Proceedings of the 17th World Congress
The International Federation of Automatic Control
Seoul, Korea, July 6-11, 2008

Optimal Protraction of a Three-Joint Robot Leg[1]

Mustafa Suphi Erden*, Kemal Leblebicioğlu**

**BioMechanical Engineering, Intelligent Mechanical Systems Group, Delft university of Technology, Delft, 2628 CD, The Netherlands, (Tel: +31 (0)15 27 86794; e-mail: m.s.erden@tudelft.nl).*
***Department of Electrical & Electronics Engineering, Computer Vision and Intelligent Systems Research Laboratory, Middle East Technical University, Ankara, 06531, Turkey, (Tel: +90 (0)312 210 4558; e-mail: kleb@metu.edu.tr)*

Abstract: In this paper protraction movement, namely stepping ahead, of a three joint robot leg is optimized for energy efficiency for any given pair of initial-final tip point positions. For the optimization a modified version of gradient descent based optimal control algorithm with Hamiltonian formulation is used. The objective function is modified in steps to jump over the infeasible and inefficient local optimums. The results of 79 optimizations are used to construct a radial basis function neural network (RBFNN) in order to interpolate between the optimized trajectories. The results are presented and discussed in the paper.

1. INTRODUCTION

Due to its advantages over wheeled and tracked systems on uneven terrains, legged locomotion has the potential to be applied in various fields (Preumont et al., 1997; Huang et al., 2003). A disadvantage of legged locomotion is its high energy consumption (Erden and Leblebicioğlu, 2007). During walking, some of the legs are in swing phase for protraction. The protracting legs do not contribute to lifting, but exist as extra weights. The protraction movement carries the tip point of the leg from a given posterior extreme position to a given anterior extreme position. Generating this movement corresponds to generating proper traces of joint angles that achieve the transfer of the tip point. The topic of this paper is generation of the protraction movement in an energy optimal way.

In the literature there are quite scarce studies which explicitly deal with the protraction movement of robot legs. Ilg et al. (1995, 1997) deal with application of reinforcement learning for the power stroke (retraction) and return stroke (protraction) of a robot leg. In these papers the focus is application of reinforcement learning, rather than the efficiency of the resulting movements. Cruse et al. (1998) describe a general control structure for protraction under the subtitle of "control of the swing movement". Similarly, Dürr et al. (2004) explicitly mention protraction as one of the four mechanically uncoupled swing movements of robot legs, besides searching, grasping, and grooming. These last two papers content with a description of very general, and in fact similar, kinematic control models, which aim to imitate the biologically inspired movements observed in insects. However, the stress on the similarities of the mentioned four movements by Dürr et al. (2004) reveals the importance of the control of such group of swing movements. This points to the need of a generic protraction movement generator applicable to similar swing movements. Especially the reflex movements to be performed when the robot comes across unexpected obstacles, holes, or hills (Espenschied et al., 1996) are considerable in this regard. The optimization performed in this paper is generalizable to any similar movement once the initial-final tip point positions are provided. Erden et al. (2004a) presents a preliminary work for the approach developed in this paper.

Biological observations reveal that the protraction time in six-legged insects is constant regardless of the speed of walking. Ferrell (1995) gives summaries of three models for biologically inspired six legged locomotion. In two of these, namely in the Wilson and Pearson models, it is explicitly stated that the protraction time is constant. This is sense-full since there is no power load on the leg during the swing motion, and it is possible to protract in a quite short and constant time. Erden and Leblebicioğlu (2007) demonstrate that the gaits with minimum protraction time are more energy efficient. In this paper efficiency corresponds to dissipation of minimum energy in the three actuators during protraction. Following the arguments of Erden and Leblebicioğlu (2007) and Ferrell (1995) the protraction time is taken to be the minimum applicable to the Robot-EA308 (Fig. 7), which is *1.5 seconds* .

2. THREE JOINT ROBOT LEG

A three joint robot leg can be considered as a three link revolute joint (RRR) manipulator which is attached to a stationary base (robot body). Therefore, the kinematic modelling and derivation of dynamic equations can be performed following the conventional robotics approaches (Fu et al., 1987). The kinematic model here is derived by defining the reference frames according to the Denavit-Hartenberg convention. In Fig. 1 a graphical representation of a three joint robot leg is given, with the attached reference frames and corresponding joint variables. In this figure the

[1]This research is supported by the research fund of Middle East Technical University as a scientific research project: BAP–2002–03–01–06.

body *(b)* and the zeroth *(0)* reference frames are attached to the stationary robot body. Therefore they can be both considered as inertial frames. (It is assumed throughout the paper that motion of the body is much more slower than the motion of the leg during protraction. Therefore, the body is assumed to be stationary.) The Denavit-Hartenberg link parameters based on Fig. 1 are given in Table 1. The homogeneous transformation matrices between the body and the zeroth frame, and between the sequential link frames can be easily computed. In (1) the tip point position of the leg with respect to the body frame is given.

$$\begin{bmatrix} p_x(\bar\theta) \\ p_y(\bar\theta) \\ p_z(\bar\theta) \end{bmatrix} = \begin{bmatrix} C\Psi(a_1 S\theta_1 + a_2 S\theta_1 C\theta_2 + a_3 S\theta_1 C\theta_{23}) + S\Psi(a_2 S\theta_2 + a_3 S\theta_{23}) \\ -(a_1 C\theta_1 + a_2 C\theta_1 C\theta_2 + a_3 C\theta_1 C\theta_{23}) \\ -S\Psi(a_1 S\theta_1 + a_2 S\theta_1 C\theta_2 + a_3 S\theta_1 C\theta_{23}) + C\Psi(a_2 S\theta_2 + a_3 S\theta_{23}) \end{bmatrix} \tag{1}$$

Table 1. Denavit-Hartenberg link parameters for the three joint robot leg.

