Adv Syst Sci Appl 2023; **01**:35–49
Published online at https://ijassa.ipu.ru.

Piecewise Constant Functions in Dynamic Optimization Problems

Olga E. Orel[1*], Evgeny N. Orel[2]

[1]*Moscow Institute of Physics and Technology (State University), Dolgoprudnyi, Russia*
[2]*Financial University under the Government of the Russian Federation, Moscow, Russia*

Abstract: We consider a direct method of dynamic optimization to approximate the global extremum. This method is based on splitting the state space into classes (cells) and constructing piecewise constant functions on the partition. Such an approach leads to a generalization of the Euler polygonal method and makes it possible to use shortest path algorithms on graphs. In the suggested algorithm, for each class we construct a path from an initial point to this class. Note that the program remembers only the terminal point of the path, functional value along the path and number of the preceding class. If one found another path with the same boundary conditions but less functional value (for the minimization problem), this path would become the current approximation. We prove that if the partition is sufficiently small, then, by using our method, we obtain the optimal polygon. The suggested approach can also be applied to problems of dynamic optimization with incomplete information (differential games, control of systems with unknown dynamics). In addition, some results of numerical solutions of optimal control problems and differential games are given.

Keywords: global extremum, optimal control, Euler polygonal method, differential games, splitting into classes, piecewise constant function

1. INTRODUCTION

In [11], [12], [9] to solve optimal control problems we used an approach based on construction of piecewise constant functions that are approximations of a Bellman function $B(x)$ or a dual function $A(x)$ (i.e., a Bellman function with time reversal). In the paper, we show that such an approach is universal and can be applied to various problems of dynamic optimization when we face different types of uncertainty. Most general consideration of optimal control problems is given in [18]. To solve a particular problem, we must split the state space into a finite number of subsets, which are called *classes*, or *cells*. The space of functions that take constant values on classes is finite dimensional. Thus the values of the functions could be computed and stored in computer memory.

Splitting the space into classes does not mean that we impose the restriction to get into points of the predetermined lattice. Our approach is a direct method and is not related to necessary and / or sufficient conditions of extremum. Sometimes, the process of search and construction of piecewise constant functions resembles imitation modelling when the scenario of future behavior of the system is played repeatedly. During this process, the behavior of the system is being improved step-by-step and eventually we get a trajectory that approximates the global extremum.

To show the relationship between the global extremum and its approximation, we consider some well-known problems, which were investigated earlier (see, for example, [3]). For these

*Corresponding author: orel.oe@phystech.edu

problems, the detailed study of various characteristic of switching surfaces, universal and equivocal surfaces, barriers, etc. was carried out. However, sometimes the solution was not complete. In what follows, we will compare the results achieved by analytic and numerical methods and establish the approximate boundaries of their applications.

The suggested approach enables to avoid use of Moiseev "elementary operation" due to moving from cell to cell rather than from point to point (for example, vertices of a lattice).

The method of piecewise constant functions originates from the Euler polygonal method, with which we start. The Euler polygonal method plays an important role not only in the applied but also in the pure mathematics [2]: it is used to solve differential equations as well as to derive Euler–Lagrange equation and to prove the existence theorem in the calculus of variations.

In the paper, we consider the application of the Euler polygonal method to numerical solution of a variational problem. As a result we can see that the algorithm constructs not only a unique polygonal line, but also a family of such lines that form a discrete central field of curves. Each curve consists of segments with constant slopes.

In the optimal control problems, by a polygonal line we mean a trajectory with a piecewise constant control.

We also consider differential games and survival problems with unknown dynamics. To construct piecewise defined functions, we can choose various approaches, which depend on a particular problem. The initial points at the self-learning stage could be chosen systematically or at random, the process could start with a minorant [14] or with a majorant of the Bellman function. One can proceed with the breadth-first or depth-first backtracking search [10], [15]. Finally, in the problems with incomplete information on phase limitations, one can use a heuristic function to speed up the process.

The algorithms and programs are demonstrated on the base of a pseudocode that is not completely formalized PDL (program design language), where we use elements of Pascal.

2. EULER FIELD OF OPTIMAL POLYGONAL LINES

Consider the variational problem

$$J(x(.)) = \int_{t_0}^{t_1} L(t, x(t), \dot{x}(t)dt \to \min, \quad t_0 < t_1, \quad x(t_0) = x_0, \quad x(t_1) = x_1$$

(for simplicity, we consider one-dimensional case, however all the results can easily be generalized to the case $x \in \mathbb{R}^n$). To construct polygonal lines, we divide the time the segment $[t_0, t_1]$ into m equal parts and pick n points with a fixed step on the Ox-axis. Thus a regular lattice (τ_i, x_j) appears in the plane. After that we join the nodes of the lattice located at the neighboring verticals by segments and look for the least-cost path on this graph, which has about mn nodes. Computation of labels of nodes goes from left to right, so it takes mn^2 macro operations to solve the problem. At the same time we construct the function $A(\tau_i, x_j)$ that is defined on the nodes of the lattice and is equal to the minimum cost of a polygonal line passing from the initial point to the current node.

Actually, the algorithm provides not only one trajectory, but rather a discrete *central field of Euler optimal polygonal lines*, starting at (t_0, x_0) and terminating at all other nodes of the lattice, i.e., the optimal motion is imitated repeatedly. The last circumstance is very important because before we find one globally optimal trajectory, we must pass through all nodes to make sure that there are no regions, at which the cost of steps is too low or even negative. By using Euler polygonal lines we significantly simplify the problem and restrict the domain of search. However, as we know, any sufficiently smooth curve can be approximated by such polygonal lines. Therefore, if we want to obtain more accurate approximation, we must consider a more fine lattice.

