PARAMETRIC TRAJECTORY OPTIMISATION FOR INCREASED PAYLOAD

André Gallant[1,2], Clément Gosselin[1,3]
[1]*Département de génie mécanique, Université Laval*
[2]*Email: andre.gallant.1@ulaval.ca*
[3]*Email: gosselin@gmc.ulaval.ca*

ABSTRACT

The load-carrying capacity of manipulators is often considered to be the same throughout their workspace. However, the actual capacity of manipulators largely depends on their posture, their velocity, their acceleration and the limits of their actuators. In this paper, a method is proposed to increase the payload capacity of manipulators through trajectory optimisation. This optimisation is performed on a task basis and therefore, the load-carrying capacity varies from task to task. An extensive analysis of the method is conducted based on its application on a planar RR serial two degree-of-freedom manipulator. This analysis evaluates the ability of the method to find feasible trajectories and compares the results with those obtained using Bang-bang type methods. It is shown that, although the trajectories produced by the proposed method are not time optimal, the method is much more versatile and much simpler to implement than its Bang-bang counterparts.

Keywords: trajectory optimisation; manipulator dynamics; parametric trajectory.

OPTIMISATION DE TRAJECTOIRE PARAMÉTRIQUE POUR L'AUGMENTATION DE LA CHARGE UTILE

RÉSUMÉ

La charge utile des manipulateurs est souvent considérée comme étant constante sur l'ensemble de l'espace de travail. Il est cependant noté que la véritable charge utile d'un manipulateur dépend de sa posture, de sa vitesse, de son accélération ainsi que des limites de ses actionneurs. Dans cet article, une méthode d'optimisation de trajectoire est proposée pour augmenter la charge utile des manipulateurs. Cette optimisation est faite en fonction de chaque tâche. La charge utile varie alors selon la tâche à accomplir. Une analyse approfondie est effectuée en étudiant l'application de cette méthode à un manipulateur sériel plan à deux degrés de liberté d'architecture RR. Les résultats de la méthode méthode proposée sont aussi comparés aux résultats obtenus avec des méthodes de type Bang-bang. Il est montré que, bien que les trajectoires obtenues avec la méthode proposée ici ne sont pas optimales, la méthode est beaucoup plus flexible et beaucoup plus facile à mettre en œuvre que les méthodes de type Bang-bang.

Mots-clés : optimisation de trajectoire ; dynamique des manipulateurs ; trajectoire paramétrique.

1. INTRODUCTION

The payload capacity of a robotic manipulator is often considered constant throughout its workspace. As a result, robots are often very heavy compared to the payload they can carry. However, this limit is not necessarily representative of the actual load-carrying capacity of the manipulator for all motions it is required to perform. To use the analogy of the human arm, the maximum weight that a human can carry at an arm's length is not nearly as much as the weight they can carry closer to the body. Indeed, the payload capacity of a manipulator depends on its position, velocity, and acceleration as dictated by the limits of its actuators and its dynamics. In this work, a method is proposed to find trajectories that enable a manipulator to execute a prescribed task with a payload exceeding its normal load-carrying capacity.

A distinction is made between two types of tasks. The first are path following tasks where the path the manipulator must take is predetermined either in the joint space or the Cartesian space. The second type of task is where an initial state of the manipulator is known and a destination state is specified but the path between the two is not predetermined. These tasks are often referred to as pick and place tasks. Additionally, a distinction is made between a path and a trajectory [1]. A path is simply a sequence of positions / orientations that the manipulator must attain to arrive to a final destination. A path can be executed an infinite number of ways by varying the velocity at each position to execute the task more or less quickly. A trajectory, on the other hand, includes the time at which each position must be attained. Thus, the velocity and the acceleration of the manipulator is also defined at each point in a trajectory.

A considerable amount of work has been done on the time optimisation of path following tasks, *i.e.*, finding the fastest trajectory that follows the path, *e.g.*, [2, 3]. An excellent review and explanation of such methods is presented in [4]. Applications for path following trajectory optimisation include for example arc welding [5, 6]. This work studies the optimisation of trajectories for pick and place tasks.

In industrial robotics, the question of time optimality is very important. Indeed, time optimal trajectories can have a great impact on production time and therefore can increase the output of manufacturing. Accordingly, considerable research has been conducted in this field. Often, the time optimal problem is considered from the point of view of optimal control, a branch of the calculus of variations [7]. Optimal control theory has been applied to many fields including the control of fighter jets [8]. Indeed any system of time varying nonlinear differential equations such as the Van der Pol equations can be studied in this way [8]. These methods can also be applied to systems of non-linear differential equations with multiple inputs [9]. Perhaps the most studied methods of time optimal trajectories for pick and place tasks are Bang-bang type methods [10–12]. These methods are named Bang-bang because the input joint efforts of the resulting trajectories are always at their limits and abruptly alternate between their upper and lower limits. According to optimal control theory, Bang-bang trajectories are a necessary condition for the time optimal control problem. These trajectories are considered time optimal but they usually have high jerk due to the switching from one limit to the other. In order to reduce the jerk of the trajectories produced by Bang-bang methods, smoothing methods have been proposed [13].

