Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2012, Article ID 142060, 16 pages
doi:10.1155/2012/142060

Research Article

Toolgraph Design of Optimal and Feasible Control Strategies for Time-Varying Dynamical Systems

Z. Kowalczuk and K. E. Olinski

Department of Decision Systems (WETI), Gdansk University of Technology, Narutowicza 11/12, 80-952 Gdansk, Poland

Correspondence should be addressed to Z. Kowalczuk, kova@pg.gda.pl

Received 30 March 2012; Accepted 21 July 2012

Academic Editor: Zoran Gajic

The paper presents a new method for designing optimal and feasible control strategies for time-variant dynamical processes. The key point of the presented idea lies in utilizing a flow graph structure for representing pertinent properties of the autonomous dynamics of a given dynamical process in a time-and-state space, which is composed of certain elementary segments. The structure is referred to as a time-and-state space toolgraph. In the procedure, each segment of the temporary state space is assigned a node of the time-and-state space toolgraph. The flow values are proportional to the cost of driving the operational point of the dynamical process between the centers of adjacent segments. Any of the discrete optimization algorithms can be applied to search for a cheapest path connecting the initial and terminal points of the sought optimal piecewise-linear trajectory of the operational points in the considered time-and-state space. Additional assumptions or restrictions concerning arbitrary forbidden zones for the operational points can be easily taken into account. In such cases the nodes representing the segments partially or entirely belonging to the finite forbidden zones are deposed from the toolgraph structure.

1. Introduction

The general idea of representing a continuous workspace by means of a graph structure has recently been explored, for instance, for searching optimal trajectories of robot movements in 2D [1] and 3D [2] workspaces, as well as for designing optimal control in multidimensional spaces [3].

The presented method is a consequent extension of our research track of designing optimal and feasible control by means of discrete optimization methods [4–6]. In particular, certain deterministic time variations of the controlled process are taken into consideration by extending the common state space with an additional time dimension.

The algorithm consists in finding a trajectory in such an extended state space (called henceforth the time-state space), which should satisfy two principal prerequisites. First, the sought trajectory circumvents certain finite areas (called the forbidden zones) which can, for instance, represent some failures previously identified. Second, the autonomous dynamics of the process is used in the process of minimizing the adopted indicator of control cost, which is a part of the problem definition and an input to a corresponding minimization procedure. The procedure starts with segmenting the defined operational workspace, which is a subset of the time-and-state space taken into consideration during the process of optimal trajectory design. For each segment a set of representative values is determined. The representative values (representatives) are all the quantities which are necessary to calculate the cost of an operational point movement in a given segment area. The next step is based on a formulation of a flow graph structure (called toolgraph), which reflects the arrangement of the segments along with their appropriate representatives. Any discrete optimization algorithm that performs optimal search for the cheapest path can be utilized for searching the time-and-state space toolgraph for the cheapest path connecting the initial and terminal nodes representing the segments, which contain initial and terminal points of the sought trajectory, respectively.

This paper is organized as follows: Section 2 gives a formal description of the presented idea and in Section 3 an extended example is presented. The paper ends by the Conclusions section.

2. Problem and Solution Description

This section provides some basic definitions (confer [5]) and gives a raw description of the presented method.

2.1. Problem Statement

Definition 2.1 (autonomous and forced dynamics). A dynamical and time-variant process is described by the following function:

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t), \tag{2.1}$$

$$\mathbf{f} : \mathcal{R}^l \times \mathcal{R}^m \times \mathcal{R} \longrightarrow \mathcal{R}^l, \tag{2.2}$$

where $\mathbf{x} = [x_1, \ldots, x_l]$ is a state vector and $\mathbf{u} = [u_1, \ldots, u_m]$ is a control vector, and $l, m \in \mathcal{N}$. Let the *autonomous dynamics* be described by

$$\dot{\mathbf{x}}_x(t) = \mathbf{f}_x(\mathbf{x}(t), t) \tag{2.3}$$

and let the *forced dynamics* be defined by a simple centroaffine transformation

$$\dot{\mathbf{x}}_u(t) = \mathbf{f}_u(\mathbf{x}(t), \mathbf{u}(t)) = \mathbf{F}\mathbf{u}(t), \tag{2.4}$$

where $\mathbf{F}$ is a fixed (time-invariant) $l \times m$ input control matrix, and the above contributors satisfy the following additivity condition:

$$\dot{\mathbf{x}}(t) = \dot{\mathbf{x}}_x(t) + \dot{\mathbf{x}}_u(t), \tag{2.5}$$

where $\mathbf{x}_x$ and $\mathbf{x}_u$ are two, autonomous and forced state components.

Definition 2.2 (operational subspace P). A subset $P = D_1 \times D_2 \times \cdots \times D_l$, where D_i, for $i = 1, \ldots, l$, represents a range along the axis x_i, which is taken into account while seeking an optimal trajectory, is said to be an operational subspace.

Definition 2.3 (forbidden zone Z). A subset Z of P prohibited for operational points is referred to as a forbidden zone. This means that the sought optimal trajectory cannot enter it.

Definition 2.4 (system trajectory $E(\mathbf{u})$). Any sequence $E(\mathbf{u})$ of consecutive states of a given dynamical system for a feasible control input $\mathbf{u}$ from a set $\mathcal{U}$ is said to be a system's trajectory or a trajectory of operational points in the system state space $P \setminus Z$.

Definition 2.5 (transition vector Λ). A transition vector Λ is an ordered set of two elements $\{\mathbf{x}_0, \mathbf{x}_k\}$

$$\Lambda = \{(\mathbf{x}_0, \mathbf{x}_k) : \mathbf{x}_0, \mathbf{x}_k \in E(\mathbf{u})\}, \tag{2.6}$$

where $\mathbf{x}_0$ is the first element and $\mathbf{x}_k$ is the last element of the sought (optimal) trajectory.

