, and WRONG_SIZE.

Referenced by Proportional_Derivative().

4.19 Quaternion Class Reference

Quaternion

- ReturnMatrix T () const

  Transformation matrix from a quaternion.

Private Attributes

- Real

4.19.3 Member Function Documentation

4.19.3.1 Quaternion Quaternion::operator+ (const Quaternion & *rhs*) const

Overload + operator.

The quaternion addition is

$$q_1 + q_2 = [s_1, v_1] + [s_2, v_2] = [s_1 + s_2, v_1 + v_2]$$

The result is not necessarily a unit quaternion even if q_1 and q_2 are unit quaternions.

Definition at line 203 of file quaternion.cpp.

Referentio8_s002le

The conjugate of a quaternion $q = [s, v]$ is $q \quad = [s, -v]$

Definition at line 283 of file quaternion.cpp.

References s_, and v_.

Referenced by i().

4.19.3.5 Quaternion Quaternion::exp () const

Exponential of a quaternion.

Let a quaternion of the form $q = [0, \theta v]$, q is not necessarily a unit quaternion. Then the exponential function is defined by $q = [\cos(\theta), v \sin(\theta)]$.

Definition at line 336 of file quaternion.cpp.

References EPSILON,0-17.6244Td[(Let)-292(a)-292(qn-17.6243Td[(Referenced)-250po)25(werby)-250(i().)]TJ/F289.9626Tf

$$\dot{v} = \frac{1}{.} E(s, v) w_b$$

$$E = sI + S(v)$$

Definition at line 388 of file quaternion.cpp.

References s_, sign(), and v_.

Referenced by Impedance::control().

4.19.3.8 ReturnMatrix Quaternion::E (const short *sign*) const

Matrix E.

See Quaternion::dot for explanation.

Definition at line 426 of file quaternion.cpp.

References BODY_FRAME, sign(), threebythreeident, and x_prod_matrix().

Referenced by Impedance::control(), and Omega().

4.19.3.9 Real Quaternion::norm () const

Return the quaternion norm.

The norm of quaternion is defined by

$$N(q) = s^{\cdot} + v \cdot v$$

Definition at line 298 of file quaternion.cpp.

References s_, and v_.

Referenced by i(), and unit().

4.19.3.10 Real Quaternion::24_prod (const Quaternion & *q*) const

Quaternion dot product.

The dot product of quaternion is defined by

$$q_1 \cdot q_. = s_1 s_. + v_1 \cdot v_.$$

Definition at line 445 of file quaternion.cpp.

Referenced by Impedance::control(), and Resolved_acc::torque_cmd().

4.20 Resolved_acc Class Reference

```
#include <controller.h>
```

4.20.1 Detailed Description

Assign the gain k_{vo}.

-

Vector part of error quaternion.

- Quaternion quat

  Temporary quaternion.

Tcol[(56Tf5211.8715311.955 ColumnVec53r &pdpp

4.20.2 Member Function Documentation

const ColumnVector &pdpp

4.20.2.1 ReturnMatrix Resolved_acc::torque_cmd (Robot_basic & *robot*, const ColumnVector & *pdpp*

- ReturnMatrix inv_kin (const Matrix &Tobj, const int mj=0)

- virtual void delta_torque

4.21.2.2 void Robot::kine_pd (Matrix & *Rot*, ColumnVector & *pos*, ColumnVector & *pos_dot*, const int *j*) const `[virtual]`

Direct kinematics with velocity.

Parameters:

Rot,: Frame j rotation matrix w.r.t to the base frame.

4.21.2.5 ReturnMatrix Robot::inv_kin_puma (const Matrix & *Tobj*, bool & *converge*) `[virtual]`

Analytic Puma inverse kinematics.1063d-1762644Td[(con)40(v)15(er)18gsealrsefectore

Implements Robot_basic.

Definition at line 352 of file kinemat.cpp.

4.21.2.8 ReturnMatrix Robot::jacobian_dot (const int *ref* = 0) const [virtual]

+acobian derivative of mobile joints expressed at frame ref.

T_e +acobian derivative expressed in based frame is

$$^0\dot{+}(q, \dot{q}) =$$

for a revolute joint and

□

00000000000100003

□

Q_i = □ 7

□ 7

5 fraisic oin.

dRotate and are modied on output.

Implements

$\dot{v}_i = R_i^T \{\dot{v}_i$

$\delta\dot{\omega}_i \ = R_i^T\{\delta\dot{\omega}_{i-1}$

4.21.2.14 voi(P)305(Robot::dqp_tor)18(queP)305((constP)305(ColumnV)100(ector)&

4.22 Robot_basic Class Refen56.:ence

```
#include <
```

Return the joint acceleration vector of available (non-immobile) joints up to and including endlink.

