Optimal clustering of a pair of irregular objects

J. Bennell[1], G. Scheithauer[2], Y. Stoyan[3], T. Romanova[3], A. Pankratov[3]

[1]*Southampton Management School, United Kingdom*

[2]*Technische Universität Dresden, Germany*

[3]*Department of Mathematical Modeling and Optimal Design, Institute for Mechanical Engineering Problems, National Academy of Sciences of Ukraine*

Abstract. Cutting and packing problems arise in many fields of applications and theory. When dealing with irregular objects, an important subproblem is the identification of the optimal clustering of two objects. Within this paper we consider a container (rectangle, circle, convex polygon) of variable sizes and two irregular objects bounded by circular arcs and/or line segments, that can be continuously translated and rotated. In addition minimal allowable distances between objects and between each object and the frontier of a container, may be imposed. The objects should be arranged within a container such that a given objective will reach its minimal value. We consider a polynomial function as the objective, which depends on the variable parameters associated with the objects and the container. The paper presents a universal mathematical model and a solution strategy which are based on the concept of phi-functions and provide new benchmark instances of finding the containing region that has either minimal area, perimeter or homothetic coefficient of a given container, as well as finding the *convex polygonal hull* (or its approximation) of a pair of objects.

Key words: *minimum containment · irregular shapes· cutting and packing · mathematical modeling · optimisation.*

1. Introduction

Cutting and packing problems, also called placement or allocation problems, are interesting theoretically and have many important applications. Applications in industry include dye-cutting in the engineering sector, parts nesting for shipbuilding, marker layout in the garment industry, glass cutting for windows and leather cutting for shoes, furniture and other goods. It is well known that even the one-dimensional version of the problem of finding the optimal usage of a given resource, the classical knapsack problem, belongs to the class of NP-hard optimisation problems. For this reason, most of the work related to cutting and packing problems employ heuristic approaches. Nevertheless, the development of exact solution methods

is an important task to broaden the range of optimal solvable cases.

One of the successful concepts in the case of irregular objects, which lends itselft to exact approaches, is phi-functions [6], [10]. Using this approach leads, in general, to multi-extremal non-linear optimisation problems. Phi-functions permit the modelling of continuous object rotations, non-overlapping, containment and distance constraints. The concept allows the computation of high quality local optima, and in some cases, the global optimum.

In this paper, we address the problem of determining the minimum size container to house two irregular objects and use the concept of phi-functions to achieve this aim. A tool to solve this problem is useful in cutting and packing for the input minismisation (minimum waste) problem and the output maximisation (knapsack) problem. In detail, we will investigate the following two-dimensional problem:

Optimal clustering problem. Given two (irregular) objects bounded by circular arcs and/or straight line segments where free continuous rotations of the objects are permitted, find the minimal sizes of a given containing region (rectangle, circle, or convex polygon) according to a given objective (a polynomial function) and placement parameters of two objects such that the objects are placed completely inside the containing region without overlap and taking in to account allowable distances between objects. We consider a number of frequently occurring objectives, i.e. minimum area, perimeter, and homothetic coefficient (scaling parameter of an equilateral K sided polygon) of a given container.

The containment problem is useful in the design of cutting and packing solution approaches in a number of ways. Some applications have constraints on the cutting process that are naturally dealt with by using a containment approach. Han et al. [18] decribe the cutting of irregular glass shapes for conservatories. Due to the guillotine cutting requirement, pieces are clustered into rectangles first, and then the rectangles are arranged on the stock sheets. Approaches designed for packing of rectangles and boxes is more common in the literature than irregular objects. One reason for this arises from the fact that stronger optimality criterion, or bounds, are available in contrast to cases of arbitrary shaped objects. The containment problem allows these approaches to be exploited for irregular shapes. Finally, placement heuristics that employ hole filling are commonly acknowledged as effective. However, this has not been efficiently implemented with irregular shapes as yet. The containment problem is equivalent to hole filling where the hole defines the size of the container, which is compared with the derived

minimal sizes of the containing region (according to the obtained value of its homothetic coefficient).

The contributions of the paper are many as follows: 1) we deal with an accurate representation of objects bounded by circular arcs and/or line segments (irregular objects) without first generating a polygonal approximation; 2) we allow continuous simultaneous translations and rotations of the objects; 3) we construct a universal mathematical model of the problem taking into account allowable distances (in the form of a non-smooth optimisation problem) using radical free phi-functions and adjusted phi-functions. Also our mathematical model can accommodate any polynomial objective function; 4) we formulate the optimal clustering problem with a wide range of applicable problems including minimal containment of a pair given shapes in a rectangle, circle or convex hull, which supports applications in packing irregular objects, glass cutting non-rectangular pieces, selecting or designing containers, and hole filling; 5) we propose the concept of the phi-tree based on the max-min structure of phi-functions and define the number of terminal nodes of each of the basic phi-trees; 6) we construct a solution tree for the optimal clustering problem and give an estimation of the maximum number of terminal nodes of the solution tree; 7) we propose two efficient algorithms in order to solve the optimal clustering problems based on the solution tree; 8) we present a new set of challenging benchmark instances.

The paper is organized as follows. In the next section, we give an overview of related work. In Section 3, we describe the phi-functions concept for containing regions with variable *metrical characteristics* and placement objects with variable *placement parameters*. In Section 4, we use phi-functions for constructing a mathematical model of the *optimal clustering problem* and present *six basic realisations* of the model. The general solution strategy, which involves the concept of phi-tree and the construction of the solution tree, and two solution algorithms is given in Section 5. We provide some computational results, illustrated with pictures, in Section 6, and finish with some concluding remarks in Section 7.

2. Related work

The literature on two dimensional irregular packing problems is large and it is not possible to comprehensively review here. It is common for cutting and packing papers to indicate

the problem type according to the typology presented in [28]. However in this paper, we deal with a sub-problem of the irregular packing problem. The sub-problem applies to both input minimisation and output maximisation, and is not specific to the number of large objects available. Hence, we have not classified this problem by this typology. In the following we identify some of the relevant papers to this work.

Solution approaches to irregular nesting problems are reviewed by [5, 14]. A tutorial covering the core geometric methodologies currently applied by researchers in cutting and packing of irregular shapes is presented by [4]. Tools of mathematical modeling of arbitrary object packing problems are given by [6, 10, 11]. Complexity investigations for cutting and packing problems can be found in [7, 9, 12, 20] .

Among the plurality of two-dimensional cutting and packing problems the open dimension problem (ODP) is the most common when packing irregular objects. In the ODP, a given set of objects must be placed feasibly within a strip of given fixed width W while minimizing the height H needed. A packing pattern is *feasible* if all objects are contained completely within the rectangle $W \times H$ and do not overlap each other. In this case, the objective to minimize H is equivalent to minimizing the perimeter. This does not remain true if the width W is also variable and the total area has to be minimized, then the objective becomes non-linear.

There is a short list of publications which deal with irregular shapes without polygonal approximations. Packing problems with irregular objects of *fixed orientation*, whose frontier can be described by a sequence of line- and arc-segments, are tackled in [8] using the *line and arc no-fit polygon*. That paper extends the heuristic orbital sliding method of calculating no-fit polygons to enable it to handle arcs and then shows the resultant no-fit polygons being utilised successfully on the two-dimensional irregular packing problem.

Several publications address the *containment problem*. One problem type is as follows: Does a set of given objects (or a single object) fit feasibly within a given containing region? Rotation of objects may or may not be permitted. The single polygon-containment problem where rotation of the polygon is allowed has been studied in [2, 15, 21]. Two-and three-polygon problems are considered in [3, 16].

