mathematics

MDPI

Article

Optimal Control Problem Solution with Phase Constraints for Group of Robots by Pontryagin Maximum Principle and Evolutionary Algorithm

Askhat Diveev [1], Elena Sofronova [1,*] and Ivan Zelinka [2,3]

[1] Department of Robotics Control, Federal Research Center "Computer Science and Control" of the Russian Academy of Sciences, 119333 Moscow, Russia; frccsc@frccsc.ru
[2] Faculty of Electrical and Electronics Engineering, Ton Duc Thang University, Ho Chi Minh City 758307, Vietnam; ivan.zelinka@tdt.edu.vn or ivan.zelinka@vsb.cz
[3] Department of Computer Science, FEI, VSB Technical University of Ostrava, 70800 Ostrava, Czech Republic
* Correspondence: sofronova_ea@mail.ru

Received: 29 October 2020; Accepted: 20 November 2020; Published: 25 November 2020

Abstract: A numerical method based on the Pontryagin maximum principle for solving an optimal control problem with static and dynamic phase constraints for a group of objects is considered. Dynamic phase constraints are introduced to avoid collisions between objects. Phase constraints are included in the functional in the form of smooth penalty functions. Additional parameters for special control modes and the terminal time of the control process were introduced. The search for additional parameters and the initial conditions for the conjugate variables was performed by the modified self-organizing migrating algorithm. An example of using this approach to solve the optimal control problem for the oncoming movement of two mobile robots is given. Simulation and comparison with direct approach showed that the problem is multimodal, and it approves application of the evolutionary algorithm for its solution.

Keywords: optimal control problem; evolutionary computation; robotics applications

1. Introduction

The optimal control belongs to complex computational problems for which there are no universal solution algorithms. The most well-known result in this area [1] transforms the optimization problem into a boundary-value problem, and the dimension of the problem doubles. The goal of solving the boundary-value problem is to find the initial conditions for conjugate variables such that the vector of state variables falls into a given terminal condition. In general, for this problem, there is no guarantee that the functional for the boundary-value problem is not unimodal and convex on the space of initial conditions of conjugate variables.

The optimal control problem with phase constraints is considered. Phase constraints are included in the functional, so they are included in the system of equations for conjugate variables. This greatly complicates the analysis of the problem on the convexity and unimodality of the target functional. The accurate solution of optimal control problem has to use additional functions and regularization of equations at the search of control [2,3]. An additional problem in solving a boundary-value problem is determination of time for checking the fulfillment of boundary conditions.

In this paper, for the numerical solution of the problem, it is proposed to use evolutionary algorithms that have shown efficiency in solving optimal control problems [4]. SOMA is a universal algorithm for various difficult optimization problems [5,6]. However, our attempt to apply SOMA to the optimal control problem of four robots with constraints has failed to find a good solution for any values of the algorithm parameters. We supposed that the modification of each possible solution in population in the process of evolution using the best current possible solution is not enough [7]. We expanded the modification of SOMA by introducing the best historical solution among randomly selected ones for each possible solution in the population.

The article consists of an introduction and eight sections. Statement of the optimal control problem with phase constraints is presented in Section 2. Section 3 contains Pontryagin maximum principle as one of main approaches for its numerical solution. Section 4 contains a description of one of the evolutionary algorithms, modified SOMA. An example is given in Section 5. The computational experiment and results are presented in Section 6. Section 7 describes the search of optimal control by direct method. Alternative non-deterministic control methods are observed in Section 8. Results and future research directions are discussed in Section 9.

2. Optimal Control Problem with Phase Constraints for Group of Robots

Consider the problem of optimal control for a group of robots with phase constraints. Given a mathematical model of control objects in the form of the system of ordinary differential equations

$$\dot{\mathbf{x}}^j = \mathbf{f}^j(\mathbf{x}^j, \mathbf{u}^j), \tag{1}$$

where $\mathbf{x}^j$ is a state space vector of control object j, $\mathbf{x}^j \in \mathbb{R}^{n_j}$, $\mathbf{u}^j$ is a control vector of object j, $\mathbf{u}^j \in \mathrm{U}^j \subseteq \mathbb{R}^{m_j}$, U^j is a compact limited set, $m_j \leqslant n_j$, $j = 1, \ldots, M$, M is a number of objects. For the system (1) initial conditions are given

$$\mathbf{x}^j(0) = \mathbf{x}^{j,0} \in \mathbb{R}^{n_j}, j = 1, \ldots, M. \tag{2}$$

Given terminal conditions

$$\mathbf{x}^j(t_{f,j}) = \mathbf{x}^{j,f} \in \mathbb{R}^{n_j}, \tag{3}$$

where $t_{f,j}$ is an unknown limited positive value, that corresponds to time when object j achieves its terminal position

$$t_{f,j} \leqslant t^+, \tag{4}$$

t^+ is a given time of achievement of terminal conditions (3),

$$t_{f,j} = \begin{cases} t, \text{if } t < t^+ \text{ and } \|\mathbf{x}^j(t) - \mathbf{x}^{j,f}\| \leqslant \varepsilon_1 \\ t^+, \text{otherwise} \end{cases}, \tag{5}$$

ε_1 is a small positive value, $j = 1, \ldots, M$. The phase constraints are given

$$\varphi_i(\mathbf{x}^j(t)) \leqslant 0,\ i = 1, \ldots, r,\ j = 1, \ldots, M. \tag{6}$$

The conditions of collision avoidance are described as

$$\chi(\mathbf{x}^j(t), \mathbf{x}^k(t)) \leqslant 0,\ j = 1, \ldots, M-1,\ k = j+1, \ldots, M. \tag{7}$$

The quality functional is given in general integral form

$$J_0 = \int_0^{t_f} f_0(\mathbf{x}^1(t), \ldots, \mathbf{x}^M(t), \mathbf{u}^1(t), \ldots, \mathbf{u}^M(t))dt \to \min, \tag{8}$$

where

$$t_f = \max\{t_{f,1}, \ldots, t_{f,M}\}. \tag{9}$$

It is necessary to find control as a time function

$$\mathbf{u}^j = \mathbf{v}^j(t),\ j = 1, \ldots, M, \tag{10}$$

in order to provide terminal conditions (3) with optimal value of functional (8) without violation of constraints (6) and with collision avoidance (7). For a numerical solution of the problem, let us insert phase constraints and terminal conditions in quality functional (8)

$$J_1 = \int_0^{t_f} f_0(\mathbf{x}^1(t), \ldots, \mathbf{x}^M(t), \mathbf{u}^1(t), \ldots, \mathbf{u}^M(t))dt + a\int_0^{t_f} \sum_{i=1}^{r}\sum_{j=1}^{M} \mu^2(\varphi_i(\mathbf{x}^j(t)))dt +$$
$$b\int_0^{t_f} \sum_{j=1}^{M-1}\sum_{k=j+1}^{M} \mu^2(\chi(\mathbf{x}^j(t), \mathbf{x}^k(t)))dt + c\sum_{j=1}^{M} ||\mathbf{x}^j(t_f) - \mathbf{x}^{j,f}|| \to \min, \tag{11}$$

where a, b, c are given positive weight coefficients, $\mu(A) = \max\{0, A\}$.