Joint	θ_i	α_i	a_i	d_i
1	θ_1	$\pi/2$	a_1	0
2	θ_2	0	a_2	0
3	θ_3	0	a_3	0

The dynamic equations are derived using the Lagrangian formalism (Fu et al., 1987), which leads to the compact dynamic equation (2). In this equation $\hat{M}$ is the mass (inertia) matrix; $\bar{C}$ is the vector of coriolis, centrifugal, and gyroscopic terms; $\bar{G}$ is the vector of gravitational terms; and $\bar{Q}$ is the vector of the three joint torques.

$$\hat{M}(\bar{q})\ddot{q} + \bar{C}(q, \dot{q}) + \bar{G}(q) = \bar{Q} \tag{2}$$

3. TRAJECTORY OPTIMIZATION

The trajectory optimization is performed using the gradient based optimal control theory (Kirk, 1970, pp.184-209, 236-240, 330-343). Gradient based optimization algorithms suffer from sticking to local optimums, especially when the dimension of vectors to be optimized is large. In trajectory optimization problems the dimension of vectors that represent the whole trajectory happen to be quite large. In order to cope with this problem it is a common approach to represent trajectories with parametrized polynomials which satisfy the initial and final position requirements. Cubic splines are widely used for this purpose (Bobrow et al., 2001; Chettibi et al, 2004; Garg et al., 2002; Saramago et al., 1998). Once the trajectories are represented by a small number of spline parameters, the trajectory optimization problem is reduced to optimizing the parameters with respect to a given objective function. Application of non-gradient based optimization algorithms, such as genetic algorithms, is also common to be used with such reduced representation of trajectories. Garg et al. (2002) reduce the problem of trajectory optimization for a two link manipulator to optimizing only two parameters with genetic algorithms.

With polynomial representations trajectories are prisoned to a limited search space defined by the structure of the polynomial. The most optimistic solution of those approaches is the best solution in the space represented by the structure of the polynomial. If it is not desired to sacrifice the freedom of the trajectory, it is not proper to adopt reduced representations of trajectories based on polynomial parameters. The attempt in this paper is to utilize an almost full representation of trajectory, in order to access a wide range of solution space for optimization. An almost full trajectory representation necessitates a large number of unknowns to be optimized. In this paper, the trajectory is represented with 51 actual joint angle values (or tip point coordinates) which makes up to 152 parameters. This size of parameters makes it difficult to be handled by non-gradient based optimization algorithms. For example, the coding of a large input vector leads to too large chromosomes for genetic algorithms. Moreover, in evolutionary optimizations, the parameters are optimized independently. Therefore, they may lead to non-smooth (non-continuous) solution sets if the parameters optimized are one to one trajectory values (as it is the case here), rather than auxiliary variables representing the trajectory (as it is the case in cubic spline representations). As a result, gradient based optimization is more proper from the stand point of producing analytical solutions with high dimensional one to one trajectory representations.

However, how to come over the problem of sticking to local optimums then? The approach here attempts to utilize gradient based optimization using the optimal control theory. The problem of local optimality is intended to be overcome by objective function modification in different epochs of gradient based optimization. In the following the formulation for optimal control, the objective function modification, and optimization results are presented. In Section 4, the optimal trajectories are processed and an interpolating RBFNN is constructed to produce near-optimal trajectories for any given initial and final point positions.

3.1 State Space Representation and Hamiltonian Formulation of Three Joint Leg System

In a robotic manipulator the actual inputs to the system are the actuator forces and torques. For the three joint leg system the actual inputs are the three joint torques, $(\bar{Q})$. However, if

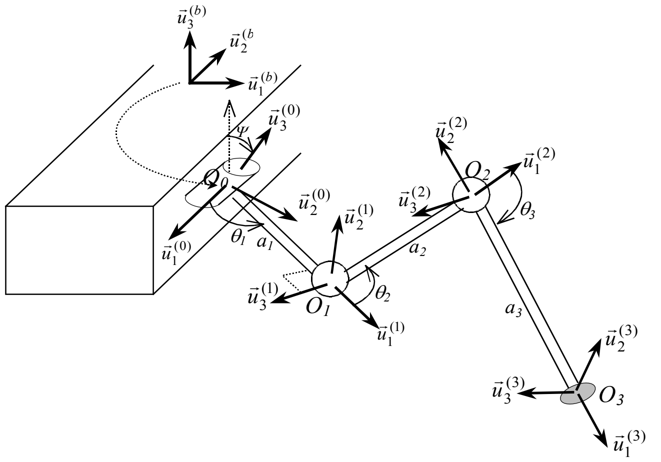

Fig. 1. Three joint robot leg: Reference frames and joint variables.

the aim is trajectory optimization, some auxiliary control variables can be chosen as the input, and the actual torques can be calculated using these (Frangos et al., 2001). Following this approach the auxiliary variables, namely the inputs to the system, are chosen to be the angular accelerations of the joints, $(\ddot{\bar{\theta}})$. After these are determined the actual joint torques can be calculated using (2). There are three kinds of requirements in the formulation: system dynamics, initial and final state conditions, and joint angle constraints. In handling these requirements the approaches in (Kirk, 1970) are followed. Initial state conditions are imposed in the initialization of the optimization; the final state conditions are imposed in the objective function as a penalty term; joint angle constraints are imposed using a supplementary state variable, named as the constraint-state; and the system dynamics are imposed in the dynamic equations of the state variables. There is no restriction on the control input. The states are of three groups: the joint angles $(\bar{\theta})$; the joint velocities $(\dot{\bar{\theta}})$; and a constraint-state. The state and input vectors of the overall system are given in (3). Following the notation of Kirk (1970) the system dynamics can be written as in (4).