In [22, 25] the translational containment of multiple polygons within another polygon is investigated, whereas the more general case, also allowing rotations, is considered in [23, 24, 19]. In [26] the authors offer an approximation algorithm of the Minkowski sum for objects

bounded by line segments and circular arcs. Authors of [15] use a mathematical programming formulation/model to solve the following containment problem: Given a convex polygon Q and a simple polygon P, can P be translated and/or rotated such that P fits within Q? This method is used in an algorithm to place small (polygonal) items into (polygonal) holes obtained after placing larger items.

The *minimal-area convex enclosure problem* consists of finding the relative position of two simple polygons such that the area of their convex enclosure is minimized. It is investigated in [17]. The technique searches along the envelope (or no-fit polygon). Authors of [13] consider the problem of circumscribing a convex polygon by a polygon of fewer sides with minimal increase in area.

Most of the approaches dealing with the interaction of two (or a few) objects are used in placement algorithms for larger instances as local decision rules. Some of the earliest approaches to irregular packing problems used the strategy of first clustering pieces within easier to handle shapes, for example [1]. However, such approaches lost popularity as computational speed and methodology improvements facilitated the direct packing of irregular objects. There are recent examples where the cutting process requires the initial clustering, for example, guillotine cuts described in [18]. Further, the more sucessful placement heuristics use hole filling strategies that is the equivalent of the containment problem described here.

3. The concept of phi-functions

Cutting and packing problems involving irregular shapes require the modelling of interaction of two objects with respect to containment, overlap constraints and allowable distances. In this section we describe the core phi-function concepts including the definition of a phi-object, describing a placement objects and a containing regions, identify the interactions between objects. Finally we describe the containment and non-overlapping constraints taking into account allowable distances using phi-functions.

3.1. Placement objects. We assume that any placement object T (an object which has to be placed into a container) considered here, is a two-dimensional one-connected phi-object [6], where a phi-object is a canonically closed point set $T \subset R^2$ $(T = cl^*(T) = cl(int(T))$ having

the same homothopic type as its interior.

Each one-connected phi-object T is given by an ordered collection of frontier elements $l_1, l_2, \ldots, l_n$, (in counter clockwise order). Each element l_i is given by tuple $(x_i, y_i, r_i, x_{c_i}, y_{c_i})$ if l_i is an arc or by tuple (x_i, y_i, r_i) if l_i is a line segment, where (x_i, y_i) and (x_{i+1}, y_{i+1}) are the end points of l_i, (x_{c_i}, y_{c_i}) is the centre point of an arc. Each l_i is a line segment, if $r_i = 0$; l_i is a "convex" arc, if $r_i > 0$; l_i is a "concave" arc, if $r_i < 0$ (assume $(x_{i+1}, y_{i+1}) = (x_1, y_1)$ for $i = n$). We call the components of the tuples, which describe elements $l_1, l_2, \ldots, l_n$, the *metrical characteristics* of placement object T.

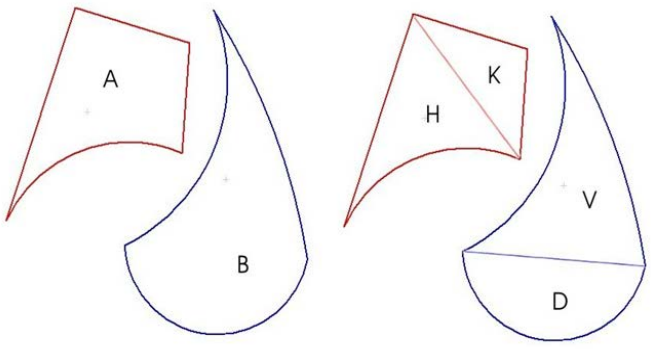
Given the ordered collection of line segments and circular arcs, we apply the decomposition algorithm given in [10] to obtain a set of basic objects that describe the object.

Authors of [10] show that each phi-object, *T*, bounded by line segments and circular arcs, may be decomposed into *n basic objects* of four types, including convex polygons (K), circular segments (D), hats (H) and horns (V).We denote a class of basic objects by $\Re$. In terms each basic object is the intersection of *primitive objects* (half-planes, circles and circular holes). Illustrations of primitive and basic objects are given in Appendix A.

Thus, we represent placement object *T* in the form

$$T = \bigcup_{j=1}^{n} T_j \text{ with } T_j \in \Re. \tag{1}$$

Example. Let us consider two objects *A* and *B* given in Figure 1a. These can be decomposed into basic objects according to formula (1), so we have $A = H \bigcup K$, $B = D \bigcup V$, $V, D, H, K \in \Re$, $n_A = n_B = 2$, see Figure 1b.



(a) (b)

Fig. 1. a) Objects *A* and *B*, b) Decomposition of *A* and *B* into basic objects

Each placement object may be described in *an analytical form***.** It follows from (1) that object T is a union of basic objects T_j, $j = 1, \ldots, n$. Further, each basic object T_j is an intersection of *primitive* objects T_{jk}, $k = 1, \ldots, n_j$ (see Appendix A). Thus, $T = \bigcup_{j=1}^{n} \bigcap_{k=1}^{n_j} T_{kj}$. We define $T_j = \{t \in R^2 : f_j(t) \le 0\}$, with $f_j(t) = \max_{k=1,\ldots,n_j} f_{jk}(t)$, $j = 1, \ldots, n$, where $f_{jk}(t)$ is an infinitely differentiable function associated with primitive object T_{jk}, being *a circle, the complement to a circle* and *a half-plane*. We take the maximum of $f_{jk}(t)$, $k = 1, \ldots, n_j$, because object T_j is defined by an intersection of *primitive* objects. Thus, if $\max_{k=1,\ldots,n_j} f_{jk}(t) \le 0$ then $f_{jk}(t) \le 0$ for all $k = 1, \ldots, n_j$. It follows that since T is a union of basic objects T_j, and we take $\min_{j=1,\ldots,n} f_j(t)$, so each *placement* object can be represented in the form $T = \{t \in R^2 : \min_{j=1,\ldots,n} f_j(t) \le 0\}$, i.e. $T = \{t \in R^2 : \min_{j=1,\ldots,n} \max_{k=1,\ldots,n_j} f_{jk} \le 0\}$. Appendix B provides and example of an object description.

The location and orientation of a *placement* object T is defined by a variable vector of its placement parameters (x_T, y_T, θ_T). The translation of object T by vector $v_T = (x_T, y_T) \in R^2$ and the rotation of T (with respect to its reference point) by angle $\theta_T \in [0, 2\pi)$ is defined by $T(x_T, y_T, \theta_T) = \{t \in R^2 : t = v_T + M(\theta_T)\tilde{t}, \forall \tilde{t} \in T(0,0,0)\}$, where $T(0,0,0)$ denotes the non-translated and non-rotated object T, where $M(\theta_T)$ is given by $M(\theta_T) = \begin{pmatrix} \cos\theta_T & \sin\theta_T \\ -\sin\theta_T & \cos\theta_T \end{pmatrix}$.

We assume here that *placement objects* have *fixed metrical characteristics* and *variable placement parameters* (x_T, y_T, θ_T).