To solve the problem stated above, we use the Pontryagin maximum principle.

3. The Pontryagin Maximum Principle

The Pontryagin maximum principle allows one to transform the problem of optimization on infinite dimensional space to the boundary-value problem for the system of differential Equations (1). Let us construct Hamilton function for this problem on the base of the system (1) and the quality functional (2) without terminal conditions

$$H(\mathbf{x}^1, \ldots, \mathbf{x}^M, \mathbf{u}^1, \ldots, \mathbf{u}^M, \psi) = -f_0(\mathbf{x}^1, \ldots, \mathbf{x}^M, \mathbf{u}^1, \ldots, \mathbf{u}^M) -$$
$$a\sum_{i=1}^{r}\sum_{j=1}^{M} \mu^2(\varphi_i(\mathbf{x}^j)) - b\sum_{j=1}^{M-1}\sum_{k=j+1}^{M} \mu^2(\chi(\mathbf{x}^j, \mathbf{x}^k)) + \psi^T \mathbf{f}^j(\mathbf{x}^j, \mathbf{u}^j), \tag{12}$$

where $\psi = [\psi_1 \ldots \psi_n]^T$ is a vector of conjugate variables, $n = n_1 + \ldots + n_M$,

$$\dot{\psi} = -\frac{\partial H(\mathbf{x}^1, \ldots, \mathbf{x}^M \mathbf{u}^1, \ldots, \mathbf{u}^M, \psi)}{\partial \mathbf{x}^1, \ldots, \partial \mathbf{x}^M}. \tag{13}$$

According to the Pontryagin maximum principle, a necessary condition for optimal control is the maximum of Hamilton function (12)

$$\max_{\mathbf{u}^1 \in U^1, \ldots, \mathbf{u}^M \in U^M} H(\mathbf{x}^1, \ldots, \mathbf{x}^M, \mathbf{u}^1, \ldots, \mathbf{u}^M, \psi). \tag{14}$$

Pontryagin maximum principle allows one to transform the optimal control problem to a boundary-value problem. It is necessary to find the initial values of conjugate variables so that the state vector reaches

terminal conditions (3). To solve the boundary-value problem, we have to solve a finite dimensional problem of nonlinear programming with the following functional

$$F(\mathbf{q}) = \sum_{j=1}^{M} \|\mathbf{x}^j(t_{f,j}) - \mathbf{x}^{j,f}\| \to \min_{\mathbf{q}\in \mathrm{Q}}, \tag{15}$$

where $\mathbf{q} = [q_1 \dots q_n]^T$, Q is a limited compact set, $q_i = \psi_i(0)$, $i = 1, \dots, n$,

$$\sum_{i=1}^{n} q_i^2 = 1. \tag{16}$$

In a boundary-value problem, it is not known exactly when it is necessary to check the boundary conditions (15). The maximum principle does not provide equations for definition of terminal time $t_{f,j}$ of the control process, while a numerical search of some possible solutions may not reach the terminal condition. To avoid this problem, let us add parameter q_{n+1} for the time limit of reaching the terminal state. As a result, the goal functional for the boundary-value problem is the following

$$\tilde{F}(\mathbf{q}) = \sum_{j=1}^{M} \|\mathbf{x}^j(t^+ + q_{n+1}) - \mathbf{x}^{f,j}\| \to \min_{\tilde{\mathbf{q}}\in\tilde{\mathrm{Q}}}, \tag{17}$$

where $\tilde{\mathbf{q}} = [q_1 \dots q_{n+1}]^T$,$\tilde{\mathrm{Q}} = \mathrm{Q} \times [q_{n+1}^-; q_{n+1}^+]$, q_{n+1}^-, q_{n+1}^+ are low and up limitations of the parameter q_{n+1}. During the search process, we can decrease time t^+ according to sign of parameter q_{n+1}. If found parameter q_{n+1} is less than zero, then t^+ is decreased and the interval for values of parameter $[q_{n+1}^-; q_{n+1}^+]$ is also narrowed.

4. Evolutionary Algorithm

The boundary-value problem may have a nonconvex and nonunimodal objective functional (15) on the parameter space $\mathbf{q}$, therefore, to solve this problem, it is advisable to use an evolutionary algorithm.

Evolutionary algorithms differ in the form of changing possible solutions. The first evolutionary algorithms appeared at the end of the XX century and continue to appear. Currently, hundreds of evolutionary algorithms are known. Most of them are named after animals, although the connection between animal behavior and computational algorithms is not strictly proven anywhere and is determined only by the statement of the author of the algorithm. Common steps of evolutionary algorithms are: generation of a set of possible solutions, assessment of solutions by objective function to find one or more best solutions, modification of solutions in accordance with the value of its objective function and with information about the values of the objective functions of other solutions by evolutionary operators.

In this work, we investigate the application of the Pontryagin maximum principle to solve the optimal control problem for a group of robots with phase constraints, and do not compare evolutionary algorithms. We applied one of the effective evolutionary algorithms, self-organizing migrating algorithm (SOMA) [5,6], with modification [7] to find the parameters, i.e., initial conditions of conjugate variables and additional parameter q_{n+1} for terminal time. The modified SOMA includes the following steps.

Generate a population of H possible solutions, taking into account

$$\bar{q}_i^j = \xi(q_i^+ - q_i^-) + q_i^-,\ i = 1, \dots, n+1,\ j = 1, \dots, H, \tag{18}$$

where $q_i^+ = 1$, $q_i^- = -1$, $i = 1, \ldots, n$, $q_{n+1}^+ = 0.2$, $q_{n+1}^- = -2.5$, H is a cardinal number of the population set, ξ is a random number from 0 to 1. Normalize the first n possible solutions according to (16)

$$q_i^j = \frac{\bar{q}_i^j}{\sqrt{\sum_{k=1}^n (\bar{q}_k^j)^2}},\ i = 1, \ldots, n,\ j = 1, \ldots, H. \tag{19}$$

In the optimization problem, we have to find a vector of optimal parameters $\mathbf{q} = [q_1 \ldots q_{n+1}]^T$ in order to receive the minimal value of functional

$$J_1(\mathbf{q}) = \sum_{j=1}^M ||\mathbf{x}(t^+ + q_{n+1}) - \mathbf{x}^{f,j}|| \to \min. \tag{20}$$

