Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2015, Article ID 931048, 10 pages
http://dx.doi.org/10.1155/2015/931048

Research Article

Assembly Line Productivity Assessment by Comparing Optimization-Simulation Algorithms of Trajectory Planning for Industrial Robots

Francisco Rubio, Carlos Llopis-Albert, Francisco Valero, and Josep Lluís Suñer

Centro de Investigación en Ingeniería Mecánica (CIIM), Universitat Politècnica de València-Camino de Vera s/n, 46022 Valencia, Spain

Correspondence should be addressed to Carlos Llopis-Albert; cllopisa@gmail.com

Received 25 July 2014; Revised 20 September 2014; Accepted 22 September 2014

Academic Editor: Shaofan Li

In this paper an analysis of productivity will be carried out from the resolution of the problem of trajectory planning of industrial robots. The analysis entails economic considerations, thus overcoming some limitations of the existing literature. Two methodologies based on optimization-simulation procedures are compared to calculate the time needed to perform an industrial robot task. The simulation methodology relies on the use of robotics and automation software called GRASP. The optimization methodology developed in this work is based on the kinematics and the dynamics of industrial robots. It allows us to pose a multiobjective optimization problem to assess the trade-offs between the economic variables by means of the Pareto fronts. The comparison is carried out for different examples and from a multidisciplinary point of view, thus, to determine the impact of using each method. Results have shown the opportunity costs of non using the methodology with optimized time trajectories. Furthermore, it allows companies to stay competitive because of the quick adaptation to rapidly changing markets.

1. Introduction

Time needed to perform a trajectory for industrial robots is a very important issue in order to improve productivity in many economic activities. Specifically, most algorithms seek to find the minimum time trajectory in order to increase the working time and subsequently to reduce the unproductive time. The existing literature shows a lack of studies that consider both the economic issues and the motion of industrial robots.

In this paper, the working times of industrial robots are compared between two different approaches while taking into account the corresponding economic impacts. The comparison is applied to several examples, which covers a wide range of parameters that govern the kinematics and dynamics of the industrial robots. The first methodology is based on a robotic simulation program called GRASP (BYG System Ltd) and the second on optimization techniques.

When the working times have been calculated, the assembly line productivity is estimated by means of the time difference, so that we can quantify the impact of each method. Productivity is quantified by conducting an economic study based on the working times of robotic tasks and, more specifically, the time needed to manufacture and assemble a certain product.

A multiobjective optimization problem is posed to assess the trade-offs between the economic variables by means of the Pareto fronts (see Section 6). These fronts will serve to determine those variables that mostly influence the increase of productivity of the assembly line. We will prove that working times (not working cycles) are critical from an economic point of view and so are the methods to obtain them.

Those times will enable us to set conclusions about which method is more useful in order to increase the productivity of the robotic system.

The paper is organized as follows. Initially, we will explain in detail the main characteristics of the trajectory planning methodology and how the time is obtained. Consequently, the economic analysis will provide insight on the productivity

of assembly lines. Finally, the conclusions will be discussed in the last section.

2. Background of the Trajectory Planning Problem

Currently there are a great number of methodologies to solve the trajectory planning problem for industrial robots which give the time needed to perform a task. But few papers tackle the analysis of productivity related to the working times obtained.

Over the years, the algorithms have been polished and the working assumptions of the robotic systems have been increasingly adjusted to real conditions. This fact has been achieved by analysing the complete behaviour of the robotic system, particularly the characteristics of the actuators and the mechanical structure of the robot. To tackle this problem other important working parameters and variables have been taken into account, such as the input torques, the energy consumed, and the power transmitted. Furthermore, the kinematic properties of the robot's links, such as the velocities, accelerations, and jerks, must be also considered. The aforementioned algorithms provide a smooth robot motion for the robotic system.

To obtain the best trajectories in terms of minimum times, some of the working parameters have been included in the appropriate objective function of the optimization procedure (input torques, the energy consumed, and the power transmitted). The optimization criteria most widely used can be sorted as follows.

(1) Minimum time required, which is directly bounded to productivity.

(2) Minimum jerk, which is bounded to the quality of work, accuracy, and equipment maintenance.

(3) Minimum energy consumed or minimum actuator effort, both linked to savings.

(4) Hybrid criteria, for example, minimum time and energy.

In the past, the early algorithms that solved the trajectory planning problem tried to minimize the time needed for performing the task (see [1–3]). One disadvantage of those minimum-time algorithms was that the trajectories had discontinuous values of acceleration and torques which led to dynamic problems during the trajectory performance. Those problems were avoided by imposing smooth trajectories to be followed, such as spline functions which have been used in both path and trajectory planning.

The early algorithms in trajectory planning sought to minimize the time needed for performing the task. The dynamics properties of actuators were neglected. A recent example of this type of algorithm can be found in [4], which determines smooth and near time-optimal path-constrained trajectories. It considers not only velocity and acceleration but also jerk.

Later, the trajectory planning problem was tackled by searching for jerk-optimal trajectories. Jerks are highly important for working with precision and without vibration. They also have an effect on the control system and the wearing of mobile parts such as joints and bars. These methods allow a reduction in errors during trajectory tracking, the stresses in the actuators and also in the mechanical structure of the robot, and the excitement of resonance frequencies. Jerk restriction is introduced by other authors [5, 6].

In [7] a method is introduced for determining smooth and time-optimal path-constrained trajectories for robotic manipulators by imposing limits on the actuator jerks.

In [8] a global minimum-jerk trajectory planning algorithm of a space manipulator is presented.

Another different approach to solving the trajectory planning problem is based on minimizing the torque and the energy consumed instead of the trajectory time or the jerk. This approach leads to smoother trajectories. An early example is seen in [9].

