Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 15 (2014) 105 – 110

www.elsevier.com/locate/procedia

21st CIRP Conference on Life Cycle Engineering

Energy efficient trajectories for an industrial ABB robot

Koen Paes[a], Wim Dewulf[a], Karel Vander Elst[a], Karel Kellens[b], Peter Slaets[a]

[a]*Department of Mechanical Engineering, KU Leuven, campus Group T, Andreas Vesaliusstraat 13, 3000 Leuven, Belgium*
[b]*Department of Mechanical Engineering, KU Leuven, Celestijnenlaan 300A, 3000 Leuven, Belgium*

Abstract

This paper presents a systematic methodology for on-site identification and energy-optimal path planning of an industrial robot. The identification experiments are carried out on-site, in a quick, non-invasive way using a CA8335 Qualistar three-phase electrical networks analyser. Next, the collected data is compared with a parametrised dynamic robot model in an optimisation routine. This routine results in the specification of a parametric dynamic robot model. The model is used as a dynamic constraint for a model predictive control problem, where other physical constraints are added i.e. the limited workspace and the constraints on the joint velocities and accelerations. A sequential quadratic programming solver is used to minimise a mechanical energy based cost function. The resulting energy-optimal path is translated into custom robot commands executable on an industrial robot. The systematic methodology is validated on an IRB1600 industrial ABB robot performing a custom pick-and-place operation. The obtained dynamic robot model is given and compared to the collected measurements. To demonstrate the possibility of energy saving by 'intelligently' programming a robot trajectory the energy and time-optimal paths are generated taking all physical constraints into account. Simulation results show a significant time and energy improvement (up to 5%) compared to most trajectories generated by the ABB software. The most remarkable result is that the fastest energy-optimal trajectory turns out to be 4% more energy efficient and 3% faster then the commercially available fastest trajectory. Additional stand-still experiments show that activation of the brakes is favoured over an actuated stand-still from an energy point of view, assuming that the start-up time when releasing the brakes is limited.

Keywords: industrial robot, energy efficiency, path planning, quadratic programming

1. Introduction

The past decade has shown a continuous increase in the price of electricity in the industrialised world and all signs point toward an even steeper increase in the future. Reduction of our energy consumption, both on a household and on an industrial scale, therefore seems primordial from an economical as well as an ecological point of view. In fact, the European commission has set the goal of reducing the energy consumption of primary energy users with 20% by the year 2020, as compared to 2007. This research focuses on the possibility of energy saving in industrial robotics. Several methods have been developed over the years, showing great potential to reduce the energy consumption of an industrial robot. These methods include the use of an intelligent braking management system or the temporal storage of energy in a capacitive buffer. An excellent survey of these methods is provided by Meike and Ribickis [6].
The drawback of these methods however lies in the fact that they all require severe changes in robot hardware and come with a high initial investment cost, which is why many companies are still reluctant to implement them. This research focuses on an alternative approach to reduce energy consumption, namely by programming robot trajectories in a more intelligent way. Currently, in the majority of the cases, industrial robots move along trajectories which are far from energy-optimal, especially when the robot has a certain amount of idle time during its cycle. Commercial robot-programming software, such as ABB's RobotStudio, only allows programming linear and joint movements and does not have the possibility of energy-optimisation, thereby discouraging the robot programmer from taking energy consumption into consideration.
This paper describes a methodology to obtain energy-optimal trajectories for an ABB IRB1600-1.45m robot and to automatically generate the RAPID-code needed to program the robot. For this purpose, first of all a dynamic model needs to be available enabling accurate torque simulation during robot movement. The measurements necessary to determine this dynamic model are obtained by carrying out identification experiments on an IRB1600 robot. Data is gathered by moving the robot along a periodic excitation trajectory, programmed in RAPID-code, and measuring all motor currents using current clamps inside the robot controller. Usually identification experiments