$$\bar{x}=\begin{bmatrix}\bar{x}_q\\ \bar{x}_{\dot{q}}\\ x_c\end{bmatrix}=\begin{bmatrix}\theta_1\\ \theta_2\\ \theta_3\\ \dot{\theta}_1\\ \dot{\theta}_2\\ \dot{\theta}_3\\ x_c\end{bmatrix}=\begin{bmatrix}x_1\\ x_2\\ x_3\\ x_4\\ x_5\\ x_6\\ x_7\end{bmatrix} \qquad \bar{u}(t)=\begin{bmatrix}\ddot{\theta}_1(t)\\ \ddot{\theta}_2(t)\\ \ddot{\theta}_3(t)\end{bmatrix} \tag{3}$$

$$\dot{\bar{x}}=\bar{a}(\bar{x}(t),\bar{u}(t),t) \tag{4}$$

The state derivatives are defined in (5). The seventh state is the constraint-state whose dynamics is defined by the function $f_c(\bar{x}(t))$. There are two kinds of constraints on the joint angles: the ones imposed by the mechanical construction of the joints, and the one that limits the tip point from going under the ground level. These constraints are given in (6), where $t_{1a}=30^0$; $t_{1b}=150^0$; $t_{2a}=0^0$; $t_{2b}=150^0$; $t_{3a}=-150^0$; $t_{3b}=-30^0$; and the ground level is taken to be $z_g=-9\ cm$.

$$\begin{aligned}
\dot{x}_1(t)&=a_1(\bar{x}(t),\bar{u}(t),t)=\dot{\theta}_1=x_4\\
\dot{x}_2(t)&=a_2(\bar{x}(t),\bar{u}(t),t)=\dot{\theta}_2=x_5\\
\dot{x}_3(t)&=a_3(\bar{x}(t),\bar{u}(t),t)=\dot{\theta}_3=x_6\\
\dot{x}_4(t)&=a_4(\bar{x}(t),\bar{u}(t),t)=\ddot{\theta}_1=u_1\\
\dot{x}_5(t)&=a_5(\bar{x}(t),\bar{u}(t),t)=\ddot{\theta}_2=u_2\\
\dot{x}_6(t)&=a_6(\bar{x}(t),\bar{u}(t),t)=\ddot{\theta}_3=u_3\\
\dot{x}_7(t)&=a_7(\bar{x}(t),\bar{u}(t),t)=f_c(\bar{x}(t))
\end{aligned}
\qquad
\dot{\bar{x}}=\begin{bmatrix}\dot{\bar{x}}_q\\ \dot{\bar{x}}_{\dot{q}}\\ \dot{x}_c\end{bmatrix}=\begin{bmatrix}\bar{x}_{\dot{q}}\\ u\\ f_c(\bar{x}(t))\end{bmatrix} \tag{5}$$

$$\begin{aligned}
&t_{1a}\leqslant\theta_1\leqslant t_{1b}\\
&t_{2a}\leqslant\theta_2\leqslant t_{2b}\\
&t_{3a}\leqslant\theta_3\leqslant t_{3b}\\
&p_z(\bar{\theta})=-S\Psi(a_1S\theta_1+a_2S\theta_1C\theta_2+a_3S\theta_1C\theta_{23})\\
&\qquad\qquad +C\Psi(a_2S\theta_2+a_3S\theta_{23})\geqslant z_g
\end{aligned} \tag{6}$$

Based on these constraints the seven f functions in (7) are determined, all of which should be greater than *0* in order the constraints to be satisfied. The dynamics of the constraint-state is defined as given in (8). The φ function used in (8) is given in (9).

$$\begin{aligned}
f_1(\bar{x}(t),t)&=\theta_1-t_{1a}\geqslant 0\\
f_2(\bar{x}(t),t)&=t_{1b}-\theta_1\geqslant 0\\
f_3(\bar{x}(t),t)&=\theta_2-t_{2a}\geqslant 0\\
f_4(\bar{x}(t),t)&=t_{2b}-\theta_2\geqslant 0\\
f_5(\bar{x}(t),t)&=\theta_3-t_{3a}\geqslant 0\\
f_6(\bar{x}(t),t)&=t_{3b}-\theta_3\geqslant 0\\
f_7(\bar{x}(t),t)&=1000(p_z(\bar{\theta})-z_g)\geqslant 0
\end{aligned} \tag{7}$$

$$\dot{x}_c(t)=f_c(\bar{x}(t))=\sum_{i=1}^{7}[f_i(\bar{x}(t))]^2\varphi(-f_i) \tag{8}$$

$$\varphi(-f_i)=\begin{Bmatrix}0, & for\ f_i(\bar{x}(t),t)\geqslant 0\\ 1, & for\ f_i(\bar{x}(t),t)<0\end{Bmatrix} \quad for\ i=1,2,...7 \tag{9}$$

The initial and final conditions for the constraint-state are set to *0*. Since the φ function is positive when the constraint is not satisfied and zero when satisfied, the derivative of the constraint-state happens to be either positive (if any of the constraints is not satisfied) or zero. The conditions that the initial and final values are zero, and the derivative is non-negative force the constraint-state to remain zero throughout the protraction period. In this way all the seven constraints in (7) are satisfied.

It is a common approach to use the integral of torque squares as an index of energy dissipation in the actuators of robotic manipulators (Bobrow et al., 2001; Garg et al., 2002; Liu et al., 2000). The torque vector of the three joint leg system is given in (10). Combining the torque square integration with the final state conditions leads to the objective function in (11), for which again the notation of Kirk (1970) is followed. The optimization problem can be stated as in (12). This leads to a two point boundary value problem in which the state and control variables are not constrained by any boundaries, the final time t_f is fixed, and $\bar{x}(t_f)$ is free (in fact it is not free, but it can be considered as free in the formulation since the requirement is imposed in the objective function by $h(\bar{x}(t_f),t_f)$).