Definition 2.6 (cost function $J(E(\mathbf{u}))$ of the system and control trajectory). Let $\mathcal{A}$ be a set of all trajectories $E \in \mathcal{A} \subset P \setminus Z$. Any real function of the class $\mathcal{A} \to \mathcal{R}^+$ is said to be a cost function $J(E(\mathbf{u}))$ of this trajectory.

Definition 2.7 (optimal trajectory E^*). Let $\Xi \subset \mathcal{A}$ be the subset of all possible trajectories that perform the desired transition Λ (i.e., which start at x_0 and terminate at x_k). We say that a trajectory E^* is (costly) optimal if it satisfies the following conditions:

$$\begin{aligned} &\forall_{E \in \Xi} J(E) \geq J(E^*), \\ &\forall_{\mathbf{x} \in E^*} \mathbf{x} \in P \setminus Z. \end{aligned} \tag{2.7}$$

Definition 2.8 (autonomous dynamics map M). The image $\dot{\mathbf{x}}_x$ of the whole space P in the transformation $\mathbf{f}_x$ is said to be an autonomous dynamics map $M(= \dot{\mathbf{x}}_x)$.

Our ultimate objective is to find the optimal and feasible control $\mathbf{u}^*(t)$ along with its corresponding state trajectory $E^* = E(\mathbf{u}^*(t)) \in (P \setminus Z)$, which implements the complete transition Λ of the dynamical system (2.1) from its initial state $\mathbf{x}(t_0) = \mathbf{x}_0$ to the target state $\mathbf{x}(t_k) = \mathbf{x}_k$ and at the same time minimizes the cost functional of the following form:

$$J(E(u)) = \int_{t_0}^{t_k} \left(\sum_{i=1}^{m} \beta_i |u_i(t)| \right) dt, \tag{2.8}$$

where β_i, $i = 1, \ldots, m$ are nonnegative factors often used for weighting, and t_k is a suitable transition-time interval resulting from the applied optimal control procedure.

2.2. Solution Method

For the dynamical systems described in Definition 2.1, we consider its simple discrete-time variability of the computationally-convenient jump type allowing for the state-space system approximation scheme depicted in Figure 1.

Leaving apart the initial conditions defined for a certain t_0, let us consider consecutive continuous-time computational time moments t_n, for $n = 1, \ldots, N$, with $t_n - t_{n-1} = h = \Delta t$, being a computation step (or a simulation insight), where $N \in \mathcal{N}$ is a discrete-time horizon under consideration.

Definition 2.9 (temporary workspace P_n). At each of the above-mentioned computational moments, a separate temporary duplicate P_n of the operational subspace P will be associated, as shown in Figure 1.

Definition 2.10 (snapshots M_n of the dynamics map). The multiplication of the subspace P allows us to define separate snapshots M_n of the dynamics map M assigned to consecutive time moments $n = 0, \ldots, N$.

The map snapshots M_n are thus assigned to each of the temporary workspaces P_n. This is, however, difficult to be portrayed on the composite scheme given in Figure 1. Instead, there are only shaded planes illustrating this effect. A better portray of map snapshots is given in the following example (see Figure 2).

Definition 2.11 (continuous-time dynamic system with discrete-time variability). The temporary (potentially time-variant) system dynamics composed of the autonomous and forced dynamics can be defined in the computational moments as

$$\dot{\mathbf{x}}(t_n) = \dot{\mathbf{x}}_x(t_n) + \dot{\mathbf{x}}_u(t_n) \tag{2.9}$$

which by the power of (2.3) and (2.4) allows for simple modeling of the system variability (which can be arbitrary induced by redefining the functions $\mathbf{f}_x$ and $\mathbf{f}_u$).

Note that the snapshots M_n of the map are included in (2.9) as $\dot{\mathbf{x}}_x$, whereas the control part $\dot{\mathbf{x}}_u$ represents the forced component (2.4) of the complete state derivative $\dot{\mathbf{x}}$. Clearly, this means that not just the snapshot M_n forms the derivative of the state.

Definition 2.12 (process computation). The procedure of calculating the operational-point trajectory in the state-space for the assumed discrete-time variability model (which can also be treated as a kind of real-system approximation) consists of the following derivative-based prediction and numerical integration steps:

(i) take $\dot{\mathbf{x}}(t_n)$ and treat it as a local estimation of the system dynamics for the future time interval $\langle t_n, t_{n+1})$,

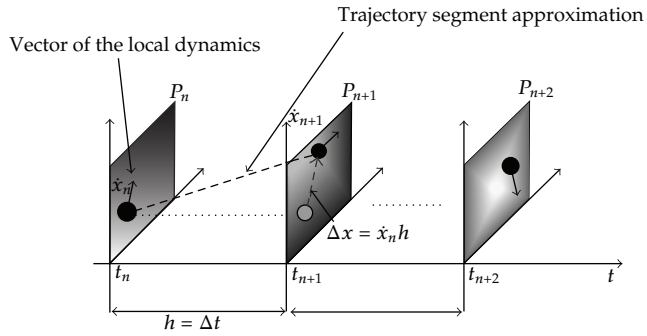


Figure 1: Continuous-time dynamic system with discrete-time variability.

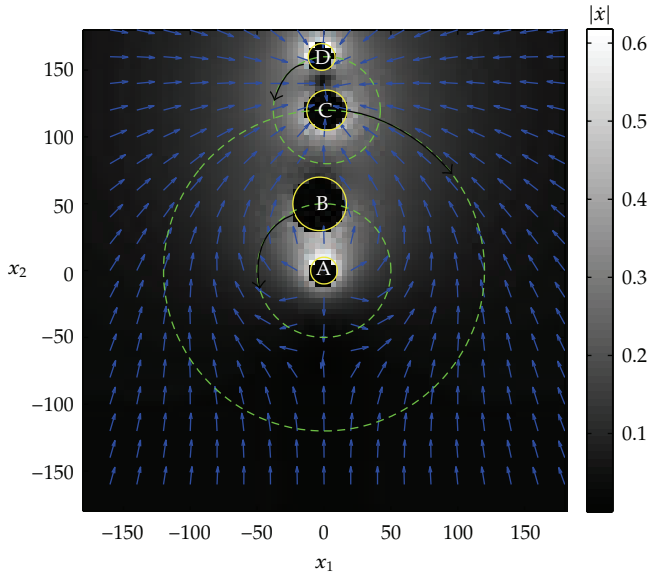


Figure 2: Autonomous dynamics map of the process for the time moment $t = 0$. Background color: the absolute value of the autonomous dynamics vector; small arrows: the direction of the autonomous dynamics vector; dotted lines: the orbits of the centers of attraction; solid lines: the borders of the forbidden zones; big arrows: the direction of the movement of the centers of attraction.