Another approach is to generate trajectories in the joint space and impose the limits of the joint efforts as constraints. In [14], cubic splines where used to describe the trajectory of each joint and trajectories were found that satisfy the dynamic constraints [14, 15]. A limitation of both of these methods is that the positions of the joints were strictly increasing (monotonic), *i.e.*, path never backtracked away from the target posture for any of the joints. In applications where the joint efforts are limited such as the increased payload capacity explored in this work, a swinging motion is sometimes necessary in order to accumulate enough kinetic energy which such methods do not allow. A similar optimisation method than the one used in this paper is presented in [16] to find singularity free parametric trajectories.

Korayem *et al.* have conducted considerable research in determining the maximum load-carrying capacity for specific tasks of many types of manipulators including redundant manipulators, flexible joint manipulators and parallel manipulators [17–20].

The method presented in this paper seeks to find smooth parametric joint trajectories that perform pick and place tasks with a payload that would normally be too heavy for the manipulator. This result is accomplished by imposing constraints on the joint efforts using the inverse dynamics model of the manipulator. A secondary objective of reducing the total trajectory time is also included in order to compare with Bang-bang methods.

Section 2 of the paper presents the proposed method, the different parametrisations studied and the formulation of the objective function and constraints. Section 3 presents an example application of the proposed method to a planar serial RR manipulator. This section also presents the results of extensive analyses of the performance of the proposed method. The final section of this work, section 4, is a discussion of the results and presents some conclusions drawn.

2. METHOD

2.1. Method Overview

As stated in section 1, the method proposed in this work applies to pick and place tasks. No path to follow is imposed and it is presumed that no obstacles are present. For such tasks, the initial state of the manipulator should include at least the position and the velocity of each joint and can include any derivative such as the acceleration and jerk. For the state of the manipulator at the end of the task, any constraint on the position and its derivatives can be imposed.

Where the proposed method differs most from many of the time-optimal methods found in the literature is in how they compute the trajectory of the manipulator from a given optimisation vector. For example in Bang-bang methods, where the trajectory is defined in the space of the joint efforts (*e.g.*, actuator torques), the effort of each joint is presumed to be maximal at all times. Therefore, the optimisation vector of these methods represents the times at which the effort of each joint switches between its maximum and its minimum. The switching times are then modified until a trajectory is found that accomplishes the required task. The method proposed in this work does the opposite. It generates position, velocity, and acceleration joint trajectories, all of which accomplish the required task. It then adjusts the joint trajectory until one is found that satisfies the limits of the actuators. In this case, the optimisation vector represents the parameters of a parametric curve that defines the trajectory of each joint as a function of time. The parametric curves can be n-th degree polynomials, Fourier series, cubic splines, or any other parametric curve. In this work, series of cubic splines are studied. This parametrisation is further explained in section 2.2. The constraints that determine whether a given trajectory lies within the limits of the actuator's capabilities are presented in section 2.3.

A secondary goal of this research is to make the method as simple to implement as possible. Therefore, the optimisation algorithm used in this work is a sequential quadratic programming (SQP) algorithm as implemented in the *fmincon* function in the optimisation toolbox of MATLAB® [21].

The objective function used in this research is simply the final time of the trajectory. This allows a better comparison to Bang-bang methods. However, it should be noted that the proposed method is very flexible and can optimise virtually any criterion such as the energy consumption. It can also impose virtually any constraints such as on the velocity, the acceleration, or the jerk of the joint trajectories. This flexibility is not found in Bang-bang methods. Of course, the complexity of the objective function chosen will have an important impact on the computation effort required and indeed on the convergence of the optimisation. It is also noted that the method proposed in this paper is local and does not guaranty a global solution. However, since the primary goal of this research was to develop a simple method for feasible trajectory optimisation, global optimality is secondary.

The proposed method can be summarised by the following algorithm and Table 1 summarises the differences between the proposed method and the often used Bang-bang method.

- Define the task in terms of initial and final conditions.
- Choose a type of parametric curve for the trajectory of each joint.
- Determine an initial guess vector. The length of this vector depends on the parametric curve chosen.
- For each iteration:
  - Compute the trajectory that satisfies the task constraints and the current optimisation vector.
  - Discretise the trajectory into N points.
  - Compute the position, velocity, and acceleration of each joint at each point in the trajectory.
  - Compute the effort required at each joint for each point to follow the computed trajectory.
  - Compute the objective function (section 2.3).
  - Compute the dynamic constraints violation at each point of the trajectory for each joint.
  - Adjust the optimisation vector (SQP) to get closer to satisfying the constraints and minimising the objective function.

Table 1. Summary of the differences between the proposed method and Bang-bang methods.

Proposed Method	**Bang-bang Methods**
• Trajectories generated in the joint space.	• Trajectories generated in the joint effort space.
• All trajectories perform the required task.	• All trajectories satisfy the dynamic constraints.
• Looks for trajectories that satisfy the constraints.	• Looks for trajectories that perform the task.
• Continuous position, velocity and acceleration.	• Discontinuous acceleration at change points.
• Does not require numerical integration.	• Requires integration for dynamics simulation.
• Not time optimal.	• Potentially time optimal.
• Can optimise for any objective function.	• Difficult to optimise anything other than time.
• Can add constraints on joint velocity or acceleration.	• Difficult to implement additional constraints.

2.2. Parametrisation

The parametrisation studied in this work is based on a series of cubic splines. Cubic splines are polynomials of degree three joined together at a number of *knots*. For a trajectory of n splines, 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 is constrained to be equal to the PVA at the beginning of the next spline.