$$\begin{aligned}
\bar{Q}(t)&=\bar{Q}(\bar{x}(t),\bar{u}(t))=\bar{Q}(\bar{\theta}(t),\dot{\bar{\theta}}(t),\ddot{\bar{\theta}}(t))\\
&=\hat{M}(\bar{q})\ddot{\bar{q}}+\bar{C}(\bar{q},\dot{\bar{q}})+\bar{G}(\bar{q})
\end{aligned} \tag{10}$$

$$\begin{aligned}
J(\bar{u})&=h(\bar{x}(t_f),t_f)+J_s(\bar{x}_s,\bar{u})\\
&=h(\bar{x}(t_f),t_f)+\int_{t_0}^{t_f} g(\bar{x}(t),\bar{u}(t),t)\,dt\\
&=(\bar{x}(t_f)-\bar{x}_{tf})^T\hat{D}_h(\bar{x}(t_f)-\bar{x}_{tf})\\
&\qquad+\int_{t_0}^{t_f}\bar{Q}(\bar{x}(t),\bar{u}(t))^T\hat{D}_g\bar{Q}(\bar{x}(t),\bar{u}(t))\,dt
\end{aligned} \tag{11}$$

$$\begin{aligned}
&\textit{Minimize} && J(\bar{u})\\
&\textit{subject to} && \dot{\bar{x}}(t)=\bar{a}(\bar{x}(t),\bar{u}(t),t)\\
& && \bar{x}(t_o)=\bar{x}_o
\end{aligned} \tag{12}$$

Following the optimal control approach (Kirk, 1970), the Hamiltonian function can be given as in (13), where $\bar{p}(t)$ corresponds to the costate vector.

$$H(\bar{x}(t),\bar{u}(t),\bar{p}(t),t) \\ = g(\bar{x}(t),\bar{u}(t),t)+\bar{p}(t)^T\bar{a}(\bar{x}(t),\bar{u}(t),t) \tag{13}$$

The necessary and boundary conditions for an optimal solution can be written as follows:

Necessary conditions for optimality:

$$\dot{\bar{x}}^*(t)=\bar{a}(\bar{x}^*(t),\bar{u}^*(t),t) \tag{14}$$

$$\dot{\bar{p}}^*(t)=\frac{-\partial H}{\partial \bar{x}}(\bar{x}^*(t),\bar{u}^*(t),\bar{p}^*(t),t) \tag{15}$$

$$\bar{0}=\frac{\partial H}{\partial \bar{u}}(\bar{x}^*(t),\bar{u}^*(t),\bar{p}^*(t),t) \tag{16}$$

Boundary conditions:

$$\bar{x}(t_0)=\bar{x}_0 \tag{17}$$

$$\bar{p}(t_f)=\frac{\partial h}{\partial \bar{x}}(\bar{x}(t_f)) \tag{18}$$

The first two equations of the necessary conditions make up two differential equations whose initial and final conditions are given by the boundary conditions equations. The trajectory of input vector is approximated with a dense discretization of 51 instants, which results in quite a high dimensional optimization input of 3×51 length. Starting with an initial input trajectory, namely 51 values of the vector $\bar{u}(t)$ for the discretized instances, the state and costate equations can be solved numerically, by forward and reverse integrations, respectively. The third equation of necessary conditions is in fact nothing but the gradient of the objective function with respect to the input vector. Therefore, $\bar{u}(t)$ can be updated in the negative direction of this gradient in order to minimize the objective function, as in (19), where *i* stands for the iteration number. After some iteration the optimal $\bar{u}(t)$ trajectory, which makes the third necessary condition as close as possible to *0*, can be achieved. This technique is called "the method of steepest descent for two-point boundary-value problems" (Kirk, 1970). The initial $\bar{u}(t)$ trajectory in this work is taken to be a zero matrix of 3×51 dimension. The value of *α* in (19) is determined by a one-dimensional search in every step. The optimization is terminated when the difference between two successive objective functions is less than 0.01.

$$\bar{u}^{(i+1)}(t_k)=\bar{u}^{(i)}(t_k)-\alpha\frac{\partial H}{\partial \bar{u}}(t_k) \ , \\ t_k=(k-1)\frac{t_f}{50} \ , \ k=1,2,\ldots,51 \tag{19}$$

3.2 Optimization with Objective Function Modification

In the objective function of (11), there are two terms, respectively related to final state conditions and energy integration. These terms are weighted by the constant matrices of $\hat{D}_h$ and $\hat{D}_g$, respectively. These two are taken to be diagonal matrices as in (20).

Table 2. Sequential values of the weights of the objective function for "objective function modified optimization"

	Step 1	*Step 2*	*Step 3*	*Step 4*	*Step 5*	*Step 6*
d_h	1	10	1	10	1	10
d_g	10	1	10	1	100	1

$$\hat{D}_h=d_h\begin{bmatrix}500&0&0&0&0&0&0\\0&500&0&0&0&0&0\\0&0&500&0&0&0&0\\0&0&0&20&0&0&0\\0&0&0&0&20&0&0\\0&0&0&0&0&20&0\\0&0&0&0&0&0&10^7\end{bmatrix} \quad \hat{D}_g=d_g\begin{bmatrix}10^6&0&0\\0&10^6&0\\0&0&10^6\end{bmatrix} \tag{20}$$

The elements of the $\hat{D}_h$ and $\hat{D}_g$ matrices are arranged to have comparable values in the two terms of the objective function. The entry corresponding to the constraint-state in the $\hat{D}_h$ matrix is chosen to be very large in order to guarantee the satisfaction of the constraints. The d_h and d_g values determine the relative strength of the two terms with respect to each other. Choosing a large d_h and a small d_g results in a better achievement of final state requirements with little decrease in total energy dissipation; while a small d_h and a large d_g results in a big decrease of total energy dissipation with non-achievement of final state requirements. Therefore, there is a trade-off between the two considerations represented by these two terms. Another problem is that, whatever values these two parameters are assigned, the optimization stops in a local optimum which does not give a satisfactory trajectory. For example, even though d_h is chosen very large compared to d_g the optimization stops at a local minimum in which the tip point is not brought to the desired tip point position. Therefore, the problem is of two fold: sticking to local optimums, and trade-off between the two terms.

