TUM Institute for Cognitive Systems (ICS)

OpenWalker

Module Description: Forward Kinematics (FKM)

Florian Bergner, Emmanuel Dean, Rogelio Guadarrama-Olvera,
Simon Armleder, and Gordon Cheng

February 14, 2020

1 Module Description

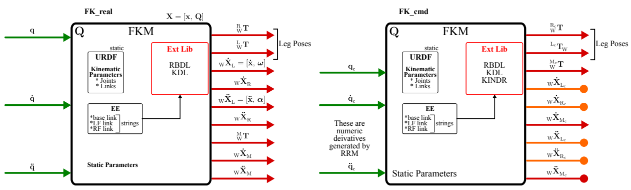

Figure 1.1: Forward Kinematics module: This module implements the forward kinematics for the robot.

The *Forward Kinematics* module (FKM) computes the forward kinematics of the robot. The OpenWalker framework employs two FKMs, one for computing the forward kinematics for the real robot, and one for the commanded robot. The forward kinematics maps the joint position space into the Cartesian space, i.e. for a given set of joint positions, the forward kinematics computes the Carthesian position (linear position and angular position) of a given end-effector. The forward kinematics of the real robot uses the robot's joint sensor measurements ($\mathbf{q}$, $\dot{\mathbf{q}}$, and $\ddot{\mathbf{q}}$) as input to compute the Carthesian positions of end-effectors of the real robot. Correspondingly, the forward kinematics of the commanded robot uses the commanded joint positions ($\mathbf{q}_c$, $\dot{\mathbf{q}}_c$, and $\ddot{\mathbf{q}}_c$) to compute the Carthesian positions of end-effectors of the commanded robot. The OpenWalker framework uses the real and commanded Carthesian end-effector positions to compute offsets which the frameworks uses to compensate the error between where it commanded the end-effectors and where they actually are.

Since the rigid multi body system (MBS) of the robot is the same for the real and the commanded robot the OpenWalker framework needs to realize only one FKM, which is then im-

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 732287.

plemented and connected once to the real joint positions and once to the commanded joint positions.

The OpenWalker framework requires the Carthesian position, velocities, and accelerations of three end-effectors, namely the left and right foot, and the center-of-mass (CoM) of the whole robot, with respect to the world. The FKM implementations provide this information to other modules of the OpenWalker project.

2 Module Connections

2.1 Inputs

Symbol	Name	Type	Description
$\mathbf{q} \in \mathbb{R}^{DOF}$	Real Robot Joint Positions	`JointPosition`	This vector contains the real joint positions of the robot. The OpenWalker framework uses this module input to compute the forward kinematics of the real robot.
$\dot{\mathbf{q}} \in \mathbb{R}^{DOF}$	Real Robot Joint Velocities	`JointVelocity`	This vector contains the real joint velocities of the robot. The OpenWalker framework uses this module input to compute the forward kinematics of the real robot.
$\ddot{\mathbf{q}} \in \mathbb{R}^{DOF}$	Real Robot Joint Accelerations	`JointAcceleration`	This vector contains the real joint accelerations of the robot. The OpenWalker framework uses this module input to compute the forward kinematics of the real robot.
$\mathbf{q}_c \in \mathbb{R}^{DOF}$	Commanded Robot Joint Positions	`JointPosition`	This vector contains the currently commanded joint positions of the robot. The OpenWalker framework uses this module input to compute the forward kinematics of the commanded robot.
$\dot{\mathbf{q}}_c \in \mathbb{R}^{DOF}$	Commanded Robot Joint Velocities	`JointVelocity`	This vector contains the currently commanded joint velocities of the robot. The OpenWalker framework uses this module input to compute the forward kinematics of the commanded robot.
$\ddot{\mathbf{q}}_c \in \mathbb{R}^{DOF}$	Commanded Robot Joint Accelerations	`JointAcceleration`	This vector contains the currently commanded joint accelerations of the robot. The OpenWalker framework uses this module input to compute the forward kinematics of the commanded robot.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 732287.