As stated above in section 2.1, the tasks to be accomplished are defined by a number of initial and final position, velocity and/or acceleration conditions. Therefore, a number of linear constraints are imposed on the selection of the splines that ensures that the task is performed. In this work, tasks were defined by position and velocity (PV) initial conditions as well as PV final conditions.

With a parametrisation of splines, the boundary conditions apply constraints on two of the splines. Specifically, the initial conditions apply constraints on the first spline and the final conditions apply constraints on the last spline. Cubic splines of this kind are fully defined when the position of the trajectory at each of the knots is known, assuming the final time T is also known. There are therefore as many optimisation variables as there are knots. The splines in this work, are all of equal length. For example, for a parametrisation of three splines, the two knots are at $T/3$ and $2T/3$.

Figure 1 illustrates the constraints associated with cubic splines for an example with two splines. These constraints are all linear and thus the coefficients of each spline for given values of z can be readily computed.

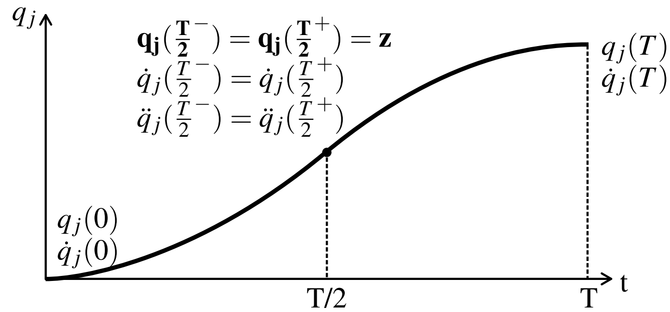


Fig. 1. Visual representation of the constraints imposed on the cubic spline trajectories. The equation in bold indicates the constraint associated with the optimisation variable. In this figure, q_j, $\dot{q}_j$, and $\ddot{q}_j$ represent the PVA of the j-th joint.

It should be noted that this method defines the trajectory of a single joint and requires that the final time of the trajectory be known. Since all the joints should finish the trajectory simultaneously, the final time should be common to all joints. Thus, an example of an optimisation vector for a two dof manipulator would have the following form:

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

where z_{ij} is the i-th optimisation parameter of the j-th joint, n_j is the total number of optimisation variables of the j-th joint (number of knots). Note that the number of optimisation variables is not necessarily equal for each joint.

2.3. Objective Function and Nonlinear Dynamic Constraints

Now that a method of generating joint trajectories is available and the associated optimisation vector has been determined, a means of evaluating the trajectories is needed. The goal of this work is to explore methods of enabling a manipulator to perform a task that would normally exceed its capabilities, thus increasing its payload. As a result, the objective function chosen is not a critical part of the optimisation, and any measure can be optimised to suit the needs of the specific application. In this work, the total trajectory time has been chosen in order to facilitate the comparison with Bang-bang methods.

So far, the proposed method allows the generation of trajectories that satisfy the boundary conditions imposed by the task and seeks to minimise the total time. If no other constraints are imposed, a trajectory performed in an infinitesimal time would be optimal and perfectly acceptable. However, the joint efforts needed to follow such a trajectory would be near infinity and therefore not feasible. For that reason, some additional constraints must be imposed to find feasible trajectories.

For a trajectory to be feasible, the generalised efforts needed to follow it must be within the limits of the capabilities of the manipulator's actuators. Generally, the required efforts at a given instant for a serial manipulator are a function of the joint PVA and can be expressed in the following form:

$$\boldsymbol{\tau} = \mathbf{A}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{b}(\mathbf{q}, \dot{\mathbf{q}}) + \mathbf{c}(\mathbf{q}) \tag{2}$$

where $\boldsymbol{\tau}$ is the generalised efforts vector, $\mathbf{A}$ is the inertial matrix, $\mathbf{b}$ contains the Coriolis and centrifugal terms, $\mathbf{c}$ contains the gravitational terms and $\mathbf{q}$, $\dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$ are the joint PVA vectors.

For a given optimisation vector from equation (1), the trajectory is fully defined and the PVA of all joints at all times during the trajectory can be obtained analytically. Therefore, once a trajectory is defined, the joint efforts can also be obtained analytically at any point of the trajectory using equation (2). Inequality constraints can then be imposed on a discretised trajectory to ensure that the manipulator is able to perform the trajectory and therefore the task. At each time t_i the nonlinear dynamics constraints of a given joint j are of the following form:

$$\tau_{j,\min} \le \tau_j(t_i) \le \tau_{j,\max} \tag{3}$$

where $\tau_{j,\min}$ and $\tau_{j,\max}$ are the minimum and maximum efforts that the j-th joint can produce.

Formally, the optimisation problem is expressed as:

$$\begin{array}{ll} \min & T(\mathbf{z}) \\ \text{s.t.} & \left.\begin{array}{l} \tau(\mathbf{z},t_i) \ge \tau_{j,\min} \\ \tau(\mathbf{z},t_i) \le \tau_{j,\max} \end{array}\right\} \quad \forall \quad i=1..N, \quad j=1..M \\ & T(\mathbf{z}) > 0 \end{array} \tag{4}$$

where $T(\mathbf{z})$ is simply the last component of $\mathbf{z}$, N is the number of discretisation points used for the verification of the dynamic constraints, and M is the number of degrees-of-freedom of the manipulator.

