Robotics and Computer Integrated Manufacturing 57 (2019) 452–464

Contents lists available at ScienceDirect

Robotics and Computer Integrated Manufacturing

journal homepage: www.elsevier.com/locate/rcim

Optimization of the energy consumption of industrial robots for automatic code generation

Michele Gadaleta[a], Marcello Pellicciari[b], Giovanni Berselli[c,*]

[a] Department of Engineering "Enzo Ferrari" (DIEF), University of Modena and Reggio Emilia, Via Vivarelli, 10, Modena 41125, Italy
[b] Department of Sciences and Methods for Engineering (DISMI), University of Modena and Reggio Emilia, Via Amendola, 2, Reggio Emilia 42122, Italy
[c] Department of Mechanical, Energy, Management and Transportation Engineering (DIME), University of Genova, Via All'Opera Pia, 15/A, Genova 16145, Italy

ARTICLE INFO

Keywords:
Robot offline programming
Industrial robotics
Energy optimization
Virtual prototyping
Automatic code generation
Sustainable manufacturing
Industry 4.0

ABSTRACT

At present, energy consumption strongly affects the financial payback period of industrial robots, as well as the related manufacturing process sustainability. Henceforth, during both design and manufacturing management stages, it becomes crucial to assess and optimize the overall energy efficiency of a robotic cell by means of digital manufacturing tools. In practice, robotic plant designers and managers should be able to provide accurate decisions also aimed at the energy optimization of the robotic processes. The strong scientific and industrial relevance of the topic has led to the development of many solutions but, unfortunately, state of the art industrial manipulators are equipped with closed controllers, which heavily limit the feasibility and performance of most of the proposed approaches. In light of the aforementioned considerations, the present paper presents a novel simulation tool, seamlessly interfaced with current robot offline programming tools used in industrial practices, which allows to automatically compute energy-optimal motion parameters, thus reducing the robot energy consumption, while also keeping the same productivity and manufacturing quality. The main advantage of this method, as compared to other optimization routines that are not conceived for direct integration with commercial industrial manipulators, is that the computed parameters are the same ones settable in the robot control codes, so that the results can automatically generate ready-to-use energy-optimal robot code. Experimental tests, performed on a KUKA Quantec KR210 R2700 prime industrial robot, have confirmed the effectiveness of the method and engineering tool.

1. Introduction

Sustainable manufacturing may be defined as a production model where both present and future needs are contemporarily accounted for, implying that social, ecological and economic impacts should be quantitatively assessed and optimized [1–3]. Manufacturing companies aim at implementing sustainable manufacturing, in order to improve profitability, by reducing both consumption of resources and global expenses, as well as by satisfying the incoming ecological impact regulatory constraints. Additionally, the huge adoption of Industrial Robots (IR), needed to satisfy the ever-increasing requirements in terms of manufacturing quality, customization and flexibility, has further raised the necessity of improving the energy efficiency of robotic plants. In fact, IR Energy Consumption (EC) has a strong impact on the overall manufacturing costs and final products carbon footprint.

Within this engineering field, present researches dealing with the improvement of energy efficiency basically follow two different approaches:

- *Eco-efficient design methods* [4], which involve the introduction of new energy-efficient hardware [5] or the improvement/re-adaption of the plant lay-out;
- *Eco-efficient programming methods*, which offer the possibility to reduce the EC also on existing plants, with a minimum investment cost and an overall higher financial impact, due to the huge number of robots installed worldwide. This approach includes the optimization of the production scheduling [6–9] and the energy-optimal IR programming [10–13].

In the following, we focus on the latter approach and, in particular, on IR energy-optimal programming for its wider impact: in fact, as clearly underlined in the past literature [12], existing robotic plants consume on the order of hundreds of kWh per day, so that any EC reduction may lead to a very substantial cost and carbon footprint

* Corresponding author.
E-mail address: giovanni.berselli@unige.it (G. Berselli).

https://doi.org/10.1016/j.rcim.2018.12.020
Received 25 April 2018; Received in revised form 7 November 2018; Accepted 29 December 2018

decrease in the long term. The present work specifically addresses current programming practices and IR controllers closed architecture, in order to propose an approach which could be effectively implemented in industry. To this purpose, the following aspects should be taken into account:

- IR controllers are robust and easy to use, but their closed architecture inhibits the adoption of many control strategies, therefore eco-efficient programming methods should be specifically designed to be implemented into state of the art IR controllers.
- In most cases, IR motions are programmed offline via dedicated robot simulation packages, such as *Delmia Robotics* (from *Dassault Systèmes* [14]), *RobotStudio* (from *ABB* [15]) or *KUKA.Sim* (from *KUKA AG* [16]), which can simulate the IR movement in a virtual plant and, subsequently, provide (as a direct output) the robot code that can be readily uploaded onto the physical system. Some of these software are vendor-specific tools (i.e. they can simulate only IR produced and sold by one vendor), whereas others are conceived as general purpose simulators. In the latter case, these tools actually provide only an approximation of the real IR trajectory. To this purpose, the simulations can be highly improved by employing a vendor-specific plug-in, namely the so-called RCS module [17], which basically acts as a black-box model that computes the IR trajectories with the same proprietary algorithms employed in the physical controller. Last generation IR RCS modules and vendor proprietary simulation tools are able to compute also the EC but its optimization is unfeasible. It is relevant to note that, differently from several other works [10], such EC computation takes into account also the EC of the IR cabinet electronics, which cannot be neglected due to its relevant contribution.
- Robotic processes programming involves many concurrent decisions and the EC optimization should be performed in few minutes without requiring relevant modifications in the design workflow and, especially, in the IR programming practices. Ideally, the EC optimization should be integrated with the aforementioned robot simulation and offline programming packages, providing the automatic generation of energy-optimal robot code.
- Commonly, IR programmers are not totally free to impose a complex end-effector trajectory, the IR motions mostly employed in practice being Point-to-Point (PtP) programmed in the joint space and linear movements of the end-effector (also circular movements are easily programmable but they are very rarely used). For what concerns such commands, the robot simulation package computes an end-effector path that cannot be adjusted in any other way rather than dividing the path itself into series of smaller motions.
- For what concerns the IR motion laws, the only setting available to the programmer is a control on the maximum velocity and acceleration limits, which by default are set as maximum. Since, generally, any information about the IR energy consumption is not available, also skilled operators usually program their robots to follow a path in the quickest possible way, despite the fact that such execution speed may be actually unnecessary. For instance, it has been shown that, in a robotic cell composed of several robots, many IR spend a lot of their time in standstill mode of operation, thus wasting the electrical energy needed to provide the motor torque that counterbalance the IR own weight and power their controller [18]. This strategy is surely not energy-optimal nor required to assure the plant production rate. An energy aware robot programming would employ the correct value of velocity and acceleration parameters that allow to perform each operation within the right amount of time.

In particular, an effective method to reduce the EC of robotic cells has been proposed by the authors in [18], in which a linear time-scaling approach has been developed on the basis of an energy consumption model, that takes into account both mechanical and electrical contributions. Outcome of this approach is the so-called *Energy Signature*, namely a function describing the motion EC at varying time-scaling ratio. The IR energy signature shows that, for a given and fixed geometrical end-effector path, it is possible to quickly compute an energy optimal scaling parameter. This paper builds upon these previous results by providing three critical advancements:

- An improved and more accurate IR system mechatronic model, which accounts for the whole chain of components from the electrical energy source to the manipulator mechanical structure, including the robot cabinet;
- A new algorithm and engineering method, in which the Energy Signature concept has been extended and where the time-scaling has been substituted by a completely novel approach, based on the IR motion velocity and acceleration limits, that can be directly and easily set within state of the art IR controllers;
- An optimization method which is interfaced with a commercial robot simulation tool, thus allowing to automatically generate a robot code that can be practically employed in the physical system.

In summary, the difference of the method, as compared to other approaches (e.g. [18–22]), lays in the fact that the optimization results can be seamlessly applied also on state of the art IR, whose closed controller allows a limited parameters setting. In addition, final results have been validated on a real industrial manipulator, confirming the practical efficacy of the proposed procedure and the capability to provide better EC reductions as compared to previous methods.

The rest of this paper is organized as follows: Section 2 provides an in-depth review of the state-of-the-art and recalls methods and concepts which are useful for the scope of this paper; Section 3 presents an overall IR energy-model; Section 4 describes how the EC is computed, leveraging on the standard functionalities offered by the robot simulation tool *Delmia Robotics*; Section 5 reports about the optimization method and provides a numerical example; Section 6 provides experimental results, whereas Section 7 reports the concluding remarks and discusses about future directions of improvement.

2. Literature review on eco-programming methods and tools

Owing to the definition provided in the Introduction, the difference between *eco-design* and *eco-programming* methods (and related engineering tools), mainly lays in the time-frame and related financial investments on which these strategies are reasonably applicable: medium-long term with higher financial investment for the eco-design of novel robotic plants, as compared to the short term and reduced investment for eco-programming (or energy optimization of robotic processes), which is applicable to both new and existing plants.

Eco-programming (also termed SW-based methods [23]) leverages on the possibility of modifying the motion planning of single and/or multiple IR in order to reduce the EC. Unfortunately, as previously recalled in the Introduction, some SW-based methods proposed in the literature are actually unpractical (since theoretical energy-optimal motions may need non-standard routines to be fully implementable in the real industrial controllers), a huge amount of proposals has been presented in the past literature (see e.g. [23] for a review). SW-based methods can be classified as: *i)* trajectory optimization strategies, whose outcome is a modification of IR paths and/or motion profiles; *ii)* scheduling optimization, where the coordination of multiple IR motions at factory level is considered. In both cases, it is necessary to define accurate system models, capable of predicting the overall EC as

function of the motion parameters. In particular, it is self-evident that the EC prediction on a virtual prototype requires the capability to evaluate the IR dynamic behaviour in terms of inertial forces, motor torques and currents, along with the various sources of energy loss both in the IR drive system (inverters, DC-bus, and rectifier) and in the mechanical parts (reducers, etc.). Naturally, the more precise the models the better the outcomes of any optimization strategy, however, it must be noted that even less precise models can depict effective optimizations.