Similarly, in [10], the authors searched for the minimum energy consumed. They proposed a method for solving the trajectory generation problem in redundant degree of freedom manipulators. They used a variational approach and the B-Spline curve was introduced to minimize the electrical energy consumed in a robot manipulator system.

The work in [11] also takes into account energy minimization for the trajectory planning problem.

In [12] the authors proposed a technique of iterative dynamic programming to plan minimum energy consumption trajectories for robotic manipulators. The dynamic programming method was modified to perform a series of dynamic programming passes over a small reconfigurable grid covering only a portion of the solution space at any one pass. Although strictly no longer a global optimization process, this iterative approach retained the ability to avoid certain poor local minima while avoiding the dimensional issue associated with a pure dynamic programming approach. The modified dynamic programming approach was verified experimentally by planning and executing a minimum energy consumed path for a Reis V15 industrial manipulator.

Afterwards, new perspectives appear for solving the trajectory planning problem. The main point was to use a weighted objective function to optimize the working parameters [13]. There, the cost function is a weighted balance of transfer time, the mean average of the torques, and power.

In this paper we will introduce two methods to solve the trajectory planning problem for industrial robots working in complex environments. The time will be used in the economical study.

In the first method, the procedure calculates the optimal trajectory by neglecting initially the potential presence of obstacles in the workspace. By removing the obstacles (real or potential) from the optimization problem, the algorithm will calculate a minimum time trajectory as a starting point. Then the procedure must take into account the real obstacles presented in the workspace. When obstacles are considered, the initial trajectory will not be feasible and will have to evolve so that it can become a solution. The way this initial trajectory evolves until a new feasible collision-free trajectory is obtained is presented in this paper. It is a direct algorithm that works in a discrete space of trajectories, approaching the

first solution to the global solution as the discretization is refined. The solutions obtained are efficient trajectories (i.e., the minimum time trajectories). All the trajectories obtained meet the physical limitations of the robot. The solution also avoids collisions and takes into account the constraint of energy consumed.

The second method calculates the times using the kinematic properties of the robotic system by means of a simulation program called GRASP.

3. Time Obtained Using the Proposed Optimization Trajectory Planner

Our objective is to calculate the minimum time trajectory (t) between the initial and final configurations. Any robot configuration $C^j = C^j(\alpha_i^j, p_k^j)$ can be expressed unequivocally by means of the Cartesian coordinates of significant points of the robot $\alpha_i^j = (\alpha_{xi}^j, \alpha_{yi}^j, \alpha_{zi}^j)$.

We calculate the time needed to go from C^i to C^f. This process is based on an optimization problem to obtain the minimum time between these two configurations. The problem is transformed into obtaining the minimum time over an interpolated trajectory between both configurations, subjected to physical constraints in the actuators.

This optimization problem can be stated as in [7] as follows:

$$\text{Find } q(t), \tau(t), t_f \tag{1}$$

between each of the two configurations (see Section 3.3),

$$\text{Minimizing } \min_{\tau\in\Omega} J = \int_0^{t_f} dt, \tag{2}$$

where $\tau(t) \in R^n$ is the vector of the actuator torques and Ω is the space state in which the vector of the actuator torques is feasible.

The optimization problem is subject to:

(1) the robot dynamics

$$M(q(t))\ddot{q}(t) + C(q(t), \dot{q}(t))\dot{q}(t) + g(q(t)) = \tau(t); \tag{3}$$

(2) unknown boundary conditions (position, velocity, and acceleration) for intermediate configurations a priori

$$\begin{aligned} q(t_{\text{int}-1}) &= q_{\text{int}-1}; \qquad q(t_{\text{int}}) = q_{\text{int}}, \\ \dot{q}(t_{\text{int}-1}) &= \dot{q}_{\text{int}-1}; \qquad \dot{q}(t_{\text{int}}) = \dot{q}_{\text{int}-1}, \\ \ddot{q}(t_{\text{int}-1}) &= \ddot{q}_{\text{int}-1}; \qquad \ddot{q}(t_{\text{int}}) = \ddot{q}_{\text{int}-1}; \end{aligned} \tag{4}$$

(3) boundary conditions for initial and final configurations

$$\begin{aligned} q(0) &= q_o; \qquad q(t_f) = q_f, \\ \dot{q}(0) &= 0; \qquad \dot{q}(t_f) = 0; \end{aligned} \tag{5}$$

(4) collision avoidance within the robot workspace

$$d_{ij} \geq r_j + w_i; \tag{6}$$

d_{ij} being the distance from any obstacle j (sphere, cylinder, or prism) to robot arm i; r_j is the characteristic radius of the obstacle and w_i is the radius of the smallest cylinder that contains the arm i;

(5) actuator torque rate limits

$$\tau_i^{\min} \leq \tau_i(t) \leq \tau_i^{\max} \quad \forall t \in [0, t_{\min}],\ i = 1, \ldots, \text{dof}; \tag{7}$$

(6) maximum power in the actuators

$$P_i^{\min} \leq \tau_i(t)\dot{q}_i(t) \leq P_i^{\max} \quad \forall t \in [0, t_{\min}],\ i = 1, \ldots, \text{dof}; \tag{8}$$

(7) maximum jerk on the actuators

$$\dddot{q}_i^{\min} \leq \dddot{q}_i(t) \leq \dddot{q}_i^{\max} \quad \forall t \in [0, t_{\min}],\ i = 1, \ldots, \text{dof}, \quad (\dddot{q}_i \text{ is the jerk of actuator } i); \tag{9}$$

(8) energy consumed

$$\sum_{j=1}^{m-1}\left(\sum_{i=1}^{\text{dof}}\varepsilon_{ij}\right) \leq E, \tag{10}$$

where ε_{ij} is the energy consumed by the actuator i between the configurations c^j and c^{j+1}.

Here, the main definitions and processes used to obtain the free-collision trajectories of the robot (and subsequently the time needed to perform the trajectory) are detailed.

