Preprints of the 19th World Congress
The International Federation of Automatic Control
Cape Town, South Africa. August 24-29, 2014

Joint Selection Criterion for Optimal Trajectory Planning for Robotic Manipulators using Dynamic Programming

Zeeshan Shareef * **Ansgar Trächtler** *

* *Heinz Nixdorf Institute, University of Paderborn, Fürstenallee 11, 33102 Paderborn, Germany (e-mail: zeeshan, ansgar.traechtler@hni.upb.de).*

Abstract: This paper presents a joint selection criterion for optimal trajectory planning using dynamic programming along a specified geometric path subject to torque and velocity constraints. A single non-stationary joint is normally considered for optimization purpose. In case of more than one non-stationary joints, the optimization by selecting any joint at random may end up with a non-optimal solution. This problem raises the importance of the proposed joint selection criterion. A novel joint selection criterion based on the geometric path information is presented for dealing with such cases having more than one non-stationary joints. In contrast of existing approaches to solve the optimization problem by considering each joint separately, the proposed criterion saves computational time, efforts and thus there is no need to solve the optimization problem multiple times with respect to each non-stationary joint to get the optimal solution. The proposed criterion can be used to optimize a given path with respect to travelling time, energy consumption or any other arbitrary form of cost function. The validity of the proposed selection criterion is tested on a parallel DELTA robot. The obtained results are compared with the optimal results obtained by other techniques which confirm the applicability of the proposed scheme.

Keywords: Dynamic programming, optimal trajectory, path planning, criterion functions, multiobjective optimisations, robotic manipulators, industrial robots

1. INTRODUCTION

Optimal trajectory planning in robotics has recently gained a lot of attention because of its extensive use not only in industrial applications but also in daily life. An optimal motion of an industrial robot is the key to success because it can help to increase the production rate and to reduce the production cost and energy consumption. The optimization in case of robotics become more difficult because of the non-linearities and coupling in the robot dynamics.

For tractability, the optimal trajectory of robotic manipulators is divided into three stages. The first stage is path planning. Extensive amount of work has been done on path planning, Latombe [1991]. The important problems addressed in path planning include collision avoidance and getting the smooth path. The second stage is trajectory planning from the given information of geometric path. In second stage, the position and the velocity of each joint is calculated as a function of time within the constraints on the joints torque and angular velocity. The third stage is trajectory tracking. This stage is responsible to track the robot's reference position and velocity that is calculated in the second stage. In this paper the main focus is on the trajectory planning under constraints.

Bobrow et al. [1985] and Shin et al. [1985] separately developed similar methods to calculate the optimal trajectory for robotic manipulators for a given specified path. Their method uses a phase plane plot to get the optimal solution in which dimensionality of the problem is reduced by converting dynamical equations to a set of second order differential equations using path parameter, defined as s from here on. The phase plane plot of $(s, \dot{s})$, sometimes referred as velocity limit curve (VLC) or the switching curve. In their method the bounds on the actuator torques were transformed to the acceleration bounds along the path. It is an indirect method and was later refined by Rajan [1985], Pfeiffer et al. [1987], Slotine et al. [1989], Shiller et al. [1992], Tarkiainen et al. [1993]. The optimal solution obtained by their approach is basically a bang-bang control as one of the actuator is always saturated. It is not only applicable for rigid manipulators, but also it can generate the time optimal solution for the cable based manipulator, as discussed by Behzadipour et al. [2006]. However, there are some drawbacks associated with this method. First, it can only solve the time minimization problem. The second problem is that the joint torques can be changed instantaneously which is impractical for real applications.

Shiller et al. [1991] proposed a solution approach for computing the time optimal trajectory for robotic manipulators in presence of obstacles. In the first step of their suggested method a near optimal path is selected using a branch bound search and a series of lower bound estimates. Finally, the global optimal solution is obtained by local

path optimization. They obtained obstacle avoidance by adding a penalty function to the motion time in local optimization.

A new approach to solve the time optimal trajectory planning is proposed by Verscheure et al. [2008]. In this approach the time optimal trajectory is calculated using convex optimization. The time optimal trajectory problem is transformed to the convex optimization through a nonlinear change of variables. Second order cone program using robust numerical algorithms are used to calculate the global optimal solution of time optimal problem. The key aspects of this method are the ease of implementation and the flexibility. This method is not only limited to time optimal trajectory planning, but also it can optimize any arbitrary cost function.