3.2. Containing regions. We consider the following containing regions shown in figure 2: a) a circle of variable radius *r*: $C = \{(x, y) : x^2 + y^2 \le r\}$, b) an axis-parallel rectangle:

$R = \{(x, y) : 0 \le x \le a,\ 0 \le y \le b\}$ where variables a and b are the horizontal and vertical dimensions respectively, c) a convex polygon K : K is given by its variable sides e_i, $i = 1, \ldots, m$, where each side $e_i = [v_i, v_{i+1}]$ of variable length t_i is defined by two variable vertices $v_i = (x_i, y_i)$ and $v_{i+1} = (x_{i+1}, y_{i+1})$, $x_{i+1} = x_i + t_i \cdot \cos\theta_i$, $y_{i+1} = y_i + t_i \cdot \sin\theta_i$, and d) a regular convex polygon with equal sides and homothetic coefficient α. Each side e_i may be given by variable vector $(x_i, y_i, \theta_i, t_i)$; αK is given by its verticies $\alpha v_i = (\alpha x_i, \alpha y_i)$, $i = 1, \ldots, m$, where α is a variable homothetic coefficient and x_i and y_i are constant (see Figure 2d), subject to for original polygon $\alpha = 1$.

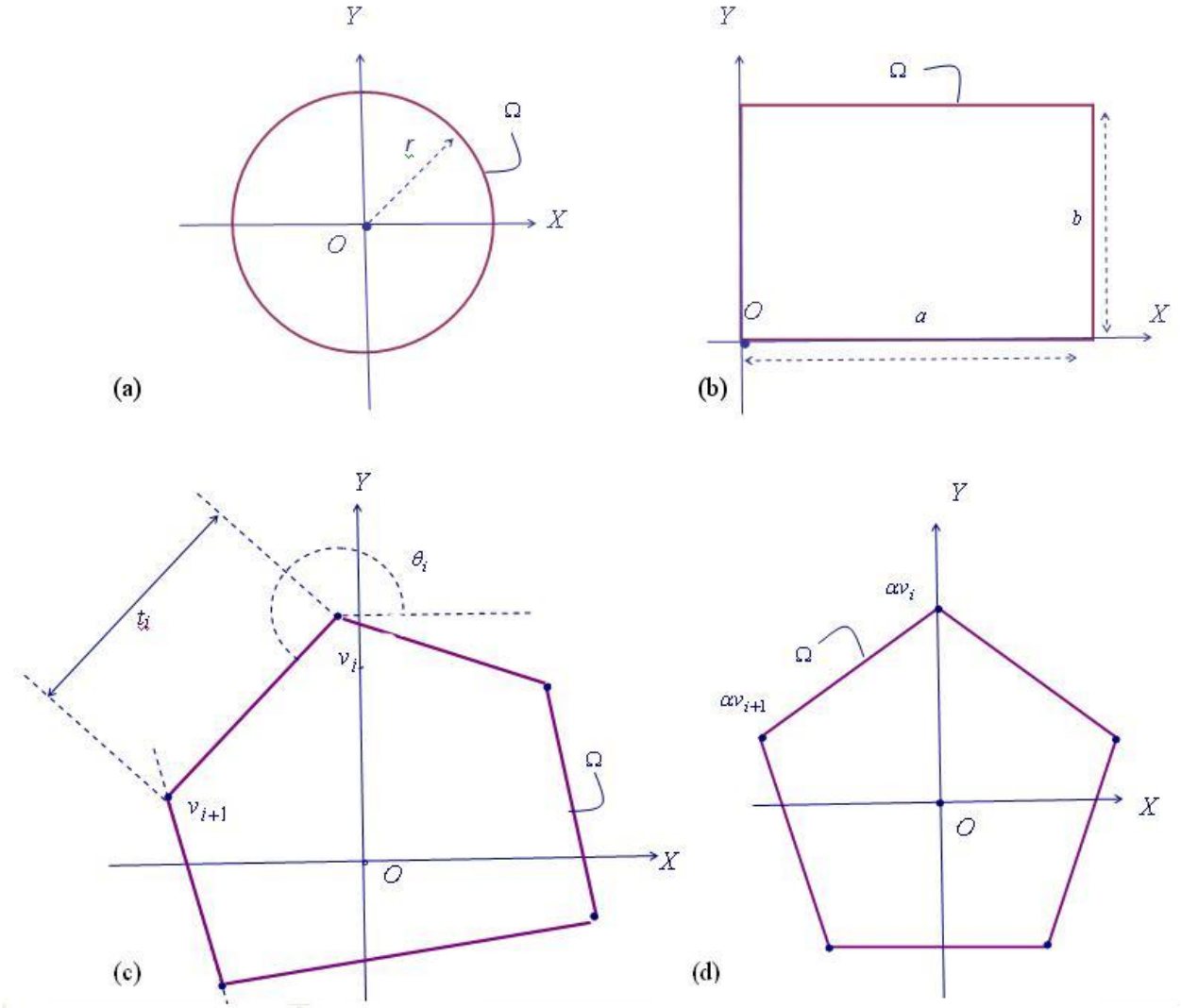


Fig. 2 Containing region Ω of variable vector *p*: (a) circle of variable radius, (b) rectangle of variable sides, (c) convex *m*-polygon of variable vertices, (d) convex polygon of variable

homothetic coefficient

Containing region Ω have fixed *pacement parameters* $(0,0,0)$ and variable *metrical characteristics* p defined above. Hereinafter, we denote a *containing region* $\Omega = \Omega(p)$.

3.3. Description of relationships between objects. In order to feasibly place two objects within a containing region, we need an analytical description of the relationships between a pair of objects A and B. We employ the phi-function technique for this. Phi-functions allow us to distinguish the following three cases: *A* and *B* are intersecting so that *A* and *B* have common interior points; *A* and *B* do not intersect, i. e. *A* and *B* do not have common points; *A* and *B* are in contact, i. e. *A* and *B* have only common frontier points. By definition, the phi-function of *A* and *B* is everywhere defined and continuous function that possesses the following characteristics: $\Phi_{AB} > 0$ if $A \bigcap B = \varnothing$; $\Phi_{AB} = 0$ if $\operatorname{int} A \bigcap \operatorname{int} B = \varnothing$ and $frA \bigcap frB \neq \varnothing$; $\Phi_{AB} < 0$ if $\operatorname{int} A \bigcap \operatorname{int} B \neq \varnothing$, where $\operatorname{int} A$, frA is the interior and the frontier of object *A*. We employ phi-functions for the description of the containment relationship $A \subseteq B$ as follows: $\Phi_{AB^*} \geq 0$, where $B^* = R^2 \setminus \operatorname{int} B$. See [6], [10] for definitions and basic features of phi-functions. According to equation (1), let

$$A = \bigcup_{i=1}^{n_A} A_i, \; B = \bigcup_{j=1}^{n_B} B_j \text{ and } A_i, B_j \in \Re. \tag{2}$$

A phi-function that characterises the non-overlapping of the pair of arbitrary shaped objects A and B and has the form

$$\Phi_{AB} = \min\{\Phi_{ij}, i = 1, 2, \ldots, n_A, \; j = 1, 2, \ldots, n_B\}, \tag{3}$$

where Φ_{ij} denotes a basic phi-function for the pair of objects $A_i, B_j \in \Re$. The complete class of basic phi-functions is given in [11].

Thus, in terms of phi-functions we describe *non-overlapping constraint in the form*: $\Phi_{AB} \geq 0$ and *containment constraint* in the form: $\Phi_{\Omega^* A} \geq 0, (\Phi_{\Omega^* B} \geq 0)$.