3.1. Robot Configuration. It is expressed in joint coordinates $c^k(q_i)$ with a view to define kinematics and dynamics of the robot. When dealing with collisions, Cartesian coordinates $c^k(\lambda_j)$ will be used, being $i = 1, \ldots \text{dof}$; $j = 1, \ldots, \text{npc}$; dof: robot degrees of freedom; npc: number of Cartesian points used for the wired model of the robot in collision detection; and k is the configuration itself; see [14–16].

3.2. Adjacent Configuration. Given a feasible configuration of the robot c^k, it is said that c^l is adjacent to it if it is feasible and meets the following two conditions.

(i) The robot end-effector occupies a position corresponding to a node of the discretized workspace in Cartesian coordinates and its distance to the end-effector position in the configuration c^k is less than a given value.

(ii) c^l is such that it minimizes the function

$$\sum_{i=1}^{\text{dof}}\left(q_i^l - q_i^k\right)^2. \tag{11}$$

3.3. Trajectory. Given a sequence of m robot configurations $= \{c^1(q_i^1), c^2(q_i^2), \ldots, c^m(q_i^m)\}$, the trajectory s is defined by means of cubic interpolation functions between adjacent configurations so that the resulting time $t_{\min}$ to perform the trajectory is minimum. We have that

$$\forall t \in \left[t_{j-1}, t_j\right[\longrightarrow q_{ij} = a_{ij} + b_{ij}t + d_{ij}t^2 + e_{ij}t^3, \tag{12}$$

where $i = 1, \ldots, \text{dof}$, $j = 1, \ldots, m-1$.

To ensure continuity, the following conditions associated with the given configurations are considered.

(a) Position: for each interval j the initial and final positions must match c^j and c^{j+1}; this gives a total of $(2\,\text{dof}\,(m-1))$ equations:

$$\begin{aligned} q_{ij}\left(t_{j-1}\right) &= q_i^j, \\ q_{ij}\left(t_j\right) &= q_i^{j+1}. \end{aligned} \tag{13}$$

(b) Velocity: the initial and final velocities of the trajectory must be zero, obtaining (2 dof) equations

$$\begin{aligned} \dot{q}_{i1}\left(t_0\right) &= 0, \\ \dot{q}_{im-1}\left(t_{m-1}\right) &= 0. \end{aligned} \tag{14}$$

When passing through each intermediate configuration, the final velocity of previous interval must be equal to the initial velocity of the next interval; that gives $(\text{dof}\,(m-2))$ equations

$$\dot{q}_{ij}\left(t_j\right) = \dot{q}_{ij+1}\left(t_j\right). \tag{15}$$

(c) Acceleration: for each intermediate configuration, the final actuator acceleration of the previous interval must be equal to the initial acceleration of the next, resulting in $(\text{dof}\,(m-2))$ equations

$$\ddot{q}_{ij}\left(t_j\right) = \ddot{q}_{ij+1}\left(t_j\right). \tag{16}$$

Knowing the time required to perform the trajectory between the different configurations, using the above equations, the coefficients of the cubic polynomials can be obtained efficiently by means of the calculation of the normal time [15].

In addition, the minimum time trajectory s must meet the following four types of constraints:

(d) maximum torque in the actuators,

(e)

$$\tau_i^{\min} \le \tau_i(t) \le \tau_i^{\max} \quad \forall t \in \left[0, t_{\min}\right],\ i = 1, \ldots, \text{dof}, \tag{17}$$

(f) maximum power in the actuators,

(g)

$$P_i^{\min} \le \tau_i(t)\,\dot{q}_i(t) \le P_i^{\max} \quad \forall t \in \left[0, t_{\min}\right],\ i = 1, \ldots, \text{dof}, \tag{18}$$

(h) maximum jerk on the actuators,

(i)

$$\dddot{q}_i^{\min} \le \dddot{q}_i(t) \le \dddot{q}_i^{\max} \quad \forall t \in \left[0, t_{\min}\right],\ i = 1, \ldots, \text{dof}, \tag{19}$$

(j) energy consumed

$$\sum_{j=1}^{m-1}\left(\sum_{i=1}^{\text{dof}}\varepsilon_{ij}\right) \le E, \tag{20}$$

where ε_{ij} is the energy consumed by the actuator i between the configurations c^j and c^{j+1}.

To obtain the minimum time, an optimization problem is solved using variables defined in time increment at each interval (see [17]) so that in the interval between c^j and c^{j+1}, the variable is $\Delta t_j = t_j - t_{j-1}$ and the objective function is

$$\sum_{j=1}^{m-1}\Delta t_j = t_{\min}. \tag{21}$$

3.4. Offspring Trajectory. Let s^j be a minimum time trajectory associated to the sequence of m configurations C^j under the conditions described in Section 3.5. It is said that the trajectory s^k is an offspring of s^j when the following conditions are met:

(a) $C^k = C^j \cup c^n$;

(b) $n \neq 1$;

(c) $n \neq m + 1$.

So the trajectories of a certain generation will have one passing configuration more than the previous generation, but they will keep the same initial and final configurations.

3.5. Obtaining of the Collision-Free Trajectory. The problem of obtaining a feasible and efficient trajectory for a robot in an environment with static obstacles allowing the motion between two given configurations (c^i and c^f) is posed. An efficient trajectory is that performed in a minimum time, with a reasonable computational cost, and subject to the limitations of the robot dynamics, the jerk constraints, and power consumption. Clearly the feasibility of the trajectory means that there are no collisions.

The proposed process for solving the problem involves the following steps which are implemented in the algorithm.

(a) Obtaining the minimum time trajectory: using the procedure described in Section 3.3, the trajectory $s_{\min}$ is obtained corresponding to the sequence of configurations $C = \{c^i, c^f\}$.