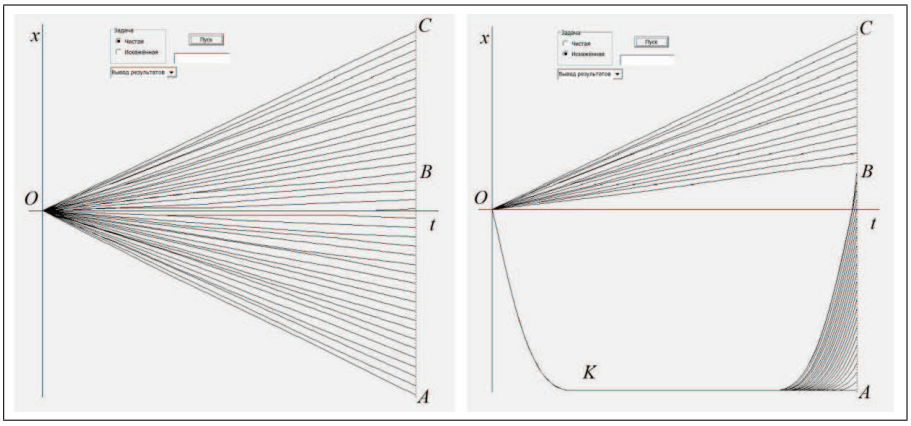

Fig. 2.1. Optimal polygonal lines on the standard and distorted Euclidean planes

The Euler polygonal method was verified by the authors on the familiar variational problems, such as the shortest curves on the Euclidean and Lobachevsky planes, the brachistochrone curve, and the minimal surfaces of revolution [16], [17]. In all cases, visually it would be difficult to distinguish between the fields of curves constructed analytically and numerically.

Figure 2.1 represents the results of the algorithm for two related problems of searching the shortest paths on the standard and "distorted" Euclidean planes. The program has constructed optimal polygonal lines connecting the origin O to the points of the line AC. On the "distorted" plane, the Lagrangian function is given by

$$L = \begin{cases} \sqrt{1+\dot{x}^2}, & x \geq 0, \\ \sqrt{1+\dot{x}^2}\left(1-\left(\frac{x}{a}\right)^2\right), & -a \leq x < 0, \end{cases}$$

where $a > 0$ is some constant. Notice that on the half-plane $x \geq 0$ the Lagrangian is the same for both problems. Thus the segments OM to the points M lying above the Ot-axis are extremals for the "distorted" plane as well. However, if M is located below the point B, the optimal extremal will not be the standard extremal represented by the segment OM, but rather a polygonal line that moves down to a point K, next goes in a horizontal direction, and finally goes up rather than.

Turning back to the general case, note that picking points x_j on the Ox-axis, we divide the axis into $n+1$ sets: single-element sets $\{x_j\}$ and all other points of the axis. With this point of view, $A(\tau_i, x_j)$ can be considered as a trivial piecewise constant function if we let it be equal to $+\infty$ at the points that are not the nodes of the lattice. Nontrivial functions $A(t,x)$ will be considered later.

3. FIXED TIME OPTIMAL CONTROL PROBLEMS

3.1. Generalized Polygonal Lines

Now consider an optimal control problem

$$J([t_0,t_1],x(.),u(.)) = \int_{t_0}^{t_1} L(t,x(t),u(t))dt \to \min,$$

$$\frac{dx(t)}{dt} = f(t,x(t),u(t)), \tag{3.1}$$

$$x(t_0) = x_0, \quad x(t_1) = x_1, \quad t_0 < t_1, \quad x(t) \in X, \quad u(t) \in U.$$

To solve it numerically, as in calculus of variations, we divide the time segment $[t_0, t_1]$ into m equal parts by the points

$$t_0 = \tau_0, \tau_1, \tau_2, \ldots, \tau_i, \ldots, \tau_m = t_1, \quad \tau_i - \tau_{i-1} = \Delta\tau = \frac{t_1 - t_0}{m}.$$

When solving variational problems, we have considered piecewise constant velocities. Now we will consider *controls* that are constant on the intervals $[\tau_{i-1}, \tau_i)$. For convenience, we restrict ourselves to the simple case when the control actions belong to the given finite set

$$V \subset U, \quad V = \{v_j\}, \quad j = 1, 2, \ldots l.$$

In general, the number l and the set of controls V might vary from point to point. To specify the trajectory γ, emanating from (t_0, x_0), it suffices to choose a finite sequence of controls $u_i \in V$ on the intervals $[\tau_{i-1}, \tau_i)$. We write

$$\gamma = (\xi_0; u_1, \xi_1; \ldots; u_{\mu-1}, \xi_{\mu-1}; u_\mu, \xi_\mu), \quad u_i \in V, \quad \xi_i \in X, \quad 1 \le \mu \le m, \tag{3.2}$$

where $\xi_0 = x_0$ and ξ_i is a state, at which the system appears at time τ_i. We will also refer to such trajectories as *polygonal lines* and to the regions with constant controls as *segments*. Let Γ be the set of polygonal lines. The value of the functional J assigned to the polygonal line γ will be called *cost* and denoted by $c(\gamma)$.

All the polygonal lines, which are constructed by the algorithm, are emanated from a common point (t_0, x_0). Here, just as in the calculus of variations, by considering polygonal lines, we give up the idea to obtain the exact solution, but the finer the step $\Delta\tau$ and the wider the set V, the closer we get (by functional) to the global extremum.

A particular segment is a function $x(t)$ defined over the time interval $[\tau_{i-1}, \tau_i]$, $1 \le i \le m$ and satisfying the differential equation

$$\frac{dx(t)}{dt} = f(t, x(t), v_k), \quad v_k \in V, \quad 1 \le k \le l \tag{3.3}$$

with constant control v_k. The interval has an initial point $(\tau_{i-1}, x(\tau_{i-1}))$, a terminal point $(\tau_i, x(\tau_i))$, and a cost

$$J([\tau_{i-1}, \tau_i], x(.), v_j) = \int_{\tau_{i-1}}^{\tau_i} L(t, x(t), v_j) dt.$$

According to the Cauchy uniqueness theorem, we can conclude that all these objects are uniquely defined by the number i of the time segment, the number k of the control, and the initial point $a = x(\tau_{i-1})$. So for the segment (3.3) we set

$$b(i, k, a) = x(\tau_i), \quad c(i, k, a) = \int_{\tau_{i-1}}^{\tau_i} L(t, x(t), v_k) dt. \tag{3.4}$$

The complexity of the problem is the following. For $\mu = m$ there are l^m sequences of the form (3.2). It is important that each polygonal line passes through its own nodes, that is, generally speaking, the polygonal lines have no break points in common, except for the initial one. Note that the number of intermediate nodes grows exponentially depending on m in contrast to the linear dependence in variational calculus, where the number of nodes is mn. On the graph with, for example, $3^{100} \approx 5 \cdot 10^{47}$ vertices, it is almost impossible to find the optimal path. If we try, just as in calculus of variations, to move through the nodes (τ_i, x_j) of the given regular lattice, we will face some difficulties in choosing controls $u \in V$, passing from points (τ_{i-1}, x_j) to neighboring points (τ_i, x_k), and the Moiseev "elementary operation" [7] will sometimes not succeed.