Accurate information about the robot EC can be retrieved using different methods, namely detailed open-source mathematical models, black-box vendor proprietary tools or online measurements. Open-source model-based approaches [8,11,18] enable fast EC optimizations, even if the EC prediction accuracy may be limited by the precision of the required model parameters. Some robot vendors confidentially provide such parameters, otherwise, system identification techniques are needed [24]. Robot vendor proprietary tools (e.g. RCS) [15,19] are accepted by industrial end-users as validation standard, however energy-optimization routines which require to iteratively communicate with the RCS module end up being very slow. Online measurements [10] have the main drawback to require the physical robot cell, thus being applicable only after the actual cell commissioning. In any case, it must be underlined that most of the energy optimization approaches found in literature could adopt any of the three methods. Regarding model-based approaches (either open-source or black-box), it is possible to simulate components belonging to different physical domains (i.e. the IR mechanical structure along with the IR electrical components). Beside the models described in the present paper, examples of detailed mechatronic approaches for EC prediction can be found in [25–28]. Once an EC prediction model is defined, the IR paths or its motion parameters can be optimized in order to be more energy efficient, with no actual need for hardware replacement. From a terminology standpoint, these optimization strategies may be applied to either PtP or Multi-Point (MP) trajectories, the main limit being the possibility to practically implement energy-optimal motions, which comply with the hard restrictions of state-of-the-art industrial controllers. As an attempt to possibly cope with such restriction, the *time scaling* approach has been presented in [18,25], leveraging on the idea that the EC can be reduced by scaling the operations execution time to an optimal value. With the exception of robotic processes representing the bottleneck tasks, which are usually of limited number within a robotic production line, in real life practice, such approach does not compromise the production rate and final cycle time, and it is applied to reduce idle times and/or speed up some slow motions. At last, when the EC of multiple IR (also in case of shared workspace) is accounted for, energy-optimal operation scheduling can be seen as viable eco-programming method. Practically speaking, when several IR operate simultaneously, the overall workload can be optimally scheduled in order to avoid queues and idle times (see e.g. [8] and [23] for a more in-depth review).

2.1. Background on IR programming methods and tools

From a rather wide point of view, which also considers the specific aspects of IR collaborative tasks with human co-workers [29], an “optimal” IR motion should account for several aspects, namely:

- *Feasibility*: the IR kinematic and dynamic constraints shall be satisfied (i.e. angular limits of the joints [30], actuators saturation in terms of torque, velocity and acceleration).
- *Safety*: collision avoidance shall be enforced at all time. In case of co-workers in a shared workspace, the velocity of every IR part may be limited in order to reduce risks arising from potential impacts with humans.
- *Optimality*: the IR motion should be optimized (once the first two criteria are met) in terms of specific cost functions, the most practically useful being time-optimality (to reduce cycle times, e.g. [31,32]) and energy consumption.

Naturally, for collaborative robotics applications within partially structured environments, real-time motion adaptivity may also become crucial [33]. Having in mind such aspects, as widely known, IR can be programmed by means of two different methods: *i)* by physically interacting with the real robot via the teach pendant/hand-guiding (*online*); *ii)* by virtual interaction, that is using a simulation and offline programming (OLP) software tool, with whom it is possible to verify and validate the robot code before the actual commissioning (*offline*). For what concerns *online methods*, hand-guiding (Fig. 1) is becoming a useful functionality for collaborative robots [34], which allows also inexpert users to simply program the IR end-effector through a defined path. Nonetheless, at the current state-of-the-art, precision positioning via hand-guiding is still an issue requiring further investigations [35]. Therefore, although less intuitive and more time consuming, the teach pendant is still widely used for several industrial tasks (e.g. assembly).

For what concerns *offline methods*, no physical robots neither auxiliary devices are required. They are indeed simulated offline by a software capable of accurately replicating the IR behaviour and, in particular, its motions in the 3D space. In practice, the operator defines and validates the process within the OLP tool, with the possibility to automatically generate the IR code ready to be loaded into the physical systems. OLP is usually adopted in complex scenarios, offering different advantages, such as:

- A digital simulation technology allowing to test different situations. Robot trajectories, plant layout and process sequencing can be evaluated in order to reduce (mostly by trial-and-error) cycle times or (as shown in the present paper) energy consumption;
- Verification and validation tasks performed within a virtual environment, so that programming errors can be identified without damaging the physical system;
- A programming phase that takes place independently from the realization of the physical system (i.e. also before its actual commissioning). Such engineering practice allows to heavily reduce the development times and the time-to-market, as well as to identify in advance errors and designs flaws;
- The possibility for operators to work offline, thus avoiding interaction with the potential dangerous zones of the physical cell;
- A robot code development phase, which is integrated into the OLP tool, making it possible to control the code quality, thus establishing best practices. In addition, the programming phase of complex operations is eased, thanks to the use of high level languages and graphical interfaces.

Robot simulation and offline programming tools have enabled the realization of the most challenging applications (e.g. the automotive body in white lines or complex part deburring), which otherwise would have needed long and expensive tuning on the real plants. As an example, Fig. 2 depicts an outlook of *Delmia Robotics* [14], the *Dassault Systèmes* OLP tool: on the bottom left, the graphical programming interface is visible; motions can be generated, modified and sequenced for later play back.

Despite its advantages, OLP is still not widely adopted as standard programming technique since it requires operators with specialized skills, as it often happens with simulation and virtual engineering. Some of the main OLP drawbacks are:

- Need of a complete CAD model of the whole cell. In several industrial design practices the cells are designed only in part, with low level of detail;
- Development of a simulation-ready model of the virtual plant: 3D CAD models, kinematic relations, physical interactions and peripheral devices behaviour must be defined by skilled users;
- State-of-the-art simulation software tools are not able to adequately simulate intelligent devices in the cell, such as PLCs, motor drives, controllers, etc. Auxiliary code, written for the specific application, is usually necessary to correctly simulate the process.

Fig. 1. Hand guiding programming for collaborative IR, [37].

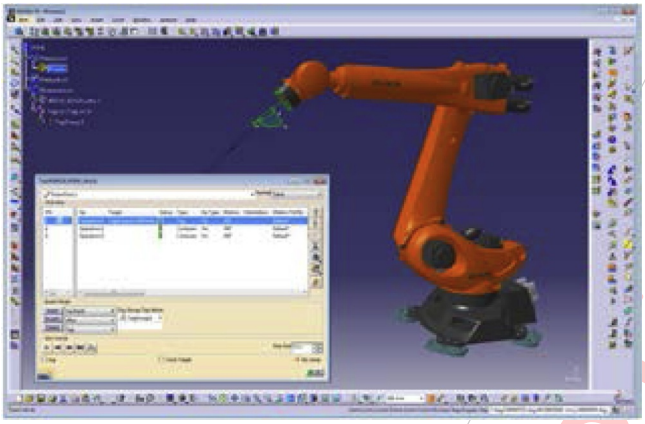

Fig. 2. Delmia Robotics robot OLP and simulation tool, [14].

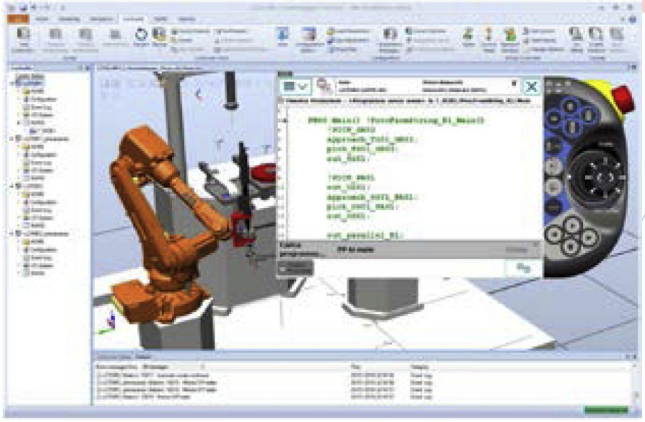

Fig. 3. ABB RobotStudio with its Virtual FlexPendant, [15].

In any case, simulation tools are continuously evolving and the Industry 4.0 paradigm is bringing the need of a full digital transformation, so that the adoption of robot offline programming and simulation tools is increasing [36], as well as their performance and features. Most of the major robot manufactures now offer a simulation software specifically created for the simulation of their robots, e.g. *RobotStudio* from *ABB* (Fig. 3) [15]. In any case, for what concerns the industrial scenario, the inclusion of energy assessment/optimization tools within such OLP environments surely represent a further step forward for the realization of better-behaved robotic work-cells.

3. Overall IR system modelling

Industrial robots are mechatronic machines constituted by mechanic, electric, and electronic components, all determining the total power consumption. For an accurate estimation of the total energy flow, a complete mechatronic model of the IR system is essential, modelling all influential peripheral devices. As depicted in Fig. 4, components which are taken into account are: *i*) the manipulator mechanical structure (including the weight balancer); *ii*) the Permanent Magnet Synchronous Motors (PMSM), along with their gearboxes and electromechanical brakes; *iii*) the drive system devices (rectifier, inverters, DC-bus and electrical brake resistor); *iv*) any auxiliary devices (e.g. control PC and cooling fans). The model described hereafter is accurate enough for retrieving an accurate EC prediction but also general enough to be used for most robots (i.e. with small or large payloads). The employed IR model computes the EC starting from the robot joints trajectories (e.g. computed within a robot simulation tool after the imposition of an end-effector path) and a series of data defining the mechatronic system. In the following, the dynamic model of each mechatronic component is described following back the system power flow.

