SAPIENZA
UNIVERSITÀ DI ROMA

MASTER'S DEGREE IN CONTROL ENGINEERING

Model-based (sensorless) collision detection and localization

ROBOTICS 2

Professor:
De Luca Alessandro

Students:
Coletta Emanuele
Palamidessi Matteo
Rugiero Carlo

Academic Year 2024/2025

Abstract

The YuMi IRB 14000 is a dual arm robot manipulator produced by the company ABB, first released in 2015. This manipulator belongs to the first generation of collaborative robots, innovative robots designed to work in close contact with humans and share the workspace with them. In this context, collision handling is a critical aspect in guaranteeing human operator safety: robots must be equipped with control schemes able to prevent and react to possible collisions with the environment and the people therein.

This paper is dedicated to the analysis of the momentum based residual method for collision monitoring, a sensorless model-based algorithm for collision detection, isolation and identification, and to its implementation and simulation on the YuMi IRB 14050 robot (the 7 DoF single arm version of the IRB 14000).

The paper is organized as follows: in the first chapter, the theory of the residual method for a generic robot is presented. In the second chapter, the kinematic and dynamic model of the ABB YuMi are obtained through a CAD analysis using the software Autodesk Fusion 360 and through the software openSYMORO. In the third chapter, the algorithm is implemented on the YuMi and some MATLAB simulations are discussed. Finally, conclusions are presented in chapter four.

Contents

Chapter 1

The residual method for collision detection and localization

Physical human-robot interaction is a growing trend in industrial practice and user service, and handling the eventuality of a collision is one of the most crucial challenges to face.

In particular, if a collision takes place, solving the problems of collision detection, isolation and identification (respectively: to determine if a collision has happened or not; to localize the collided link and the contact point; to estimate the contact force), is a fundamental task to carry out in order to plan a reaction strategy in real time.

To this aim, several methods have been proposed in the dedicated literature [3], and the residual method based on the generalized momentum observer (from now addressed simply as "residual method" for compactness) proved to be the most successful one.

This is due to multiple reasons: first of all, the residual method requires no additional external sensors, since the only measured quantities processed are the state variables of the robot, measured by proprioceptive sensors at the joints. In second place, this algorithm doesn't require the inversion of the robot inertia matrix nor the estimation of the joint accelerations, hence limiting the negative effect of noise and the computational complexity. Finally, in the ideal case, this method leads to a set of n linear, decoupled and configuration independent equations for the residuals, simplifying the isolation and identification problems.

In this chapter, the theory of the momentum based residual method is presented, emphasizing its strengths and limits.

1.1 Residual dynamics

Consider a robot manipulator with rigid links and rigid joints described by a model of the form

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{c}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q}) + \boldsymbol{\tau}_{\mathbf{f}} = \boldsymbol{\tau}_{\mathbf{m}} + \boldsymbol{\tau}_{\mathbf{k}}, \tag{1.1}$$

where

- $\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}$ $(n \times 1)$ are, respectively, the joint positions, velocities and accelerations;
- $\mathbf{M}(\mathbf{q})$ $(n \times n)$ is the robot inertia matrix;
- $\mathbf{c}(\mathbf{q}, \dot{\mathbf{q}})$ $(n \times 1)$ is the vector of Coriolis and centrifugal terms;
- $\mathbf{g}(\mathbf{q})$ $(n \times 1)$ is the vector of gravity terms;
- $\boldsymbol{\tau}_\mathbf{f}$ $(n \times 1)$ is the vector of friction terms;
- $\boldsymbol{\tau}_\mathbf{m}$ $(n \times 1)$ is the vector of joint torques commanded by the motors;
- $\boldsymbol{\tau}_\mathbf{k}$ $(n \times 1)$ is the unknown vector of joint torques generated by the external collision forces.

The main idea behind the residual method is to introduce the $n \times 1$ vector variable $\mathbf{r}$, namely the residual vector, and implement it in such a way that its dynamics is governed by the equation

$$\dot{\mathbf{r}} = \mathbf{K_o}(\boldsymbol{\tau}_\mathbf{k} - \mathbf{r}), \tag{1.2}$$

where $\mathbf{K_o}$ is a $n \times n$ diagonal positive definite matrix:

$$\mathbf{K_o} = \mathrm{diag}\{\mathrm{K_{o,i}}\}. \tag{1.3}$$

The term $\boldsymbol{\tau}_\mathbf{k}$ is not known and it is therefore not available as it is for the implementation, but it can be computed through the robot state measures by means of (1.1), leading to

$$\dot{\mathbf{r}} = \mathbf{K_o}(\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{c}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q}) + \boldsymbol{\tau}_\mathbf{f} - \boldsymbol{\tau}_\mathbf{m} - \mathbf{r}). \tag{1.4}$$

Notice however that such equation depends on the joint accelerations, which are not in general available for measurements and whose estimation is usually quite noisy.

In order to remove such dependence, the generalized momentum of the robot is introduced, defined as

$$\mathbf{p} = \mathbf{M}(\mathbf{q})\dot{\mathbf{q}}. \tag{1.5}$$

Clearly, its derivative with respect to time is

$$\dot{\mathbf{p}} = \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \dot{\mathbf{M}}(\mathbf{q})\dot{\mathbf{q}}. \tag{1.6}$$

Substituting the term $\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}}$ inside the residual dynamics, one obtains

$$\dot{\mathbf{r}} = \mathbf{K_o}(\dot{\mathbf{p}} - \dot{\mathbf{M}}(\mathbf{q})\dot{\mathbf{q}} + \mathbf{c}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q}) + \boldsymbol{\tau}_\mathbf{f} - \boldsymbol{\tau}_\mathbf{m} - \mathbf{r}). \tag{1.7}$$

In order to simplify the notation, define

$$\boldsymbol{\beta}(\mathbf{q}, \dot{\mathbf{q}}) := \mathbf{c}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q}) + \boldsymbol{\tau}_\mathbf{f} - \dot{\mathbf{M}}(\mathbf{q})\dot{\mathbf{q}} \tag{1.8}$$

and

$$\hat{\dot{\mathbf{p}}} := \mathbf{r} - \boldsymbol{\beta}(\mathbf{q}, \dot{\mathbf{q}}) + \boldsymbol{\tau}_{\mathbf{m}}, \tag{1.9}$$

so that (1.7) re-writes as

$$\dot{\mathbf{r}} = \mathbf{K_o}(\dot{\mathbf{p}} + \boldsymbol{\beta}(\mathbf{q}, \dot{\mathbf{q}}) - \boldsymbol{\tau}_{\mathbf{m}} - \mathbf{r}) \tag{1.10}$$

or, even more compact, as

$$\dot{\mathbf{r}} = \mathbf{K_o}(\dot{\mathbf{p}} - \hat{\dot{\mathbf{p}}}). \tag{1.11}$$

This equation is however not yet implementable as it is since the term $\dot{\mathbf{p}}$ cannot be obtained only through the robot state measures as it happens instead for $\mathbf{p}$. To this aim, the integral operator between 0 and the current time t is applied to both sides of the equation, leading to the final version of the residual dynamics, which is the one actually implemented:

$$\mathbf{r} = \mathbf{K_o}(\mathbf{p} - \mathbf{p_0} - \hat{\mathbf{p}}), \tag{1.12}$$

where

$$\hat{\mathbf{p}} = \int_0^{\mathrm{t}} (\mathbf{r} - \boldsymbol{\beta}(\mathbf{q}, \dot{\mathbf{q}}) + \boldsymbol{\tau}_{\mathbf{m}})\mathrm{dt}. \tag{1.13}$$

Such notation for $\hat{\mathbf{p}}$ and $\hat{\dot{\mathbf{p}}}$ has been adopted because those terms are indeed an estimate, respectively, for the variation of the momentum between the current and the initial time $\mathbf{p} - \mathbf{p}_0$, and for its derivative $\dot{\mathbf{p}}$. The residual dynamics as a whole can be therefore thought as an observer for the robot generalized momentum, hence justifying its name.

Although the last implementation-oriented form of this dynamics appears to be non-linear, the true evolution of the residuals is governed by (1.2), which is an asymptotically stable linear and decoupled dynamics in the state variable $\mathbf{r}$ with input $\boldsymbol{\tau}_{\mathbf{k}}$. This entails that, while $\boldsymbol{\tau}_{\mathbf{k}} = \mathbf{0}$, the elements of $\mathbf{r}$ will converge exponentially to zero with time constants $T_{o,i} = \frac{1}{K_{o,i}}$, thus the bigger $K_{o,i}$, the smaller $T_{o,i}$ and the faster the response of r_i. The speed of response of the residual system is a crucial aspect in the implementation of the method: ideally, the robot should react in the shortest time possible in order to minimize human injury.

Furthermore, in the limit case of $\mathbf{K_o} \to \infty$, the dynamics of $\mathbf{r}$ becomes just $\mathbf{r} = \boldsymbol{\tau}_{\mathbf{k}}$, making the residuals a sort of virtual sensor for the joint torques generated by external forces.

Finally, the initial condition is usually set to

$$\mathbf{r}(0) = \mathbf{r_0} = \mathbf{0}, \tag{1.14}$$

so to have, at least theoretically, all the residuals be exactly zero until a collision occurs, avoiding any initial transient. In practice, noise and numerical approximations are unavoidable, thus the free evolution of the residuals is usually characterized by a fast and asymptotically stable transient.

This phenomenon can be limited by choosing $\mathbf{K_o}$ small enough to filter out high frequency noise, which however reduces the response speed to collision events. The choice of values of the gain matrix $\mathbf{K_o}$ is therefore a compromise between the need for a fast collision detection and the desired noise rejection.

Another implementation issue worth mentioning is the calculation of $\boldsymbol{\beta}(\mathbf{q}, \dot{\mathbf{q}})$ in (1.8), which requires the term $\dot{\mathbf{M}}(\mathbf{q})$ to be computed. If this term is not available or too heavy in terms of computational complexity, it might be easier to implement the following equivalent version: let $\mathbf{S}(\mathbf{q}, \dot{\mathbf{q}})$ be a factorization of the Coriolis and centrifugal term $\mathbf{c}(\mathbf{q}, \dot{\mathbf{q}})$

$$\mathbf{c}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{S}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} \tag{1.15}$$

such that the term $\dot{\mathbf{M}}(\mathbf{q}) - 2\mathbf{S}(\mathbf{q}, \dot{\mathbf{q}})$ is skew-symmetric, which means that

$$\dot{\mathbf{M}}(\mathbf{q}) - 2\mathbf{S}(\mathbf{q}, \dot{\mathbf{q}}) = -\dot{\mathbf{M}}^{\mathbf{T}}(\mathbf{q}) + 2\mathbf{S}^{\mathbf{T}}(\mathbf{q}, \dot{\mathbf{q}}). \tag{1.16}$$

Being $\mathbf{M}(\mathbf{q})$ a symmetric matrix, $\dot{\mathbf{M}}(\mathbf{q})$ is also symmetric, thus (1.16) is equivalent to

$$\dot{\mathbf{M}}(\mathbf{q}) = \mathbf{S}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{S}^{\mathbf{T}}(\mathbf{q}, \dot{\mathbf{q}}). \tag{1.17}$$

The alternative expression for $\boldsymbol{\beta}(\mathbf{q}, \dot{\mathbf{q}})$ is therefore obtained by substituting (1.17) in (1.8), leading to

$$\boldsymbol{\beta}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{g}(\mathbf{q}) + \boldsymbol{\tau}_{\mathbf{f}} - \mathbf{S}^{\mathbf{T}}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}}. \tag{1.18}$$

The actual convenience of this form depends on the availability of the terms involved.

Furthermore, in a discretized implementation of the residual method, the term $\dot{\mathbf{M}}(\mathbf{q})$ can be also approximated as

$$\dot{\mathbf{M}}_{\mathbf{k}}(\mathbf{q}) \approx \frac{\mathbf{M}_{\mathbf{k}}(\mathbf{q}) - \mathbf{M}_{\mathbf{k-1}}(\mathbf{q})}{T_s}, \tag{1.19}$$

where $\mathbf{M}_{\mathbf{i}}(\mathbf{q})$ is the value of the inertia matrix at step i and T_s is the sampling time.

1.2 Collision detection

During a collision event with the i-th link of the manipulator, an external wrench

$$\mathcal{F}_{ext} = \begin{bmatrix} \mathbf{f_{ext}} \\ \mathbf{m_{ext}} \end{bmatrix} \tag{1.20}$$

is exchanged between the link and the environment, producing the joint torque

$$\boldsymbol{\tau}_{\mathbf{k}} = \mathbf{J}_{\mathbf{c}}^{\mathbf{T}}(\mathbf{q})\mathcal{F}_{ext}, \tag{1.21}$$

where $\mathbf{J_c}(\mathbf{q})$ is the (unknown) geometric Jacobian matrix at the (unknown) collision point.

This external joint torque excites the dynamics of the residuals according to (1.2), and the collision detection can be therefore performed by monitoring either the norm of the residual vector or the magnitude of its components: ideally, a collision is detected whenever the monitoring signal is different from zero. In practice, due to model uncertainties, noise and disturbances, the monitoring signal might be non-zero even when no collision occurs. To avoid false positives, a threshold is defined, which endows the system with a margin of robustness, at the price of increased detection delay and possible false negatives. The threshold is usually tuned experimentally, and can also be chosen adaptively to overcome the aforementioned issues.

More in general, a different monitoring signal can be chosen by designing an appropriate filter in cascade with the residual dynamics. This allows to focus on a specific collision frequency bandwidth, i.e. on a specific type of collision: for example, using a high pass filter allows to focus only on fast and stiff impacts, ignoring what may be instead an intentional contact with the robot.