For objects *A* and *B* given in Figure 1, based on (2) and (3), we have

non-overlapping constraint: $\Phi_{AB} \geq 0$, where $\Phi_{AB} = \min\{\Phi_{VH}, \Phi_{VK}, \Phi_{DH}, \Phi_{DK}\}$, Φ_{VH}, Φ_{VK}, Φ_{DH}, Φ_{DK} are phi-functions for basic objects.

containment constraint : $\Phi_{\Omega^* A} \geq 0, (\Phi_{\Omega^* B} \geq 0)$, where $\Omega^* = R^2 \setminus \operatorname{int}\Omega$, Ω is a circular, a rectangular or a convex polygonal region and $\Phi_{\Omega^* A} = \min\{\Phi_{\Omega^* V}, \Phi_{\Omega^* D}\}$, $\Phi_{\Omega^* B} = \min\{\Phi_{\Omega^* K}, \Phi_{\Omega^* H}\}$, $\Phi_{\Omega^* V}, \Phi_{\Omega^* D}, \Phi_{\Omega^* H}$, $\Phi_{\Omega^* K}$ are basic phi-functions. It should be noted that $\Phi_{\Omega^* A}$ for $\Omega \equiv K$ with variable verticies we consider in the form: $\Phi_{\Omega^* A} = \min\{\Phi_{P_k A},\ k = 1,...,m\}$, where $\Phi_{P_k A}$ is phi-function for half-plane P_k and object A.

We take into account *distance constraints* replacing phi-functions in *non-overlapping* and *containment constraints* with adjusted phi-functions. By definition an adjusted *phi*-function of A and B is an everywhere defined continuous function $\hat{\Phi}^{AB}$, such that $\hat{\Phi}^{AB} > 0$, if dist(*A*,*B*)$> \rho$, $\hat{\Phi}^{AB} = 0$, if dist(*A*,*B*)$= \rho$, $\hat{\Phi}^{AB} < 0$, if dist(*A*,*B*)$< \rho$, where ρ is the minimum allowable distance between objects *A* and *B*. Here, dist(*A*,*B*) = $\min\limits_{a \in A, b \in B} d(a,b)$ and $d(a,b)$ is the Euclidean distance between two points *a* and *b* in R^2. In particular, we have dist(*A*,*B*)$\geq \rho \Leftrightarrow \hat{\Phi}^{AB} \geq 0$.

4. Mathematical model

In terms of phi-functions we can formulate *the optimal clustering problem* as a constrained optimisation problem:

$$F(u^*) = \min\{F(u) : u \in W\}, \tag{4}$$

$$W = \left\{u \in R^\sigma : \hat{\Phi}^{\Omega^* A} \geq 0, \hat{\Phi}^{\Omega^* B} \geq 0, \hat{\Phi}^{AB} \geq 0, \lambda \geq 0\right\}, \tag{5}$$

where $F(u)$ is a polynomial function, $u = (p, u_A, u_B) \in R^\sigma$ is a vector of variables, R^σ is Euclidean space of σ dimension, p is a vector of variable metrical characteristics of Ω, $(u_A, u_B) = (x_A, y_A, \theta_A, x_B, y_B, \theta_B)$ is vector of variable placement parameters of objects A and B, W denotes the corresponding set of feasible solutions (the solution space), $\hat{\Phi}^{AB}$ is an adjusted phi-function for objects *A* and *B* taking into account a given minimal allowable distance ρ between the objects, $\hat{\Phi}^{\Omega^* A}$ is an adjusted phi-function for objects *A* and Ω^*, $\hat{\Phi}^{\Omega^* B}$ is an

adjusted phi-function for objects B and Ω^* taking into account a given minimal allowable distances ρ' between each object and the frontier of Ω, $\lambda \geq 0$ involves a system of additional restrictions on the metrical variables p that ensures the validity of the shape of a container (for instance, $r > 0$ for a circle, $a, b > 0$ for a rectangle, homothetic coefficient $\alpha > 0$ for convex polygon). It should be noted that if $\rho = 0$ and $\rho' = 0$, then we use the original phi-function instead of the adjusted phi-functions in (5).

We consider six realizations of problem (4)-(5), denoted by P1,.., P6, with respect to the shape of the containing region Ω and the form of the objective function $F(u)$:

P1: $\Omega \equiv R$, $F_1(u) = a \cdot b$ (area of R):

$u = (a, b, u_A, u_B) \in R^\sigma$,

$\sigma = 8$ (for rotatable A and B), $\sigma = 6$ (for non-rotatable A and B),

$\lambda \geq 0$ means: $a > 0, b > 0$;

P2: $\Omega \equiv R$, $F_2(u) = a + b$ (half-perimeter of R):

$u = (a, b, u_A, u_B) \in R^\sigma$,

$\sigma = 8$ (for rotatable A and B), $\sigma = 6$ (for non-rotatable A and B),

$\lambda \geq 0$ means: $a > 0, b > 0$;

P3: $\Omega \equiv C$, $F_3(u) = r$ (radius of C):

$u = (r, u_A, u_B) \in R^\sigma$,

$\sigma = 7$ (for rotatable A and B), $\sigma = 5$ (for non-rotatable A and B),

$\lambda \geq 0$ means: $r > 0$;

P4: $\Omega \equiv K$, $F_4(u) = \sum_{i=1}^{m} t_i$ (perimeter of K):

$u = (x_1, y_1, \theta_1, t_1, \ldots, x_m, y_m, \theta_m, t_m, u_A, u_B) \in R^\sigma$,

$\sigma = 4m + 6$ (for rotatable A and B), $\sigma = 4m + 4$ (for non-rotatable A and B),

$\lambda \geq 0$ means: $x_{i+1} - x_i - t_i \cdot \cos\theta_i = 0$, $y_{i+1} - y_i - t_i \cdot \sin\theta_i = 0$,

$(x_i - x_{i+1}) \cdot \cos\theta_i + (y_i - y_{i+1}) \cdot \sin\theta_i \geq 0$, s.t. $m + 1 \equiv 1$, $t_i \geq 0$, $\sum_{i=1}^{m} t_i > 0$;

P5: $\Omega \equiv K$, $F_5(u) = \sum_{i=1}^{m} (x_i y_{i+1} - y_i x_{i+1})$ s.t. $m+1 \equiv 1$, (doubled area of *K*):

$u = (x_1, y_1, \theta_1, t_1, \ldots, x_m, y_m, \theta_m, t_m, u_A, u_B) \in R^{\sigma}$,

$\sigma = 4m + 6$ (for rotatable *A* and *B*), $\sigma = 4m + 4$ (for non-rotatable *A* and *B*),

$\lambda \geq 0$ means: $x_{i+1} - x_i - t_i \cdot \cos\theta_i = 0$, $y_{i+1} - y_i - t_i \cdot \sin\theta_i = 0$,

$(x_i - x_{i+1}) \cdot \cos\theta_i + (y_i - y_{i+1}) \cdot \sin\theta_i \geq 0$, s.t $m+1 \equiv 1$, $t_i \geq 0$, $\sum_{i=1}^{m} t_i > 0$;

P6: ($\Omega \equiv \alpha K$), $F_6(u) = \alpha$ (homothetic coefficient of *K*):

$u = (\alpha, u_A, u_B) \in R^{\sigma}$,

$\sigma = 7$ (for rotatable objects *A* and *B*), $\sigma = 5$ (for non-rotatable objects *A* and *B*),

$\lambda \geq 0$ means: $\alpha > 0$.

Remark. We may include additional restrictions on variables in $\lambda \geq 0$. For instance, to fix any variable of vector $u = (p, u_A, u_B)$ or give allowable rotation ranges for θ_A, θ_B.

It is clear that, $W \neq \varnothing$, since F is bounded from below, hence problems (4)-(5) are always solvable. The objectives we consider here are linear $\{F_2, F_3, F_4, F_6\}$ or quadratic $\{F_1, F_5\}$. The phi-functions in (5) are composed of $\min$- and $\max$-combinations of linear and\or non-linear functions including $\sin$- and $\cos$-terms [11]. System $\lambda \geq 0$, involves linear and\or non-linear functions including $\sin$- and $\cos$-terms. As a result, the set W of feasible solutions is non-convex, leading to many local extrema. Hence, the *optimal clustering problem* (4)-(5) is a nonsmooth optimisation problem.