3.2. Splitting into Cells

We suggest the following algorithm to solve the problem. We should split the set X into a finite number n of subsets

$$X = \bigcup_{j=1}^{n} X_j, \quad X_j \bigcap X_k = \emptyset,$$

which we refer to as *classes*, or *cells*. We recommend that the diameters of the sets X_j should be as small as possible. During processing of the algorithm, if there are two points in the same cell at time τ_i, we must remove less promising one, so that not more than mn nodes are stored in the memory, just as in calculus of variations. If $x \in X_j$, we will write $\bar{x} = j$. The points $x, y \in X_j$ of one cell X_j will be regarded as equivalent: $x \sim y$. As was already mentioned, we can achieve that each cell except for one consists of only one point, and all other points are contained in a common cell. Then, just as in variational calculus, there will appear a regular lattice in plane.

Let

$$(\xi_0, \xi_1, \ldots, \xi_\mu) \tag{3.5}$$

be y-coordinates of break points of a polygonal line (3.2). Setting $\bar{\xi}_i = \nu_i$, we get a sequence of integers

$$(\nu_0, \nu_1, \ldots, \nu_\mu), \tag{3.6}$$

representing the numbers of cells, through which the polygonal line passes. The program is supposed to include procedures to find (exact or approximate) values of the functions $b(i, j, a)$ and $c(i, j, a)$ for arbitrary integers $i = 0, 1, \ldots, m$, $j = 1, \ldots, l$ and a real number $a \in X$ (3.4).

3.3. Optimization Algorithm

While processing, the program forms the arrays

$$s[i, j], \quad \varphi[i, j], \quad w[i, j], \quad A[i, j], \quad i = 1, \ldots, m, \quad j = 1, \ldots, n$$

which could be interpreted as follows: $\sigma = s[i, j]$ is a number of the preceding cell X_σ; $x = \varphi[i, j] \in X_j$ – is a "best" point of the cell X_j, into which the system is got at the moment τ_i; $u = w[i, j] \in V$ is a control bringing the system to the point $(\tau_i, \varphi[i, j])$; $A[i, j]$ is a Hamilton function of action (the analogue of a Bellman function with time reversal). The function $A(i, j)$ represents the estimation of the cost of an optimal polygonal line starting at (t_0, x_0) and terminating at X_j at instant τ_i. Generally, the program consists of four subsequent procedures. The comments are enclosed in braces.

Program
"Solution of a fixed time optimal control problem in the class of polygonal lines"
{*Set initial values of action:*}
$A[0, \bar{x}_0] := 0$;
$A[i, j] := \infty$ for all $i, j > 0$;
Fill in the left vertical column $t = \tau_1$;
Fill in the other columns $t = \tau_i, \quad i > 1$, **from left to right**;
Construct the polygonal line by moving backward;
End of program

Now we describe procedures constituting the program.

Procedure "Fill in the left vertical column"
for $k := 1$ to l do begin {*sorting all the controls of* V}
$y := b\,(1, k, x_0)\,;\quad \nu := \bar{y};\quad r := c\,(1, k, x_0)\,;$ {*next node* (τ_1, y) *and action* $r = A(1, \bar{y})$}
if $r < A\,[1, \nu]$
then begin {*value* $A\,[1, \nu]$ *is decreased*}
$s\,[1, \nu] := \bar{x}_0$; {*initial class*} $A\,[1, \nu] := r$; {*action*}
$\varphi[1, \nu] := y;\quad w[1, \nu] := k$; {*control* $\bar{x}_0 \to \nu$}
end;
end;
End of procedure

Procedure "Fill in the other columns"
for $i := 2$ to m do {*processing vertical columns* $i > 1$ *from left to right*}
for $j := 1$ to n do begin {*passing through cells for* $t = \tau_{i-1}$}
$x := \varphi[i-1, j];\quad p := A\,[i-1, j]\,;$
for $k := 1$ to l do begin {*sorting all the controls of* V}
$y := b\,(i, k, x)\,;\quad \nu := \bar{y};\quad r := p + c\,(i, k, x)\,;$ {*next node* (τ_i, y) *and action* $r = A(i, \bar{y})$}
if $r < A\,[i, \nu]$
then begin {*value* $A\,[i, \nu]$ *is decreased*}
$s\,[i, \nu] := j$; {*preceding class*} $A\,[i, \nu] := r$; {*action*}
$\varphi[i, \nu] := y;\quad w[i, \nu] := k$; {*control* $j \to \nu$}
end;
end;
end;
End of procedure

Procedure "Construct γ**, starting at** (t_0, x_0) **and bringing the system to** x_1 **at** $t_1 = t_0 + m \cdot \Delta\tau$"
$j := \bar{x}_1;\quad y := \varphi[i, j];$
if $x_1 \neq y$ then STOP; {*polygonal line does not exist*}
$u := w[m, j];\quad j := s\,[y, u]\,;$
$\gamma := (y, u)$; {*construction starts from the end; now a polygonal line consists of the pair* (y, u)}
for $i := m - 1$ by -1 to 1 do begin {*backward motion*}
$y := \varphi[i, j];\quad u := w[i, j];\ j := s\,[x, u]\,;$
$\gamma := (y, u) \vee \gamma$; {*concatenation: the pair* (y, u) *is attached to the obtained trajectory* γ *on the left*}
end;
$\gamma := x_0 \vee \gamma;$
End of procedure

As a result, we get the sequence of coordinates and controls

$$\gamma = (\xi_0; u_1, \xi_1; \ldots; u_{m-1}, \xi_{m-1}; u_m, \xi_m)\,, \tag{3.7}$$

where $\xi_0 = x_0$, $\xi_m = x_1$.

3.4. Justification of the Algorithm

We will show that for a sufficiently fine partition X into cells, the algorithm constructs optimal polygonal lines.