1.3 Collision isolation

Let's consider again equation (1.21), which relates the external contact wrench to the joint torques it generates. The transformation matrix between those two quantities is the transpose of the geometric Jacobian, $\mathbf{J}_\mathbf{c}^\mathbf{T}(\mathbf{q})$, which is a configuration dependent $n \times 6$ matrix.

The Jacobian possesses some properties worth mentioning: in first place, it depends only on the current robot configuration, making the residuals insensitive to joint velocities and accelerations; in other words, the analysis of the residual method can be carried on based only on static considerations.

In second place, if its kernel is non-empty, there exist a subspace of wrenches that do not produce any joint torque, therefore not affecting the residual dynamics. This entails that this technique only detects collisions where the external wrench produce work on the robot system.

Lastly, the contact point Jacobian matrix possesses the property of directionality, a structural property fundamental for collision isolation. Let's consider a point on a generic link i of the robot manipulator: the direct kinematics of such point does not depend on successive joint variables ($i + 1$ to n). This means the corresponding geometric Jacobian has the form

$$\mathbf{J}_\mathbf{c}(\mathbf{q}) = \left[* * * | \mathbf{O} \right], \tag{1.22}$$

namely, the last $n - i$ columns are zeros.

As a consequence, when a collision occurs on link i, no external joint torque is generated at the joints $i + 1$ to n, due to the dependence on $\mathbf{J}_\mathbf{c}^\mathbf{T}(\mathbf{q})$. This, together with the decoupled nature of the dynamics (1.2), guarantees that the collision on the i-th link does not activate the last $n-i$ residuals. The colliding link can be therefore reconstructed as

$$i_c = max\{i \in \{1,...,n\} : r_i \neq 0\}. \tag{1.23}$$

Once the link isolation problem is solved, the next step is to isolate the contact point. Since the collided link is now known, it is possible to factorize the unknown contact point geometric Jacobian $\mathbf{J_c}(\mathbf{q})$ as

$$\mathbf{J_c}(\mathbf{q}) = \mathbf{J}_{\mathbf{c},\mathrm{i}}\mathbf{J}_{\mathrm{i}}(\mathbf{q}), \tag{1.24}$$

where $\mathbf{J}_{\mathrm{i}}(\mathbf{q})$ is the known geometric Jacobian at the origin of the frame of link i and $\mathbf{J_{c,i}}$ is an unknown and constant 6×6 transformation matrix defined as

$$\mathbf{J}_{\mathbf{c},\mathrm{i}} = \begin{bmatrix} \mathbf{I_{3x3}} & -\mathbf{S}({}^0\mathbf{r}_{\mathrm{i},\mathbf{c}}) \\ \mathbf{O_{3x3}} & \mathbf{I_{3x3}} \end{bmatrix}, \tag{1.25}$$

where ${}^0\mathbf{r}_{\mathrm{i},\mathbf{c}}$ is the unknown position vector from the origin of frame i to the contact point, expressed in frame 0, and $\mathbf{S}({}^0\mathbf{r}_{\mathrm{i},\mathbf{c}})$ is the skew-symmetric matrix associated with it: given $\mathbf{r} = \begin{bmatrix} r_x & r_y & r_z \end{bmatrix}^T$,

$$\mathbf{S}(\mathbf{r}) = \begin{bmatrix} 0 & -r_z & r_y \\ r_z & 0 & -r_x \\ -r_y & r_x & 0 \end{bmatrix}. \tag{1.26}$$

All the uncertainty of the original geometric Jacobian is therefore concentrated on the position vector ${}^0\mathbf{r}_{\mathrm{i},\mathbf{c}}$. Assuming the contact point doesn't move during the collision, the vector ${}^0\mathbf{r}_{\mathrm{i},\mathbf{c}}$ and the transformation matrix $\mathbf{J}_{\mathbf{c},\mathrm{i}}$ are both constant.

Furthermore, it is possible to transform the external wrench $\mathcal{F}_{ext}$, applied on the contact point on the i-th link, to an equivalent wrench applied to the origin of the i-th frame using the relation

$$\mathcal{F}_i = \mathbf{J}_{\mathbf{c},\mathrm{i}}^{\mathbf{T}}\mathcal{F}_{ext}. \tag{1.27}$$

By virtue of (1.21), and being $\mathbf{r}$ an estimate of $\boldsymbol{\tau}_\mathbf{k}$ (for a big enough $\mathbf{K_o}$), an estimate of $\mathcal{F}_i$ is

$$\hat{\mathcal{F}}_i = (\mathbf{J}_{\mathrm{i}}^{\mathbf{T}}(\mathbf{q}))^{\#}\mathbf{r}. \tag{1.28}$$

To simplify the problem, assume that during the collision no external moments are applied to the colliding link, as it is in the case of impulsive collisions, resulting in an external wrench of the form

$$\mathcal{F}_{ext} = \begin{bmatrix} \mathbf{f_{ext}} \\ \mathbf{0_{3x1}} \end{bmatrix}. \tag{1.29}$$

Under this assumption, and once the estimate (1.28) is obtained, the vector ${}^0\mathbf{r}_{\mathrm{i},\mathbf{c}}$ is obtained solving the system of equations (1.27)

$$\hat{\mathcal{F}}_i = \begin{bmatrix} \hat{\mathbf{f}}_i \\ \hat{\mathbf{m}}_i \end{bmatrix} = \begin{bmatrix} \mathbf{I_{3x3}} & \mathbf{O_{3x3}} \\ -\mathbf{S}^{\mathbf{T}}({}^0\mathbf{r}_{\mathrm{i},\mathrm{c}}) & \mathbf{I_{3x3}} \end{bmatrix} \begin{bmatrix} \mathbf{f_{ext}} \\ \mathbf{0_{3x1}} \end{bmatrix}, \tag{1.30}$$

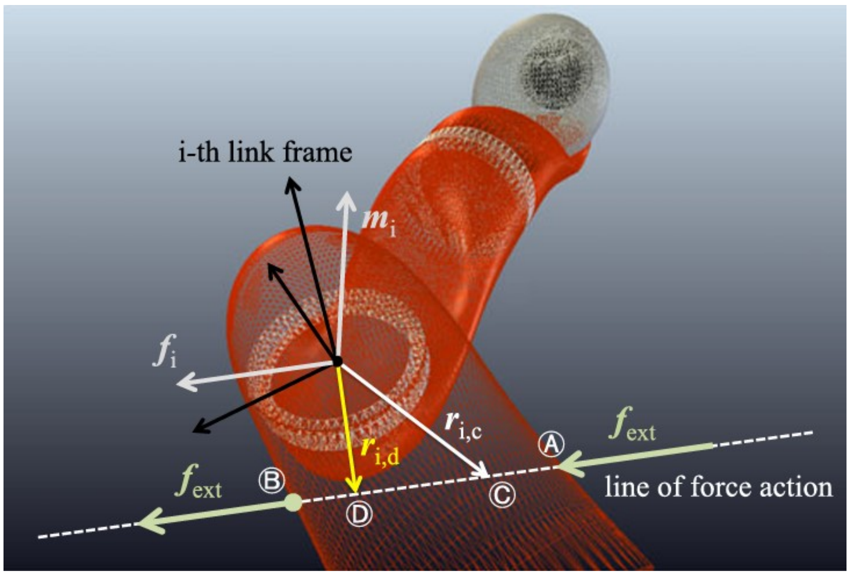


Figure 1.1: The line of action of the contact force: copyrights to [3].

which is equivalent to solve

$$\hat{\mathbf{m}}_i = -\mathbf{S}^\mathbf{T}({}^0\mathbf{r_{i,c}})\hat{\mathbf{f}}_i, \tag{1.31}$$

After algebraic manipulation (the skew symmetric matrix $\mathbf{S}(\cdot)$ can be interpreted as a vector product operator, which is commutative under sign change),

$$\hat{\mathbf{m}}_i = \mathbf{S}^\mathbf{T}(\hat{\mathbf{f}}_\mathrm{i})^0\mathbf{r}_{\mathrm{i},\mathbf{c}}. \tag{1.32}$$

Notice that the matrix $\mathbf{S}(\hat{\mathbf{f}}_\mathrm{i})$ has structurally rank 2 as long as $\hat{\mathbf{f}}_\mathrm{i} \neq \mathbf{0_{3x1}}$. This entails the full solution to (1.32) is a subspace of dimension 1, representing the line of action of the collision force. This is because different, properly scaled, forces acting along this line of action produce the same motion, and therefore the same effect on the residuals, assuming the contact holds.

By using the pseudo-inverse, the minimum distance vector (minimum norm solution)

$$^0\mathbf{r}_{\mathrm{i},\mathbf{d}} = \left(\mathbf{S}^\mathbf{T}(\hat{\mathbf{f}}_\mathrm{i})\right)^\# \hat{\mathbf{m}}_\mathrm{i} \tag{1.33}$$

is obtained. It is then possible to reconstruct the line of action along which the force is applied, which is

$$^0\mathbf{r}_{\mathrm{i},\mathbf{c}} = {}^0\mathbf{r}_{\mathrm{i},\mathbf{d}} + \lambda\frac{\hat{\mathbf{f}}_\mathrm{i}}{||\hat{\mathbf{f}}_\mathrm{i}||}, \tag{1.34}$$

where λ is a scalar.

For example, as depicted in Figure 1.1, a pushing or a pulling force applied on the line of action on points symmetric with respect to $^0\mathbf{r}_{\mathrm{i},\mathbf{d}}$ produce the same work. Eventually, it is possible to reduce the solution space to just two points by intersecting the line of action with the geometric boundary of the link, if known.

1.4 Collision identification

The last step in the collision analysis is the identification, i.e. the estimation of the external impacting force.

Once the isolation problem is solved and $\mathbf{J_c^T}$ is fully known, an estimate of the external wrench is obtained from equation (1.21) simply as

$$\hat{\mathcal{F}}_{ext} = (\mathbf{J_c^T}(\mathbf{q}))^{\#}\mathbf{r}. \tag{1.35}$$

Unfortunately, an accurate estimation of both the collision point and the collision force is often impossible. This mainly happens because loss of information is unavoidable when $\mathbf{J}_\mathrm{i}$ is not full rank, which is always the case for a colliding link $i_c < 6$, but can also happen for $i_c \geq 6$ in the presence of singularities.

Notice also that the contact point isolation and force identification provide good results under the assumption that $\mathbf{r}$ is an accurate estimate of $\boldsymbol{\tau}_\mathbf{k}$, which is only true for a large residual gain $\mathbf{K_o}$.

Furthermore, the overall algorithm assumes only one collision happening at the same time, and thus it is unreliable in case of multiple simultaneous collisions.

Chapter 2

Kinematic and dynamic model of the ABB YuMi robot

This chapter is dedicated to the derivation of the kinematic and dynamic model of the YuMi robot. This is a fundamental step toward implementation since the residual method is a model-based technique and its reliability is strongly dependent on the accuracy of the model used.

Two main problems must be taken into account when deriving the model. First, while the datasheet of the ABB YuMi [1] provides kinematic and geometric parameters, it does not contain the values of dynamic parameters for the links. To solve this issue, a CAD analysis is performed using the 3D model files provided by ABB [1].

The second problem regards computation times: the residual method is conceived to monitor the robot dynamics online, and a fast response is crucial for real-time reaction strategies. The model must therefore be as computationally efficient as possible; even in a simulation environment, where real-time execution is not necessary a requirement, it is still desirable to have optimized functions to bring down simulation times.

For instance, a model in symbolic form, obtained through the recursive version of the moving frames algorithm (Euler-Lagrange approach), is too complex to handle for a 7 DoF robot. For this reason, an optimized Newton-Euler algorithm must be looked for.

Several techniques and software packages are available for the robotics community to obtain the dynamic model of a given robot. As a first attempt, the MATLAB Robotics System Toolbox was explored, but the functions available in the toolbox resulted in extremely slow simulation times. Moreover, while the kinematic model of the robot in the toolbox is very accurate, the dynamic parameters are not realistic for the geometry of the links. This approach was thus discarded.

Secondly, a custom MATLAB implementation of the Newton-Euler (NE) algorithm has been tested. Even though a single call to our function was fast enough for a real-time implementation, repeated calls inside the `ode45` solver (see Section 3.1) resulted in increased simulation times.

In conclusion, the model obtained through openSYMORO [2][4], a Python based software which generates dynamic models using the modified DH convention and a very efficient version of the Newton-Euler algorithm, proved to run with computation times comparable to our NE implementation for single function calls, but was much faster in repeated calls inside `ode45`.

2.1 Kinematic model

The ABB YuMi single arm is an open kinematic chain manipulator with fixed base and rigid links connected through 7 revolute joints, as shown and labelled in Figure 2.1.

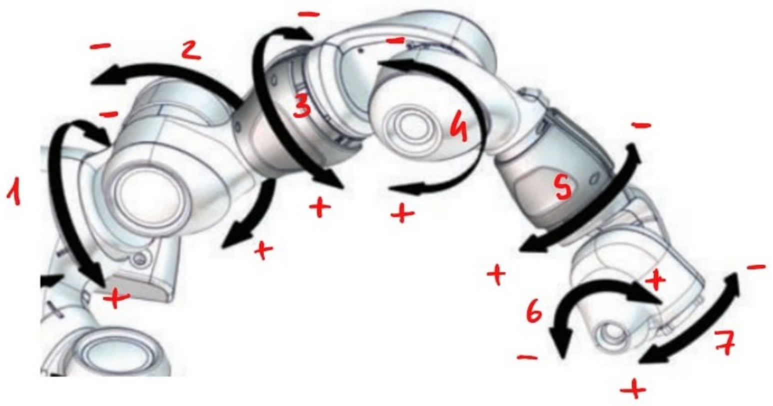


Figure 2.1: Joints labelling.