(b) Search for collisions: the first configuration from $s_{\min}$ which has collision c^c is determined, and a previous configuration c^a is searched for whose distance is less than d_{seg} (so that the smallest patterned obstacle used to represent the work environment can never be between c^c y c^a).

Table 1: Kinematic characteristics of PUMA 560 robot.

Joint	1	2	3	4	5	6
Minimum angle (°)	−160.0	−215.0	−45.0	−140.0	−100.0	−266.0
Maximum angle (°)	160.0	35.0	225.0	140.0	100.0	266.0
Maximum velocity (°/s)	82.0	54.0	122.0	228.0	241.0	228.0

(c) Obtaining adjacent configurations: up to six new adjacent configurations to c^a can be achieved as defined in Section 3.2 (c_j^a $j = 1, \ldots, 6$).

(d) Obtaining offspring trajectories: for each one of the l adjacent configurations obtained in the previous section that have no collision with obstacles, the offspring trajectory s_k is obtained from $s_{\min}$, such that $C^k = C \cup c_k^a$ $(k = 1, \ldots, l)$.

(e) Trajectory selection: the generated trajectories are introduced in previous section (d) on the set of trajectories ordered by time $T_t = \{s_1 \cdots s_p\}$, taking the minimum time trajectory s_1 and checking for no collisions as it was done in previous section (b). If s_1 has no collision, the algorithm goes to the next section; otherwise it returns to section (c) and the process is repeated.

(f) Refining the trajectory: in case that the collision-free trajectory s_1 does not belong to the first generation (direct offspring of $s_{\min}$, with a sequence of three configurations), we have s_1 such that $C^1 = \{c^i, c^2, c^3, \ldots, c^{m-1}, c^f\}$ (m being the number of configurations that define the trajectory).

$m - 2$ sets of configurations C_p^1 are taken such that $C^1 = C_p^1 \cup c^p$ for $p = 2, \ldots, m-1$, obtaining the corresponding set of collision-free trajectories T_r. If it is empty then it is said that s_1 cannot be reduced; otherwise the process is repeated for the new trajectories and the results are included in T_r. The process finishes when the algorithm cannot obtain new trajectories.

Finally the trajectory s_1 is included in T_r and the reduced trajectory s_r is defined as the trajectory belonging to T_r with minimum time.

The proposed solution to the problem is s_r, which will be a minimum time offspring trajectory $s_{\min}$ and with a small number of passing configurations.

4. Time Obtained Using a Robotic Environment Simulation Program Called GRASP

The simulation program GRASP10 for robotic environments is used to obtain the time needed to perform the motion between two given configurations. Among other tasks, GRASP10 can model and simulate the robot kinematic behavior. In this paper we have used the original model of PUMA 560 robot that comes with the program as a comparator. It is assumed that the point to point trajectory calculation procedures of GRASP10 correspond to the real robot. The robot kinematic characteristics are shown in Table 1.

It should be noted that GRASP10 does not perform dynamic calculations but only kinematic ones. It is therefore very important to indicate the maximum working velocities, which have been obtained from actual robot by considering the properties of each actuator, primarily its maximum power, and the working torque.

When the trajectory is generated by GRASP10, the working actuators act simultaneously and at least one of them is moved to its maximum speed, calculated as follows for each actuator: (see Table 1)

$$\omega_{\max} = \frac{P}{t_{\min}}, \tag{22}$$

where P stands for the power and $t_{\min}$ for the torque in the corresponding actuator.

This methodology (use of GRASP10 for calculating the time required to perform a trajectory) has been applied to the same examples that have been resolved by the optimization algorithm explained in Section 3 in order to compare their efficiency.

For each example, the initial data are the initial and final configurations and the kinematics of the robot. The obstacle has been incorporated after the generation of the path, so that it burdens the previously calculated one.

5. Productivity and Economic Study

In this section the productivity will be quantified by conducting an economic study based on the working times of robotic tasks.

The aim is to increase the profitability of production lines by designing flexible manufacturing systems. This allows companies to stay competitive because of the quick adaptation to rapidly changing markets. For instance, by adjusting the working hours in assembly lines or by deciding which products are more suitable to be manufactured according to the current demand.

This is performed by posing a multiobjective optimization problem, which makes use of the optimization algorithm, above presented, to solve the kinematics and dynamics of robot arms. The optimization method finds the minimum time trajectory to perform industrial tasks in production lines while taking into consideration the physical constraints of the real posed problem and then economic issues are also considered in the process.

Furthermore, Pareto fronts will be introduced, which will serve to determine those variables which mainly affect the improved productivity. To be more precise, the multiobjective optimization problem allows obtaining the Pareto frontiers,

which provides information about the trade-offs between the competing variables (i.e., execution times and benefits for the different products that can be manufactured at the production line).

Therefore, the economic study starts by defining the economic objective function to be used. It is formulated as follows:

$$\text{Max } B = \frac{1}{(1+r)^T}\left[\sum_{p=1}^{n}\left(P_p - C_p\right)\cdot N_p\left(t\right)\right], \quad (23)$$

where B is the objective function to be maximized and represents the current value of the net benefit from a generic product (in €) defined as the revenue of the items manufactured at a production line minus total costs; r is the annual discount rate; T represents number of years; P_p is the market unitary price of the product p (in €); C_p stands for the unitary cost to perform the product p (in €), ranging from costs of raw materials, energy, amortization, labor force, maintenance, and taxes to direct and indirect costs; $N_p(t)$ is a function accounting for the number of products carried out per hour. It is calculated like

$$N_p\left(t\right) = \frac{K}{t(S_k)^{\mu}}, \quad (24)$$

where S_k is the set of tasks needed to manufacture and assemble a certain product (p) and it constitutes the work load, where k represents the number of tasks. $t(S_k) = \sum_{j\in S_k} t_j$ is the cumulated task time and it is called the product time. A cubic function of $t_{\min}$ has been considered. μ is a parameter that refers to the economic environment and the market seasonality. K is a constant related to the current number of working hours per year.

