Neural Network Application to Optimal Control of Nonlinear Systems

J. Kasac[1], B. Novakovic[2]
[1]*Institute for Defense Studies, Research and Development, Zagreb, Croatia.*
[2]*Faculty of Mechanical Engineering and Naval Architecture, University of Zagreb, Croatia.*

Abstract

This paper presents the derivation of the numerical algorithm for optimal control of nonlinear multivariable systems with control and state vectors constraints. The algorithm derivation is based on the backpropagation-through-time (BPTT) algorithm which is used as a learning algorithm for recurrent neural networks. This approach is not based on Lagrange multiplier techniques and the calculus of variations. The derived algorithm is used for the control of the cooperative work of two robots with two degrees of freedom. The main problem is the determination of the control vectors of robots for the transfer of rigid load from the initial state to the final one in a fixed time while maintaining constant distance between robot hands and avoiding the cross-section of the robot hands.

1 Introduction

Optimal control of the non-linear robot model with a defined optimal criterion still presents a relatively difficult task. The problem becomes more complex when two or more robots work in cooperation on a common task sharing workspace, time, constraints, and the cost function.

The cooperative robots control has several specific requests which makes it difficult. First, the dynamic influence of robots on each other through the load must be avoided if we want to decouple the overall system dynamics. In other words, there is a need to avoid force transmission on the load. If this request is not fulfilled, then we cannot independently use particular robot models.

The second request is load transfer within a given time interval subject to the control vector constrains (Canny [1], Bien [2]). The standard trajectory planning, which independently determines a robot's paths, cannot guarantee achieving the final state within a given terminal time. The third, very important request is collision avoidance (Chang [3], O'Donnell [4]).

There are many methods which treat separately above-mentioned requirements. A natural way to meet these requirements is the optimal control formulation of the cooperative robots control problem. However, the optimal control of nonlinear systems with control and state vector constraints presents a relatively difficult task, particularly in case of the state vector constraints which can be very complicated in robotics control. Therefore, various methods of numerical solution of the optimal control problem have been developed (Bryson [5], Fan [6], Sage [7]).

A solution to the computational complexity of numerical optimal control algorithms are algorithms providing a high level of parallelism of computational tasks executions. The neural networks learning algorithms present an example of this kind of algorithms. In this paper we use the backpropagation-through-time algorithm (Werbos [8]), which is a learning algorithm for recurrent neural networks (Pearlmutter [9], Baldi [10]).

The BPTT algorithm presents a time generalization of the backpropagation algorithm (Rumelhart [11]), where the error (which is minimised) is given along a specified time interval. The essence of the backpropagation algorithm is a simple and precise calculation of derivations of the cost function in relation to system parameters and the adjustment of parameters in line with those derivations in only one transfer through the network. The BPTT algorithm expands this method through application to dynamic systems for which direct calculation of derivations can be very complex. A solution to this problem lies in the chain rule for order derivations (Wan [12], Piche [13]), which results in error backpropagation, i.e. parameter adjustment backward in time.

2 Optimal Control Problem with Constraints

It is considered a nonlinear discrete optimal control problem with a fixed beginning and terminal time. A continuous optimal control problem can be approximated by discrete one using the first difference approximation.

2.1 Problem Formulation

The problem consists of determining the control vector $\mathbf{u}(t)$ with the aim to minimize the cost function

$$J_0 = \min_{\mathbf{u}(i)} \tau \sum_{i=0}^{N-1} \hat{F}(\mathbf{x}(i), \mathbf{u}(i)), \tag{1}$$

subject to the constraints defined by the plant equations

$$\mathbf{x}(i+1) = \mathbf{f}(\mathbf{x}(i), \mathbf{u}(i)), \tag{2}$$

and the initial and final condition of the state vector

$$\mathbf{x}(0) = \mathbf{x}_0, \qquad \mathbf{x}(N) = \mathbf{x}_f, \tag{3}$$

subject to the control and state vector inequality constraints

$$\mathbf{g}(\mathbf{x}(i), \mathbf{u}(i)) \geq \mathbf{0}, \tag{4}$$

and the equality constraints

$$\mathbf{h}(\mathbf{x}(i), \mathbf{u}(i)) = \mathbf{0}, \tag{5}$$

for $i = 0, 1, ..., N-1$, where N is the number of sampling intervals, $\tau = (t_f - t_0)/N$ is the sampling interval, where $\mathbf{x}(i)$ is n-dimensional state vector, $\mathbf{u}(i)$ is m-dimensional control vector, $\mathbf{g}(\cdot)$ is N_p-dimensional vector function of inequality constraints, and $\mathbf{h}(\cdot)$ is N_q-dimensional vector function of equality constraints.