To derive a kinematic model, the modified Denavit-Hartenberg (DHM) convention was used, as required by openSYMORO. In the DHM convention, once a base frame is set, the n link frames are assigned based on the following rules:

- α_i is the twist angle from z_{i-1} to z_i, measured about x_{i-1};
- a_i is the distance from z_{i-1} to z_i, measured along x_{i-1};
- d_i is the distance from x_{i-1} to x_i, measured along z_i;
- θ_i is the angle from x_{i-1} to x_i, measured about z_i.

Overall, the modified DH sets the origin of frames on the base of each link, while in the standard DH version they are placed at the tip.

This choice leads to an equivalent kinematic description of the robot, where the homogeneous transformation matrix between successive frames is defined as

$$^{i-1}\mathbf{A}_i = \begin{bmatrix} c\theta_i & -s\theta_i & 0 & a_i \\ c\alpha_i s\theta_i & c\alpha_i c\theta_i & -s\alpha_i & -d_i s\alpha_i \\ s\alpha_i s\theta_i & s\alpha_i c\theta_i & c\alpha_i & d_i c\alpha_i \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{2.1}$$

DHM parameters of the YuMi are reported in Table 2.1, while the corresponding frame assignment is shown in Figure 2.2, where the corresponding configuration is

$$\mathbf{q} = \begin{bmatrix} 0 & \pi & 0 & -\frac{\pi}{2} & \pi & \pi & \pi \end{bmatrix}^T . \tag{2.2}$$

i	α_i [rad]	a_i [m]	d_i [m]	θ_i [rad]
1	0	0	0.166	q_1
2	$\pi/2$	0.03	0	q_2
3	$\pi/2$	0.03	0.2515	q_3
4	$-\pi/2$	0.0405	0	q_4
5	$-\pi/2$	0.0405	0.265	q_5
6	$-\pi/2$	0.027	0	q_6
7	$-\pi/2$	0.027	0.036	q_7

Table 2.1: Parameters of the modified DH convention.

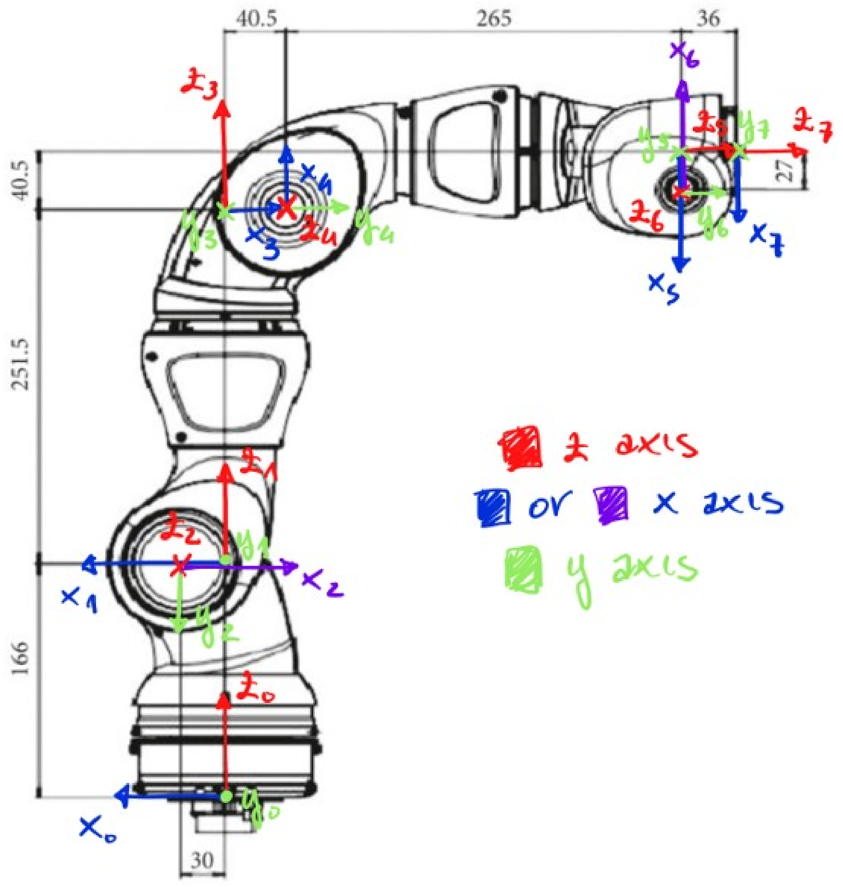


Figure 2.2: Modified DH frames. Dimensions in mm.

2.2 Dynamic model

2.2.1 Nominal model

Together with the geometric parameters of the modified DH frames, openSYMORO needs 10 inertial parameters to be specified for each link, for a total of 70, to generate the model:

- XX_i,XY_i,XZ_i,YY_i,YZ_i,ZZ_i, the elements of the inertia matrix of link i around the origin of frame i;
- MX_i,MY_i,MZ_i, the elements of the first order moment of link i;
- m_i, the mass of link i.

Viscous and static friction and actuator inertia are not considered in this work.

The links of the YuMi are made of a magnesium alloy that makes the overall structure very lightweight, for a total of 9.5 kg [1]. ABB provides 3D `.step` files for each link, which were imported in the Autodesk Fusion 360 CAD software. In CAD, the material of each link was set to magnesium AZ63A, for a total estimated weight of 9.447 kg. Each link was translated and rotated in accordance with the corresponding DHM frame. From this, the CAD software can compute an estimate of the required inertial parameters. First order moments are computed as the In product of center of mass (CoM) and mass.

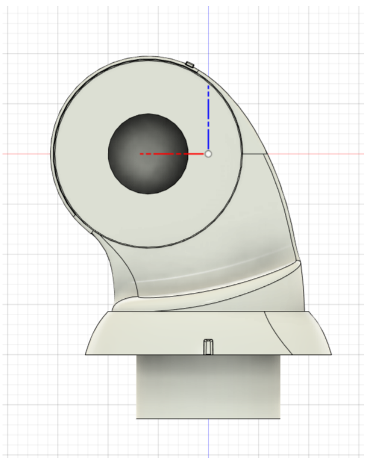
The results of the CAD analysis are shown in Figure 2.3 and summarized in Table 2.2.

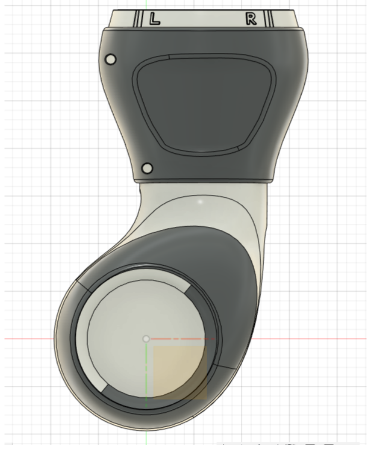
Parameter	Link 1	Link 2	Link 3	Link 4	Link 5	Link 6	Link 7
m [kg]	1.761	2.577	1.408	2.158	0.628	0.859	0.056
CoM_x [m]	0.009	0.017	0.019	0.023	0.007	0.011	0
CoM_y [m]	-0.023	-0.063	0.026	0.056	0.027	-0.012	-0.001
CoM_z [m]	-0.05	-0.013	-0.035	-0.017	-0.054	-0.01	-0.011
XX [kg m^2]	0.011	0.024	0.007	0.017	0.004	0.001	$1.980 \cdot 10^{-5}$
XY [kg m^2]	0.001	0.005	-0.001	-0.005	0	0	≈ 0
XZ [kg m^2]	0	0	0	0	0	0	≈ 0
YY [kg m^2]	0.01	0.005	0.007	0.005	0.003	0.001	$1.971 \cdot 10^{-5}$
YZ [kg m^2]	0	0	0	0	0	0	≈ 0
ZZ [kg m^2]	0.004	0.024	0.004	0.017	0.001	0.001	$1.578 \cdot 10^{-5}$

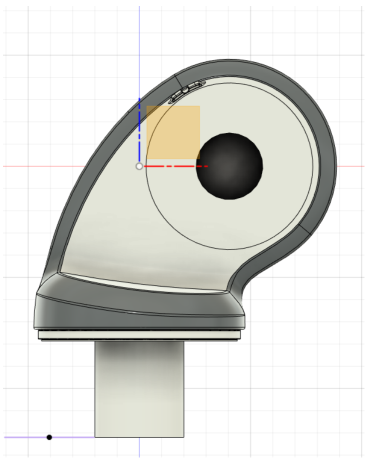
Table 2.2: The 54 dynamic parameters.

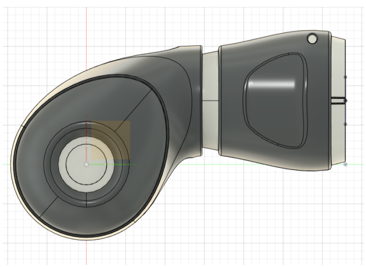
The total number of dynamic parameters turned out to be less than 70: some were structurally zero while the off-diagonal terms of the inertia matrix of the seventh link were approximated to zero since they were three orders of magnitude smaller than the diagonal elements. Overall, the dynamic parameters were reduced to 52.

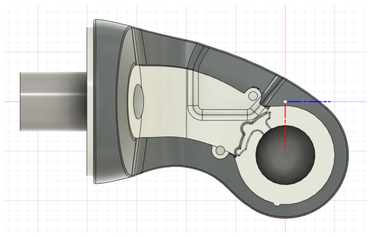
In order to obtain an even more efficient model in terms of simulation time, the number of parameters was further reduced to 29 dynamic coefficients using the "Base Parameters" functionality of openSYMORO.

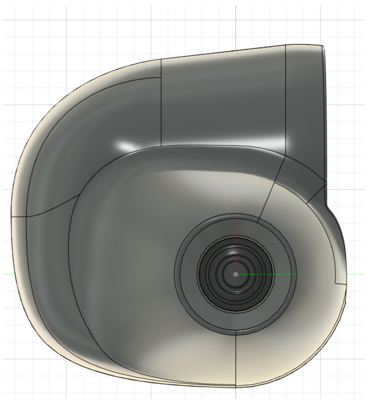

(a) Link 1.


(b) Link 2.


(c) Link 3.


(d) Link 4.


(e) Link 5.


(f) Link 6.

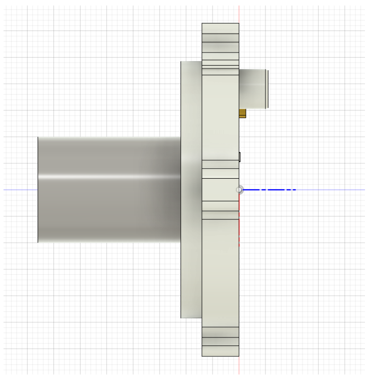

(g) Link 7.

Figure 2.3: Links modelling in Autodesk Fusion 360 CAD.

2.2.2 Uncertain model

In order to generate a more realistic model, up to 20% of uncertainty on each dynamic parameter is considered. Code-wise, these are represented as a multiplicative constant on the nominal value of each dynamic parameter, ranging from 0.8 to 1.2. In particular, three different uncertainties are considered for each link:

- One for the elements of the inertia matrix $(XX_i, XY_i, XZ_i, YY_i, YZ_i, ZZ_i)$;
- One for the center of mass $(CoM_{i,x}, CoM_{i,y}, CoM_{i,z})$;
- One for the mass (m_i).

The uncertainty on the first order moment of a given link is computed as the product of the uncertainty on the mass and the center of mass of such link.

Note that the robot dynamics is simulated using the uncertain model, while the control and the residual dynamics are always computed with the nominal model. The effect of different levels of uncertainty on collision detection, isolation and identification during the execution of different tasks, is shown in Chapter 3.

2.2.3 MATLAB implementation

openSYMORO generates Python code to compute the dynamic model of the robot. The generated code already takes into account the multiplicative constant to represent uncertainty, as explained in section 2.2.2; the nominal model can be reconstructed by setting the uncertainty value to 1 for all the parameters. The files are manually converted into three MATLAB functions, respectively to compute:

- The inertia matrix of the robot $\mathbf{M}(\mathbf{q})$;
- The nonlinear terms $\mathbf{n}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{c}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q})$;
- The Jacobian at the origin of each frame $\mathbf{J_i}(\mathbf{q})$.

In conclusion, a dynamic model of the form

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{n}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{u} \tag{2.3}$$

is obtained for the YuMi robot. Moreover, the gravity term can be computed as

$$\mathbf{g}(\mathbf{q}) = \mathbf{n}(\mathbf{q}, \mathbf{0}). \tag{2.4}$$

Chapter 3

Simulations on the ABB YuMi robot

3.1 General simulation structure

This chapter is dedicated to the presentation of some MATLAB simulations performed on the YuMi robot. In order to realize a realistic simulation, the controller, together with the residual dynamics and the collision handling scheme, was implemented in discrete form using a zero-order hold, with a sampling time of 1 ms, while the robot dynamics was instead simulated in continuous time using the `ode45` solver.

In particular, the residual dynamics was computed as in (1.12), where the integral in (1.13) was approximated by Euler integration and the term $\dot{\mathbf{M}}(\mathbf{q})$ was computed as in (1.19).

Detection, isolation and identification are performed simultaneously, as soon as a collision is detected, and once per collision. A collision is marked as detected on link i when no residual $j > i$ is above its threshold while residual i is above its threshold for two sampling times. This is done because the response speed is not the same for each residual, and the event of residual $j < i$ activating earlier than residual i has to be discarded, otherwise the collision would be isolated on the wrong link.