Selection and peer-review under responsibility of the International Scientific Committee of the 21st CIRP Conference on Life Cycle Engineering in the person of the Conference Chair Prof. Terje K. Lien
doi:10.1016/j.procir.2014.06.043

are carried out in an academic environment [9], where a robot is fitted with more accurate measuring equipment and trajectories can be programmed directly, rather than through commercial software. This on-site method will not achieve the same accuracy, however it has the major advantage of being applicable to any industrial robot, in its own production environment. Set-up and execution of the identification experiment would take no more than twenty minutes after which the robot can continue its task.
After data collection and processing, the dynamic parameters are estimated in Matlab. Using the 'Robotics'-toolbox, developed by Peter Corke [1], and a first estimate of the robot dynamic parameters an initial dynamic model is created. An objective function then minimises the difference between the measured motor torques and the torques predicted by the dynamic model to obtain a reliable dynamic model of the robot.
This dynamic model is used to find the most energy-optimal trajectory for a typical pick-and-place movement. The trajectory is then programmed in RobotStudio, where the RobotStudio simulation tool is used to compare energy consumption of sub-optimal trajectories with the optimal trajectory. Furthermore, the energy consumption of the robot at stand-still is measured and evaluated.

2. Dynamic robot model

To describe the dynamics of the robot, each link is attributed with a specific mass, centre of mass and inertia tensor. The equations of motion which describe the applied joint torques $\boldsymbol{\tau} \in \Re^n$ as a function of the joint angles $\mathbf{q}$[1] $\in \Re^n$ can be written as [7]

$$\boldsymbol{\tau} = \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) + \boldsymbol{\tau}_f(\dot{\mathbf{q}}), \tag{1}$$

where $\mathbf{M}(\mathbf{q}) \in \Re^{n\times n}$ is a positive definite mass matrix, $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) \in \Re^{n\times n}$ is a matrix accounting for Coriolis and centrifugal effects, $\mathbf{G}(\mathbf{q})$ is the torque exerted by gravity and $\boldsymbol{\tau}_f(\dot{\mathbf{q}})$ is the friction torque.
Experiments have shown that friction can consume up to thirty percent of the actuator torque, so an adequate friction model is necessary. Several very accurate non-linear friction models have been proposed [2], but most frequently a simple model, combining Coulomb and viscous friction, is appropriate. In this case the friction term can be written as

$$\boldsymbol{\tau}_f(\dot{\mathbf{q}}) = \mathbf{f}_\mathbf{C}\ \mathrm{sign}(\dot{\mathbf{q}}) + \mathbf{f}_\mathbf{v}\dot{\mathbf{q}}, \tag{2}$$

with $\mathbf{f}_\mathbf{C}$ and $\mathbf{f}_\mathbf{v}$ the Coulomb and viscous friction coefficients respectively.
In this way, each link can be described by 12 dynamic parameters (1 for the mass, 3 for the centre of mass, 6 for the inertia tensor and 2 for the friction model). The dynamic model of a 6-dof robot, such as the IRB1600, thus contains 72 parameters, which need to be determined experimentally.
Because the identification experiments are carried out on-site, with the intention of minimising any disturbance of the original robot set-up, joint torques cannot be measured directly. Instead all motor currents were measured inside the IRC5 controller, using current clamps.
To calculate the resulting joint torques from these current measurements, technical specifications, such as motor torque constants, torque characteristics, transmission ratios and efficiencies are needed. ABB's company policy however restricts the public distribution of technical data. Only the motor torque constants of the 6 permanent-magnet synchronous motors (PMSM) were provided, all information regarding the transmissions is considered classified.
Therefore, any energy loss occurring in the transmission is neglected in our model. Fortunately the transmission ratios can easily be determined experimentally, making use of the resolvers, which each PMSM is equipped with. Table 1 shows motor torque constants and transmission ratios for all joints.

Table 1. Motor and transmission data for the IRB1600-1.45m

Joint	Motor torque constant (Nm/Arms)	Transmission ratio
1	0.64	130
2	0.64	130
3	0.70	104
4	0.47	60.1
5	0.47	66.8
6	0.47	64.1

3. Identification experiments

After designing and programming an appropriate excitation trajectory for an IRB1600-1.45m robot, joint movements are measured by the motor resolvers, while motor currents are measured by current clamps connected to CA8335 Qualistar+ analysers.
These analysers are equipped with a 'inrush' mode, have a fixed sampling rate of approximately 3200 Hz, leading to a full memory (since the analysers cannot directly stream data to a pc) after approximately 80 seconds, thereby imposing a strict upper limit on the duration of the identification experiment.
Special care has to be taken to synchronise the position and current measurements. Synchronisation is accomplished by setting and resetting a digital output signal at specific joint positions and measuring (along with the motor currents) the voltage of this output signal inside the controller with the CA8335 analyzers.