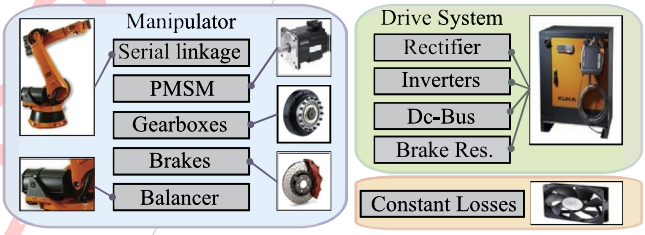

Fig. 4. Components contributing to IR energy consumption.

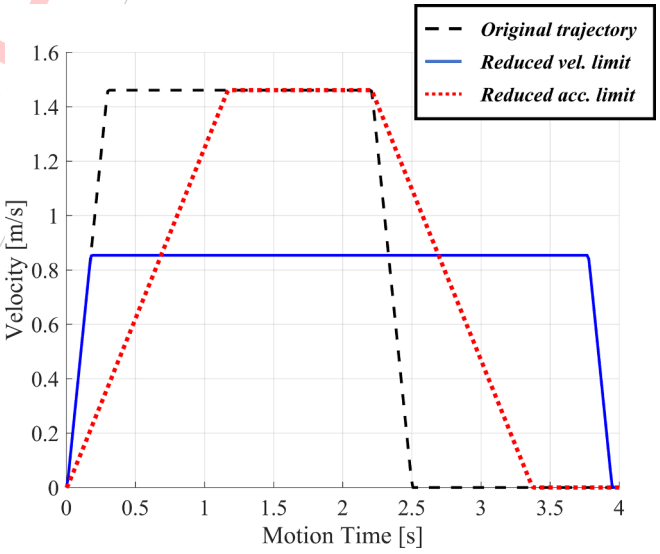

Fig. 5. Robot end-effector velocity when executing a linear motion with different velocity and acceleration limits.

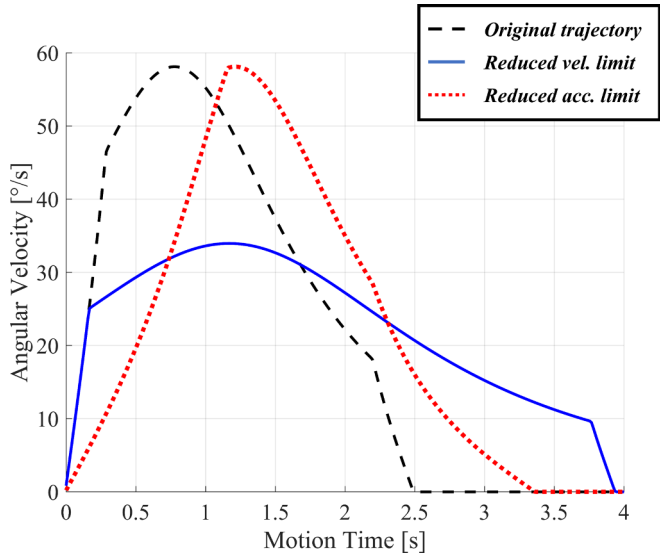

Fig. 6. Angular velocity of the first robot joint when executing a linear motion with different velocity and acceleration limits.

3.1. Mechanical components

The procedure for the EC computation starts within a robot simulation tool, where the user specifies an end-effector geometric path and imposes certain values for the maximum velocity and acceleration parameters, which are settable also in the controller when programming a real robot motion. As a simple example, let us consider an IR performing a linear movement.

In this case, Fig. 5 reports an example of end-effector speed for different values of velocity and acceleration limits: the black curve depicts the original trajectory, whereas the blue and the red curves respectively show a case where either velocity limit only or acceleration limit only are reduced, modifying the corresponding parameters in the robot code. Given the end-effector path and its velocity/acceleration limits, the robot simulation tool computes the IR joint trajectories by inverse kinematics. As an example, Fig. 6 reports the angular velocity of the first robot joint when executing the linear motions depicted in Fig. 6. Once the joint trajectories are known, the motor torque can be computed with standard procedures. In particular, a typical IR mechanical structure is composed by a series of rigid links connected by revolute joints whose torques can be computed using the well-known Lagrange formulation [38]. Indicating with $\boldsymbol{\tau_j}$ the 6 × 1 vector of joint torques and with $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$ and $\ddot{\boldsymbol{q}}$ the 6 × 1 vectors of joint position, velocity and acceleration, for each time instant, it is possible to write:

$$\boldsymbol{\tau_j} = \boldsymbol{H}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{\tau_g}(\boldsymbol{q}) + \boldsymbol{J}^T(\boldsymbol{q})\boldsymbol{f} \tag{1}$$

where $\boldsymbol{H}$ is the 6 × 6 symmetric, positive-definite inertia matrix. $\boldsymbol{C}$ is the 6 × 6 matrix such that $\boldsymbol{C}\dot{\boldsymbol{q}}$ is the vector of Coriolis and centrifugal terms. $\boldsymbol{\tau_g}$ is the 6 × 1 vector of gravity terms. $\boldsymbol{J}^T$ is the transpose of the 6 × 6 IR Jacobian matrix, $\boldsymbol{J}$, which is multiplied for the 6 × 1 vector, $\boldsymbol{f}$,of the external forces and torques applied on the end-effector.

In parallel, common anthropomorphic arms, due to the geometry of the system, are characterized by high torque loads acting on the second link. In fact, when the IR end-effector reaches the far most position from its base, torque demands on the second joint increase (in order to hold the IR gravimetric load). To reduce such undesired effect, especially when dealing with IR characterized by rather high payloads, a balancing mechanism is usually provided. The IR balancer is usually composed by either mechanical springs or hydro-pneumatic cylinders mounted between the first and the second links. With reference to Fig. 7, the balancing system can be modelled with the following equations:

$$\begin{cases} d = \sqrt{x_b^2 + y_b^2} \\ l_0 = d - r \\ \vartheta = q_{2,bal0} - q_2 \\ l_f = \sqrt{(d - r\cos\vartheta)^2 + (r\sin\vartheta)^2} \\ f_{sp} = k_{sp}(l_f - l_0) + f_0 \\ \tau_b = -f_{sp}\frac{dr\sin\vartheta}{l_f} \end{cases} \tag{2}$$

where x_b and y_b define the position of the spring mounting point on the first link with respect to the second joint axis, r is the spring force acting radius, $q_{2,\ bal0}$ is the second joint angular position in which the balancer torque is null, k_s is the spring constant and f_0 the spring pre-load. The torque τ_b is used to correct the second joint torque obtained with Eq. (1). As known, the required joint torques characteristics are not compatible with common electrical motors, so that gear reducers (generally with high reduction ratios) are employed. Henceforth, the torque, $\boldsymbol{\tau_m}$, and velocity, $\dot{\boldsymbol{q}}_{\boldsymbol{m}}$, required to the six motors can be computed as:

$$\boldsymbol{\tau_m} = \boldsymbol{G}^{-1}\boldsymbol{\tau_j} + \boldsymbol{\tau_i} + \boldsymbol{\tau_f} \tag{3a}$$

$$\dot{\boldsymbol{q}}_{\boldsymbol{m}} = \boldsymbol{G}\dot{\boldsymbol{q}} \tag{3b}$$

where $\boldsymbol{G}$ is a 6 × 6 gear reduction ratio matrix. The terms $\boldsymbol{\tau_i}$ and $\boldsymbol{\tau_f}$ stand for 6 × 1 vectors of torque contributions due to inertias and frictions, which can be computed as:

$$\boldsymbol{\tau_i} = (\boldsymbol{J_g} + \boldsymbol{J_m})\ddot{\boldsymbol{q}}_{\boldsymbol{m}} \tag{4a}$$

$$\boldsymbol{\tau_f} = \boldsymbol{K_c}\,\mathrm{sign}(\dot{\boldsymbol{q}}_{\boldsymbol{m}}) + \boldsymbol{K_v}\dot{\boldsymbol{q}}_{\boldsymbol{m}} \tag{4b}$$

where $\boldsymbol{J_g}$ and $\boldsymbol{J_m}$ are, respectively, 6 × 6 matrices of the gears and the motors inertias measured at the motors shaft axis, whereas $\boldsymbol{K_c}$ and $\boldsymbol{K_v}$ are 6 × 6 matrices of Coulomb and viscous friction coefficients also referred to the motors shaft. To obtain better results, $\boldsymbol{K_c}$ is not constant but related to the load with:

$$\boldsymbol{K_C} = \boldsymbol{A}\ abs(\boldsymbol{G}^{-1}\boldsymbol{\tau_j}) + \boldsymbol{B} \tag{5}$$

where $\boldsymbol{A}$ and $\boldsymbol{B}$ are 6 × 6 matrices of the load dependent Coulomb friction parameter. The *abs()* function computes the element-wise absolute value of its argument.

