Mechanism and Machine Theory 121 (2018) 502–514

Contents lists available at ScienceDirect

Mechanism and Machine Theory

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

Research paper

Extending the capabilities of robotic manipulators using trajectory optimization

André Gallant*, Clément Gosselin

Département de génie mécanique, Université Laval, Quebec, QC, Canada

ARTICLE INFO

Article history:
Received 23 May 2017
Revised 16 August 2017
Accepted 13 September 2017
Available online 24 November 2017

Keywords:
Trajectory optimization
Robot dynamics
Parametric trajectory
Robot payload capacity

ABSTRACT

The payload capacity of robotic manipulators is often considered to be the same throughout their workspace. However, the actual capacity largely depends on posture, velocity, acceleration and actuator limits. This work studies a method to increase the payload capacity of manipulators through trajectory optimization. This optimization is performed on a task basis and therefore, the load-carrying capacity varies from one task to another. Although the studied method is general and is not limited to specific robot architectures, an analysis of the method is conducted based on its application to a planar RR serial manipulator in a vertical plane. This manipulator is the most appropriate as a simple test case because most manipulators are built in such a way that most of the vertical motion of the manipulator is done by two parallel revolute joints: planar RR mechanism. Simulation and experimental results show that the payload capacity can be greatly increased compared to nominal values. It is also shown that, although the trajectories produced are not time optimal, the method is much more versatile and much simpler to implement than some other optimal control methods. The accompanying video provides a summary of the method and the results.

1. Introduction

The payload capacity of robotic manipulators is often considered to be constant throughout their workspace. One of two assumptions is often made; either that there is no significant change in payload capacity from one configuration to another; or that the payload capacity in the worst case is sufficient to accomplish a given task. For some manipulators, such as industrial robots with high reduction ratios in all joints, these are reasonable assumptions in practice. However, such assumptions tend to lead to very large and massive robots with relatively low payload capacities. In these manipulators, a large portion of the effort supplied by their actuators is used simply to support the large weight of subsequent actuators.

Much like a person cannot hold the same weight with a fully extended arm as closer to his or her body, a robot's actual payload capacity is configuration dependent. Moreover, the true payload capacity, especially for manipulators with low gear ratios, can be greatly influenced by the entire dynamics including velocity and acceleration.

In applications such as human-robot cooperation, humanoid robotics, and other applications, low reduction ratios may provide many advantages. For example, some applications require a high degree of backdrivability or high speed. Such applications are not well suited to joints with large gear ratios. The drawback of this is often a reduction in the joints' force/torque capabilities which limit the payload capacity. Another type of application that could benefit from the methods

* Corresponding author.
E-mail address: andre.gallant.1@ulaval.ca (A. Gallant).

https://doi.org/10.1016/j.mechmachtheory.2017.09.016
0094-114X/

studied in this work are applications where the tasks required are dynamic and are subject to frequent changes. In such cases, an easy to implement method that finds a trajectory that respects many non-linear constraints would be invaluable.

Thus, the goal of this paper is to explore the gains in payload capacity that can be achieved through motion optimization. Two objectives are targeted specifically: the methods must be easily and quickly implementable, and the methods must extend the capabilities of the manipulator in question by finding smooth trajectories that are experimentally feasible.

The motion optimization problem can be divided into two categories: geometric path following and trajectory planning with no prescribed path [1]. Geometric path following has been studied by many researchers with a calculus of variations approach. The calculus of variations applied to the motion of robotic manipulators is optimal control and often makes use of *Pontryagin's maximum principle* [2]. Solutions to the optimal control problem, especially the time-optimal control problem, for the motion of robotic manipulators following a prescribed path have been developed mostly in the mid 1980s [3–5]. An excellent review and explanation of such methods is presented in [6].

These methods are able to take into account the coupled and nonlinear equations of motion to generate a truly optimal path following trajectory. However, it was shown that at each point in a time-optimal trajectory, at least one of the control inputs, torque or force, is at a limit [7]. This results in trajectories that are difficult to follow in practice because such trajectories switch from maximum acceleration to maximum deceleration instantaneously (sometimes at multiple points in a single trajectory). This can cause heavy wear in a manipulator's joints.

In order to alleviate some of the practical problems of the aforementioned optimal control methods, many researchers have approached the geometric path following problem from other angles. Notably, by ensuring that the trajectory planner produces smooth trajectories by using parametric curves [8–10]. These methods are necessarily sub-optimal but produce smooth trajectories.

Geometric path following can include dynamic constraints and can result in great manipulator performances but the problem of finding a geometric path is very difficult. For certain tasks such as arc welding, and machining, a geometric path is naturally imposed by the task. However, in the case where increasing the payload capacity is desired, it is conceivable, even probable, that there are many paths that would be impossible to follow even with the best optimization methods. It is therefore clear that in order to succeed in significantly increasing the payload capacity of a manipulator using it's dynamic capabilities, one must be able to modify both its path and its trajectory which is the second category of motion optimization problems.

As with geometric path following, many researchers have approached the problem from an optimal control perspective by invoking *Pontryagin's maximum principle*. A few hybrid methods were developed [11,12] that rely on solving the geometric path following optimal control problem. These methods start by defining a geometric path using a parametric curve, then finding the optimal control law for following that path. The methods then change the path and iterate until a suitable path and trajectory are found. These methods produce optimal trajectories but not for optimal paths and produce the same type of high jerk trajectories as the path following optimal control methods.

Another approach is to apply Pontryagin's maximum principle directly, not on a specific path. Methods of this kind have been applied to many types of nonlinear control systems, such as the control of fighter jets [13]. In fact, any system of time varying nonlinear differential equations such as the Van der Pol equations can be studied in this way [13]. Extending these methods to systems of nonlinear differential equations with multiple inputs is not simple but can be done [14].

It can be shown through Pontryagin's maximum principle that solutions to the time optimal control problem of systems such as robotic manipulators are of type bang-bang [15]. Indeed, many researchers have developed methods that either force or converge towards bang-singular-bang trajectories for robotic manipulators [16–19]. Again, problems with the premature wear of robotic joints can occur due to the bang-bang nature of these solutions. Some have tried to smooth the trajectories by imposing constraints on jerk or the derivative of the control inputs [20].

The fact that time optimal trajectories are of type bang-singular-bang is a *necessary* condition but not a *sufficient* one, *i.e.*, not all bang-bang trajectories are optimal. As such, finding an optimal bang-bang trajectory is not trivial and requires, in the case of most robotic manipulators, computing the numerical integration of the equations of motion several times which is computationally very intensive.

Grid based combinatorial optimization methods have also been applied to the motion optimization problem [17,21,22], but these methods have been determined to be either prohibitively computationally intensive or to produce non-optimal trajectories.

