ikpy Documentation

Release 1.0.0

Pierre Manceron

January 05, 2016

Contents

Here you can find the documentation of the Inverse Kinematics API.

If you search getting started guides and tutorials, go to the Github repository.

CHAPTER 1

ikpy package

1.1 ikpy.chain module

class ikpy.chain.**Chain**(*links*, *active_links=0*, *profile=''*, ***kwargs*)

Bases: object

The base Chain class

Parameters

- **links** (*list*) – List of the links of the chain
- **active_links** (*list*) – The list of the positions of the active links

forward_kinematics(*joints*, *full_kinematics=False*)

Returns the transformation matrix of the forward kinematics

Parameters

- **joints** (*list*) – The list of the positions of each joint. Note : Inactive joints must be in the list.
- **full_kinematics** (*bool*) – Return the transorfmation matrixes of each joint

Returns The transformation matrix

inverse_kinematic(*target*, *initial_position*, *first_active_joint=0*, ***kwargs*)

Computes the inverse kinematic on the specified target

Parameters

- **target** (*numpy.array*) – The target of the inverse kinematic, in meters
- **initial_position** (*numpy.array*) – the initial position of each joint of the chain
- **first_active_joint** (*int*) – The first active joint$

Returns The list of the positions of each joint according to the target. Note : Inactive joints are in the list.

plot(*joints*, *ax*, *target=None*, *show=False*)

Plots the Chain using Matplotlib

Parameters

- **joints** (*list*) – The list of the positions of each joint
- **ax** (*matplotlib.axes.Axes*) – A matplotlib axes

- **target** (*numpy.array*) – An optional target
- **show** (*bool*) – Display the axe. Defaults to False

classmethod from_urdf_file(*urdf_file*, *base_elements=['base_link']*, *last_link_vector=None*, *base_elements_type='joint'*)

Creates a chain from an URDF file

Parameters

- **urdf_file** (*string*) – The path of the URDF file
- **base_elements** (*list of strings*) – List of the links beginning the chain
- **last_link_vector** (*numpy.array*) – Optional : The translation vector of the tip.

ikpy.chain.**pinv**()

1.2 ikpy.link module

class ikpy.link.**Link**(*name*, *bounds=(None*, *None)*)

Bases: object

Base Link class.

Parameters

- **name** (*string*) – The name of the link
- **bounds** (*tuple*) – Optional : The bounds of the link. Defaults to None
- **use_symbolic_matrix** (*bool*) – wether the transformation matrix is stored as Numpy array or as a Sympy symbolic matrix.

get_transformation_matrix(*theta*)

class ikpy.link.**URDFLink**(*name*, *translation_vector*, *orientation*, *rotation*, *bounds=(None*, *None)*, *angle_representation='rpy'*, *use_symbolic_matrix=False*)

Bases: *ikpy.link.Link*

Link in URDF representation.

Parameters

- **name** (*string*) – The name of the link
- **bounds** (*tuple*) – Optional : The bounds of the link. Defaults to None
- **translation_vector** (*numpy.array*) – The translation vector. (In URDF, attribute "xyz" of the "origin" element)
- **orientation** (*numpy.array*) – The orientation of the link. (In URDF, attribute "rpy" of the "origin" element)
- **rotation** (*numpy.array*) – The rotation axis of the link. (In URDF, attribute "xyz" of the "axis" element)
- **angle_representation** (*string*) – Optionnal : The representation used by the angle. Currently supported representations : rpy. Defaults to rpy, the URDF standard.
- **use_symbolic_matrix** (*bool*) – wether the transformation matrix is stored as a Numpy array or as a Sympy symbolic matrix.

Returns The link object

Return type *URDFLink*

Example

URDFlink()

get_transformation_matrix(*theta*)

class ikpy.link.**DHLink**(*name*, *d=0*, *a=0*, *bounds=None*, *use_symbolic_matrix=True*)
Bases: *ikpy.link.Link*

Link in Denavit-Hartenberg representation.

Parameters

- **name** (*string*) – The name of the link
- **bounds** (*tuple*) – Optional : The bounds of the link. Defaults to None
- **d** (*float*) – offset along previous z to the common normal
- **a** (*float*) – offset along previous to the common normal
- **use_symbolic_matrix** (*bool*) – wether the transformation matrix is stored as Numpy array or as a Sympy symbolic matrix.