The approach adopted to overcome these two problems is to make optimizations with changing the d_h and d_g parameters sequentially. In this way the local optimum is jumped over by the change of objective function, and the two terms are enforced sequentially one after the other. The steps of optimization with objective function modification are given as follows:

```
Step 1:
    i) Initialize $\bar{u}(t_k)=\bar{0}$   for  $k=0,1,\ldots,51$.
    ii) $d_h=d_{h,1}$; $d_g=d_{g,1}$.
    iii) Perform optimization.
    iv) Output  $\bar{u}(t_k)=\bar{u}(t_k)^{[2]}$   for  $k=0,1,\ldots,51$.
Step i:
    v) Initialize  $\bar{u}(t_k)=\bar{u}(t_k)^{[i-1]}$   for  $k=0,1,\ldots,51$.
    vi) $d_h=d_{h,i}$; $d_g=d_{g,i}$.
    vii) Perform optimization.
    viii) Output  $\bar{u}(t_k)=\bar{u}(t_k)^{[i]}$   for  $k=0,1,\ldots,51$.
for i=2,...,6.
```

The changing values of d_h and d_g are given in Table 2. In Steps 1-4, a regular change is followed. In the step before the last, the jump of d_g is increased to give a last impulse in the direction of increased d_g, before the actual values of the last step are applied.

For the protraction movement it is possible to define a region of departure in the backward of the leg, and a region of arrival in the forward. These regions are determined to be two squares whose corners are given as [*(8,-7,-9);(12,-7,-9);(12,-3,-9);(8,-3,-9)*] and [*(8,3,-9);(12,3,-9);(12,7,-9);(8,7,-9)*] *(cm)*, respectively, with respect to the body frame. The tip point is assumed to depart from a point in the square of departure to a point in the square of arrival. The ground level is taken to be *-9 cm* in the z-direction. Nine points from each square are chosen as the sample positions for optimization. Eight of these are distributed homogeneously throughout the periphery of the square, and the remaining is taken to be the centre point. As a result, $9\times 9=81$ initial-final position pairs are used in the optimization; therefore, a total of *81* optimal trajectories are obtained to represent the protraction from the square of departure to the square of arrival. These trajectories are used in Section 4 to interpolate a trajectory for any given initial-final position points in those squares. The set of the initial and final point positions used for optimization are given in Table 3.

3.3 Comparative Results of "Optimization With Single Objective Function" and "Optimization With Objective Function Modification"

In Fig. 2 the joint angles, velocities, and accelerations resulted from the 81 optimizations with "single objective function" are given in the first three rows. The single objective function corresponds to only Step 6 of Table 2. The last row in Fig. 2 shows the tip point position errors between the actual tip point and the desired final tip point positions. Among the results in Fig. 2, a few of the optimizations ended with feasible trajectories, which came close to the desired final tip point position with a movement of lifting up the tip point in the start. Some of these feasible ones are shown by arrows in some of the figures in Fig. 2. Most of the remaining optimizations created infeasible trajectories which cannot be considered as a proper protraction at all. Not only the infeasible ones but also the few feasible trajectories are the local minimums of the single objective function. A much more improvement of those trajectories is possible with the objective function modified optimization.

In Fig. 3 the joint angles, velocities, accelerations, and tip point position errors resulted from the 81 optimizations with "objective function modification" are given. It is immediately realized how structured the figures are compared to the ones in Fig. 2. In 79 of the results given in Fig. 3, the optimization was successful to create a proper protraction: In the starting phase the leg is first retracted towards the body with raising the tip point above the ground level; in the middle phase the protraction towards the front is maintained; and in the end phase the leg is extracted carrying the tip point towards the desired final position.

Table 3. Set of initial and final point positions used for optimization.

Set of Initial Positions (m)				***Set of Final Positions (m)***		
$x^{(b)}$	$y^{(b)}$	$z^{(b)}$		$x^{(b)}$	$y^{(b)}$	$z^{(b)}$
0.08	-0.03	-0.09		0.08	0.03	-0.09
0.08	-0.05	-0.09		0.08	0.05	-0.09
0.08	-0.07	-0.09		0.08	0.07	-0.09
0.10	-0.03	-0.09		0.10	0.03	-0.09
0.10	-0.05	-0.09		0.10	0.05	-0.09
0.10	-0.07	-0.09		0.10	0.07	-0.09
0.12	-0.03	-0.09		0.12	0.03	-0.09
0.12	-0.05	-0.09		0.12	0.05	-0.09
0.12	-0.07	-0.09		0.12	0.07	-0.09

Among the 81 optimizations in Fig. 3, only two of them resulted in a different behaviour. These two are shown with arrows on the figures. In these two, the optimization did not manage to bring the tip point to the desired final position.

A comparison of the two optimizations regarding to cost minimization is given in Fig. 4. This figure shows the improvement for an infeasible case with "single objective function optimization". The fist row figures show the leg movement from different sights for the optimization result with "single objective function". The second row figures show the leg movement for the optimization result with "objective function modification". It is clear that the movement on the second row figures corresponds to a proper protraction. The figures in the last row of Fig. 4 show the main cost function values (nominal cost), whose parameters are given by the *Step 6* of Table 2. The dashed lines show the values for "optimization with single objective function", and the solid lines show the values for "optimization with modified objective function". The most left one of these depict the change of the main objective function (nominal cost). The ultimate aim of both optimizations is to minimize this function. As expected the cost decreases continuously in the case of "optimization with single objective function". In the case of "optimization with objective function modification" the nominal cost increases and decreases with the modifications in the cost function. However, ultimately when the nominal cost function of *Step 6* is used, the result is considerably lower than the one with "optimization with single objective function". The middle figure in the row depicts the value of $h(\bar{x}(t_f),t_f)$, which is related with the achievement of final state requirements with the d_h value in *Step 6.* A lower value of this function means that the final state requirements are attained better. It is seen that this value is lower in the "optimization with objective function modification" compared to the case of "optimization with single objective function". The right most figure in the bottom row of Fig. 4 depicts the value of $\int_{t_0}^{t_f} g(\bar{x}(t),\bar{u}(t),t)dt$ which signifies the sum of torque squares with the d_g value in *Step 6*. The difference for this value between the two cases is striking. In the case of