(ii) calculate the next state vector for the time moment t_{n+1} as

$$\mathbf{x}(t_{n+1}) = \mathbf{x}(t_n) + h\dot{\mathbf{x}}(t_n). \tag{2.10}$$

Note that the rectangular integration scheme applied above for modeling the time-variability of dynamical systems is widely utilized in the procedures of computer simulation of variant or invariant continuous-time systems [7].

Definition 2.13 (segmentation of the temporary operational subspaces). Segmentation of the temporary operational subspace P_n consists in dividing it, for the time moments t_n,

$n = 0, \ldots, N$, into a set $\mathcal{S}_n$ of S segments denoted as Φ_j^n, for $j = 1, \ldots, S$, according to the following relation:

$$P_n = \bigcup_{j=1}^{S} \Phi_j^n, \tag{2.11}$$

where each pair of segments for any discrete time n satisfies the following separability condition:

$$\left(\Phi_i^n - \delta\Phi_i^n\right) \cap \left(\Phi_j^n - \delta\Phi_j^n\right) = \emptyset \tag{2.12}$$

for $i, j = 1, \ldots, S$; $i \neq j$, where δ denotes the closure of the corresponding set. Clearly, the size of the segments is typically selected in accordance with some fidelity criteria of the applied discrete approximation of the original process.

For the computational purposes we need a definition of a segment neighborhood and a generic concept of a distance (norm). Let us thus assume the common Euclidean norm $\|\cdot\|_e$ and the following definition.

Definition 2.14 (segment's neighborhood). Any segment Φ_B^n is in the neighborhood of another segment Φ_A^n, that is, $\Phi_B^n \in \mathcal{N}(\Phi_A^n)$, if their geometrical centers $\mathbf{x}_B$ and $\mathbf{x}_A$ satisfy a simple neighboring condition:

$$\|\mathbf{x}_B - \mathbf{x}_A\|_e \leq r, \tag{2.13}$$

where $\|\cdot\|_e$ means the Euclidean norm, and r is a fixed neighborhood size.

Definition 2.15 (representatives). Any variable used in optimization and assigned to a segment is said to be a representative.

The autonomous dynamics map of Definition 2.8, being the derivative of the vector function $\mathbf{x}(t)$, makes a principal source of candidates for a rational selection of the segment representatives. Note that in accordance to the common idea of domain and codomain of the transformation $\mathbf{f}_x$, from a practical implementation viewpoint, the map means a kind of doubling of the problem dimensioning with respect to P_n. This fact is merely represented in Figure 1 by the colors (or rather shades) applied to different temporary subspaces P_n. When the local evolution $\dot{\mathbf{x}}$ of some of the coordinates of the state vector is not relevant for the optimization considered, the co-domain can be reduced. At the same time the designer can introduce other representatives relevant for the optimization procedure and criterion.

Definition 2.16 (predecessor and successor workspaces). For any pair (P_n, P_{n+1}), P_n is called a predecessor and P_{n+1} is referred to as a successor (on the time scale).

Definition 2.17 (time-and-state operational space). The time-and-state operational (sub)space, concerning all the time moments t_n, $n = 0, \ldots, N$, is composed as follows:

$$\mathcal{P} = \{P_1, P_2, \ldots, P_n, \ldots, P_N\}. \tag{2.14}$$

In spite of the above definition of the (global) time-and-state operational space, and according to the scheme depicted in Figure 1, the principal optimization takes place in particular temporary workspaces P_n. Clearly, based on the procedure judgment shown in Figure 1, the time domain has to be treated separately.

Definition 2.18 (state-space graph $\mathcal{G}_n$). A flow graph representing the arrangement of the segments of the temporary operational workspace P_n, $n = 0, \ldots, N$, is said to be a state-space graph. According to the common idea [8] of flow graphs:

$$\mathcal{G}_n = (\mathcal{W}_n, \mathfrak{D}_n, \mathcal{K}_n), \tag{2.15}$$

we interpret its basic structure as a set $\mathcal{W}_n$ of nodes, each of which is assigned a set of the values of the representatives of its corresponding segments. Only the edges connecting two nodes representing segments located in their mutual neighborhood form a set $\mathfrak{D}_n$ of the edges. Each edge $d \in \mathfrak{D}_n$ is assigned a flow value $k \in \mathcal{K}_n$.

Definition 2.19 (time-and-state space toolgraph). According to the definition of the time-and-state operational space, and based on the state-space graphs of all the workspaces P_n, a time-and-state space toolgraph is composed by adding costless, directed edges connecting all the pairs of equivalent (in the subspace geometry) segments in the predecessor and successor workspaces.

2.3. Algorithm Description

Basically, the presented algorithm takes the above-defined principal notions (the time-and-state space $\mathcal{D}$ composed of the temporary operational workspaces P_n, the transition vector Λ, and the segmentation form) to minimize the adopted cost function $J(E(\mathbf{u}))$. Optionally, we can determine the feasible control set $\mathcal{U}$ and the forbidden zone Z. The latter can be used to represent some constraints portraying, for instance, faults previously detected in the system.

Definition 2.20 (the toolgraph optimization algorithm). Initialization with the input data: dynamic process description, operational time-and-state space, forbidden zones, initial and terminal trajectory points, and time horizon is as follows.

(i) Step 1. Segmentation.

