ITM Web of Conferences 6, 02004 (2016)
DOI: 10.1051/itmconf/20160602004
© Owned by the authors, published by EDP Sciences, 2016

Automatic approach to stabilization and control for multi robot teams by multilayer network operator

Askhat Diveev[1, 2, a] and Elizaveta Shmalko[1]

[1] *Federal Research Center "Computer Science and Control" of Russian Academy of Sciences, 119333 Moscow, Russia*
[2] *Peoples' Friendship University of Russia, 117198 Moscow, Russia*

Abstract. The paper describes a novel methodology for synthesis a high-level control of autonomous multi robot teams. The approach is based on multilayer network operator method that belongs to a symbolic regression class. Synthesis is accomplished in three steps: stabilizing robots about some given position in a state space, finding optimal trajectories of robots' motion as sets of stabilizing points and then approximating all the points of optimal trajectories by some multi-dimensional function of state variables. The feasibility and effectiveness of the proposed approach is verified on simulations of the task of control synthesis for three mobile robots parking in the constrained space.

1 Introduction

The research and development of autonomous multi robot teams become a hot topic in recent years. Multi-robot control poses crucial issues due to inter-agent communications and increased number of agents. The most standard way to design an optimal motion and coordination control for multi robot teams involves constructing stabilization system for each robot in a team about some given position in a state space and finding optimal trajectories so that robots can optimally move with regard to some criteria from any initial position to some given terminal position. The optimal trajectories herein consist of sets of points where robots are stabilized.

There are a lot of solutions to stabilization problem. Specialists apply PID-controllers [1, 2] using deviation of the state vector from the given stabilization point in a space. Potential fields [3] method has also gained popularity recently. In this method a feedback control function is synthesized in order to minimize Euclidean norm of the difference between the value of the state vector of the robot and the point of stabilization. Stabilization can also be provided through Lyapunov functions [4, 5].

Once stabilization is done, different techniques can be applied to provide optimal trajectories. But the main challenge in designing optimal trajectories for a team of robots is to take into account dynamical constraints that guarantee the absence of collisions between agents. Such constraints should be considered separaty or included into quality criteria. Collision avoidance problem can be solved by the method of receding horizon [6] when the moves of robots are computed ahead and the moments of possible collisions are detected. If the robot is going to face a static obstacle then the trajectory of movement is adjusted. If the robot is going to face with another robot then the robots are prioritized by the distance to the termination point, and the robot with the lowest priority stops or commits some manoeuvre to avoid collision.

So we apply different techniques for different parts of the synthesis problem. Another drawback of the existing methods is that they provide a set of optimal trajectories for different possible initial conditions. And in case of a wide initial domain such set of optimal trajectories can occupy a large portion of the memory.

In the present paper we propose a single method of multilayer network operator to feedback control system synthesis. The method belongs to a class of symbolic regression methods and derives from the method of network operator [7]. The method allows automatically constructing such multidimensional function of the robots' state variables that can substitute the whole set of optimal trajectories of robots moving from different initial positions. We illustrated the main features of the method on the example. Simulation results are presented for a team of three mobile robots aiming to move to a given terminal position from eight different initial conditions.

2 Problem statement of control synthesis for multi robot team

Consider a system of similar mobile robots

$$\dot{x}_i = u_{1,i}\cos(\theta_i)\,,\ \dot{y}_i = u_{1,i}\sin(\theta_i)\,,$$

$$\dot{\theta}_i = \frac{u_{1,i}}{L}\mathrm{tg}(u_{2,i})\,,\ i = 1,\dots,N\,, \qquad (1)$$

[a] Corresponding author: aidiveev@mail.ru

where (x_i, y_i) are coordinates of the geometric center of the robot i, θ_i is an angle between the roll axis of the robot i and x-axis of the fixed coordinate system, L is an overall dimensional parameter of robot, $u_{1,i}$, $u_{2,i}$ are components of the control vector of the robot i, N is a number of robots.

Given constraints on control

$$u_1^- \le u_{1,i} \le u_1^+,\ u_2^- \le u_{2,i} \le u_2^+ . \quad (2)$$

Fig.1 shows a layout view of the described mobile robot.