In summary, for the method presented in this work, all trajectories generated by the parametrisation perform the task required but the method tries to find one that is feasible. This is in contrast to Bang-bang methods where all trajectories generated are feasible but the method tries to find one that performs the required task. One of the main advantages of the proposed method is that it does not require a numerical integration of the dynamics model in order to compute the objective function or the nonlinear constraints. Conversely, this computationally costly numerical integration is required for the computing of the objective function for Bang-bang methods.

3. CASE STUDY AND METHOD EVALUATION

3.1. Manipulator

The manipulator studied in this work is the planar RR mechanism shown in Fig. 2. The geometric and inertial properties of this manipulator are summarised in Table 2.

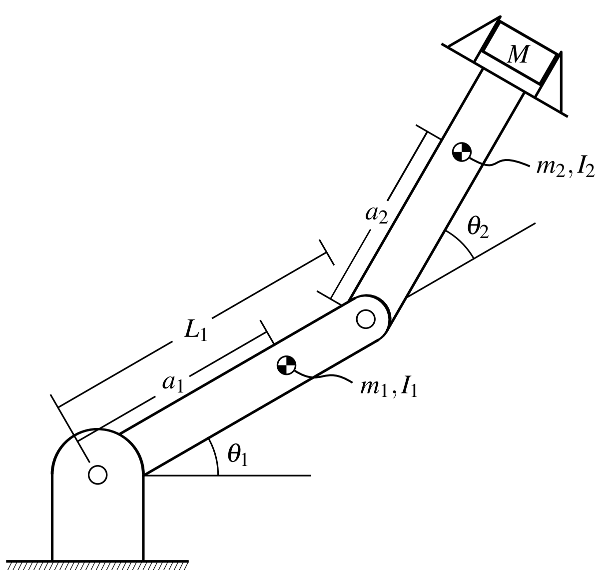


Fig. 2. RR manipulator studied in this work.

Table 2. Geometric and inertial parameters of the RR manipulator studied in this work.

L_1	a_1	a_2	m_1	m_2	I_1	I_2
0.250	0.198	0.143	0.193	0.115	1.149e-3	4.993e-04

The units in this table are SI units: m for lengths, kg for masses and $kg \cdot m^2$ for moments of inertia. It is noted that the length of the second link is not needed in this analysis since the mass m_2, its relative centre a_2 and its moment of inertia I_2 take into account the mass of the second link as well as the payload M. Similarly, the mass of the second actuator can be taken into account in m_1, a_1, and I_1.

3.2. Analysis Methodology

In order to evaluate the performance of the proposed method, two analyses have been conducted. Both analyses have many points in common. Firstly, four tasks have been established on which the proposed method was evaluated. Secondly, the effect of the number of splines has been studied. Lastly, the effect of the initial guess on the convergence of the optimisation has been studied. Both analyses use the same tasks, number of splines, and method for generating random initial guesses.

3.2.1. Prescribed Tasks

In this work, four tasks have been established on which the performance of the proposed method was evaluated. Each of these tasks have Bang-bang solutions. All four tasks are defined by PV initial conditions as well as PV final conditions and are presented in Table 3. In this table, $q_j(0)$ and $\dot{q}_j(0)$ are the PV initial conditions of the j-th joint and $q_j(T)$ and $\dot{q}_j(T)$ are the PV final conditions of the j-th joint. A visual representation of these tasks is shown in Fig. 3.

Table 3. Tasks to be executed by the manipulator during the analyses performed in this work.

Task	$q_1(0)$	$\dot{q}_1(0)$	$q_2(0)$	$\dot{q}_2(0)$	$q_1(T)$	$\dot{q}_1(T)$	$q_2(T)$	$\dot{q}_2(T)$
(T1)	-2.09	0	0	0	1.01	0	0.26	0
(T2)	-4.01	0	0	0	-0.29	0	0.17	0
(T3)	-1.57	0	0	0	0.87	4.20	-1.61	3.20
(T4)	0.79	0	-0.79	0	-1.57	10.74	0	-5.49

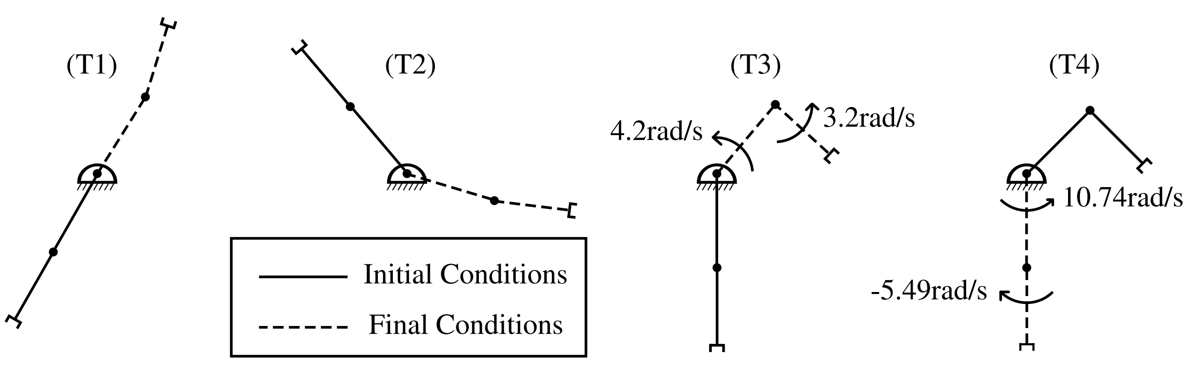


Fig. 3. A visual representation of the four tasks to be performed.