This assumes that, at the start of the collision event, the residual of the colliding link is activated at most two sampling times after the first residual (as in, the residual that activates first, and not the residual associated to the first link). With a high enough $\mathbf{K_o}$ and with thresholds appropriately chosen, the residual dynamics is fast enough and this assumption holds, at least in the nominal case. Furthermore, with this monitoring scheme, the collision is always detected with a delay of at most 2 ms.

3.1.1 Controller

The collisions are simulated while the robot moves in controlled motion.

At first, a regulation task to the desired position $\mathbf{q_d} = \begin{bmatrix} 0 & \pi & 0 & -\frac{\pi}{2} & \pi & \pi & \pi \end{bmatrix}^T$ (Figure 2.2) is performed, with every joint starting at rest with the initial position $q_{0,i} = \frac{\pi}{2}$, which is roughly an average of the final positions.

Secondly, a tracking problem in the joint space in considered: each joint has to follow the sinusoidal trajectory $q_{d,i} = \sin(4t)$, hence with joint velocity $\dot{q}_{d,i} = 4\cos(4t)$ and joint acceleration $\ddot{q}_{d,i} = -16\sin(4t)$, starting from the same initial state as before. Joint space tracking is preferred over a more realistic Cartesian task for a more compact and comprehensible plot of the results.

The robot is controlled by means of the feedback linearization scheme

$$\boldsymbol{\tau}_\mathbf{m} = \hat{\mathbf{M}}(\mathbf{q})(\ddot{\mathbf{q}}_\mathbf{d} + \mathbf{K}_\mathbf{p}(\mathbf{q}_\mathbf{d} - \mathbf{q}) + \mathbf{K}_\mathbf{d}(\dot{\mathbf{q}}_\mathbf{d} - \dot{\mathbf{q}})) + \hat{\mathbf{n}}(\mathbf{q}, \dot{\mathbf{q}}), \tag{3.1}$$

where:

- $\mathbf{q}_\mathbf{d}$, $\dot{\mathbf{q}}_\mathbf{d}$ and $\ddot{\mathbf{q}}_\mathbf{d}$ are, respectively, the desired joint trajectory, velocity and acceleration, as functions of time;
- $\mathbf{q}$ and $\dot{\mathbf{q}}$ are, respectively, the measured values of joint position and velocity;
- $\hat{\mathbf{M}}(\mathbf{q})$ and $\hat{\mathbf{n}}(\mathbf{q}, \dot{\mathbf{q}})$ are, respectively, the nominal robot inertia matrix and the nominal robot nonlinear terms;
- $\mathbf{K}_\mathbf{p}$ and $\mathbf{K}_\mathbf{d}$ are, respectively, the proportional and derivative gain.

Since control effort and actuators saturation are not considered in these simulations, the values of the two gains can be freely chosen and they are set to $\mathbf{K}_\mathbf{p} = diag\{200\}$ and $\mathbf{K}_\mathbf{d} = diag\{200\}$.

Notice that, for the regulation task, $\dot{\mathbf{q}}_\mathbf{d}$ and $\ddot{\mathbf{q}}_\mathbf{d}$ are identically zero, hence the controller 3.1 is just

$$\boldsymbol{\tau}_\mathbf{m} = \hat{\mathbf{M}}(\mathbf{q})(\mathbf{K}_\mathbf{p}(\mathbf{q}_\mathbf{d} - \mathbf{q}) - \mathbf{K}_\mathbf{d}\dot{\mathbf{q}}) + \hat{\mathbf{n}}(\mathbf{q}, \dot{\mathbf{q}}). \tag{3.2}$$

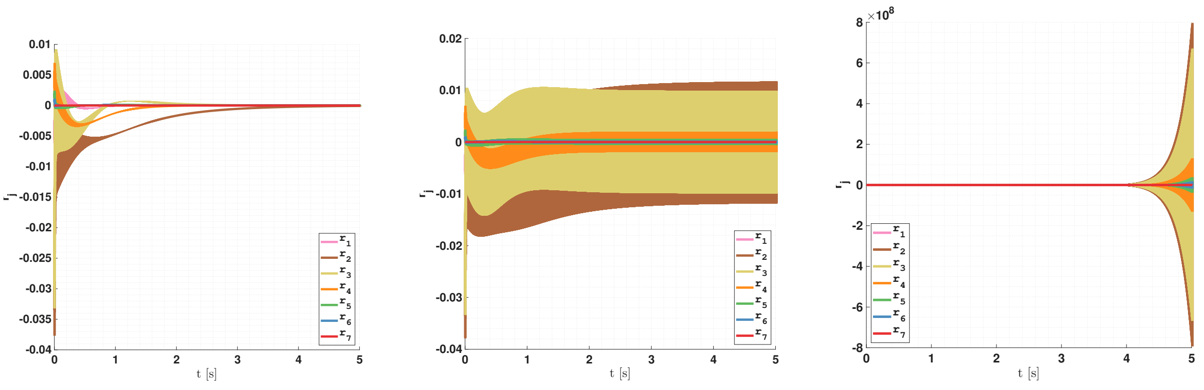
3.1.2 Stability of the residual dynamics

Theoretically, the residual dynamics (1.2), is asymptotically stable as long as $\mathbf{K}_\mathbf{o}$ is positive definite. In practice, the discretization of the control scheme sets an upper bound on the value of $\mathbf{K}_\mathbf{o}$, past which the residual dynamics becomes unstable.

In the present case, a diagonal matrix of the form $\mathbf{K}_\mathbf{o} = diag\{K_o\}$ is chosen for the simulations. The value $K_o = 2000$ s^{-1} was found, experimentally, to be the limit of instability for the residual dynamics (Figure 3.1). A value of $K_o = 1000$ s^{-1} is thus chosen.

3.1.3 Contact force modeling

The collision force is modeled as an impulsive wrench of the form (1.29) acting on a constant contact point for a time interval of 10 ms. Notice that, structurally, no collision on link 1 can be detected since a wrench of this form doesn't produce any work if applied to the first link.



(a) $K_o = 1995$ (b) $K_o = 2000$ (c) $K_o = 2005$

Figure 3.1: Residual dynamics in nominal conditions under regulation task for different values of K_o. Notice the high-frequency oscillating behaviour in all three cases of each residual, which eventually converges in (a) and diverges in (c), while it settles at a constant amplitude at steady state in (b).

Forces $\mathbf{f_{ext}}$ of magnitude between $||\mathbf{f_{ext}}||_{min} = 10$ N and $||\mathbf{f_{ext}}||_{max} = 500$ N are considered, as in [3], where a crash test between the KUKA LWR (a lightweight robot of weight comparable with the YuMi) and a human head is simulated using a mannequin.

3.1.4 Thresholds tuning

The choice of the thresholds is the most delicate part of the parameter tuning. The ideal threshold value should be small enough to be exceeded in less than 1 ms (the sampling time) when the corresponding residual is activated, even by the smallest force in the range. At the same time, it should not be too small to activate the residual when a big force is applied to a precedent link; this is because some interactions in the residual dynamics are present, mostly due to discretization and uncertainties.

The threshold for the i-th residual has been tuned empirically in the following way: a force with norm $||\mathbf{f_{ext}}|| = 10$ N, the minimum considered in this analysis, is applied to the origin of frame i, and the value of r_i after 1 ms is measured.

The steady state value for the threshold is chosen depending on the numerical conditioning of the residual. The values of the last four residuals vary widely between their excited and not excited states, and 50% of the value of the corresponding r_i is enough to activate them without false positives. The second and third residual, instead, turned out to be more problematic, showing very little difference between their steady state and excited values, especially under uncertainties. For this reason their thresholds are very close to their peaks, and the performance on those links are not as good as for the others. Notice however that a collision is more likely to happen at the terminal links of the robot, limiting the relevance of this issue.

Furthermore, the initial transient must be dealt with separately, as it often presents initial peaks exceeding the constant thresholds. To solve this issue, the full threshold is modeled as a negative exponential quickly converging to its steady state constant value.

3.2 Results

This section presents the results of some relevant simulations, where collisions on different links, with different contact forces and points are tested. All the simulations use a sampling time for the control and residual computation of 1 ms and a residual gain of $\mathbf{K_o} = diag\{1000\}$ s^{-1}.

Using the Parallel Computing Toolbox in MATLAB, multiple simulations are executed in parallel considering different uncertainties:

- 1) nominal model (no uncertainty);
- 2) random uncertainty on each of the dynamic parameters in the range $0\% - 5\%$;
- 3) random uncertainty on each of the dynamic parameters in the range $5\% - 10\%$;
- 4) random uncertainty on each of the dynamic parameters in the range $10\% - 15\%$;
- 5) random uncertainty on each of the dynamic parameters in the range $15\% - 20\%$.

Each time a new simulation is launched, the uncertainty values are generated randomly with uniform distribution in the desired range.

For the nominal case, the response of the residuals is deterministic, and it is possible to set pretty tight thresholds, namely

$$\boldsymbol{\epsilon}_{nom} = \begin{bmatrix} 0.05 & 0.05 & 0.03 & 0.05 & 0.05 & 0.05 & 0.05 \end{bmatrix}^T \cdot e^{1-30t}. \tag{3.3}$$

In the real uncertain case, instead, the thresholds must be relaxed and they have been set to:

$$\boldsymbol{\epsilon}_{unc} = \begin{bmatrix} 1 & 1.2 & 0.1 & 1 & 1 & 1 & 1 \end{bmatrix}^T \cdot e^{3-4t}. \tag{3.4}$$

Figure 3.2 shows a close-up of the residual transient for a simulation undergoing 5% uncertainty. Notice the residuals exceeding nominal thresholds, justifying why a relaxation of the thresholds is required in the presence of uncertainty.

3.2.1 Regulation task

The collision monitoring scheme presented in Section 3.1 is first tested for the regulation task described in Section 3.1.1 under the controller (3.2). Figure 3.3 shows the nominal evolution of joints and residuals without any collision.

First, a collision on the origin of link 4 is simulated, and detection results with different contact forces are shown. Next, an off-origin collision on link 6 is considered, and results on isolation and identification are discussed.

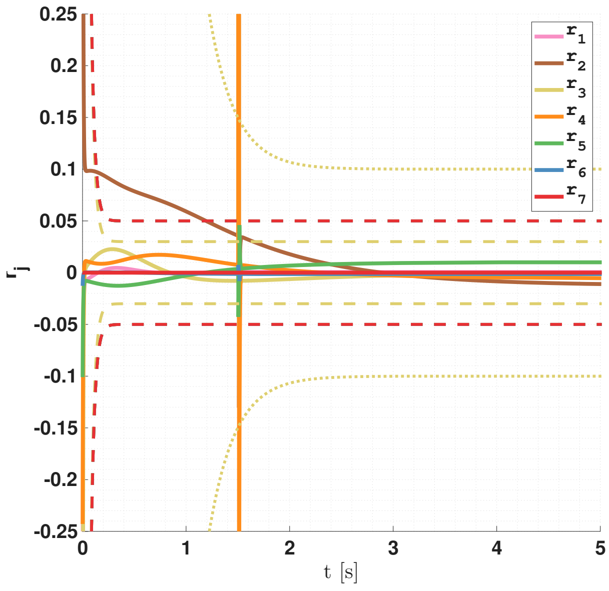
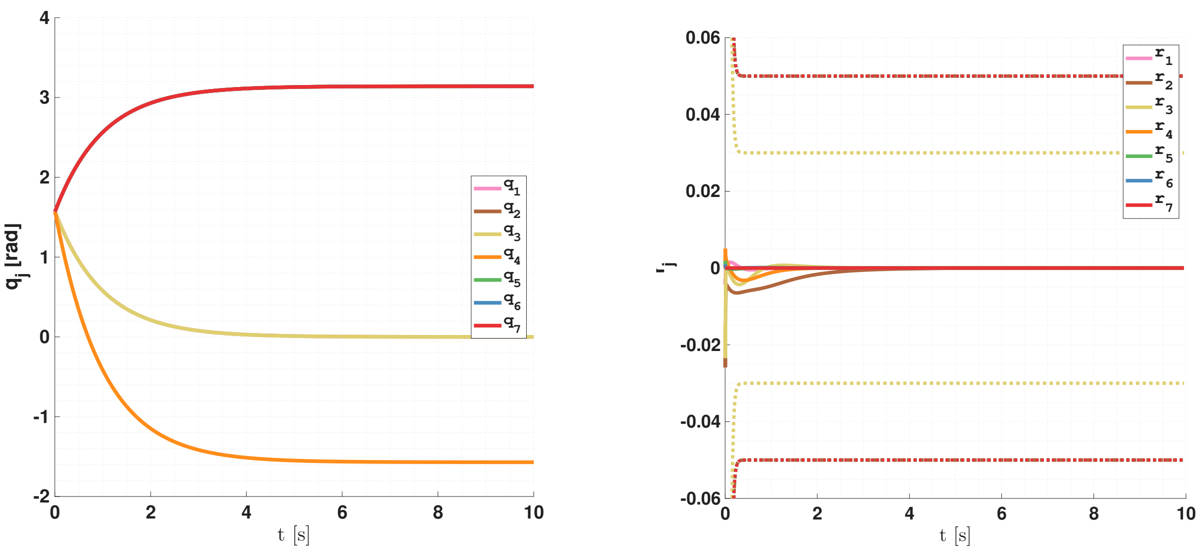


Figure 3.2: Nominal case thresholds (dashed line) and relaxed thresholds (dotted line) in the case of 5% uncertainty. Each threshold has the same colour of the corresponding residual (some are coincident, see (3.3) and (3.4)).



(a) Joints.

(b) Residuals.

Figure 3.3: Regulation task with feedback linearization control in nominal collision-free conditions. Each threshold is represented with a dotted line with the same colour of the corresponding residual (some are coincident, see (3.3) and (3.4)).

