Lattice.query

Find a path between two poses

P.**query**(**start**, **goal**) finds a path ($N \times 3$) from pose **start** (1×3) to pose **goal** (1×3). The pose is expressed as [X,Y,THETA].

Link

manipulator Link class

A Link object holds all information related to a robot joint and link such as kinematics parameters, rigid-body inertial parameters, motor and transmission parameters.

Constructors

Link	general constructor
Prismatic	construct a prismatic joint+link using standard DH
PrismaticMDH	construct a prismatic joint+link using modified DH
Revolute	construct a revolute joint+link using standard DH
RevoluteMDH	construct a revolute joint+link using modified DH

Information/display methods

display	print the link parameters in human readable form
dyn	display link dynamic parameters
type	joint type: 'R' or 'P'

Conversion methods

char	convert to string

Operation methods

A	link transform matrix
friction	friction force
nofriction	Link object with friction parameters set to zero%

Testing methods

islimit	test if joint exceeds soft limit
isrevolute	test if joint is revolute
isprismatic	test if joint is prismatic
issym	test if joint+link has symbolic parameters

Overloaded operators

+	concatenate links, result is a SerialLink object

Properties (read/write)

theta	kinematic: joint angle
d	kinematic: link offset
a	kinematic: link length
alpha	kinematic: link twist
jointtype	kinematic: 'R' if revolute, 'P' if prismatic
mdh	kinematic: 0 if standard D&H, else 1
offset	kinematic: joint variable offset
qlim	kinematic: joint variable limits [min max]
m	dynamic: link mass
r	dynamic: link COG wrt link coordinate frame 3×1
I	dynamic: link inertia matrix, symmetric 3×3, about link COG.
B	dynamic: link viscous friction (motor referred)
Tc	dynamic: link Coulomb friction
G	actuator: gear ratio
Jm	actuator: motor inertia (motor referred)

Examples

```
L = Link([0 1.2 0.3 pi/2]);
L = Link('revolute', 'd', 1.2, 'a', 0.3, 'alpha', pi/2);
L = Revolute('d', 1.2, 'a', 0.3, 'alpha', pi/2);
```

Notes

- This is a reference class object.
- Link objects can be used in vectors and arrays.
- Convenience subclasses are Revolute, Prismatic, RevoluteMDH and PrismaticMDH.

References

- Robotics, Vision & Control, P. Corke, Springer 2011, Chap 7.

See also

Link, Revolute, Prismatic, SerialLink, RevoluteMDH, PrismaticMDH

Link.Link

Create robot link object

This the class constructor which has several call signatures.

L = **Link**() is a **Link** object with default parameters.

L = **Link**(**lnk**) is a **Link** object that is a deep copy of the link object **lnk** and has type **Link**, even if **lnk** is a subclass.

L = **Link**(**options**) is a link object with the kinematic and dynamic parameters specified by the key/value pairs.

Options

'theta', TH	joint angle, if not specified joint is revolute
'd', D	joint extension, if not specified joint is prismatic
'a', A	joint offset (default 0)
'alpha', A	joint twist (default 0)
'standard'	defined using standard D&H parameters (default).
'modified'	defined using modified D&H parameters.
'offset', O	joint variable offset (default 0)
'qlim', **L**	joint limit (default [])
'I', I	link inertia matrix (3×1, 6×1 or 3×3)
'r', R	link centre of gravity (3×1)
'm', M	link mass (1×1)
'G', G	motor gear ratio (default 1)
'B', B	joint friction, motor referenced (default 0)
'Jm', J	motor inertia, motor referenced (default 0)
'Tc', T	Coulomb friction, motor referenced (1×1 or 2×1), (default [0 0])
'revolute'	for a revolute joint (default)
'prismatic'	for a prismatic joint 'p'
'standard'	for standard D&H parameters (default).
'modified'	for modified D&H parameters.
'sym'	consider all parameter values as symbolic not numeric

Notes

- It is an error to specify both 'theta' and 'd'
- The joint variable, either theta or d, is provided as an argument to the A() method.
- The link inertia matrix (3×3) is symmetric and can be specified by giving a 3×3 matrix, the diagonal elements [Ixx Iyy Izz], or the moments and products of inertia [Ixx Iyy Izz Ixy Iyz Ixz].
- All friction quantities are referenced to the motor not the load.
- Gear ratio is used only to convert motor referenced quantities such as friction and interia to the link frame.