2.2 Penalty Method Approach

The next step is the expansion of the cost function (1) by adding penalty functions for constraints

$$J = J_0 + J_1 + J_2 + J_3, \tag{6}$$

where J_1 is the penalty function for the final boundary condition, J_2 is the penalty function for inequality constraints (4), J_3 is the penalty function for equality constraints (5).

The introduction of the penalty functions provides a transformation of the problem from (1) to (5) to the next form

$$J = \min_{\mathbf{u}(i)} \left(\tau \sum_{i=0}^{N-1} F(\mathbf{x}(i), \mathbf{u}(i)) + J_1 \right), \tag{7}$$

$$\mathbf{x}(i+1) = \mathbf{f}(\mathbf{x}(i), \mathbf{u}(i)), \qquad \mathbf{x}(0) = \mathbf{x}_0, \tag{8}$$

where the sum on the right side of the above-mentioned expression includes the penalty function for equality and inequality constraints. The problem is thus reduced on only one type of constraints - plant equations. Concrete forms of the penalty functions will be shown on the example of cooperative robots work.

The gradient descent algorithm is used for minimizing the cost function

$$\mathbf{u}^{(l+1)}(i) = \mathbf{u}^{(l)}(i) - \eta \frac{\partial J}{\partial \mathbf{u}^{(l)}(i)} \tag{9}$$

where $i = 0, 1, \ldots, N-1$, $l = 1, 2, \ldots, M$, while η is the convergence coefficient, index l represents the l-th iteration of the gradient algorithm and M is the number of iterations of the gradient algorithm.

2.3 Calculation of the gradient

In this subsection, basic steps in the calculation of the gradient of the cost function (7) subject to the constraints (8) will be described very briefly .

The gradient of the cost function (7) in l-th iteration is

$$\frac{\partial J}{\partial u_k(j)} = \tau \sum_{i=0}^{N-1} \frac{\partial F(i)}{\partial u_k(j)} + \frac{\partial J_1}{\partial u_k(j)} \tag{10}$$

where $F(i) \equiv F(\mathbf{x}(i), \mathbf{u}(i))$.

Only addends for which $i \geq j$ remains on the right side of the eqn (10),

$$\tau \sum_{i=0}^{N-1} \frac{\partial F(i)}{\partial u_k(j)} = \tau \frac{\partial F(j)}{\partial u_k(j)} + \tau \sum_{i=j+1}^{N-1} \frac{\partial F(i)}{\partial u_k(j)} \tag{11}$$

The addends on the right side of the previous equation depend on $u_k(j)$ implicitly through $\mathbf{x}(i)$ for $i > j$, and it follows that

$$\frac{\partial F(i)}{\partial u_k(j)} = \sum_{r=1}^{n} \frac{\partial F(i)}{\partial x_r(i)} \frac{\partial x_r(i)}{\partial u_k(j)} \tag{12}$$

The next step is the calculation of partial derivations $\frac{\partial x_r(i)}{\partial u_k(j)}$ on the right side of the eqn (12). On the basis of eqn (8), it is obtained

$$\frac{\partial x_r(j+1)}{\partial u_k(j)} = \frac{\partial f_r(j)}{\partial u_k(j)}, \tag{13}$$

$$\frac{\partial x_r(i)}{\partial u_k(j)} = \sum_{p=1}^{n} \frac{\partial f_r(i-1)}{\partial x_p(i-1)} \frac{\partial x_p(i-1)}{\partial u_k(j)}, \tag{14}$$

for $r = 1, 2, \ldots, n$, $k = 1, 2, \ldots, m$, $j = 0, 1, \ldots, N-1$, $i = j+2, \ldots, N-1$, where $f_r(j) \equiv f_r(\mathbf{x}(j), \mathbf{u}(j))$. The above-mentioned iterative expression is the chain rule for ordered derivations.