Each one of these tasks is performed by the robot arm, which uses a certain time to describe the optimal trajectory. As above mentioned, the developed algorithm (Section 3) returns the minimum time $t_{\min p}$ to perform the task of the robot arm in order to obtain the product p, while considering the time of the other tasks as constant. The lower the time used by the robot to perform its task, the greater the number of products manufactured per hour. Then, the cumulative time of all tasks can be defined as follows:

$$t\left(S_k\right) = t_{\min p} + \sum_{j\notin S_{\text{robot}}}^{k} t_j. \quad (25)$$

Besides, the amount that an additional item adds to a company's total revenue during a period is called the marginal revenue of the product (MRP).

This factor is defined as the additional products manufactured per hour because of reducing the time used by the robot arm ($t_{\min p}$). The additional products manufactured increase the company's output and, therefore, the company's total revenue.

The marginal revenue product can be obtained by multiplying the marginal product (MP) of the factor by the marginal revenue (MR). In a perfectly competitive market, the marginal revenue a company receives equals the market-determined price of the product P_p.

Therefore, for companies in perfect competition, the marginal revenue product MRP can be expressed as follows:

$$\text{MRP} = \text{MP} \times P_p. \quad (26)$$

The law of diminishing marginal returns tells us that if the quantity of a factor is increased while other inputs are held constant, its marginal product will eventually decline. If marginal product is falling (MP ↓), MRP must be falling as well (MRP ↓).

The marginal revenue of a product (calculated in Section 7) will be used to obtain the total annual benefits in an assembly line, as well as to determine the trade-offs between the benefits and the times obtained in the multiobjective optimization problem.

6. Pareto Optimality

Many real-world problems face two different types of mathematical difficulties. Those difficulties are the existence of multiple and conflicting objectives and a highly complex search space. Contrary to a single optimal solution, competing goals entail a set of compromise solutions generally denoted by the Pareto-optimal, for example, [18]. When there is lack of preference information, none of the corresponding trade-offs between decision variables could be said to be better than that of others. The optimal set of solutions in multiobjective optimization problems is named the Pareto-optimal set.

A solution is defined as Pareto optimal if no improvement in one objective can be accomplished without adversely affecting at least one other objective. In the objective space, the hypersurface that represents all possible Pareto-optimal solutions is termed as the Pareto front or frontier. A design that is located along the Pareto front is neither better nor worse than any other solution along the Pareto front. Hence, the solutions that compose the Pareto-optimal set are equivalently optimal. The objective of multiobjective optimization using this technique is to generate as many Pareto-optimal solutions as possible to adequately represent the Pareto front. This allows obtaining sufficient information for a trade-off decision between competing variables. The Pareto front can be discontinuous, concave, or convex and, in general, is not known a priori.

The concept of domination enables the comparison of a set of designs with multiple objectives. Such a concept is not required for single objective optimization on account of the fact that the value of the objective function is the only measure of the quality of the design.

Then, in a direct comparison of two designs, if one design dominates another, the dominating design is superior and nearer to the Pareto front. Instead, if neither design dominates the other, the designs are nondominant to each other. Therefore, the best designs (with equally good objective vectors) in an arbitrary set of solutions can be distinguished because they are not dominated by any other design in the set; they compose the nondominated subset. Similarly, the designs that compose the Pareto-optimal set are the nondominated set associated with the entire feasible space and are located along the Pareto front.

Table 2: Working times needed to perform the trajectory for each example.

Optimization algorithm	Example	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
	Working time (s)	0.7488	0.5909	0.7511	1.5557	0.7488	0.5115	1.3920	0.7118	0.4529	1.0000
	Example	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
	Working time (s)	0.7488	0.5909	0.7511	1.5557	0.7488	0.5115	1.3920	0.7118	0.4529	1.0000
Simulation software GRASP	Example	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
	Working time (s)	0.7500	0.6200	0.7700	1.6000	0.7600	0.5300	1.4000	0.7200	0.4600	1.2000
	Example	**11**	**12**	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**
	Working time (s)	0.7500	0.6000	0.7600	1.5600	0.7500	0.5200	1.4300	0.7400	0.4600	1.3000

Consequently, the trajectory planning optimization problem in robotics can be defined as finding a motion law along a given geometric path, taking into account some predefined requirements, while generating minimum trajectory time of the robot arm. The inputs of the trajectory planning problem are the geometric path and the kinematic and dynamic constraints, while the output is the trajectory of the joints (or of the end-effector). The trajectory is expressed as a time sequence of position, velocity, and acceleration values. The optimized trajectory should also meet the physical limitations of the robot, the constraint of energy consumption and collisions avoidance.

The times obtained from the simulation-optimization procedures lead to different benefits. Therefore the Pareto fronts can be determined, thus showing the trade-offs between the benefits and the obtained times. They show the opportunity cost of time higher than the minimum (see Section 7).

7. Results of the Application of the Methodology to Different Examples

The proposed multiobjective optimization methodology and the Pareto optimality have been applied to different examples in order to set the above mentioned trade-offs between the benefits and the obtained times. Therefore, now the objective is conducting an economic study to quantify the productivity of the assembly line by comparing the minimum time trajectory to perform certain industrial tasks. It is obtained using two methods: an optimization algorithm and the simulation software GRASP.

These times are obtained while taking into consideration the physical constraints of the real working problem and the economic issues involved in the process. Twenty examples have been solved. The examples differ in the initial and final configurations of the robot, that is, the location of the end-effector (see Table 2). The optimization algorithm simulates the PUMA 560 robot.