One interesting approach is to guarantee smoothness by generating inherently smooth parametric joint trajectories. The kinematic and dynamic constraints that ensure the feasibility of a resulting trajectory can be imposed in the optimization process. Many methods have been developed that consider only kinematic constraints such as maximum velocity, acceleration and even jerk [23,24]. For many industrial manipulators, it can be reasonably assumed that the maximum acceleration that each actuator can produce, either changes little as a function of the manipulator's configuration, or that the least maximum acceleration is sufficient for the given task. When trying to increase the payload capacity of a manipulator, these assumptions on the kinematic constraints can be quite restrictive.

Due to the coupled and nonlinear nature of the equations of motion of most robotic manipulators, imposing effort (force or torque) constraints on the generation of trajectories is more difficult than imposing kinematic joint constraints. A gradient based method for optimizing the motion of robotic manipulators was developed in [25], where cubic B-splines were used to define the trajectory of each joint. Dynamic constraints are then imposed on a finite number of points along this trajectory. This methodology is very similar to the one used in this work. Others have extended this method by considering so-called

payload constraints [26]. These payload constraints were defined at the gripper level, that is, the constraints imposed that the payload object would not slip out of the grasp of the gripper. In this work, the payload is assumed to be grasped in such a manner that it would not slip out.

The goal of this work is to explore how these methods can be used to increase the payload capacity of robotic manipulators. In most other trajectory generation problems, finding an optimal trajectory means finding the best objective function. In the case of increasing the payload capacity, simply finding a feasible trajectory is a success. As such, in this work objective function optimization is considered secondary to satisfying the constraints, *i.e.,* finding *any* feasible trajectory. As such, an extensive analysis is done by evaluating the rate at which the method finds feasible solutions with a random initial guess in addition to the quality of the resulting trajectories. This work builds upon the preliminary results reported in [27] by extending the numerical analysis, adding a comparison of two different parametric curve types, and most importantly adding an experimental analysis.

Following this introduction, Section 2 states the optimal control problem in different forms for context. Section 3 presents the studied method and the different parametric curves used to define the joint trajectories. Section 4 provides a numerical analysis of the performance of the method using a planar serial RR manipulator as an example application of the studied method. Section 5 provides experimental validation of the results obtained. The work closes with Section 6 in which a discussion of the results and some conclusions are drawn.

2. Problem statement

Since the goal of this work is to apply trajectory optimization methods to the problem of lifting heavy payloads, the choice of the objective function is not as critical as finding feasible trajectories, *i.e.*, the constraints are more important than the objective function for this study. The most common objective function and arguably the most useful in practical applications is the minimization of the motion time. Therefore, the time optimal motion problem is studied in this work.

For reference, the time optimization problem as studied in the optimal control approach is stated as

$$\begin{aligned} \min_{\boldsymbol{\tau}(t)} \quad & \int_0^T \mathrm{d}t \\ \text{s.t.} \quad & \left.\begin{aligned} \boldsymbol{\tau}(t) &\succeq \boldsymbol{\tau}_{\min} \\ \boldsymbol{\tau}(t) &\preceq \boldsymbol{\tau}_{\max} \end{aligned}\right\} \forall\ t \in [0, T] \\ & \mathbf{x}(T) = \mathbf{x}_T \end{aligned} \tag{1}$$

where T is the final time, $\boldsymbol{\tau}$ is the time varying vector of joint effort (force or torque), $\mathbf{x}$ is the state of the manipulator (position and velocity of each joint) as a function of time, $\mathbf{x}_T$ is the final state of the manipulator that must be met in order to accomplish the prescribed task, and $\succeq$ and $\preceq$ represent piecewise inequality constraints. The torque constraints in Eq. (1) can be enforced relatively easily but the $\mathbf{x}(T) = \mathbf{x}_T$ constraint requires solving a two-point boundary problem for a system of $2f$ coupled nonlinear differential equations, where f stands for the number of degrees of freedom of the manipulator. This is extremely difficult to do even numerically.

The direct optimal control problem studied in this work is slightly different and can be stated as

$$\begin{aligned} \min_{\mathbf{x}(t)} \quad & T \\ \text{s.t.} \quad & \left.\begin{aligned} \boldsymbol{\tau}(\mathbf{x}(t)) &\succeq \boldsymbol{\tau}_{\min} \\ \boldsymbol{\tau}(\mathbf{x}(t)) &\preceq \boldsymbol{\tau}_{\max} \end{aligned}\right\} \forall\ t \in [0, T] \\ & \mathbf{x}(0) = \mathbf{x}_0 \\ & \mathbf{x}(T) = \mathbf{x}_T \end{aligned} \tag{2}$$

where $\mathbf{x}_0$ is the initial state of the manipulator. In this case, the joint effort vector $\boldsymbol{\tau}$ is not directly generated by the optimization method but can be computed from the generated trajectory with

$$\boldsymbol{\tau} = \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{c}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{g}(\mathbf{q}) \tag{3}$$

where $\mathbf{M}$ is the generalized inertia matrix, $\mathbf{c}$ includes the centrifugal and Coriolis effects, and $\mathbf{g}$ includes the gravitational terms. The joint positions, velocities and accelerations are represented by $\mathbf{q}$, $\dot{\mathbf{q}}$, and $\ddot{\mathbf{q}}$, respectively, and collectively, they define the kinematic state[1] of the manipulator, *i.e.*,

$$\mathbf{x} \equiv \begin{bmatrix} \mathbf{q} \\ \dot{\mathbf{q}} \\ \ddot{\mathbf{q}} \end{bmatrix}. \tag{4}$$

In the problem formulated in Eq. (2), respecting the constraints on the initial and final states of the manipulator is trivial whereas the constraints on the inputs (force or torque) are much more difficult to impose. This is in contrast to the optimal control problem stated in Eq. (1).

[1] In this work, the kinematic state includes the joint accelerations in contrast to the typical use of the word state which includes only the joint positions and velocities.

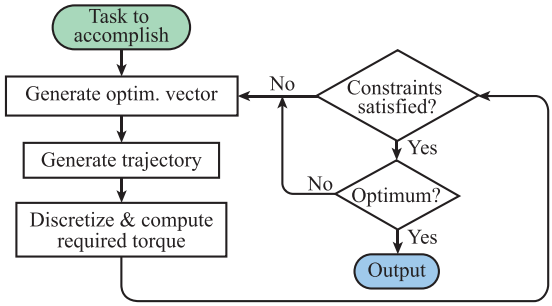


Fig. 1. Outline of optimization procedure.