Old syntax

L = **Link**(**dh**, **options**) is a link object using the specified kinematic convention and with parameters:

- **dh** = [THETA D A ALPHA SIGMA OFFSET] where SIGMA=0 for a revolute and 1 for a prismatic joint; and OFFSET is a constant displacement between the user joint variable and the value used by the kinematic model.
- **dh** = [THETA D A ALPHA SIGMA] where OFFSET is zero.
- **dh** = [THETA D A ALPHA], joint is assumed revolute and OFFSET is zero.

Options

'standard'	for standard D&H parameters (default).
'modified'	for modified D&H parameters.
'revolute'	for a revolute joint, can be abbreviated to 'r' (default)
'prismatic'	for a prismatic joint, can be abbreviated to 'p'

Notes

- The parameter D is unused in a revolute joint, it is simply a placeholder in the vector and the value given is ignored.
- The parameter THETA is unused in a prismatic joint, it is simply a placeholder in the vector and the value given is ignored.

Examples

A standard Denavit-Hartenberg link

```
L3 = Link('d', 0.15005, 'a', 0.0203, 'alpha', -pi/2);
```

since 'theta' is not specified the joint is assumed to be revolute, and since the kinematic convention is not specified it is assumed 'standard'.

Using the old syntax

```
L3 = Link([ 0, 0.15005, 0.0203, -pi/2], 'standard');
```

the flag 'standard' is not strictly necessary but adds clarity. Only 4 parameters are specified so sigma is assumed to be zero, ie. the joint is revolute.

```
L3 = Link([ 0, 0.15005, 0.0203, -pi/2, 0], 'standard');
```

the flag 'standard' is not strictly necessary but adds clarity. 5 parameters are specified and sigma is set to zero, ie. the joint is revolute.

```
L3 = Link([ 0, 0.15005, 0.0203, -pi/2, 1], 'standard');
```

the flag 'standard' is not strictly necessary but adds clarity. 5 parameters are specified and sigma is set to one, ie. the joint is prismatic.

For a modified Denavit-Hartenberg revolute joint

```
L3 = Link([ 0, 0.15005, 0.0203, -pi/2, 0], 'modified');
```

Notes

- Link object is a reference object, a subclass of Handle object.
- Link objects can be used in vectors and arrays.
- The joint offset is a constant added to the joint angle variable before forward kinematics and subtracted after inverse kinematics. It is useful if you want the robot to adopt a 'sensible' pose for zero joint angle configuration.
- The link dynamic (inertial and motor) parameters are all set to zero. These must be set by explicitly assigning the object properties: m, r, I, Jm, B, Tc.
- The gear ratio is set to 1 by default, meaning that motor friction and inertia will be considered if they are non-zero.

See also

Revolute, Prismatic, RevoluteMDH, PrismaticMDH

Link.A

Link transform matrix

T = L.A(**q**) is an SE3 object representing the transformation between link frames when the link variable **q** which is either the Denavit-Hartenberg parameter THETA (revolute) or D (prismatic). For:

- standard DH parameters, this is from the previous frame to the current.
- modified DH parameters, this is from the current frame to the next.

Notes

- For a revolute joint the THETA parameter of the link is ignored, and **q** used instead.
- For a prismatic joint the D parameter of the link is ignored, and **q** used instead.
- The link offset parameter is added to **q** before computation of the transformation matrix.

See also

SerialLink.fkine

Link.char

Convert to string

s = L.**char**() is a string showing link parameters in a compact single line format. If L is a vector of **Link** objects return a string with one line per **Link**.

See also

Link.display

Link.display

Display parameters

L.**display**() displays the link parameters in compact single line format. If L is a vector of **Link** objects displays one line per element.

Notes

- This method is invoked implicitly at the command line when the result of an expression is a Link object and the command has no trailing semicolon.

See also

Link.char, Link.dyn, SerialLink.showlink

Link.dyn

Show inertial properties of link

L.**dyn**() displays the inertial properties of the link object in a multi-line format. The properties shown are mass, centre of mass, inertia, friction, gear ratio and motor properties.