- void set_q (const ColumnVector &q)

  Set the joint position vector.

- void set_q (const Matrix &q)

  Set the joint position vector.

- void set_q (const Real q, const int i)

- virtual ReturnMatrix **inv_kin_rhino** (const Matrix &Tobj, bool &converge)=0
- virtual ReturnMatrix **inv_kin_puma** (const Matrix &Tobj, bool &converge)=0
- virtual ReturnMatrix **inv_kin_schilling** (const Matrix &Tobj, bool &converge)=0
- virtual ReturnMatrix jacobian (const int ref=0) const

  Jacobian of mobile links expressed at frame ref.

- virtual ReturnMatrix **jacobian** (const int endlink, const int ref) const=0
- virtual ReturnMatrix **jacobian_dot** (const int ref=0) const=0
- ReturnMatrix jacobian_DLS_inv (const double eps, const double lambda_max, const int ref=0) const

  Inverse Jacobian based on damped least squares inverse.

- virtual void **dTdqi** (Matrix &dRot, ColumnVector &dp, const int i)=0
- virtual ReturnMatrix **dTdqi** (const int i)=0
- ReturnMatrix acceleration (const ColumnVector &q, const ColumnVector &qp, const ColumnVector &tau)

  Joint0g0G/F229tion contact force.

- ReturnMatrix acceleration (const ColumnVector &q, const ColumnVector &qp, const ColumnVector &tau, const ColumnVector &Fext, const ColumnVector &Next)

  Joint0g0G/F229tion.

- ReturnMatrix inertia (const ColumnVector &q)

  Inertia of the manipulator.

- virtual ReturnMatrix **torque_novelocity** (const ColumnVector &qpp)=0
- virtual ReturnMatrix **torque** (const ColumnVector &q, const ColumnVector &qp, const qpp)=0

- ReturnMatrix dtau_dq (const ColumnVector &q, const ColumnVector &qp, const ColumnVector &qpp)

4.22.2 Member Enumeration Documentation

4.22.2.1 enum Robot_basic::EnumRobotType `[private]`

enum EnumRobotType

Enumerator:

HINO:

dh_parameter,: true if DH notation, false if modified DH notation.

min_inertial_para,: true inertial parameter are in minimal form.

Allocate memory for vectors and matrix pointers. Initialize all the Links instance.

Definition at line 343 of file robot.cpp.

References a, cleanUpPointers(), da, dF, df, dN, dn, dof, dp, dvp, dw, dwp, F, f, f_nv, fix, GRAVITY, gravity, links, N, n, n_nv, p, pp, R, threebythreeident, vp, w, wp, and z0.

4.22.3.3 Robot_basic::Robot_basic (const Matrix & *initrobot*, const Matrix & *initmotor*, const bool

Definition at line 92 of file kinemat.cpp.

Referenced by Clik::endeff_pos_ori_err(), Impedance::Impedance(), kinematics_-demo(), and main().

4.22.4.6 void Robot_basic::kine (Matrix & *Rot*, ColumnVector & *pos*, const int *j*) const

Direct kinematics at end effector.

Parameters:

- ***Rot,:*** Frame j orientation.
- ***pos,:*** Frame j position.
- ***j,:*** Selected frame.

Definition at line 102 of file kinemat.cpp.

4.22.4.7 ReturnMatrix Robot_basic::kine_pd (const int *j* = 0) const

Direct kinematics with velocity.

Return a 3×5 matrix. The first three columns are the frame j to the base rotation, the fourth column is the frame j w.r.t to the base postion vector and the last column is the

then the joint acceleration is

$$\ddot{q} = B^{-1}(q) \tau - J^{T}(q) f -$$

4.23 Spl_cubic Class Reference

```
#include <trajectory.h
```

- double final_time

  Spline final time.

4.25 Spl_Quaternion Class Reference

```
#include <trajectory.h>
```

4.25.1 Detailed Description

Cubic quaternions spline.

Definition at line 147 of file trajectory.h.

Public Member Functions

- Spl_Quaternion ()
- **Spl_Quaternion** (const std::string &filename)
- .-194.8687-471.313cmBT/F2a65Spl_Quaternion **(**const(&filename))]TJ.96094.8456.3698687471.313cm0g16626650014.71.313cm0g040cm52T456.3698

4.26 Stewart Class Reference

```
#include <stewart.h>
```

4.26.1 Detailed Description

Stewart definitions.

Definition at line 143 of file stewart.h.

Public Member Functions

- Stewart ()

  Default Constructor.

- Stewart (const Matrix InitPlat, bool Joint=true)

  Constructor.

Set the inertia matrix of the platform.

•

Return the first intermediate jacobian matrix (reverse) of the platform.

- ReturnMatrix Find_InvJacob2 ()

  Return the second intermediate jacobian matrix (reverse) of the platform.