Imposing constraints on a continuous time interval — such as the constraints on $\boldsymbol{\tau}$ — would be difficult to impose in a practical optimization procedure. The direct optimal control problem can thus be approximated by

$$\begin{aligned} \min_{\mathbf{x}(t)} \quad & T \\ \text{s.t.} \quad & \left.\begin{aligned} \boldsymbol{\tau}(\mathbf{x}(t_i)) \geq \boldsymbol{\tau}_{\min} \\ \boldsymbol{\tau}(\mathbf{x}(t_i)) \leq \boldsymbol{\tau}_{\max} \end{aligned}\right\} \forall\, i = 0, \ldots, N \\ & \mathbf{x}(0) = \mathbf{x}_0 \\ & \mathbf{x}(T) = \mathbf{x}_T \end{aligned} \tag{5}$$

where the generated trajectory is discretized into N portions of equal duration T/N. As such, the final time of the trajectory T would be equal to t_N. This is a sufficiently good approximation if N is reasonably large. The method studied in this work seeks near optimal solutions to the direct optimal control problem of Eq. (5).

3. Trajectory optimization method

3.1. Method overview

An outline of the optimization procedure used in this work is shown in Fig. 1. First, a task to be accomplished by the robot is defined. Since the geometric path of the motion is not predetermined, knowing $\mathbf{x}_0$ and $\mathbf{x}_T$ is sufficient to define a task. As it is shown in a subsequent section, the task does not necessarily need to include all the state parameters. For example a task could impose only position and velocity boundary conditions while another might impose position, velocity, acceleration, as well as jerk boundary conditions. This is entirely managed by the trajectory generation part of the optimization procedure.

Once a task is defined, an optimization vector is determined, either from an initial guess or modified from a previous iteration. From this a trajectory is generated on which the constraints from Eq. (5) are checked. The method iterates in this manner until a local optimum is found. It should be noted that this local optimum might not even represent a feasible trajectory, let alone a globally optimal one. In this work, the optimization procedure used is an SQP algorithm as implemented in Matlab's *fmincon* function.

3.2. Joint trajectory generation

As stated in Section 3.1, one of the most important steps of the optimization procedure is to generate smooth trajectories that satisfy the boundary conditions as prescribed by the task, for a given optimization vector. In this work, two types of parametric curves are studied and compared. The first generation technique makes use of cubic spline trajectories while the second uses nth degree Bernstein polynomials most commonly used to compute Bézier curves. Since the method is designed to optimize trajectories for manipulators with multiple joints, the optimization vector contains parameters for the generation of joint trajectories for each joint.

3.2.1. Cubic splines

Cubic splines are polynomials of degree three joined together at a number of *knots* [28]. For a trajectory of n spline segments, there are therefore $n-1$ knots. At each of these knots, the trajectory must be continuous in position, velocity and acceleration (PVA). In other words, the PVA at the end of each spline segment is constrained to be equal to the PVA at the beginning of the next segment.

As stated in Section 3.1, the task to be accomplished by the manipulator is defined by a number of initial and final conditions. For the spline trajectory generation technique used in this work, the position and velocity of each joint at the start and the end of the trajectory are defined in the task specification.

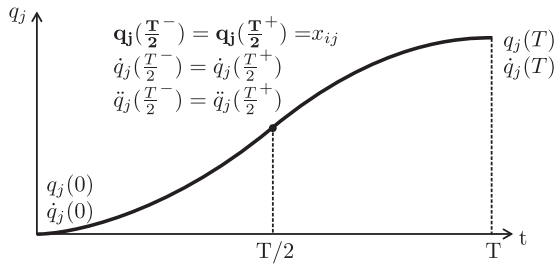


Fig. 2. Illustration of the constraints imposed on the cubic spline trajectories. The equation in bold indicates the constraint associated with the optimization variable. In this figure, q_j, $\dot{q}_j$, and $\ddot{q}_j$ represent the PVA of the jth joint.

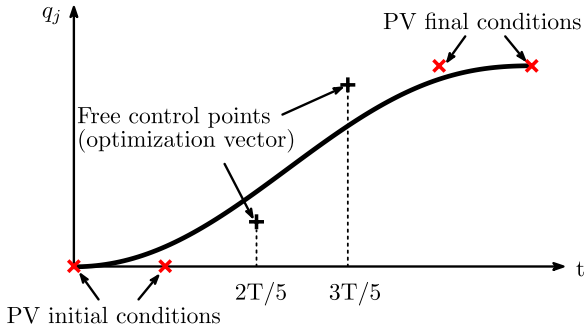


Fig. 3. An example Bernstein polynomial and its control points. In this case, position and velocity (PV) constraints are imposed at the beginning and the end of the trajectory of the joint which constrain the the first two and the last two control points, respectively.

Fig. 2 illustrates the constraints associated with cubic splines for an example with only two splines. All constraints are linear which makes the computation of the coefficients of each third order spline segment relatively straightforward. The only unknowns are the position of the joint at the knot (mid point) and the final trajectory time. These two values are components of the optimization vector.

For a manipulator with f degrees of freedom in which n_i is the number of knots in the spline used to describe the trajectory of joint i, the number of components of the optimization vector, noted c, is given by

$$c = \sum_{i=1}^{f} n_i + 1 \tag{6}$$

where the $+1$ term corresponds to the fact that the total time to perform the trajectory is also included in the optimization vector.

For a manipulator with 2 degrees of freedom, for example, if the number of knots at joints 1 and 2 are respectively n_1 and n_2, the total number of components in the optimization vector is $(n_1 + n_2 + 1)$ and this vector, noted $\mathbf{z}$ takes the following form:

$$\mathbf{z} = \begin{bmatrix} z_{11} & \cdots & z_{n_1 1} & z_{12} & \cdots & z_{n_2 2} & T \end{bmatrix}^{\mathrm{T}} \tag{7}$$

where z_{ij} is the ith optimization parameter of the jth joint, n_j is the total number of optimization variables of the jth joint (number of knots) and T is the total trajectory time. It is noted that the number of optimization variables is not necessarily equal for each joint.

3.2.2. Bernstein polynomials (Bézier curves)

Bernstein polynomials are similar to cubic splines in that there are a number of *knots* that define the curve; however, there are several key differences. The first difference is that the curve does not necessarily pass through the knots (called control points to distinguish them from spline knots). In fact, in general the curve only passes through the first and last control points. The second difference is that the generated curve is an nth degree polynomial, not a series of cubic polynomials. A third difference would be that any number of boundary conditions can be imposed by the task, as opposed to only position and velocity in the case of cubic splines as implemented in this work.

Fig. 3 shows an example of the type of Bézier curves implemented in this work. In this figure, the control points marked by a red "$\times$" are strictly constrained by the boundary conditions and the control points marked by a black "+" can be changed to alter the trajectory.