By manipulating with the expressions from eqn (10) to eqn (14), it is obtained a backward in time iterative algorithm for calculation of the gradient of the cost function in relation to the control vector.

The basic characteristic of this algorithm is derivation without using the calculus of variations and Lagrange multiplier techniques. This fact provides an obvious geometrical interpretation of the gradient algorithm which, on the other hand, provides a rough approximation of the gradient of the penalty function without the accuracy of the optimal solution being lost. This will be demonstrated on the example of cooperative work of two robots.

3 Optimal Control of the Cooperative Robots Work

In this section, the derived algorithm will be used to the control of the cooperative work of two robots with two degrees of freedom.

3.1 Dynamics of the robot with two degrees of freedom

The non-linear dynamic model of the manipulator with two degrees of freedom (Heiman [14]) is presented through cylindrical coordinates in the form of

$$\begin{bmatrix} M_{11}(\mathbf{q}) & 0 \\ 0 & M_{22}(\mathbf{q}) \end{bmatrix} \begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix} + \begin{bmatrix} N_1(\mathbf{q}, \dot{\mathbf{q}}) \\ N_2(\mathbf{q}, \dot{\mathbf{q}}) \end{bmatrix} = \begin{bmatrix} P_1(t) \\ P_2(t) \end{bmatrix}, \qquad (15)$$

where

$$\begin{aligned} M_{11} &= I_1 + I_2 + (m+M)q_2^2 + 2Maq_2 + Ma^2, \\ M_{22} &= m + M, \\ N_1 &= 2[(m+M)q_2 + Ma]\dot{q}_1\dot{q}_2, \\ N_2 &= -[mq_2 + M(a+q_2)]\dot{q}_1^2, \end{aligned}$$

and where $\mathbf{q} = [q_1\ q_2]^T$ stands for cylindrical coordinates of the center of the mass of link 3 (Figure 1.a), M is total mass (manipulator hand and load), m is link mass, I_1 is the total moment of inertia of links 1 and 2 in relation to axis Z, I_2 is the moment of inertia of link 3 in relation to the axis which is parallel to axis Y which goes through point S, a is the distance between the center of mass M and point S. $P_1(t)$ stands for the control moment of rotation freedom of motion q_1, while $P_2(t)$ is the control force of the translation freedom of motion q_2. The numerical values of the above-mentioned parameters are $M = 50kg$, $m = 97kg$, $I_1 + I_2 = 193kgm^2$, $a = 1.1m$, $P_1^{\max} = 600Nm$, $P_2^{\max} = 500N$, where $P_1^{\max}$ is the maximum allowed moment and $P_2^{\max}$ is the maximum allowed force.

If we transform the mentioned system of the second-order differential equations into the system of the first-order differential equations, then the following coordinate transformation will be introduced:

$x_1 = q_1$, $x_2 = \dot{q}_1$, $x_3 = q_2$, $x_4 = \dot{q}_2$, $u_1 = P_1$, $u_2 = P_2$.

The same dynamic eqns (15) are used for the second robot which is located $d = 3\ m$ from the first one (Figure 1.b). The coordinates of the second robot are marked with q_3, q_4 while control variables are marked with u_3, u_4.

3.2 Problem formulation

Cooperative work is mutual coordinate work on a common defined goal. The cooperation of two robots is useful when a single robot, i.e. manipulator, cannot manipulate a load that is too heavy or too large. This example considers the problem of rigid load transfer from one place to another by using two robots (or two arms). To avoid the dynamic coupling of the robots, the distance between the robots' hands (i.e. joints between the robots hands and the load) must be constant and the same as the load length.

The problem considered here represents the transformation of the initial robots state from

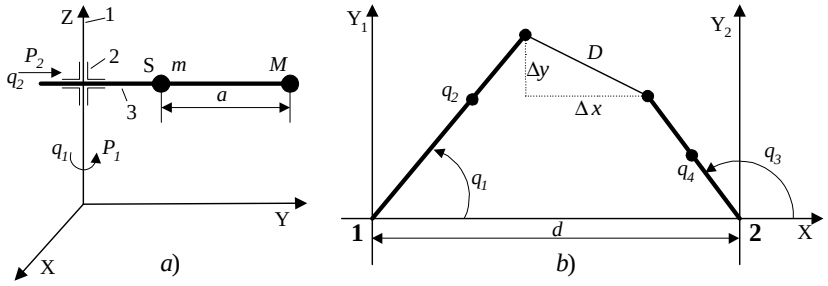

Figure 1: Robot with two degrees of freedom - rotation and translation.