This industrial robot arm is probably the most common robot in university laboratories. It is a 6-R (revolution) type robot, with 6 degrees of freedom, and uses direct current (DC) servo motors as its actuators. This robot is a compact computer-controlled robot not only to perform service tasks, but also to carry out medium-to-lightweight assembly, welding, materials handling, packaging, inspection applications, personal care, and so forth. The Series 500 is the most widely used model in the PUMA line of electrically driven robots.

With a 36-inch reach and 5-pound payload capacity, this robot is designed with a high degree of flexibility and reliability. The range of these angles from θ_1 to θ_6 is the following (320°, 250°, 270°, 280°, 200°, and 532°). The corresponding link lengths from L1 to L6 are (432, 432, 433, 56, 56, and 37.5) mm.

Table 2 presents the results obtained for the developed algorithm and GRASP, that is, the working time required for the robot to perform the industrial tasks.

With regard to the economic issues associated with the robot industrial tasks we suppose for all examples the following quantities and considerations:

unitary cost to produce a certain item: 0.8 € (without considering the cost of the energy consumed);

item price: 1 €. For the sake of simplicity, we assume that only one product is produced at this point.

When the cost of the energy consumed is considered, the different examples have different costs because of the different working times. Therefore cost of the consumed energy: 0.0676 €/kWh (it is an average cost). This cost has been added to the cost of 0.8 €.

For reasons of clarity, the manufactured products are obtained in only one shift of 8 hours (365 working days in a year). The benefits B are presented for a period of one year.

The time of the other industrial tasks needed to produce the item has been defined as 90 s, that is, the summation of times shown in (25), $\sum_{j \notin S_{\text{robot}}}^{k} t_j$.

The optimization algorithm presents different working times for the different examples, thus leading also to different benefits.

For instance, the case number 19, which has no constraints in both the jerk and the energy consumed, presents the maximum annual revenue (Figure 1). Contrary, case 4, with severe physical constraints, shows the minimum benefits.

The benefits for all examples obtained by means of the optimization algorithm are depicted in Figure 1. With the current demand, the mean value of the benefit for the 20 examples analyzed is 23,142.40 €/year, while the standard deviation is 90.32.

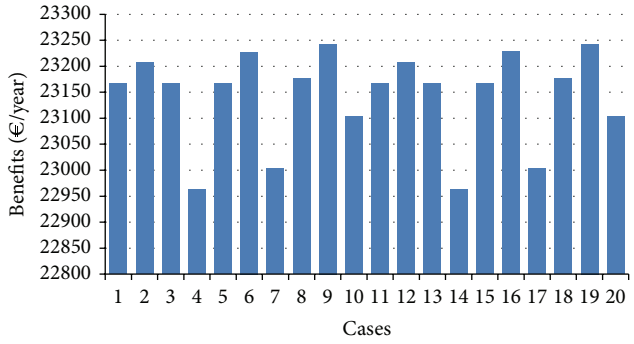

Figure 1: Annual revenue for each example based on the current demand.

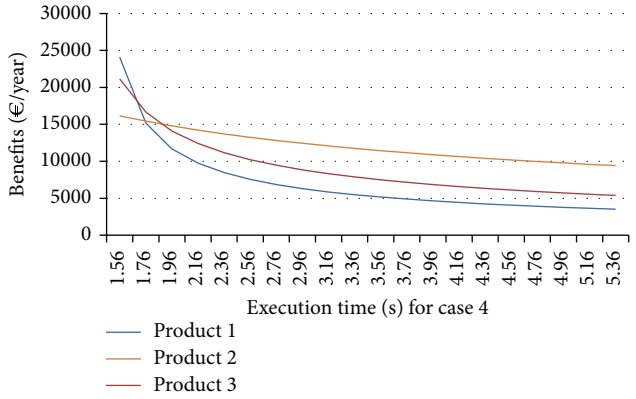

Figure 2: Pareto frontiers obtained with the optimization algorithm for "example 4" and the three different products manufactured.

The industrial tasks are carried out in only a few seconds, so that the time scale is based on seconds. Note that the GRASP working times are similar to those obtained with the developed optimization algorithm, although slightly higher.

Then total benefits are also similar to those reported in Figure 1, although the higher working times lead to lower benefits.

Consequently, a lower mean value of the benefits is obtained, that is, 23,132.84 €/year with a standard deviation of 96.02.

Note that different prices, costs, and number of items produced by year may lead to higher differences in benefits between the optimization algorithm and GRASP. However, the selected values are only intended to illustrate the importance of using efficient algorithms of robot trajectory optimization for saving time and reducing costs in production lines.

In addition, a new analysis is carried out using Pareto optimality to illustrate the loss of benefits on account of not using optimization algorithms.

For that, we consider that three different products can be manufactured and assembled in the same production line. The loss of benefits is represented by the Pareto fronts for three different products. They differ in their cumulative time to be manufactured and assembled but share the same working time ($t_{\min p}$) of the robot arm.

Then the minimum trajectory time for case 4 is used for the three products, in this example, 1.55 s. These products also differ in the total costs (without considering the energy costs), prices, and values of the parameter μ, which is intended to simulate different economic environments and market seasonality. The total cost of Product 1 is 0.8 €, Product 2 is 0.82 €, and Product 3 is 0.84 €, while the prices are 1.0 € for Product 1, 1.05 € for Product 2, and 1.03 € for Product 3. In this analysis, $t(S_k)$ has been defined as a cubic function of t_{minp}. The parameter μ takes the values for each product of 0.6, 0.5, and 0.55, respectively.

Consequently, if the market conditions do not change and the optimization algorithm is not used, the minimum trajectory time is not obtained.

In this scenario, there is a benefit loss due to the fact that robot arm may present higher working times.