Dynamic programming is a very useful optimization technique which can solve the problem of optimization for highly nonlinear, complex systems under strong constraints(Bellman [1957]). For the first time, Vukobratovic et al. [1982] proposed the dynamic programming algorithm to find the energy optimal trajectory for robotic manipulators for a given path. Shin et al. [1986] proposed the dynamic programming algorithm to solve the optimal trajectory planning in terms of path parameter s. First the dynamics of the robots are converted to a second order differential equation using the path parameter. The optimal pseudo-velocity $\dot{s}$ is calculated using dynamic programming (DP) to get optimal solution of the given cost function under constraints. The main advantage of this method is the reduction in dimensionality as the dynamics of the robots are expressed in terms of the path parameter s.

Singh et al. [1987] proposed another dynamic programming algorithm for the optimal trajectory planning. Instead of reducing the dimensionality of the problem by path parameter (s), the optimization problem was solved by considering a single non-stationary joint on the robot. By dynamic programming, this problem is reduced to search over the velocity of one moving manipulator link. The constraints on the other joints, either stationary or non-stationary, are taken into account inside the dynamic programming algorithm loop. This algorithm does not propose the joint selection criterion in case of more than one non-stationary joints to get the global optimal solution.

An inverse dynamic-based dynamic programming (DP) approach for optimal trajectory planning of robotic manipulators is discussed by Yen [1995]. This inverse dynamic-based DP offers some advantages over conventional DP approach, i.e., it eliminates the interpolation requirement, and the requirement of integration of motion equations is also avoided. Field et al. [1996] proposed an iterative dynamic programming approach to minimum energy trajectory planning. In this modified dynamic programming approach a series of dynamic programming passes over a small reconfigurable grid size. This approach has advantages of avoiding poor local minima and curse of dimensionality, and providing parallel structure to reduce computational time. Beside these modification in DP algorithm for optimal trajectory planning, the criterion for joint selection in the DP algorithm proposed by Singh et al. [1987] was not defined which many lead to the local convergence.

In this paper, we present heuristic based joint selection criteria, for the case of more than one non-stationary joints, to implement the dynamic programming algorithm proposed by Singh et al. [1987]. In contrast of existing approach to solve the optimization problem by considering each non-stationary joint individually, the proposed criterion saves time and the need to solve the optimization problem multiple times with respect to each non-stationary joint to get the global optimal solution. Joint selection can be made easily from the predefined path information either given in Joints space or in Cartesian space. The remaining paper is outlined as follows. Section 2 gives detail problem description. In Section 3, the dynamic programming algorithm proposed by Singh et al. [1987] is discussed and the importance of the proposed criterion is explained. The joint selection criterion is discussed in Section 4. Section 5 presents numerical examples based on optimal trajectory planning of a parallel DELTA robot and the optimization results from different techniques are compared in details. Section 6 gives the conclusion drawn from our results.

2. PROBLEM FORMULATION FOR OPTIMAL TRAJECTORY PLANNING

The robot dynamics having n degree of freedom (DOF) motion with joint angles $\mathbf{q} = (q_1, q_2, \cdots, q_n) \in \mathbb{R}^n$, can be expressed in the terms of the applied joint torques $\boldsymbol{\tau} = (\tau_1, \tau_2, \cdots, \tau_n)^T \in \mathbb{R}^n$ in (1).

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

In (1), $\mathbf{M}(\mathbf{q}) \in \mathbb{R}^{n\times n}$ is positive definite mass matrix, $\mathbf{C}(\mathbf{q}, \dot{\mathbf{q}}) \in \mathbb{R}^{n\times n}$ is Centrifugal and Coriolis coefficient matrix, $\mathbf{G}(\mathbf{q})$ is gravitational force vector and $\boldsymbol{\tau}$ is the set of realizable torques that can be expressed in terms of joint velocity and joint position.

$$\boldsymbol{\tau} = (\tau_1, \tau_2, \tau_3, \cdots, \tau_n)^T \tag{2}$$

In the optimal trajectory planning the path is predefined either in Joint space ($\mathbf{q}$) or in Cartesian Space ($\mathbf{p}$). The path given in the Cartesian space ($\mathbf{p}$) can be converted to the Joint space using the inverse kinematics.

$$\mathbf{q} = \Psi(\mathbf{p}) \tag{3}$$

The initial and the final conditions for the dynamic programming can be found easily from the given path information. These terminal conditions are shown in (4) and (5). From the given initial and final condition on the angular joints, it is clear that the manipulator must be stationary at the start and end points and the velocities must be zero. These are the equality constraints or the terminal conditions for the optimization problem.

$$\mathbf{p}(t_0) = \mathbf{p}_0, \quad \mathbf{p}(t_f) = \mathbf{p}_f \tag{4}$$

$$\dot{\mathbf{p}}(t_0) = 0, \quad \dot{\mathbf{p}}(t_f) = 0 \tag{5}$$

In optimal trajectory planning, the main objective is to optimize the cost functions within the given constraints. The most important constraints for robotic manipulators trajectory planning describe the limitation on actuator force/torque. Some additional constraints can also be imposed on the joint velocities as well. The constraints on the actuator force/torque and the joint velocity are given in (7) and (6) respectively. The upper and lower limits on the actuator's joint velocity and torques can be find from the actuator's data sheet.