As the complete description of Bernstein polynomials is not the topic of this work, the details of the computation of the underlying Bernstein polynomials are omitted for brevity, though information on this topic is readily available [28]. There are, however, a few details specific to the implementation in this work that should be discussed; most notably, all control

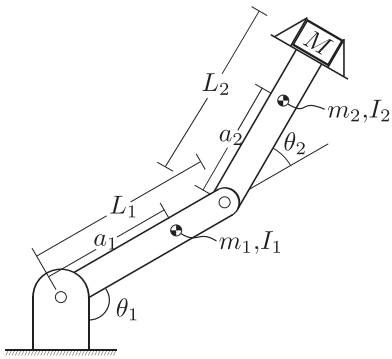


Fig. 4. Planar two-degree-of-freedom RR manipulator used as an example in this work.

Table 1
Kinematic and inertial properties of the planar two-degree-of-freedom RR manipulator studied in this work.

L_1	L_2	a_1	a_2	m_1	m_2	I_1	I_2
0.250	0.234	0.198	0.143	0.193	0.115	1.15e−3	4.99e−4

points are evenly distributed in time throughout the trajectory. This ensures that there are no loops in the trajectory, since a trajectory that goes backwards in time is nonsense. The fact that the t-coordinates are imposed by the number of control points and the final trajectory time T also reduces the number of parameters needed to define a trajectory.

The first n control points are used to ensure that the trajectory satisfies the n initial conditions and the last m control points bind the trajectory to satisfy the m final conditions. Since the remaining control points are free, their q-coordinate (joint position) values are defined in the optimization vector, much the same as with the cubic splines. The optimization vector is thus identical to the one presented in Eq. (7), but where n_j represents the number of *free* control points of each joint trajectory.

4. Numerical experiments

4.1. Example manipulator

For the purpose of illustration, the manipulator studied in this work is the planar two-degree-of-freedom RR mechanism shown in Fig. 4. The kinematic and inertial properties of this manipulator are summarized in Table 1.

This manipulator architecture may seem rather simple for the purpose of testing methods like the ones in this work; however, it was chosen quite deliberately. Observing, for example, a 6R general decoupled manipulator one can notice that the first joint's axis is vertical, therefore not exerting direct lifting forces. The second and third joints of such manipulators often have parallel axes in the horizontal plane, together constituting a planar RR manipulator. The last three joints have generally mush less control on the position of the end effector, rather they control its orientation. The choice of a vertically planar RR mechanism seems appropriate for the illustration of the goal of this work. Indeed, even a shoulder/elbow combination on a humanoid robot's arm can be seen as a planar RR manipulator when considering its lifting capacity. However, it should be noted that the results obtained with the planar RR manipulator will not necessarily be transferable to all other manipulators, but it is an appropriate base case study.

The units in this table are SI units: m for lengths, kg for masses and $\mathrm{kg}\cdot\mathrm{m}^2$ for moments of inertia. It is noted that the mass of the second actuator is considered to be part of the first link and is thus included in m_1, a_1 and I_1. Similarly, when a payload is added at the end-effector, m_2, a_2 and I_2 change accordingly. The values in Table 1 are those of the manipulator with no payload.

4.2. Actuator limits

For all numerical experiments performed in this work, joint force or torque limits were imposed to simulate a physical manipulator with motor limits. The actuator limits are

$$\boldsymbol{\tau}_{min} = \begin{bmatrix} -0.36 \text{ Nm} \\ -0.095 \text{ Nm} \end{bmatrix}, \quad \boldsymbol{\tau}_{max} = \begin{bmatrix} 0.36 \text{ Nm} \\ 0.095 \text{ Nm} \end{bmatrix} \tag{8}$$

With these actuator limits and the properties of the manipulator shown in Table 1, the manipulator is not capable of holding itself in a fully extended horizontal configuration in static mode. This effectively means that the manipulator has a negative payload capacity.

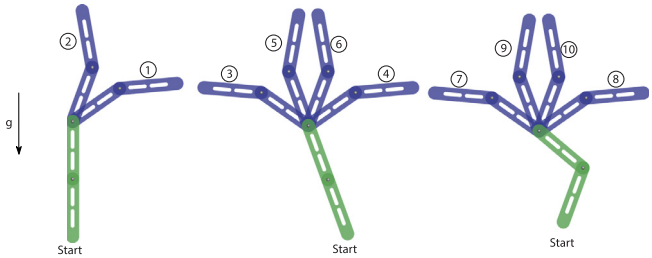


Fig. 5. Graphical representation of the 10 tasks to be performed. The green manipulators on the bottom represent the initial postures while the 10 numbered blue manipulators represent the final postures of all 10 tasks.

Table 2
Tasks to be executed by the manipulator.

Task	q_0	q_T	Task	q_0	q_T
(T1)	$[0, 0]^T$	$[125°, -30°]^T$	**(T2)**	$[0, 0]^T$	$[160°, 30°]^T$
(T3)	$[20°, 0]^T$	$[-125°, 30°]^T$	**(T4)**	$[20°, 0]^T$	$[125°, -30°]^T$
(T5)	$[20°, 0]^T$	$[-160°, -30°]^T$	**(T6)**	$[20°, 0]^T$	$[160°, 30°]^T$
(T7)	$[50°, -70°]^T$	$[-125°, 30°]^T$	**(T8)**	$[50°, -70°]^T$	$[125°, -30°]^T$
(T9)	$[50°, -70°]^T$	$[-160°, -30°]^T$	**(T10)**	$[50°, -70°]^T$	$[160°, 30°]^T$

4.3. Tasks

For all numerical experiments, the same 10 tasks are used as the testing context. These tasks are chosen such that the experimental prototype is capable of attaining the starting position reliably, as well as to provide a variety of tasks on which to evaluate the method. All tasks are pick-and-place type tasks where the manipulator begins and ends the trajectory with zero velocity and the position of each joint is fully constrained at both ends of the trajectory.

The tasks are shown in Fig. 5 and the values of the joint coordinates of the initial and final postures are summarised in Table 2. By observation of Fig. 5, some tasks may seem redundant at first glance, for example, the final postures of tasks T3 and T4 are the same but mirrored about the vertical axis. However, it is noted that lifting on the same side as the manipulator's initial posture, such as in task T4, is not at all the same task as being able to swing down before lifting on the other side, such as in task T3.

4.4. Methodology

In order to evaluate the effectiveness of the method, two sets of numerical experiments are conducted. The first is designed to evaluate the properties of the method such as the convergence rate and the objective value with the different trajectory generation techniques. The second set of numerical experiments aims to evaluate how much of a payload increase can be gained by employing the studied method.