(ii) Step 2. Determination of the representatives.

(iii) Step 3. Are the fidelity conditions met?

  (1) If yes, go to Step 4.
  (2) If no, resize the segments, introduce new forbidden zones, and go to Step 1.

(iv) Step 4. Creation of the toolgraph.

(v) Step 5. T-graph search for the cheapest path connecting the initial and the terminal nodes.

(vi) Step 6. Has the cheapest path been found?

  (1) If yes, go to Step 7.
  (2) If no, terminate the algorithm.

(vii) Step 7. The reference trajectory synthesis (based on the path found in the previous step).

As can be seen in Definition 2.20, we start the procedure by segmenting the temporary subspaces P_n which results in a limited set of S segments, for $n = 0, \ldots, N$. Each segment of a set of representatives is assigned. Certainly, the size of the segments is chosen in accordance to an assumed discretization error, or fidelity index (discussed later on in the examples).

Next, the state space graphs are formed. According to Definition 2.18, each node from $\mathcal{W}_n$ is associated with a certain segment of P_n. Its edges connect only nodes representing the segments neighboring in P_n. The edge flow values are determined according to the adopted function cost. Consequently, the time-and-state space toolgraph is built as a composition of the "consecutive" state-space graphs according to Definition 2.19.

For practical optimization problems, any discrete optimization algorithm, which is suitable for finding the cheapest path between two selected nodes, can be utilized. If such a path exists, it represents a sequence of segments, which should be visited by the optimal trajectory. This sequence of segments implies a reference trajectory, which can be tracked by an executive controller in a procedure of a suboptimal fulfillment of the stated control task.

3. Optimization Examples

This section provides an illustrative example which utilizes the above-mentioned formal definitions and the optimization algorithm. The minimized criterion is given by (2.8) with the restriction that the resulting trajectory is piecewise linear in the considered time-and-state space. The example concerns a two-dimensional time-variant dynamical process described in the following subsection.

3.1. Process Description

The exemplary process is described by the following set of equations:

$$\begin{aligned}
\dot{x}_1(t) &= \frac{GM_A x_1}{d_A(x_1(t), x_2(t))^2} - \frac{GM_B(x_1(t) - x_{1B}(t))}{d_B(x_1(t), x_2(t))^2} \\
&\quad - \frac{GM_C(x_1(t) - x_{1C}(t))}{d_C(x_1(t), x_2(t))^2} - \frac{GM_D(x_1(t) - x_{1D}(t))}{d_D(x_1(t), x_2(t))^2}, \\
\dot{x}_2(t) &= \frac{GM_A x_2}{d_A(x_1(t), x_2(t))^2} - \frac{GM_B(x_2(t) - x_{2B}(t))}{d_B(x_1(t), x_2(t))^2} \\
&\quad - \frac{GM_C(x_2(t) - x_{2C}(t))}{d_C(x_1(t), x_2(t))^2} - \frac{GM_D(x_2(t) - x_{2D}(t))}{d_D(x_1(t), x_2(t))^2},
\end{aligned} \tag{3.1}$$

where $G = 6.673 \times 10^{-3}$, $M_A = 7.9891 \times 10^2$, $M_B = 4.8685 \times 10^2$, $M_C = 8.9736 \times 10^2$, and $M_D = 7.3476 \times 10^2$ are the process constant parameters. The auxiliary model variables and functions given above are defined as follows:

$$\begin{aligned}
d_A(x_1(t), x_2(t)) &= \sqrt{(x_1(t))^2 + (x_2(t))^2}, \\
d_B(x_1(t), x_2(t)) &= \sqrt{(x_1(t) - x_{1B}(t))^2 + (x_2(t) - x_{2B}(t))^2},
\end{aligned}$$

$$d_C(x_1(t), x_2(t)) = \sqrt{(x_1(t) - x_{1C}(t))^2 + (x_2(t) - x_{2C}(t))^2},$$
$$d_D(x_1(t), x_2(t)) = \sqrt{(x_1(t) - x_{1D}(t))^2 + (x_2(t) - x_{2D}(t))^2},$$
(3.2)

where

$$\begin{aligned} x_{1B}(t) &= R_B \sin \alpha_B(t), \qquad x_{2B}(t) = R_B \cos \alpha_B(t), \\ x_{1C}(t) &= R_C \sin \alpha_C(t), \qquad x_{2C}(t) = R_C \cos \alpha_C(t), \\ x_{1D}(t) &= R_D \sin \alpha_D(t) + R_C \sin \alpha_C(t), \\ x_{2D}(t) &= R_D \cos \alpha_D(t) + R_C \cos \alpha_C(t), \\ R_B &= 50, \\ R_C &= 120, \\ R_D &= 40 \end{aligned} \tag{3.3}$$

and α_B, α_C, α_D are the internal process state variables characterized by derivatives

$$\begin{aligned} \dot{\alpha}_B(t) &= \Omega_B, \\ \dot{\alpha}_C(t) &= \Omega_C, \\ \dot{\alpha}_D(t) &= \Omega_D \end{aligned} \tag{3.4}$$

of the following constant values: $\Omega_B = 65.306 \times 10^{-3}$, $\Omega_C = 17.56 \times 10^{-3}$, $\Omega_D = 96.728 \times 10^{-3}$.

The autonomous dynamics map of the described process is shown in Figure 2, where certain three (B, C, D) centers of attraction can be seen, which circulating with the angular velocities ($\Omega_B, \Omega_C, \Omega_D$, resp.) around the center (A) are the source of process time-variability. Additionally, the forbidden zones Z_A, Z_B, Z_C, Z_D are defined in the forms of circular areas, which have the radii $r_A = 10$, $r_B = 20$, $r_C = 15$, and $r_D = 10$, respectively. The midpoints of the forbidden zones cover the midpoints of the respective centers of "attraction."

3.2. *Algorithm Implementation*

In theory, an infinite time horizon of optimization is a simple and trivial solution for the time horizon setting. In computer applications, we prefer the concept of "sufficiently" large horizons, which is determined according to our prior knowledge on the optimization process and the limitations of available computing power, whereas the time step size was chosen empirically so as to obtain the desired "granularity" of the solution.