3.2. Electric motors

Due to their high dynamic characteristics, Permanent Magnet Synchronous Motors (PMSM) are typically used in industrial robots [39]. These motors generally present permanent magnets on the rotor and three-phase windings on the stator, which require to be correctly controlled by the drive system. For purposes of this work, a lumped parameter model of an equivalent DC-motor is used, the dynamic behaviour of each of the IR motors being governed by the following equations:

$$\begin{cases} V_m = RI_m + R_c I_{m,c} \\ V_m = RI_m + L\frac{dI_{m,t}}{dt} + pK_v\dot{q}_m \\ \tau_m = K_t I_{m,t} \\ I_m = I_{m,c} + I_{m,t} \end{cases} \tag{5}$$

where V_m and I_m are the motor equivalent voltage and current, R, L and R_c are *respectively* the stator resistance, the stator inductance and the core resistance used for efficiently model PMSM core losses, similar to [40]. The terms $I_{m,\ c}$ and $I_{m,\ t}$ indicate the current flowing through the core resistance and the current generating the torque. At last, the terms K_v, K_t and p respectively indicate the back emf constant, the torque constant and the number of pole pairs. As shown in [40], this equivalent model parameters are related to the real PMSM phase parameters by a multiplying factor of 3/2.

Starting from required motor shafts torques and velocities computed

from Eq. (3a and b), it is possible to calculate the required motor equivalent voltage and current solving the system of Eq. (5). Then, the instantaneous power, P_m, delivered to the motor is simply given by:

$$P_m = V_m I_m \tag{6}$$

Note that, for security issues, common IR motors are equipped with normally closed brakes, that are kept opened during the motions and released (after a predefined period, t_b) whenever the robot is stationary. During operation, a little power flow, P_{br}, excites some solenoids keeping the brakes open. This energy flow is automatically stopped in case of power failures, thus causing the brakes release and avoiding dangerous situations. Whenever the brakes are closed, the IR is kept stationary and the PMSM do not consume energy. Theoretically, a reduction of t_b always leads to an energy benefit. Nonetheless, some practical limitations should be accounted for, such as:

- IR brakes can naturally provide a limited life before failure (i.e. a limited number of switching cycles). Therefore, the time period t_b cannot be decreased over a certain threshold, since such reduction would increase the number of brake switches, thus reducing the IR lifecycle.
- Brakes switching produces vibrations that may not be acceptable in some situations [19].

In any case, the power consumption to keep the brakes opened will be taken into account hereafter for the computation of the total IR energy consumption.

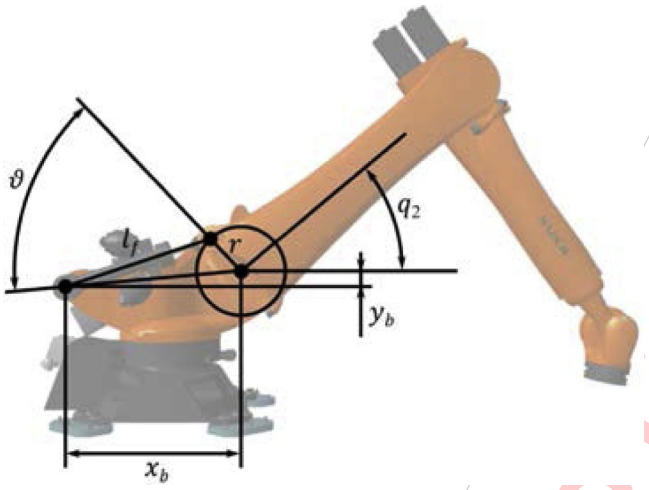

Fig. 7. Gravity balancer system and its model scheme.

3.3. Drive system

The drive system comprises the group of components between the mains (i.e. the electric energy source) and the motors, with the aim of correctly modulating the electrical power for motors alimentation. A schematic diagram for the drive components and their connections for an actual IR is given in Fig. 8 and explained in the following. Starting from the three-phase mains, a rectifier creates a common DC-bus from which the six motors draw the required power modulated through six inverters. This architecture allows for energy exchange between motors, in fact during braking phases, energy is pumped into the DC-bus and used by other motors or stored into the capacitance causing an increasing of the DC-bus voltage. For security issues, a braking resistor intervenes whenever the DC-bus voltage exceeds a predefined limit, thus dissipating energy and restoring the voltage within a secure threshold. The abovementioned drive components dynamic behaviour can be predicted using complex and very accurate models [41,42], which however require a deep knowledge of their electrical and logical architecture, generally unknown for industrial robots. For the purpose of EC prediction, simpler yet accurate models have been developed and briefly reported hereafter.

The inverter model is based on the simple power balance:

$$P_{inv} = P_m + P_{loss,inv} \tag{7}$$

where the inverter input power, P_{inv}, is the sum of the power required by the connected motor, P_m, computed with Eq. (6), and the inverter losses, $P_{loss,\ inv}$, computed as:

$$P_{loss,inv} = K_c I_m^2 + K_{sw}|I_m| \tag{8}$$

where the terms K_c and K_{sw} are the conduction and switching-losses coefficients used to compute the respective losses terms, both related to the motor current I_m computed in Eq. (5). With the knowledge of the power required by each inverter, it is possible to estimate the behaviour of the DC-bus, namely its voltage, V_{dc}, the energy stored into the capacitance, E_c, and the energy dissipated through the braking resistance, E_r. The adopted governing equations are:

$$P_{dc} = P_{inv,tot} + P_c + P_r \text{ where } P_{inv,tot} = \sum_{i=1}^{6} P_{inv,i} \tag{9}$$

$$P_c = \begin{cases} 0 & if\ V_{dc} < V_{dc,min} \\ -P_{inv,tot} & if\ V_{dc,min} \le V_{dc} \le V_{dc,min} \\ 0 & if\ V_{dc} > V_{dc,max} \end{cases} \tag{10}$$

$$P_r = \begin{cases} 0 & if\ V_{dc} \le V_{dc,max} \\ -P_{inv,tot} & if\ V_{dc} > V_{dc,max} \end{cases} \tag{11}$$

$$E_c = \frac{1}{2} C V_{dc}^2 = \int_{t_0}^{t} P_c dt \tag{12}$$

where P_{dc} is the power entering the DC-bus, P_c the power flowing into the capacitance, P_r the power dissipated through the brake resistance, and $P_{inv,\ i}$ is the power required by the i-th inverter. The DC-bus minimum voltage $V_{dc,\ min}$ is imposed by the mains AC-grid; for standard European three-phase 400 V AC-grid, $V_{dc,\ min} \approx 565\,V$. The DC-bus maximum voltage depends on the robot system and it is usually set around 700 V. The energy stored into the capacitor, E_c, depends on its capacitance, C. As initial condition, it is reasonably assumed $V_{dc}(t_0) = V_{dc,min}$, meaning no energy stored inside the bus capacitance. A schematic depicting the qualitative DC-bus voltage behaviour as imposed by Eqs. (9–12) is shown in Fig. 9.

Now, similar to the inverters, the rectifier input power, P_{rec}, is computed correlating its losses to the current flowing into the DC-bus, I_{dc}:

$$I_{dc} = P_{dc}/V_{dc} \tag{13a}$$

$$P_{rec} = P_{dc} + P_{loss,rec} \tag{13b}$$

$$P_{loss,rec} = K_c I_{dc}^2 + K_{sw}|I_{dc}| \tag{13c}$$

Using this model, the power flow computation concerning rectifier and each inverter requires two parameters, namely K_c and K_{sw}, which can be identified using identification techniques, described in e.g. [43] and related references.

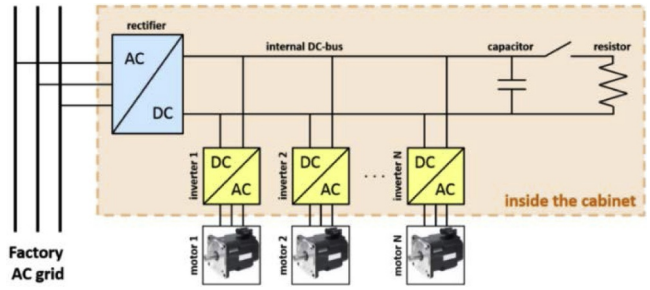

Fig. 8. Schematic of the drive system components.

3.4. Other constant losses

The components described above can predict the power flow from the mains to the mechanical linkage system. However, for a correct EC prediction, the model must account also for other energy losses, e.g. cooling fans and electronics. Generally, the power consumption of these devices is considered constant. On the other hand, two different states are considered hereafter, which are correlated with the brakes state. In fact, when brakes are closed, parts of the system are automatically switched to a standby mode, consuming considerably less energy.

$$P_{const} = \begin{cases} P_{standby} & \text{if brakes are closed} \\ P_{operative} & \text{if brakes are opened} \end{cases} \tag{14}$$

Finally, the total IR power consumption, P_{ir}, is obtained as:

$$P_{ir} = P_{rec} + P_{const} \tag{15}$$

The total energy that the IR draws from the mains in the time interval [0, T] is, then, given by:

$$E_{ir} = \int_0^T P_{ir}(t)dt \tag{16}$$

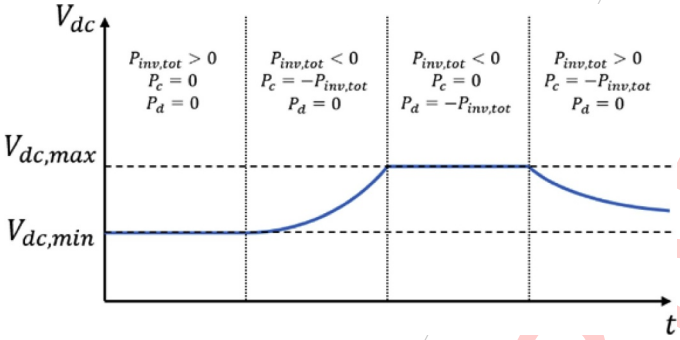

Fig. 9. Behaviour of the robot DC-bus voltage in different situations.