5. General solution strategy

The formulaton given in (4) and (5) is a non-smooth optimisation problem and so can not be solved directly. In this section we describe a branching tree structure to define the feasible region by a set of sub-regions and, using these sub-regions, solution strategies to find the local and global extrema.

5.1 The solution tree

Let us consider phi-functions Φ_{AB} , $\Phi_{A\Omega^*}$ and $\Phi_{B\Omega^*}$ which take part in describing the feasible region W (solution space) in mathematical model (4)-(5). Here A and B are composed objects given by $A = \bigcup_{t=1}^{n_A} A_t$, $B = \bigcup_{l=1}^{n_B} B_l$, A_t , $B_l \in \Re$ according to (2).

Let $\Phi_k \in \{\Phi_{A_tB_l}, \Phi_{A_t\Omega^*}, \Phi_{B_l\Omega^*}\}$ be a basic phi-function, $k = 1,.2,..,n$, $n = n_A \cdot n_B + n_A + n_B$, $t = 1,.2,..,n_A$, $l = 1,.2,..,n_B$, where $n_A \cdot n_B$ is the number of basic phi-functions $\Phi_{A_tB_l}$ in phi-function Φ_{AB} ; $n_A + n_B$ is the number of basic phi-functions $\Phi_{A_t\Omega^*}$ and $\Phi_{B_l\Omega^*}$ in phi-functions $\Phi_{A\Omega^*}$ and $\Phi_{B\Omega^*}$.

By construction [10] each basic phi-function Φ_k may be given in the form:

$$\Phi_k = \max_{i=1,..,\eta_k} f_i^k = \max_{i=1,..,\eta_k} \min_{j=1,...,J_i^k} f_{ij}^k ,$$

where f_{ij}^k are infinitely-differentiable functions. Since $\min_{j=1,...,J_i^k} f_{ij}^k \geq 0$ is equivalent to $f_{ij}^k \geq 0$ for all j, and $\max_{i=1,..,\eta_k} f_i^k \geq 0$ means at least one of the inequalities, say $f_{i_0}^k \geq 0$ has to be fulfilled, each of these terms can be considered as a system of (in general non-linear) inequalities. This can be solved using a branching scheme.

For each inequality $\Phi_k \geq 0$ we may construct a tree, called a *basic phi-tree* and noted by $\Im_k$ and η_k means the number of terminal nodes of the basic phi-tree. Each terminal node of $\Im_k$ corresponds to a system of inequalities $f_i^k \geq 0$, $i = 1,2,...,\eta_k$.

The solution tree $\Im$ describes feasible region W of problem (4)-(5) and is constructed as follows, see Figure 3.

The tree root corresponds to inequality system $\lambda \geq 0$.

On *the first level* of $\Im$ we have $\tau_1 = \eta_1$ of nodes, where η_1 is the number of terminal nodes of basic phi-tree $\Im_1$ describing $\Phi_1 \geq 0$, where $\Phi_1 = \max_{i=1,..,\eta_1} f_i^1$, $f_i^1 = \min_{j=1,...,J_i^1} f_{ij}^1$. To each node there corresponds a system of inequalities $\lambda \geq 0$, $f_{i_1}^1 \geq 0$.

On *the second level* of $\mathfrak{I}$ to each node of the first level we add η_2 terminal nodes of basic phi-tree $\mathfrak{I}_2$ describing $\Phi_2 \geq 0$, where $\Phi_2 = \max\limits_{i=1,..,\eta_2} f_i^2$, $f_i^2 = \min\limits_{j=1,...,J_i^2} f_{ij}^2$, i.e. the number of nodes of the second level of $\mathfrak{I}$ becomes $\tau_2 = \eta_1 \cdot \eta_2$.To each node there corresponds a system of inequalities $\lambda \geq 0,\ f_{i_1}^1 \geq 0,\ f_{i_2}^2 \geq 0$.

On *the k-level* of $\mathfrak{I}$ to each node of the (*k*-1)-level level of $\mathfrak{I}$ we add η_k terminal nodes of basic phi-tree $\mathfrak{I}_k$ describing $\Phi_k \geq 0$, $\Phi_k = \max\limits_{i=1,..,\eta_k} f_i^k$, $f_i^k = \min\limits_{j=1,...,J_i^k} f_{ij}^k$, i.e. the number of nodes of the *k*-level of $\mathfrak{I}$ becomes $\tau_k = \eta_1 \cdot \eta_2 \cdot \ldots \cdot \eta_k$ .To each node there corresponds a system of inequalities $\lambda \geq 0,\ f_{i_1}^1 \geq 0,\ f_{i_2}^2 \geq 0, \ldots,\ f_{i_k}^k \geq 0$.

Note that $\tau_n = \eta_1 \cdot \eta_2 \cdot \ldots \cdot \eta_{n-1} \cdot \eta_n = \eta$, where η is the number of terminal nodes of the solution tree $\mathfrak{I}$.

Now we may present a feasible region W of problem (4)-(5) as a union of subregions $W_s, s = 1, 2, \ldots, \eta$ [9]. Each W_s corresponds to *s*-th terminal node of $\mathfrak{I}$ and therefore W_s is determined by a system of inequalities $\lambda \geq 0,\ f_{s_k}^k \geq 0, k = 1, \ldots, n$.

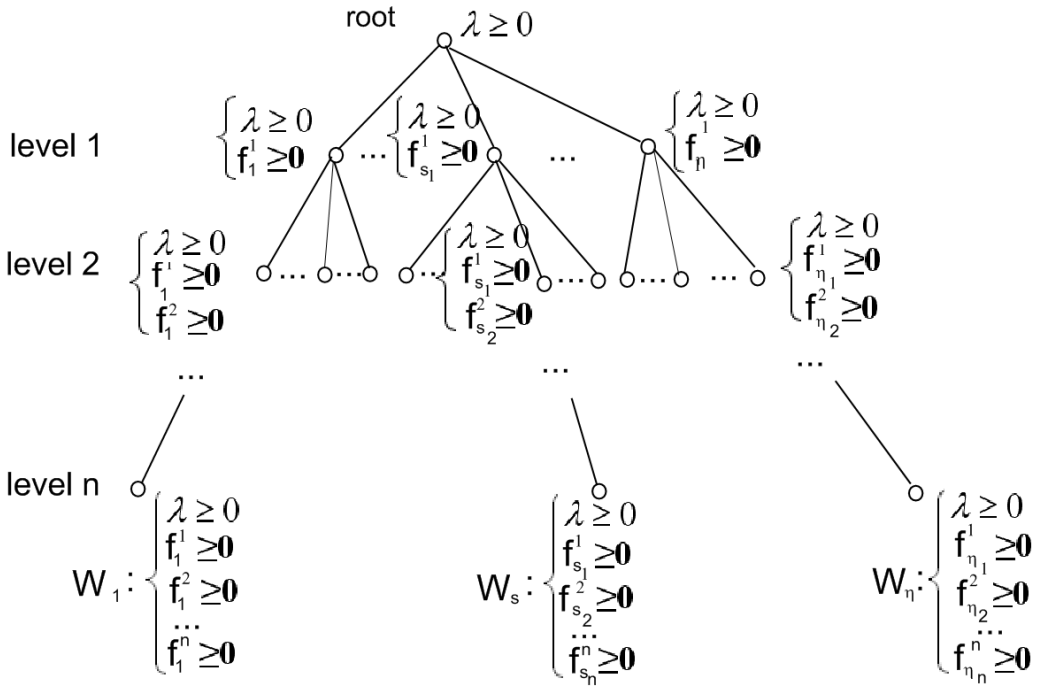


Fig. 3 Forming inequality systems which describe subsets $W_s, s = 1, 2, \ldots, \eta$

Since $W = \bigcup_{s=1}^{\eta} W_s$ we may reduce the nonsmooth optimisation problem (4)-(5) to the following problem for $u = (p, u_A, u_B)$:

$$F(u^*) = \min\{F(u^{s*}), s = 1, \dots, \eta\}, \tag{6}$$

where

$$F(u^{s*}) = \min F(u) \text{ s.t. } u \in W_s, \tag{7}$$

Clearly, the global optimum solution can be obtained and proved by inspecting and exactly solving all of the subproblems defined in (7) .