According to Definition 2.13, the segmentation procedure consists of dividing the operational subspaces P_n, $n = 0, \ldots, N$, being, in this case, hypercubes described by $D_1 \times D_2 = (-250, 250) \times (-250, 250)$, into the set of segments of diameters $(r_{x_1} = 10, r_{x_2} = 10)$, all located

along the time axis at a step size $\Delta t = 10$. For each segment the representative values are calculated as the averages of the autonomous dynamics computed for particular directions:

$$F_{\mathrm{AVR}\Phi^n_{j,i}} = \frac{1}{N_p}\sum_{k=1}^{N_p} F_{\mathrm{DYN}^n_{i,k}}, \tag{3.5}$$

where $F_{\mathrm{DYN}^n_{i,k}}$ denotes the kth sample of the coordinate x_i of the autonomous dynamics map for the discrete time moments n, and $N_p = 25$ denotes the number of samples per segment in this following examples.

The fidelity of the applied discrete representation for the segment $\Phi^n_{j,i}$ is assessed as

$$e_{\Phi^n_{j,i}} = \sum_{k=1}^{N_p} \left| F_{\mathrm{DYN}^n_{i,k}} - F_{\mathrm{AVR}\Phi^n_{j,i}} \right|. \tag{3.6}$$

If $e_{\Phi^n_{j,i}}$ exceeds some given threshold, which can be an empirically estimated value, we assume that the segment $\Phi^n_{j,i}$ along with its representatives constitutes an infeasible representation of the autonomous dynamics in the area of the time-and-state space occupied by this segment. In such cases either the size of the segment is changed or the segment is included to the forbidden zone (as an area of high dynamics).

Having defined the neighborhood of the segments, the construction of the state-space graphs and time-and-state space toolgraph comes according to Definitions 2.18 and 2.19, respectively. The flow values are determined as the cost of driving the operational point between each two neighboring segments according to (2.8).

The last stage consists in searching the above-defined time-and-state space toolgraph for the cheapest path connecting the initial and the terminal node. We can use, for instance, the Dijkstra's algorithm [9] for this purpose.

3.3. Problem 1

The first problem concerns finding a trajectory of the operational point, which minimizes (2.8), with the initial location at $(x_{1o} = -250, x_{2o} = -250, t_0 = 0)$ and the terminal point (x_{1k}, x_{2k}, t_k) satisfying the following condition:

$$\begin{gathered} \sqrt{(x_{1k} - x_{1D})^2 + (x_{2k} - x_{2D})^2} \le 1.5 r_d, \\ (x_{1k}, x_{2k}) \notin Z_D, \\ t_k \le t_N, \end{gathered} \tag{3.7}$$

where the assumed time horizon is $t_N = 2000$. The results are presented in Figure 3. The total cost indicator is equal to $J_1 = 43.454 \times 10^{-2}$. It can be clearly seen that the optimal strategy consists in keeping the operational point at the initial location till a certain time moment $(t_* = 1405)$ and then to start the transition process.

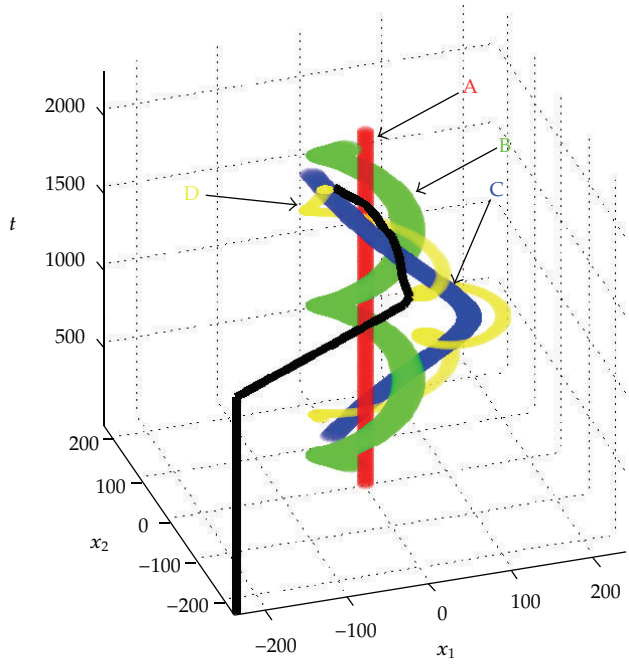


Figure 3: Resulting optimal trajectory for Problem 1 (solid black line); the forbidden zones of the respective centers of attractions: *A*, *B*, *C*, *D* (colored areas).

3.4. Problem 2

In this case the time horizon was shortened to $t_N = 1600$, which resulted in a higher value $J_2 = 52.494 \times 10^{-2}$ of the total cost indicator as compared to the one of Problem 1. The remaining parameters are the same as in Problem 1. The results are depicted in Figure 4.

3.5. Problem 3

Further shortening of the time horizon $t_N = 1000$ leads to a greater control cost $J_3 = 150.744 \times 10^{-2}$ with the trajectory of the operational point depicted in Figure 5.

3.6. Problem 4

This time the problem of driving the operational point from the initial condition ($x_{1o} = 0$, $x_{2o} = 0$, $t_0 = 0$) to the terminal point ($x_{1k} > 180$, $x_{2k} > 180$, $t_k \leq t_N$) is considered. The time horizon is $t_N = 1000$. The resulting total cost is equal to $J_4 = 275.368 \times 10^{-2}$. The designed trajectory is presented in Figure 6.

3.7. Problem 5

This case is a variation of Problem 4 with the time horizon lengthened to $t_N = 4000$, which resulted in a reduction of the total cost $J_5 = 202.905 \times 10^{-2}$. The computed trajectory is presented in Figure 7.