Fix an arbitrary natural number $\mu \leq m$ and consider a set of polygonal lines (6), consisting of μ segments. There are finitely many such polygonal lines, so the set $Y_\mu = \{x_\mu\}$ of their

right-most nodes (points) is also finite. Let ρ_μ be the minimum distance between pairs of points of the set Y_μ, and let $r = \min[r_\mu | 1 \le \mu \le m]$. It is clear that $r > 0$.

Theorem 3.1:
If the diameters of all the sets X_j are less than r, then the program constructs the optimal polygonal line from (t_0, x_0) to (t_1, x_1), and the cost of this polygonal line is equal to

$$A[m, j], \quad m = \frac{t_1 - t_0}{\Delta\tau}, \quad j = \bar{x}_1.$$

Proof
Consider arbitrary natural numbers $\mu \le m$ and $j \le n$. Since the distances between the points of the set Y_μ are greater than the diameter of the set X_j, then the set $Y_\mu \cap X_j$ consists of at most one element. Denote this element by $z_{\mu j}$ if it exists. Therefore, the polygonal line γ, terminating at or passing through the cell X_j at instant τ_μ, has a node $z_{\mu j}$ at this set. If $Y_\mu \cap X_j = \emptyset$, then there are no polygonal line passing through the cell X_j at instant τ_μ. Then for this cell we will have $A[\mu, j] = \infty$.

Now we prove the statement by induction on the time step m. For $m = 0$ it is trivial because the polygonal line consisting on 0 steps degenerates into the point (t_0, x_0) and, by construction, $A[0, \bar{x}_0] := 0$.

Assume the statement is true for $\mu = m - 1$ steps and prove it for the next step m. Let (3.7) be an optimal polygonal line. Then

$$\gamma_1 = (x_0; u_1, \xi_1; \ldots; u_{m-1}, \xi_{m-1})$$

is also an optimal polygonal line, going to (τ_{m-1}, ξ_{m-1}). By inductive hypothesis, this line or the polygonal line of equal cost would have already been constructed by the algorithm if the point (τ_{m-1}, ξ_{m-1}) were finite. We have $c(\gamma_1) = A[m-1, j]$, where $j = \bar{\xi}_{m-1}$. Let $x = z_{m-1,j}$. Analyse what happens when we fill in the vertical column m as the program processes segments emanating from (τ_{m-1}, ξ_{m-1}). Under control $u_m = v_k \in V$ the system gets to (τ_m, ξ_m). The cost of this step is equal to $c(m, k, x)$.

Therefore, by inductive hypothesis, we have

$$r = c(\gamma_1) + c(i, k, x) = c(\gamma).$$

The last equality is valid by virtue of the additivity of a cost functional. As a result, we will definitely get $A\left[m, \bar{\xi}_m\right] \le c(\gamma)$. From the other hand, the strict inequality $A\left[m, \bar{\xi}_m\right] < c(\gamma)$ is impossible since otherwise the trajectory γ would not be optimal. □

3.5. Economic Example

Consider the following dynamic optimization problem arisen in economics [19]. The conceptual meaning of the problem is as follows. A firm has received an order for H units of product to be delivered by time T. It seeks a production schedule for filling this order at the specified delivery date at minimum cost, bearing in mind that unit production cost rises linearly with the production rate and that the unit cost of holding inventory per unit time is constant. Let $x(t)$ denote the inventory accumulated by time t. Then we have $x(0) = 0$ and must achieve $x(T) = H$. The inventory level, at any moment, is the cumulated past production; the rate of change of inventory is the production rate $x'(t)$. Thus the firm's total cost at any moment t is

$$C_1\big(x'(t)\big)^2 + C_2 x(t),$$

where the first term is the total production cost, the product of the unit cost of production and the level of production; the second term is the total cost of holding inventory; and C_1 and C_2 are positive constants. For simplicity, in what follows we set $C_1 = C_2 = 1$. The firm's

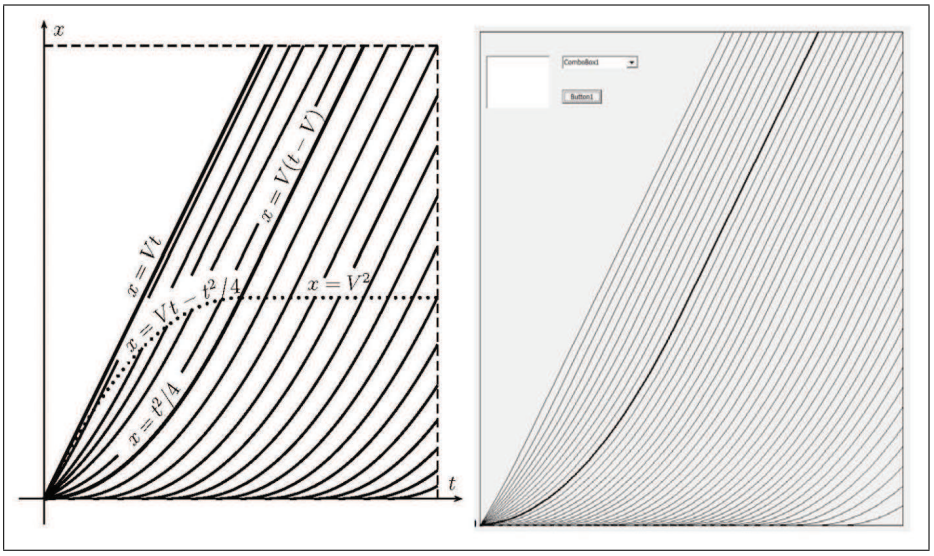

Fig. 3.2. Optimal field of extremals and quasi-optimal field of polygonal lines in the economic problem

objective is to determine a production rate $u(t) = x'(t)$ and inventory accumulation $x(t)$ for $0 < t < T$ such that

$$J = \int_0^T \left[u^2(t) + x(t)\right] dt \to \min,$$

$$\frac{dx(t)}{dt} = u(t), \quad x(0) = 0, \quad x(T) = H \geq 0, \quad 0 \leq u \leq V, \quad x(t) \geq 0.$$

To solve this problem, in [19] we have constructed an optimal central field of Pontryagin extremals. Moreover, the problem was solved numerically with the help of the partition into cells considered above. The results of both investigations is represented in Figure 3.2. The demarcation line to the maximum production rate V is shown on the left part of the figure by the dashed line.