Subproblems (7) are in general non-linear mathematical programming problems and they may be solved by standard local optimisation techniques (e.g. interior point method, feasible direction method). In particular, problems P2 and P6, which are subproblems of (7), are linear optimisation problems when *A* and *B* are polygons with allowable minimal distances of zero and continuous rotations of *A* and *B* are not allowed. In these cases we can apply LP methods and can derive proved optimal solutions.

5.2. Evaluation of the number of terminal nodes of branching tree $\Im$. The number η of terminal nodes of the solution tree $\Im$ depends on the number η' of terminal nodes of phi-tree $\Im_{AB}$ for $\Phi_{AB} \ge 0$, and the number η'' of the terminal nodes of phi-tree $\Im_{A\Omega^*}$ for $\Phi_{A\Omega^*} \ge 0$, $\Phi_{B\Omega^*} \ge 0$. Since, $A = \bigcup_{i=1}^{n_A} A_i$, $B = \bigcup_{j=1}^{n_B} B_j$, $A_i \in \Re$, $B_j \in \Re$ according to (2) and, therefore, phi-functions Φ_{AB}, $\Phi_{A\Omega^*}$, $\Phi_{B\Omega^*}$ may be given in the form:

$$\Phi_{AB} = \min\{\Phi_{ij}, i = 1, \dots, n_A, j = 1, \dots, n_B\} = \min\{\Phi_k, k = 1, \dots, n_A \cdot n_B\},$$
$$\Phi_{A\Omega^*} = \min\{\Phi_{A_i\Omega^*}, i = 1, 2, \dots, n_A\}, \Phi_{B\Omega^*} = \min\{\Phi_{B_j\Omega^*}, j = 1, 2, \dots, n_B\}.$$

We denote: $n' = n_A \cdot n_B$ and $n'' = \max\{n_A, n_B\} = n_A$, and the upper estimation of the number of terminal nodes of $\Im_{AB}$ and $\Im_{A\Omega^*}$ by η'^* and η''^* respectively, which we derive below.

The upper estimation η^* of the number of terminal nodes of the solution tree $\Im$ for problem (6)-(7) arises from all combinations of the terminal nodes of $\Im_{AB}$, $\Im_{A\Omega^*}$ and $\Im_{B\Omega^*}$, and is defined as

$$\eta^* = (\eta'^*) \cdot (\eta''^*)^2 . \quad (8)$$

The number of terminal nodes for $\Im_{AB}$ and $\Im_{A\Omega^*}$ respectively are

$$\eta' = \eta'_1 \cdot \eta'_2 \cdot \ldots \cdot \eta'_{n'}, \ \eta'' = \eta''_1 \cdot \eta''_2 \cdot \ldots \cdot \eta''_{n''}, \quad (9)$$

Based on (9), we have

$$\eta'^* = (\max\{\eta'_k, k = 1, 2, \ldots, n'\})^{n'}, \ \eta''^* = (\max\{\eta''_k, k = 1, 2, \ldots, n''\})^{n''} . \quad (10)$$

For constructing the solution space of problem (6)-(7) we use the basic phi-functions, which is a complete class of phi-functions for all pairs of basic objects of $\Re = \{K, D, H, V\}$: phi-functions for non-overlapping constraints $\Phi_{AB} \in \{\Phi_t, t = 1, 2, \ldots, 10\}$, $A \in \Re$, $B \in \Re$, and phi-functions for containment constraints $\Phi_{A\Omega^*} \in \{\Phi_p, p = 1, 2, \ldots, 4\}$, $A \in \Re$.

Then estimations (10) are reduced to

$$\eta'^* = (\max\{\eta'_t, t = 1, 2, \ldots, 10\})^{n'}, \eta''^* = (\max\{\eta''_p, p = 1, 2, \ldots, 4\})^{n''} . \quad (11)$$

Table 1 and 2 define the upper estimations $\eta'_t, t = 1, 2, \ldots, 10$, and $\eta''_p, p = 1, 2, \ldots, 4$, respectively, according to the formulas of the basic phi-functions given in [11]. We note that in tables 1 and 2 notations m_A, m_B mean the numbers of sides of convex polygons $K_A \subset A, K_B \subset B$.

Table 1. Estimations of the number of terminal nodes for phi-tree $\Im_{AB}$

$A \backslash B$	V	H	D	K_B
V	η'_1=385			
H	η'_2=197	η'_3=28		
D	η'_4=45	η'_5=35	η'_6=19	
K_A	$\eta'_7 = 2(2m_A^2 + 6m_A + 7)$	$\eta'_8 = m_A^2 + 5$	$\eta'_9 = 3(m_A + 1)$	$\eta'_{10} = m_A + m_B$

Table 2. Estimations of the number of terminal nodes for phi-tree $\Im_{A\Omega^*}$

$\Omega^* \backslash A$	V	H	D	K
R^*	η''_{R1}=12	η''_{R2}=1	η''_{R3}=12	η''_{R4}=1
C^*	η''_{C1}=2	η''_{C2}=1	η''_{C3}=2	η''_{C4}=1
K^*	$\eta''_{K1} = m^2 - m$	η''_{K2}=1	$\eta''_{K3} = m^2 - m$	η''_{K4}=1

Based on relations (8) - (11) and tables 1 and 2, we obtain

$$\eta_R^* = 12^{2n''} \cdot (\max\{\eta'_1, \eta'_7, \eta'_{10}\})^{n'}, \quad (12)$$
$$\eta_C^* = 2^{2n''} \cdot (\max\{\eta'_1, \eta'_7, \eta'_{10}\})^{n'},$$
$$\eta_K^* = (m^2 - m)^{2n''} \cdot (\max\{\eta'_1, \eta'_7, \eta'_{10}\})^{n'}.$$

The increase in the number of terminal nodes of the solution tree shows that the optimal clustering problem is NP-hard, and in most cases to consider all subregions W_s and, therefore, to obtain proved optimal solution is an unrealistic task .

5.3 Example of the solution tree. In this subsection we provide an example of a solution tree for problem P3 considering two circular segments and a containing circle.

Let Ω be a circular container of radius r. We consider two objects: $A = D_1$ and $B = D_2$, where D_i are circular segments given as $D_i = T_i \bigcap C_i$, where C_i are circles, T_i are appropriate triangles with two tangent sides to C_i, $i \in \{1, 2\}$. Center points O_i of the circles are poles of D_i, $i \in \{1, 2\}$. Appendix C explains the rationale for this composition to derive a circular segment.

The mathematical model (4)-(5) can be written in the form

$$\min r \quad s.t. \quad u \in W, \quad (13)$$
$$W = \{u \in R^7 : \Phi_{D_1 D_2} \geq 0, \Phi_{D_i C^*} \geq 0, r > 0, \ r \leq r_i, \ i = 1, 2\},$$

where $u = (r, x_1, y_1, \theta_1, x_2, y_2, \theta_2) \in R^7$.

In order to reduce problem (13) to problem (6)-(7) we need to construct an inequality systems describing subregions W_s, $s = 1, 2, \ldots, \eta$.