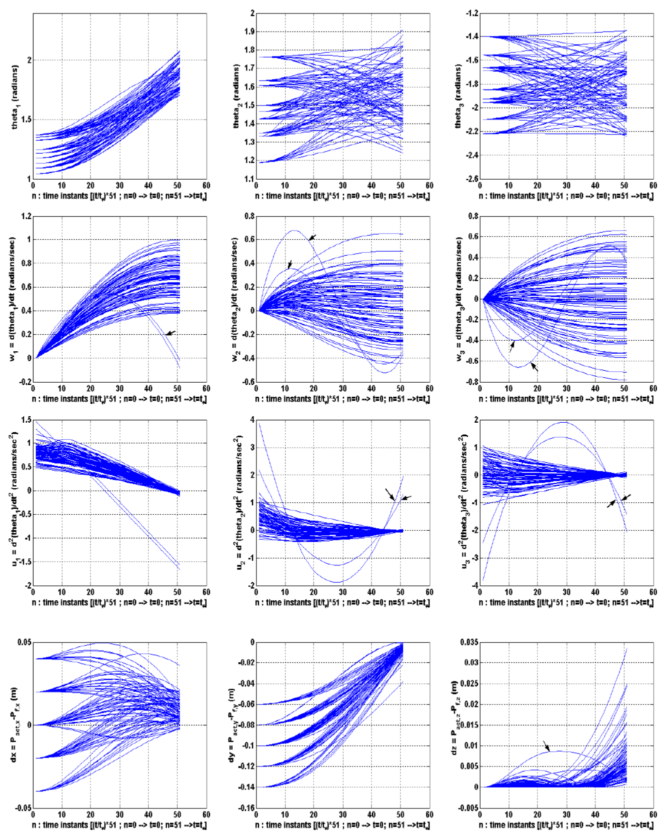
Fig. 2. Results of "optimization with single objective function". Rows from top to bottom: Joint angles, joint angle velocities, joint angle accelerations, tip point position errors.

“optimization with objective function modification” it is possible to observe a step by step decrease of this value to much more lower values compared to the one in “optimization with single objective function”.

The “optimization with single objective function” suffers from the local optimums which mostly lead to infeasible trajectories. With the “optimization with objective function modification” it is possible to jump over these local optimums and generate more feasible and efficient trajectories. With the six steps and the associated cost parameters mentioned in Table 2, the “optimization with objective function modification” was successful to generate feasible and more efficient trajectories for the 79 cases of the 81 optimizations. These results, however, are sure not the global optimums, since the optimization is still based on the gradient information. A problem related to “optimization with objective function modification” is the long time required for the termination. As can be seen in Fig. 4 “optimization with objective function modification” necessitates around 200 iterations which is quite much for a gradient based optimization. However, the improvement is considerable and leads to satisfactory results to be used in controller design. The long time of termination did not create a problem for this work since the optimization is performed only to create a training set to be used in the design of protraction controller.

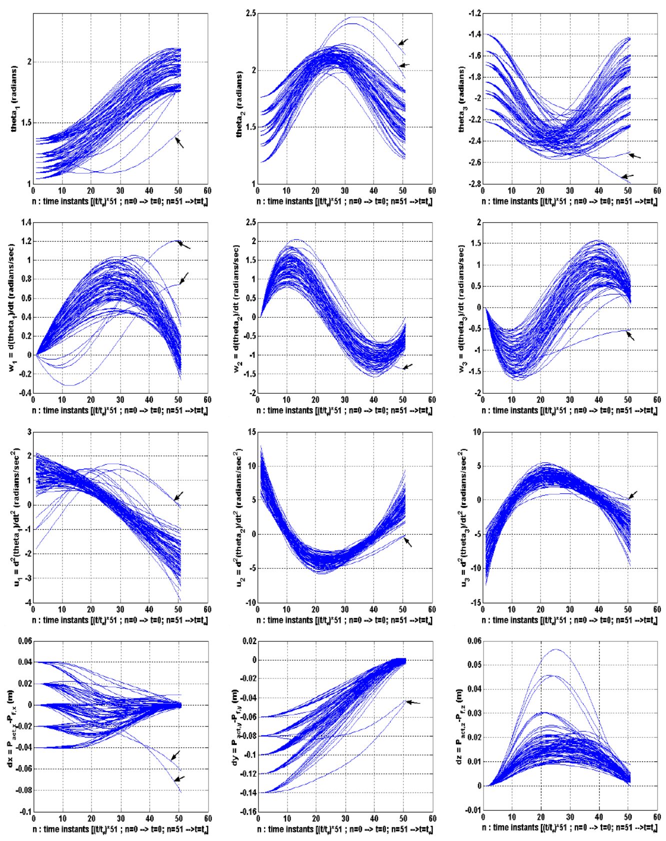
Fig. 3. Results of “optimization with objective function modification". Rows from top to bottom: Joint angles, joint angle velocities, joint angle accelerations, tip point position errors.

