2016 American Control Conference (ACC)
Boston Marriott Copley Place
July 6-8, 2016. Boston, MA, USA

Path-Constrained Trajectory Planning for Robot Service Life Optimization

Chung-Yen Lin[1], Yu Zhao[1], Masayoshi Tomizuka[1], and Wenjie Chen[2]

***Abstract*— The return-on-investment (ROI) of robotization is defined as the difference between the money earned through robotization and the investment put on the robotic system. In industrial practice, the ROI is often prioritized over other factors. There are two ways to help increase the ROI, namely, 1) improving robot productivity and 2) maximizing the robot service life. In fact, faster robot motion (i.e., higher productivity as preferred) with larger joint torque would lead to a higher chance of hardware failure. Therefore, it is of interest to develop a method that can maximize the robot service life without sacrificing the productivity. This paper presents a trajectory re-planning algorithm for this purpose. The idea is to increase the time duration for some periods of motion to extend the reducer life, while reduce the time duration for some other periods of motion to shorten the cycle time. The performance of the proposed method is evaluated on the FANUC M16*i*B robot system. The results show that the method can improve the expected worst-case service life by approximately 10%. Computation issues of the proposed algorithm are discussed.**

I. INTRODUCTION

In most industrial robotic applications, the two major criteria, precision and productivity, are often emphasized as the ultimate objectives. When deploying the robots to the production lines, however, the return-on-investment (ROI) is actually one of the most important factors that the robot end-users need to consider. ROI is a comparison of the money earned through robotization to the investment on the robotic system itself. While improving robot performance (e.g., precision and productivity) can help increase the ROI, maximizing the robot service life is another major approach to achieve a larger return. In other words, these three criteria are the key components of the attractiveness of robotization. This paper will study how, by motion re-planning, the robot service life can be prolonged, without sacrificing the robot precision (i.e., not changing the motion path) or productivity (i.e., not increasing the cycle time).

Maximizing the robot service life means to enable the robot working in a reliable operation status without failures (especially hardware failures) for the longest time. The most commonly seen failure in current industrial robots is the reducer failure for robot joints, which limits the overall robot service life. The reducer life is usually determined by the joint speed and reducer torque. Therefore, prolonging the reducer life without constraints would generally increase the cycle time (as the motion acceleration/speed is reduced) and decrease the robot utility. However, given a set of pre-planned motions, it is possible that these motion are not up to the robot physical limits. Even for the same speed/acceleration, reducer loads for different postures would be quite different. That means that it is possible to make a balance between reducing the robot speed for one period of motion to increase the reducer life and increasing the robot speed for some other periods of motion to keep the cycle time unchanged. Based on this idea, this paper presents a path-constrained trajectory planning method for optimizing the robot service life. The performance of the proposed method is evaluated on the FANUC M16*i*B robot system.

*This work was supported by FANUC Corporation, Japan.

[1]Chung-Yen Lin, Yu Zhao, and Masayoshi Tomizuka are with Department of Mechanical Engineering, University of California, Berkeley, CA 94720, USA. Email: {chung_yen, yzhao334, tomizuka} at berkeley.edu

[2] Wenjie Chen is with FANUC Corporation, Oshino-mura, Yamanashi-ken, Japan. Email: wjchen at berkeley.edu

II. PROBLEM FORMULATION

The problem can be termed as an optimization problem to achieve the optimal practical trade-off. More specifically, the optimization problem can be described as follows. Given a set of continuous motions, re-plan the motions such that a) the robot service life (particularly the reducer service life) is maximized, while b) keeping the motion path unchanged, c) keeping the cycle time not increased, and d) not violating the other limitations such as motor speed limit, and reducer speed/torque limit. This section will focus on how to mathematically formulate the problem.

A. Reducer Service Life Calculation

As the goal of the proposed trajectory planner is to maximize the reducer service life, it is natural to set the cost function as an estimate of the reducer service life. In [1], it is shown that the expected reducer service life can be computed by:

$$L = \frac{\lambda}{|\dot{q}|_{\text{avg}}} \left(\frac{1}{|u|_{\text{avg}}} \right)^c \tag{1}$$

where c and λ are constants that may variate from one reducer model to another. $|u|_{\text{avg}}$ and $|\dot{q}|_{\text{avg}}$ are, respectively, the average torque and the average speed defined by:

$$|u|_{\text{avg}} = \left(\frac{\sum_{k=1}^{K} t_k |\dot{q}_k| |u_k|^c}{\sum_{k=1}^{K} t_k |\dot{q}_k|} \right)^{\frac{1}{c}}$$

$$|\dot{q}|_{\text{avg}} = \frac{\sum_{k=1}^{K} t_k |\dot{q}_k|}{\sum_{k=1}^{K} t_k}$$

where t_k is the time interval in the k-th segment of motion, K is the number of total segments, u_k is the motor torque, and $\dot{q}_k$ is the motor speed.