For each vector of parameters, we set a historical vector. Initially, historical vectors contain zero elements

$$\tilde{q}_i^j = 0,\ i = 1, \ldots, n+1,\ j = 1, \ldots, H. \tag{21}$$

Calculate the values of functional for each possible solution

$$f_j = J_1(\mathbf{q}^j),\ j = 1, \ldots, H. \tag{22}$$

Find the best possible solution $\mathbf{q}^{j_0}$ on a stage of evolution

$$J_1(\mathbf{q}^{j_0}) = \min\{f_1, \ldots, f_H\}. \tag{23}$$

For each historical vector, find the best vector among the randomly selected ones $\tilde{\mathbf{q}}^j$ in current population

$$J_1(\mathbf{q}^{j*}) = \min\{f_{j_1}, \ldots, f_{j_K}\}, \tag{24}$$

where $j_i \in \{1, \ldots, H\}$, $i = 1, \ldots, K$. Transform each historical vector

$$\tilde{q}_i^j \leftarrow \alpha \tilde{q}_i^j + \beta(q_i^{j^*} - q_i^j), \tag{25}$$

where $i = 1, \ldots, n+1$, $j = 1, \ldots, H$, α and β are parameters of the algorithm, positive numbers less than one. Let us set a step $t = \delta$. Calculate some new values for each possible solution

$$\hat{q}_i^j(t) = \begin{cases} q_i^j + \tilde{q}_j^i + t(q_i^{j_0} - q_i^j), \text{ if } \xi < P_{rt} \\ q_i^j + \tilde{q}_j^i, \text{otherwise} \end{cases}, \tag{26}$$

where $i = 1, \ldots, n+1$, P_{rt} is a parameter of the algorithm. Check each component of a new vector for restrictions

$$q_i^j(t) = \frac{\hat{q}_i^j(t)}{\sqrt{\sum_{k=1}^n (\hat{q}_k^j(t))^2}},\ i = 1, \ldots, n, \tag{27}$$

$$q_{n+1}^j = \begin{cases} q_{n+1}^+, \text{ if } \hat{q}_{n+1}^j(t) > q_{n+1}^+ \\ q_{n+1}^-, \text{ if } \hat{q}_{n+1}^j(t) < q_{n+1}^- \\ \hat{q}_{n+1}^j(t), \text{otherwise} \end{cases}. \tag{28}$$

Calculate the functional for a new vector

$$f_j(t) = J_1(\mathbf{q}^j(t)). \tag{29}$$

If $f_j(t) \leqslant f_j$, then we change possible solution $\mathbf{q}^j$ by a new vector

$$\mathbf{q}^j \leftarrow \mathbf{q}^j(t), \tag{30}$$

$$f_j \leftarrow f_j(t). \tag{31}$$

Increase t

$$t = t + \delta. \tag{32}$$

If $t < P_{length}$ then repeat calculations (25)–(32), P_{length} is a parameter of the algorithm. Repeat calculations (22)–(32) for all possible solutions in the population. Then again, find the best solution (23) and change historical vector (25). Repeat all stages R times. The last best vector is a solution of the optimization problem.

An applied algorithm with historical vector is called a modified SOMA. The value of parameter $\beta = 0$ transforms the algorithm from modified SOMA to classical SOMA. Pseudo code of the modified SOMA has the following form, see Algorithm 1.

Algorithm 1: Modified SOMA for Optimal Control Problem.

Procedure ModSOMA ($H, R, P_{length}, P_{rt}, \delta$; var $\mathbf{q}^{j_0}$)
for ($j = 1, \ldots, H$) //generation of initial population
  $s = 0$
  for ($i = 1, \ldots, n$) //normalization of first n elements of each $\mathbf{q}^j$
    $q_i^j = 2 \cdot Random - 1$
    $s = s + (q_i^j)^2$
  end for
  for ($i = 1, \ldots, n$)
    $q_i^j = q_i^j / \sqrt{s}$
    $\tilde{q}_i^j = 0$ //* setting of initial values of historical vector
  end for
  $q_{n+1}^j = q_{n+1}^- + Random \cdot (q_{n+1}^+ - q_{n+1}^-)$
  $\tilde{q}_{n+1}^j = 0$ //*
  $f_j = J_1(\mathbf{q}^j)$ //estimation of each possible solution
end for
for ($r = 1, \ldots, R$) ; //generations
  $j_0 = 1$
  for $j = 2, \ldots, H$ //search for the best current solution
    if $f_j < f_{j_0}$ **then**
      $j_0 = j$
  end for
  for ($j = 1, \ldots, H$) //evolution of all solutions
    $j^* = Random(H)$ //*
    for ($l = 1, \ldots, K$) //* search for the best solutions among randomly selected ones
      $j_l = Random(H)$ //*

Algorithm 1: *Cont.*

```
    if f_{j_l} < f_{j^*} then
      j^* = j_l          //*
    end if               ;                                          //*
  end for                //*
  for (i = 1, ..., n+1)  //* transformation of historical vectors
    q̃_i^j = α q̃_i^j + Random · β(q_i^{j^*} − q_i^j)    //*
  end for                //*
  t = δ
  while t < P_length do      //termination condition
    s = 0
    for (i = 1, ..., n)   //calculation of new values of parameters
      if Random < P_rt then
          q̂_i^j(t) = q_i^j + q̃_i^j + t(q_i^{j_0} − q_i^j)
      else
          q̂_i^j = q_i^j + q̃_i^j
      end if
      s = s + (q̃_i^j)^2
    end for
    for (i = 1, ..., n)       //normalization
      q_i^j(t) = q̂_i^j(t)/√s
    end for
    if Random < P_rt then
      q̂_{n+1}^j(t) = q_{n+1}^j + q̃_{n+1}^j + t(q_{n+1}^{j_0} − q_{n+1}^j)
    else
      q̂_{n+1}^j(t) = q̂_{n+1}^j(t)
    end if
    if q̂_{n+1}^j > q_{n+1}^+ then
      q_{n+1}^j = q_{n+1}^+
    end if
    if q̂_{n+1}^j < q_{n+1}^− then
      q_{n+1}^j = q_{n+1}^−
    end if
    f_j(t) = J_1(q^j(t))          //estimation of new vector of parameters
    if f_j(t) < f_j then
      f_j = f_j(t)
      for (i = 1, ..., n+1)
        q_i^j = q_i^j(t)       //transformation of vector of parameters
      end for
    end if
    t = t + δ
  end while do
 end for (j = H)
end for (r = R)
```

In pseudo code, subroutine Random generates a random real number from 0 to 1, and subroutine *Random(A)* generates random integer number from 0 to $A - 1$. We used * in comments to highlight the modification of modified SOMA in comparison to original SOMA.

The effectiveness of modified SOMA, as with all evolutionary algorithms, depends on the parameters that influence the number of computational operations, i.e., number of elements in initial population (H), number of generations (R), number of evolutions (P). To evaluate one single solution, we need to simulate the whole system, thus, for the problem, we have to calculate the functional minimum $H + nRP$ times, where n depends on parameter of algorithm P_{length}.

As for all evolutionary algorithms, the convergence of modified SOMA is determined by probability. The more solutions are looked through, the more is the probability to find the optimal one. In evolutionary algorithms, the value of goal function depends on the number of generations as descending exponent. If the solution is not improved for some generations, then the search is stopped, and the best current solution is considered to be the solution to the problem. The optimal control problem with phase constraints is not unimodal, and the search algorithm is not deterministic, thus, to find the solution, the algorithm ran multiple times.

5. An Example

Consider a control problem for two similar mobile robots. Mathematical model of control objects has the following form

$$\begin{aligned} \dot{x}^j &= 0.5(u_1^j + u_2^j)\cos\theta^j, \\ \dot{y}^j &= 0.5(u_1^j + u_2^j)\sin\theta^j, \\ \dot{\theta}^j &= 0.5(u_1^j - u_2^j), \end{aligned} \tag{33}$$

where $j = 1, 2$.