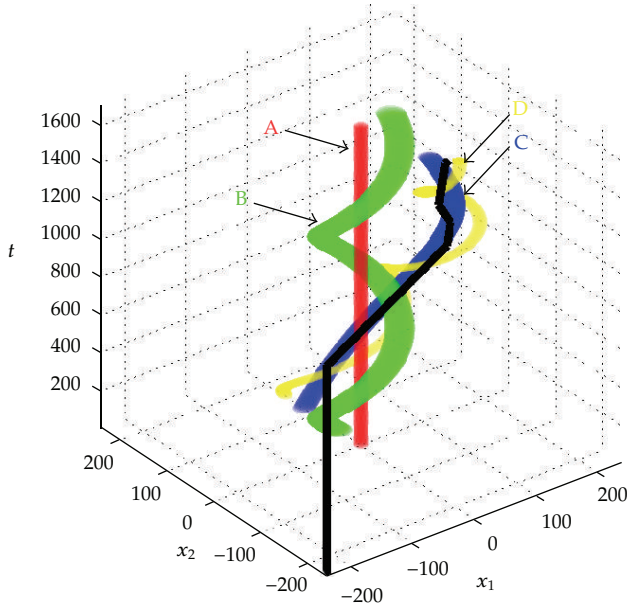


Figure 4: Optimal trajectory for Problem 2 (solid black line); the forbidden zones of the respective centers of attractions: *A*, *B*, *C*, *D* (colored areas).

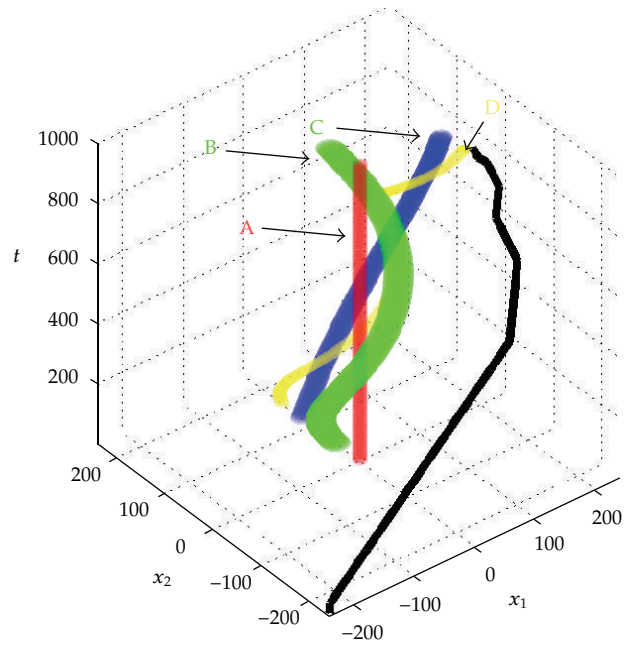


Figure 5: Optimal trajectory for Problem 3 (solid black line); the forbidden zones of the respective centers of attractions: *A*, *B*, *C*, *D* (colored areas).

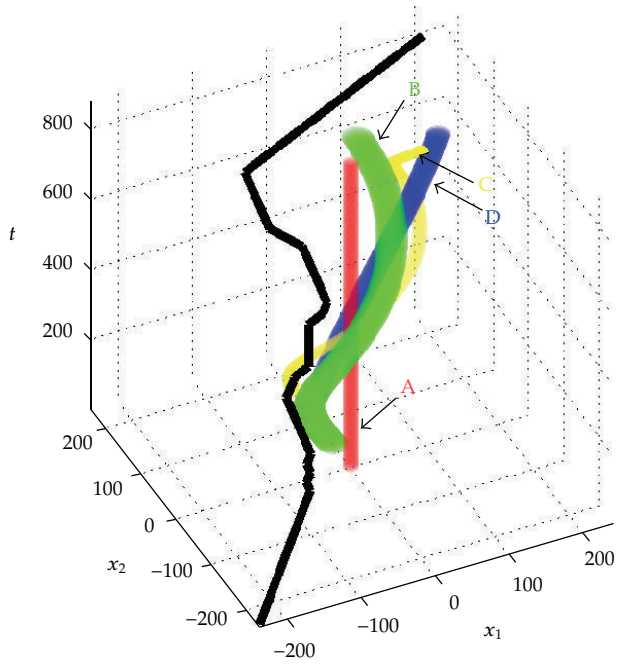


Figure 6: Optimal trajectory for Problem 4 (solid black line); the forbidden zones of the respective centers of attractions: *A*, *B*, *C*, *D* (colored areas).

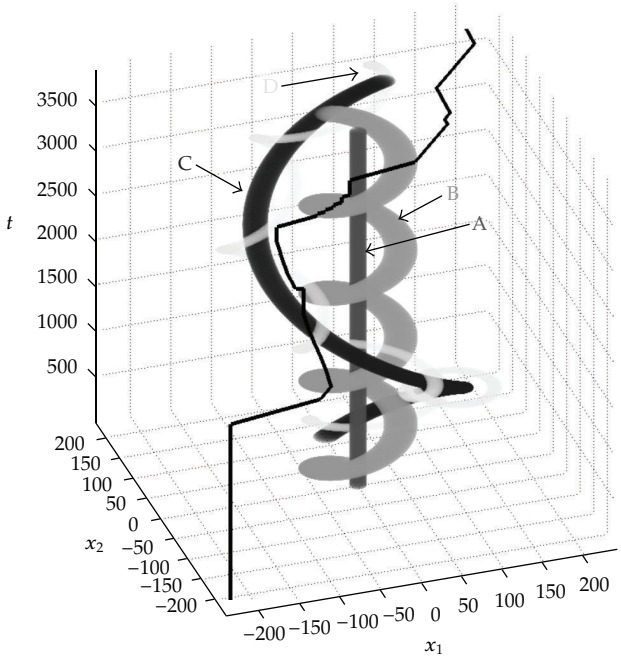


Figure 7: Optimal trajectory for Problem 5 (solid black line); the forbidden zones of the respective centers of attractions: *A*, *B*, *C*, *D* (grayed areas).