Let us now consider inequality $\Phi_{D_1 D_2} \geq 0$. Since $D_1 = T_1 \bigcap C_1$ and $D_2 = T_2 \bigcap C_2$ we have

$$\Phi_{D_1 D_2} = \max\{\Phi_{D_1 T_2}, \Phi_{D_1 C_2}\}, \quad (14)$$

where $\Phi_{D_1 T_2} = \max\{\Phi_{T_1 T_2}, \Phi_{C_1 T_2}\}$, $\Phi_{D_1 C_2} = \max\{\Phi_{C_2 T_1}, \Phi_{C_1 C_2}\}$.

Consequently, we consider phi-functions of basic objects $\Phi_{C_1 C_2}$, $\Phi_{T_1 T_2}$, $\Phi_{C_1 T_2}$, $\Phi_{C_2 T_1}$ and given in [10, 11].

Before we give explicit forms of the phi-functions we remind the reader that each point $(\tilde{x}, \tilde{y}) \in A(0,0,0)$ in the eigen coordinate system of A is transformed into point (x, y) given a translation of (x_A, y_A) and rotated by angle θ_A as follows:

$$x = \tilde{x} \cdot \cos\theta_A + \tilde{y} \cdot \sin\theta_A + x_A,\ y = -\tilde{x} \cdot \sin\theta_A + \tilde{y} \cdot \cos\theta_A + y_A.$$

Each straight line $\tilde{L} = \{(x, y) \in R^2 \,|\, \tilde{\alpha}x + \tilde{\beta}y + \tilde{\gamma} = 0, \tilde{\alpha}^2 + \tilde{\beta}^2 = 1\}$ is transformed into straight line $L = \{(x, y) \in R^2 \,|\, \alpha x + \beta y + \gamma = 0\}$, where $\alpha = \tilde{\alpha} \cdot \cos\theta_A + \tilde{\beta} \cdot \sin\theta_A$, $\beta = -\tilde{\alpha} \cdot \sin\theta_A + \tilde{\beta} \cdot \cos\theta_A$, $\gamma = \tilde{\gamma} - \alpha \cdot x_A - \beta \cdot y_A$, θ_A is rotation parameter, (x_A, y_A) is translation vector A.

Let (x_i', y_i'), $i = 1, 2, 3$, be the vertices of T_1, and (x_j'', y_j''), $j = 1, 2, 3$, those of T_2, and $T_1 = \{(x, y) : \varphi_i \le 0,\ i = 1, 2, 3\}$, $T_2 = \{(x, y) : \psi_j \le 0,\ i = 1, 2, 3\}$, $\varphi_i = \alpha_i' x + \beta_i' y + \gamma_i'$, $\psi_j = \alpha_j'' x + \beta_j'' y + \gamma_j''$.

The phi-function for T_1 and T_2 is defined as follows:

$$\Phi_{T_1T_2} = \max\{\max_{1\le i\le 3} \min_{1\le j\le 3} \varphi_{ij}, \max_{1\le j\le 3} \min_{1\le i\le 3} \psi_{ji}\}, \tag{15}$$

where $\varphi_{ij} = \alpha_i' x_j'' + \beta_i' y_j'' + \gamma_i'$, $\psi_{ji} = \alpha_j'' x_i' + \beta_j'' y_i' + \gamma_j''$.

The phi-function for two circles C_i of radii r_i and center points (x_{C_i}, y_{C_i}), $i = 1, 2,$ is

$$\Phi_{C_1C_2} = \phi = (x_{C_1} - x_{C_2})^2 + (y_{C_1} - y_{C_2})^2 - (r_1 + r_2)^2. \tag{16}$$

The phi-functions $\Phi_{C_1T_2}$ and $\Phi_{C_2T_1}$ are defined by

$$\Phi_{CT} = \max\{\chi_i, \min\{\omega_i, \psi_i\}, i = 1, 2, 3\}, \tag{17}$$

where $T = \{(x, y) : \alpha_i x + \beta_i y + \gamma_i \le 0, \alpha_i^2 + \beta_i^2 = 1,\ i = 1, 2, 3\}$ is a triangle, C is a circle of radii r_C and center points (x_C, y_C),

$$\chi_i = \alpha_i x_C + \beta_i y_C + \gamma_i - r_C,$$

$$\omega_i = (x_C - x_i)^2 + (y_C - y_i)^2 - r_C^2,$$

$$\psi_i = (\beta_{i-1} - \beta_i)(x_C - x_i) - (\alpha_{i-1} - \alpha_i)(y_C - y_i) + r_C(\alpha_{i-1}\beta_i - \alpha_i\beta_{i-1}).$$

We below apply notations χ_i', ω_i', ψ_i' for $\Phi_{C_1T_2}$ and χ_i'', ω_i'', ψ_i'' for $\Phi_{C_2T_1}$.

In order to ensure $D_i \subset \Omega$ we set $r \le r_i$, $i = 1, 2$, here, since this simplifies the containment constraints, and define phi-functions $\Phi_{C^* D_i}$ for $i = 1, 2$ in the form:

$$\Phi_{C^* D_i} = g_i = \min_{j=1,2} g_{ij}, \quad g_{ij} = r^2 - x_{ij}^2 - y_{ij}^2, \; i = 1, 2, \tag{18}$$

where (x_{ij}, y_{ij}), $j = 1, 2$, are end points of the base of D_i.

Given the above phi-functions, we can now construct the branching tree $\Im$ for problem (13) based on phi-trees (denoted by $\Im_{AB}$, $\Im_{A\Omega^*}$ and $\Im_{B\Omega^*}$) for $\Phi_{D_1 D_2} \ge 0$, $\Phi_{D_1 C^*} \ge 0$ and $\Phi_{D_2 C^*} \ge 0$.

First consider phi-tree $\Im_{AB}$ for $\Phi_{D_1 D_2} \ge 0$. Let υ_{11} denote the root node, and $\upsilon_{ij}^{i-1,k}$ denote the j-th node of the i-th level of the tree ($\upsilon_{ij}^{i-1,k}$ is an offspring of the k-th node of the $(i-1)$-th level of the tree), then the phi-tree for $\Phi_{D_1 D_2} \ge 0$, taking into account relations (14)-(18), has the following form:

level 1: $\upsilon_{11} \leftrightarrow \Phi_{D_1 D_2} \ge 0$

level 2: $\upsilon_{21}^{11} \leftrightarrow \Phi_{D_1 T_2} \ge 0$, $\upsilon_{22}^{11} \leftrightarrow \Phi_{D_1 C_2} \ge 0$

level 3: $\upsilon_{31}^{21} \leftrightarrow \Phi_{T_1 T_2} \ge 0$, $\upsilon_{32}^{21} \leftrightarrow \Phi_{C_1 T_2} \ge 0$, $\upsilon_{33}^{22} \leftrightarrow \Phi_{T_1 C_2} \ge 0$, $\upsilon_{34}^{22} \leftrightarrow \Phi_{C_1 C_2} \ge 0$

level 4: $\upsilon_{41}^{31} \leftrightarrow \{\varphi_{1j} \ge 0, j = 1, 2, 3, \quad \upsilon_{42}^{31} \leftrightarrow \{\varphi_{2j} \ge 0, j = 1, 2, 3, \; \upsilon_{43}^{31} \leftrightarrow \{\varphi_{3j} \ge 0, j = 1, 2, 3,$

$\upsilon_{43}^{31} \leftrightarrow \{\varphi_{3j} \ge 0, j = 1, 2, 3, \upsilon_{44}^{31} \leftrightarrow \{\psi_{1i} \ge 0, i = 1, 2, 3, \; \upsilon_{45}^{31} \leftrightarrow \{\psi_{2i} \ge 0, i = 1, 2, 3,$