Returns The link object

Return type *DHLink*

get_transformation_matrix(*theta*, *a*)
Computes the homogeneous transformation matrix for this link.

class ikpy.link.**OriginLink**
Bases: *ikpy.link.Link*

The link at the origin of the robot

get_transformation_matrix(*theta*)

1.3 ikpy.inverse_kinematics module

ikpy.inverse_kinematics.**inverse_kinematic_optimization**(*chain*, *target*, *starting_nodes_angles*, *bounds=None*, *first_active_joint=0*, *regularization_parameter=None*, *max_iter=None*, ***kwargs*)
Computes the inverse kinematic on the specified target with an optimization method

1.4 ikpy.geometry_utils module

ikpy.geometry_utils.**Rx_matrix**(*theta*)
Rotation matrix around the X axis

ikpy.geometry_utils.**Rz_matrix**(*theta*)
Rotation matrix around the Z axis

ikpy.geometry_utils.**symbolic_Rz_matrix**(*symbolic_theta*)
Matrice symbolique de rotation autour de l'axe Z

ikpy.geometry_utils.**Ry_matrix**(*theta*)
Rotation matrix around the Y axis

ikpy.geometry_utils.**rotation_matrix**(*phi*, *theta*, *psi*)
Retourne la matrice de rotation décrite par les angles d'Euler donnés en paramètres

ikpy.geometry_utils.**symbolic_rotation_matrix**(*phi*, *theta*, *symbolic_psi*)
Retourne une matrice de rotation où psi est symbolique

ikpy.geometry_utils.**rpy_matrix**(*roll*, *pitch*, *yaw*)
Returns a rotation matrix described by the extrinsinc roll, pitch, yaw coordinates

ikpy.geometry_utils.**axis_rotation_matrix**(*axis*, *theta*)
Returns a rotation matrix around the given axis

ikpy.geometry_utils.**symbolic_axis_rotation_matrix**(*axis*, *symbolic_theta*)
Returns a rotation matrix around the given axis

ikpy.geometry_utils.**homogeneous_translation_matrix**(*trans_x*, *trans_y*, *trans_z*)
Returns a translation matrix the homogeneous space

ikpy.geometry_utils.**from_transformation_matrix**(*transformation_matrix*)
Converts a transformation matrix to a tuple (rotation_matrix, translation_vector)

ikpy.geometry_utils.**to_transformation_matrix**(*rotation_matrix*, *translation*)
Converts a tuple (rotation_matrix, translation_vector) to a transformation matrix

ikpy.geometry_utils.**cartesian_to_homogeneous**(*cartesian_matrix*, *matrix_type='numpy'*)
Converts a cartesian matrix to an homogenous matrix

ikpy.geometry_utils.**cartesian_to_homogeneous_vectors**(*cartesian_vector*, *matrix_type='numpy'*)
Converts a cartesian vector to an homogenous vector

ikpy.geometry_utils.**homogeneous_to_cartesian_vectors**(*homogeneous_vector*)
Converts a cartesian vector to an homogenous vector

ikpy.geometry_utils.**homogeneous_to_cartesian**(*homogeneous_matrix*)
Converts a cartesian vector to an homogenous matrix

1.5 ikpy.URDF_utils module

ikpy.URDF_utils.**find_next_joint**(*root*, *current_link*, *next_joints*)
Find the next joint in the URDF tree

ikpy.URDF_utils.**find_next_link**(*root*, *current_joint*, *next_links*)
Find the next link in the URDF tree

ikpy.URDF_utils.**find_parent_link**(*root*, *joint_name*)

ikpy.URDF_utils.**get_chain_from_joints**(*urdf_file*, *joints*)

ikpy.URDF_utils.**get_urdf_parameters**(*urdf_file*, *base_elements=['base_link']*, *last_link_vector=None*, *base_elements_type='joint'*)
Returns translated parameters from the given URDF file

Parameters

- **urdf_file** (*string*) – The path of the URDF file

- **base_elements** (*list of strings*) – List of the links beginning the chain
- **last_link_vector** (*numpy.array*) – Optional : The translation vector of the tip.

ikpy.URDF_utils.**get_motor_parameters**(*json_file*)
Returns a dictionnary with joints as keys, and a description (dict) of each joint as value

ikpy.URDF_utils.**convert_angle_to_pypot**(*angle*, *joint*, ***kwargs*)
Converts an angle to a PyPot-compatible format

ikpy.URDF_utils.**convert_angle_from_pypot**(*angle*, *joint*, ***kwargs*)
Converts an angle to a PyPot-compatible format

ikpy.URDF_utils.**convert_angle_limit**(*angle*, *joint*, ***kwargs*)
Converts the limit angle of the PyPot JSON file to the internal format

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

i

u

T

U