If L is a vector of **Link** objects show properties for each link.

See also

SerialLink.dyn

Link.friction

Joint friction force

f = L.**friction**(**qd**) is the joint **friction** force/torque ($1 \times N$) for joint velocity **qd** ($1 \times N$). The **friction** model includes:

- Viscous **friction** which is a linear function of velocity.
- Coulomb **friction** which is proportional to sign(**qd**).

Notes

- The **friction** value should be added to the motor output torque, it has a negative value when **qd**>0.
- The returned **friction** value is referred to the output of the gearbox.
- The **friction** parameters in the Link object are referred to the motor.
- Motor viscous **friction** is scaled up by G^2.
- Motor Coulomb **friction** is scaled up by G.
- The appropriate Coulomb **friction** value to use in the non-symmetric case depends on the sign of the joint velocity, not the motor velocity.
- The absolute value of the gear ratio is used. Negative gear ratios are tricky: the Puma560 has negative gear ratio for joints 1 and 3.

See also

Link.nofriction

Link.horzcat

Concatenate link objects

[L1 L2] is a vector that contains deep copies of the **Link** class objects L1 and L2.

Notes

- The elements of the vector are all of type Link.
- If the elements were of a subclass type they are convered to type Link.
- Extends to arbitrary number of objects in list.

See also

Link.plus

Link.islimit

Test joint limits

L.**islimit**(**q**) is true (1) if **q** is outside the soft limits set for this joint.

Note

- The limits are not currently used by any Toolbox functions.

Link.isprismatic

Test if joint is prismatic

L.**isprismatic**() is true (1) if joint is prismatic.

See also

Link.isrevolute

Link.isrevolute

Test if joint is revolute

L.**isrevolute**() is true (1) if joint is revolute.

See also

Link.isprismatic

Link.issym

Check if link is a symbolic model

res = L.**issym**() is true if the **Link** L has any symbolic parameters.

See also

Link.sym

Link.nofriction

Remove friction

ln = L.**nofriction**() is a link object with the same parameters as L except nonlinear (Coulomb) friction parameter is zero.

ln = L.**nofriction**('all') as above except that viscous and Coulomb friction are set to zero.

ln = L.**nofriction**('coulomb') as above except that Coulomb friction is set to zero.

ln = L.**nofriction**('viscous') as above except that viscous friction is set to zero.

Notes

- Forward dynamic simulation can be very slow with finite Coulomb friction.

See also

Link.friction, SerialLink.nofriction, SerialLink.fdyn

Link.plus

Concatenate link objects into a robot

L1+L2 is a SerialLink object formed from deep copies of the **Link** class objects L1 and L2.

Notes

- The elements can belong to any of the Link subclasses.
- Extends to arbitrary number of objects, eg. L1+L2+L3+L4.

See also

SerialLink, SerialLink.plus, Link.horzcat

Link.set.I

Set link inertia

L.I = [Ixx Iyy Izz] sets link inertia to a diagonal matrix.

L.I = [Ixx Iyy Izz Ixy Iyz Ixz] sets link inertia to a symmetric matrix with specified inertia and product of intertia elements.

L.I = M set **Link** inertia matrix to M (3×3) which must be symmetric.

Link.set.r

Set centre of gravity

L.r = R sets the link centre of gravity (COG) to R (3-vector).

Link.set.Tc

Set Coulomb friction

L.Tc = F sets Coulomb friction parameters to [F -F], for a symmetric Coulomb friction model.

L.Tc = [FP FM] sets Coulomb friction to [FP FM], for an asymmetric Coulomb friction model. FP>0 and FM<0. FP is applied for a positive joint velocity and FM for a negative joint velocity.

Notes

- The friction parameters are defined as being positive for a positive joint velocity, the friction force computed by Link.friction uses the negative of the friction parameter, that is, the force opposing motion of the joint.

See also

Link.friction

Link.sym

Convert link parameters to symbolic type

LS = L.sym is a **Link** object in which all the parameters are symbolic ('sym') type.

See also

Link.issym

Link.type

Joint type

c = L.**type**() is a character 'R' or 'P' depending on whether joint is revolute or prismatic respectively. If L is a vector of **Link** objects return an array of characters in joint order.

See also

SerialLink.config