4. Computation of a single motion energy consumption

4.1. The generalized motion

Usually, a production process requires robot motions with fixed duration, e.g. when the preceding and following motions are process relevant and not modifiable. In case the robot is held in standstill, the motors have to produce the required holding torque until the brakes are released. The energy consumed during the waiting time must be taken into account for a correct energy optimization, so that a convenient approach is to conceptually consider each robot motion as a sequence of a moving phase and a steady phase. Doing so, the EC of this generalized motion can be computed as the sum of the two phases, such that:

$$E_{ir,gmov} = \underbrace{\int_0^{T_{mov}} P_{ir}(t)dt}_{E_{ir,mov}} + \underbrace{\int_{T_{mov}}^{T_{fix}} P_{ir}(t)dt}_{E_{ir,ste}} \tag{17}$$

where T_{mov} and T_{fix} are, respectively, the duration of the moving phase and the total fixed available time for the motion. Starting form this general case, motions without the steady phase or without the moving phase can be easily obtained respectively setting $T_{fix} = T_{mov}$ or $T_{mov} = 0$. Obviously, the condition $T_{mov} \leq T_{fix}$ must be always verified, otherwise the motion is considered infeasible.

4.2. Trajectory computation

Due to typical restrictions of actual IR controllers, it is not possible to execute a generic trajectory. The programmers make use of the basic commands provided by the robot manufacturers for defining the spatial path that the robot end-effector has to follow, although having no possibilities to exactly impose the evolution in time of such motion (trajectory). The trajectory can be approximately modified by setting several motion parameters (with specific code), such as maximum velocity and acceleration limits, but the exact relation of these parameters with the trajectory formula is unknown and protected as trade secret by the robot manufacturer. In the past, robot programmers considered this restriction as a simplification; common practice was to run the robot at the maximum speed allowed by the process with the only intent to comply with the imposed cycle time. In parallel, the predictive simulation of the production line became more and more important leading to the definition of the Realistic Robot Simulation (RRS) interface, [18]. Each manufacturer provides a *Robot Controller Simulation* (RCS) specific module, which works as a black-box computing the trajectories with the same algorithms used in the real controller. Today the RRS-interface is the world-wide de-facto standard for precise simulation of robot motion behaviour. In this work, *Delmia Robotics V5* (from *Dassault Systèmes*) was used as simulation environment taking advantage of the accurate trajectory computation retrieved by the RCS module relative to a *KUKA KR C4* controller connected to a *KUKA Quantec KR 210 R2700 prime* robot. After the simulation within *Delmia*, information about IR joint trajectory can be exported into a file.

4.3. Energy consumption computation

With reference to Fig. 10a, a Matlab implementation of the IR model described in Section 3 uses the *Delmia Robotics* simulation data to compute the IR moving phase EC and corrects it by summing the steady phase EC (computed on the basis of the available time for the motion). Since a single motion is analysed, a correct initial condition of the system must be imposed: it is assumed that at the initial instant no usable energy is stored into the DC-bus capacitance, so that the system is stationary. This hypothesis is the most logical one if only one motion is considered, since it is impossible to determine the initial state without simulating the entire robot cycle (planned in future works).

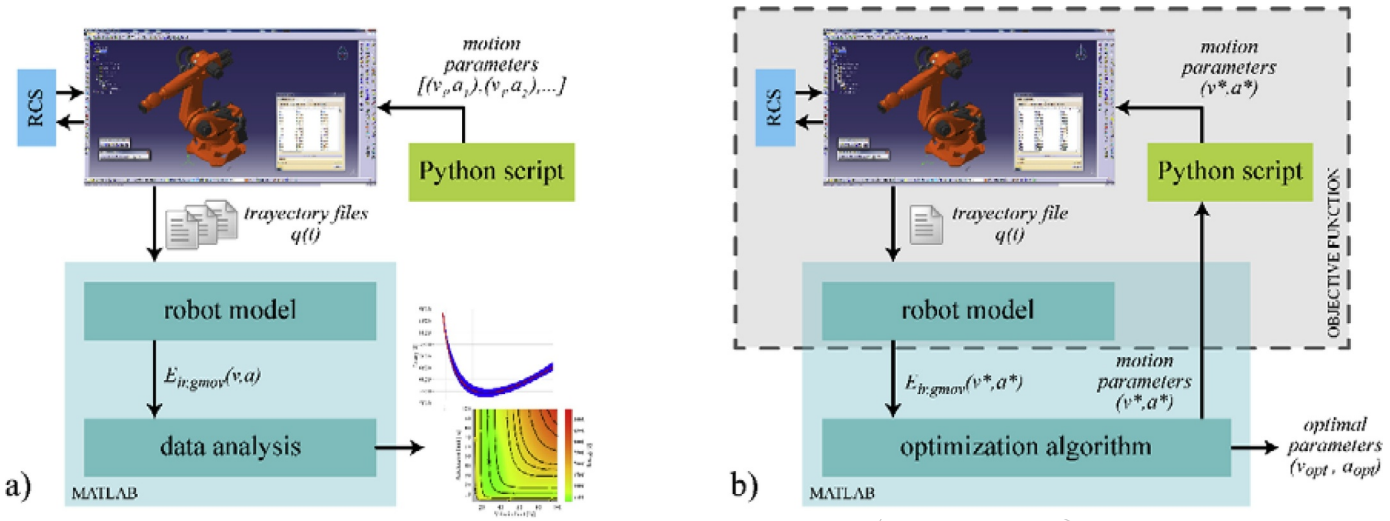

Fig. 10. Schematic of the procedure for motion parameter analysis (a) and optimization (b).

5. Determining optimal motion parameters

As mentioned before, robot tasks are actually programmed using motion commands provided by the manufacturer, which define the interpolating trajectory between determined targets. Each command requires the definition of some parameters to be set by the programmer in function of the desired motion characteristics. The actual programming trend is uniquely based on a time criterion, so that the task motions parameters are chosen as to complete the desired operations within the imposed cycle time. No attention is usually paid to the robot EC mainly because of the lack of adequate tools for assisting the designer on this purpose. Only in last years, with the continuously increasing attention to energy efficiency, some simulation software (e.g. *ABB RobotStudio* [16]) implemented the computation of an estimate of the robot energy consumption, however without providing any built-in optimization routine. In order to fill this lack, a tool is developed which integrates with state of the art simulation software, with the aim of assisting the programmer in an accurate choice of the motion parameters. Respect to other solutions (see [23] for an adequate literature review), this tool focuses on ease of use and seamless integration with state of the art IR controllers, in order to generate ready-to-use robot code which fully implements the energy-optimal trajectories calculated. To this purpose, as said, two commonly settable parameters have been chosen for their effectiveness and frequent use, namely the velocity and acceleration limits of the motion, although the method could be easily extended considering every other freely adjustable parameter.

The tool is conceived for real-life industrial use, and it is designed to support robot engineers to quickly optimize the energy consumption of robot motions, without requiring high-level expertise and with a limited effort and computing time. Furthermore, the tool can optimize the EC of any robot motion, even with a fixed motion time, with the only exception of those subjected to velocity constraints (e.g. gluing or laser welding paths, where the end effector must be very accurate in keeping a constant velocity), while it could also be integrated in energy-optimal operation scheduling tools, providing further gains.

5.1. Motion parameters analysis

As an example, a standard linear motion has been analysed varying the imposed limits of velocity v, and acceleration, a. At first, a series of simulations has been executed in *Delmia Robotics* changing both v and a on a predefined grid of combinations. The process has been automated using a Python script, which repetitively sets the parameters within *Delmia Robotics*, runs the simulation and exports the results into a file. Secondly, for each generated file, the generalized motion EC has been computed as explained in Section 4. Thirdly, the obtained information has been analysed and graphically arranged to be better interpretable. In Fig. 11, data are presented as a combination of a colour map indicating the motion EC and an isoline contour map showing the moving phase time duration.

The corresponding limits of (maximum) velocity and acceleration are expressed on the axes as percentage of the robot maximum ones. This kind of plot clearly shows the dependence of the energy consumption on the imposed parameters and permits to find the optimal ones, indicated by the "X" point. It is interesting to see that, for a given motion time, the EC can be reduced by simply setting the velocity and acceleration limits with the command instructions available in any IR control code. Furthermore, the proposed method can be used to enhance the EC saving potentials not only for a single task (as done in this paper) but also for a complete cycle, composed of multiple motions and IR standstill times, previously modified by leveraging on the traditional time-scaling approach described in [18]. For comparison purposes, in Fig. 12, the data are also arranged into another form. The red line is the *Energy Signature* of the analysed motion obtained with a linear trajectory scaling of the fastest possible motion [18]. The blue area indicates the motion times and energy consumptions obtainable changing the velocity and acceleration limits. It is evident that a better minimum of energy can be reached tuning the motion parameter instead of linearly scaling the trajectory. The *Energy Signature* of a robot motion has been defined in [44] as the analytical formulation of the robot EC as function of the task execution time, and it has been successfully exploited for time-scaling approaches [18]. On the other hand, in this paper, the concept of Energy Signature is radically extended, defining it as the representation of the robot EC as function of any motion and/or IR parameters (in the present work such parameters being the maximum velocity and acceleration limits). It is important to underline that the proposed approach is better performing in terms of EC reduction as well as it is much easier to adopt in industrial practice, since it has been natively conceived to be seamlessly implemented within state of the art industrial controllers. In fact, in other relevant works [10], energy-optimal trajectories can be realized only with robot vendor unique features (i.e. *Emily driver*, available only for KUKA robots) [45], which introduce drastic limitations in the usability of the code and are very rarely used in industrial practice, only for unconventional tasks. Furthermore, the proposed approach could be used to compute the EC with either the model-based approach previously described, but also (alternatively) by either exploiting the robot vendor proprietary "black box" tools or any other method retrieving an adequate EC prediction.