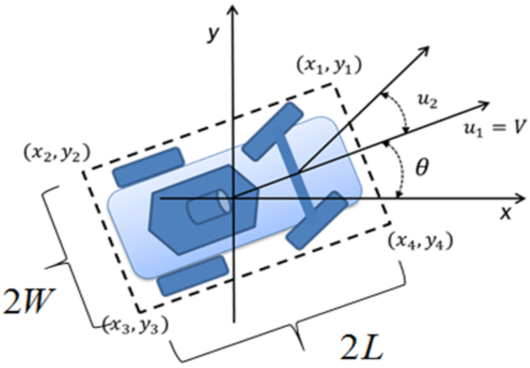

Figure 1. Overall domain of the robot and control.

Here (x_k, y_k) are coordinates of a rectangular domain of the robot, $2W$ is a width of the domain, $2L$ is a length correspondingly.

To check the absence of collisions between robots and with other static geometrical obstacles we need to know coordinates of corners of each robot. To determine the coordinates of coners of the robot i let us rotate the coordinate system by an angle θ_i. Coordinates of the geometric center of the robot i in the rotated coordinate system are defined as follows

$$\tilde{x}_i = x_i \cos(\theta_i) + y_i \sin(\theta_i),$$

$$\tilde{y}_i = -x_i \sin(\theta_i) + y_i \cos(\theta_i) . \quad (3)$$

Thus calculate the coordinates of corners as

$$\tilde{x}_{k,i} = \tilde{x}_i + (-1)^{\lfloor k/2 \rfloor} L,\ \tilde{y}_{k,i} = \tilde{y}_i - (-1)^{\lfloor (k+1)/2 \rfloor} W, \quad (4)$$

$$k = 1,2,3,4.$$

To obtain coordinates of corners of the robot in a fixed coordinate system we do inverse conversion with respect to (2)

$$x_{k,i} = \tilde{x}_{k,i} \cos(\theta_i) - \tilde{y}_{k,i} \sin(\theta_i),$$

$$y_{k,i} = \tilde{x}_{k,i} \sin(\theta_i) + \tilde{y}_{k,i} \cos(\theta_i), \quad (5)$$

$$k = 1,2,3,4.$$

To check if the collision of the robot i with the robot j happens we need to determine the penetration of every corner of the robot j into the domain of the robot i, and vice versa the penetration of every corner of the robot i into the domain of the robot j. We initially calculate the coordinates of corners of the robot j in rotated by θ_i coordinate system

$$\tilde{x}_{k,j}(\theta_i) = x_{k,j} \cos(\theta_i) + y_{k,j} \sin(\theta_i),$$

$$\tilde{y}_{k,j}(\theta_i) = -x_{k,j} \sin(\theta_i) + y_{k,j} \cos(\theta_i), \quad (6)$$

$$k = 1,2,3,4.$$

The corner $(x_{k,j}, y_{k,j})$ of the robot j penetrates the domain of the robot i when the following conditions are met

$$(\tilde{x}_{k,j}(\theta_i) \le \tilde{x}_i + L) \wedge (\tilde{x}_{k,j}(\theta_i) \ge \tilde{x}_i - L) \wedge$$

$$\wedge (\tilde{y}_{k,j}(\theta_i) \le \tilde{y}_i + W) \wedge (\tilde{y}_{k,j}(\theta_i) \ge \tilde{y}_i - W), \quad (7)$$

$$k = 1,2,3,4.$$

For complete collision checking we determine as well if the corners of the robot i penetrates the domain of the robot j.

$$(\tilde{x}_{k,i}(\theta_j) \le \tilde{x}_j + L) \wedge (\tilde{x}_{k,i}(\theta_j) \ge \tilde{x}_j - L) \wedge$$

$$\wedge (\tilde{y}_{k,i}(\theta_j) \le \tilde{y}_j + W) \wedge (\tilde{y}_{k,i}(\theta_j) \ge \tilde{y}_i - W), \quad (8)$$

$$k = 1,2,3,4.$$

Conditions (7) or (8) detect the penetration of the domain of one robot by some corner of another robot.