Moreover, the multiobjective optimization problem allows obtaining the Pareto frontiers, which provides information about the trade-off of the decision variables. One main trade-off is between the benefits and the working time (i.e., the Pareto frontier). Results for example 4 are shown in Figure 2, considering the manufacturing of three different products. That is, the algorithm allows quantifying the benefit loss because of not using the optimization algorithm. Each solution in the front will have optimal objective function value, optimal value of variables, and the constraints value. All constraints will be satisfied by any solution in the Pareto optimal front.

It is worth pointing out that for the analyzed examples, the differences between their annual energy costs are almost negligible compared with the other costs.

Furthermore, the concept of nondomination sorting can be used to categorize each design within a set into a hierarchy of nondominated levels or fronts. Each different level of nondomination represents a relative distance from the Pareto front. The best nondominated front is closest to the Pareto front and each subsequent front lags further behind and is, hence, increasingly inferior.

Through this sorting, each design is associated with a front that defines the quality of the design relative to the rest of the group. To isolate the various fronts, the designs that belong to the nondominated subset of the entire group are first identified. These designs are the best in the group, the closest to (or members of) the Pareto front. For instance, in our multiobjective trajectory optimization, the nondominated subset (i.e., the best solution in terms of greater benefits) is represented by the Pareto frontier of Product 2 for times higher than 1.96 s (see Figure 2).

Any design belonging to this front is then temporarily set aside and another comparison process determines the next level of nondominated designs from the remaining population.

This nondominated subset is the front of Product 3 and the procedure is repeated until the entire population has been sorted into the appropriate level, that is, the Pareto frontier of Product 1.

Note, however, that for working times lower than 1.76 s the Pareto frontier of Product 1 dominates the frontier of Product 2. That is, higher benefits are expected to be achieved in the assembly line for Product 1.

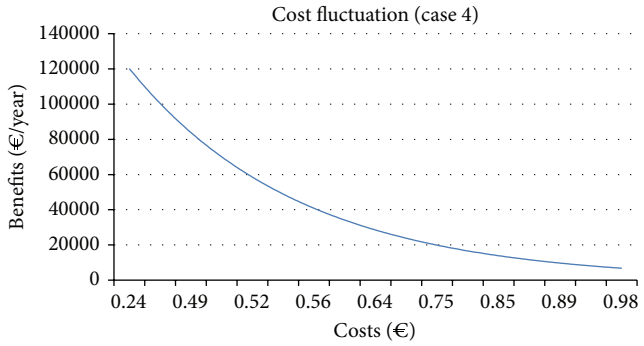

Figure 3: Benefits obtained for Product 1 and "example 4" versus costs.

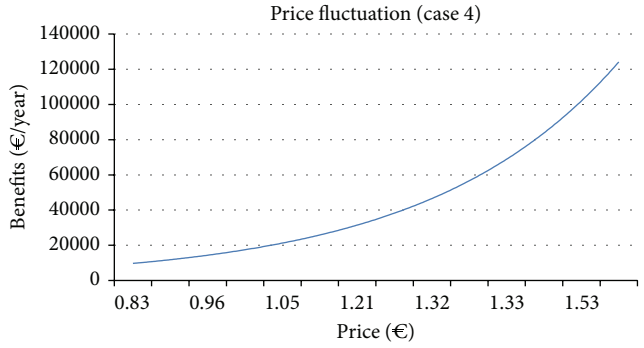

Figure 4: Benefits obtained for Product 1 and "example 4" versus price.

Additionally, for the twenty examples analyzed in this work, a normal distribution function of the cost (C) and the price (P) has been defined for a certain product based on the current market economic fluctuations. The benefits resulting from these market fluctuations are provided in Figures 3 and 4.

Figure 3 has been obtained when a normal distribution of the cost fluctuation is considered on account of hypothetical market changes. The statistics that define the normal distribution (mean and variance) are based on the current total costs above mentioned.

Figure 4 has been obtained when a normal distribution of the price fluctuation is considered on account of hypothetical market changes. The statistics that define the normal distribution (mean and variance) are based on the current prices.

For the sake of conciseness, the cost and price fluctuations have been considered only for Product 1. Its current market values are $m_c = 0.8\,€$ for costs and $m_p = 1\,€$ for prices, while the standard deviation (σ) is defined as $m/3$. With these functions, the market changes, regarding costs and prices, are intended to be modeled. $C(C = C_p * N_p)$ gives the cost to manufacture p products (N_p). P_p gives the revenue as defined by (2), that is, multiplying the price of the products manufactured.

These functions are sampled, with these values being used in the multiobjective optimization problem to obtain the results presented in Figures 3 and 4.

These figures show a hypothetic fluctuation in the market conditions with regard to costs and prices for case 4.

The fluctuations can be directly translated into benefits using the algorithm, thus allowing managers in the decision making process regarding which products should be manufactured, and also to define an efficient scheduling. Therefore, the design and planning of the robot tasks are considerably improved.

Finally, the algorithm has also been run for example 4 to simulate an increase in the future demand of a 20% compared with the current demand (i.e., a total of 137,778 products manufactured per year). The aim is to answer the question about how many extra working hours are needed to respond to that increase in the demand. The best solution is given by the optimization algorithm, since it reports the minimum trajectory time. The solution is that we need an additional 3,504 hours per year to meet such demand. This information can be used during the decision making process to design an efficient scheduling.

8. Conclusions

This work deals with trajectory planning of industrial robots for assembly lines in a cost-efficient way, thus overcoming limitations of the economic analysis methods which are currently available. It has been demonstrated that the multiobjective optimization algorithm finds the minimum time trajectory of industrial robots and the maximum annual revenue. This means greater annual revenue and better adaptation to market fluctuations in terms of costs, prices, and product demands. This is carried out by taking under consideration the physical constraints of the real working problem and the economic issues involved in the process. The proposed procedure has been successfully validated in different examples of robotic industrial tasks, where a better planning and design of production lines have been found.