The control is limited

$$u_i^- \leqslant u_i^j \leqslant u_i^+, \tag{34}$$

where $j = 1, 2$, u_i^-, u_i^+ are the given constraints, $i = 1, 2$. For the system (33) , the initial conditions are

$$x^j(0) = x_0^j,\ y^j(0) = y_0^j,\ \theta^j(0) = \theta_0^j. \tag{35}$$

The terminal conditions are

$$x^j(t) = x_f^j,\ y^j(t) = y_f^j,\ \theta^j(t) = \theta_f^j. \tag{36}$$

The static phase constraints are

$$\varphi_i(x^j, y^j) = r_i^2 - (x_i^* - x^j)^2 - (y_i^* - y^j) \leqslant 0, \tag{37}$$

where $j = 1, 2$, r_i, x_i^*, y_i^* are the given parameters of constraints, $i = 1, \ldots, r$, r is a number of static phase constraints. For two robots, we have only one dynamic phase constraint

$$\chi(\mathbf{x}^1, \mathbf{x}^2) = d^2 - (x^1 - x^2)^2 - (y^1 - y^2)^2 \leqslant 0, \tag{38}$$

where d is a given minimal distance between robots.

A quality functional is

$$J = t_f + \sum_{j=1}^{2} \delta_j(t_f), \tag{39}$$

where

$$t_f = \begin{cases} t, \text{ if } t < t^+ \text{and} \sum_{j=1}^{2} \delta_j(t) < \varepsilon \\ t^+ \text{ otherwise} \end{cases}, \tag{40}$$

$$\delta_j(t) = \sqrt{(\Delta_x^j)^2(t) + (\Delta_y^j)^2(t) + (\Delta_\theta^j)^2(t)}, \tag{41}$$
$$\Delta_x^j(t) = x^j(t) - x_f^j,$$
$$\Delta_y^j(t) = y^j(t) - y_f^j,$$
$$\Delta_\theta^j(t) = \theta^j(t) - \theta_f^j,$$

$j = 1, 2$, ε is a small positive value.

To obtain equations for conjugate variables, all constraints are included in the quality criterion, and terminal conditions are excluded as follows

$$J = t_f + a\int_0^{t_f} \sum_{j=1}^{2} \mu^2(\varphi_i(x^j, y^j))dt + b\int_0^{t_f} \mu^2(\chi(\mathbf{x}^1, \mathbf{x}^2))dt. \tag{42}$$

Suggesting that the problem is not abnormal, let us write down the Hamilton function in the following form

$$H(\mathbf{x}, \mathbf{u}, \psi) = -1 - \sum_{j=1}^{2} \mu^2(\varphi_i(x^j, y^j)) - a\mu^2(\chi(\mathbf{x}^1, \mathbf{x}^2)) + 0.5\sum_{j=1}^{2} \psi_{1+3(j-1)}(u_1^j + u_2^j)\cos\theta^j +$$
$$0.5\sum_{j=1}^{2} \psi_{2+3(j-1)}(u_1^j + u_2^j)\sin\theta^j + 0.5\sum_{j=1}^{2} \psi_{3j}(u_1^j - u_2^j). \tag{43}$$

As a result, the differential equations for conjugate variables are

$$\dot{\psi}_{1+3(j-1)} = -\frac{\partial H(\mathbf{x}, \mathbf{u}, \psi)}{\partial x^j} = 4\mu(r_1^2 - (x_1^* - x^j)^2 - (y_1^* - y^j))(x_1^* - x^j) +$$
$$4\mu(r_1^2 - (x_2^* - x^j)^2 - (y_2^* - y^j))(x_2^* - x^j) - 4\mu(d^2 - (x^1 - x^2)^2 - (y^1 - y^2))(x^1 - x^2),$$
$$\dot{\psi}_{2+3(j-1)} = -\frac{\partial H(\mathbf{x}, \mathbf{u}, \psi)}{\partial y^j} = 4\mu(r_1^2 - (x_1^* - x^j)^2 - (y_1^* - y^j))(y_1^* - y^j) + \tag{44}$$
$$4\mu(r_1^2 - (x_2^* - x^j)^2 - (y_2^* - y^j))(y_2^* - y^j) + 4\mu(d^2 - (x^1 - x^2)^2 - (y^1 - y^2))(y^1 - y^2),$$
$$\dot{\psi}_{3j} = -\frac{\partial H(\mathbf{x}, \mathbf{u}, \psi)}{\partial \theta^j} = 0.5\psi_{1+3(j-1)}(u_1^j + u_2^j)\sin\theta^j - 0.5\psi_{2+3(j-1)}(u_1^j + u_2^j)\cos\theta^j,$$

where $j = 1, 2$. Optimal control is calculated from equations

$$\tilde{u}_1^j = \begin{cases} u_1^{j+}, \text{ if } W_j + \psi_{3j} > 0 \\ u_1^{j-}, \text{ if } W_j + \psi_{3j} < 0 \\ \text{special control mode, if } W_j + \psi_{3j} = 0 \end{cases}, \tag{45}$$

$$\tilde{u}_2^j = \begin{cases} u_2^{j+}, \text{ if } W_j - \psi_{3j} > 0 \\ u_2^{j-}, \text{ if } W_j - \psi_{3j} < 0 \\ \text{special control mode, if } W_j - \psi_{3j} = 0 \end{cases}, \tag{46}$$

where

$$W_j = \psi_{1+3(j-1)}\sin\theta^j + \psi_{2+3(j-1)}\cos\theta^j, \tag{47}$$

u_1^{j+}, u_2^{j+}, u_1^{j-}, u_2^{j-} are upper and lower values of control for robot j, $j = 1, 2$.

The nonlinear programming problem consists of finding initial conditions for conjugate variables

$$q_i = \psi_i(0),\ i = 1, \ldots, 6, \tag{48}$$

so that initial conditions have to allocate on a sphere with a unit radius

$$\sqrt{\sum_{i=1}^{6} q_i^2} = 1, \tag{49}$$

as well as terminal time

$$t_f = t^+ + q_7, \tag{50}$$

and special control modes

$$u_1^1 = q_8, \text{ if } |W_1 + \psi_3| < \varepsilon_0, \tag{51}$$

$$u_2^1 = q_9, \text{ if } |W_1 - \psi_3| < \varepsilon_0, \tag{52}$$

$$u_1^2 = q_{10}, \text{ if } |W_2 + \psi_6| < \varepsilon_0, \tag{53}$$

$$u_2^2 = q_{11}, \text{ if } |W_1 - \psi_6| < \varepsilon_0, \tag{54}$$

where ε_0 is a small positive value.

A goal function for the nonlinear programming has the following form

$$F = \sum_{j=1}^{2} \delta_j(t_f) + a \int_0^{t_f} \sum_{j=1}^{2} \mu^2(\varphi_i(x^j, y^j))dt + b \int_0^{t_f} \mu^2(\chi(\mathbf{x}^1, \mathbf{x}^2))dt. \tag{55}$$

6. Computational Experiment

The problem had the following parameters: a number of objects $M = 2$, dimensions of objects $n_1 = 3$, $n_2 = 3$, dimension of state space vector $n = 3 + 3 = 6$, dimensions of control $m_1 = 2$, $m_2 = 2$, dimension of control space $m = 2 + 2 = 4$ dimension of parameter vector $n + 1 + m = 11$.