4. AUTONOMOUS PROBLEMS OF OPTIMAL CONTROL

The method of partition into classes for autonomous optimal control problems

$$\int_0^T L(t, x(t), u(t))dt \to \min,$$

$$\frac{dx(t)}{dt} = f(x(t), u(t)),$$

$$x(0) = x_0, \quad x(T) = x_1, \quad x(t) \in X, \quad u(t) \in U,$$

(termination time $T > 0$ of the process is not fixed) has been considered in [9]- [11] in details. Generally speaking, the state $x(t)$ and the control $u(t)$ are vectors.

In frames of a simple version, to solve the problem numerically, we should take a finite set $V \subset U$ and a time step $\Delta\tau$. Thus, the polygonal lines are defined. They consist of segments, i.e., solutions of the differential equation $\dot{x} = f(x, u)$ on the interval $[0, \Delta\tau]$ with constant control $u \in V$. Now we need to divide X into finite number of classes X_i and run the program representing a generalization of any algorithm of finding the shortest path on a graph [9], [11].

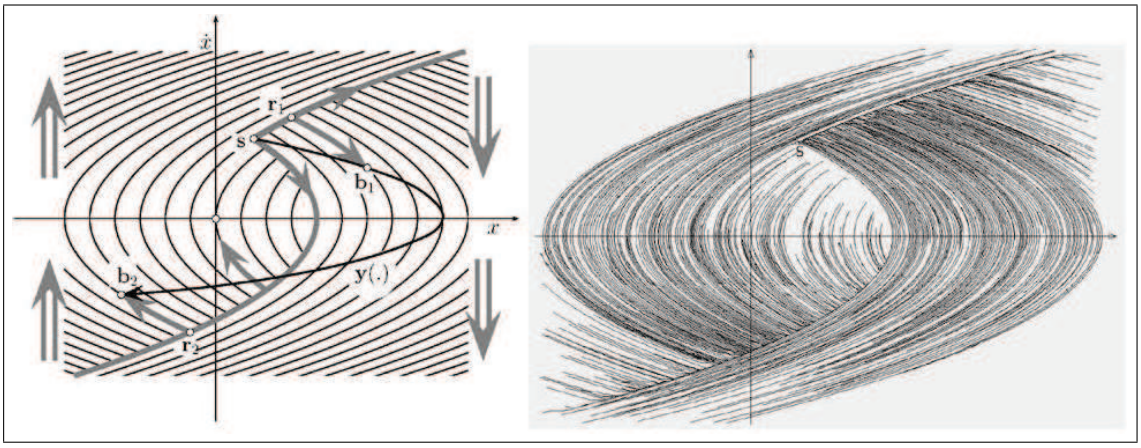

Fig. 4.3. Central field of extremals and polygonal lines in the Bushaw problem

This can be a breadth-first search, such as the Dijkstra algorithm (equal price algorithm) or the Nillson algorithm (A^* search algorithm). However, it can be a backtracking depth-first search [10], [15] or a combination of these algorithms. That machinery could be very important to solve more complicated optimization problems with any lack of information.

Consider few examples on the subject.

4.1. Fel'dbaum-Bushaw Problem

This time-optimal control problem (it is also called an optimal stopping problem) could be stated as follows [20]– [21]:

$$\frac{dx}{dt} = \dot{x}, \quad \frac{d\dot{x}}{dt} = u, \quad |u| \le 1, \quad x(0) = x_0, \quad x(T) = x_1, \quad T \to \min\,.$$

Figure 4.3 shows the central extremal fields and polygonal lines for the fixed initial point [17].

4.2. Car Motion to the Parking Place

The detailed consideration of this problem can be found in [3]. At first glance, it seems that at each particular case the problem could be solved by ruler-and-compass constructions. Nevertheless, the authors are not aware of the exact analytic solution of this problem. Assuming the speed of a material point (x, y) is constant and the curvature is limited, we can describe the problem as follows:

$$\frac{dx}{dt} = \cos\varphi, \ \frac{dy}{dt} = \sin\varphi, \ \frac{d\varphi}{dt} = u, \quad |u| \le 1,$$
$$x(0) = x_0, \ x(T) \in G, \quad T \to \min\,.$$

Here by a parking place we mean a point or its neighborhood. Note that the state space is three-dimensional, so a point (x, y) in plane corresponds to an entire state set (x, y, φ), where φ is an arbitrary angle.

Figure 4.4 illustrates the results of numerical solution of the problem for various initial and terminal conditions. In all the cases, we must eventually reach a small square. On the left-hand part of the figure, we are allowed to get to the square at any slope. On the right, we must get to the square in the direction specified by an arrow, precisely, at an angle that is very close to the arrow. Thus, in the first case the target set is a *two-dimensional* neighborhood of a point (x, y), in the second case the target set is a *three-dimensional* neighborhood of a point