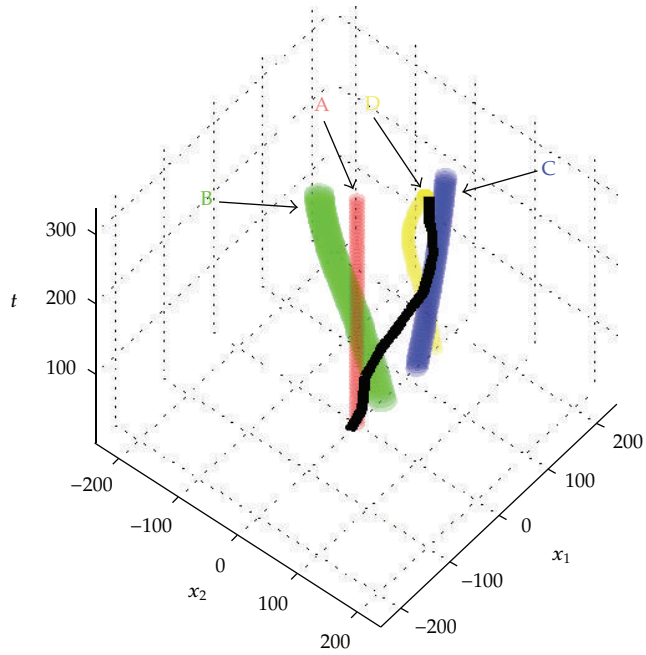


Figure 8: Optimal trajectory for Problem 6 (solid black line); the forbidden zones of the respective centers of attractions: *A*, *B*, *C*, *D* (colored areas).

3.8. Problem 6

The last example presents the design of the optimal trajectory with the initial point at (x_{1o} = 2.5, x_{2o} = −17.5, t_0 = 0) and the terminal point satisfying (3.7) with the time horizon t_N = 4000. The total cost $J_5 = 24.752 \times 10^{-2}$. The gained results are depicted in Figure 8.

4. Conclusions

This work presents an effective concept of designing optimal and feasible control strategies for time-varying dynamical processes based on discrete optimization approach and its solutions.

After segmenting the composed time-and-state workspace of the process dynamics, a time-and-state space toolgraph (made of temporary state-space graphs) is constructed that represents all the pertinent properties of the autonomous dynamics of the considered dynamical process. Additional assumptions or restrictions concerning arbitrary forbidden zones for the operational points can be easily taken into account. Numerical computations are performed by means of any discrete optimization algorithm which searches for the cheapest path connecting the node initial and terminal nodes. The presented work has been implemented based on the basic form of the Dijkstra's algorithm, whose performance can be improved by applying various modifications given in the literature, for instance, in [10].

The resulting cheapest path describes an optimal piecewise-linear trajectory of the operational points in the considered time-and-state space. Note that such a solution may need some kind of smoothing for most real-time process cases.

A clear suboptimal property of the proposed approach results from the fact that the computed reference trajectory can only pass through the centers of the segments. In this context the size of segments has a key influence on the quality of the ultimate optimal solution being sought.

Thus, in general, this interdisciplinary paper concerns research into formal optimization methods and tools. Namely, it shows how to constructively predefine the control optimisation problem or to convert complex optimal control problems for nonlinear time-variant objects with constraints into a graph structure ready to be processed by any of the discrete optimization methods, in particular, by the standard numerical methods of graph optimisation. In the light of theory application to problems arising in engineering, this method can be used for on- or off-line numerically solved practical optimal control or path-finding problems for time-varying objects or environments.

Though this paper is an investigative proposition of a new numerical optimization method, an abstract but nontrivial and imaginative example of several variations has also been given to illustrate the effectiveness of the proposed methodology. The example presented inclusive of a system of variant attraction centers is a rough analog to the solar system. Other business and industry applications are to be developed in order to give a wider proof of the advantages of the proposed method in a confrontation with the scientific challenges of the real world. We also hope that this paper should encourage scientists and engineers to apply both this approach and the discrete optimisation methods.

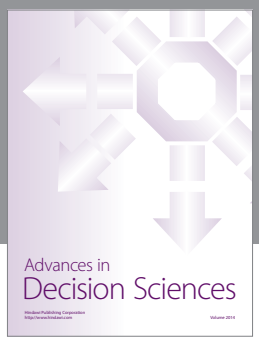
Complexity of the presented method principally results from the complexity of the algorithm applied for searching the toolgraph. Some clues concerning the possibilities of improving the graph search process can be found in [11, 12], for instance. It is clear that the difficulty of the proposed method is shaped mostly by the numerical complexity of searching the T-graph. In a straightforward tactic one makes use of the Dijkstra algorithm ($O(n^2)$). Other, mostly heuristic enhancements (like the renowned algorithm A^*) can also be employed in order to improve the efficiency of the proposed numerical optimisation procedure.

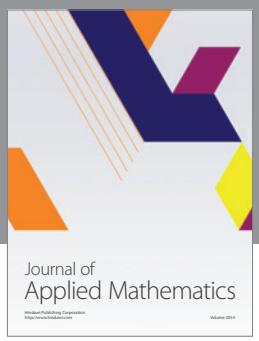
In spite of the presented feasibility of the approach, a general problem of the so-called curse of dimensionality has always to be taken into account in all research of this type. This means that there are limits which render such approaches infeasible in some problems of practical interest. Nevertheless, nowadays, in view of the persistent progress in computer technology, finding constructive methods of solving problems appears to be most substantial in science, technology, and engineering.