- ReturnMatrix jacobian_dot ()

  Return time deriative of the inverse jacobian matrix of the platform.

- ReturnMatrix Find_dl ()

Platform speed.

- ColumnVector ddq

  Platform acceleration.

- ColumnVector pR

  Platform center of mass (in its own referential).

- ColumnVector gravity

  Gravity vector.

- Matrix pIp

  Platform Inertia (local ref.).

- Real mp

  Platform mass.

- Real p

  Pitch of the ballscrew (links).

- Real n

  Gear ratio (links motor).

- Real Js

  Moment of inertia (ballscrew).

- Real Jm

  Moment of inertia (motor).

- Real bs

  Viscous damping coefficient of the ballscrew.

4.26.3.5 ReturnMatrix Stewart::Find_Alpha ()

- m_p is the mass of the platform.
- G is the gravity.

4.26.3.14 ReturnMatrix Stewart::JointSpaceForceVct (const Real *Gravity* = `GRAVITY`)

4.26.3.19 ReturnMatrix Stewart::ForwardDyn (const ColumnVector *T*, const Real *Gravity* = `GRAVITY`)

Return the acceleration vector of the platform (ddq).h(867-)]TJET101-504.3786-686.10T6711. (732996.732996.-

- $I_{6\times6}$ is the Identity matrix.
- J_s is the mass moment of inertia of the ballscrew.
- J_m is the mass moment of inertia of the motor.

4.27 Trajectory_Select Class Reference

```
#include <trajectory.h>
```

4.27.1 Detailed Description

Trajectory class selection.

Definition at line 164 of file trajectory.h.

Public Member Functions

- Trajectory_Select ()

  Constructor.

Chapter 5

ROBOOP, A Robotics Object Oriented Package in C++ File Documentation

5.1 bench.cpp File Reference

5.1.1 Detailed Description

A benchmark le.

Prints the time, on the console, to perform certain operations.

De nition in le bench.cpp

5.1.2.3 Real Stewart_ddq[ ]

Initial value:

```
{-10.0, -10.0, -10, -10.0, -10, -10}
```

Definition at line 92 of file bench.cpp.

Referenced by stewartmain().

5.1.2.4 Real Stewart_dq[ ]

Initial value:

```
{0.2, 0.3, -0.4, 0.1, -1.4, 0.1}
```

Definition at line 90 of file bench.cpp.

Referenced by stewartmain().

5.1.2.5 Real Stewart_Ini[ ]

Initial value:

```
{1.758, 2.8, -1.015, 0.225, 0.0, -0.228, 3.358, 0.05, 4.237, 0.1406, 10, 12.5, 0.5, 0.35, 0.0, 0.0, 0.0
 1.6021, 3.07, -0.925, 0.1125, 0.1949, -0.228, 3.358, 0.05, 4.237, 0.1406, 10, 12.5, 0.5, 0.35, 0.0, 0.
 -1.7580, 2.8, -1.015, -0.1125, 0.1949, -0.228, 3.358, 0.05, 4.237, 0.1406, 10, 12.5, 0.5, 0.35, 0.0,
 -1.6021, 3.07, -0.925, -0.225, 0.0, -0.228, 3.358, 0.05, 4.237, 0.1406, 10, 12.5, 0.5, 0.35, 0.0, 0.0,
 0.0, 2.8, 2.03, -0.1125, -0.1949, -0.228, 3.358, 0.05, 4.237, 0.1406, 10, 12.5, 0.5, 0.35, 0.0, 0.0,
 0.0, 3.07, 1.85, 0.1125, -0.1949, -0.228, 3.358, 0.05, 4.237, 0.1406, 10, 12.5, 0.5, 0.35, 0.0, 0.0,
 0.0, 0.0, -0.114, 1.001, 0.59, 0.843, 10, 0.12, 0.04, 0.5, 0.5, 0.5, 1.5, 0.5, 0.0 4.25.44, 0.443}
```

Definition at line 75 of file bench.cpp.

Referenced by stewartmain().

5.1.2.6 Real Stewart_l[ ]

Initial value:

5.1.2.7 Real Stewart_q

5.3 clik.h File Reference

5.3.1 Detailed Description

Header file for Clik class definitions.

Definition in file clik.h.

```
#include "robot.h"
```

Classes

-

5.4 comp_dq.cpp File Reference

5.4.1 Detailed Description

Delta torque (linearized dynamics).

Definition in file

Variables

5.8 control_select.cpp File Reference

5.8.1 Detailed Description

5.9 control_select.h File Reference

5.9.1 Detailed Description

Header file for Control_Select class definitions.

Definition in file control_select.h.

```
#include <string>
#include "controller.h"7086.(#include)-600("controller.
```

5.14 demo_2dof_pd.cpp File Reference

5.14.1 Detailed Description

A demo file.