$$\dot{q}_{i,min} \leq \dot{q}_i \leq \dot{q}_{i,max} \quad i = 1, 2, \cdots, n \tag{6}$$

$$\tau_{i,min} \leq \tau_i \leq \tau_{i,max} \quad i = 1, 2, \cdots, n \tag{7}$$

Additional constraints can also be imposed like on joint acceleration. In any optimization, the most important term is the cost function J that has to be optimized. Normally, the most commonly used objective function is time-optimality. This time-optimality objective function can be combined with other criteria such as energy minimization, fuel minimization or any other arbitrary cost function. This is the flexibility of dynamic programming that more general objective functions can be defined. A generalized cost function is shown in (8). In (8), κ is the weight factor. If $\kappa = 0$, then (8) will be an energy minimization problem and, if $\kappa = 1$, then this cost function will present a time minimization problem.

$$J = \int_{t_0}^{t_f} (\kappa + (1 - \kappa)u^2) dt \tag{8}$$

In cost function the final time (t_f) can be fixed or it can be free depending upon the problem. For example, the final time cannot be fixed in case the time optimization problem is considered. The structure and the dynamics of the robot manipulator are similar to the general optimal trajectory problem which consist of the systems dynamics, terminal conditions and the inequality constraints.

3. OPTIMAL TRAJECTORY PLANNING USING DYNAMIC PROGRAMMING

As discussed earlier, dynamic programming is a very useful tool to solve the optimization problem under strong constraints for any arbitrary objective function. An optimal trajectory planning problem can be easily solved using dynamic programming. The first step of dynamic programming is to discretize the given problem. The discretization of the problem is necessary to solve this problem using digital computers. Let suppose, the given path in Cartesian space ($\mathbf{p}$) or in Joint Space ($\mathbf{q}$) is discretized in N_p segments. The continuous dynamical equations and the terminal conditions, discussed in Section 2, are given in (9)-(12) in discretized form.

$$\begin{aligned} \tau(k) = \mathbf{M}(\mathbf{q}(k))(\dot{\mathbf{q}}(k+1) - \dot{\mathbf{q}}(k)) \\ +\mathbf{C}(\mathbf{q}(k), \dot{\mathbf{q}}(k))(\mathbf{q}(k+1) - \mathbf{q}(k)) + \mathbf{G}(\mathbf{q}(k)), \\ k = 1, 2, \cdots, N_p \end{aligned} \tag{9}$$

$$\mathbf{q}(k) = \Psi(\mathbf{p}(k)), \quad k = 1, 2, \cdots, N_p \tag{10}$$

$$\mathbf{p}(0) = \mathbf{p}_0, \quad \mathbf{p}(N_p) = \mathbf{p}_f \tag{11}$$

$$\dot{\mathbf{p}}(0) = 0, \quad \dot{\mathbf{p}}(N_p) = 0 \tag{12}$$

Normally, to get the optimal trajectory of robotic manipulator for a given path using dynamic programming, a search is implied over position ($\mathbf{q}$) and velocity ($\dot{\mathbf{q}}$) of each joint. In case of n-DOF robotic manipulator, the dynamic programming algorithm has to search over $2n$ variables. As the information of the path is already given as a priori information, the dimensionality of the problem is reduced to search over a single variable, i.e. joint velocity. The allowable range of joint velocity is discretized into N_v segments. So, the algorithm will search over the all values of joint velocity ($\dot{\mathbf{q}}(j), j = 1, 2, \ldots, N_v$) to optimize the cost function for given path.

$$\begin{aligned} \dot{\mathbf{q}}_{min} &= \dot{\mathbf{q}}(1) \\ \dot{\mathbf{q}}_{max} &= \dot{\mathbf{q}}(N_v) \end{aligned} \tag{13}$$

To proceed for the optimal solution, first a non-stationary joint is selected as a reference. In this script, i^* is used the index of reference non-stationary joint i.e. q_{i^*} and k as the index of discrete points along the given path that is discretized into N_p segments, i.e. $k = 1, 2, \ldots, N_p$.