B. Path-Constrained Trajectory Planning

The problem of path-constrained trajectory planning is to change the robot motion speed and acceleration simultaneously while keeping the robot path (in either the task space or the joint space) unchanged [2]. Consider a motion trajectory $q(t)$ defined in the joint space. We can parametrize it by introducing a variable $s = s(t) \in [0, 1]$ such that $q = f_q(s)$, where the function f_q describes the mapping from the parametric space to the robot joint space. This mapping allows to perform trajectory planning along the path of q by re-scheduling the parameter $s(t)$. Since the proposed trajectory planner is formulated in the parametric space, the re-planned trajectory will automatically satisfy the path-fixed constraint. This technique has been used in [3,4,5] with applications to time-optimal trajectory planning and in [6] with an application to energy-optimal trajectory planning.

It is of interest to note that as the motion path is unchanged, it is unnecessary to explicitly introduce singularities avoidance methods [7] to the planner.

C. Non-increasing Cycling Time

To solve a planning problem with the cost function (1), it is natural to consider time interval t_k as a variable. This allows us to manipulate the trajectory by controlling the time intervals of each path segment. However, when using this formulation, the constraint of non-increasing cycling time needs to be described explicitly in the optimization problem as $\sum_{k=1}^{K} t_k \leq \bar{T}$, where $\bar{T}$ is an upper bound of the traveling time. An alternative is to partition the path into K segments with equal intervals Δt and to manipulate the trajectory by controlling the parametric variable s at each path segment. By doing this, equation (1) can be simplified as:

$$L = \frac{\lambda \sum_{k=1}^{K} \Delta t}{\sum_{k=1}^{K} \Delta t |\dot{q}_k|} \left(\frac{\sum_{k=1}^{K} \Delta t |\dot{q}_k|}{\sum_{k=1}^{K} \Delta t |\dot{q}_k| |u_k|^c} \right) = \frac{\lambda K}{\sum_{k=1}^{K} |\dot{q}_k| |u_k|^c}$$

It is important to note that, in the above formulation, the variables of time interval t_k are pruned off from the cost function. It implicitly means that the cycle time of all possible choices of trajectories with this expression would remain unchanged. More precisely, it automatically satisfies the constraint $\sum_{k=1}^{K} t_k = K\Delta t \leq \bar{T}$ as long as the designed time interval Δt satisfies $\Delta t \leq \frac{\bar{T}}{K}$.

D. Robot Model in Parametric Space

Similar to many other trajectory planning problems, the maximum service life trajectory cannot violate robot physical limits. A popular approach to deal with the robot constraints is to represent the whole robot dynamics and robot kinematic in terms of the decision variable s [8]. To be precise, we have:

$$q = f_q(s) \tag{2}$$
$$\dot{q} = \nabla f_q(s) \dot{s} \tag{3}$$
$$\ddot{q} = \nabla^2 f_q(s) \dot{s}^2 + \nabla f_q(s) \ddot{s} \tag{4}$$
$$\begin{aligned} u =& M(q)\ddot{q} + G(q) + C(q, \dot{q})\dot{q} + F(\dot{q}) \\ =& M\left(f_q(s)\right) \left(\nabla^2 f_q(s) \dot{s}^2 + \nabla f_q(s) \ddot{s} \right) + G(f_q(s)) \\ &+ C\left(f_q(s), \nabla f_q(s) \dot{s}\right) \nabla f_q(s) \dot{s} + F\left(\nabla f_q(s) \dot{s}\right) \end{aligned} \tag{5}$$

where M is the inertia matrix, C is the Coriolis and centrifugal force matrix, F is the friction force, and G is the gravity force. With this parametric space robot model, the constraints of not violating robot physical limits would simply serve as a set of inequality nonlinear constraints:

$$\begin{aligned} \underline{u} \leq u_k(s_k, \dot{s}_k, \ddot{s}_k) \leq \bar{u} \\ \underline{\dot{q}} \leq \dot{q}_k(s_k, \dot{s}_k) \leq \bar{\dot{q}} \end{aligned}$$

where $\{\underline{u}, \bar{u}\}$ and $\{\underline{\dot{q}}, \bar{\dot{q}}\}$ are motor torque limits and motor speed limits, respectively. This paper only considers the speed limits and the torque limits. An extension of considering acceleration limits can be done without additional difficulties.