This demo file shows a two degree of freedom robots controller by a pd controller. The robot is define by the file "conf/rr_dh.conf", while the controller is defined by the file "conf/pd_2dof.conf". The desired joint trajectory is defined by the file "conf/q_-2dof.dat";

Definition in file demo_2dof_pd.cpp.

```
#include "gnugraph.h"
#include "controller.h"
#include "control_select.h"
#include "dynamics_sim.h"
#include "robot.h"
#include "trajectory.h"
```

Classes

-

  This is an example of customize Dynamics class.

5.16 dynamics_sim.cpp File Reference

5.16.1 Detailed Description

Basic dynamics simulation class.

Definition in file dynamics_sim.cpp.

```
#include "dynamics_sim.h"
#include "robot.h"
```

Variables

- static const char

5.17 dynamics_sim.h File Reference

5.17.1 Detailed Description

Header file for

- #define WANT_STRING
-

Variables

5.21 invkine.cpp File Reference

5.21.1 Detailed Description

Inverse kinematics solutions.

Definition in file invkine.cpp.

```
#include <stdexcept>
#include "robot.h"
```

Defines

•

•

Definition at line 725 of file quaternion.cpp.

References Slerp().

Referenced by Spl_Quaternion::quat(), and Spl_Quaternion::quat_w().

5.23.2.7

5.24 quaternion.h File Reference

5.24.1 Detailed Description

Trapezoidal quaternion scalar part integration.

- ReturnMatrix

$$c_1 = \frac{\sin(t\theta)}{}$$

5.24.2.6

$$Squad\,(p, a, b, q, t) = U^{d}$$

5.25 robot.cpp File Reference

Compare the robot DH table with the Puma DH table. The function return true if the tables are similar (same alpha and similar a and d parameters).

Definition at line 1615 of file robot.cpp.

References Link::get_a(), Link::get_alpha(), Link::get_d(), Robot_basic::get_dof(), Link::get_joint_type(), isZero(), Robot_basic::links, and robot.

Referenced by mRobot_min_para::robotType_inv_kin(), and mRobot::robotType_-inv_kin().

5.25.2.4 bool Rhino_DH (const Robot_basic & *robot*)

Return true if the robot is like a Rhino on DH notation.

Compare the robot DH table with the Puma DH table. The function return true if the tables are similar (same alpha and similar a and d parameters).

Definition at line 1483 of file robot.cpp.

References Link::get_a(), Link::get_alpha(), Link::get_d(), Robot_basic::get_dof(), Link::get_joint_type(), isZero(), Robot_basic::links, and robot.

Referenced by Robot::robotType_inv_kin().

5.25.2.5 bool Rhino_mDH (const Robot_basic & *robot*)

Return true if the robot is like a Rhino on modified DH notation.

Compare the robot DH table with the Puma DH table. The function return true if the tables are similar (same alpha and similar a and d parameters).

Definition at line 1583 of file robot.cpp.

References Link::get_a(), Link::get_alpha(), Link::get_d(), Robot_basic::get_dof(), Link::get_joint_type(), isZero(), Robot_basic::links, and robot.

Referenced by mRobot_min_para::robotType_inv_kin(), and mRobot::robotType_-inv_kin().

5.25.2.6 bool Schilling_DH (const Robot_basic & *robot*)

Return true if the robot is like a Schilling on DH notation.

Compare the robot DH table with the Schilling DH table. The function return true if the tables are similar (same alpha and similar a and d parameters).

Definition at line 1549 of file robot.cpp.

References Link::get_a(), Link::get_alpha(), Link::get_d(), Robot_basic::get_dof(), Link::get_joint_type(), isZero(), Robot_basic::links, and robot.

Referenced by Robot::robotType_inv_kin().

5.25.2.7 bool Schilling_mDH (const

5.26 robot.h File Reference

5.26.1 Detailed Description

Robots class definitions.

Definition in file robot.h.

```
#include "utils.h"
```

Classes

5.28 sensitiv.cpp File Reference

5.28.1 Detailed Description

Delta torque (linearized dynamics).

Definition in file

Stewart class definitions.

Definition in file stewart.h

•

5.31 trajectory.cpp File Reference

5.31.1 Detailed Description

Definition in file

trajectory.cpp

5.32 trajectory.h File Reference

5.32.1 Detailed Description

Header file for trajectory generation class.

Definition in file trajectory.h

5.33 utils.cpp File Reference

5.33.1 Detailed Description

Utility functions.

Definition in file utils.cpp.

```
#include "utils.h"
```

Defines

- #define PGROW -0.20
- #define PSHRNK

- void

- ReturnMatrix irotk (const Matrix &R)

    Obtain axis from a rotation matrix.

-