For a team of N robots there is enough to check the collisions between pairs of robots, so the number of collision checks for the team will be equal to permutations of N things taken 2 at a time

$$A_N^2 = N(N-1) .$$

Given a set of initial conditions for the team of robots

$$X_0 = \{((x_1^{0,1}, y_1^{0,1}, \theta_1^{0,1}), \ldots, (x_N^{0,1}, y_N^{0,1}, \theta_N^{0,1})), \ldots.$$

$$\ldots, ((x_1^{0,M}, y_1^{0,M}, \theta_1^{0,M}), \ldots, (x_N^{0,M}, y_N^{0,M}, \theta_N^{0,M})) . \quad (9)$$

Terminal conditions are also given

$$| x_k(t_f) - x_k^f | \le \varepsilon\ ,\ | y_k(t_f) - y_k^f | \le \varepsilon ,$$

$$| \theta_k(t_f) - \theta_k^f | \le \varepsilon,\ k = 1, \ldots, N , \quad (10)$$

where t_f is not given bounded above runtime of meeting terminal conditions, $t_f \le t^+$, ε is a given small positive value, t^+ is a given time limit for meeting terminal conditions.

Given a quality criterion

$$J=\sum_{i=1}^{M}\left(\sum_{k=1}^{N}\int_{0}^{t_f} f_0(x_k(t),y_k(t),\theta_k(t),u_{1,k}(t),u_{2,k}(t))dt\right)_i \qquad (11)$$

where the lower index i at brackets $(\ldots)_i$ means that the expression in brackets is calculated for initial values $((x_1^{0,i},y_1^{0,i},\theta_1^{0,i}),\ldots,(x_N^{0,i},y_N^{0,i},\theta_N^{0,i}))$, $1\le i\le M$.

Define control as a function

$$u_{i,k}=h_{i,k}(x_1,y_1,\theta_1,\ldots,x_N,y_N,\theta_N)\,, \qquad (12)$$

$$i=1,2\,,\ k=1,\ldots,N\,,$$

which meets the constraints on control (2) and for any initial conditions from (9) in timeframe restricted by $t_f\le t^+$ reaches the objective (10) with minimum value of the quality criterion (11).

To solve the stated problem of control system synthesis the method of network operator can be used directly. In [8] it has been successfully applied for two mobile robot parking task. But when the number of arguments increases the direct synthesis is computationally hard and time consuming. In this case we propose to apply a three-stage synthesis algorithm. On the first stage we synthesize a stabilization system for every robot about some given point in a state space. As far as we have a group of similar robots we can synthesize a stabilizing control function for one robot and apply it for each robot in a team.

We define a point in a state space for the robot k, (x_k^*,y_k^*,θ_k^*). Deviations of the component of the state vector from the defined point are Δx_k, Δy_k, $\Delta\theta_k$ and control will be defined as

$$u_{i,k}=g_i(x_k^*-x_k,y_k^*-y_k,\theta_k^*-\theta_k)\,,\ i=1,2\,, \qquad (13)$$

which provides the achievement of the termination point (x_k^*,y_k^*,θ_k^*) from any point of the domain

$$\mathrm{X}_{0,k}=\{x_k^*\pm\Delta x_k\,,y_k^*\pm\Delta y_k\,,\theta_k^*\pm\Delta\theta_k\}\,. \qquad (14)$$

As optimization criterion we choose a time of achieving the objective

$$J_s=\sum_{i=1}^{8}(t_1)_i\to\min\,, \qquad (15)$$

where sum is calculated for initial conditions from (14)

$$\begin{aligned}&(x_k^*+(-1)^{\lfloor (i-1)/4\rfloor}\Delta x_k\,,\\ &y_k^*+(-1)^{\lfloor (i-1)/2\rfloor}\Delta y_k\,, \qquad (16)\\ &\theta_k^*+(-1)^{i-1}\Delta\theta_k)\,,\ i=1,\ldots,8\,,\end{aligned}$$

$$t_1=\begin{cases}t,\ \text{if}\ t<t^+\ \text{and}\\ \quad \max\{|x_k^*-x(t)|,|y_k^*-y(t)|,|\theta_k^*-\theta(t)|\}<\varepsilon\\ t^+\ \text{- otherwise}\end{cases} \qquad (17)$$