III. OPTIMAL CONTROL BASED TRAJECTORY PLANNER

Although it is not unusual to consider a trajectory planning problem as an optimal control problem [9], a key contribution of this paper is to apply this idea to the parametric space for a path-constrained planning problem. Namely, we assume that there is a dynamic model f_{sc} and a continuous control input signal v_t in the parametric space governing the value of the parameter s in the following way:

$$\left[\dot{s}_t, \ddot{s}_t, \dddot{s}_t\right] = f_{sc}^{\mathrm{T}}\left(s_t, \dot{s}_t, \ddot{s}_t, v_t\right) \tag{6}$$

By discretizing the model with a sampling interval Δt, we have:

$$\left[s_{k+1}, \dot{s}_{k+1}, \ddot{s}_{k+1}\right] = f_{sd}^{\mathrm{T}}\left(s_k, \dot{s}_k, \ddot{s}_k, v_k; \Delta t\right) \tag{7}$$

With this discrete-time dynamics in the parametric space, the trajectory planning task becomes the problem of finding a sequence of control inputs $\{v_k\}$ that drive the parameter s from zero to one within K-steps. The overall problem is formulated as follows:

$$\begin{aligned} \max_{s_k, \dot{s}_k, \ddot{s}_k, v_k} \quad & \frac{\lambda K}{\sum_{k=1}^{K} |\dot{q}_k(s_k, \dot{s}_k)| |u_k(s_k, \dot{s}_k, \ddot{s}_k)|^c} \\ \text{subject to} \quad & \forall k : \underline{\dot{q}} \leq \dot{q}_k(s_k, \dot{s}_k) \leq \bar{\dot{q}} \\ & \underline{u} \leq u_k(s_k, \dot{s}_k, \ddot{s}_k) \leq \bar{u} \\ & \left[s_{k+1}, \dot{s}_{k+1}, \ddot{s}_{k+1}\right] = f_{sd}^{\mathrm{T}}\left(s_k, \dot{s}_k, \ddot{s}_k, v_k; \Delta t\right) \\ & s_1 = 0, s_K = 1 \end{aligned}$$

Note that this formulation implicitly restricts the parametric space variable s to be non-decreasing since any reverse action in the parametric space can only decrease the performance index.

The discrete-time dynamic model f_{sd} in the parametric space is a design variable. It describes the relationship between states $\{s, \dot{s}, \ddot{s}\}$ and input v at two consecutive time steps. A simple choice is to set it as a third-order kinematic model, which has the equation:

$$\begin{bmatrix} s_{k+1} \\ \dot{s}_{k+1} \\ \ddot{s}_{k+1} \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & \Delta t & \frac{\Delta t^2}{2!} \\ 0 & 1 & \Delta t \\ 0 & 0 & 1 \end{bmatrix}}_{A_s(\Delta t)} \underbrace{\begin{bmatrix} s_k \\ \dot{s}_k \\ \ddot{s}_k \end{bmatrix}}_{z_k} + \underbrace{\begin{bmatrix} \frac{\Delta t^3}{3!} \\ \frac{\Delta t^2}{2!} \\ \Delta t \end{bmatrix}}_{B_s(\Delta t)} v_k \tag{8}$$

For simplicity of notation, we define nonlinear function J_k and g_k as follows:

$$J_k(z_k, v_k) := \frac{1}{\lambda K} |\dot{q}_k(s_k, \dot{s}_k)||u_k(s_k, \dot{s}_k, \ddot{s}_k)|^c \tag{9}$$

$$g_k(z_k, v_k) := \begin{bmatrix} \dot{q}_k(s_k, \dot{s}_k) - \bar{\dot{q}} \\ -\dot{q}_k(s_k, \dot{s}_k) + \underline{\dot{q}} \\ u_k(s_k, \dot{s}_k, \ddot{s}_k) - \bar{u} \\ -u_k(s_k, \dot{s}_k, \ddot{s}_k) + \underline{u} \end{bmatrix} \tag{10}$$

Then, the original optimization problem becomes:

$$\begin{aligned} \max_{\mathbf{z},\mathbf{v}} \quad & \frac{1}{\sum_{k=1}^{K} J_k(z_k, v_k)} \\ \text{subject to} \quad & \forall k : z_{k+1} = A_s(\Delta t) z_k + B_s(\Delta t) v_k \\ & g_k(z_k, v_k) \le 0 \\ & s_1 = 0, s_K = 1 \end{aligned}$$

where the decision variables $\mathbf{z}$ and $\mathbf{v}$ are lifted vectors of $\{z_k\}$ and $\{v_k\}$, respectively. Now given an arbitrary parametric interval Δt, we use the above program to find a sequence of inputs that drive the parameter s from zero to one in the parametric space. The resulting trajectory corresponds to the reducer torques that maximize the robot service life.

This result can be extended to multi-joint robots. By considering the worst case service life among all robot joints, we have a max-min problem as follows:

$$\begin{aligned} \max_{\mathbf{z},\mathbf{v}} \quad & \min_{i \in \mathcal{I}} \frac{1}{\sum_{k=1}^{K} J_k^{(i)}(z_k, v_k)} \\ \text{subject to} \quad & \forall k : z_{k+1} = A_s(\Delta t) z_k + B_s(\Delta t) v_k \\ & \forall k, i : g_k^{(i)}(z_k, v_k) \le 0 \\ & s_1 = 0, s_K = 1 \end{aligned} \tag{11}$$

where $\mathcal{I} = \{1, 2, \ldots, n\}$ is the set of robot joints.