4.4.1. Initial guess and optimization vector length

As mentioned above, the first numerical experiment seeks to evaluate the convergence rate and the objective with the different trajectory generation techniques. In order to achieve this evaluation, the optimization method is executed 50 times with random initial guesses for each task and for each trajectory generation technique. The initial guess is bounded between $\pm\pi$ rad for all but the last component of the optimization vector (final trajectory time) which is bounded between 0 and 10 s. Note that these bounds only apply to the generation of the initial guess and do not represent constraints in the optimization problem.

As the number of spline segments or the order of the Bernstein polynomials increases, the set of possible trajectories also increases. This increases the chances that a suitable solution exists but also increases the length of the optimization vector and thus the complexity of the optimization problem. In order to study this effect, the number of spline segments and the order of the polynomial are also varied. For the spline trajectories, 3 to 10 segments are used and polynomials of degree 5 to 12 are studied. This corresponds to the same optimization vector length, e.g., a spline trajectory with 3 segments needs 2 optimization parameters per joint plus the final trajectory time which is identical to a trajectory of polynomials of degree 5.

In total, the optimization method is executed 2500 times with cubic splines and another 2500 times with Bernstein polynomials.

4.4.2. Payload increase capabilities

Since the main goal of this research is to find a simple method to enable an increased payload capacity for robotic manipulators, the second set of experiments investigates the method's ability to find solutions with an increasingly heavy

Table 3
Payload masses used in the analysis.

Weight (g)	21.2	31.2	51.8	100.6	121.9	130.7	150.5

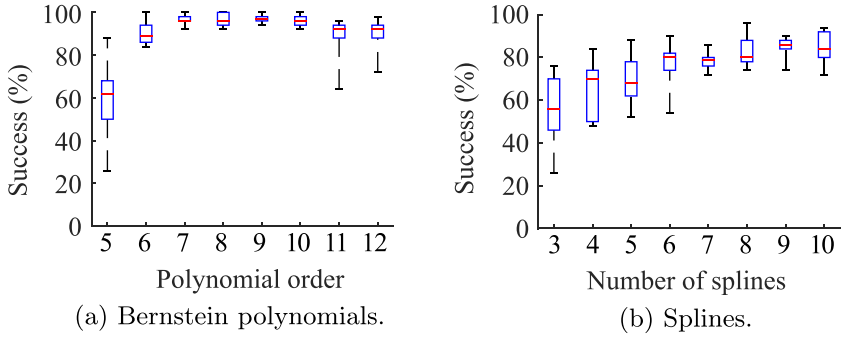


Fig. 6. Boxplots of the success rate of the 50 optimizations.

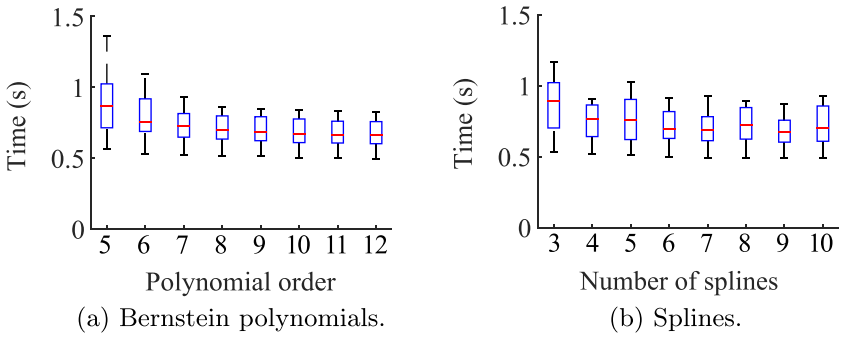


Fig. 7. Boxplots of the best trajectory time (objective function).

payload. Specifically, Table 3 presents the payload masses used in these numerical experiments. These masses correspond to the physical payloads used for the experimental validation discussed in a subsequent section.

The procedure used in the second set of experiments is as follows. First, for a given task and trajectory generation technique, the optimization procedure is executed with the lowest payload mass. The payload mass is then increased until the optimization is unable to find a solution after 50 randomly generated initial guesses. The experiment then proceeds to the next task and eventually to the next trajectory generation technique (higher order polynomial or greater number of spline segments).

4.5. Numerical results

The results of the first set of experiments are shown in Figs. 6 and 7. The processor used for these computations is an Intel Core i5-4690 Haswell 3.5 GHz Quad-Core CPU. The optimizations were performed in parallel with 4 threads using Matlab's Parallel Computing Toolbox and the time to perform 50 optimizations ranged from 30 to 100 s, increasing linearly with the length of the optimization vector.

The results of the second set of numerical experiments are shown in Fig. 8 which shows to which extent mass can be added to the manipulator for each of the 10 tasks. It can be observed that the method is actually very successful at increasing the payload capacity of the manipulator, especially for certain tasks. With the Bernstein polynomials, one can observe a reliable increase in payload capacity as the degree of the polynomial increases up to 11. The same cannot be said for the spline trajectories. Moreover, the Bernstein polynomials outperform the splines in all but the lower polynomial degrees.

Another interesting result is to observe the changes in the trajectory as mass is added to the end effector. Fig. 9 shows the example of task 1 with polynomials of degree 11 with all the masses presented in Table 3. Fig. 9a shows the trajectories for the first joint and Fig. 9b shows the trajectories for the second joint. Not surprisingly, the trajectory time increases as mass is added. It can also be observed that the method adapts very well to the extra payload: swinging back and forth, adding energy into the system, until the task can be accomplished. This is especially apparent in Fig. 9a.

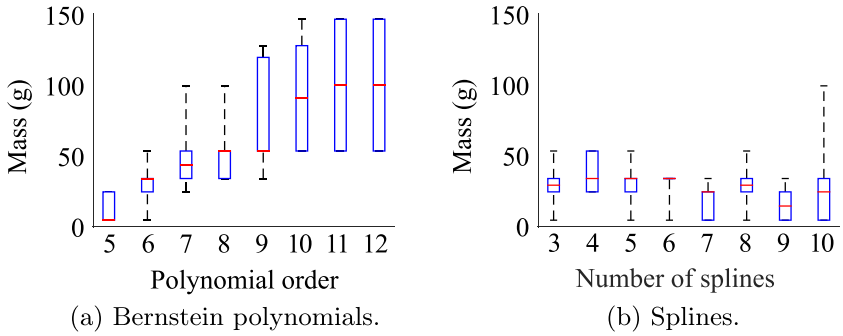


Fig. 8. Boxplots of the results of the second set of numerical experiments; a comparison between Bernstein polynomials and splines.