Once the stabilization system is synthesized and the control function (13) is received the robots motion control is performed via changing the position of the stabilization points (x_k^*,y_k^*,θ_k^*). So we need to find spacial optimal trajectories made-up of the points in a state space

$$\mathrm{T}_{k,j}=((x_{k,j,1}^*,y_{k,j,1}^*,\theta_{k,j,1}^*),\ldots,$$

$$\ldots,(x_{k,j,r}^*,y_{k,j,r_j}^*,\theta_{k,j,r_j}^*))\,, \qquad (18)$$

$$k=1,\ldots,N\,,\ j=1,\ldots,M\,,$$

where the trajectories are built for each robot k moving from the initial position $(x_k^{0,k},y_k^{0,k},\theta_k^{0,k})$ to the terminal position

$$x_{k,j,r_j}^*=x_k^f\,,\ y_{k,j,r_j}^*=y_k^f\,,\ \theta_{k,j,r_j}^*=\theta_k^f\,. \qquad (19)$$

The optimality of the trajectories as sets of points in the state space is defined according to the quality criterion (11) with addition of penalties in case if collisions.

Fig. 2 illustrates how the corner k of the robot j penetrates the overall domain of the robot i. The penalty value for the penetration is defined every time as minimal size of penetration

$$p(i,j,t)=\min\{|\tilde{x}_{2,i}-\tilde{x}_i-\tilde{x}_{k,j}|,|\tilde{x}_{1,i}-\tilde{x}_i-\tilde{x}_{k,j}|,$$

$$|\tilde{y}_{1,i}-\tilde{y}_i-\tilde{y}_{k,j}|,|\tilde{y}_{3,i}-\tilde{y}_i-\tilde{y}_{k,j}|\}\,. \qquad (20)$$

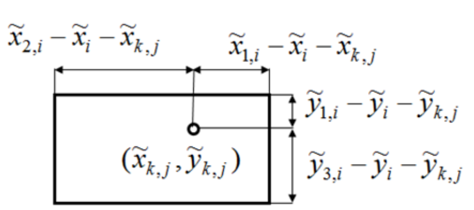
$\tilde{x}_{2,i}-\tilde{x}_i-\tilde{x}_{k,j}$
$\tilde{x}_{1,i}-\tilde{x}_i-\tilde{x}_{k,j}$
$\tilde{y}_{1,i}-\tilde{y}_i-\tilde{y}_{k,j}$
$(\tilde{x}_{k,j},\tilde{y}_{k,j})$
$\tilde{y}_{3,i}-\tilde{y}_i-\tilde{y}_{k,j}$

Figure 2. Penalty definition for collision.

Integrals of penalty values are computed for all M initial points and the results are summed and added to the functional (11). So the resulting quality functional is following

$$\tilde{J}=J+\sum_{l=1}^{M}\left(\int_0^{t_f}\sum_{i=1}^{N}\sum_{j=1}^{N}p(i,j,t)dt\right)_l\to\min\,. \qquad (21)$$

Once the optimal trajectories as sets of points in the state space (18) are obtained we can proceed with the 3rd stage of our approach to accomolish the synthesis. On the last stage we build up a system of differentioal equations which partial solutions approximate the points of the received optimal trajectories

$$\dot{x}_k^* = g_{1,k}(x_1^f - x_1^*, \ldots, x_N^f - x_N^*,$$

$$y_1^f - y_1^*, \ldots, y_N^f - y_N^*, \theta_1^f - \theta_1^*, \ldots, \theta_N^f - \theta_N^*), \quad (22)$$

$$\dot{y}_k^* = g_{2,k}(x_1^f - x_1^*, \ldots, x_N^f - x_N^*,$$

$$y_1^f - y_1^*, \ldots, y_N^f - y_N^*, \theta_1^f - \theta_1^*, \ldots, \theta_N^f - \theta_N^*), \quad (23)$$

$$\dot{\theta}_k^* = g_{3,k}(x_1^f - x_1^*, \ldots, x_N^f - x_N^*,$$

$$y_1^f - y_1^*, \ldots, y_N^f - y_N^*, \theta_1^f - \theta_1^*, \ldots, \theta_N^f - \theta_N^*), \quad (24)$$

where right parts are the desired functions $\mathbf{R}^{3N} \to \mathbf{R}^{3N}$, $k = 1, \ldots, N$.