By solving the optimization problem (11), we have a discrete-time parametric-space trajectory described by z_k^* and v_k^*. The continuous-time trajectory can be reconstructed by:

$$\begin{aligned} s^*(t) &= s_k^* + \dot{s}_k^* \tau + \ddot{s}_k^* \frac{\tau^2}{2!} + v_k^* \frac{\tau^3}{3!} \\ \dot{s}^*(t) &= \dot{s}_k^* + \ddot{s}_k^* \tau + v_k^* \frac{\tau^2}{2!} \\ \ddot{s}^*(t) &= \ddot{s}_k^* + v_k^* \tau \end{aligned}$$

where $(k-1)\Delta t \le t \le k\Delta t$ and $\tau = (t - (k-1)\Delta t)$. Then, substituting the sequence of parameters $\{s^*(t), \dot{s}^*(t), \ddot{s}^*(t)\}$ into the robot dynamics (2)-(5) gives us the joint space trajectory and the corresponding motor torques.

IV. ALGORITHM

The program shown in (11) is not convex due to the non-convexities in the cost function, the robot dynamics, and the path parameterization. A possible solution to this problem is the approximate dynamic programming (ADP) [10]. A benefit of solving ADP is that it provides the explicit control policy for all initial condition s_1. However, it requires constructing a cost-to-go function in each step of ADP, and hence will cost a lot of computation and memory. As the initial condition s_1 is always zero in our application, there is no need to spend such computation/memory for finding the complete control policy.

An alternative solution is the sequential quadratic programming (SQP) [11,12]. The idea of SQP is to linearize the constraints with respect to an operation point $(\bar{\mathbf{z}}, \bar{\mathbf{v}})$, then solve an approximate convex program locally by fitting a quadratic program (QP) [12] around $(\bar{\mathbf{z}}, \bar{\mathbf{v}})$. This approximate QP can be solved directly using commercial optimization solvers. The solution will be used to update the operation point and the next iteration of SQP would be performed at the updated point.

Note that there exist some issues of applying the standard SQP algorithm to the problem. Namely, computing the Hessian matrix of the cost function (with a large number of parametric segments K) can be nontrivial. This paper therefore provides a better formulation so that the Hessian matrix in each iteration of SQP can be obtained efficiently. More precisely, we reformulate the problem so that the new cost function has a sparse Hessian matrix.

We first consider the single joint case. Note that since J is a positive function, the maximization problem can be reformulated as a minimization problem by taking the reciprocal of the cost function as follows:

$$\forall i : \ \arg\max_{\mathbf{z},\mathbf{v}} \frac{1}{\sum_{k=1}^{K} J_k^{(i)}(z_k, v_k)} = \arg\min_{\mathbf{z},\mathbf{v}} \sum_{k=1}^{K} J_k^{(i)}(z_k, v_k)$$

By extending this idea to multi-joint robot, we obtain the following min-max problem:

$$\begin{aligned} \min_{\mathbf{z},\mathbf{v}} \quad & \max_{i \in \mathcal{I}} \sum_{k=1}^{K} J_k^{(i)}(z_k, v_k) \\ \text{subject to} \quad & \forall k : z_{k+1} = A_s(\Delta t) z_k + B_s(\Delta t) v_k \\ & \forall k, i : g_k^{(i)}(z_k, v_k) \le 0 \\ & s_1 = 0, s_K = 1 \end{aligned}$$

Taking the Lagrangian of the above problem (without considering the linear constraints) yields:

$$\begin{aligned} \mathcal{L}(\mathbf{z}, \mathbf{v}, \mu) = & \left(\max_{i \in \mathcal{I}} \sum_{k=1}^{K} J_k^{(i)}(z_k, v_k) \right) \\ & + \sum_{i \in \mathcal{I}} \sum_{k=1}^{K} (\mu_k^{(i)})^{\mathrm{T}} g_k^{(i)}(z_k, v_k) \end{aligned}$$

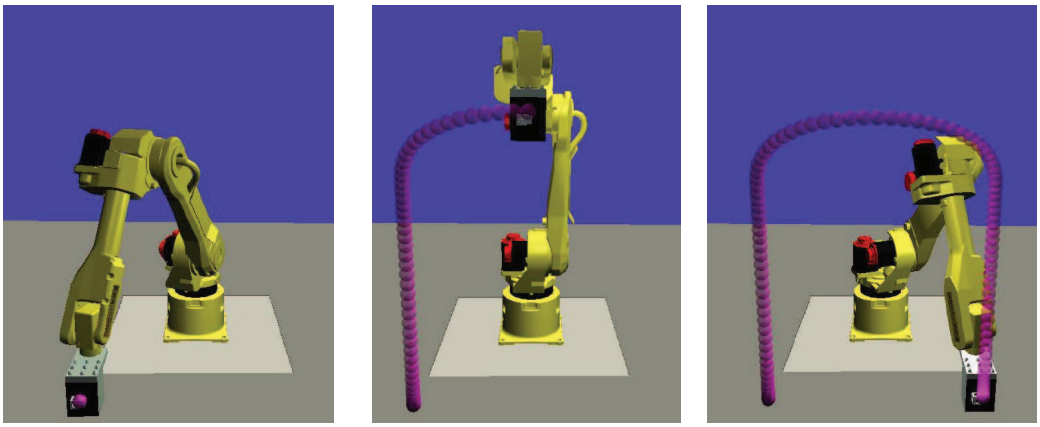

Fig. 1. A path used to test the life-maximum trajectory planning algorithm. The path is described on the X-Z plane while the end-effector orientation is fixed along the entire path.