Consider the path movement from point k to $k+1$ and we want to optimize this segment. Assuming that the discretization of given path is very fine (i.e. N_p is large), the travelled distance will becomes small and acceleration, $\mathbf{M}(\mathbf{q}), \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})$ and $\mathbf{G}(\mathbf{q})$ do not change significantly over a single interval. For a possible velocity ($\dot{q}_{i^*}(j_k)$) at point k and an admissible velocity ($\dot{q}_{i^*}(j_{k+1})$) at point $(k+1)$, the joint acceleration at point k can be calculated using (14).

$$\ddot{q}_{i^*}(k) = \frac{[\dot{q}_{i^*}(j_{k+1})]^2 - [\dot{q}_{i^*}(j_k)]^2}{2[q_{i^*}(k+1) - q_{i^*}(k)]} \tag{14}$$

The time required to travel from point k to $k+1$ with joint velocity $\dot{q}_{i^*}(j_{k+1})$ is given by (15).

$$\Delta t(k) = \frac{2[q_{i^*}(k+1) - q_{i*}(k)]}{[\dot{q}_{i^*}(j_{k+1}) + \dot{q}_{i^*}(j_k)]} \tag{15}$$

All other joints must also cover the distance from point k to $k+1$ in the same time. The velocities of other joints, to cover the distance in the same time interval, is calculated using (16).

$$\dot{q}_m(j_k) = \frac{2[q_m(k+1) - q_m(k)]}{\Delta t(k)} - \dot{q}_m(j_{k+1}) \tag{16}$$

If the velocity of any joint $\dot{q}_m(j_k), [m = 1, 2, \cdots, n,\ m \neq i^*]$ does not lie in the joint velocity interval, the velocity of the reference non-stationary joint ($\dot{q}_{i^*}(j_k)$) is considered to be inadmissible and not considered for further calculations. If all the joints satisfy the velocity constraints, the accelerations for all other joints are calculated using (14). Once we have the values of displacement, velocity and acceleration of all joints, the joint torques required to travel from point k to $k+1$ can be calculated using (9). After calculating the all joint torques, the second

inequality constraint (7) is checked. If any of the joint torques does not satisfy the inequality torque constraint, $\dot{q}_{i^*}(j_k)$ said to be inadmissible.

The next step after satisfying the all constraints is to calculate the incremental performance index to move from point k to $k+1$. Here $\Phi(\dot{q}_{i^*}(j_k), k)$ indicates the cost to move from point k to $k+1$ with a joint velocity of $\dot{q}_{i^*}(j_k)$. Bellman's optimality principle is applied to calculate the minimum performance index(Bellman [1957]).

$$J^f(\dot{q}_{i^*}(j_k), k) = \min_{\dot{q}_{i^*}(j_{k+1}), j=1,2,\cdots,N_v} [\Phi(\dot{q}_{i^*}(j_k), k)] + J^f(\dot{q}_{i^*}(j_{k+1}), k+1) \quad (17)$$

Equation-(17) searches all over the admissible discretized velocities at point k. Thus, every admissible velocity of joint i^* at point k gives a unique admissible velocity of the same joint at point $k+1$. In parallel, the velocities, accelerations and torques can be calculated corresponding to the optimal conditions.

3.1 Algorithm

With the help of these formulas, the dynamic programming algorithm can be stated in detail. The algorithm is as follows:
Step 1: Discretize the given Cartesian path $\mathbf{p}$ into N_p segments (a total of N_p+1 points).
Step 2: Calculate the joint displacement on each point using inverse kinematics, (10).
Step 3: Select a joint which is non-stationary along the movement on given path. Denote this reference non-stationary joint as i^*.
Step 4: Discretize the allowable velocity of joints into N_v segments. Now we have a grid, N_p points on $q_{i^*} - axis$ (columns) and N_v points on $\dot{q}_{i^*} - axis$ (rows).
Step 5: Associate the each point $(q_{N_p}, \dot{q}_{N_v})$ of rectangular grid a cost $C_{k,j}$ and a pointer $P_{k,j}$ indicating the next row. Set the cost $C_{k,j}$ at all nodes to infinity except for the cost of final state. This is set to zero. Similarly, initialize all the pointers $P_{k,j}$ to null. Set the column counter k to $N_p(k = 1, 2, \cdots, N_p,\ j = 1, 2, \cdots, N_v)$.
Step 6: If the column counter k is zero, then stop and go to Step 16.
Step 7: Otherwise set the current-row counter j_{k-1} to zero. Term j_{k-1} represent the index of discretized velocity at point $k-1$. Similarly j_k represents the index of discretized velocity at point k.
Step 8: If $j_{k-1} = N_v$, go to Step 15.
Step 9: Otherwise, set the new-row counter j_k to zero.
Step 10: If $j_k = N_v$, go to Step 14.
Step 11: Generate the curve that connects the point $(k-1, j_{k-1})$ to (k, j_k). For this displacement calculate the joint velocities, accelerations and torques for reference joint. If any of the inequality constraint is not satisfied then set this velocity as inadmissible and go to Step 13.
Step 12: Calculate the incremental performance index to move from point $(k-1, j_{k-1})$ to (k, j_k). The cost of this displacement is calculated by adding the C_{k,j_k} to the incremental performance index. If this cost is less than the cost $C_{k-1,j_{k-1}}$, then set $C_{k-1,j_{k-1}}$ to this cost. Set the pointer $P_{k-1,j_{k-1}}$ to point the grid point (k, j_k) and it produce the minimum cost.
Step 13: Increment the next-row counter j_k and go to Step 10.
Step 14: Increment the current row counter j_{k-1} and go to Step 8.
Step 15: Decrement the column counter k and go to Step 6.
Step 16: Calculate the optimal velocity sequence for joint i^* by tracking the pointer from initial to final state.
Step 17: Calculate the corresponding velocities and the forces/torques of all other joints.