It can be observed that two of the tasks (T1 and T2) are pick and place type tasks where the velocity of each joint is zero at the beginning and at the end of the trajectory. The other two trajectories (T3 and T4) start at standstill but end with a given joint velocity. The latter tasks can be much more difficult to accomplish.

3.2.2. Number of Splines

As the number of splines increases, the set of possible trajectories also increases. That is, trajectories with many knots whose position can be changed have more freedom and can potentially generate more interesting solutions. On the other hand, an increase in the number of splines also leads to an increase in the number of optimisation variables. This could lead to an increase in the solution time and indeed the difficulty of finding a suitable trajectory that satisfies the dynamic constraints set by the joint effort limits. In order to study the effect of the number of splines on the performance of the method, series of two to ten splines were evaluated. This corresponds to optimisations of one to nine variables per joint.

3.2.3. Initial Guess

The convergence of local optimisation methods such as the SQP algorithm used in this work, can be very sensitive to the initial guess provided. Therefore, a large number of randomly generated initial guesses were sampled in order to study the sensitivity to the initial guess.

In order to limit the generation of nonsensical initial guesses, some constraints on the range of the random number generator were imposed. These constraints should be chosen so as to limit the initial guesses to sensible values but should not be excessively restrictive. As such, randomly generated positions between πrad and $-\pi$rad have been used in this study.

3.2.4. First Analysis

The first analysis sought to evaluate the ability of the method to perform the four tasks described in section 3.2.1. This evaluation was performed with one hundred random initial guesses for each number of splines. The criteria measured in this analysis were the computation time, the rate of success, and the trajectory time (objective function), the latter of which was compared to the trajectory time of the Bang-bang solution.

For all tasks, the effort limits (dynamic constraints from equation (3)) were set to a fixed percentage of the static effort. The static effort here is defined as the effort required to maintain the manipulator in its horizontal posture, *i.e.*, where $\theta_1 = \theta_2 = 0$. For this analysis, the effort limits were set to 50% of the static effort for the first joint and 75% for the second joint. The reasoning behind the choice of different percentages for the joints is twofold. First, the actuators close to the base of the manipulator tend to be the largest and strongest. Therefore, the benefits of limiting the effort of these joints would often be amplified with respect to the other joints. The second reason is that the effort at the first joint is more configuration dependent than the effort at the second joint. Specifically, the effort at the first joint depends on the position of the second. As a result, reductions in the effort of the joints farther from the base tend to be harder to achieve. It should be noted that these assumptions are not necessarily true for all manipulators or applications.

3.2.5. Second Analysis

Since the main goal of this research was to find a simple method to enable an increased payload for robotic manipulators, the second analysis studied the method's ability to find solutions with progessively stricter dynamic constraints. In other words the effort limits were lowered until the optimisation method was no longer able to find a feasible trajectory. As such, the values of the dynamic constraints were not fixed values but rather decreased as solutions were found. Specifically, Table 4 presents the values used for the second analysis. These values maintain the same 2:3 ratio as the effort limits used in the first analysis.

The procedure of the second analysis was the following. First, for a given task and number of splines, the optimisation procedure was executed with the largest effort limits from Table 4 (100% and 150%). This procedure was repeated with randomly generated initial guesses until a feasible trajectory was found. Then

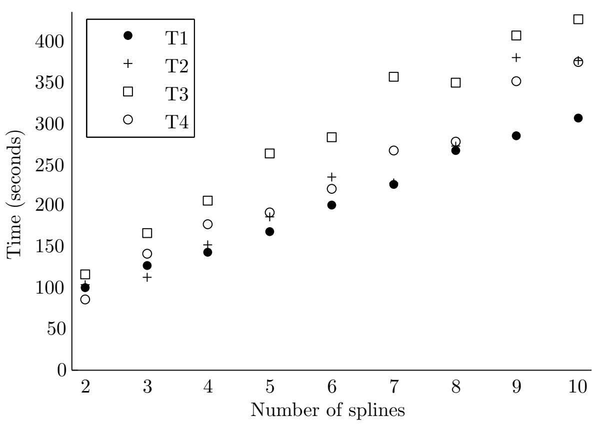
Table 4. List of the fifteen effort limits of each joint as a percentage of the static effort for the second analysis conducted in this work.

Joint 1 (%)	100	90	80	70	60	50	40	34	28	24	20	16	12	10	8
Joint 2 (%)	150	135	120	105	90	75	60	51	42	36	30	24	18	15	12

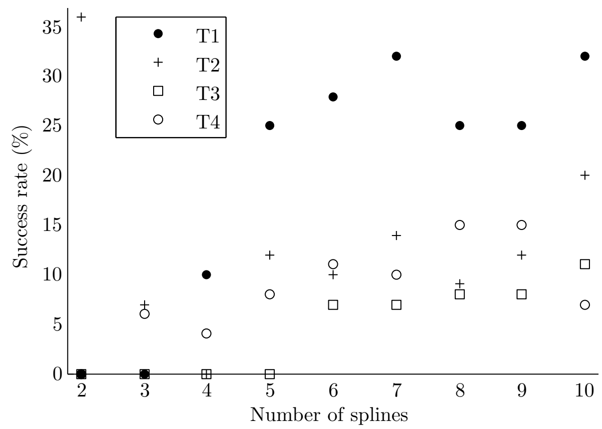
the optimisation was restarted with the next effort limits and so on until no trajectory could be found after two hundred initial guesses. When no feasible trajectory could be found, the analysis moved on to the next task and eventually to the next number of splines.