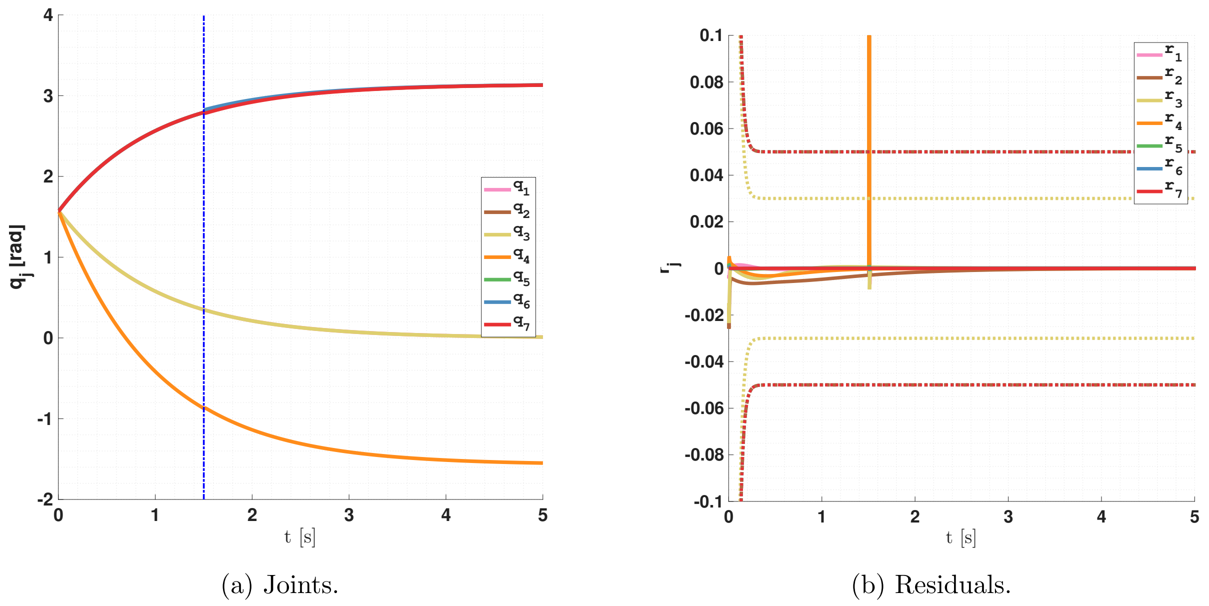


Figure 3.4: Regulation task in nominal conditions with 100 N collision force in the origin of frame 4.

Collision on link 4

In the first simulation, the collision is on the origin of link 4 with external force

$$\mathcal{F}_{ext} = \frac{100}{\sqrt{3}} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \text{ N} \tag{3.5}$$

applied at time $t = 1.5$ s, during the transient of the regulation task.

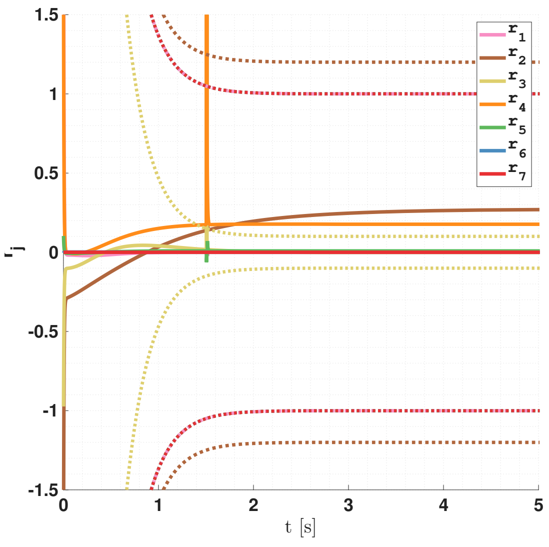
Figure 3.4 shows the resulting joint and residual values in the nominal case. A collision is correctly detected at $t = 1.501$ s, when the residual of link 4 is activated (a blue dotted vertical line is printed at the collision time). A close-up of the residual values in shown to highlight their behaviour near the thresholds.

As expected, isolation of the contact point and identification of the contact force are not accurate: the estimates are indeed

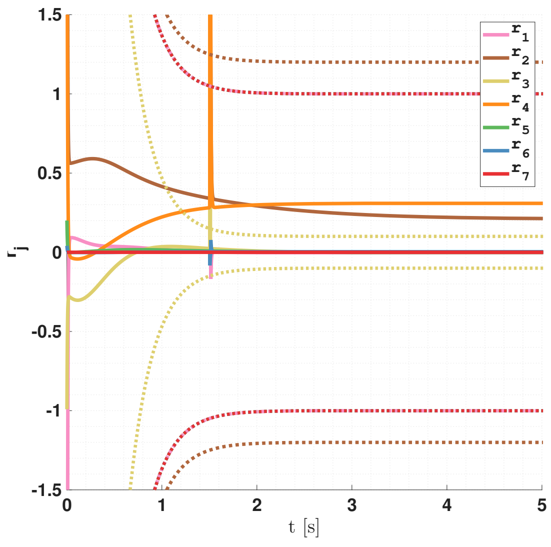
$$^{0}\mathbf{r}_{4,\mathbf{d}} = \begin{bmatrix} -0.0017 \\ -0.0007 \\ 0.0142 \end{bmatrix} \text{ m} \tag{3.6}$$

and

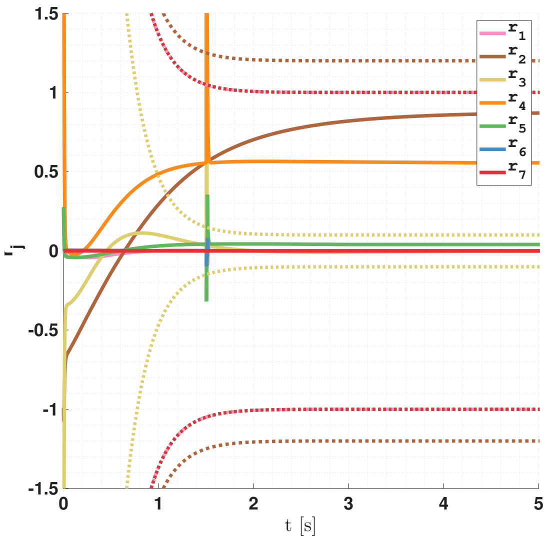
$$\hat{\mathcal{F}}_{ext} = \begin{bmatrix} 59.3696 \\ 49.6715 \\ -4.8100 \\ 1.4515 \\ 1.2157 \\ -0.1168 \end{bmatrix} \begin{bmatrix} \text{N} \\ \text{Nm} \end{bmatrix}. \tag{3.7}$$



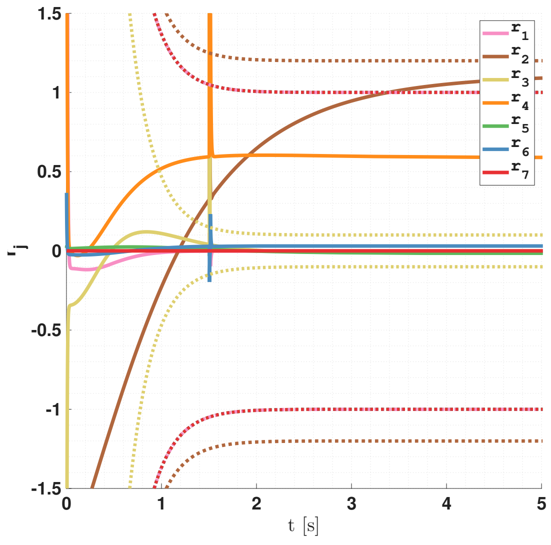
(a) Uncertainty: 0%-5%.



(b) Uncertainty: 5%-10%.



(c) Uncertainty: 10%-15%.



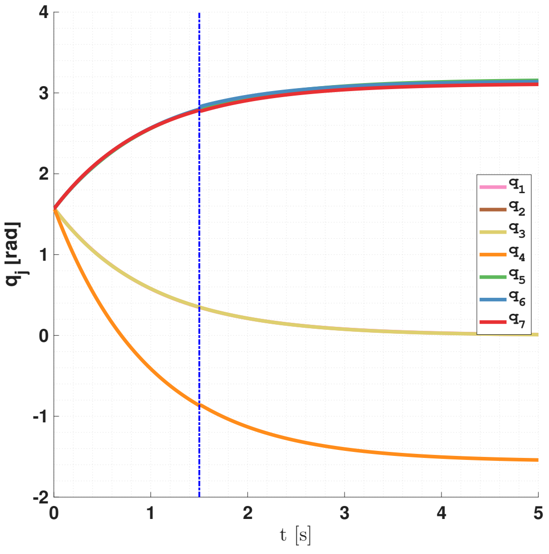
(d) Uncertainty: 15%-20%.

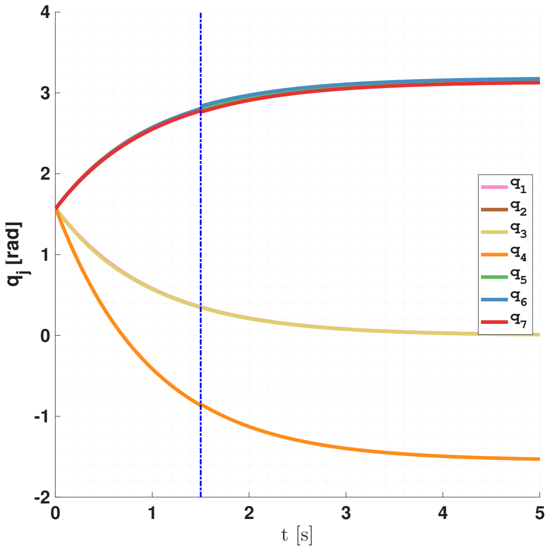
Figure 3.5: Residuals and thresholds for regulation task in uncertain conditions with 100 N collision force in the origin of frame 4.

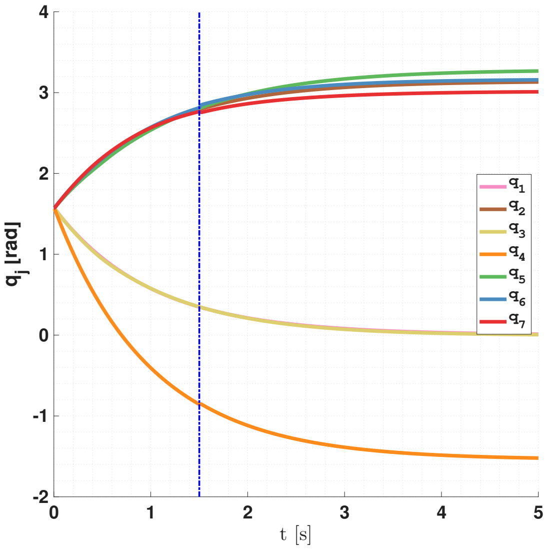
There is a progressive loss of information as the Jacobian matrix loses rank: in this case the first and second components of ${}^{0}\mathbf{r}_{4,\mathbf{d}}$ and consequently of $\hat{\mathcal{F}}_{ext}$ are close to the real values, while the remaining components present non negligible errors.

In the following, results of isolation and identification are only shown for the sixth and seventh links.

For the same regulation task, Figure 3.5 shows the residuals with different uncertainties on the dynamic parameters and relaxed thresholds. In all cases, a collision is correctly detected on link 4 at $t = 1.501$ s, with no false positives nor false negatives for the whole duration of the simulation. The increasing uncertainty however leads to increasing steady state error on the joint positions, as shown in Figure 3.6.


(a) Uncertainty: 0%-5%.


(b) Uncertainty: 5%-10%.


(c) Uncertainty: 10%-15%.

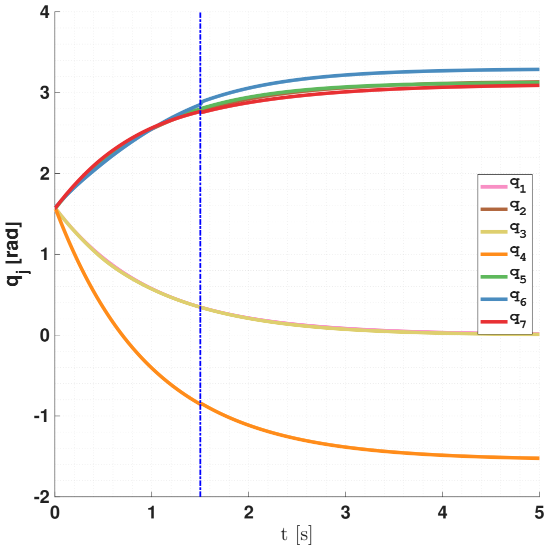
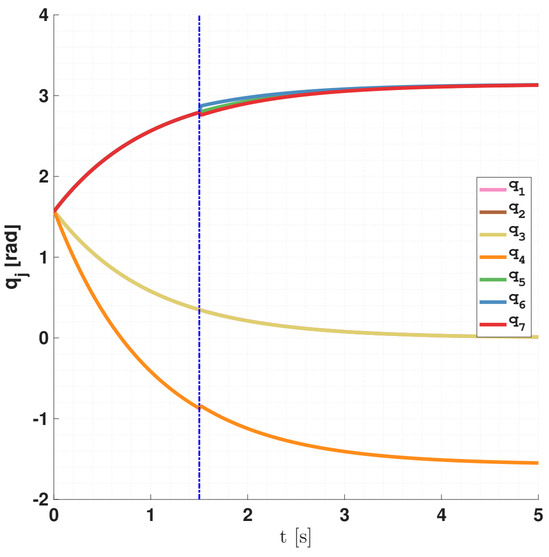
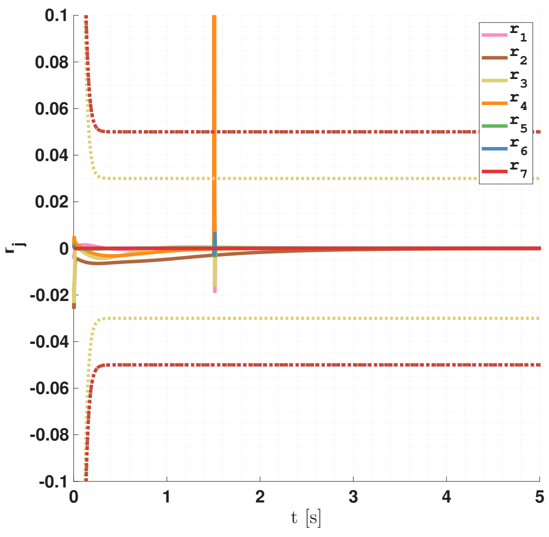

(d) Uncertainty: 15%-20%.