$x_1(0) = 0\ rad$, $x_2(0) = 0\ rad/s$, $x_3(0) = 0.4\ m$, $x_4(0) = 0\ m/s$,
$x_5(0) = \pi\ rad$, $x_6(0) = 0\ rad/s$, $x_7(0) = 0.2\ m$, $x_8(0) = 0\ m/s$,
into the final state
$x_1(t_f) = \pi/2\ rad$, $x_2(t_f) = 0\ rad/s$, $x_3(t_f) = 0.4\ m$, $x_4(t_f) = 0\ m/s$,
$x_5(t_f) = 2.62\ rad$, $x_6(t_f) = 0\ rad/s$, $x_7(t_f) = 2.35\ m$, $x_8(t_f) = 0\ m/s$,
including the control constraints

$$|u_k(i)| \leq u_k^{\max}, \qquad k = 1, ..., 4, \quad i = 0, ..., N-1, \tag{16}$$

where $u_1^{\max} = u_3^{\max} = 600\ Nm$, $u_2^{\max} = u_4^{\max} = 500\ N$ in the fixed time $t_f = 2s$, on condition the distance between the manipulators' hands $r(i)$ in i-th time interval and load length $D = 0.2\ m$ are equal

$$r(i) = D, \qquad i = 0, ..., N, \tag{17}$$

where

$$r(i) = \sqrt{\Delta x_i^2 + \Delta y_i^2}, \tag{18}$$

and

$$\begin{aligned} \Delta x_i &= d + (x_7(i) + a)\cos x_5(i) - (x_3(i) + a)\cos x_1(i), \\ \Delta y_i &= (x_7(i) + a)\sin x_5(i) - (x_3(i) + a)\sin x_1(i). \end{aligned} \tag{19}$$

On the basis of the above mentioned problem formulation, a set of penalty functions for constraints are obtained. The penalty function for the final boundary condition is

$$J_1 = K_B \sum_{k=1}^{8} (x_k(N) - x_k(t_f))^2, \tag{20}$$

where K_B is the coefficient of the penalty function. The penalty function for control vectors inequality constraints (16) is

$$\begin{aligned} J_2 = \hat{K}_V \sum_{i=0}^{N-1} \sum_{k=1}^{4} [&(u_k^{\max} - u_k(i))^2 H^-(u_k^{\max} - u_k(i)) \\ + &(u_k^{\max} + u_k(i))^2 H^-(u_k^{\max} + u_k(i))], \end{aligned} \tag{21}$$

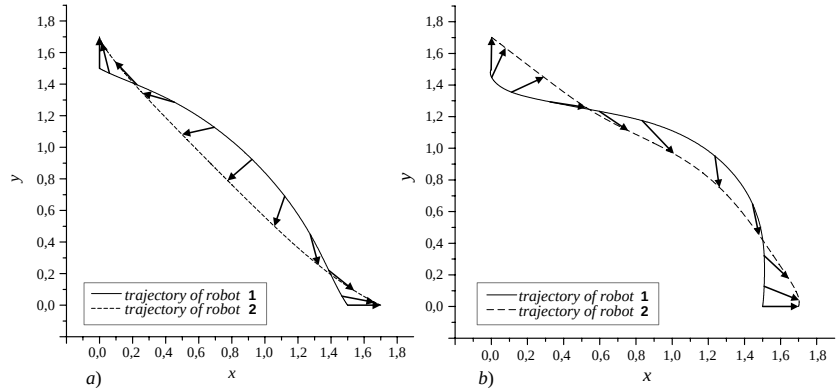

Figure 2: The trajectories of the robots' hands in the plane x-y with positions of load in different time intervals a) without the condition of avoiding the cross-section of the robots' hands, and b) with the condition of avoiding the cross-section of the robots' hands.

where $H^-(z)$ is the Heaviside step function defined as

$$H^-(z) = \begin{cases} 0, \text{if } z \geq 0 \\ 1, \text{if } z < 0 \end{cases} \tag{22}$$

while $\hat{K}_V$ is the coefficient of the penalty function for inequality constraints. The penalty function for equality constraints (17) is

$$J_3 = K_E \sum_{i=0}^{N} (r(i) - D)^2, \tag{23}$$

where $\hat{K}_E$ is the coefficient of the penalty function of equality constraints.