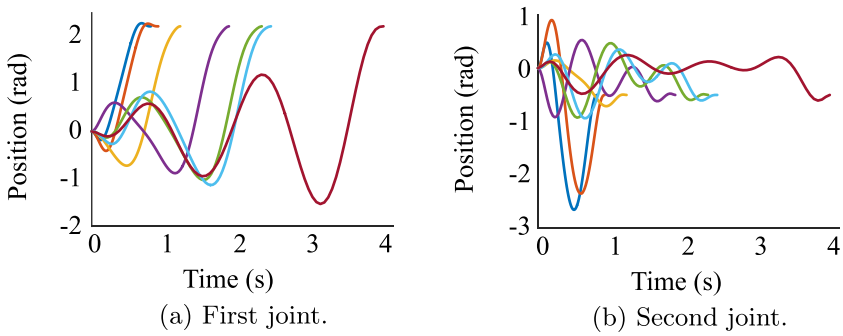


Fig. 9. Example trajectories of task 1 with polynomials of degree 11 as the payload is increased. The trajectory corresponding to each mass can easily be identified: the larger the payload, the longer the trajectory.

Fig. 10. Prototype of a planar RR manipulator mounted on a vertical plane with the extra payload masses.

5. Experimental validation

In order to validate the results reported in Section 4.5 on a real robot, a few of the results obtained numerically are tested experimentally. First, it is desired to determine if the trajectories found in the first set of numerical experiments of Section 4.4 are indeed feasible with the actuator limits imposed in the numerical analysis. Then, it is desired to verify that the manipulator is capable of lifting the masses as reported in the second set of numerical experiments (Figs. 8 and 9). These validations are performed on the prototype shown in Fig. 10 in which the brass weights used for the payload experiments can be seen attached to the end of the distal link. The geometric and inertial properties of the prototype are

Table 4
Specifications of the prototype's motors.

Motor	Gear ratio	Maximum torque
Motor 1	1:1	0.55 Nm
Motor 2	5.4:1	0.20 Nm

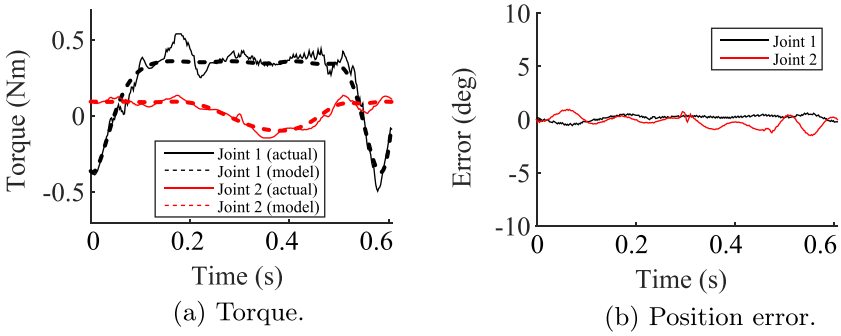


Fig. 11. Comparison of the numerical and experimental results for the RR manipulator performing task 1 with Bernstein polynomials of degree 11. No torque limits were imposed in the control while following the trajectory.

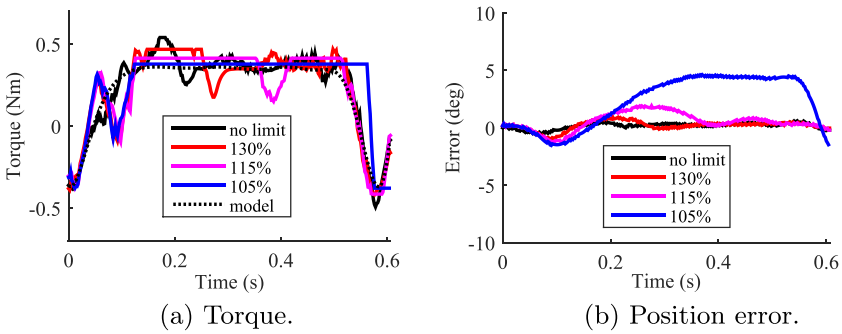


Fig. 12. An example of the RR manipulator performing task 1 with Bernstein polynomials of degree 11 with progressively decreasing torque limits in the controller (results are sown only the first joint).

those given in Table 1 of Section 4.1. The motors used for this prototype are low torque, high speed motors and their relevant specifications are listed in Table 4. Since the motors have little or no gear reduction, the maximum speed is not a limiting factor. Moreover, the maximum torques are relatively low, which is very appropriate to test the effectiveness of the studied method. It should be noted that the rated maximum torque of the actuators shown in Table 4 are not the same as the limits set in the optimization procedure. This enables the observation of the disparity between the numerical results and the experimental results.

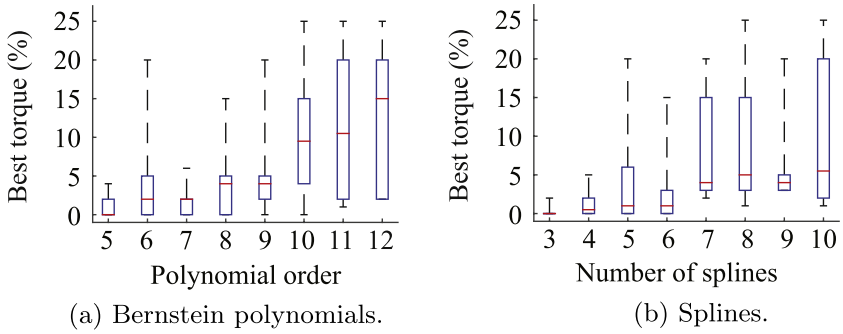
5.1. Validation of the first set of numerical experiments

Several optimization results obtained in the first set of numerical simulations were tested on the experimental RR manipulator while recording the torque supplied by each motor. Fig. 11 shows one such experiment with the fastest trajectory (solution with the best objective value) of the first task using Bernstein polynomials of degree 11.

It can be observed that, if the torque is not limited in the controller,[2] the torques used by the manipulator to perform the trajectory follows the torque computed using the numerical model quite well. However, the peak torque exceeds the limit by about 50%. If the torques are saturated in the controller of the manipulator, the results shown in Fig. 12 are obtained for the same trajectory. It can be observed that the error increases with stricter torque limits. In order to increase the clarity of the data represented in Fig. 12, only the results for the first joint are shown, but very similar results are obtained for the second joint.

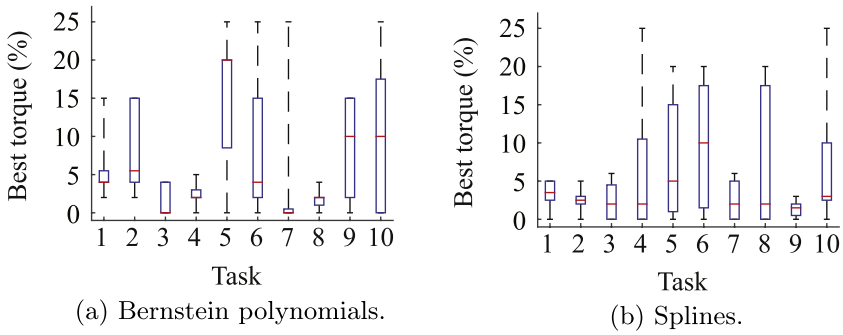
The results shown in Figs. 11 and 12 are for a specific example trajectory. In order to obtain more general results, the same procedure is repeated experimentally on the fastest trajectory of all combinations of tasks and parametrizations. Fig. 13

[2] In reality, there is a current limit imposed on the drive of each motor for safety. If one observes closely, there is a very short moment of saturation on the first joint just before the 0.2s mark. This saturation is extremely short and it does not significantly affect the trajectory.