4. GENERALIZATION OF OPTIMAL TRAJECTORIES WITH INTERPOLATION

4.1 Manipulation of Optimal Trajectories

The optimization is capable of decreasing the torque square sum with approaching the tip point to the final position; however this approaching does not result in exact placement of the tip point to the desired final position. Most of the time slight, but sometimes considerable, position differences occur between the actual and desired final tip point positions. For the interpolation purposes, this is an improper situation regarding to the data set. In order to overcome this final position deviation, the optimized trajectories are modified to bring the final tip points to the desired positions. This is performed by adding the difference vector between the actual and desired final positions to all position vectors, multiplied by a weight decreasing towards the start of trajectory (21). This manipulation effects the most successful optimizations in a negligible amount. However, the less successful optimizations, in which the final point position is not close to the desired position, are manipulated in a considerable amount. The effect of optimal trajectory manipulation is shown in Fig. 6 for two sample initial-final position pairs.

$$\bar{P}_{mod}(t_k)=\bar{P}_{op}(t_k)+(\bar{P}_f-\bar{P}_{mod}(t_{51}))\frac{k}{51}\ , \quad k=0,1,2,...,51. \tag{21}$$

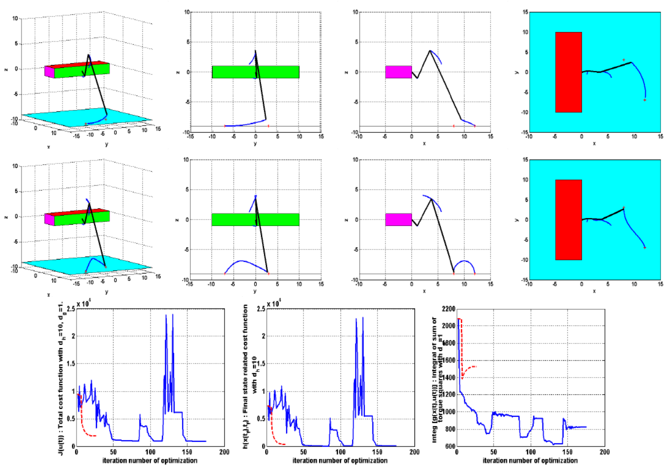
Fig. 4. Results of "optimization with single objective function" (the first row and the dashed lines in the last row) and "optimization with objective function modification" (the second row and solid lines in the last row). (Initial tip point position: [12, -7, -9] (cm); final tip point position: [8, 2, -9] (cm).)

4.2 RBFNN for Interpolation:

In order to generalize the optimal trajectories for the continuous domain of anterior and posterior tip point position regions, it is necessary to construct an interpolation algorithm, which generates the trajectory for any given initial and final tip point positions. For this purpose a RBFNN with pure-linear output neurons is a proper choice (Fig. 5). The inputs of this interpolating network are the coordinates of the initial and final tip point positions, which makes up to 6 input variables. The outputs are the tip point positions corresponding to the 51 discretized instances throughout the protraction period, which makes up to a total of 153 output variables. The inner neurons in an RBFNN correspond to the representatives of clusters of the input space. The weights between the input and inner neurons determine the location of the representatives in the input space. Given any input, its distance to the representatives are calculated, and the inner neurons are activated with an inverse proportion to this distance. Namely, the representative which is closest to the input is activated the most. The weights between the inner and output neurons generate the output vectors corresponding to the representative neurons. Since these weights are multiplied with the activation of inner neurons, the output is a weighted sum of the effect of all representatives.

Training of the RBFNN according to the optimized trajectories is a matter of generating the inner neurons, namely the cluster representatives, and the weights corresponding to those. The number of neurons to be generated is a matter of design preference, in which data storage capabilities should be considered. In this work the function, *newrb(),* in the neural network toolbox of MATLAB is utilized for training the RBFNN with 30 inner neurons. The output values of the RBFNN are given in (22). Two trajectories produced by the interpolation of the RBFNN for two sample initial-final position points from the training set are shown in Fig. 6.

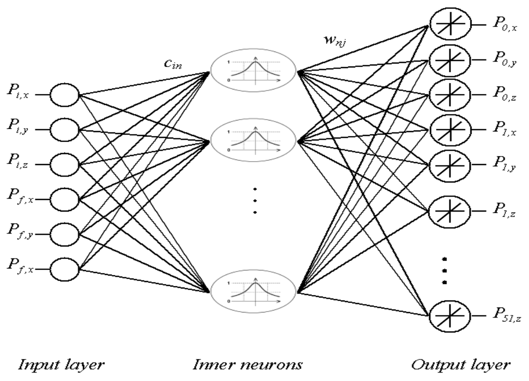

Fig. 5. Trajectory interpolating RBFNN.

$$[output]_j=\sum_{n=1}^{30} w_{nj}\,a_n=\sum_{n=1}^{30} w_{nj}\,gaussian\Big(\sum_{i=1}^{6}\big|[input]_i-c_{in}\big|\Big) \quad (22)$$
$$i=1,2,\ldots,6\ ;\ n=1,2,\ldots,30\ ;\ j=1,2,\ldots,153$$

In Fig. 7, three slides are given to show the Robot-EA308 while its right middle leg is protracting following the path generated by the controller. The left most figure shows how the leg is pulled towards the body while the tip point is raised up. In the middle and right-most figures the whole protraction can be seen from different perspectives. These results, and also the joint angle trajectories in Fig. 3, reveal that the energy optimal protraction has the following character: The leg is pulled towards the body while the tip point is raised up; then it is stretched out while the tip point is going down to approach to the destination. This behaviour is quite different from the conventionally adapted protraction movements in which the leg is first stretched out while raising the tip point, and then pulled towards the body while the tip point is approached down to the destination.

6. CONCLUSIONS

In this work the protraction movement of a three-joint robot leg is handled. In multi legged system applications mostly some limited protraction behaviours are adopted based on intuitive feeling of physical behaviour. Those applications limit the freedom of leg placements since the system can use only the preplanned structures. Moreover, mostly the adopted behaviours are not optimized considering energy consumption. The optimization presented in this work creates energy optimal trajectories for any given initial-final tip point position pairs. With the generalization of the optimized trajectories it is shown that the system does not need to be limited to some specific protraction behaviours, but it can handle any protraction once the initial-final tip point positions are given in the range of design. The result of this work shows that an energy optimal protraction is characterized by pulling the leg towards the body, rather than stretching out, in the rising phase of the movement.