Figure 3.6: Joint positions for regulation task in uncertain conditions with 100 N collision force in the origin of frame 4.



(a) Joints.



(b) Residuals.

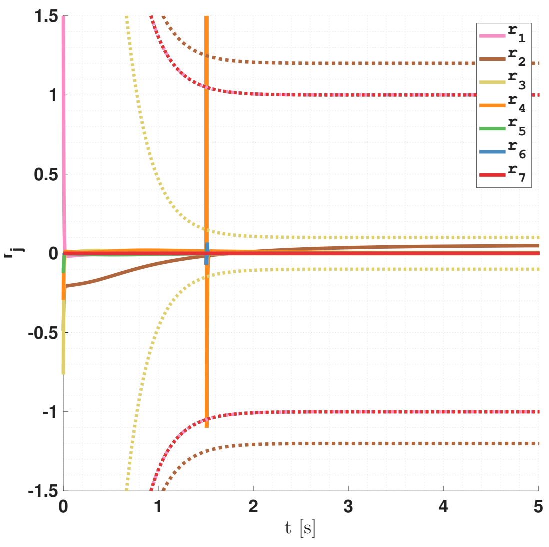
Figure 3.7: Regulation task in nominal conditions with 250 N collision force in the origin of frame 4.

The same simulation is now repeated with a force

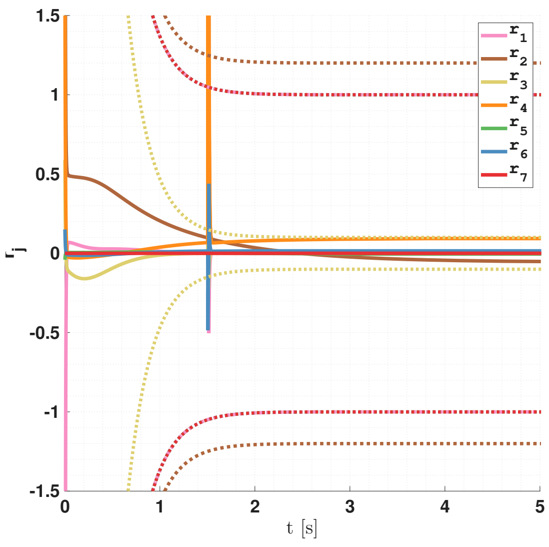
$$\mathcal{F}_{ext} = \frac{250}{\sqrt{3}} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \text{ N}, \tag{3.8}$$

i.e. increasing its magnitude from 100 N to 250 N.

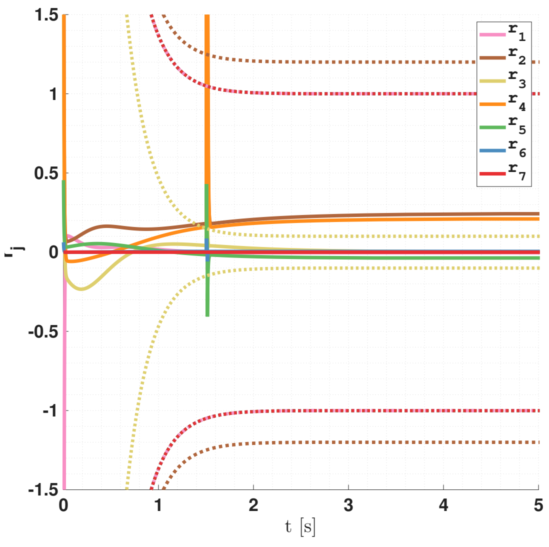
As in the previous simulation, the detection scheme recognizes the correct collided link both in the nominal case (Figure 3.7) and up to 15% of uncertainty. Increasing the uncertainty leads however to a false positive detection on link 3 at time $t = 1.137$ s, which also hides successive false positives and the actual collision. The uncertain plots are shown in Figure 3.8 for the residuals and in Figure 3.9 for the joint positions.



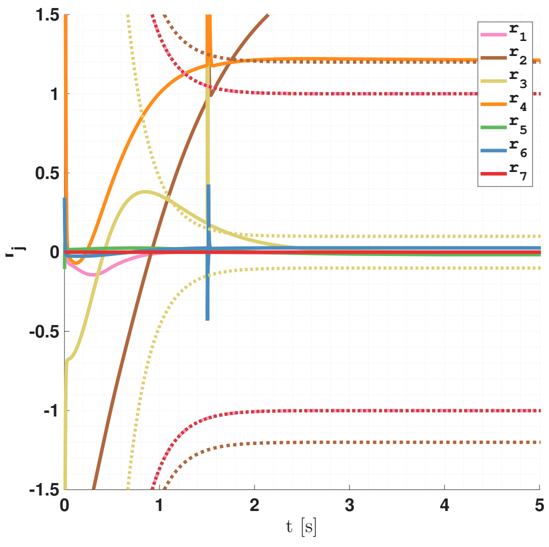
(a) Uncertainty: 0%-5%.



(b) Uncertainty: 5%-10%.

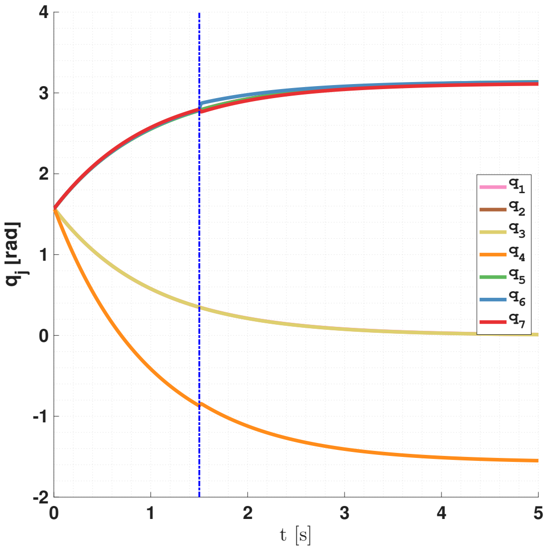


(c) Uncertainty: 10%-15%.

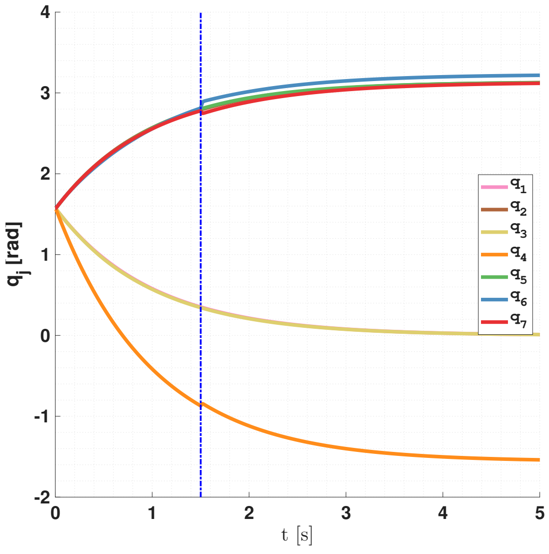


(d) Uncertainty: 15%-20%.

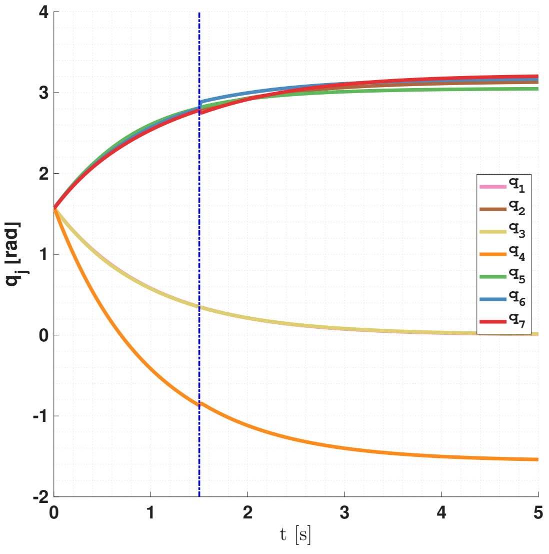
Figure 3.8: Residuals and thresholds for regulation task in uncertain conditions with 250 N collision force in the origin of frame 4.



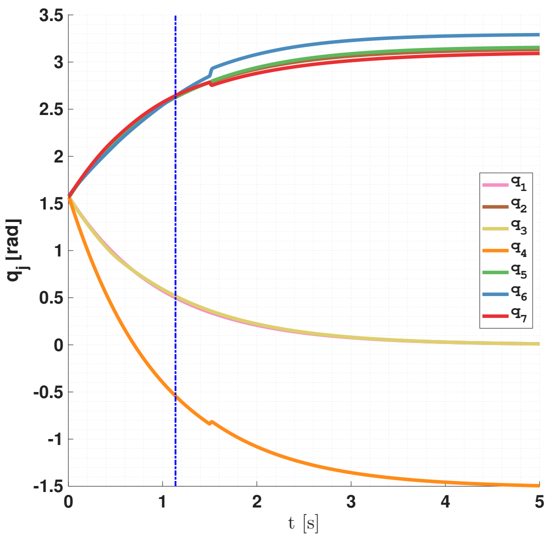
(a) Uncertainty: 0%-5%.



(b) Uncertainty: 5%-10%.

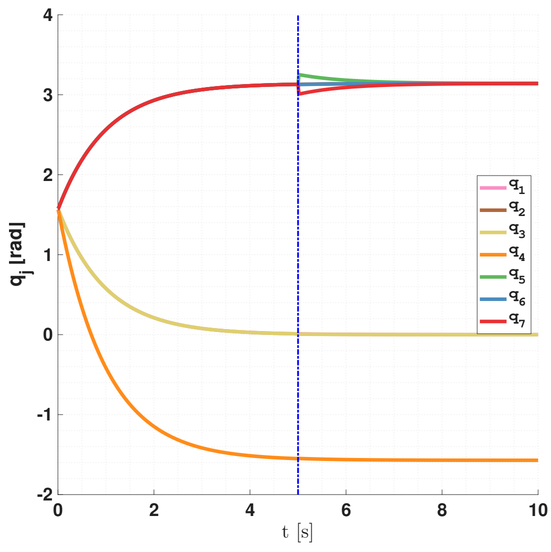


(c) Uncertainty: 10%-15%.

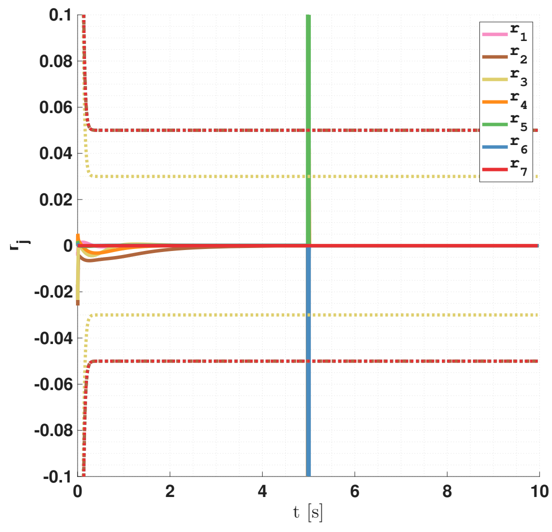


(d) Uncertainty: 15%-20%.

Figure 3.9: Joint positions for regulation task in uncertain conditions with 250 N collision force in the origin of frame 4.



(a) Joints.



(b) Residuals.

Figure 3.10: Regulation task in nominal conditions with off-origin collision force on frame 6.

Off-origin collision on link 6

The next simulation tests the performance of the isolation and identification algorithms, which work only if the collided link is at least the sixth (see Section 1.4). As explained in Section 1.3, the isolation algorithm does not return the exact contact point but the minimum distance vector from the origin of the collided link to the line of action of the force (${}^0\mathbf{r}_{\mathrm{i,d}}$). The major issue in those simulations is to understand if the value returned by the algorithm is the correct one, since ${}^0\mathbf{r}_{\mathrm{i,d}}$ is not in general evident.

The collision here simulated is therefore constructed to easily visualize ${}^0\mathbf{r}_{\mathrm{i,d}}$.

A collision force of

$$\mathcal{F}_{ext} = \begin{bmatrix} 0 \\ -10 \\ 0 \end{bmatrix} \text{ N} \tag{3.9}$$

is applied on link 6 at time $t = 5$ s at a distance of

$${}^0\mathbf{r}_{\mathbf{6,c}} = \begin{bmatrix} 0.05 \\ 0.03 \\ 0 \end{bmatrix} \text{ m} \tag{3.10}$$

from the origin of frame 6, simulating the robot being hit on its outer shell.