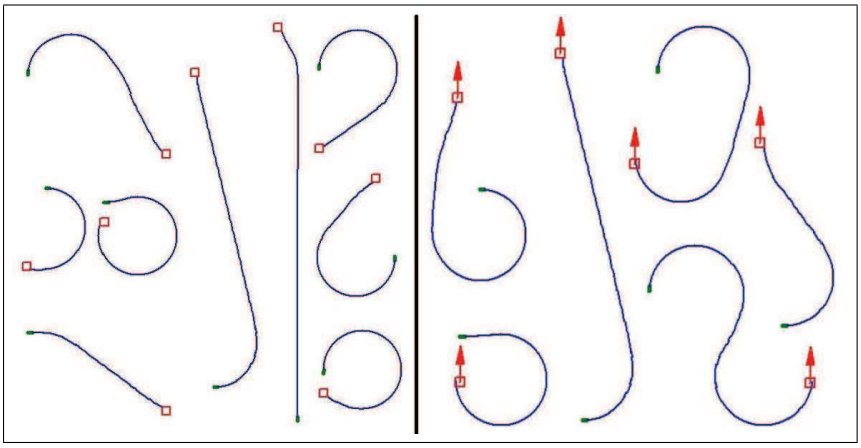

Fig. 4.4. Car motion to the parking place

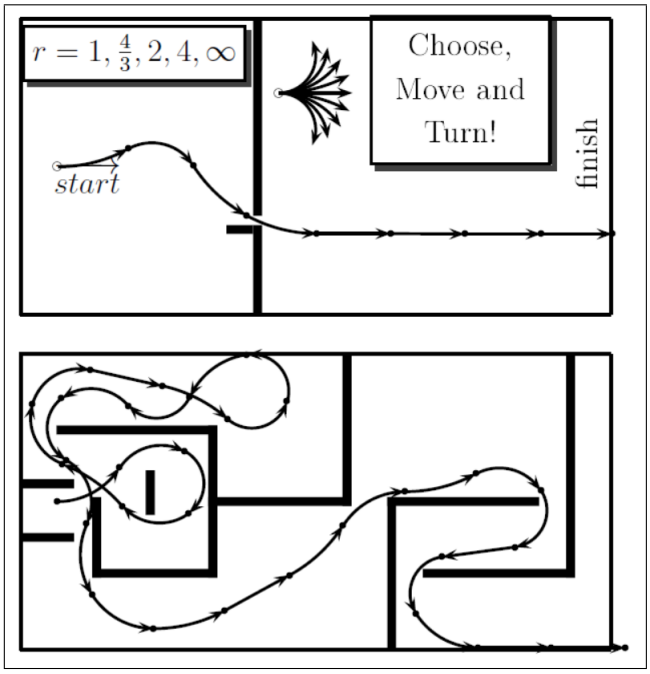

Fig. 4.5. Finding a way out of the labyrinth

(x, y, φ). Obviously, on the left-hand part of the figure the task is a little simpler than on the right-hand side. This can be seen also from the form of the "polygonal lines".

Figure 4.5 shows the solution of the problem on finding a way out of the labyrinth. The car must get out bypassing all the obstacles. Here the region of motion, which is a rectangle of size 8×16, is partitioned into squares with side $\frac{8}{10}$, and the set of directions — into intervals of length $\frac{2\pi}{15}$. So we get $10 \times 20 \times 30 = 3000$ cells. This problem and its solution can be used in robotics and artificial intelligence. Therefore, in [9], [11] there was suggested to use the model for testing methods of solving dynamic optimization problems. However, since then no method to solve the problem on finding a way out of the labyrinth has appeared.

5. DIFFERENTIAL GAMES

Now we consider three differential games with two players (see [3]). In all three games, speeds of player P (pursuer) and of player E (evader) are constant. In addition, pursuer P is to be faster than evader E.

5.1. Simplest Pursuit Game

In frames of this game, the players make a simple motion. Each of them moves with its own constant speed, changing the direction at his own discretion. The game terminates when P captures E (i.e., the distance PE becomes less than a certain prescribed positive quantity). Write kinematic equations:

$$\dot{x}_1 = v_1 \cos\varphi_1, \;\; \dot{y}_1 = v_1 \sin\varphi_1, \;\; \dot{x}_2 = v_2 \cos\varphi_2, \;\; \dot{y}_2 = v_2 \sin\varphi_2, \;\; v_1 > v_2 > 0,$$

where $\varphi_1(t)$ and $\varphi_2(t)$ are controls of the players. For each player, motion along the straight line PE would be optimal, where P moves toward E and E moves away from P. The authors wrote a program, but the game figure is not very informative: one would see only the segments of the straight line PE. The state space is one-dimensional, because the state is completely defined by the distance between the players. In the program, we used the largest distance R (i.e., the distance at which the game terminates with the win for player P), and the segment $[0, R]$ was, as usual, divided into finite number of cells.

5.2. Homicidal Chauffeur Game

A driver attempts to run down a pedestrian at minimum time; the motion is described by the following kinematic equations:

$$\begin{array}{lll} \dot{x}_1 = v_1 \cos\varphi_1, & \dot{y}_1 = v_1 \sin\varphi_1, & \dot{\varphi}_1 = \frac{v_1}{r} u, \\ \dot{x}_2 = v_2 \cos\varphi_2, & \dot{y}_2 = v_2 \sin\varphi_2, & v_1 > v_2 > 0. \end{array}$$

This game was studied by several authors [1], [3], however an exact analytic solution has not been found as yet. Figure 5.6 represents some parties constructed by computer. After self-learning, each player acts optimally himself.

In this game, the state space is two-dimensional, since it is completely determined by the distance between the players and the angle between the sight line PE and the velocity of P.

Figure 5.6 shows 8 parties, each terminates with the victory of pursuer P. The trajectory of P is depicted by an ordinary line, whereas the trajectory of E — by a bold line. In addition, along the path of each player, we put markers at equal intervals of time. The markers of pursuer P are depicted by deleted circles, whereas the markers of evader E are represented by solid squares. Therefore, at each party, the numbers of circles and squares are the same (sometimes, the last and the first markers are not printed well). The initial location of P is always the origin, but the initial direction of P and the initial coordinates of E are specified at random. In parties 2, 3, 5, 6, and 7 player P captured the opponent easily, since he didn't need to perform a sharp turn as the initial velocity of P was directed approximately toward E. Notice that in parties 1 and 4, player P made a turn maneuver immediately. In the beginning of party 8, evader E caught the tail of P to prohibit him from turning. However P, who has a larger speed, broke away from the pursuit and, having made a loop, aimed for E. To extend the time of the game, evader E has changed the direction, but this has only delayed his defeat. The situation just described is represented in 5.7.