The following values of parameters of the algorithm were used: $x_0^1 = 0$, $y_0^1 = 0$, $\theta_0^1 = 0$, $x_0^2 = 10$, $y_0^2 = 10$, $\theta_0^2 = 0$, $x_f^1 = 10$, $y_f^1 = 10$, $\theta_f^1 = 0$, $x_f^2 = 0$, $y_f^2 = 0$, $\theta_f^2 = 0$, $u_i^+ = 10$, $u_i^- = -10$, $i = 1, 2$, $t^+ = 3.5$ s, $x_1^* = 5$, $y_1^* = 8$, $r_1 = 2$, $x_2^* = 5$, $y_2^* = 2$, $r_2 = 2$, $d = 2$, $\varepsilon_0 = 0.1$, $a = 2$, $b = 2$, $\varepsilon = 0.01$, $H = 32$, $P = 256$, $P_{length} = 8$, $P_{rt} = 0.33$, $\Delta = 0.22$, $\beta = 0.2$, $\alpha = 0.3$, $K = 7$, constraints on parameter values are $q_i^- = -1$, $q_i^+ = 1$, $i = 1, \ldots, 6$, $q_7^- = -2$, $q_7^+ = 1$, $q_i^- = -10$, $q_i^+ = 10$, $i = 8, \ldots, 11$.

An obtained optimal vector of parameters was

$$\mathbf{q} = [0.7722\ \ 0.2142\ \ -0.2825\ \ -0.2968\ \ -0.23\ \ 0.3702\ \ -1.5112\ \ 0.0179\ \ 3.3965\ \ -2.96\ \ -0.5945]^T.$$

This solution provided the value of the goal function (55) $F = 0.1238$.

To obtain the solution of optimal control problem for (33) and (44) by using the Pontryagin maximum principle, the complexity of search was the following: $H = 32$, $P = 256$, $R = H$, $P_{length} = 8$, $step = 0.22$, $n = 8/0.22 = 36$, $H + nRP = 32 + 36 \times 32 \times 256 = 294,944$, i.e., the functional was calculated 294,944 times. Simulation was performed in Lazarus, an open-source Free Pascal-based software, on PC with Intel Core i7, 2.8 GHz, OS Win 7. Series of 10 runs was implemented. The CPU time for 10 runs was approx. 20 min, i.e., 1 run was approx. 2 min.

The trajectories of the robots are shown in Figure 1. On Figure 1, red circles present the static constraints. Plots of obtained control are presented on Figures 2–5. Figures 2–5 show that optimal control includes sectors of special control modes [8]. The controls u_2^1 and u_1^2 have sliding modes.

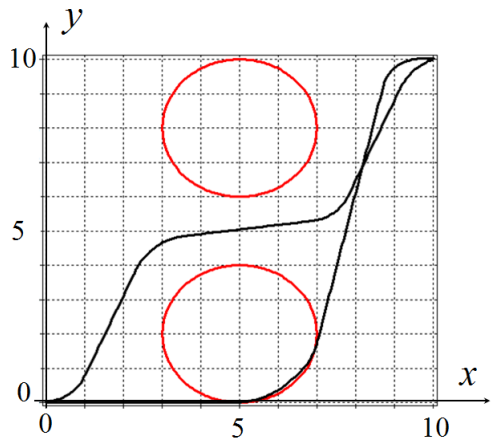

Figure 1. Trajectories of movement of robots.

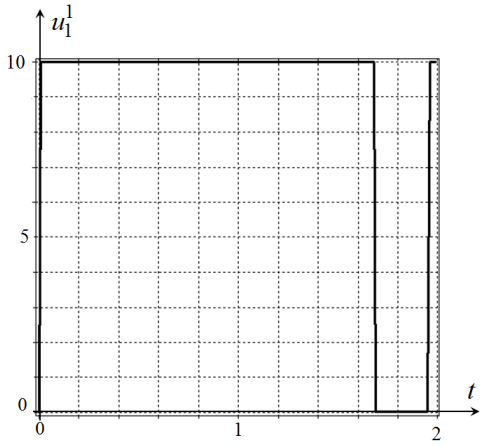

Figure 2. Control u_1^1.

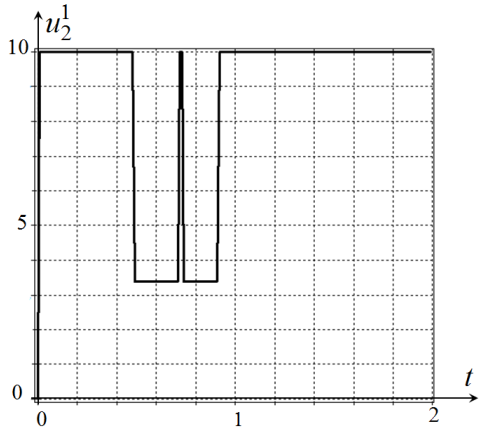

Figure 3. Control u_2^1.

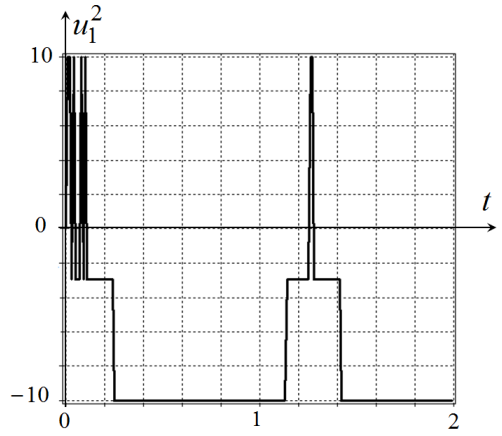

Figure 4. Control u_1^2.

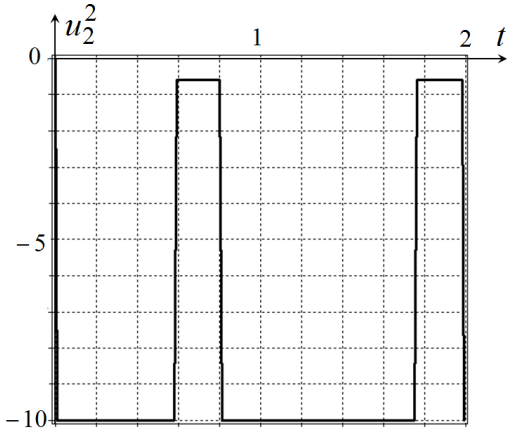

Figure 5. Control u_2^2.