This dynamical programming algorithm was proposed by Singh et al. [1987]. In this algorithm, first optimal velocity sequence of joint i^* is calculated and then the corresponding velocities and torques of other joints are calculated (Step 17). This algorithm is interpolation free as the path is known and the algorithm always moves from one discrete position to the next position of all joints.

4. JOINT SELECTION CRITERION

The dynamic programming algorithm discussed in Section 3, proposed by Singh et al. [1987], is practically applicable and can solve the optimal trajectory planning with any arbitrary cost function. Besides the applicability of this algorithm, it has some limitations which requires modification. First, this algorithm does not give any rule for selecting the reference non-stationary joint in case of more than one non-stationary joints (Step 3 of algorithm). In some cases, it happens that there is more than one non-stationary joints when following the given path. For example, in case of parallel robots, there is always more than one joints that are non-stationary during the movement of the manipulator on the given path. In this scenario, it is always a difficult task to choose a non-stationary joint amongst more than one non-stationary joints because choosing a non-stationary joint randomly does not guarantee the global optimal solution and may end up with a non-optimal solution. An alternative way is to solve the whole problem with respect to each non-stationary joint and get the optimal solution by comparing the result of each individual case.

In this paper, we present a novel heuristic based joint selection criterion in case of more than one non-stationary joints.

Criterion 1. To get the global optimal solution, select the non-stationary joint as a reference (Step 3) which has the lowest sum of absolute differences between pairs of successive discrete angles along the given path.

$$\text{Reference Joint}(i^*) = \underset{i=1,2,\cdots,n_{dyn}}{\arg\min} \left(\sum_{m=1}^{N_p-1} |q_i(m+1) - q_i(m)| \right) \quad (18)$$

In Equation-18, n_{dyn} represents the number of non-stationary joints during the movement of the manipulator on the given path.

The sum of the absolute differences of each non-stationary joint between pairs of successive discrete angles must be comparable. If the sum of absolute differences of a non-stationary joint is very small and not comparable to the other non-stationary joints, although it satisfy the above

Table 1. Dynamic Coefficients and Actuator Characteristics of DELTA D4-500 robot

Parameter	Description	Value
L_A	Length of arm	0.15 m
L_B	Length of forearm	0.4 m
R_A	Distance from the centre of base to the motor joint	0.1 m
R_B	Distance from the centre of travelling to the joint	0.04 m
τ_{min}	Minimum joint Torque	-35.2 Nm
τ_{max}	Maximum joint Torque	35.2 Nm
m_n	Mass of Travelling Plate	0.222 Kg
m_{ab}	Mass of the forearm	0.248 Kg
m_{br}	Mass of the arm	0.14 Kg
m_c	Mass of the elbow	0.042 Kg
I_m	Inertia of the motor	3.96×10^{-5}

mentioned criterion but the optimal trajectory can not be calculated by selecting this joint as a reference.

The modification that is necessary in this algorithm is to calculate of velocities and forces/torques and to check the constraints for all other joints in parallel to the reference joint inside the loop (Step 11), instead of calculating at the end (Step 17). It might be possible that at some nodes the constraints of the reference joint are satisfied but any other joint does not satisfy these constraints at the same node with same travelling distance. By using the existing algorithm, this problem will arise only at the last step of algorithm and the available option will be to solve the problem again, that will be computationally expensive and requires a lot of time. The proposed modification will handle this problem inside the loop. In case of constraints violation by any joint, the corresponding velocity will be set inadmissible and there will be no need to repeat the whole problem in case of any non-reference joint's constraint violation.

Until now we have only the experimental evidence for the validity of this heuristic based joint selection criterion and currently working on its mathematical proof.