5.3. Game of Two Cars

The kinematic equations of players are similar: if speeds satisfy the inequality $v_1 > v_2 > 0$, then the following equations are valid:

$$\begin{array}{lll} \dot{x}_1 = v_1 \cos\varphi_1, & \dot{y}_1 = v_1 \sin\varphi_1, & \dot{\varphi}_1 = \frac{v_1}{r_1} u_1, \\ \dot{x}_2 = v_2 \cos\varphi_2, & \dot{y}_2 = v_2 \sin\varphi_2, & \dot{\varphi}_2 = \frac{v_2}{r_2} u_2. \end{array}$$

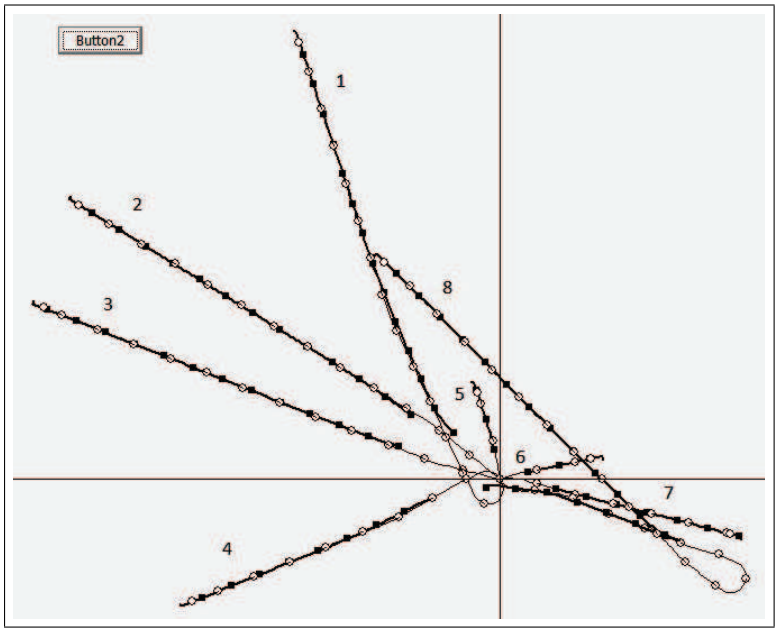

Fig. 5.6. Several parties of the homicidal chauffeur game

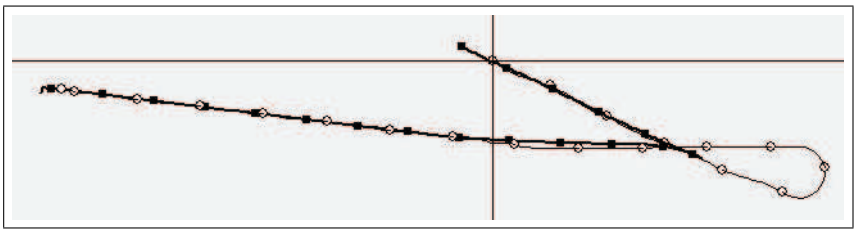

Fig. 5.7. Homicidal chauffeur game. Evader E is catching the tail of P

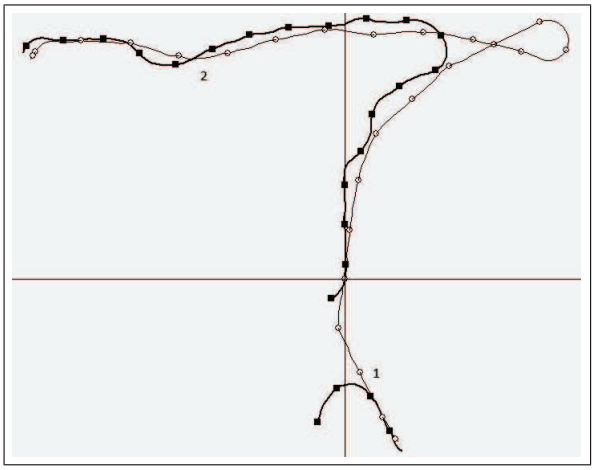

Fig. 5.8. Two parties of the game of two cars

The state space is three-dimensional. The coordinates are completely determined by distances between players and angles between the sight line PE and the velocities of players. Optimal behaviours of players found by the program in two parties are shown in Figure 5.8. Limitations of turning radii are the same for both players. In the first party, the motion is to

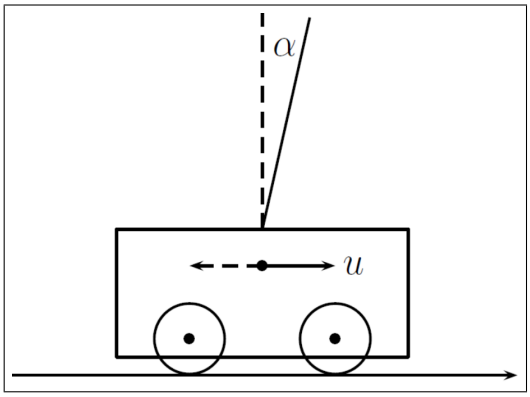

Fig. 6.9. Inverted pendulum on the cart

some extent toward each other. In the second game, player E appeared in the back of P and caught his tail, but then he had to change direction.

In all three parties, the program involves two steps. At the first step (imitation modelling), for one-step scenarios we used Monte–Carlo method, whereas at the second step testing was conducted. Each player didn't know actual strategy of the other and produced its own payoff. It was a payoff when both players act optimally. At the initial instant, player P took the payoff to be identical zero, whereas for player E the payoff was infinite. In what follows, the payoff for P was increasing, and for E it was decreasing. As the self-learning progressed, pursuer P was working out his own strategy by minimax principle, whereas evader E — by maximin one.

6. OPTIMIZATION IN LACK OF INFORMATION

Dynamic optimization problems are not limited to optimal control models when decisions are made in conditions of complete information, or dynamic games when we have no information about the behavior of the opponent. Some other kinds of lack of information are possible. For example, the behavior of the environment could be described by random laws. We will focus on a model, in which the control block has no information about kinematic equations of an object. This is the case for the biological systems. Consider a well-known example of holding a pole in an unstable vertical equilibrium [6], [8].

6.1. Problem on Plane Inverted Pendulum

The system in this example consists of an inverted pendulum mounted to a motorized cart. The pendulum will simply fall over if the cart isn't moved to balance it. The objective of the control system is to balance the inverted pendulum by applying a force to the cart that the pendulum is attached to. One wants to maximize the time during which the pendulum is balanced. Regarding the cart as a massive body, we get the system of Lagrange's equations of second kind:

$$\ddot{x} = u, \quad l_0\ddot{\theta} + u\cos\theta - g\sin\theta = 0, \tag{6.8}$$

where x is a cart position coordinate; u is an acceleration of the cart, which is a control; θ is a pendulum angle from vertical Oy; l_0 is a reduced length to pendulum center of mass; g is the acceleration of gravity. In frames of numerical experiments, we choose the following values of parameters and ranges of phase coordinates: $g = 980$ cm/s^2, $u = \pm 1000$ cm/s^2, $l_0 = 100$ cm, $|\theta| \le 0.2$ rad., $|x| \le 240$ cm, $\Delta\tau = 0.02$ s.