3.1. Design of the excitation trajectory

The choice of the excitation trajectory is a critical issue for the identification of the dynamic model parameters. The chosen trajectory should sufficiently excite the system, otherwise

[1] Written in bold, $\mathbf{q}$ represents a vector containing all six joint angles; $\mathbf{q} = [\theta_i]$

some parameters may become unidentifiable or very sensitive to noise on the measured data. On the other hand, aspects such as joint flexibilities or nonlinearities which are not included in the dynamic model[1], should be excited as little as possible by the chosen trajectory.
For our experiments we have chosen a periodic excitation trajectory, which takes the form of a finite Fourier series [8, 10]. The angular position q_i for each joint i is written as a finite Fourier series, all sharing the same fundamental frequency ω_f, so the motion becomes periodic.

$$q_i(t) = q_{i0} + \sum_{k=1}^{N} a_k^i \sin(k\omega_f t + \varphi_k^i) \tag{3}$$

Using periodic excitation offers several advantages

- The signal-to-noise ratio can be improved by data averaging over the number of measured periods. The averaged trajectory $\bar{q}$ and torque $\bar{\tau}$ are simply obtained from

$$\bar{q} = \frac{1}{M}\sum_{m=1}^{M} q_m, \; \bar{\tau} = \frac{1}{M}\sum_{m=1}^{M} \tau_m \tag{4}$$

  where M is the number of measured periods. This noise reduction technique is preferable over filtering, since filtering colours the noise and consequently complicates an efficient parameter estimation.

- The standard deviation of the noise on the measured signals can be calculated without performing additional measurement:

$$\sigma_q(k) = \sqrt{\frac{1}{M-1}\sum_{m=1}^{M}(q_m(k) - \bar{q}(k))^2} \tag{5}$$

  where M is the number of periods and $q_m(k)$ is the k-th sample of the m-th period. Similar reasoning applies for σ_τ^2.

- The calculation of the joint velocities and accelerations can be performed by analytical differentiation of equation (3). The measured resolver readings are first approximated, using a least squares method, by a Fourier series, which can then be differentiated once and twice to obtain joint velocities and accelerations respectively.

- All joints are excited simultaneously in one experiment, reducing the experiment time needed at the robot.

As a fundamental frequency a value of 0,1 Hz is chosen. This leads to a periodic motion with a period of 10 seconds, allowing measurement and averaging over 7 periods, due to the time limit on the CA8335 analysers. Two higher harmonic terms are included at 0,4 Hz and 0,7 Hz, still low enough so as not to excite flexibilities which are usually located around 20-30 Hz.
The choice of the different Fourier coefficients, a_k^i and φ_k^i, must lead to a feasible trajectory. The actual robot kinematics and dynamics impose a number of constraints on the trajectory; maximum and minimum joint angles, velocities and accelerations must be taken into account for each joint. Furthermore there are constraints imposed by the actual environment of the robot to avoid collision; in the case discussed herein, a metal gate was located 80 cm behind the robot. Maximum motor currents also lead to constraints on the applied joint torques.
To select the proper Fourier coefficients, a kinematic model of the robot was programmed in Matlab, checking limits on joint angles, velocities and accelerations on one hand and using forward kinematics to visualise the resulting trajectory on the other hand. Some trial-and-error leads to a result, which satisfies all constraints. The resulting excitation trajectory is implemented in RAPID-code by defining a large number of *jointtargets*, each lying on the excitation trajectory, separated by a time interval of 0,01 second. Figure 1 shows a RobotStudio-screenshot of the programmed excitation trajectory.