2.2 Outputs

FK of the Real Robot

Symbol	Name	Type	Description
${}^{L}_{W}\mathbf{T} \in \mathbb{R}^{4\times4}$	Left Foot Coordinate Frame	`HomogeneousTransformation`	This homogeneous transformation matrix transforms coordinates in the left foot coordinate frame L to the world coordinate frame W.
${}^{R}_{W}\mathbf{T} \in \mathbb{R}^{4\times4}$	Right Foot Coordinate Frame	`HomogeneousTransformation`	This homogeneous transformation matrix transforms coordinates in the right foot coordinate frame R to the world coordinate frame W.
${}^{M}_{W}\mathbf{T} \in \mathbb{R}^{4\times4}$	CoM Coordinate Frame	`HomogeneousTransformation`	This homogeneous transformation matrix transforms coordinates in the CoM coordinate frame M to the world coordinate frame W.
${}_{W}\dot{\mathbf{X}}_{L} \in \mathbb{R}^{6}$	Left Foot Velocity	`CartesianVelocity`	This vector contains the linear and angular velocities of the left foot L with respect to the world coordinate frame W.
${}_{W}\dot{\mathbf{X}}_{R} \in \mathbb{R}^{6}$	Right Foot Velocity	`CartesianVelocity`	This vector contains the linear and angular velocities of the right foot R with respect to the world coordinate frame W.
${}_{W}\dot{\mathbf{X}}_{M} \in \mathbb{R}^{6}$	CoM Velocity	`CartesianVelocity`	This vector contains the linear and angular velocities of the CoM with respect to the world coordinate frame W.
${}_{W}\ddot{\mathbf{X}}_{L} \in \mathbb{R}^{6}$	Left Foot Acceleration	`CartesianAcceleration`	This vector contains the linear and angular accelerations of the left foot L with respect to the world coordinate frame W.
${}_{W}\ddot{\mathbf{X}}_{R} \in \mathbb{R}^{6}$	Right Foot Acceleration	`CartesianAcceleration`	This vector contains the linear and angular accelerations of the right foot R with respect to the world coordinate frame W.
${}_{W}\ddot{\mathbf{X}}_{M} \in \mathbb{R}^{6}$	CoM Acceleration	`CartesianAcceleration`	This vector contains the linear and angular accelerations of the CoM with respect to the world coordinate frame W.

FK of the Commanded Robot

Symbol	Name	Type	Description
${}^{L_c}_{W}\mathbf{T} \in \mathbb{R}^{4\times4}$	Left Commanded Foot Coordinate Frame	`HomogeneousTransformation`	This homogeneous transformation matrix transforms coordinates in the left commanded foot coordinate frame L to the world coordinate frame W.
${}^{R_c}_{W}\mathbf{T} \in \mathbb{R}^{4\times4}$	Right Commanded Foot Coordinate Frame	`HomogeneousTransformation`	This homogeneous transformation matrix transforms coordinates in the right commanded foot coordinate frame R to the world coordinate frame W.
${}^{M_c}_{W}\mathbf{T} \in \mathbb{R}^{4\times4}$	CoM Commanded Coordinate Frame	`HomogeneousTransformation`	This homogeneous transformation matrix transforms coordinates in the CoM commanded coordinate frame M to the world coordinate frame W.
${}_{W}\dot{\mathbf{X}}_{L_c} \in \mathbb{R}^{6}$	Left Commanded Foot Velocity	`CartesianVelocity`	This vector contains the linear and angular velocities of the left commanded foot L with respect to the world coordinate frame W.
${}_{W}\dot{\mathbf{X}}_{R_c} \in \mathbb{R}^{6}$	Right Commanded Foot Velocity	`CartesianVelocity`	This vector contains the linear and angular velocities of the commanded right foot R with respect to the world coordinate frame W.
${}_{W}\dot{\mathbf{X}}_{M_c} \in \mathbb{R}^{6}$	CoM Commanded Velocity	`CartesianVelocity`	This vector contains the linear and angular velocities of the commanded CoM with respect to the world coordinate frame W.
${}_{W}\ddot{\mathbf{X}}_{L_c} \in \mathbb{R}^{6}$	Left Commanded Foot Acceleration	`CartesianAcceleration`	This vector contains the linear and angular accelerations of the commanded left foot L with respect to the world coordinate frame W.
${}_{W}\ddot{\mathbf{X}}_{R_c} \in \mathbb{R}^{6}$	Right Commanded Foot Acceleration	`CartesianAcceleration`	This vector contains the linear and angular accelerations of the commanded right foot R with respect to the world coordinate frame W.
${}_{W}\ddot{\mathbf{X}}_{M_c} \in \mathbb{R}^{6}$	CoM Commanded Acceleration	`CartesianAcceleration`	This vector contains the linear and angular accelerations of the commanded CoM with respect to the world coordinate frame W.