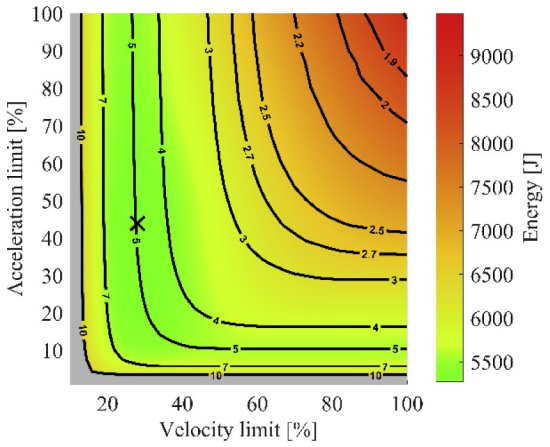

Fig. 11. Energy consumption (colour map) and motion time (isolines) as function of velocity and acceleration limits. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

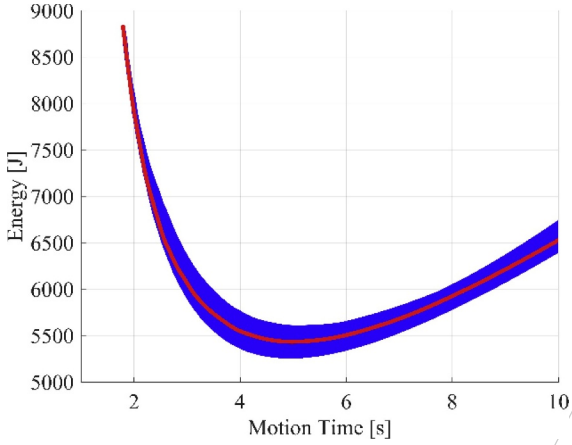

Fig. 12. Energy Signature (red line) compared to possible EC variations when varying velocity & acceleration (blue area). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

5.2. The optimization tool

With the just described motion analysis an enormous quantity of information has been retrieved, which permits an accurate understanding of the motion EC. However, usual practical cases only require the knowledge of the energy-optimal parameters. To reach this intent with the minimal computational effort, an optimization procedure has been created, the conceptual scheme being illustrated in Fig. 10b. The software package *Matlab* controls the optimization process through a pattern search algorithm aimed at minimizing the generalized motion EC with the nonlinear constraint of the fixed motion duration. The objective function involves a Python script for setting the optimization parameters in *Delmia Robotics*, run the simulation and export the trajectory data file, which is used for the energy computation (as described in Section 4). At the end of the process, the optimal motion parameters are retrieved. The optimization of the motion previously analysed requires 33 simulation loops and produces results matching those derivable from the map in Fig. 11 confirming the validity of the approach. It is interesting to underline that 98% of the computational time is spent for the computation of the trajectory. This is principally due to the limited functionalities offered by *Delmia Robotics* in the automation of the simulation process. Furthermore, not much attention has been devoted to the script efficiency, since this objective goes beyond the purposes of this work. On the other hand, an integration of these procedures directly into the simulation software can enormously increase the computational efficiency offering a useful tool for an energy efficient programming.

6. Experimental validation

6.1. Measurement method and tools

In order to validate the results obtained via the proposed optimization tools, a set of experiments have been executed on an industrial robot *KUKA KR210 R2700 prime,* one of the most common robot used in automotive body in white assembly lines. Experimental assessments have been made by measuring the electrical power flow through the wires between the IR cabinet and the mains grid, so as to take into account the overall IR energy consumption. In particular, the measurement of three-phase electrical loads to retrieve the power consumption requires the concurrent estimation of voltages and currents for each line. With reference to Figs. 13 and 14, three active differential probes *Testec TT-SI 9002* have been used to reduce the mains grid voltages (400 V AC) to acceptable values for an acquisition device, i.e. in the range ± 10 V. In addition, current measurements have been performed via three clamps *Chauvin Arnoux PAC 22*, namely Hall-effect current clamps allowing to estimate direct and alternating currents up to 1400A, with a bandwidth up to 10 kHz. Measured currents are transformed into proportional voltages in the range ± 10 V. Analog signals coming from differential probes and current clamps have been acquired using a *Data Translation DT9826* acquisition module, that allows to monitor 8 analog signals, in the range of ± 10 V, with 24 bit resolution, at a maximum frequency of 41.666 kHz. Digitalized data are transferred to a PC using an USB connection. The acquisition module can be configured and controlled through the connected PC using different software tools provided by the module manufacturer. All equipment and other accessory devices (transformers, connectors, etc.) have been arranged into a practical box (Fig. 13), equipped with connectors allowing an easy installation between the IR and the mains grid, connections being illustrated in the schematic of Fig. 14. In practice, current clamps measure the current flowing into each line, while the differential probes measure the tension between lines and neutral wires. Transduced signals in the range of ± 10 V are acquired by the measurement module with a sampling rate of 40 kHz and sent to the PC.

The adopted equipment retrieves as output the instantaneous values of voltage and current for each line, which need to be elaborated to obtain the information of interest. Starting from the measurements of voltage, v_l, and current, i_l, for each line, $l = 1, 2, 3$, the total instantaneous power consumption can be obtained as:

$$P_{tot} = \sum_{l=1}^{3} P_l = \sum_{l=1}^{3} v_l \cdot i_l \tag{18}$$

In most of the currently industrially adopted electric devices, the electrical power enters the system through a diode rectifier. The effect of the diodes causes impulsive flowing of the currents that reach high values in short periods of time. Accounting for this effect, to obtain good measurements, a high acquisition frequency is required: data have been acquired synchronously with a frequency of 40 kHz.

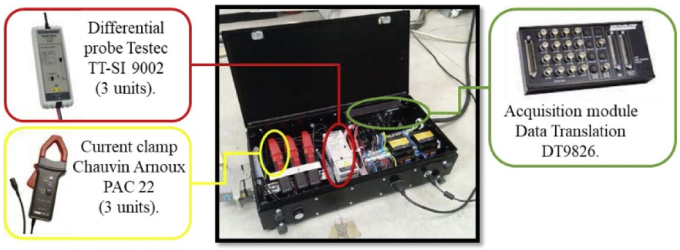

Fig. 13. Equipment for electrical measurements arranged into an "energy measurement box".

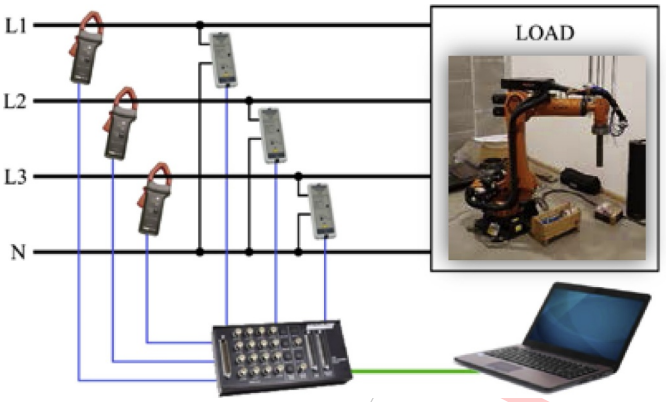

Fig. 14. Schematic of the electrical connections inside the measurement box.

6.2. Experimental tests

Actual experiments have been performed by enforcing the end-effector of the KUKA KR210 R2700 prime either through a PtP motion programmed in the joint space or a linear path programmed in the operational space (thus imposing the motion of all six IR joints). All motions have been tested at varying velocity and acceleration limits, the path starts from point1 (X 1860.0, Y 1620.0, Z 1520.0, A 0.0, B 90.0, C 0.0, S 22, T 27) to point2 (X −745.0, Y 1300.0, Z 215.0, A 90.0, B 90.0, C 0.0, S 22, T 27; coordinates are given using KUKA standard), and it has been chosen together with a robot system integrator. As for the payload, three particular load case scenarios have been considered, namely no external load (hereafter referred to as *Load Type 1*, Fig. 15a), IR carrying a metal bar with uniform cross section and mass equalling 66 Kg (hereafter referred to as *Load Type 2*, Fig. 15b), IR carrying a metal bar with non-uniform cross section and mass equalling 133 Kg (hereafter referred to as *Load Type 3*, Fig. 15c). In all cases, the closing time of the IR electromechanical brakes has been set to 1 s. In addition, several repetitions have been performed in order to allow the IR to reach a stable operational condition (i.e. a stable working temperature). In such case, taking into account the inevitable measurement errors and environmental variations, the discrepancy between the EC model prediction and the measured value being always trustworthy (as previously proven by the authors in [46]). For what concerns the model parameters, the IR inertial properties and reducers parameters are confidential data from the IR manufacturer (also retrievable via well-known identification techniques [24]), whereas PMSM and drive system parameters have been retrieved resorting to the method described in [43]. Similarly to Fig. 11, experimental measurements are collected in colour maps providing the IR energy consumption and motion time (isolines) as function of the imposed velocity and acceleration limits. Numerical values for either Joint PtP or Linear motions and Load Types 1, 2, and 3 are shown in Fig. 16. In addition, the condition of maximum and minimum energy consumption for these three are summarized in Tables 1 and 2. In particular, note that both tables reports the actual parameters to be selected by the user on the real IR controller: Table 1, which depicts the case of PtP joint motions, reports the values of Velocity and Acceleration Limits as percentage, 100% naturally representing the maximum imposable value; Table 2, which depicts the case of linear end-effector motion, reports the maximum values of velocity and acceleration (expressed in m/s and m/s^2) reached by the end-effector during its motion. For what concerns, Joint PtP motions, Table 1 highlights the possibility to reduce the energy consumption from values of 9.3 kJ, 13.0 kJ, and 12.5 kJ (corresponding to the IR operating at maximum joint speed and acceleration) to values of 6.4 kJ, 9.3 kJ, and 8.5 kJ (corresponding to the IR operating at speed and acceleration obtained via the proposed optimization procedure), the EC reduction being 31.2%, 28.4%, and 32% respectively. For what concerns linear end-effector motions, Table 2 highlights the possibility