A block diagram of the control system of a mobile robot is presented in fig.3.

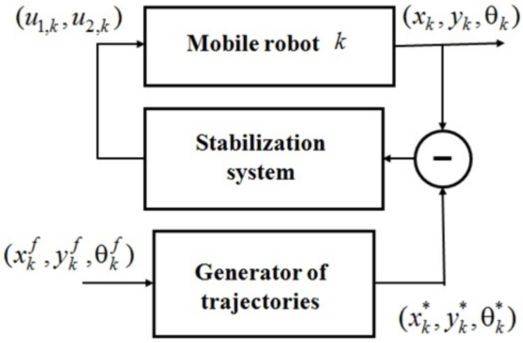
$(u_{1,k}, u_{2,k})$
Mobile robot k
(x_k, y_k, θ_k)
Stabilization system
−
$(x_k^f, y_k^f, \theta_k^f)$
Generator of trajectories
$(x_k^*, y_k^*, \theta_k^*)$

Figure 3. Control system of mobile robot.

Generator of trajectories in fig. 3 implements the solution of the system (22) - (24).

To identify a model of the generator of trajectories we use the multilayer network operator method as well.

3 Multilayer network operator

The core idea of the method of the network operator is that a mathematical expression is coded in the form of integer upper-triangular matrix. The cells of the matrix correspond by its numbers to the proper variables, parameters and mathematical operations. The search for the needed mathematical expression, which satisfies the objectives and criteria, is performed by the evolutionary algorithm on the basis of the principle of small variations of basic solution [9].

The multilayer network operator is a set of network operators connected with each other. The connections are described in a connectivity matrix of size $K \times 2I$, where K is a number of network operators or layers, I is a number of input in each layer. The outputs of the multilayer network operator are stored in an output matrix of size $M \times 2$, where M is a dimension of the vector-function described by the multilayer network operator.

Consider an example of coding a mathematical expression by multilayer network operator.

Given a vector mathematical expression

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} \cos(x_1 + \sin(x_2 + q_2)) \\ e^{-q_1 x_1 + q_2} \cos(x_2^2 - x_1^2) \end{bmatrix}.$$

and the following sets of elementary functions

$$\mathrm{F}_0 = \{f_{0,1} = x_1, f_{0,2} = x_2, f_{0,3} = q_1, f_{0,4} = q_2\},$$

$$\mathrm{F}_1 = \{f_{1,1}(z) = z, f_{1,2}(z) = -z, f_{1,3}(z) = e^z,$$

$$f_{1,4}(z) = z^2, f_{1,5}(z) = \sin(z), f_{1,6}(z) = \sin(z)\},$$

$$\mathrm{F}_2 = \{f_{2,1}(z_1, z_2) = z_1 + z_2, f_{2,1}(z_1, z_2) = z_1 z_2\}.$$

Let us build a network operator graph of the mathematical expression according to [2]. The resulting graph is shown on fig. 4.

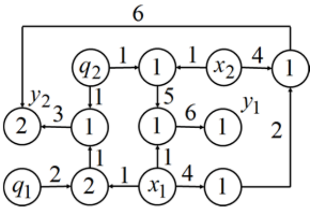

Figure 4. Network operator graph of the expression

We split the network operator graph into several subgraphs. Each subgraph is a layer. Fig.5 shows three layers of the spited graph.