In this way, the collision happens during steady state, when the robot is in the configuration shown in Figure 2.2, with a force entering the plane of the figure. The line of action is therefore orthogonal to the plane, and the minimum distance point is the

Uncertainty	Detection time [s]	Isolated link	Estimated point [m]	Estimated force [N]	Estimated momentum [Nm]
Nominal	5.001	6	$\begin{pmatrix}0.0509\\0\\-0.018\end{pmatrix}$	$\begin{pmatrix}-0.0067\\-9.9626\\0.0659\end{pmatrix}$	$\begin{pmatrix}0\\0.0168\\-0.0001\end{pmatrix}$
0%-5%	5.001	5	$\begin{pmatrix}0.0434\\0.0036\\0.0008\end{pmatrix}$	$\begin{pmatrix}0.8180\\-9.7024\\0.2118\end{pmatrix}$	$\begin{pmatrix}-0.0939\\1.1341\\-0.0300\end{pmatrix}$
5%-10%	5.001	5	$\begin{pmatrix}0.0825\\-0.0304\\0.0693\end{pmatrix}$	$\begin{pmatrix}-2.6560\\-9.2421\\-0.6377\end{pmatrix}$	$\begin{pmatrix}-0.9510\\-3.4667\\-0.3740\end{pmatrix}$
10%-15%	5.001	5	$\begin{pmatrix}-0.0818\\-0.0064\\0.0291\end{pmatrix}$	$\begin{pmatrix}0.8110\\-9.5440\\0.2171\end{pmatrix}$	$\begin{pmatrix}-0.0266\\0.3097\\-0.0065\end{pmatrix}$
15%-20%	2.145	2	$\begin{pmatrix}-1.0409\\-0.1687\\-5.8336\end{pmatrix}$	$\begin{pmatrix}-0.1862\\-0.0340\\0.0342\end{pmatrix}$	$\begin{pmatrix}-0.0057\\0.0310\\0.0001\end{pmatrix}$

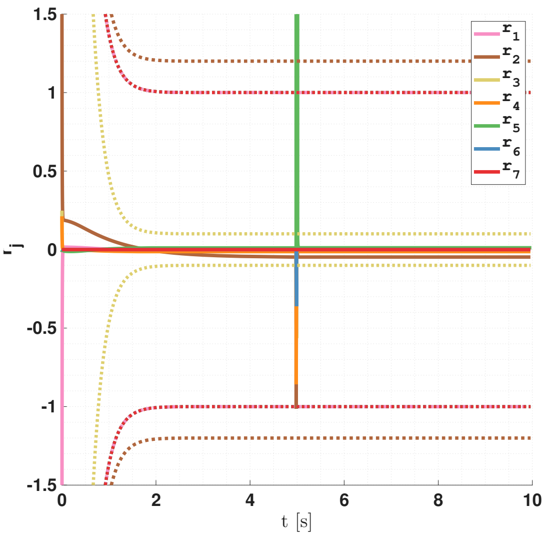
Table 3.1: Detection, Isolation and Identification results of a collision for 10 N off-origin collision force on link 6 at $t = 5$ s.

intersection of the line of action and the plane itself. In conclusion, the expected result is

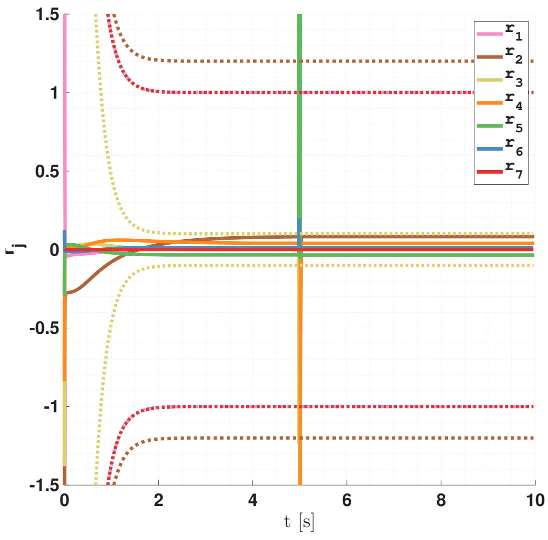
$$^{0}\mathbf{r_{6,d}} = \begin{bmatrix}0.05\\0\\0\end{bmatrix} \text{ m.} \tag{3.11}$$

The weak collision force is purposely chosen in such a way to make the joint configuration change the least possible, so that the returned result is as close as possible to the expected one.

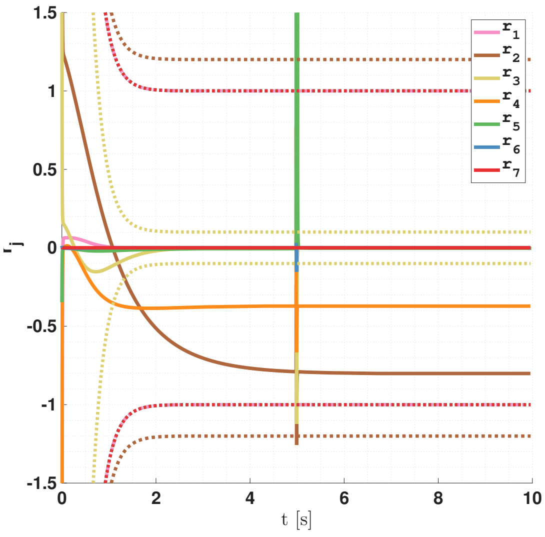
Results of detection, isolation and identification are reported in Table 3.1. In the nominal case, as shown in figure 3.10, the correct link is detected and the contact point and force are correctly isolated and identified. With up to 5% of uncertainty, the wrong link is isolated, but the contact point and force estimated are close to the real ones. Past 5% of uncertainty only the detection time is correct. At 20% uncertainty a false positive is triggered at about 2 s (Figure 3.11). The evolution of the joints is shown in Figure 3.12.



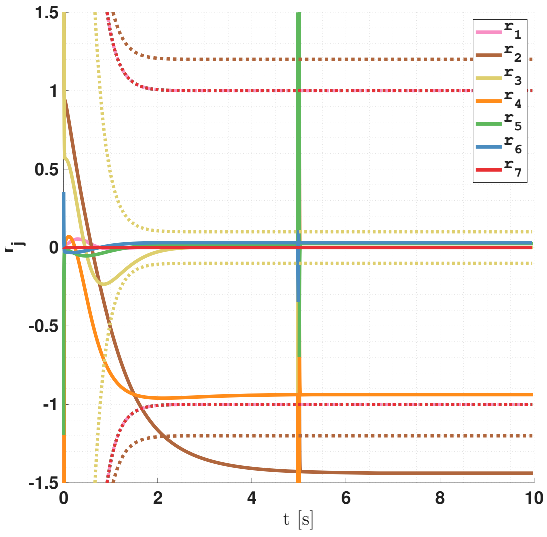
(a) Uncertainty: 0%-5%.



(b) Uncertainty: 5%-10%.

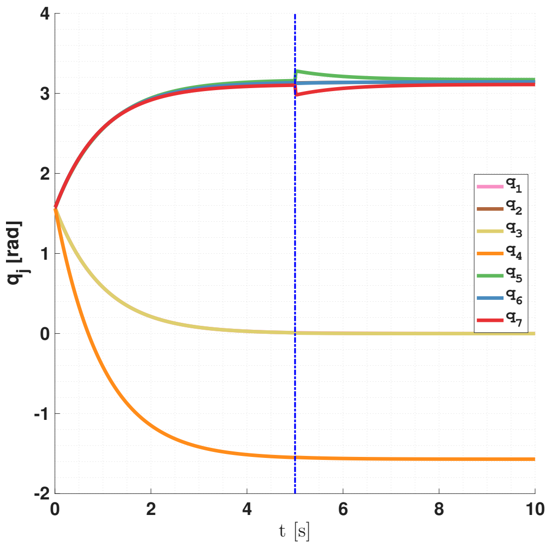


(c) Uncertainty: 10%-15%.

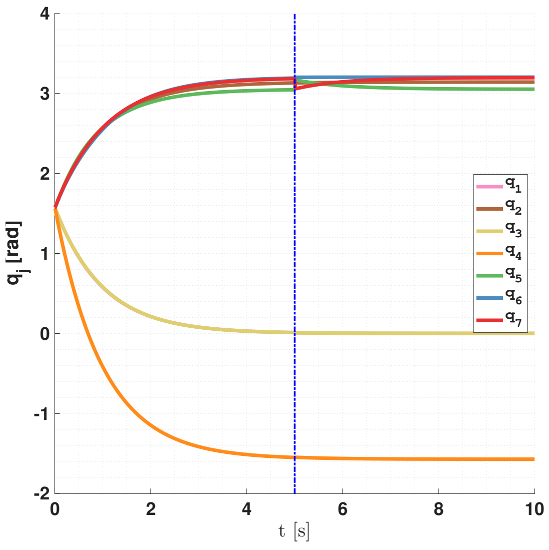


(d) Uncertainty: 15%-20%.

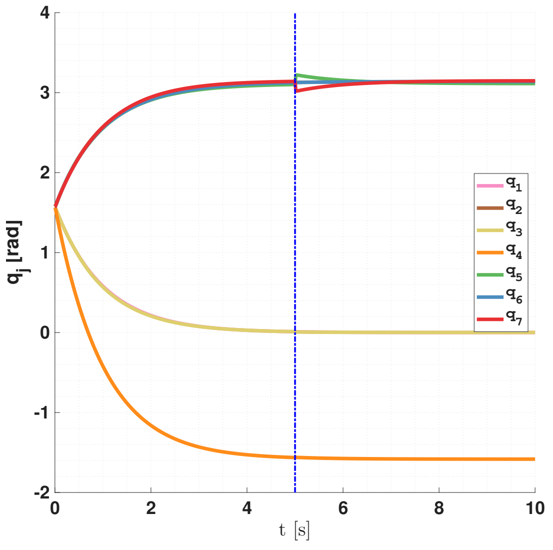
Figure 3.11: Residuals and thresholds for regulation task in uncertain conditions with off-origin collision force on frame 6.



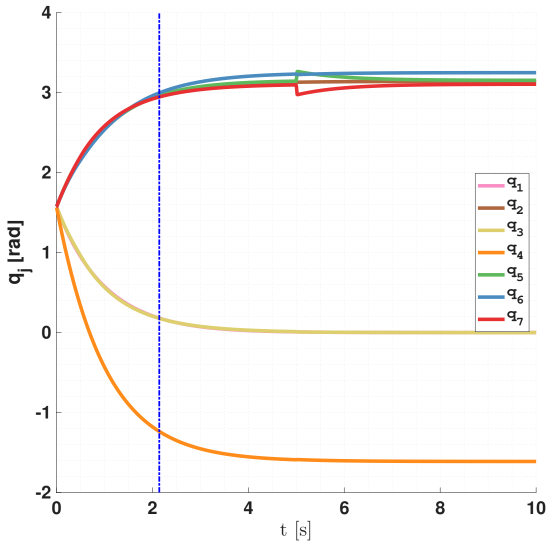
(a) Uncertainty: 0%-5%.



(b) Uncertainty: 5%-10%.

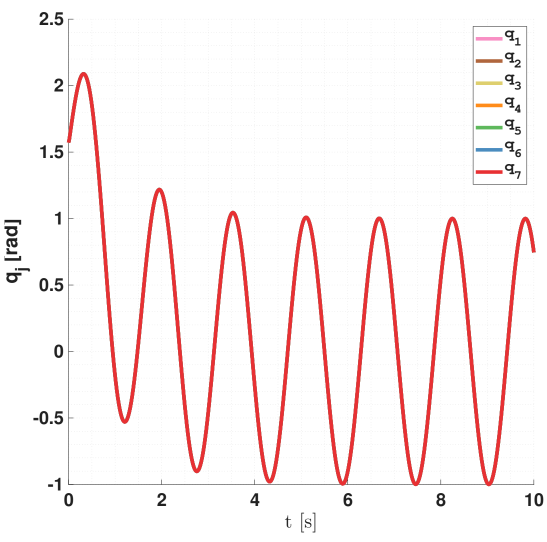


(c) Uncertainty: 10%-15%.

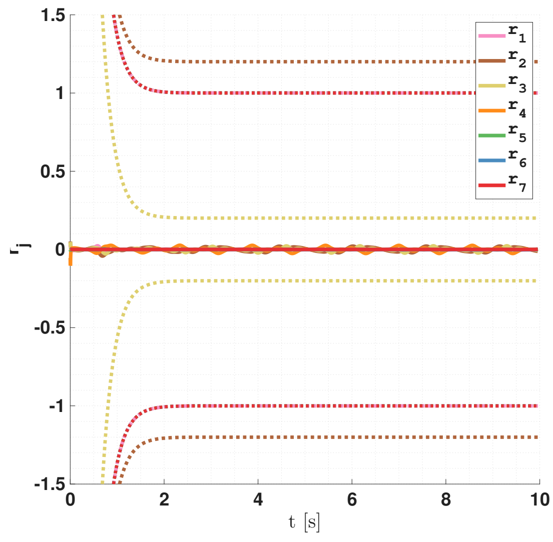


(d) Uncertainty: 15%-20%.

Figure 3.12: Joint positions for regulation task in uncertain conditions with off-origin collision force on frame 6.



(a) Joints.



(b) Residuals.

Figure 3.13: Trajectory tracking task with feedback linearization control in nominal collision-free conditions.

3.2.2 Trajectory tracking task

The second set of simulations involves collisions happening while the robot is performing the trajectory tracking task described in Section (3.1.1) (a sinusoidal in the joint space) under the controller 3.1. Joints and residuals evolution in nominal collision-free conditions are shown in Figure 3.13.

This time, two collisions occur sequentially in the same simulation, to highlight the ability of detecting multiple non-simultaneous collisions.