2.3 Inter-Connections

The inputs of the FKMs are connected to the outputs of the Real Robot Module (RRM) which provides the joint positions, velocities, and accelerations of the real and the commanded robot. The outputs of FKMs (real and commanded) are connected to modules that require the Cartesian positions, velocities, and accelerations of the left/right foot and the CoM end-effectors with respect to the world. The real FKM is connected to

- the Zero-Moment-Point Module (ZMPM),
- the Center-of-Mass Module (CoMM) for the real robot, and

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 732287.

- the Foot Compliance Model Module (FCMM).

The commanded FKM is connected to

- the Center-of-Mass Module (CoMM) for the commanded robot, and
- the Foot Trajectory Generator Module (FTGM).

The ZMPM module uses ${}^{\mathrm{L}}_{\mathrm{W}}\mathbf{T}$, ${}^{\mathrm{R}}_{\mathrm{W}}\mathbf{T}$, ${}^{\mathrm{M}}_{\mathrm{W}}\mathbf{T}$, and ${}_{\mathrm{W}}\dot{\mathbf{X}}_{\mathrm{M}}$ in combination with information of the foot FT sensors and the IMU to compute the linear position, and velocity, of the Zero-Moment-Point (ZMP). The CoMMs (real and commanded) fuse ${}^{\mathrm{M}}_{\mathrm{W}}\mathbf{T}$ and ${}_{\mathrm{W}}\dot{\mathbf{X}}_{\mathrm{M}}$ with IMU information in a model based filter to estimate the Cartesian position and velocity of the CoM and the linear position and velocity of the capture point (CP). The FCMM requires ${}^{\mathrm{L}}_{\mathrm{W}}\mathbf{T}$, ${}^{\mathrm{R}}_{\mathrm{W}}\mathbf{T}$, ${}_{\mathrm{W}}\dot{\mathbf{X}}_{\mathrm{L}}$, and ${}_{\mathrm{W}}\dot{\mathbf{X}}_{\mathrm{R}}$ compute the homogeneous transformation for the offset of the feet coordinate frames. The FTGM requires ${}^{\mathrm{L_c}}_{\mathrm{W}}\mathbf{T}$ and ${}^{\mathrm{R_c}}_{\mathrm{W}}\mathbf{T}$ to compute the reference Cartesian positions, velocities, and accelerations of the feet.

2.4 Common Methods

This module uses kinematic parameters such as joint properties (location, type), and link properties (location, length) to build up a rigid multi body system (MBS) that represents the kinematic model of the robot. The MBS is a tree of links and joints where the joints connect links. Relative spatial transformations between links and joints describe the spatial relation between parent and child links. Then the transformation of all coordinate frames within the MBS can be computed with respect to a reference coordinate frame by traveling along the branches of the tree and chaining up relative transformations between parent and child links. This recursive method of computing the transformations of link coordinate frames with respect to a reference coordinate frame has in contrast to the symbolic code generation method the advantage that existing models can be extended and more easily analyzed [1]. Furthermore, the recursive method does not require the complex generation of code from symbolic expressions.

References

[1] Martin L. Felis, RBDL: An efficient rigid-body dynamics library using recursive algorithms, Autonomous Robots 41 (2): 495–511, 2017.

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 732287.