5. NUMERICAL EXAMPLES

For the validation purpose, the dynamic programming algorithm with the proposed joint selection criterion and the modification is applied on a parallel DELTA robot. As discussed earlier, in parallel robots there are always more than one non-stationary joints during the movement of manipulator along the given path. So, the parallel robot would be a good test bench to test the proposed joint selection criterion. DELTA robot D4- 500 from CODIAN Robotics is used to test the trajectory planning. The dynamical model of the parallel DELTA robot is discussed by Codourey [1996]. The actuator characteristics and the dynamic coefficient of DELTA D4-500 robot are summarized in Table-1 and Fig. 1 shows definition of the geometric parameters used in Table-1.

In this section, the proposed criterion is tested on different paths using DELTA robot to support the validation. Numerical examples are considered and the results and compared with some state-of-the-art optimization methods. All numerical simulations were performed on Intel Core I5 2.60GHz Laptop with 4GB of RAM.

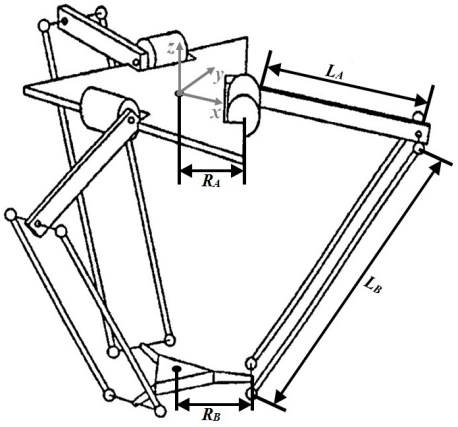

Fig. 1. DELTA robot and its geometric parameters (Codourey [1996])

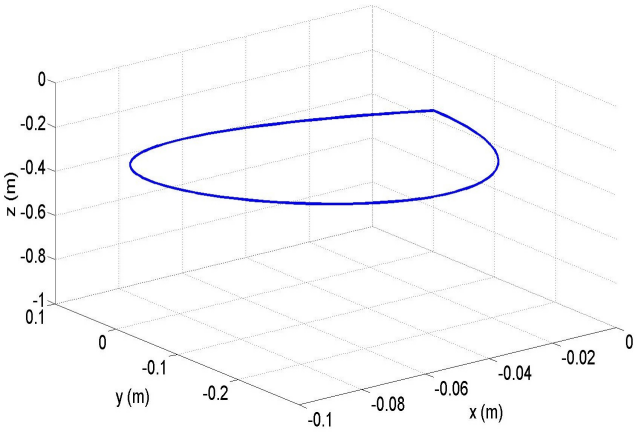

Fig. 2. Given Path for Example 1

5.1 Example 1

In first example, a path, shown in Fig. 2, is considered for optimization using different optimization methods and the results are compared with obtained by the algorithm proposed by Singh et al. [1987], to show the validation of proposed criterion. The cost function in this example is to find the optimal trajectory to minimize the travelling time. All the three joints of the DELTA robot are non-stationary while following the given path.

For the purpose of comparison and to show the validation of proposed criterion, the optimal trajectory of the given path is calculated using three different techniques; the proposed modified dynamic programming algorithm, the dynamic programming using path parameter s proposed by Shin et al. [1986] and the phase plane method proposed by Bobrow et al. [1985] and Shin et al. [1985]. The grid of 80×80 is used in the above mentioned optimization techniques to find the optimal trajectory (i.e. $N_p = 80, N_v = 80$). The results obtained by different techniques are summarized in Table-2.

The computational time (t_{comp}) and the optimized value of cost function obtained by different techniques, shown

Table 2. Comparison of the optimal results by different optimization techniques (Example-1)

Modified Singh et al. [1987]				Shin et al. [1986]		Phase Plane Method	
Reference Joint	Sum of absolute difference	Cost	t_{comp}(sec)	Cost	t_{comp}(sec)	Cost	t_{comp}(sec)
Joint-1	15.2207	0.2580	125.360	0.2680	125.517	0.2655	14.722
Joint-2	28.8829	0.3997	125.234				
Joint-3	19.0677	0.2971	126.454				

Table 3. Comparison of the optimal results by different optimization techniques (Example-2)

Modified Singh et al. [1987]				Shin et al. [1986]		Phase Plane Method	
Reference Joint	Sum of absolute difference	Cost	t_{comp}(sec)	Cost	t_{comp}(sec)	Cost	t_{comp}(sec)
Joint-1	11.7408	0.4300	31.513	0.3972	33.829	0.3986	15.284
Joint-2	11.0382	0.3996	31.857				
Joint-3	11.9253	0.4367	31.902				