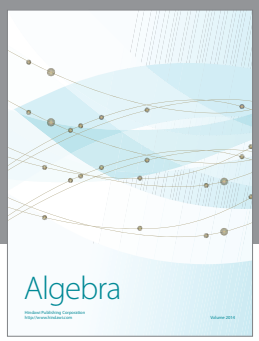
References

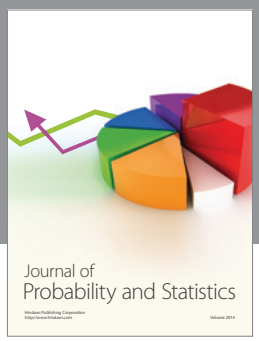
[1] L. Jaulin, "Path planning using intervals and graphs," *Reliable Computing*, vol. 7, no. 1, pp. 1–15, 2001.
[2] E. Masehian and A. N. D. G. Habibi, "Robot path planning in 3D space using binary integer programming," *International Journal of Mechanical, Industrial and Aerospace Engineering*, vol. 1, no. 1, pp. 26–31, 2007.
[3] Z. Kowalczuk, K. Rudzinska-Kormanska, and A. N. D. K.E. Olinski, "Designing nonlinear control systems by state-space flow graph optimization," in *Proceedings of the 11th IFAC Symposium on Large Scale Systems CD-ROM*, IFAC, Gdansk, Poland, 2007.
[4] Z. Kowalczuk and A. N. D. K.E. Olinski, "Optimal and safe control planning with the use of discrete optimization," in *Recent Advances in Control and Automation*, K. Malinowski and L. Rutkowski, Eds., pp. 283–292, Academic Publishing House EXIT, Warsaw, Poland, 2007.
[5] Z. Kowalczuk and K. E. Oliński, "Suboptimal fault tolerant control design with the use of discrete optimization," *International Journal of Applied Mathematics and Computer Science*, vol. 18, no. 4, pp. 561–568, 2008.

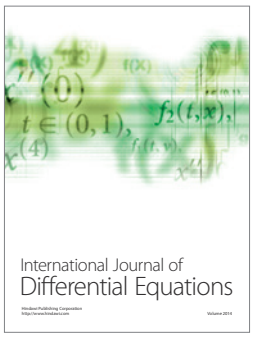
[6] Z. Kowalczuk and A. N. D. K. E. Olinski, "Designing optimal operational-point trajectories using an intelligent sub-strategy agent-based approach," in *Smart Information and Knowledge Management: Advances, Challenges, and Critical Issues*, E. Szczerbicki and N. T. Nguyen, Eds., vol. 260 of *Studies in Computational Intelligence*, pp. 273–282, Springer, Berlin, Germany, 2010.
[7] J. M. Smith, *Mathematical Modeling and Digital Simulation for Engineers and Scientists*, John Wiley & Sons, New York, NY, USA, 1977.
[8] R. Diestel, *Graph Theory*, vol. 1, Springer, New York, NY, USA, 1st edition, 2000.
[9] S. S. Skiena, *The Algorithm Design Manual*, Springer, New York, NY, USA, 1997.
[10] R. V. Helgason, J. L. Kennington, and B. D. Stewart, "The one-to-one shortest-path problem: an empirical analysis with the two-tree Dijkstra algorithm," *Computational Optimization and Applications*, vol. 2, no. 1, pp. 47–75, 1993.
[11] D. P. Bertsekas, *Dynamic Programming and Optimal Control*, Athena Scientific, Nashua, NH, USA, 2005.
[12] M. L. Fredman and R. E. Tarjan, "Fibonacci heaps and their uses in improved network optimization algorithms," *Journal of the Association for Computing Machinery*, vol. 34, no. 3, pp. 596–615, 1987.

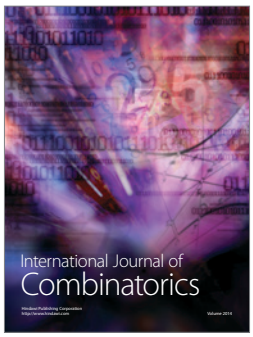


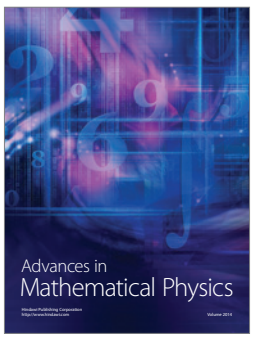


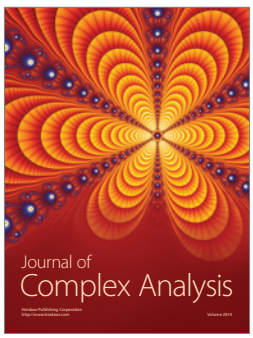


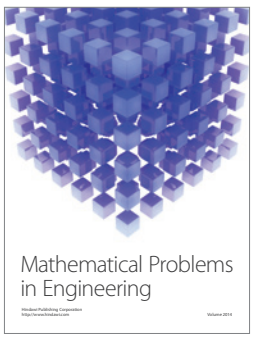


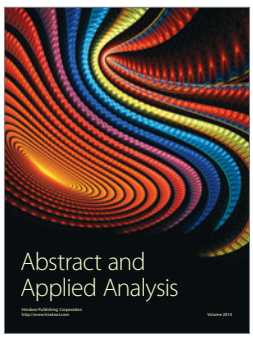


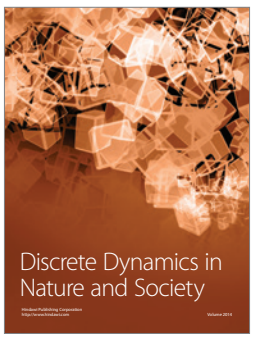


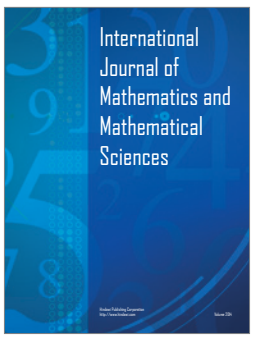


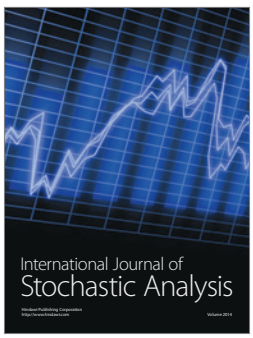












Hindawi

Submit your manuscripts at
http://www.hindawi.com