The results from different examples have been compared using two methodologies, an optimization procedure and a simulation technique.

We have checked that the results obtained with the optimization procedure lead to lower working times and therefore greater annual revenues in comparison with those obtained with the simulation technique. Consequently the number of products manufactured and profits are increased while the number of the shifts required is reduced. The core of this paper is the procedure to obtain the best working times in real complex industrial robots with many degrees of freedom and mechanical constraints.

This has shown the worth of the methodology, with the overall objective of improving the profitability of production lines by designing flexible manufacturing systems. Furthermore, an entire set of equally optimal solutions for each process, the Pareto-optimal sets, are generated.

This provides information about the trade-offs between the different competing variables of the multiobjective optimization problem (i.e., working times and profits for the different products that can be manufactured at the production line).

Once the optimal time to perform each process is obtained in a cost-effective manner the results can be used for improving a wide variety of robotic industrial tasks. This can help managers in the decision making process regarding which products should be manufactured and to define an efficient scheduling to produce them. This is because it allows adjusting the number of shifts needed according to the existing demand of the products manufactured. Then companies may stay competitive because the algorithm allows a quick adaptation to rapidly changing markets.

As a further research this methodology will be extended to deal with new decision variables in the multiobjective optimization problem such as the energy consumed and time simultaneously since they are conflicting variables.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

This paper has been possible due to the funding from the Science and Innovation Ministry of the Spanish Government by means of the Researching and Technologic Development Project DPI2010-20814-C02-01 (IDEMOV).

References

[1] J. E. Bobrow, S. Dubowsky, and J. S. Gibson, "Time-optimal control of robotic manipulators along specied paths," *International Journal of Robotics Research*, vol. 4, no. 3, pp. 3–17, 1985.

[2] K. G. Shin and N. D. McKay, "Minimum -time control of robotic manipulators with geometric path constraints," *IEEE Transactions on Automatic Control*, vol. AC-30, no. 6, pp. 531–541, 1985.

[3] Y. Chen and A. A. Desrochers, "Structure of minimum-time control law for robotic manipulators with constrained paths," in *Proceedings of the IEEE International Conference on Robotics and Automation*, pp. 971–976, Scottsdale, Ariz, USA, May 1989.

[4] J. Mattmüller and D. Gisler, "Calculating a near time-optimal jerk-constrained trajectory along a specified smooth path," *International Journal of Advanced Manufacturing Technology*, vol. 45, no. 9-10, pp. 1007–1016, 2009.

[5] K. J. Kyriakopoulos and G. N. Saridis, "Minimum jerk path generation," in *Proceedings of the IEEE International Conference on Robotics and Automation*, vol. 1, pp. 364–369, Philadelphia, Pa, USA, 1988.

[6] D. Simon and C. Işık, "A trigonometric trajectory generator for robotic arms," *International Journal of Control*, vol. 57, no. 3, pp. 505–517, 1993.

[7] D. Constantinescu and E. A. Croft, "Smooth and time-optimal trajectory planning for industrial manipulators along specified paths," *Journal of Robotic Systems*, vol. 17, no. 5, pp. 233–249, 2000.

[8] P. Huang, Y. Xu, and B. Liang, "Global minimum-jerk trajectory planning of space manipulator," *International Journal of Control, Automation and Systems*, vol. 4, no. 4, pp. 405–413, 2006.

[9] D. Garg and C. Ruengcharungpong, "Force balance and energy optimization in cooperating manipulators," in *Proceedings of the 23rd Annual Pittsburgh Modeling and Simulation Conference*, pp. 2017–2024, Pittsburgh, Pa, USA, 1992.

[10] A. R. Hirakawa and A. Kawamura, "Proposal of trajectory generation for redundant manipulators using variational approach applied to minimization of consumed electrical energy," in *Proceedings of the 4th International Workshop on Advanced Motion Control (AMC '96)*, pp. 687–692, Mie, Japan, March 1996.

[11] B. H. Cho, B. S. Choi, and J. M. Lee, "Time-optimal trajectory planning for a robot system under torque and impulse constraints," *International Journal of Control, Automation and Systems*, vol. 4, no. 1, pp. 10–16, 2006.

[12] G. Field and Y. Stepanenko, "Iterative dynamic programming: an approach to minimum energy trajectory planning for robotic manipulators," in *Proceedings of the 13th IEEE International Conference on Robotics and Automation*, pp. 2755–2760, Minneapolis, Minn, USA, April 1996.

[13] X. F. Zha, "Optimal pose trajectory planning for robot manipulators," *Mechanism and Machine Theory*, vol. 37, no. 10, pp. 1063–1086, 2002.

[14] F. Valero, V. Mata, and A. Besa, "Trajectory planning in workspaces with obstacles taking into account the dynamic robot behaviour," *Mechanism and Machine Theory*, vol. 41, no. 5, pp. 525–536, 2006.

[15] F. J. Valero, J. L. Suñer, V. Mata, and F. J. Rubio, "Optimal time trajectory for robots with torque jerk and energy constraints," in *Proceedings of the ECCOMAS Thematic Conference on Multibody Dynamics*, 2009.

[16] F. Rubio, F. Valero, J. Sunyer, and V. Mata, "Direct step-by-step method for industrial robot path planning," *Industrial Robot*, vol. 36, no. 6, pp. 594–607, 2009.

[17] J. L. Suñer, F. J. Valero, J. J. Ródenas, and A. Besa, "Comparación entre procedimientos de solución de la interpolación por funciones splines para la planificación de trayectorias de robots industriales," in *Proceedings of the 8th Congreso Iberoamericano de Ingeniería Mecánica*, Cusco, Peru, October 2007.

[18] R. Sanders, "The pareto principle: its use and abuse," *Journal of Product & Brand Management*, vol. 1, no. 2, pp. 37–40, 1992.