The solution shown in Figure 2.a is obtained by using the cost function $J = J_1 + J_2 + J_3$. However, the cross-section of the robots' hands occurs during the time period from $t_1 = 0.84\ s$ to $t_2 = 1.24\ s$. In an actual situation it will be a collision of the robots' hands in time $t_1 = 0.84\ s$. Therefore, the condition of avoiding the cross-section of the robots hands is necessary to include in the cost function.

From the viewpoint of the coordinate system of the first robot, the cross-section of the robots' hands' lines is in the point $(x_p(i), y_p(i))$, where

$$x_p(i) = d\frac{\tan q_3(i)}{\tan q_3(i) - \tan q_1(i)}, \quad y_p(i) = d\frac{\tan q_1(i) \cdot \tan q_3(i)}{\tan q_3(i) - \tan q_1(i)}. \tag{24}$$

The projections of this point on the coordinate axes which pass along the robots hands are

$$\begin{aligned} q_2^{cs}(i) &= x_p(i)\cos q_1(i) + y_p(i)\sin q_1(i), \\ q_4^{cs}(i) &= (x_p(i) - d)\cos q_3(i) + y_p(i)\sin q_3(i). \end{aligned} \tag{25}$$

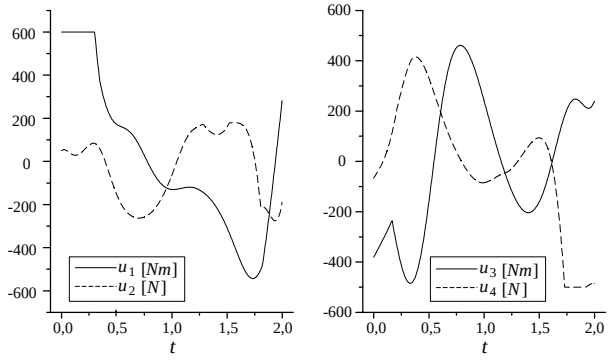

Figure 3: Time dependence of robots control functions.

A sufficient condition for avoiding the cross-section of the robots' hands is to satisfy whichever of the following state vectors inequality constraints

$$\begin{array}{ll} g_1(i) = q_2^{cs}(i) - a - q_2(i) \geq 0, \\ g_2(i) = q_4^{cs}(i) - a - q_4(i) \geq 0, \\ g_3(i) = -q_2^{cs}(i) \geq 0, \\ g_4(i) = -q_4^{cs}(i) \geq 0, \end{array} \tag{26}$$

for $i = 0, 1, ..., N$.

the penalty function for the state vectors inequality constraints is obtained on the basis of inequality (26),

$$J_2^S = K_S \sum_{i=0}^{N} \sum_{k=1}^{4} g_k^2(i) \prod_{l=1}^{4} H^-(g_l(i)). \tag{27}$$

It is obvious that the gradient calculation of the above-mentioned penalty function is very complicated task. The gradient approximation of the penalty function means a certain deviation from the exact direction of the overall cost function gradient. However, the approximation of the gradient does not means the approximation of the optimal solution but only slower convergence toward the optimal solution.

Namely, the following rough approximation of the penalty function gradient is used

$$\frac{\partial J_2^S}{\partial q_2(j)} = \frac{\partial J_2^S}{\partial q_4(j)} = K_S \prod_{l=1}^{4} H^-(g_l(j)), \quad \frac{\partial J_2^S}{\partial q_1(j)} = \frac{\partial J_2^S}{\partial q_3(j)} = 0, \tag{28}$$

The gradient algorithm with the constant coefficient of convergence is used. The values of the constants are $N = 1000$, $M = 160000$, $\eta = 3000$, $K_B = 100$, $K_V = 0.01$, $K_E = 1.0$, $K_S = 0.5$.