The state space is four-dimensional, the state coordinates are position (x) and velocity ($\dot{x}$) of the cart, pendulum angle from vertical (θ) and its angular velocity ($\dot{\theta}$). Following

Michie [6], we divide the state space into 162 classes according to threshold values: in angle ± 0.1; ± 0.017; 0, in angular velocity ± 0.87; in position of the cart ± 80, in the velocity of the cart ± 50.

At each instant of processing, the control block knows only the ordinal number of class, in which the object is located. On the basis of only this consistent information, the block must make a decision: $u = 1000$ or $u = -1000$. The cost for the lack of knowledge of dynamics is a doubling of the random access memory: action now depends not only on the state classes, but also on the control ± 1000. The algorithms of decision making and self-learning could be different. In the Michie program, self-learning is based on the empiric approach. After some learning period, the pendulum eventually fell down though it has been balanced for a long time. The authors used another program that is based on the idea of a priori optimism and the method of forward error correction [13]. As a result, at the end of self-learning, the pendulum was balanced about vertical position forever.

7. CONCLUSION

Direct methods of dynamic optimization considered in the paper are efficient in the problem on global extremum rather than on local extremum. Moreover, this method can be used in the case when searching analytic solution involves enormous time costs. Partition of the state set into classes must be such that the corresponding data set could be stored in the computer memory. Diameters of classes must be as small as possible.

Creating piecewise constant functions most of all resembles imitation modelling. As is known, piecewise constant functions are used when it is difficult to obtain the solution in the form of mathematical equations. If assumptions of theorem, which was proved in the paper, are satisfied, the algorithm constructs the optimal curve in the class of polygonal lines. If the assumptions are not satisfied, it is still possible to speak about optimization if we recognize it as a process, at which extra resources of time and memory appear.

Note that in the simpler problem on searching global extremum of functions of several variables, it is enough to equate the gradient to zero and after that to test critical points. At that case, direct methods are not required. It is shown in textbooks on mathematical analysis. The picture changes drastically and direct methods have become very actual, when we pass to problems of dynamic optimization, since otherwise we have to construct and analyse central field of extremals, and common approaches have not been worked out yet.

In the paper, we showed not only relevance and efficiency of direct methods in the dynamic optimization problems of various types, but also demonstrated internal and external similarity of analytic and direct methods. This suggests the idea that in particular cases, combination of both methods would be fruitful. The presented algorithms and programs can be applied to various engineering problems connected with optimal control; see, for instance, the recent papers [4, 5].

REFERENCES

1. Breakwell, J. V., Speyer, J. L., & Bryson, A. E. (1963). Optimization and control of nonlinear systems using the second variations, *SIAM J. Control*, **1**, 193–223.
2. Elsgolts, L. E. (1970) *Differential equations and calculus of variations*, Moscow, Russia: Mir Publishers.
3. Isaacs, R. (1965). *Differential games*, New York, NY: John Wiley and Sons.
4. Kramar, V., Alchakov, V., & Osadchenko, A. (2021). The Optimal Control of the Vessel's Automatic Dynamic Positioning System Under Deviation, *Adv. Syst. Sci. Appl.*, **21**(1), 1–10.
5. Mitrishkin, Y. (2021). Plasma Magnetic Control Systems in D-shaped Tokamaks and Imitation Digital Computer Platform in Real Time for Controlling Plasma Current and

Shape, *Adv. Syst. Sci. Appl.*, **22**(1), 1–14.
6. Michie, D., Johnston, R. (1984). *The creative computer. Machine intelligence and human knowledge*, New York, NY: The Viking Press.
7. Moiseev, N. N. (1975) *Elements of the theory of optimas systems*, Moscow, Russia: Nauka.
8. Neimark, U. I., Kogan, N. Ya., & Savelyev, V. P. (1985). *Dynamics models of control theory*, Moscow, Russia: Nauka.
9. Orel, E. N. (1989) A method for solving optimal control problems, *Sov. Math., Dokl.*, **39**(3), 614–617.
10. Orel, E. N. (1991). Adjustment of heuristic functions in search and decision-making processes. *Sov. Phys., Dokl.*, **36**(4), 272–273.
11. Orel, E. N. (1990) Algorithms for searching quasioptimal control using partitioning of the state space, *Zh. Vychisl. Mat. Mat. Fiz.*, **30**(9), 1283–1293.
12. Orel, E. N. (1979) Approximation of Bellman's function by piecewise constant functions, *U.S.S.R. Comput. Math. Math. Phys.*, **18**(4), 95–107.
13. Orel, E. N. (1998). Discrete boundary problems and Optimization processes on graphs, *Artificial Intelligence in Engineering Systems*, 3–33.
14. Orel, E. N. (1977). Finding a shortest path in a graph using the minorant of the Bellman function, *Autom. Remote Control*, **38**, 235–237.
15. Orel, E. N. (1992). Heuristic of teaching in search problems, *Engineering Cybernetics*, **1**(5), 69–81.
16. Orel, E. N., Orel, O. E. (2015). Central field of trajectories in problems of optimal control and calculus of variations, *Accounting. Analysis. Audit*, **3**, 35–42.
17. Orel, E. N., Orel, O. E. (2014). Central fields of optimal trajectories, *Dokl. Math.*, **90**(2), 651–653.
18. *Orel, E. N., Orel, O. E.* (2017) Optimality conditions for central fields of trajectories, *Diff. Eq. Cont. Proc.*, **1**, 53–77.
19. Orel, E. N., Orel, O. E. (2016). Optimal control of the production process when fulfilling an order by a given deadline, *Econom. Math, Met.*, **52**(2), 65–77.
20. Pontryagin, L. S., Boltyanskii, V. G., Gamkrelidze, R. V., & Mischenko, E. F. (1962). *The mathematical theory of optimal processes*, New York, NY: John Wiley and Sons.
21. Zelikin, M. I. (2004). *Optimal control theory and calculus of varianions*, Moscow, Russia: Editorial URSS.