(a) Bernstein polynomials. (b) Splines.

Fig. 13. Boxplot of the lowest possible torque saturation as a function of the parametrization used.



(a) Bernstein polynomials. (b) Splines.

Fig. 14. Boxplot of the lowest possible torque saturation as a function of the task.

shows the strictest torque limits that the manipulator was capable of tolerating while keeping the trajectory following error to less than 10° for all joints. These torque limits are expressed in terms of a percentage of overshoot with respect to the theoretical limits imposed in the numerical experiments of Section 4. In other words, a value of 0 means that the manipulator is able to perform the trajectory with no more torque than was imposed in the numerical experiments.

In Fig. 13, it can be observed that the chosen parametrization has some impact on the ability to limit the torque in practice. For instance, it seems that polynomials of degree 10 and above produce trajectories that are more difficult to follow experimentally. It should be noted that the trajectories produced are faster than the lower polynomial degrees as it was reported in the first set of numerical experiments of Section 4.5. This suggests that a balance should be struck between the theoretical performance of the motion optimization and the performance of the manipulator in practice.

Obviously, some tasks may also be easier than others. As it can be seen in Fig. 14, tasks in which the manipulator is able to swing before lifting on the opposite side, such as tasks 3 and 7, seem to follow the theoretical torques slightly more easily than the other tasks.

It is pointed out that the manipulator is able to perform all tasks with less than 25% more torque than theoretical limits. This validates that exploiting the dynamics of the manipulator is feasible in practice.

5.2. Validation of the payload increase capability

The second set of numerical experiments aimed to determine the maximum mass that can be added to the manipulator's end effector while performing the given pick-and-place tasks. In order to validate the results obtained numerically, several trajectories were tested on the experimental prototype. To demonstrate that the manipulator is effectively capable of lifting the masses used in the numerical experiments, Fig. 15 shows experimental data of the manipulator lifting an extra payload of 150.5 g with a trajectory of Bernstein polynomials of degree 10.

It can be observed in Fig. 15 that, when no limits are imposed in the control, the torque used by the manipulator to handle the 150.5 g payload exceeds the limits imposed in the numerical simulations. Similarly to what was presented for the experimental validation of the first set of numerical experiments, Fig. 16 shows the torque and position error of the first joint with for progressively decreasing torque limits.

It can be observed that when a significant payload mass is added[3] the torque cannot be limited as much as for smaller payload masses. Still, increasing the mass by 60% for a manipulator that is not even strong enough to hold itself in a fully extended horizontal configuration in static mode is a considerable increase in payload capacity.

[3] The moving parts of the manipulator have a total mass of 254 g, and hence a payload mass of 150 g is almost a 60% increase in mass at the end effector.

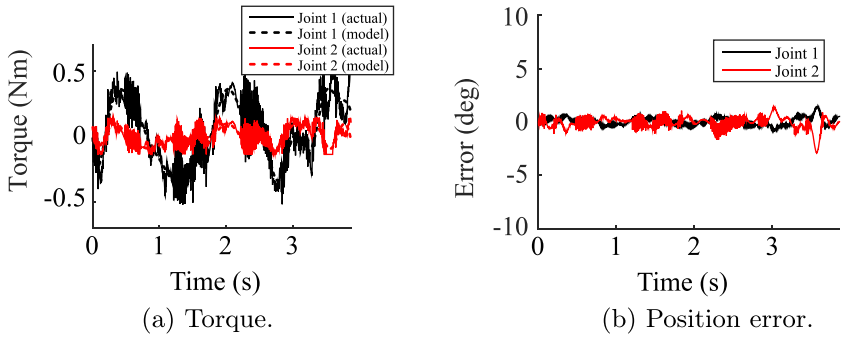


Fig. 15. An example of the RR manipulator performing task 1 with an extra payload of 150.5 g. No torque limits were imposed in the control.

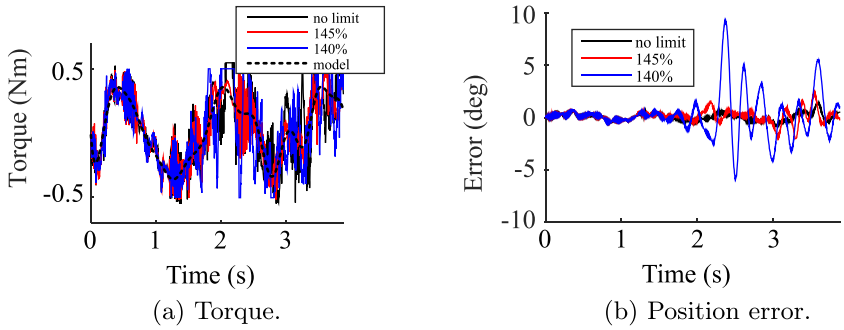


Fig. 16. An example of the RR manipulator performing task 1 with an extra payload mass of 150.5 g for progressively decreasing torque limits in the controller (first joint).

It can also be observed that the position error tends to be relatively small towards the end of the trajectories. This indicates that the saturation of the torques adversely affects the trajectory following during the high speed parts of the trajectory, but that this error can be reduced during the deceleration phase at the end of the trajectory. Since the path to be followed is not included in the task definition, reducing the position error at the final configuration is considered more important than reducing the position error during the trajectory.

6. Conclusion

This work studied the gains in load-carrying capacity that robotic manipulators can achieve when the motion is optimized such that the full dynamic capabilities of the actuators are exploited. In fact, it was shown through a case study on a RR serial manipulator that even a manipulator that is not capable of statically holding its own weight is capable of performing several tasks with up to an additional 60% mass at its end-effector. This indicates that a considerable increase in payload capacity can be achieved by optimizing the trajectories. These results were then experimentally validated.

The method chosen for this work is easy to implement and guaranties smooth trajectories that are suitable to use in practice on manipulators. While this work optimizes the trajectory time, the method is very flexible and any performance criterion can be used. The method is also very flexible in the constraints that can be imposed, such as jerk or other kinematic constraints, though these were not tested in this work. These properties make the method very attractive compared to other methods such as optimal control based methods. It must be noted, however that the method does not necessarily provide absolute optimality, but rather focuses on providing trajectories that can be readily executed by physical manipulators.