3.3. Results

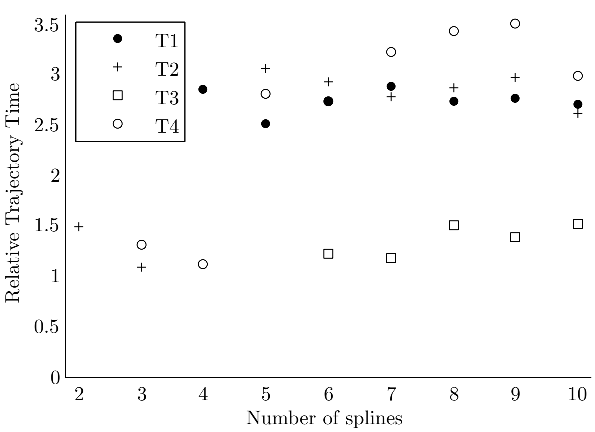
The results of the first analysis are shown in Fig. 4. Figure 4a shows the computation time, Fig. 4b shows the success rate of the of the method with randomly generated initial guesses, and Fig. 4c and 4d show, respectively, the average and the minimum trajectory times relative to the corresponding optimal Bang-bang trajectory. All of these figures are a function of the number of splines and the prescribed task.



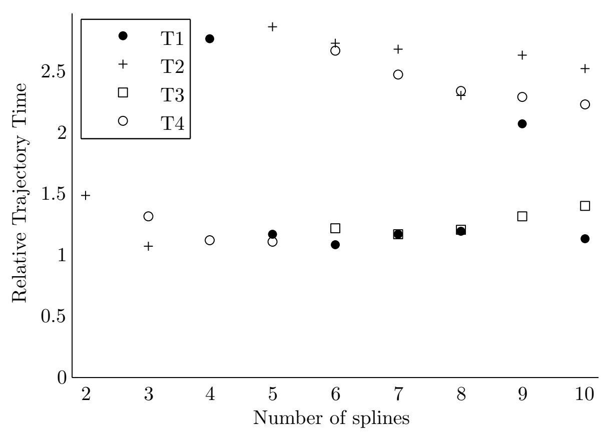
(a) Computation time for 100 optimisations (4 threads). Lower is better.



(b) Number of successful optimisations (%). Higher is better.



(c) Average trajectory time relative to Bang-bang trajectory time. Lower is better.

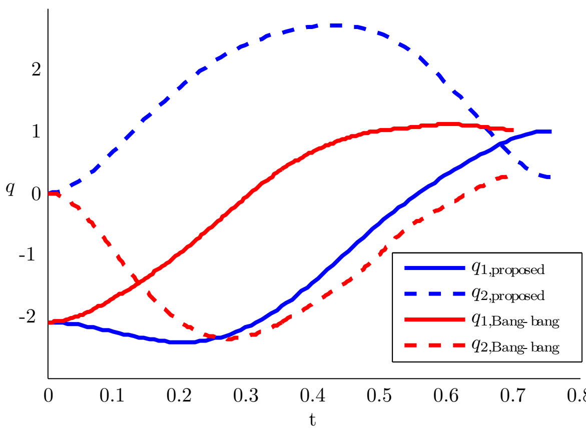


(d) Min trajectory time relative to Bang-bang trajectory time. Lower is better.

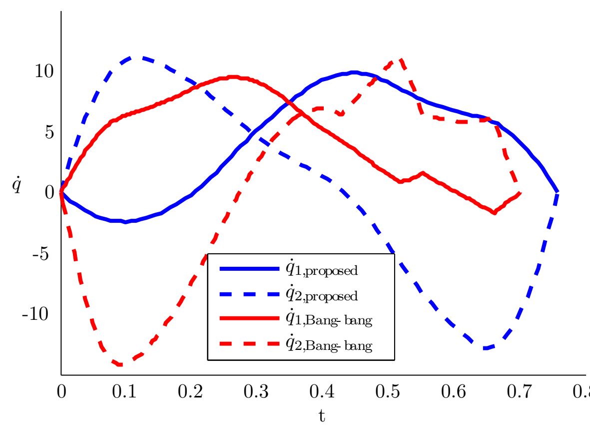
Fig. 4. Results of the first analysis.

From Fig. 4a, it can be observed that, unsurprisingly, the computation time increases with the number of splines. However, it seems that the likelihood of finding successful trajectories from randomly generated initial guesses and the ability to adapt to difficult tasks is improved with an increased number of splines. Only the successful trajectories are included in Fig. 4c and 4d, *i.e.*, trajectories that satisfied the dynamic constraints. A value of 1 in these figures means that the computed trajectory time is the same as the Bang-bang trajectory time. The trajectory times of the Bang-bang optimal trajectories for each task were 0.7s, 0.64s, 0.9s, and 0.85s, respectively.