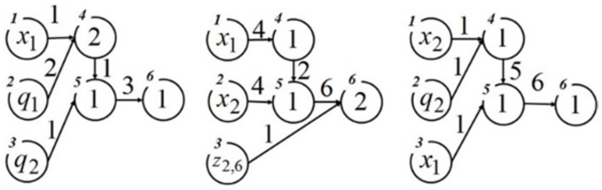

Figure 5. Layers of the network operator

The layers are described by the following matrices

$$\mathbf{\Psi}_1 = \begin{bmatrix} 0&0&0&1&0&0 \\ 0&0&0&1&0&0 \\ 0&0&0&0&1&0 \\ 0&0&0&1&5&0 \\ 0&0&0&0&1&6 \\ 0&0&0&0&0&1 \end{bmatrix}, \mathbf{\Psi}_2 = \begin{bmatrix} 0&0&0&1&0&0 \\ 0&0&0&2&0&0 \\ 0&0&0&0&1&0 \\ 0&0&0&1&1&0 \\ 0&0&0&0&1&3 \\ 0&0&0&0&0&1 \end{bmatrix}, \mathbf{\Psi}_3 = \begin{bmatrix} 0&0&0&4&0&0 \\ 0&0&0&0&4&0 \\ 0&0&0&0&0&1 \\ 0&0&0&1&2&0 \\ 0&0&0&0&1&6 \\ 0&0&0&0&0&2 \end{bmatrix}$$

and the connectivity matrix

$$\mathbf{\Sigma} = \begin{bmatrix} 0&2&0&4&0&1 \\ 0&1&0&3&0&4 \\ 0&1&0&2&2&6 \end{bmatrix}.$$

The output matrix is

$$\mathbf{\Omega} = \begin{bmatrix} 1&6 \\ 3&6 \end{bmatrix},$$

where the upper row indicates the number of layer and the bottom row contains the corresponding number of output in the indicated layer.

For backward transformations from the multilayer network operator to the mathematical expression we use the following formula

$$z_{k,r}=\begin{cases} z^0_{\sigma_{k,2r}}, \text{ if } \sigma_{k,2r-1}=0 \\ z_{\sigma_{k,2r-1},\sigma_{k,2r}} \text{ - otherwise} \end{cases},$$

$$z_{k,j}\leftarrow\begin{cases} \chi_{\psi_{k,j,i}}\left(z_{k,j},\rho_{\psi_{k,i,j}}\left(z_{k,i}\right)\right), \text{if } \psi_{k,i,j}\neq 0 \\ z_{k,j} \text{ - otherwise} \end{cases},$$

where $\mathbf{\Psi}_k=[\psi_{k,i,j}]$, $\mathbf{\Sigma}=[\sigma_{k,m}]$, $m=\overline{1,2I}$, k is a layer number, $k=\overline{1,N}$, r is a number of input, $r=\overline{1,I}$, i is a number of row in the network operator matrix, $i=\overline{1,L}$, j is a number of column in the network operator matrix, $j=\overline{i+1,L-2}$, $\rho_d(z)$ is a unary operation number d, $\chi_g(z_1,z_2)$ is a binary operation number g.

4 Simulations

In the simulation example three similar mobile robots are considered: $N=3$, $L=2$, $W=1$. We considered 8 initial points X_0:

$\{((-10,2,0),(-10,4,0),(-10,6,0)),((10,2,0),(-10,4,0),(-10,6,0)),$
$((-10,2,0),(10,4,0),(-10,6,0)),((-10,2,0),(-10,4,0),(10,6,0)),$
$((10,2,0),(10,4,0),(-10,6,0)),((10,2,0),(-10,4,0),(10,6,0)),$
$((-10,2,0),(10,4,0),(10,6,0)),((10,2,0),(10,4,0),(10,6,0))\}$.

Terminal conditions were

$x_i-x_{i,f}=0,\ \ y_i-y_{i,f}=0,\ \ \theta_i-\theta_{i,f}=0$, $i=1,2,3$,

where $x_{1,f}=-5,\ y_{1,f}=0,\ \theta_{1,f}=0,\quad x_{2,f}=0,$ $y_{2,f}=0,\ \ \theta_{2,f}=0,\ \ x_{3,f}=5,\ \ y_{3,f}=0,\ \theta_{3,f}=0$.