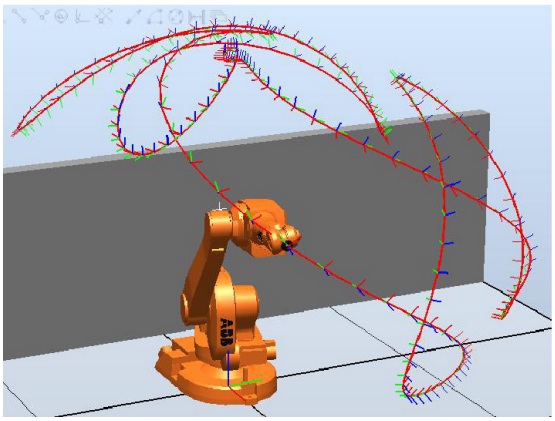

Figure 1. Excitation trajectory programmed in RobotStudio.

3.2. Measurement results

A RAPID program collects all joint positions along with the digital output values to synchronise the joint measurements with the current measurements. For each motor, all three phases of the stator current are measured at a sampling rate of 3200 Hz with the CA8335 analysers, along with the voltage of the digital output signal.
To determine the resulting torque from these current measurements, the three phases of the stator current are usually transformed to two current components, (i_q, i_d), in a reference frame which rotates along with the rotor. i_q represents the torque-producing component of the stator current, while i_d represents the flux producing component [4].
These components depend on the torque-angle δ, which is very hard to determine accurately. However, the PMSM drive system in the IRC5-controller employs a constant torque-angle control

[1] While these elements would ideally be present in the model, additional sensors on the joints would be required for the identification procedure. Coupled with the fact that most industrial manipulators are quite stiff, these effects are ignored.

strategy, meaning the torque-angle δ is kept at a constant 90° (or −90°) to achieve maximum torque. In this case, the PMSM behaves much like a DC-motor and the torque-producing component i_q can simply be calculated by adding the three stator currents as space-vectors and multiplying by 2/3.
Care must be taken at high angular velocities however, because in this region flux weakening is used to reach these high velocities and the assumption of $\delta = 90°$ is no longer valid. For this reason, the excitation trajectory is designed in such a way that joint velocities do not exceed 80% of the maximum joint velocity limit, so the assumption of constant torque-angle holds.

The only ambiguity still remaining is the sign of the torque-angle δ and thus the sign of the resulting torque. One way to get around this problem would be to measure stator voltages along with the currents and then use the sign of the power and the sign of the joint velocities to determine the sign of the torque. Unfortunately the voltage inputs of the analysers were already used to measure the synchronisation signal. Instead, initial estimates were made for the mass and friction coefficients of each link and using the joint position and velocity data, a rough view of the resulting torque could be determined. Figure 2 shows this method for joint 3; the thin blue line representing the measured torque-producing component i_q, and the bold red line representing the rough estimate of the torque (scaled down to the current values). Agreement is far from perfect, but suffices to determine the sign of i_q.
After correction of the sign, the i_q-values were multiplied by

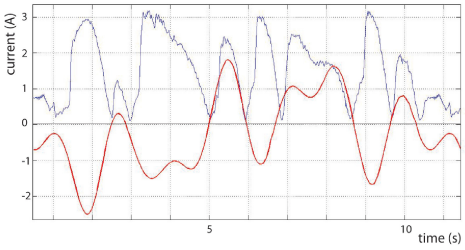

Figure 2. Determining the sign of i_q, by comparing the measured torque-producing current i_q (blue) with the scaled-down initial torque estimate (red).

the corresponding motor torque constants and transmission ratios to produce the resulting joint torque values. Finally torque data was re-sampled to 100 Hz, to match the joint data and all data was averaged over the 7 measurement periods.

4. Determination of dynamic parameters

Using the data obtained in the identification experiment, a numerical routine was created in Matlab to determine all dynamic parameters of the robot. We gratefully made use of the 'Robotics'-toolbox [1], which allows robot objects to be created within Matlab and which provides many functions that are useful for the study and simulation of any type of serial-link manipulator.

Next, an objective function F is defined

$$F = \sqrt{\frac{1}{6N}\sum_{n=1}^{N}\sum_{i=1}^{6}(\tau(i,n) - \tau_{est}(i,n))^2} \qquad (6)$$

which 'measures' the rms-difference between the measured torques and the torque estimates from the dynamic model. In this equation $\tau(i,n)$ represents the n-th torque data sample of joint i.
The Matlab-function *fminunc* is applied to this objective function, performing an unconstrained non-linear optimisation thereby determining the parameters of the robot object for which this function reaches a minimum.