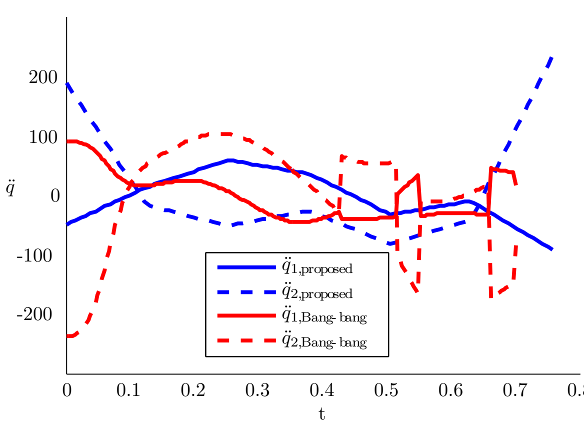
It can be observed that the gains in performance from an increase in the number of splines seem to taper off after six splines. Since the computation time continues to increase with more than six splines, this seems to be the optimal number of splines for this application. Note that for other applications, this number might be different. It can also be observed that for certain tasks, the best trajectories obtained with the proposed method perform nearly as well as the optimal Bang-bang trajectories. This is remarkable due to the fact that the trajectories obtained with the proposed method are smooth in position, velocity, and acceleration. To illustrate this, Fig. 5 shows a comparison between one of the trajectories obtained with six splines and the optimal Bang-bang trajectory for the first task. In this figure, it is observed that the trajectory obtained with the proposed method is continuous in acceleration and the Bang-bang trajectory has discontinuous acceleration whenever one of the joint efforts switches from one limit to the other.



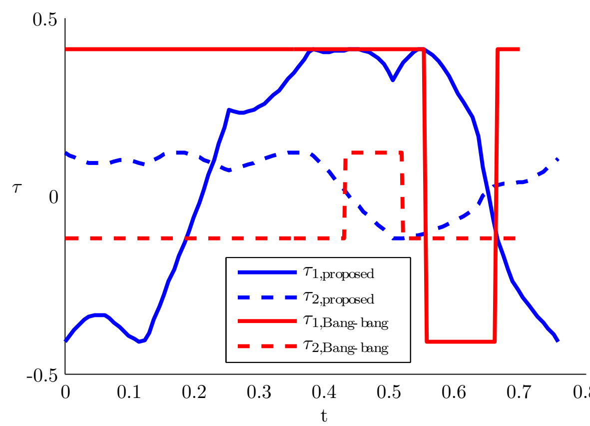
(a) Joint positions.



(b) Joint velocities.



(c) Joint accelerations.



(d) Joint efforts.

Fig. 5. Comparison of trajectory obtained with the proposed method and Bang-bang method.

The results of the second analysis are shown in Fig. 6 which shows to which extent the actuator effort limits can be reduced. It can be observed that the proposed method is actually quite capable of increasing the payload capacity of the manipulator especially for certain tasks. Again the ideal number of splines for this application seems to be six. The performance for certain tasks is sometimes even diminished with higher numbers of splines while for other tasks, the performance increases slightly. This fluctuation is most likely due to the arbitrary nature of the randomly generated initial guesses. It should also be noted that the proposed method is able to find smooth solutions to most tasks with only 30% of the static effort for the first joint (45% for the second joint). For both analyses, the number of discretisation points N where the dynamic constraints were verified was one hundred.

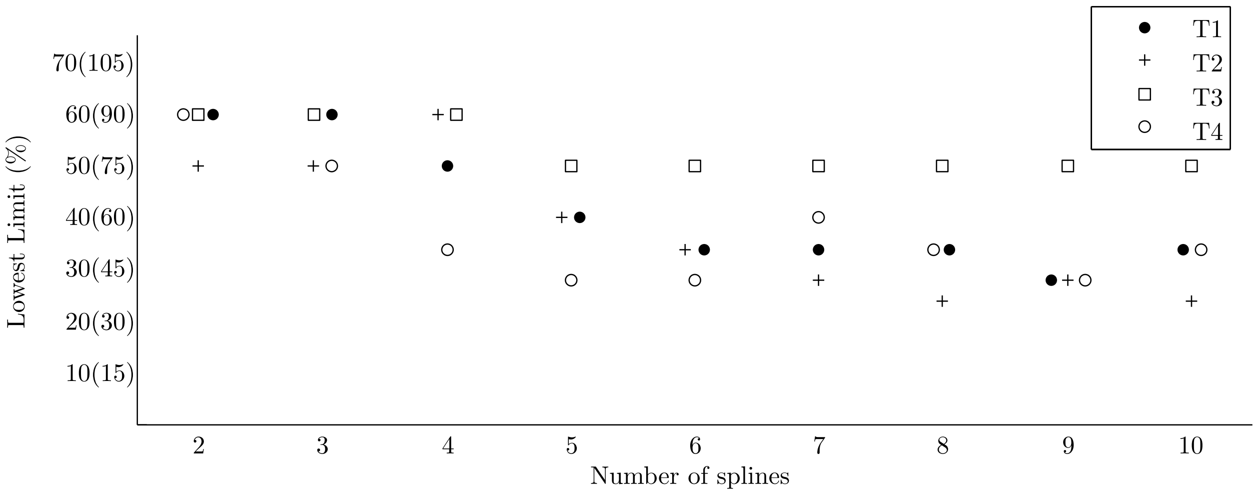


Fig. 6. Results of the second analysis. On the y-axis, the the lowest limit for which a successful trajectory was found is expressed as percentages of the static efforts.

4. CONCLUSION

In this work, a trajectory optimisation method was proposed to increase the load-carrying capacity of manipulators. It was shown that the payload capacity of manipulators can be considerably increased through trajectory optimisation on a task basis. It should be noted that the proposed method does not produce time optimal solutions such as those obtained with Bang-bang methods. However, it was shown that for some tasks, the trajectories obtained can be very close to the optimal Bang-bang solution. Since the primary goal of this research was to enable manipulators to perform tasks with heavier loads than their carrying capacity, the proposed method is considered successful. It was also shown that the solutions found are not necessarily globally optimal. In fact, the initial guess has a great impact on the convergence of the method.