7. Search of Optimal Control by Direct Method

The same problem was solved by direct numerical method. Control for each robot was searched as a piece-wise liner function on interval as follows

$$u_i^j = \begin{cases} u_i^+, \text{ if } u_i^+ \le \tilde{u}_i^j \\ u_i^-, \text{ if } \tilde{u}_i^j \le u_i^-, \\ \tilde{u}_i^j, \text{ otherwise} \end{cases} \tag{56}$$

where

$$\tilde{u}_1^1 = q_i + (q_{i+1} - q_i)\frac{(t - i\Delta t)}{\Delta t}, \tag{57}$$

$$\tilde{u}_2^1 = q_{i+K} + (q_{i+K+1} - q_{i+K})\frac{(t - i\Delta t)}{\Delta t}, \tag{58}$$

$$\tilde{u}_1^2 = q_{i+2K} + (q_{i+2K+1} - q_{i+2K})\frac{(t - i\Delta t)}{\Delta t}, \tag{59}$$

$$\tilde{u}_2^2 = q_{i+3K} + (q_{i+3K+1} - q_{i+3K})\frac{(t - i\Delta t)}{\Delta t}, \tag{60}$$

$i = 1, \ldots, K$, $t \in [i\Delta t; (i+1)\Delta t)$, $\Delta t = 0.25$,

$$K = \left\lfloor \frac{t^+}{\Delta t} \right\rfloor + 1 = \left\lfloor \frac{2.5}{0.25} \right\rfloor + 1 = 11, \tag{61}$$

Eleven time intervals were used. For each control, it was necessary to find 11 parameters of piece-wise linear function at the boundaries of intervals. Totally, we searched for forty-five parameters $\mathbf{q} = [q_1 \ldots q_{45}]^T$, forty-four parameters were for control of two robots, and q_{45} was for terminal time $t^+ + q_{45}$. Values of parameters were constrained

$$q_i^- \leqslant q_i \leqslant q_i^+, \; i = 1, \ldots, 45, \tag{62}$$

where $q_i^- = -20$, $q_i^+ = 20$, $i = 1, \ldots, 44$, $q_{45}^- = -0.8$, $q_{45}^+ = 0.8$.

The parameter search was also performed by modified SOMA. Vector of obtained parameters was as follows

$$\begin{aligned} \mathbf{q} = [&15.0129\ 17.4687\ 17.6772\ 2.2532\ 12.5488\ 9.0041\ 0.3886\ 19.9999\ 19.3899\ 19.01849 \\ &-8.8472\ -15.2669\ 10.5164\ 19.9044\ 16.5724\ 18.0328\ 17.5597\ 18.3563\ 16.8607\ 8,2100 \\ &-18.7127\ 4.3724\ -8.9377\ -0.7936\ 10.6309\ 18.9660\ 18.4597\ 18.9742\ 17.2781\ 14.6362 \\ &18.4152\ -13.7367\ -2.8521\ 17.0837\ 18.1753\ 17.5278\ 9.4618\ 19.5178\ 17.6007\ 11.3479 \\ &-0.6898\ -5.4015\ 19.9215\ 9.6850\ 0.6628]^T. \end{aligned}$$

For direct approach, when we searched for 45 parameters by modified SOMA, the complexity of the algorithm was the following: $H = 32$, $P = 1024$, $R = H$, $P_{length} = 8$, $step = 0.22$, $n = 8/0.22 = 36$, $H + nRP = 32 + 36 \times 32 \times 1024 = 1{,}179{,}680$. Simulation was performed on PC with Intel Core i7, 2.8 GHz. A series of 10 runs was implemented. The CPU time for 10 runs was approx. 3 hours and 10 min, i.e., 1 run was approx. 19 min. We used $P = 1024$ for direct approach, because the number of searched parameters was 45, and it was several times bigger than 11 parameters in the first experiment.

The obtained solution is presented on Figure 6. On Figures 7–10, the plots of direct controls are presented.

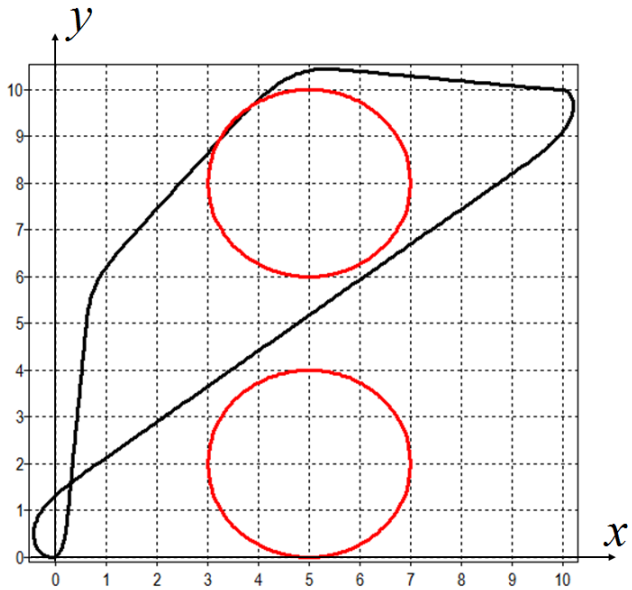

Figure 6. Trajectories of robots obtained by direct method.

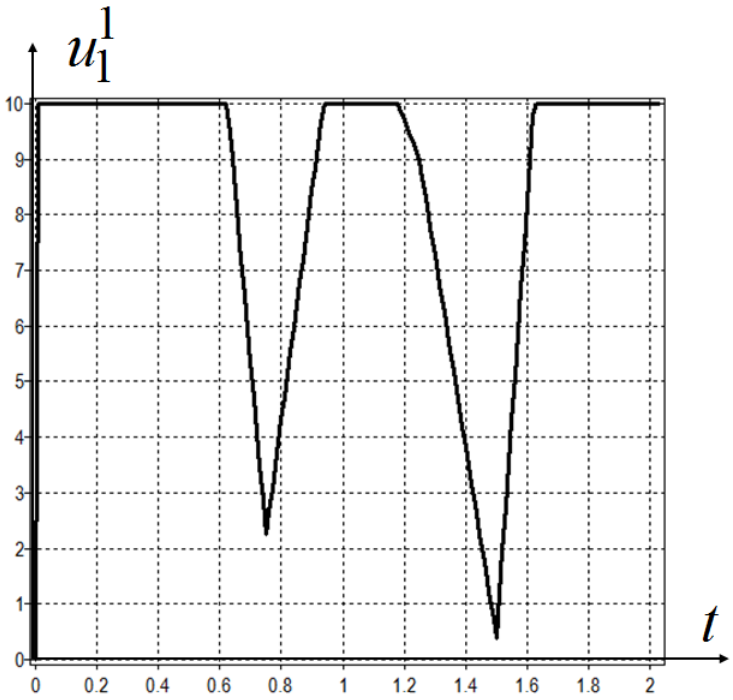

Figure 7. Direct control u_1^1.

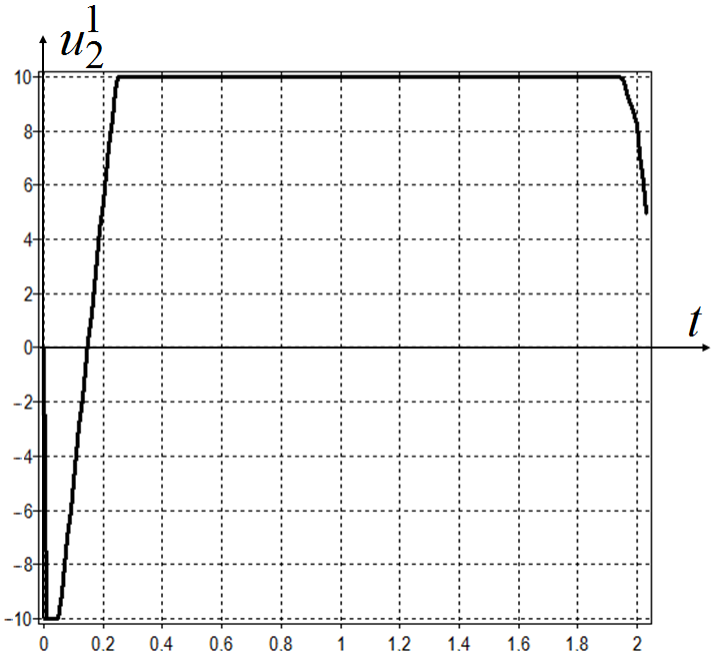

Figure 8. Direct control u_2^1.