It should be noted however that the objective function is far from being convex and has a multitude of local minima. A reliable initial estimate of the dynamic parameters is therefore crucial for the success of this optimisation routine. Masses, centers of mass and moments of inertia were estimated initially using the physical knowledge available about the robot. Starting from these estimates, all dynamic parameters are further refined by the routine until a minimum for the objective function is reached (table 3). Note that all matrix cross-terms are assumed to be zero.

Table 2. Dynamic parameters for the first three joint of the IRB1600-1.45m

Joint	1	2	3
$m\ [kg]$	-	31	47
$c_x\ [m]$	-	-0,38	-0,02
$c_y\ [m]$	-	0,01	-0,01
$c_z\ [m]$	-	0,23	0,07
$I_{xx}\ [kgm^2]$	-	0,12	0,065
$I_{yy}\ [kgm^2]$	-	0,53	0,091
$I_{zz}\ [kgm^2]$	0,35	0,54	0,012
$f_v\ [Nms]$	48	75	32
$f_C\ [Nm]$	25	12	7

The non-convex nature of the objective function remains a critical point in this routine and it should be noted that methods have been developed in which a non-linear transformation of parameters leads to a convex optimisation problem, resulting in a much more accurate estimate of the dynamic parameters. More details can be found in the work of Verscheure et al. [11].

The dynamic model is validated by comparing it with the ABB dynamic model of the IRB1600 available in the RobotStudio-software. RobotStudio allows for simulation of the total motor power and total energy consumption of the robot during any trajectory. Unfortunately, the individual motor powers cannot be determined in RobotStudio.
The 'validation' trajectory, differing from the original excitation trajectory, was programmed and joint torques τ_{est} were com-

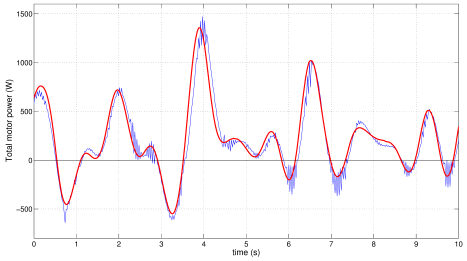

Figure 3. Total power comparison of RobotStudio model (thin, blue) and the identified model (bold, red) for a validation trajectory.

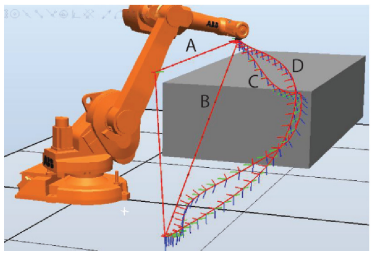

Figure 4. Right-angle trajectory (A), shortest-distance trajectory (B), energy-optimal trajectory (C) and time-optimal trajectory (D) for a pick-and-place manoeuvre.

puted using our dynamic model. The total power at any instant t can then be calculated as

$$P_{tot}(t) = \sum_{i=1}^{6} \tau_{est}(i,t)\dot{Q}(i,t) \tag{7}$$

where $\tau_{est}(i,t)$ and $\dot{Q}(i,t)$ represent torque and angular velocity of joint i, at time t.
Figure 3 depicts the total power from the simulation in thin and the total power resulting from our dynamic model in bold, showing excellent agreement between both.

5. Trajectory optimisation

Since the dynamic model can be used to determine total motor power and total energy consumption for each trajectory, it is now possible to develop a constraint-based optimisation-algorithm, which can determine either a time-optimal or an energy-optimal trajectory. Optimisation can be applied both for the case of a fixed trajectory where only the velocity along the trajectory can be modified as for the case where only starting and endpoint are fixed and the shape of the trajectory itself can vary (such as a pick-and-place manoeuvre).

For the optimisation procedure the ACADO-toolkit – an algorithm collection for automatic control and dynamic optimisation with a user-friendly Matlab interface – is used [3]. This toolkit offers functionality to solve optimal control problems, using an SQP type algorithm, combined with a standard Runge-Kutta integrator for the state integration.