The proposed method is very flexible and can be adapted to many trajectory optimisation problems. For example, the total energy could be optimised and velocity, acceleration, or jerk constraints could readily be added. The dynamic constraints studied in this paper were static, *i.e.*, the values for the maximum and minimum effort limits were constant. However, the limits of electric actuators are often tied to the velocity which could be taken into account with the proposed method. This flexibility is not present in other optimal control methods such as Bang-bang methods.

ACKNOWLEDGMENT

The authors would like to thank the Natural Sciences and Engineering Research Council of Canada (NSERC) and the Canada Research Chair program for their financial support.

REFERENCES

1. Hwang, Y.K. and Ahuja, N. "Gross motion planning — a survey." *ACM Computing Surveys (CSUR)*, Vol. 24, No. 3, pp. 219–291, 1992.
2. Bobrow, J.E. "Optimal robot plant planning using the minimum-time criterion." *Robotics and Automation, IEEE Journal of*, Vol. 4, No. 4, pp. 443–450, 1988.
3. Slotine, J.J. and Yang, H.S. "Improving the efficiency of time-optimal path-following algorithms." *Robotics and Automation, IEEE Transactions on*, Vol. 5, No. 1, pp. 118–124, 1989.
4. Choset, H., Lynch, K.M., Hutchinson, S., Kantor, G., Burgard, W., Kavraki, L.E. and Thrun, S. *Principles of Robot Motion: Theory, Algorithms, and Implementations*. MIT press, 2005.
5. Kim, D.W., Choi, J.S. and Nnaji, B. "Robot arc welding operations planning with a rotating/tilting positioner." *International journal of production research*, Vol. 36, No. 4, pp. 957–979, 1998.
6. Kim, K.Y., Kim, D.W. and Nnaji, B.O. "Robot arc welding task sequencing using genetic algorithms." *IIE Transactions*, Vol. 34, No. 10, pp. 865–880, 2002.
7. Kirk, D.E. *Optimal control theory: an introduction*. Prentice-Hall, 1970.
8. Kaya, C.Y. and Noakes, J.L. "Computations and time-optimal controls." *Optimal control applications and methods*, Vol. 17, No. 3, pp. 171–185, 1996.
9. Lucas, S.K. and Kaya, C.Y. "Switching-time computation for bang-bang control laws." In "American Control Conference, 2001. Proceedings of the 2001," Vol. 1, pp. 176–181. IEEE, 2001.
10. Weinreb, A. and Bryson, A. "Optimal control of systems with hard control bounds." *Automatic Control, IEEE Transactions on*, Vol. 30, No. 11, pp. 1135–1138, 1985.
11. Sahar, G. and Hollerbach, J.M. "Planning of minimum-time trajectories for robot arms." *The International Journal of Robotics Research*, Vol. 5, No. 3, pp. 90–100, 1986.
12. Meier, E.B. and Ryson, A.E. "Efficient algorithm for time-optimal control of a two-link manipulator." *Journal of Guidance, Control, and Dynamics*, Vol. 13, No. 5, pp. 859–866, 1990.
13. Bertrand, R. and Epenoy, R. "New smoothing techniques for solving bang–bang optimal control problems—numerical results and statistical interpretation." *Optimal Control Applications and Methods*, Vol. 23, No. 4, pp. 171–197, 2002.
14. Jamhour, E. and André, P. "Planning smooth trajectories along parametric paths." *Mathematics and Computers in Simulation*, Vol. 41, No. 5, pp. 615–626, 1996.
15. Perumaal, S.S. and Jawahar, N. "Automated trajectory planner of industrial robot for pick-and-place task." *Int J Adv Robotic Sy*, Vol. 10, No. 100, 2013.
16. Carretero, J.A., Ebrahimi, I. and Boudreau, R. "Overall motion planning for kinematically redundant parallel manipulators." *Journal of Mechanisms and Robotics*, Vol. 4, No. 2, p. 024502, 2012.
17. Korayem, M. and Shokri, M. "Maximum dynamic load carrying capacity of a 6ups-stewart platform manipulator." *Scientia Iranica*, Vol. 15, No. 1, pp. 131–143, 2008.
18. Korayem, M., Haghighi, R., Nikoobin, A., Alamdari, A. and Korayem, A. "Determining maximum load carrying capacity of flexible link manipulators." *Scientia Iranica, Transactions B: Mechanical Engineering*, Vol. 16, No. 5, pp. 440–450, 2009.
19. Korayem, M. and Nikoobin, A. "Maximum payload path planning for redundant manipulator using indirect solution of optimal control problem." *The International Journal of Advanced Manufacturing Technology*, Vol. 44, No. 7-8, pp. 725–736, 2009.
20. Korayem, M.H., Haghighi, R., Korayem, A., Nikoobin, A. and Alamdari, A. "Determining maximum load carrying capacity of planar flexible-link robot: closed-loop approach." *Robotica*, Vol. 28, No. 07, pp. 959–973, 2010.
21. Mathworks. "Matlab® fmincon algorithms." [Online; accessed 29-January-2015], 2015. URL \url{http://www.mathworks.com/help/optim/ug/constrained-nonlinear-optimization-algorithms.html}