The quality functional was a total time of achiving terminal conditions

$$J=t_f\rightarrow\min,$$

where

$$t_f=\begin{cases} t, \text{ if } \max\{|x_i(t)-x_{i,f}|, |y_i(t)-y_{i,f}|, \\ \qquad |\theta_i(t)-\theta_{i,f}|,\ i=1,2,3\}<\varepsilon \\ t^+ \text{ - otherwise} \end{cases},$$

$\varepsilon=0{,}01$, $t^+=12$.

We set static phase constraints as two rectangular domains.

On the first stage we solved stabilization problem by the network operator. The received stabilization control allows robots stabilizing in a given point as shown in fig.6.

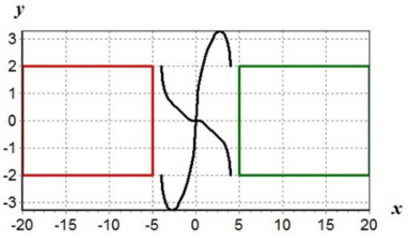

Figure 6. Stabilization of the system from initial positions (-4,-2,0), (4,2,0), (-4,2,0), (4,-2,0)

For control synthesis we used a multilayer network operator of 4 layers sized 16×16. As a result, the system of generating optimal trajectories has been received. Fig.7 illustrates the generated trajectories for one initial position.

Thus, we demonstrated a new approach for synthesis of optimal motion control of multi robot team. The received optimal trajectories allow robots achieving the terminal conditions with optimal values of functionals and not violating the phase constraints and avoiding collisions.

This work has been supported by the Russian Fund of Basic Researches (14-08-00008-a) and by the grant of the President of Russian Federation (MK-6277.2015.8).

References

1. K. Åström and T. Hägglund, *PID controllers: Theory, Design and Tuning* (1995)
2. M. Fliess and C. Join, *15th IFAC Symp. on Syst. Ident.* (2009)
3. D.P. Garg and G.K. Fricker, *iNaCoMM2013* (2013)
4. R. Bellman, SIAM J. on Cont., **1** (1962)
5. S.G. Nersesov and W. M. Haddad, IEEE Trans. on Automatic Contr., **51**, 203 (2006)
6. T. Kaviczky, F. Borelli, K. Fregene, D. Godbole and G.J.Balas, IEEE Trans. on Cont. Syst. Tech., **16**, (2008)
7. A.I. Diveev and E.A. Sofronova, *Proc. 7th Int. Conf. on Contr. and Automat. (ICCA'09)*, pp. 701-708 (2009)
8. A.I. Diveev and E.Yu. Shmalko, *Proc. of 2015 Europ. Contr. Conf.*, pp. 709-714. (2015)
9. A.I. Diveev, IFAC-PapersOnLine, **48**, 028 (2015).

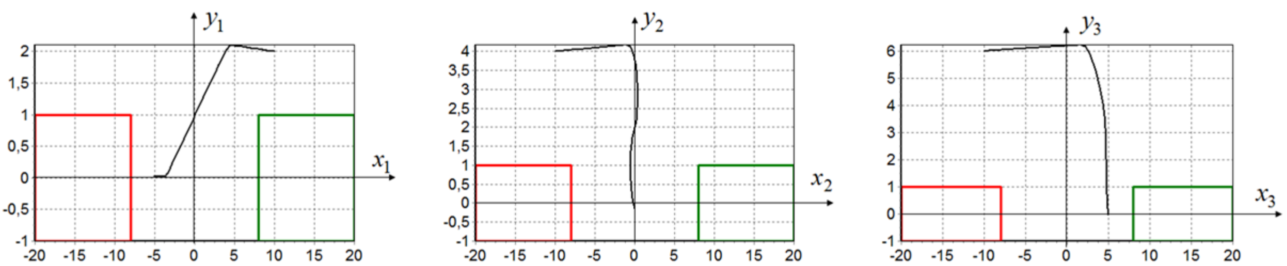

Figure 7. Optimal trajectory of the centre of the 1st, 2nd and 3rd robots from initial position $((10,2,0),(-10,4,0),(-10,6,0))$.