The optimization is based on the gradient descent algorithm, with the Hamiltonian formulation for optimal control problems. The conventional application suffers from sticking to infeasible and inefficient local optimums. To overcome this, the approach of "optimization with objective function modification" is introduced. The results reveal that this

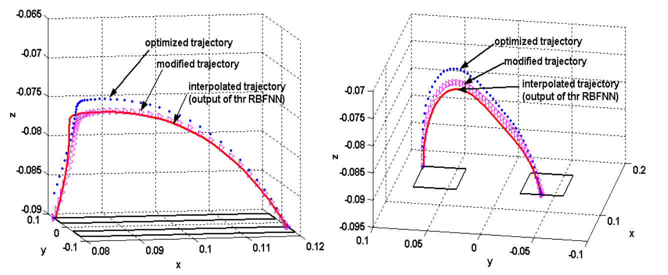

Fig. 6. The trajectories produced as a result of the interpolation with the RBFNN for two trained input of initial-final tip point positions. (unit: m)

method is successful to overcome both feasible and infeasible local optimum solutions of conventional "optimization with single objective function". The optimal trajectories are generalized to a continuous space by interpolation using RBFNN.

For the optimization algorithm a considerable drawback is the long duration for the termination. This is because a full optimization is performed for every step, namely for all the objective functions described by the steps. This was not a problem for the work presented here, because the optimization is performed once and the recorded results are used in generalizations. However, this may be a significant problem in some applications in which fast responses of optimization are required. The improvement of the idea of "optimization with modified objective function" in the direction of speeding up the algorithm remains as a future work. A crucial attempt may be to modify the objective function not in steps of sequential optimizations, but continuously throughout a single optimization. This might be performed by some intelligent and continuous modification of the objective function based on the values of a nominal objective function and its gradient.

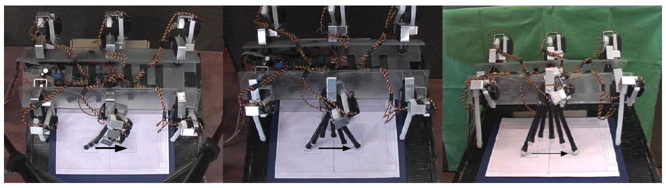

Fig. 7. Protraction of the right middle leg of robot EA308, following the trajectory generated by the controller. The initial and final tip point positions are [0.11, -0.06, -0.09] and [0.11, 0.06, -0.09], respectively.

REFERENCES

Bobrow, J.E., Martin, B., Sohl, G., Wang, E.C., Park, F.C, Kim, K. (2001). Optimal robot motions for physical criteria. *Journal of Robotic Systems*, **18 (12)**, 785-792.

Chettibi, T., Lehtihet, H.E., Haddad, M., and Hanchi, S. (2004). Minimum cost trajectory planning for industrial robots. *European Journal of Mechanics A/Solids*, **23**, 703-715.

Cruse, H., Kindermann, T., Schumm, M., Dean, J. and Schmitz, J. (1998). Walknet-a biologically inspired network to control six legged walking. *Neural Networks*, **11**,1435-1447.

Dürr, V., Schmitz, J., and Cruse, H. (2004). Behaviour-based modeling of hexapod locomotion: linking biology and technical application. *Arthropod Structure & Development*, **33**, 237-250.

Erden, M.S. and Leblebicioğlu, K, (2004a). Fuzzy controller design for a three joint robot leg in protraction phase - an optimal behavior inspired fuzzy controller design. In: *Proceedings of the First International Conference On Informatics In Control, Automation And Robotics*, Setúbal, Portugal, **2**, 302-306.

Erden, M.S., Leblebicioğlu, K. and Halıcı, U. (2004b). Multi-agent system based fuzzy controller design with genetic tuning for a service mobile manipulator robot in the hand-over task. *Journal of Intelligent and Robotic Systems*, **38**, 287-306.

Erden, M.S. and Leblebicioğlu, K. (2007). Analysis of wave gaits for energy efficiency. *Autonomous Robots,* **23**, 213-230.

Espenschied, K.S., Quinn, R.D., Beer, R.D. and Chiel, H.J. (1996). Biologically based distributed control and local reflexes improve rough terrain locomotion in a hexapod robot. *Robotics and Autonomous Systems*, **18**, 59-64.

Ferrell, C. (1995). A comparison of three insect-inspired locomotion controllers. *Robotics and Autonomous Systems*, **16**,135-159.

Frangos, C. and Yavin, Y. (2001). Control of a three-link manipulator with inequality constraints on the trajectories of its joints. *Computers and Mathematics with Applications*, **41**, 1562-1574.

Fu, K.S., Gonzalez, R.C. and Lee, C.S.G. (1987). *Robotics*, McGraw-Hill, Inc.

Garg, D.P. and Kumar, M. (2002). Optimization techniques applied to multiple manipulators for path planning and torque minimization. *Engineering Applications of Intelligence*, **15**, 241-252.

Huang, Q.J. and Nomani, K. (2003). Humanitarian mine detecting six-legged walking robot and hybrid neuro walking control with position/force control. *Mechatronics*, **13**, 773-790.

Ilg, W. and Bernes, K. (1995). A learning architecture based on reinforcement learning for adaptive control of the walking machine LAURON. *Robotics and Autonomous Systems*, **15**, 321-334.

Ilg, W., Bernes, K., Mühlfriedel, and T., Dillman, R. (1997). Hybrid learning concepts based on self-organizing neural networks for adaptive control of walking machines. *Robotics and Autonomous Systems*, **22**, 317-327.

Kirk, D.E (1970). *Optimal Control Theory – An Introduction*, Prentice-Hall Inc., Englewood Cliffs, New Jersey.

Liu, J.F. and Abdel-Malek, K. (2000). Robust control of planar dual-arm cooperative manipulators. *Robotics and Computer-Integrated Manufacturing*, **16 (2-3)**, 109-120.

Preumont, A., Alexandre, P., Doroftei, I. and Goffin, F. (1997). A conceptual walking vehicle for planetory exploration. *Mechatronics*, **7 (3)**, 287-296.

Saramago, S.F.P. and Stefen Jr., V. (1998). Optimization of the trajectory planning of robot manipulators taking into account the dynamics of the system. *Mech. Mach. Theory*, **33 (7)**, 883-894.