to reduce the energy consumption from values of 10.1 kJ, 11.0 kJ, and 12.2 kJ (corresponding to the IR operating at its maximum speed and acceleration) to values of 8.1 kJ, 9.0 kJ, and 10.2 kJ (corresponding to the IR operating at speed and acceleration obtained via the proposed optimization procedure), the EC reduction being 19.8%, 18.2%, and 16.4% respectively. Such results confirm the validity and effectiveness of the proposed approach, while, for a correct comparative evaluation with other solutions, it must be noted that other works (e.g. [10]) do not take into account the relevant contribution of the energy drawn by the IR cabinet (which would heavily reduce the final EC reduction values reported in such literature) and also obtain EC reduction by enforcing time scaling in those standstills which are claimed to be artefacts of the internal IR path-planner (which are either required by process constraints and therefore cannot be eliminated, or they may be eliminated by simply interlacing the motions via standard programming techniques). In fact, the present work deals with single robot motion optimization, besides, the method could be subsequently integrated with energy-optimal operation scheduling tools. Finally, it is interesting to note that these energy-optimal values are not trivial. In particular, the experimental activity highlights that, generically speaking, reducing velocity and acceleration as much as possible (within the limits enforced by the production constraints) does not always lead to a reduction in energy consumption, thus further confirming the practical usefulness of the Energy Signature concept.

Table 1
Conditions of maximum and minimum energy consumption for Joint PtP motions.

	Maximum energy consumption condition			Minimum energy consumption condition		
Load type	*Velocity limit*	*Acceleration limit*	*Energy consumption*	*Velocity limit*	*Acceleration limit*	*Energy consumption*
1	100%	100%	9300 J	20%	90%	6400 J
2	100%	100%	13000 J	20%	100%	9300 J
3	100%	100%	12500 J	30%	50%	8500 J

Table 2
Conditions of maximum and minimum energy consumption for linear end-effector motions.

	Maximum energy consumption condition			Minimum energy consumption condition		
Load type	*Velocity limit*	*Acceleration limit*	*Energy consumption*	*Velocity limit*	*Acceleration limit*	*Energy consumption*
1	1.7 m/s	10 m/s^2	10100 J	0.68 m/s	4 m/s^2	8100 J
2	1.7 m/s	10 m/s^2	11000 J	0.68 m/s	10 m/s^2	9000 J
3	1.7 m/s	10 m/s^2	12200 J	0.68 m/s	6 m/s^2	10200 J

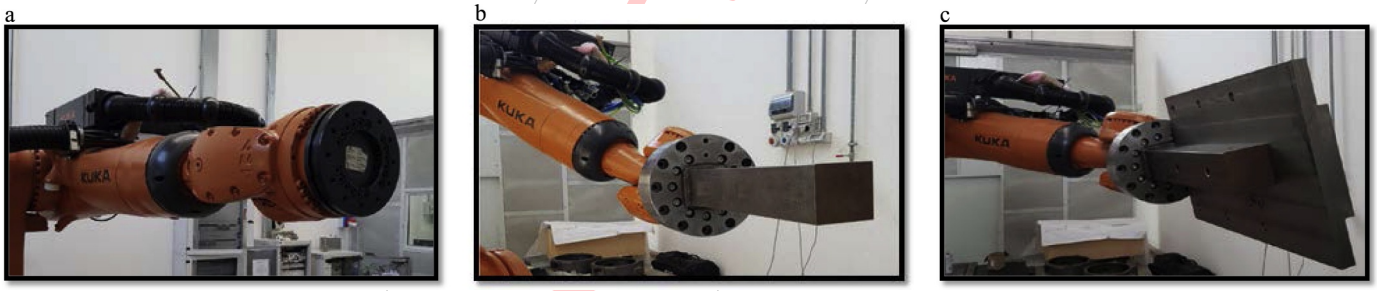

Fig. 15. (a) Load Type 1: no additional load applied on the IR end effector. (b) Load Type 2: beam with uniform cross section mounted on the IR end effector. (c) Load Type 3: beam with uniform cross section mounted on the IR end effector.

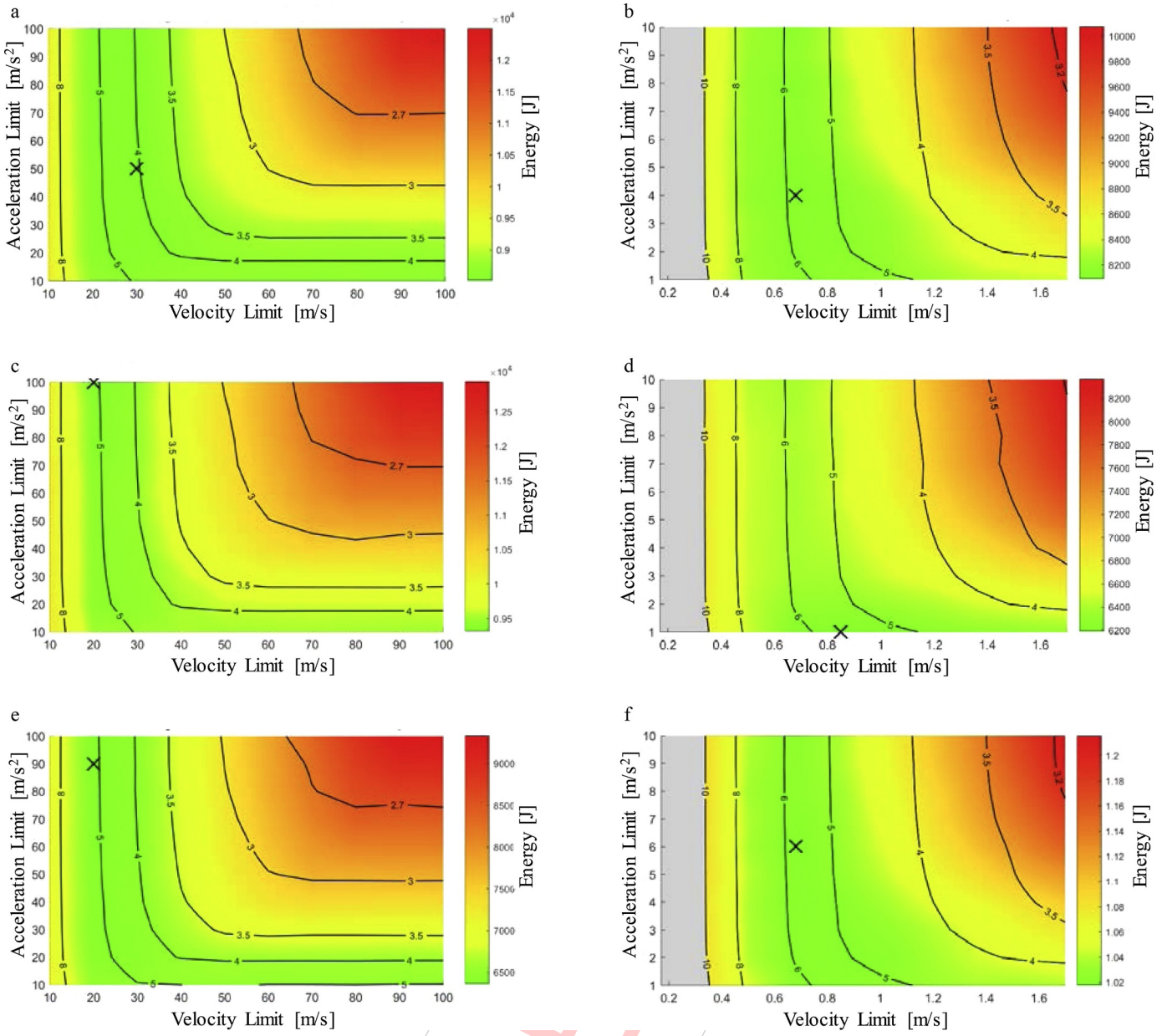

Fig. 16. Experimental energy consumption and motion time as function of velocity and acceleration limits. Graphs are related to either Joint PtP motion or linear motion of the end effector for different payload (Load Types 1, 2, and 3).

7. Conclusions

In this paper a novel energy consumption optimization method, based on the intensive use of a complete and accurate industrial robot model, has been proposed. In particular, starting from accurate trajectories exported from simulations in *Delmia Robotics* environment, a robot motion has been analysed, obtaining the relation between the adjustable motion parameters and the energy consumption. Simulation results show that an accurate choice of the motion velocity and acceleration limits can lead to meaningful energy consumption reduction. Owing to this observation, a practical tool has been developed for assisting the designer in the robot energy efficient programming, by retrieving the optimal motion parameter combination also considering available time constraints. In addition, a structured experimental activity, developed on a single robot subjected to different loading conditions, has also been performed. In particular, the robot energy consumption has been retrieved by measuring the electrical power flow through the grid (i.e. by considering every actual source of power loss and/or storage). Experimental results confirm the validity of the proposed approach, highlighting that, due to the non-trivial nature of the modelled systems, an unbounded reduction of robot speed and acceleration does not consequently mean a reduction of energy expenditure.