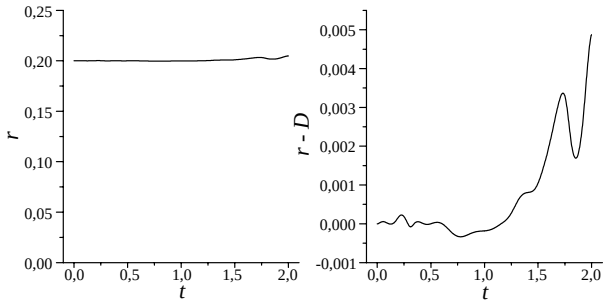

Figure 4: The distance between the manipulator hands r, and difference between this distance and load length D.

By extending the previous cost function with the penalty function (27), i.e. approximating the penalty function gradient by eqns (28), a solution shown in Figure 2.b is obtained. Figure 3 shows time dependence of robot control functions, while Figure 4 shows time dependence of the distance between the manipulators' hands r, and the difference between this distance and load length D. Because of the existing error (about 2.5% for $M = 160000$ iteration), there is a need for an appropriate level of elasticity of the joint between the robot hands and the load.

4 Conclusions

This paper has shown a possibilities of application of the backpropagation-through-time algorithm to optimal control of cooperative robots' work. In this problem must be satisfy the set of relatively complex constraints on the robots' state vectors. The penalty function gradients for those constraints can be extremely complicated. Therefore, it is significant that good results can be obtained through a rough approximation of those derivatives, but only by the expense of a decreased convergence speed. Bearing in mind the specific geometrical interpretation of the gradient algorithm, we can conclude that the gradient algorithm is very effective in solving the constraint optimal control problem with the mentioned type of approximations.

Since proving the existence of a solution to the optimal control constraints problem is generally a very difficult problem, treating constraints with penalty functions is very useful also as an indication of the solubility of the problem. If the penalty function does not converge to zero (for a stable algorithm with a constant convergence coefficient), that is a definite sign that the problem does not have a solution within the given constraints.

The speed of convergence of the algorithm does not depend much on the order of the system as much as it depends on the number of constraints, i.e. penalty functions.

References

[1] Canny, J. F., *The Complexity of Robot Motion Planning,* MA: MIT Press, Cambridge, 1988.

[2] Bien, Z. & Lee, J., A minimum-time trajectory planning method for two robots, *IEEE Trans. Robot. Automat.*, **8(3)**, pp. 414-418, 1992.

[3] Chang, C., Chung, M. J. & Lee, B. H., Collision avoidance of two robot manipulators by minimum delay time, *IEEE Trans. Syst., Man, Cybern.*, **24(3)**, pp. 517-522, 1994.

[4] Shin, K. G. & Zheng, Q., Minimum-time collision-free trajectory planning for dual-robot systems, *IEEE Trans. Robot. Automat.*, **8(6)**, pp. 641-644, 1992.

[5] A. E. Bryson Jr. & Y. Ho, *Applied Optimal Control,* Blaisdell, New York, 1969.

[6] L. T. Fan & C. S. Wang, *The Discrete Maximum Principle,* John Wiley, New York, 1964.

[7] A. P. Sage & C. C. White, *Optimum Systems Control,* Prentice-Hall, New Jersey, 1977.

[8] Werbos, P. J., Backpropagation through time: What it does and how to do it, *Proc. IEEE*, **78(10)**, pp. 1550-1560, 1990.

[9] Pearlmutter, B. A., Gradient Calculations for Dynamic Recurrent Neural Networks: A Survey, *IEEE Trans. on Neural Networks*, **6(5)**, pp. 1212-1228, 1995.

[10] Baldi, P., Gradient Descent Learning Algorithm Overviev: A General Dynamical Systems Perspective, *IEEE Trans. on Neural Networks*, **6(1)**, pp. 182-195, 1995.

[11] Rumelhart, D. E. & McClelland, J. L., *Parallel Distributed Processing,* Vol. 1., MA: The MIT Press, 1986.

[12] Wan, E. & Beaufays, F., Diagrammatic Derivation of Gradient Algorithms for Neural Networks, *Neural Computation*, **8(1)**, pp. 182-201, 1996.

[13] Piche, S. W., Steepest Descent Algorithms for Neural Network Controllers and Filters, *IEEE Trans. on Neural Networks*, **5(2)**, pp. 198-212, 1994.

[14] Heiman, B., Loose, H. & Schuster G., Contribution to optimal control of an industrial robot, 4^{th} *CISM-IFTOMM Symp. on Theory and Pract. of Rob. and Manip.*, Zaborow, Poland, pp. 211-219, 1981.