Sequential collisions on links 5 and 7

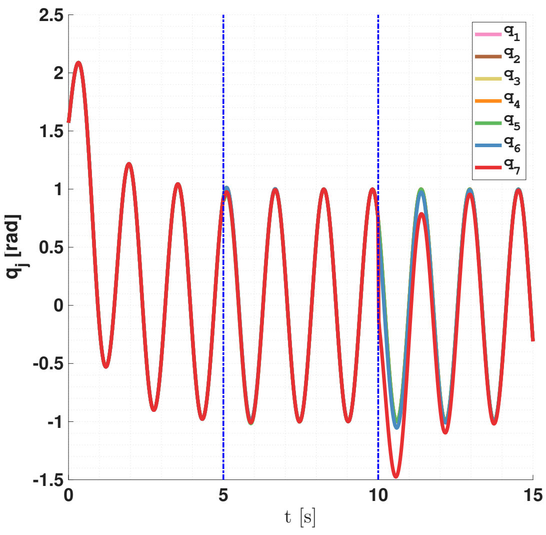
The two collisions are simulated, respectively, on link 5 at time $t = 5$ s with the external force (3.5) and on link 7 at $t = 10$ s with external force

$$\mathcal{F}_{ext} = \begin{bmatrix} 10 \\ 10 \\ 10 \end{bmatrix} \text{ N}, \tag{3.12}$$

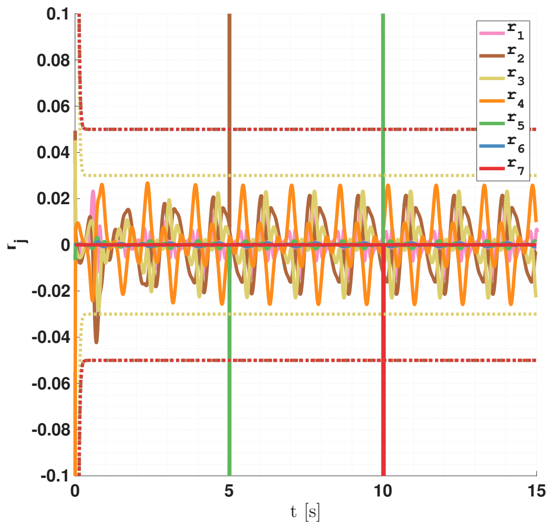
both at the origin of the respective frame.

In particular, the second collision simulates a contact on the end effector of the robot, showing that the residual method can also be used to estimate the contact force at the end effector without any additional sensor.

Figure 3.14 shows the results in nominal conditions; the two collisions are correctly



(a) Joints.



(b) Residuals.

Figure 3.14: Trajectory tracking task in nominal conditions with sequential collisions on origin of frame 5 and 7.

detected and the end effector force estimate is very accurate: it is obtained

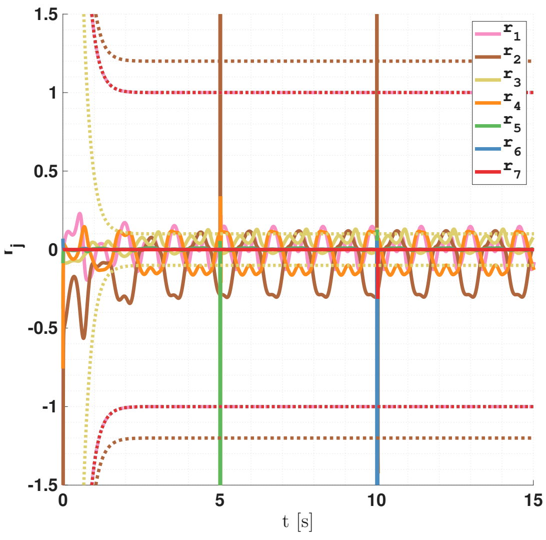
$$
{}^{0}\mathbf{r}_{7,\mathbf{d}} = \begin{bmatrix} 7.372 \\ 2.457 \\ -9.657 \end{bmatrix} \cdot 10^{-4} \text{ m} \tag{3.13}
$$

for the contact point and

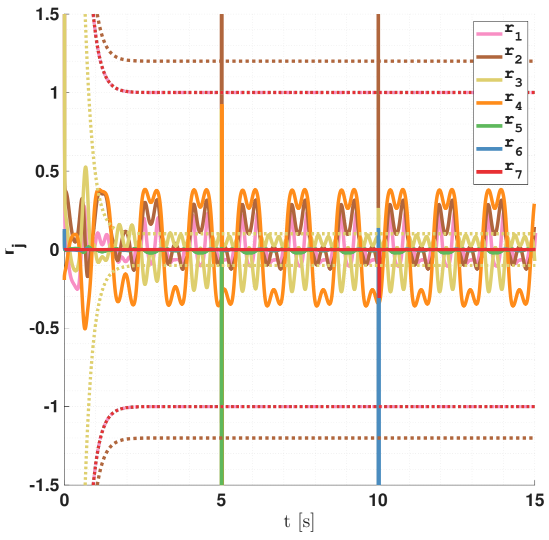
$$
\hat{\mathcal{F}}_{ext} = \begin{bmatrix} 9.8748 \\ 9.8512 \\ 10.0452 \\ 0.0014 \\ 0.0014 \\ 0.0014 \end{bmatrix} \begin{bmatrix} \text{N} \\ \text{Nm} \end{bmatrix} \tag{3.14}
$$

for the contact wrench.

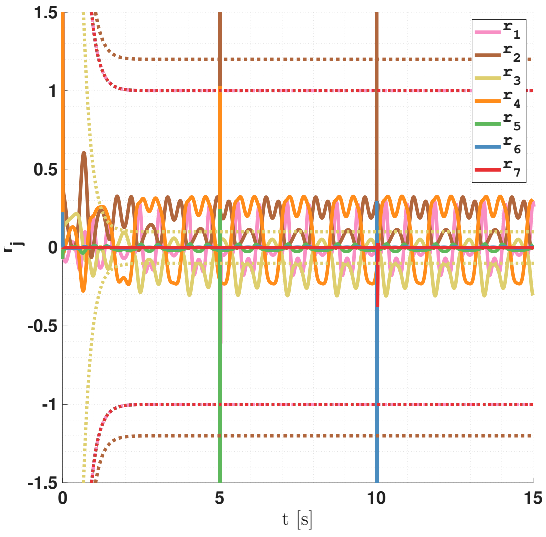
In presence of uncertainties (Figure 3.15 for the residuals and Figure 3.16 for the joints), instead, many false positives are generated, especially on link 3.



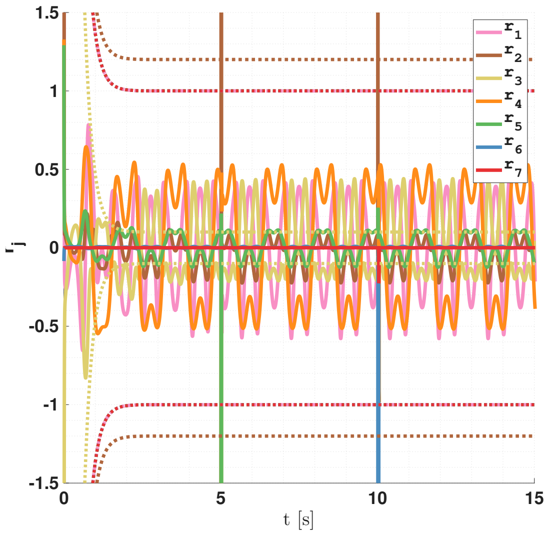
(a) Uncertainty: 0%-5%



(b) Uncertainty: 5%-10%

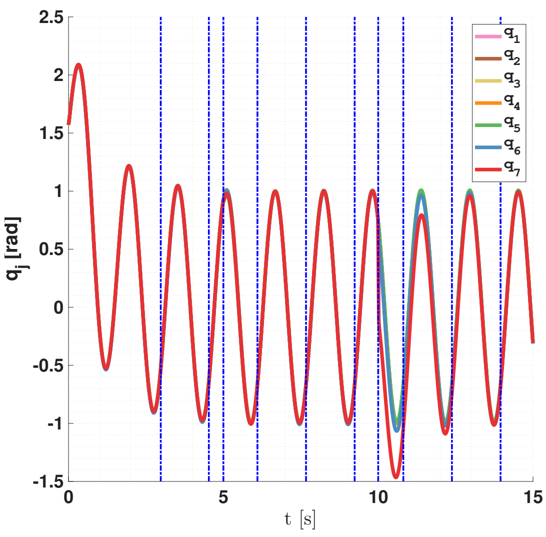


(c) Uncertainty: 10%-15%

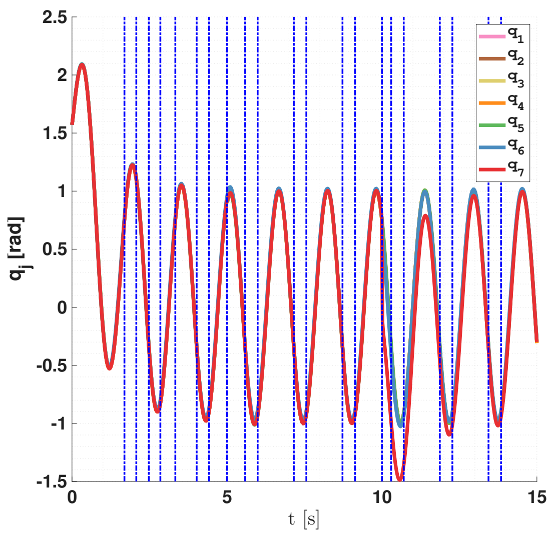


(d) Uncertainty: 15%-20%

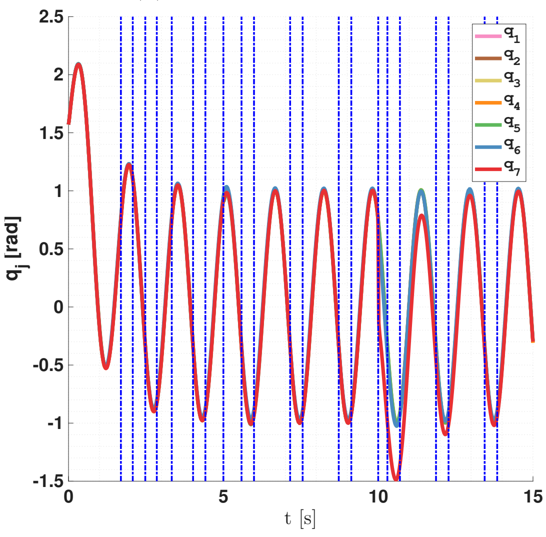
Figure 3.15: Residuals and thresholds for trajectory tracking task in uncertain conditions with sequential collisions on origin of frame 5 and 7.



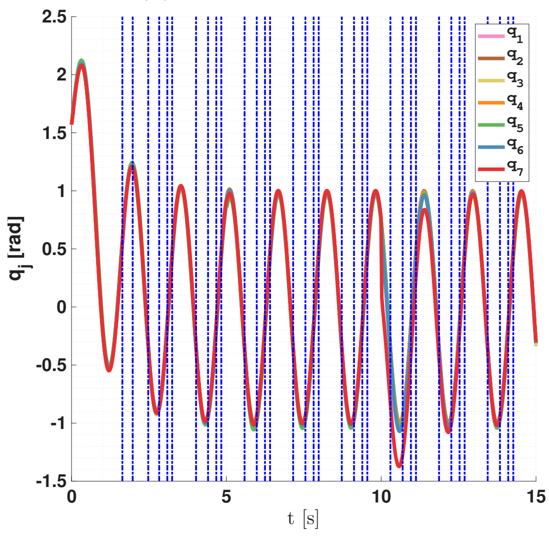
(a) Uncertainty: 0%-5%.



(b) Uncertainty: 5%-10%.



(c) Uncertainty: 10%-15%.



(d) Uncertainty: 15%-20%.

Figure 3.16: Joint positions for trajectory tracking task in uncertain conditions with sequential collisions on origin of frame 5 and 7.

Chapter 4

Conclusion

In conclusion, the momentum based residual method proved to be a reliable method for monitoring collisions on the robot YuMi by ABB, including detection, isolation and identification. Sequential collisions are also handled, as long as they are not simultaneous. In nominal conditions, reconstructing the correct collided link is always possible for suitably defined thresholds. However, as model uncertainty increases, false negatives and false positives begin to appear more frequently. The estimation of the contact point and of the contact force is always possible (out of singularities) if the collided link is at least the sixth, but it is unreliable under great degrees of uncertainties. This monitoring scheme was tested for a collision force with magnitude between 10 N $\leq ||\mathbf{f_{ext}}|| \leq$ 500 N and has a general response time of 1 ms, hence it is a reliable system for real-time collision reaction strategies.

Further developments towards an actual implementation on the real robot could include a more complete dynamic model, with viscous and static friction and actuator inertia, which were not considered in this work, and take into account control effort and saturation of the actuators.

Bibliography

[1] ABB. Single-arm YuMi® - IRB 14050. URL `https://new.abb.com/products/robotics/robots/collaborative-robots/yumi/single-arm`.

[2] W. Khalil, A. Vijayalingam, B. Khomutenko, I. Mukhanov, P. Lemoine, and G. Ecorchard. OpenSYMORO: An open-source software package for Symbolic Modelling of Robots. pages 1206–1211, July 2014. URL `https://hal.science/hal-01025919`.

[3] Sami Haddadin, Alessandro De Luca, Alin Albu-Schäffer. Robot Collisions: A Survey on Detection, Isolation, and Identification. *IEEE Transactions on Robotics*, 33(6): 1292–1312, 2017. doi: http://doi.org/10.1109/TRO.2017.2723903.

[4] X. Yang. OpenSYMORO: An open-source software package for Symbolic Modelling of Robots. URL `https://github.com/xu-yang16/symoro-python3`.