As a small case-study, a typical pick-and-place manoeuvre is both time-optimised and energy-optimised and these optimal trajectories are subsequently compared to several straight-forward trajectories that are indicative for robot operations in industry.
In case of the time-optimal control problem, twelve differential states, $x_1 ... x_{12}$, were defined in ACADO (6 joint positions and 6 joint velocities) along with the 6 joint torques as control inputs, $u_1 ... u_6$. Constraints on all joint positions, joint velocities (both based on the technical specifications of the robot) and joint torques (directly related to motor current limits) were added, as well as constraints fixing the starting and end-position of the pick-and-place manoeuvre. The initial velocity and end-velocity are both chosen to be zero.
In case of the energy-optimal control problem an extra differential state, x_{13}, is defined:

$$x_{13}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}) = \boldsymbol{\tau}\ \dot{\mathbf{q}}, \tag{8}$$

Assuming the energy optimal movement results in a low acceleration $\ddot{\mathbf{q}}$, the contributions of the gravitational and Coriolis effects in equation 1 will be negligible compared to the friction torque. Equation 8 can therefore be reduced to:

$$x_{13}(\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}) \sim \sum_{i=1}^{6} \dot{q}_i^2. \tag{9}$$

which is the selected objective function. Integration of x_{13} over time results in a quantity proportional to the total mechanical energy consumption of the robot.

The resulting time-optimal and energy-optimal trajectories were programmed in RobotStudio, along with two very straight-forward trajectories for comparison; one trajectory making a purely vertical, lifting movement, followed by a horizontal movement, the other being the straight-line, shortest-distance trajectory between starting and endpoint. All four trajectories can be seen in figure 4.

All four trajectories were simulated several times, at varying velocities up to v_{max}, the maximum velocity allowed along the trajectory. For each simulation, energy consumption and trajectory time were stored. The results of these simulations can be seen in figure 5, confirming the time-optimal trajectory to be faster and the energy-optimal trajectory to consume less energy than the test trajectories A and B.
At first glance one might expect the straight-line trajectory, executed at maximum velocity, to be the fastest trajectory. However, due to the complex joint movements needed to create a straight line, the time-optimal trajectory has a 12% shorter trajectory time than trajectory B executed at maximum velocity. Regarding energy consumption, one can see a steep decrease in consumption for trajectories A and B when velocity is reduced, illustrating a significant amount of energy can already be saved by simply limiting robot acceleration and velocity, without changing the trajectory. This method is explored in more

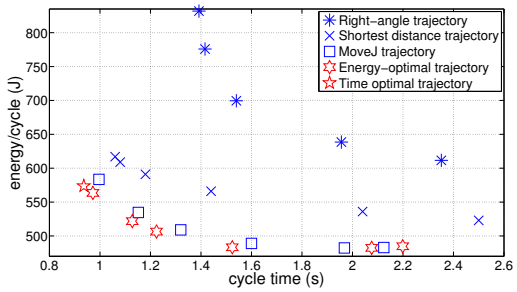

Figure 5. RobotStudio-simulation results for 5 pick-and-place trajectories.

detail in [5]. The energy-optimal trajectory consumes 4% less energy than the fastest joint movement trajectory in Robot Studio – a free point-to-point movement – and additionally the execution time is 3% less. In general a case by case analysis should be made whether the decrease in energy consumption weighs up against the corresponding increased trajectory time.

6. Robot at stand-still

An additional experiment was conducted to determine the energy consumption of the industrial robot at stand-still. Two cases were considered: a fully actuated system and a system with the brakes activated. Figure 6 shows the different robot poses and table 3 the corresponding power consumption. This power consumption (5-35 W) is marginal compared to the power consumption of the moving robot (500-600 W). The energy corresponding to the braking current shown in figure 7 is limited to approximately 12 Joule compared to 5-35 W for an actuated stand-still. Therefore from an energy efficiency point of view an activation of the brakes is preferred when it can be implemented in software and exhibits a limited time delay.