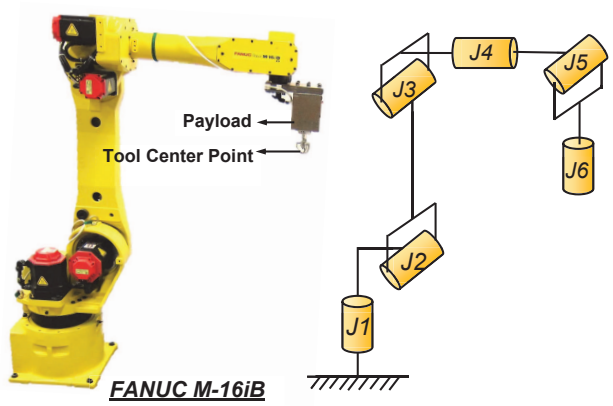

Fig. 2. Configuration of FANUC M-16*i*B robot system

where μ is the Lagrange multiplier. Then given an operation point $(\bar{\mathbf{z}}, \bar{\mathbf{v}})$, the SQP subproblem can be obtained by:

$$\begin{aligned} \min_{\mathbf{z},\mathbf{v}} \quad & \nabla\mathcal{L}(\bar{\mathbf{z}},\bar{\mathbf{v}},\bar{\mu}) \begin{bmatrix}\tilde{\mathbf{z}}\\ \tilde{\mathbf{v}}\end{bmatrix} + \frac{1}{2}\begin{bmatrix}\tilde{\mathbf{z}}\\ \tilde{\mathbf{v}}\end{bmatrix}^{\mathrm{T}} \nabla^2\mathcal{L}(\bar{\mathbf{z}},\bar{\mathbf{v}},\bar{\mu}) \begin{bmatrix}\tilde{\mathbf{z}}\\ \tilde{\mathbf{v}}\end{bmatrix} \\ \text{subject to} \quad & \forall k: z_{k+1} = A_s(\Delta t) z_k + B_s(\Delta t) v_k \\ & \forall k, i: g_k^{(i)}(\bar{z}_k, \bar{v}_k) + \nabla g_k^{(i)}(\bar{z}_k, \bar{v}_k) \begin{bmatrix}\tilde{z}_k\\ \tilde{v}_k\end{bmatrix} \le 0 \\ & s_1 = 0, s_K = 1 \end{aligned}$$

where $\tilde{\mathbf{z}} := \mathbf{z} - \bar{\mathbf{z}}$ and $\tilde{\mathbf{v}} := \mathbf{v} - \bar{\mathbf{v}}$.

Note that as the values of $(\bar{\mathbf{z}}, \bar{\mathbf{v}})$ have been assigned in each SQP iteration, we can eliminate the maximization term in the Lagrangian by pre-computing the following quantity at the beginning of each SQP step:

$$\bar{i} = \arg\max_{i\in\mathcal{I}} \sum_{k=1}^{K} J_k^{(i)}(\bar{z}_k, \bar{v}_k) \tag{12}$$

This results in a modified Lagrangian function:

$$\begin{aligned} \mathcal{L}(\bar{\mathbf{z}},\bar{\mathbf{v}},\bar{\mu}) &= \sum_{k=1}^{K} J_k^{(\bar{i})}(\bar{z}_k,\bar{v}_k) + \sum_{i\in\mathcal{I}}\sum_{k=1}^{K} (\bar{\mu}_k^{(i)})^{\mathrm{T}} g_k^{(i)}(\bar{z}_k,\bar{v}_k) \\ &= \sum_{k=1}^{K} \underbrace{\left(J_k^{(\bar{i})}(\bar{z}_k,\bar{v}_k) + \sum_{i\in\mathcal{I}} (\bar{\mu}_k^{(i)})^{\mathrm{T}} g_k^{(i)}(\bar{z}_k,\bar{v}_k) \right)}_{\mathcal{L}_k(\bar{z}_k,\bar{v}_k,\bar{\mu})} \end{aligned}$$

where $\mathcal{L}_k$ is a one-step Lagrangian that is a function of $(\bar{z}_k, \bar{v}_k)$. Since the original Lagrangian $\mathcal{L}$ has been decoupled into K one-step Lagrangian functions, we are able to take advantage of the sparsity in Hessian matrix [13] to get the modified SQP subproblem:

$$\begin{aligned} \min_{\mathbf{z},\mathbf{v}} \quad & \sum_{k=1}^{K} \left(\nabla\mathcal{L}_k \begin{bmatrix}\tilde{z}_k\\ \tilde{v}_k\end{bmatrix} + \frac{1}{2}\begin{bmatrix}\tilde{z}_k\\ \tilde{v}_k\end{bmatrix}^{\mathrm{T}} \nabla^2\mathcal{L}_k \begin{bmatrix}\tilde{z}_k\\ \tilde{v}_k\end{bmatrix} \right) \quad (13) \\ \text{subject to} \quad & \forall k: z_{k+1} = A_s(\Delta t) z_k + B_s(\Delta t) v_k \\ & \forall k, i: g_k^{(i)}(\bar{z}_k, \bar{v}_k) + \nabla g_k^{(i)}(\bar{z}_k, \bar{v}_k) \begin{bmatrix}\tilde{z}_k\\ \tilde{v}_k\end{bmatrix} \le 0 \\ & s_1 = 0, s_K = 1 \end{aligned}$$

This allows us to solve the original problem using K number of 4×4 Hessian matrices rather than using a $4K \times 4K$ Hessian matrix.

V. CASE STUDIES

The proposed trajectory planning algorithm is applied to maximize the service life of the FANUC M-16*i*B robot [14]. The FANUC M-16*i*B robot is a medium-size robot used for high-speed applications such as laser cutting and material handling. It has six joints driven by indirect drive mechanisms. Each joint is equipped with a built-in motor encoder. The sampling rates of all the sensors and the robot controller are set to be 1kHz. A payload with a weight of 18.37kg is mounted at the last joint as the end-effector to mimic the real-world working conditions. The configuration of the FANUC M-16*i*B robot system is shown in Fig. 2.

TABLE I

SPECIFICATION OF THE FANUC M-16iB ROBOT

	J_1	J_2	J_3	J_4	J_5	J_6
Maximum Torque (Nm)	1396.5	1402.3	382.7	45.2	44.6	32.5
Maximum Motor Speed (rpm)	4000	4000	5000	5000	5000	5000
Maximum Reducer Speed (rpm)	40	50	60	–	–	–
Constant c in Service Life Calculation (1)	10/3	10/3	10/3	–	–	–
Constant λ in Service Life Calculation (1)	8.811×10^{13}	1.211×10^{14}	1.127×10^{12}	–	–	–

The specifications of the reducer models and robot physical limits of the FANUC M-16iB robot are listed in Table I. Note that the joints J_4, J_5, and J_6 are equipped with customized gear boxes or harmonic drives (instead of typical RV reducers). Therefore, we did not incorporate the service life costs of the last three joints into the trajectory planning algorithm. The motor speed limits and the motor torque limits for these joints, however, are still considered in the optimization problem.

The robot tool center point (TCP) path used to test the proposed life-maximum trajectory planner is illustrated in Fig. 1. The initial trajectory of the planner is set to have constant speed in the parametric space. Namely, the velocity terms $\{\dot{s}_k\}$ are equal to $\frac{1}{(K-1)\Delta t}$, while the position terms $\{s_k\}$ are such that the difference between the consecutive variables is $\frac{1}{K-1}$. This initialization is feasible since it behaves exactly like the original trajectory. An alternative way to initialize the optimization is to start with a random point and solve a feasibility problem [15]. The feasibility problem has at least one solution which is the original trajectory.

Fig. 3 and Fig. 4 show, respectively, the TCP trajectories and the joint space trajectories obtained from the initial setting and the proposed trajectory planning method. Since the estimated reducer life is proportional to the torque to the power of $\frac{10}{3}$, it is expected that the trajectory planner would prefer to minimize the amplitude of the motor torques whenever the motor speed is nonzero. For example, due to the gravity effect, the torques required for driving the robot moving upward would be larger than the torques required for driving the robot moving downward. Therefore, the trajectory planner will slow down the first part of the trajectory and speed up the last part of the trajectory. To clearly illustrate the effectiveness of the proposed method, a point-to-point mapping from the initial trajectory to the life-maximum trajectory is visualized in Fig. 5.

Fig. 6 and Fig. 7 show, respectively, the instantaneous costs $J_k^{(i)}$ and the cumulative costs $\sum_k J_k^{(i)}$ for the first three joints. It is seen that the joint J_3 has the largest costs (i.e., the shortest expected life) in the initial trajectory. Therefore, the planner would prefer to minimize the cost for the joint J_3 because it plays the important role of determining the reducer service life.

Table II compares the reducer service lives between the initial trajectory and the life-maximum trajectory. As expected, the proposed trajectory planner improved the worst-case service life (which happens on the joint J_3) by 10.5%.