Acknowledgements

The research leading to these results has received funding from the European Community's Seventh Framework Programme under grant agreement No. 609391 (AREUS). The Authors would also personally thanks the SIR SpA [47] R&D Department and Mr Federico Grassia for their work and support.

References

[1] A.D. Jayal, F. Badurdeen, O.W. Dillon Jr., I.S. Jawahir, Sustainable manufacturing: modeling and optimization challenges at the product, process and system levels, CIRP J. Manuf. Sci. Technol. 2 (3) (2010) 144–152.

[2] F. Jovane, et al., The incoming global technological and industrial revolution towards competitive sustainable manufacturing, CIRP Ann 57 (2) (2008) 641–659.
[3] W.F. Gaughran, S. Burke, P. Phelan, Intelligent manufacturing and environmental sustainability, Rob. Comput. Integr. Manuf. 23 (6) (2007) 704–711.
[4] G. Berselli, F. Balugani, M. Pellicciari, M. Gadaleta, Energy-optimal motions for Servo-Systems: a comparison of spline interpolants and performance indexes using a CAD-based approach, Rob. Comput. Integr. Manuf. 40 (2016) 55–65.
[5] G.M Masters, Renewable and Efficient Electric Power Systems, Wiley and IEEE press, 2013.
[6] N. Sundstrom, O. Wigstrom, B. Lennartson, Conflict between energy, stability, and robustness in production schedules, IEEE Trans. Autom. Sci. Eng. 14 (2) (2017) 658–668.
[7] M. Pellicciari, A. Avotins, K. Bengtsson, B. Lennartson, G. Berselli, N. Bey, D. Meike, AREUS - Innovative hardware and software for sustainable industrial robotics, IEEE CASE Int. Conf. Autom. Sci. Eng. (2015) 1325–1332.
[8] O. Wigstrom, B. Lennartson, A. Vergnano, C. Breitholtz, High-level scheduling of energy optimal trajectories, IEEE Trans. Autom. Sci. Eng. 10 (1) (2013) 57–64.
[9] E. Glorieux, S. Riazi, B. Lennartson, Productivity/energy optimisation of trajectories and coordination for cyclic multi-robot systems, Rob. Comput. Integr. Manuf. 49 (2018) 152–161.
[10] S. Riazi, K. Bengtsson, R. Bischoff, A. Aurnhammer, O. Wigström, B. Lennartson, Energy and peak-power optimization of existing time-optimal robot trajectories, IEEE CASE Int. Conf. Autom Sci. Eng. (2016) 321–327.
[11] A. Mohammed, B. Schmidt, L. Wang, L. Gao, Minimizing energy consumption for robot arm movement, Procedia CIRP 25 (2014) 400–405.
[12] C. Bryan, M. Grenwalt, A. Stienecker, Energy consumption reduction in industrial robots, Proceedings of ASEE North Conference, 2010, pp. 1–4.
[13] S. Björkenstam, D. Gleeson, R. Bohlin, J.S. Carlson, B. Lennartson, Energy efficient and collision free motion of industrial robots using optimal control, IEEE CASE Int. Conf. Autom. Sci. Eng. (2013) 510–515.
[14] ⟨https://www.3ds.com/products-services/delmia/products/v6/portfolio/d/digital-manufacturing-and-production/s/robotics-programmers/p/robotics-offline-programming/⟩ (accessed 30.05.17).
[15] ⟨http://new.abb.com/products/robotics/robotstudio/downloads⟩ (accessed 30.05.17).
[16] ⟨https://www.kuka.com/en-gb/products/robotics-systems/software/simulation-planning-optimization/kuka_sim⟩ (accessed 30.05.17).
[17] ⟨http://www.realistic-robot-simulation.org/⟩ (accessed 30.05.17).
[18] M. Pellicciari, G. Berselli, F. Leali, A. Vergnano, A minimal touch approach for optimizing energy efficiency in pick-and-place manipulators, IEEE ICAR Int. Conf. Adv. Rob., (2011) 100–105.
[19] D. Meike, Increasing energy efficiency of robotized production systems in automobile manufacturing, Ph.D. Thesis Riga Technical University, 2013.
[20] J. Gregory, A. Olivares, E. Staffetti, Energy-optimal trajectory planning for robot manipulators with holonomic constraints, Syst. Control Lett. 61 (2) (2012) 279–291.
[21] A.A. Ata, Optimal trajectory planning of manipulators: a review, J. Eng. Sci. Technol. 2 (1) (2007) 32–54.
[22] S.E. Sergaki, G.S. Stavrakakis, A.D. Pouliezos, Optimal robot speed trajectory by minimization of the actuator motor electromechanical losses, J. Intell. Rob. Syst. 33 (2) (2002) 187–207.
[23] Carabin, G., Wehrle, E., Vidoni, R. A Review on Energy-saving optimization methods for robotic and automatic systems, 6(4), 39, pp. 1–21, 2017.
[24] L Ljung, System Identification: Theory for the User, Second edition, Prentice Hall, Englewood Cliffs, New Jersey, 1999.
[25] D. Meike, M. Pellicciari, G. Berselli, A. Vergnano, L. Ribickis, Increasing the energy efficiency of multi-robot production lines in the automotive industry, IEEE Int. Conf. Autom. Sci. Eng. (2012) 700–705 art. no. 6386391.
[26] A. Fenucci, M. Indri, F. Romanelli, An off-line robot motion planning approach for the reduction of the energy consumption, IEEE International Conference on Emerging Technologies and Factory Automation, Germany, 2016, pp. 1–8.
[27] T. He, Y. Zhang, F. Sun, X. Shi, Immune optimization based multi-objective six-DOF trajectory planning for industrial robot manipulators, World Congress on Intelligent Control and Automation, China, 2016, pp. 2945–2950.
[28] A. Mohammed, B. Schmidt, L. Wang, L. Gao, Minimizing energy consumption for robot arm movement, Procedia CIRP 25 (2014) 400–405.
[29] H. Ding, Control of robotic systems for safe interaction with human operators, Proceedings-of IJCAI International Joint Conference on Artificial Intelligence, 22, str, 2011, p. 2792.
[30] H. Liu, X. Lai, W. Wu, Time-optimal and jerk-continuous trajectory planning for robot manipulators with kinematic constraints, Rob. Comput. Integr. Manuf. 29 (2) (2013) 309–317.
[31] A. Gasparetto, A. Lanzutti, R. Vidoni, V. Zanotto, Experimental validation and comparative analysis of optimal time-jerk algorithms for trajectory planning, Rob. Comput. Integr. Manuf. 28 (2) (2012) 164–181.
[32] A. Gasparetto, V. Zanotto, A technique for time-jerk optimal planning of robot trajectories, Rob. Comput. Integr. Manuf. 24 (3) (2008) 415–426.
[33] J. Fang, J. Zhao, T. Mei, J. Chen, Online optimization scheme with dual-mode controller for redundancy-resolution with torque constraints, Rob. Comput. Integr. Manuf. 40 (2016) 44–54.
[34] G. Michalos, et al. Design considerations for safe human-robot collaborative workplaces, Procedia CIRP 37 (2015) 248–253.
[35] M. Safeea, R. Bearee, P. Neto, End-effector precise hand-guiding for collaborative robots, ROBOT 2017, Adv. Intell. Syst. Comput. 694 (2018) 595–605.
[36] F. Leali, A. Vergnano, F. Pini, M. Pellicciari, G. Berselli, A workcell calibration method for enhancing accuracy in robot machining of aerospace parts, Int. J. Adv. Manuf. Technol. 85 (1-4) (2016) 47–55.
[37] KUKA. Hand guiding using KUKA Light Weight Robot – demonstration in Hannover fair 2011.
[38] B. Siciliano, O. Khatib (Eds.), Springer Handbook of Robotics, Springer-Verlag, Berlin Heidelberg, 2008.
[39] B.K. Bose, Modern Power Electronics and AC Drives, Prentice Hall, Upper Saddle River, NewJersey, 2002.
[40] M. Pellicciari, G. Berselli, F. Balugani, On designing optimal trajectories for servo-actuated mechanisms: detailed virtual prototyping, IEEE/ASME Trans. Mechatron. 20 (5) (2014) 2039–2052.
[41] L. Malesani, L. Rossetto, P. Tenti, P. Tomasin, AC/DC/AC PWM converter with reduced energy storage in the DC link, IEEE Trans. Ind. Appl. 31 (2) (1995) 287–292.
[42] P. Pillay, R Krishnan, Modeling, simulation, and analysis of permanent-magnet motor drives, Part I: the permanent-magnet synchronous motor drive, IEEE Trans. Ind. Appl. 25 (2) (1989) 265–273.
[43] E. Oliva, G. Berselli, M. Pellicciari, A.O. Andrisano, An engineering method for the power flow assessment in servo-actuated automated machinery: mechatronic modeling and experimental evaluation, Rob. Comput. Integr. Manuf. 38 (2016) 31–41.
[44] A. Vergnano, C. Thorstensson, B. Lennartson, P. Falkman, M. Pellicciari, C. Yuan, S. Biller, F Leali, Embedding detailed robot energy optimization into high-level scheduling, IEEE International Conference on Automation Science and Engineering, 2010, pp. 386–392.
[45] E. Vidarsson, Experimental evaluation of energy optimized trajectories for industrial robots, MSC Thesis Chalmers University of Technology (available online), 2015.
[46] D. Meike, M. Pellicciari, G. Berselli, Energy efficient use of multirobot production lines in the automotive industry: detailed system modeling and optimization, IEEE Trans. Autom. Sci. Eng. 11 (3) (2014) 798–809.
[47] ⟨http://www.sir-mo.it⟩ (accessed 30.05.17).