This work also compares two types of parametric curves, Bernstein polynomials and cubic splines. It was shown that Bernstein polynomials of degree 9 seem to provide the best trade-off between the computation time, objective value, and performance on a prototype manipulator for the case study of this work.

It should also be pointed out that, although a planar two-degree-of-freedom manipulator was used in this work to demonstrate the usefulness of such methods, the concept can readily be extended to any architecture of serial robot. The choice of a planar RR manipulator, as was discussed in Section 4.1, is based on the reasonable assumption that the payload carrying effort of even more complex architectures is often mainly done by two parallel joints. These two joints essentially form a planar RR mechanism as part of the manipulator as a whole. This suggests that the results in this work would work well in general. Of course, further investigation would be required to verify this.

Finally, the degree to which the payload of a manipulator could be increased for a given task would also vary with respect to the gear ratio of the manipulator's joints. The inertia of the manipulator's links, as perceived at the actuator, is reduced by the square of the gear ratio while and the inertia of the actuator's rotor remains constant. This means that

for manipulators with large gear ratios, the effects of a dynamic trajectory are somewhat limited. Correspondingly, one could reasonably expect to get limited increases in payload capacity using the methods studied in this work. However, as mentioned in the introduction, manipulators with large gear ratios are not necessarily the types of manipulators targeted by this work.

Acknowledgments

The authors would like to thank the Natural Sciences and Engineering Research Council of Canada (NSERC, grant no. DG-89715) and the Canada Research Chair program for their financial support (Canada Research Chair in Robotics and Mechatronics).

Supplementary material

Supplementary material associated with this article can be found, in the online version, at 10.1016/j.mechmachtheory.2017.09.016

References

[1] Y.K. Hwang, N. Ahuja, Gross motion planning — a survey, ACM Comput. Surveys (CSUR) 24 (3) (1992) 219–291.
[2] L.S. Pontryagin, V.G. Boltyanskii, R.V. Gamkrelidze, E.F. Mishchenko, The mathematical theory of optimal processes, Interscience (1962).
[3] J.E. Bobrow, S. Dubowsky, J.S. Gibson, On the optimal control of robotic manipulators with actuator constraints, in: American Control Conference, 1983, IEEE, 1983, pp. 782–787.
[4] K. Shin, N. McKay, Minimum-time control of robotic manipulators with geometric path constraints, IEEE Trans. Autom. Control 30 (6) (1985) 531–541.
[5] J.E. Bobrow, S. Dubowsky, J.S. Gibson, Time-optimal control of robotic manipulators along specified paths, Int. J. Rob. Res. 4 (3) (1985) 3–17.
[6] H. Choset, K.M. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L.E. Kavraki, S. Thrun, Principles of Robot Motion: Theory, Algorithms, and Implementations, MIT Press, 2005.
[7] Y. Chen, A.A. Desrochers, Structure of minimum-time control law for robotic manipulators with constrained paths, in: Robotics and Automation, 1989. Proceedings., 1989 IEEE International Conference on, IEEE, 1989, pp. 971–976.
[8] S.E. Thompson, R.V. Patel, Formulation of joint trajectories for industrial robots using b-splines, IEEE Trans. Ind. Electron. IE-34 (2) (1987) 192–199.
[9] D. Costantinescu, E.A. Croft, Smooth and time-optimal trajectory planning for industrial manipulators along specified paths, J. Rob. Syst. 17 (5) (2000) 233–249.
[10] A. Gasparetto, V. Zanotto, A new method for smooth trajectory planning of robot manipulators, Mech. Mach. Theory 42 (4) (2007) 455–471.
[11] V. Rajan, Minimum time trajectory planning, in: Robotics and Automation. Proceedings. 1985 IEEE International Conference on, IEEE, 2, 1985, pp. 759–764.
[12] K. Shin, N. McKay, Selection of near-minimum time geometric paths for robotic manipulators, IEEE Trans. Autom. Control 31 (6) (1986) 501–511.
[13] C.Y. Kaya, J.L. Noakes, Computations and time-optimal controls, Opt. Control Appl. Methods 17 (3) (1996) 171–185.
[14] S.K. Lucas, C.Y. Kaya, Switching-time computation for bang-bang control laws, in: Proceedings of the IEEE American Control Conference, IEEE, 1, 2001, pp. 176–181.
[15] D.E. Kirk, Optimal Control Theory: An Introduction, Prentice-Hall, 1970.
[16] A. Weinreb, A. Bryson, Optimal control of systems with hard control bounds, IEEE Trans. Autom. Control 30 (11) (1985) 1135–1138.
[17] G. Sahar, J.M. Hollerbach, Planning of minimum-time trajectories for robot arms, The Int. J. Rob. Res. 5 (3) (1986) 90–100.
[18] H. Geering, L. Guzzella, S. Hepner, C. Onder, Time-optimal motions of robots in assembly tasks, IEEE Trans. Autom. Control 31 (6) (1986) 512–518.
[19] E.-B. Meier, A.E. Ryson, Efficient algorithm for time-optimal control of a two-link manipulator, J. Guidance Control Dyn. 13 (5) (1990) 859–866.
[20] R. Bertrand, R. Epenoy, New smoothing techniques for solving bang–bang optimal control problems-numerical results and statistical interpretation, Opt. Control Appl. Methods 23 (4) (2002) 171–197.
[21] K. Shin, N. McKay, A dynamic programming approach to trajectory planning of robotic manipulators, IEEE Trans. Autom. Control 31 (6) (1986) 491–500.
[22] K. Kaltsoukalas, S. Makris, G. Chryssolouris, On generating the motion of industrial robot manipulators, Rob. Comput.-Integr. Manuf. 32 (2015) 65–71.
[23] C. Lin, P. Chang, J. Luh, Formulation and optimization of cubic polynomial joint trajectories for industrial robots, IEEE Trans. Autom. Control 28 (12) (1983) 1066–1074.
[24] S.S. Perumaal, N. Jawahar, Automated trajectory planner of industrial robot for pick-and-place task, Int. J. Adv. Rob. Syst. 10 (2) (2013) 100.
[25] Y.-C. Chen, Solving robot trajectory planning problems with uniform cubic b-splines, Opt. Control Appl. Methods 12 (4) (1991) 247–262.
[26] S.F.P. Saramago, M. Ceccarelli, An optimum robot path planning with payload constraints, Robotica 20 (4) (2002) 395.
[27] A. Gallant, C. Gosselin, Parametric trajectory optimisation for increased payload, Trans. Can. Soc. Mech. Eng. 40 (2) (2016) 125–137.
[28] G.E. Farin, Curves and Surfaces for CAGD: A Practical Guide, Morgan Kaufmann, 2002.