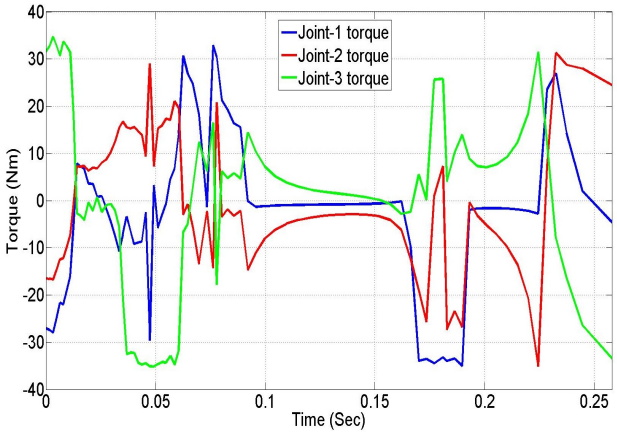

Fig. 3. Optimal torque of joints

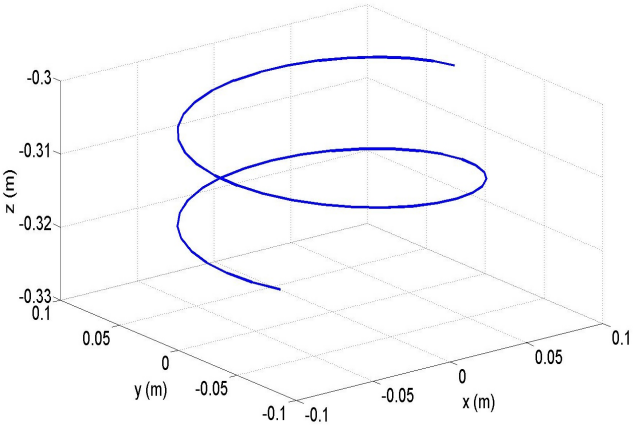

Fig. 4. Path for Example 2

in Table-2, are comparable to each other. To get the optimal solution using the dynamic programming algorithm proposed by Singh et al. [1987], the problem is solved three times with respect to each non-stationary joint. The different optimal costs are obtained by considering the each joint separately as a reference and the global optimal solution is obtained by considering the Joint-1 as reference joint. According to the proposed joint selection criterion, discussed in Section 4, the Joint-1 must give the optimal solution as it has the lowest sum of absolute differences and the sum of absolute differences of Joint-1 is comparable to all other non-stationary joints. The results in Table-2 show the validation of the proposed joint selection criterion. The optimal joints torque for the optimal trajectory, shown in Fig. 3, are within the limits of joint torques summarized in Table-1.

5.2 Example 2

In Example-2, a 3D spring movement, shown in Fig. 4, is considered for the optimization. The objective in this example is to find a time-optimal trajectory while satisfying the constraints on joint torques. Just like the example 1, this problem is also solved by three different optimization techniques and the results are compared. A grid size of 50×50 is used in the optimization algorithms. The results obtained by different techniques are shown in Table-3.

The results obtained by different techniques, shown in Table-3, are comparable to each other. Joint-1 gives the global optimal solution when it is considered as a reference

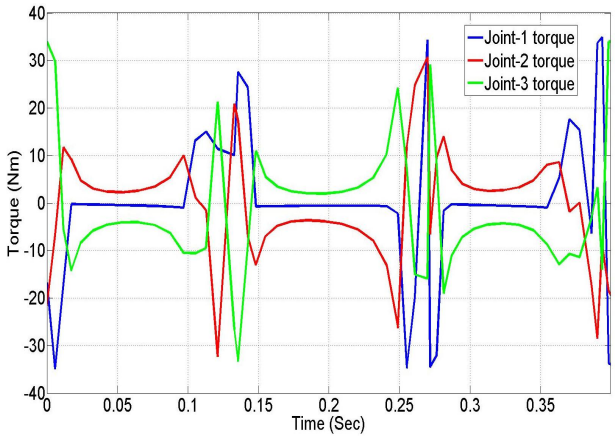

Fig. 5. Optimal torque of joints

joint in the dynamic programming algorithm proposed by Singh et al. [1987]. As per proposed joint selection criterion the Joint-1 must give the global optimal solution as it fulfils the proposed criterion, its sum of absolute differences between pairs of successive discrete points is lowest and comparable to the all other non-stationary joint's value. The joints torque for the time-optimal trajectory are shown in Fig. 5.

In this section two examples are discussed to show the validity of the proposed joint selection criterion. The results obtained by the modified Singh et al. [1987] algorithm are compared with the results obtained by phase plane method, proposed by Bobrow et al. [1985], and the

dynamic programming using path parameter s, proposed by Shin et al. [1986]. The results show the validation of the joint selection criterion that is useful in the selection of a non-stationary reference joint among more than one non-stationary joints to get the global optimal solution by using DP algorithm proposed by Singh et al. [1987]. The reference joint selection for the global optimal solution can be made directly from the given path information by using this joint selection criterion. The computational time taken by different optimization techniques is also given in the respective tables of the examples and one can compare the computational efficiency of different techniques. As phase plane method is dedicated to solve the time minimization problem, so it is computational efficient. The only drawback of the dynamic programming method, also evident from these examples, is that it is computationally expensive as it has to check all possible combinations. These joint selection criterion is tested on many other optimal trajectories planning for DELTA robot. To keep the length of the paper under limits, only two examples are presented.