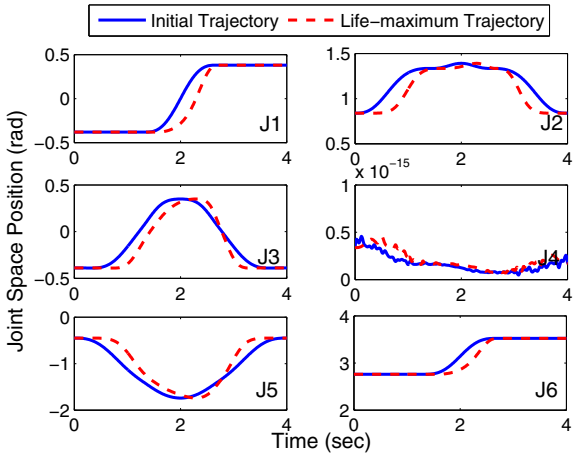

Fig. 3. Joint space trajectory references of the initial trajectory (blue solid line) and the life-maximum trajectory (red dot line)

TABLE II

COMPARISON OF EXPECTED REDUCER SERVICE LIVES (UNIT: KHOURS)

	J_1	J_2	J_3
Initial Trajectory	1716.4	203.4	22.9
Life-maximum Trajectory	658.7	126.6	25.3

However, the service lives for the joints J_1 and J_2 are degraded as a trade-off. Note that since the optimizer tends to improve the service life only for the joint J_3 (the worst one), it will not take into account the compromise made by the joints J_1 and J_2. It may not be a good design when the average life of all robot joints is an important issue. In this case, it is possible to replace the cost function in (11) with the following one to address the problem:

$$\max_{\mathbf{z},\mathbf{v}} \quad \sum_{i\in\mathcal{I}} \left(\sum_{k=1}^{K} J_k^{(i)}(z_k, v_k) \right)^{-1}$$

Table III compares the CPU times of solving the program (11) with the standard SQP and the SQP with modified subproblem (13). It is seen that the modified program leads to substantial savings in the computation by taking advantage of the sparsity in the Hessian matrix. More precisely, the computational time of proposed formulation increases linearly with the number of partition points K. This result follows from the fact that the Hessian matrix in the modified program can be computed by K small sized Hessian matrices.

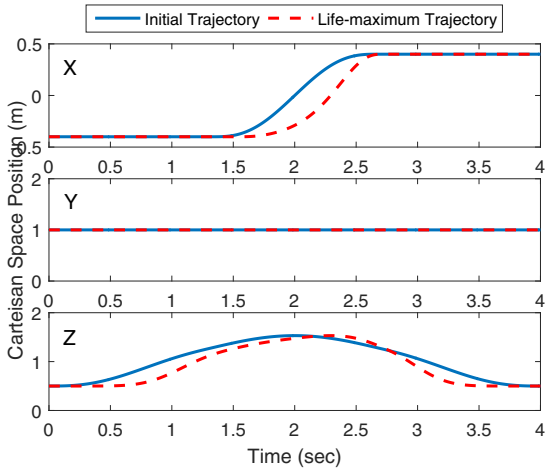

Fig. 4. TCP Cartesian space position references of the initial trajectory (blue solid line) and the life-maximum trajectory (red dot line)

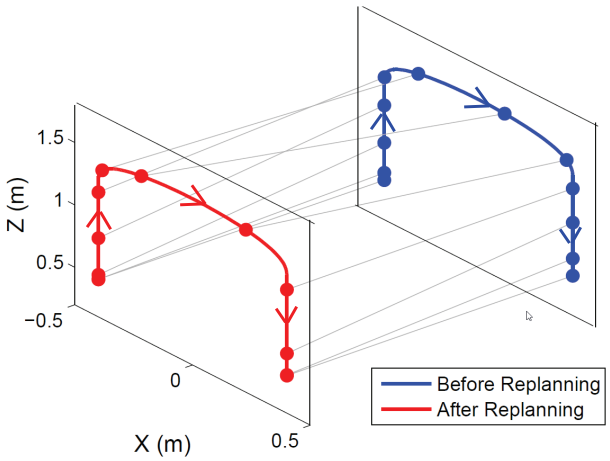

Fig. 5. A point-to-point comparison of TCP Cartesian space trajectory references

TABLE III

A QUANTITIVE COMPARISON OF DIFFERENT SQP FORMULATIONS

(UNIT: SEC)

	Standard SQP	SQP with (13)
$K = 10$	42.1	9.5
$K = 20$	184.0	21.2
$K = 30$	404.2	27.7
$K = 50$	1185.6	51.3
$K = 100$	3615.2	103.1