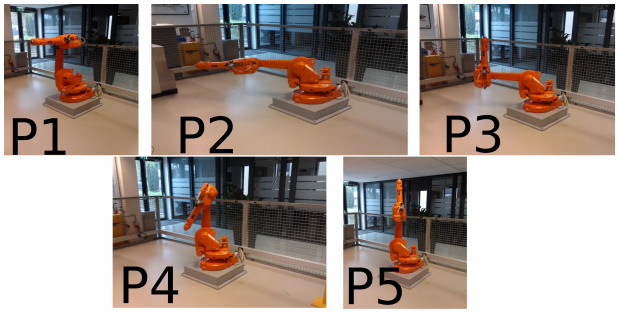

Figure 6. Five different stand-still poses of the industrial robot.

7. Conclusion

This research project proposes a non-invasive identification strategy for industrial robots. The obtained dynamic model is used in an optimisation routine to determine both time-optimal and energy-optimal trajectories and finally simulation of a pick-and-place movement revealed the superiority of these optimised trajectories over the more straight-forward trajectories which are uses in industry. Activation of the brakes is favoured over an actuated stand-still when the execution time is not the bottleneck in the system. Further research is needed to integrate a robot stand-still option in the optimisation procedure.

Table 3. Power consumption of the first three joints of the robot for the five stand-still positions shown in figure 6

Pose	joint 1	joint 2	joint 3	Total power
P1	7 W	2 W	0 W	9 W
P2	5 W	0 W	0 W	5 W
P3	5 W	30 W	0 W	35 W
P4	7 W	2 W	4 W	13 W
P5	5 W	2 W	0 W	7 W

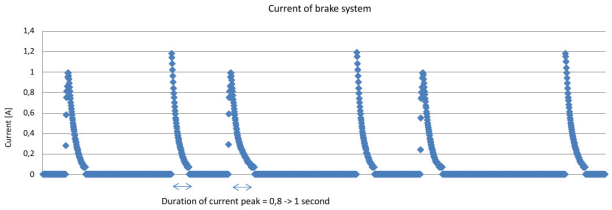

Figure 7. Current flowing through the electric circuit of robot brakes at a voltage of 24V.

8. References

[1] P. I. Corke. *Robotics, Vision & Control: Fundamental Algorithms in Matlab*. Springer, 2011.
[2] B. Daemi, M.and Heimann. Analyzing energy consumption of industrial robots. In *Emerging Technologies Factory Automation (ETFA), 2011 IEEE 16th Conference on*, pages 1–4, 2011.
[3] B. Houska, H. J. Ferreau, and M. Diehl. ACADO toolkit-An open-source framework for automatic control and dynamic optimization. *Optimal Control Applications and Methods*, 9999(9999):n/a+, 2010.
[4] R. Krishnan. *Electric motor drives: modeling, analysis, and control*. Prentice Hall PTR, 2001.
[5] D. Meike and L. Ribickis. Energy efficient use of robotics in the automobile industry. In *Advanced Robotics (ICAR), 2011 15th International Conference on*, pages 507–511, 2011.
[6] D. Meike and L. Ribickis. Recuperated energy savings potential and approaches in industrial robotics. In *CASE*, pages 299–303. IEEE, 2011.
[7] L. Sciavicco and B. Siciliano. *Modelling and Control of Robot Manipulators*. Advanced Textbooks in Control and Signal Processing. Springer-Verlag, 2000.
[8] J. Swevers, C. Ganseman, D. B. Tukel, J. de Schutter, and H. Van Brussel. Optimal robot excitation and identification. *Robotics and Automation, IEEE Transactions on*, 13(5):730–740, 1997.
[9] J. Swevers, W. Verdonck, and J. De Schutter. Dynamic model identification for industrial robots. *Control Systems, IEEE*, 27(5):58–71, 2007.
[10] J. Swevers, W. Verdonck, B. Naumer, S. Pieters, and E. Biber. An experimental robot load identification method for industrial application. *I. J. Robotic Res.*, 21(8):701–712, 2002.
[11] D. Verscheure, B. Demeulenaere, J. Swevers, J. D. Schutter, and M. Diehl. Time-optimal path tracking for robots: A convex optimization approach. *IEEE Trans. Automat. Contr.*, 54(10):2318–2327, 2009.