6. CONCLUSION

In this paper a joint selection criterion is proposed to select a non-stationary reference joint to optimize the given path using dynamic programming algorithm proposed by Singh et al. [1987]. The non-stationary reference joint can be selected using the proposed criterion directly from the given path information. By using this criterion the computational time of the problem is reduced and there is no need to solve the problem multiple times by considering the each non-stationary joint as a reference. The applicability and the validation of the proposed criterion are supported by numerical examples. The optimal results obtained by using the joint selection criterion and modified algorithm are compared with the results obtained by some state-of-the-art techniques. Currently, we are working on the mathematical proof of proposed heuristic based joint selection criterion.

ACKNOWLEDGEMENTS

The authors would like to thank the International Graduate School of Dynamic Intelligent System at the University of Paderborn for the financial support of this research.

REFERENCES

Behzadipour, S. and Khajepour, A. (2006). Time-optimal trajectory planning in cable-based manipulators. *IEEE Trans. on Robotics*, 22(3), 559–563.

Bellman, R. (1957). Dynamic programming. *Princeton University Press*, Princeton, N.J.

Bobrow, J. E., Dubowsky, S., and Gibson, J. S. (1985). Time-optimal control of robotic manipulators along specified paths. *Int. J. Robotic Research*, 4(3).

Codourey, A. (1996). Dynamic modelling and mass matrix evaluation of the DELTA parallel robot for axes decoupling control. In *Proc. of IEEE Int. Conf. on Intelligent Robots and Systems*, 3, 1211–1218, Osaka, Japan.

Field, G. and Stepanenko, Y. (1996). Iterative dynamic programming: an approach to minimum energy trajectory planning for robotic manipulators. In *Proc. of IEEE Int. Conf. on Robotics and Automation*, 3, 2755–2760, Minnesota, US, 1996.

Latombe, J. (1991). Robot motion planning. *Kluwer Academic Publishers*, Boston/Dordrecht/London.

Pfeiffer, F. and Johanni, R. (1987). A concept for manipulator trajectory planning. *IEEE Journal of Robotics and Automation*, 3(2).

Rajan, V. T. (1985). Minimum time trajectory planning. in *Proc. International Conference on Robotics and Automation*, 759–764, New York, US.

Shiller, Z. (1992). On singular points and arcs in path constrained time optimal motions. *ASME, Dyn. Syst. Contr. Division DSC-42*.

Shiller, Z. and Dubowsky, S. (1991). On computing the global time-optimal motions of robotic manipulators in presence of obstacles. *IEEE Trans. Robotics and Automation*, 7(6), 785–797.

Shiller, Z. and Lu, H. H. (1992). Computation of path constrained time optimal motions with dynamic singularities. Trans. of the *ASME Journal of Dynamic Systems, Measurement and Control*, 114(1), 34–40.

Shin, K. G. and McKay, N. D. (1985). Minimum-time control of robotic manipulators with geometric path constraints. *IEEE Trans. on Automatic Control*, 30(6).

Shin, K. G. and McKay, N. D. (1986). A dynamic programming approach to trajectory planning of robotic manipulators. *IEEE Trans. on Automatic Control*, 31(6).

Singh, S. and Leu, M. C. (1987). Optimal trajectory generation for robotic manipulators using dynamic programming. *Journal of Dynamic Systems, Measurement and Control*, 109(2), 88–96.

Slotine, J. J. E. and Yang, H. S. (1989). Improving the efficiency of time-optimal path-following algorithms. *IEEE Trans. on Robotics and Automation*, 5(1).

Tarkiainen, M. and Shiller, Z. (1993). Time optimal motions of manipulators with actuator dynamics. In *Proc. of IEEE Int. Conf. on Robotics and Automation*, 2, 725–730, Atlanta, US.

Verscheure, D., Demeulenaere, B., Swevers, J., Schutter, J. D., and Diehl, M. (2008). Practical time-optimal trajectory planning for robots: a convex optimization approach. *IEEE Trans. on Automatic Control*.

Vukobratovic, M. and Kircanski, M. (1982). A method for optimal synthesis of manipulation robot trajectories. *Journal of Dynamic Systems, Measurement and Control*, 104(2), 188–193.

Yen, V. (1995). An inverse dynamic-based dynamic programming method for optimal point-to-point trajectory planning of robotic manipulators. *Int. Journal of System Sciences*, 26(2).