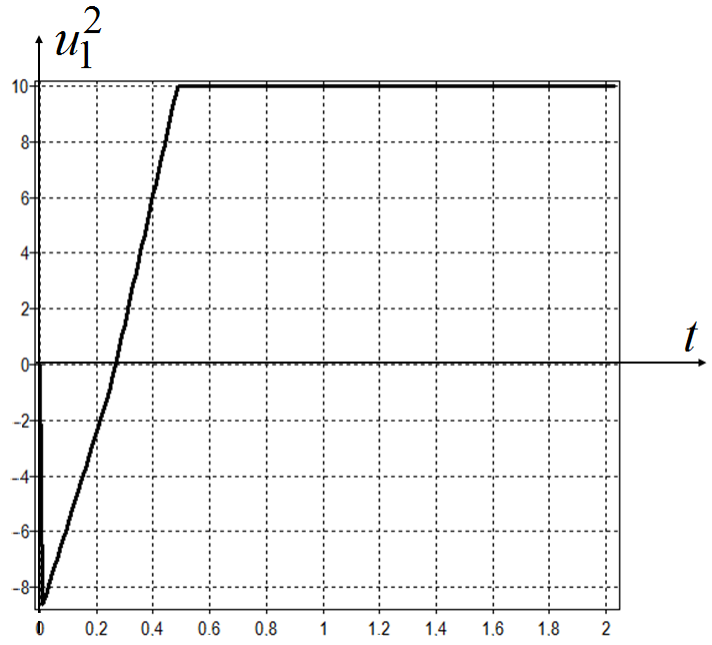

Figure 9. Direct control u_1^2.

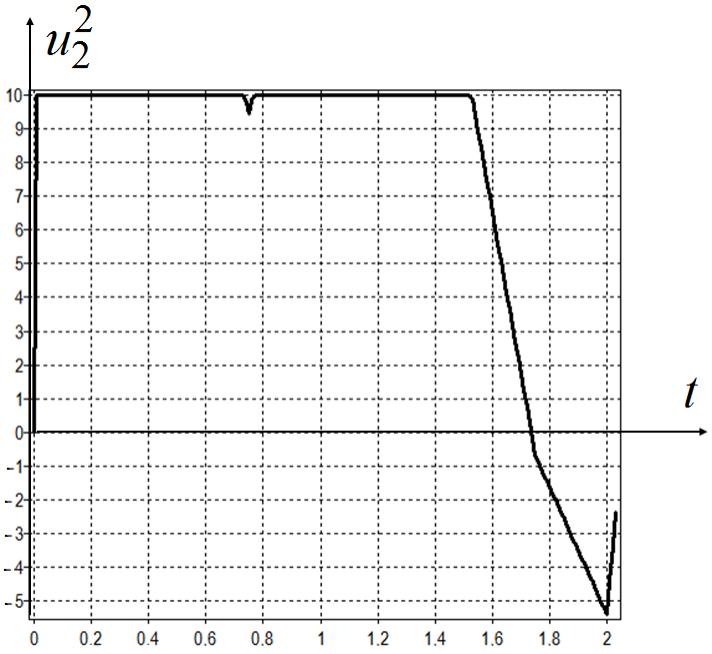

Figure 10. Direct control u_2^2.

8. An Alternative Non-Deterministic Control

One of the most important issues for swarm robotics applications is catching up with moving targets and avoiding multiple dynamic obstacles. It is complicated in that it requires an algorithm to work in real time to avoid obstacles that are standing or moving in an unknown environment, where the robot does not know their position until detecting them by sensors arranged on the robot. As an alternative to the method presented above, the use of swarm intelligence algorithms as reported in [9] can be discussed.

The paper [9] presents a method for swarm robot to catch the moving target and to avoid multiple dynamic obstacles in the unknown environment. An imaginary map is built, representing N targets, M obstacles and N robots and a swarm intelligence algorithm is used to control them so that targets are captured correctly and in the shortest time. The robot dynamics can be viewed as a flow of water moving from high to low. The flow of water is the robot trajectory that is divided into a set of points created by an algorithm called SOMA [5,6]. Simulation results are also presented to show that the obstacle avoidance and catching target task can be reached using this method. All details about those experiments are discussed in [9]. Results are also visualized in selected videos (https://zelinkaivan65.wixsite.com/ivanzelinka/videa). The typical example is in Figure 11.

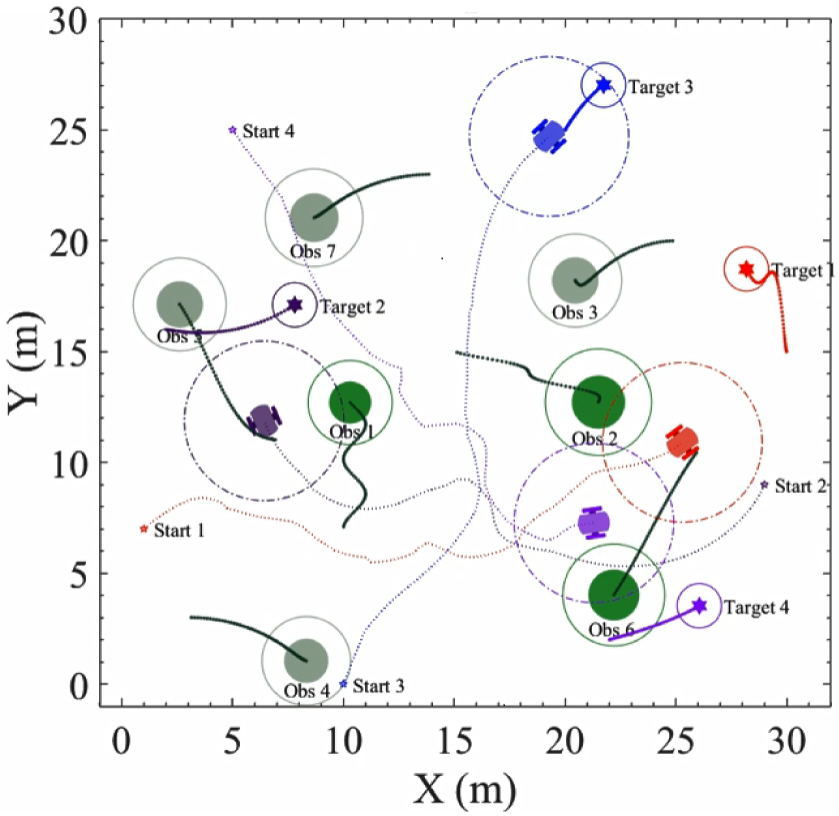

Figure 11. Swarm robot control by SOMA.