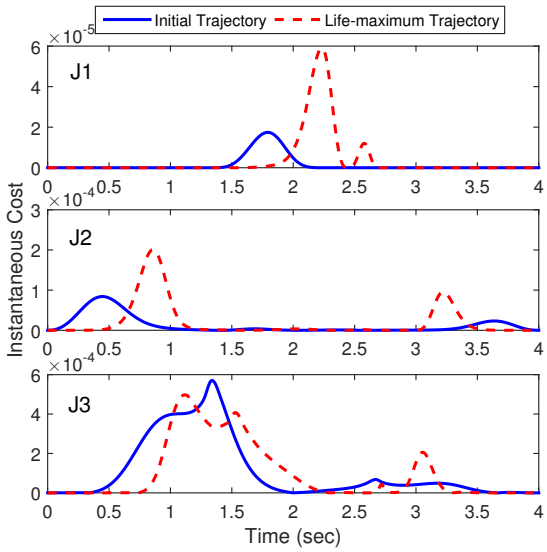

Fig. 6. Instantaneous costs of the initial trajectory (blue solid line) and the life-maximum trajectory (red dot line)

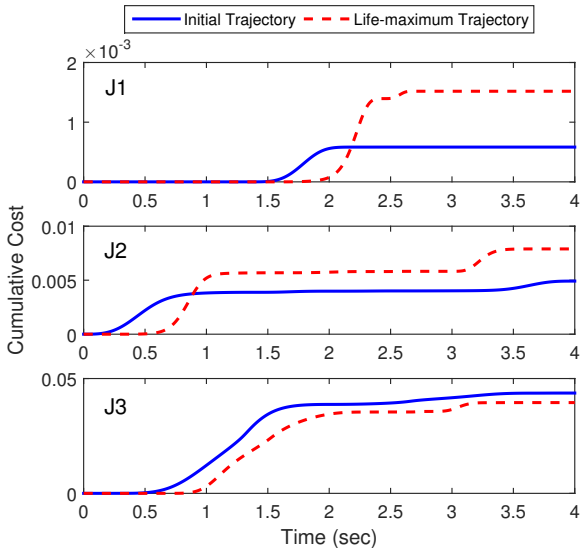

Fig. 7. Cumulative costs of the initial trajectory (blue solid line) and the life-maximum trajectory (red dot line)

VI. CONCLUSIONS

This paper presented a path-constrained trajectory planning method for optimizing the robot service life. The trajectory planner was formulated as an optimization problem and was solved by a modified version of SQP. The complexity of the proposed optimization solver is linear with respect to the trajectory length. Experimental results on FANUC M16*i*B robot showed that the proposed method can improve the expected worst-case service life by approximately 10%.

REFERENCES

[1] *(Feb. 2015) Nabtesco RV-C series.* [Online]. Available: http://www.nabtescomotioncontrol.com/pdfs/RV-Cseries.pdf

[2] D. Costantinescu and E. 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.

[3] Z. Shiller and S. Dubowsky, "On the optimal control of robotic manipulators with actuator and end-effector constraints," in *Robotics and Automation. Proceedings. 1985 IEEE International Conference on*, vol. 2, Mar 1985, pp. 614–620.

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

[5] D. Verscheure, B. Demeulenaere, J. Swevers, J. De Schutter, and M. Diehl, "Time-optimal path tracking for robots: A convex optimization approach," *Automatic Control, IEEE Transactions on*, vol. 54, no. 10, pp. 2318–2327, Oct 2009.

[6] K. Paes, W. Dewulf, K. V. Elst, K. Kellens, and P. Slaets, "Energy efficient trajectories for an industrial ABB robot," in *the 21st CIRP Conference on Life Cycle Engineering*, 2014, pp. 105 – 110.

[7] T. Yoshikawa, "Analysis and control of robot manipulators with redundancy," *Robotics research: the first international symposium*, pp. 735–747, 1984.

[8] S. M. LaValle, *Planning Algorithms*. New York, NY, USA: Cambridge University Press, 2006.

[9] S. J. Anderson, S. C. Peters, T. E. Pilutti, and K. Iagnemma, "An optimal-control-based framework for trajectory planning, threat assessment, and semi-autonomous control of passenger vehicles in hazard avoidance scenarios," *International Journal of Vehicle Autonomous Systems*, vol. 8, no. 2-4, pp. 190–216, 2010.

[10] D. P. Bertsekas, *Dynamic Programming and Optimal Control*, 2nd ed. Athena Scientific, 2000.

[11] P. T. Boggs and J. W. Tolle, "Sequential quadratic programming," *Acta numerica*, vol. 4, pp. 1–51, 1995.

[12] J. Nocedal and S. Wright, *Numerical Optimization*, ser. Springer Series in Operations Research and Financial Engineering. Springer New York, 2006.

[13] J. Betts, *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*, 2nd ed. Society for Industrial and Applied Mathematics, 2010.

[14] *(Feb. 2015) FANUC M-16iB/T series*. [Online]. Available: http://www.autocells.com/usr/docs/FANUCPDFDatasheets/FANUCM-16iBT.pdf

[15] P. Spellucci, "A new technique for inconsistent qp problems in the sqp method," *Mathematical Methods of Operations Research*, vol. 47, no. 3, pp. 355–400, 1998.