Besides the long-standing methods such as potential field method [10,11], and the vector field histogram [12], several new methods such as "follow the gap method" [13], and barrier function [14], or artificial intelligence methods such as genetic algorithm [15], neural network [16], and fuzzy logic [17,18] also demonstrate their effectiveness. Among the methods of artificial intelligence used to solve the problem as a function optimization problem, the self-organizing migrating algorithm (SOMA) emerges as a fast, powerful and efficient algorithm [5,6].

9. Discussion

The optimal control problem for two mobile robots with phase constraints was considered. To solve the problem, an approach based on the Pontryagin maximum principle was used. The mathematical model of robots include linear control in the right parts of differential equations; that is why the optimal control has sectors of special control modes. It should be noted that we used two robots to test the proposed

technology and partially to test methodology. A larger group is required to fully test the proposed methodology and it will be our future research, but in the case of many robots, the optimization problem will go on backstage and the collision avoidance will become the real problem.

To solve a boundary-value problem and search of initial conditions of conjugate variables, the modified SOMA was used. Additional parameters for terminal conditions check and control in special modes were introduced.

The optimal control problem was also solved by direct approach. Control time was divided into intervals, and control at each interval was a piece-wise linear function. Additional parameter was also a time of terminal conditions check. The direct approach showed another character of objects movement.

The aim of this paper was to show for the first time how modern evolutionary algorithms can be applied to solution of boundary-value problems that occur when we solve the optimal control problem by an indirect method based on the Pontryagin maximum principle. Other known applications of evolutionary algorithms were mainly with direct approach [4].

Thus, one can conclude that the considered problem is multimodal and application of evolutionary algorithms to both direct and indirect approaches is expedient and prospective. The next research will be focused on an extensive comparative study of classical and swarm control based methods.

10. Patents

Certificate of software registration No.2020619668 Diveev A.I., Sofronova E.A. "Optimal control of group of robots based on Pontryagin maximum principle" 21 August 2020.

Certificate of software registration No.2020619960 Diveev A.I., Sofronova E.A. "Optimal control of group of robots by direct approach based on piece-wise linear approximation", 26 August 2020.

Author Contributions: Formal analysis, A.D.; Investigation, A.D., E.S.; Methodology, I.Z.; Software, E.S.; Supervision, I.Z.; Validation, E.S.; Writing—original draft, A.D.; Writing—review and editing, E.S. All the authors have equal contribution. All authors have read and agreed to the published version of the manuscript.

Funding: This research is supported by the Ministry of Science and Higher Education of the Russian Federation, project No 075-15-2020-799.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Pontryagin, L.S.; Boltyanskii, V.G.; Gamkrelidze,R.V.; Mishchenko, E.F. *The Mathematical Theory of Optimal Processes*; Interscience: New York, NY, USA, 1962.
2. Arutyunov,A.V.; Karamzin, D.Y.; Pereira, F.L. Maximum principle in problems with mixed constraints under weak assumptions of regularity. *Optimization* **2010**, *59*, 1067–1083. [CrossRef]
3. Karamzin, D.; Pereira, F.L. On a Few Questions Regarding the Study of State-Constrained Problems in Optimal Control. *J. Optim. Theory Appl.* **2019**, *180*, 235–255. [CrossRef]
4. Diveev, A.I.; Konstantinov, S.V. Study of the practical convergence of evolutionary algorithms for the optimal program control of a wheeled robot. *J. Comput. Syst. Sci. Int.* **2018**, *57*, 561–580. [CrossRef]
5. Zelinka, I. SOMA—Self organizing migrating algorithm. In *New Optimization Techniques in Engineering*; Babu, B.V., Onwubolu, G.; Eds.; Springer: Berlin/Heidelberg, Germany, 2004; Volume 33, Chapter 7, ISBN: 3-540-20167X.
6. Zelinka, I. SOMA—Self organizing migrating algorithm. In *Self-Organizing Migrating Algorithm. Methodology and Implementation*; Series: Studies in Computational Intelligence; Davendra, D., Zelinka, I., Eds.; Springer: Cham, Switzerland, 2016; pp. 3–39. ISBN 978-3-319-28161-2.
7. Diveev, A.; Sofronova, E.; Shmalko, E. Modified SOMA for Optimal Control Problem. In Proceedings of the 2019 IEEE Congress on Evolutionary Computation, (CEC 2019), Wellington, New Zealand, 10–13 June 2019; pp. 2894–2899.

8. Gabasov, R.; Kirillova, F.M. Optimal control with special sectors. *Autom. Remote Control* **1969**, *30*, 1554–1563.
9. Diep, Q.B.; Zelinka, I.; Senkerik, R. An algorithm for swarm robot to avoid multiple dynamic obstacles and to catch the moving target. In *Artificial Intelligence and Soft Computing. ICAISC 2019*; Lecture Notes in Computer Science; Rutkowski, L., Scherer, R., Korytkowski, M., Pedrycz, W., Tadeusiewicz, R., Zurada, J.M., Eds.; Springer: Cham, Switzerland, 2019; pp. 666–675.
10. Borenstein, J.; Koren, Y. Real-time obstacle avoidance for fast mobile robots. *IEEE Trans. Syst. Man Cybern.* **1989**, *19*, 1179–1187. [CrossRef]
11. Koren, Y.; Borenstein, J. Potential field methods and their inherent limitations for mobile robot navigation. In Proceedings of the IEEE International Conference on Robotics and Automation, Sacramento, CA, USA, 9–11 April 1991; pp. 1398–1404.
12. Borenstein, J.; Koren, Y. The vector field histogram-fast obstacle avoidance for mobile robots. *IEEE Trans. Robot. Autom.* **1991**, *7*, 278–288. [CrossRef]
13. Sezer, V.; Gokasan, M. A Novel obstacle avoidance algorithm: "Follow the gap method". *Robot. Auton. Syst.* **2012**, *60*, 1123–1134. [CrossRef]
14. Chen, Y.; Peng, H.; Grizzle, J. Obstacle avoidance for low-speed autonomous vehicles with barrier function. *IEEE Trans. Control. Syst. Technol.* **2017**, *99*, 1–13. [CrossRef]
15. Hu, Y.; Yang, S.X. A knowledge based genetic algorithm for path planning of a mobile robot. *IEEE Int. Conf. Robot. Autom.* **2004**, *5*, 4350–4355.
16. Duguleana, M.; Mogan, G. Neural networks based reinforcement learning for mobile robots obstacle avoidance. *Expert Syst. Appl.* **2016**, *62*, 104–115. [CrossRef]
17. Reignier, P. Fuzzy logic techniques for mobile robot obstacle avoidance. *Robot. Auton. Syst.* **1994**, *12*, 143–153. [CrossRef]
18. Zavlangas, P.G.; Tzafestas, S.G.; Althoefer, K. Fuzzy obstacle avoidance and navigation for omnidirectional mobile robots. In Proceedings of the European Symposium on Intelligent Techniques, Aachen, Germany, 14–15 September 2000; pp. 375–382.

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

© 2020 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (http://creativecommons.org/licenses/by/4.0/).