$\upsilon_{46}^{31} \leftrightarrow \{\psi_{3i} \ge 0, i = 1, 2, 3, \; \upsilon_{47}^{32} \leftrightarrow \{\chi_1' \ge 0, \upsilon_{48}^{32} \leftrightarrow \{\chi_2' \ge 0, \upsilon_{49}^{32} \leftrightarrow \{\chi_3' \ge 0,$

$\upsilon_{4,10}^{32} \leftrightarrow \{\omega_1' \ge 0, \psi_1' \ge 0, \; \upsilon_{4,11}^{32} \leftrightarrow \{\omega_2' \ge 0, \psi_2' \ge 0, \upsilon_{4,12}^{32} \leftrightarrow \{\omega_3' \ge 0, \psi_3' \ge 0,$

$\upsilon_{4,13}^{33} \leftrightarrow \{\chi_1'' \ge 0, \upsilon_{4,14}^{33} \leftrightarrow \{\chi_2'' \ge 0, \; \upsilon_{4,15}^{33} \leftrightarrow \{\chi_3'' \ge 0, \upsilon_{4,16}^{33} \leftrightarrow \{\omega_1'' \ge 0, \psi_1'' \ge 0,$

$\upsilon_{4,17}^{33} \leftrightarrow \{\omega_2'' \ge 0, \psi_2'' \ge 0$, $\upsilon_{4,18}^{33} \leftrightarrow \{\omega_3'' \ge 0, \psi_3'' \ge 0, \; \upsilon_{4,19}^{34} \leftrightarrow \{\phi \ge 0$.

Thus, the number of terminal nodes, η_{AB}, of the phi-tree $\Im_{AB}$ for $\Phi_{D_1 D_2} \ge 0$ is equal to 19. For each terminal node υ_k, $k = 1, \ldots, 19$, there corresponds a system of inequalities. The

diagram of the phi-tree $\Im_{AB}$ for $\Phi_{D_1D_2} \geq 0$ is given in Appendix D.

Phi-tree $\Im_{A\Omega^*}$ for $\Phi_{D_1C*} \geq 0$ consists of only one node corresponding to the system of inequalities $g_{11} \geq 0, g_{12} \geq 0$. The same is true for phi-tree $\Im_{B\Omega^*}$ for $\Phi_{D_2C*} \geq 0$ and the system of inequalities $g_{21} \geq 0, g_{22} \geq 0,$ where g_{ij} is given in (18). Therefore $\eta_{\Omega^* A} = 1$ and $\eta_{\Omega^* B} = 1$.

Solution tree $\Im$ for problem (13) describes the system of inequalities $\Phi_{D_1D_2} \geq 0, \Phi_{D_iC*} \geq 0, r > 0, -r + r_i \geq 0, i = 1, 2$, and involves phi-trees $\Im_{AB}$, $\Im_{A\Omega^*}$ and $\Im_{B\Omega^*}$. Hence, the terminal nodes of $\Im$ is the to system of inequalities for each terminal node of $\Im_{AB}$ combined with: 1) the system of inequalities $r > 0, -r + r_i \geq 0, i = 1, 2$; 2) one of inequality system of the last level of $\Im_{A\Omega^*}$; 3) one of inequality system of the last level of $\Im_{B\Omega^*}$.

Thus, the number of terminal nodes $\eta = \eta_{AB} \cdot \eta_{\Omega^* A} \cdot \eta_{\Omega^* B} = 19 \cdot 1 \cdot 1 = 19$.

For example, we form the inequlity system corresponding to the 19-th terminal node of $\Im$ by adding to inequality system $\phi \geq 0$ (the system corresponds to node $\upsilon^{34}_{4,19}$ of $\Im_{AB}$) the following inequality systems: 1) $r > 0, -r + r_i \geq 0, i = 1, 2$; 2) $g_{11} \geq 0, g_{12} \geq 0$; 3) $g_{21} \geq 0, g_{22} \geq 0$.

Finally, we obtain the system of inequalities $\phi \geq 0,\ r > 0,\ -r + r_1 \geq 0, -r + r_2 \geq 0,$ $g_{11} \geq 0,\ g_{12} \geq 0,\ g_{21} \geq 0,\ g_{22} \geq 0$, which describes sub-region W_{19} in (7).

5.4 Solution algorithms.

In order to solve the optimal custering problems defined in Section 4, we propose two algorithms. The first requires a comprehensive search for local extrema on all subregions and provides the global extremum provided each subproblem (7) can be solved optimally. The second is considerably faster and only searches one highly promising starting point, hence is only locally optimal. In order to search for local minima of subproblems (7) we use IPOPT (see [26]) that only guarantees a local optimum for non-linear programming problems.

Algorithm 1. The algorithm is based on the branching scheme described above and generates the inequality systems that describe $W_s \subset W$ from (7), using the solution tree $\Im$.

Algorithm 1 employs an accelerated search (branching scheme) of the inequality systems corresponding to nodes $v_{s_k}^k$, $s_k = 1,\dots,\eta_k$, $k = 1,\dots,n$ of $\Im$. In order to discard unpromising nodes at the k-level of the search tree $\Im$ we apply cut rules. The rules use the incompatibility of systems and the upper bound value of the objective function.

First we find an upper estimation F^0 of our objective function $F(u)$. Then we perform an exhaustive search of nodes $v_{s_1}^1, s_1 = 1,\dots,\eta_1$, of the first level of $\Im$ (see Figure 3) sequentially and solve optimisation problems corresponding to node $v_{s_1}^1$ in the form:

$$F(u_{s_1}^{1*}) = \min\{F(u) : u \in V_{s_1}^1\}, \qquad V_{s_1}^1 = \{u \in R^\sigma : f_{s_1}^1 \ge 0, \lambda \ge 0\}.$$

If $V_{s_1}^1 \neq \varnothing$ and $F(u_{s_1}^{1*}) < F^0$, then we consider sequentially each offspring $v_{s_2}^2$, $s_2 = 1,...,\eta_2$, of node $v_{s_1}^1$ and solve optimisation problems corresponding to node $v_{s_2}^2$ in the form: $F(u_{s_2}^{2*}) = \min\{F(u) : u \in V_{s_2}^2\}, \qquad V_{s_2}^2 = \{u \in R^\sigma : f_{s_1}^1 \ge 0 : f_{s_2}^2 \ge 0, \lambda \ge 0\}.$

Otherwise we cut node $v_{s_1}^1$.

If $V_{s_2}^2 \neq \varnothing$ and $F(u_{s_2}^{2*}) < F^0$, then we consider sequentially each offspring $v_{s_3}^3$, $s_3 = 1,...,\eta_3$, of node $v_{s_2}^2$ and solve appropriate optimisation problems corresponding to node $v_{s_3}^3$ by analogy of previous step, otherwise we cut node $v_{s_2}^2$ and so on .

On the last level of $\Im$ we solve an optimisation problem corresponding to node $v_{s_n}^n$ $F(u_{s_n}^{n*}) = \min\{F(u) : u \in W_s = V_{s_n}^n\}, \qquad V_{s_n}^n = \{u \in R^\sigma : f_{s_1}^1 \ge 0,\ f_{s_2}^2 \ge 0,\dots,\ f_{s_n}^n \ge 0, \lambda \ge 0\}.$ If $V_{s_n}^n \neq \varnothing$ and $F(u_{s_n}^{n*}) < F^0$, then we set $F^0 = F(u_{s_n}^{n*})$ and take point $u_{s_n}^{n*}$ as the best solution.

The complexity of Algorithm 1 depends on the number $\widehat{\eta} \le \eta$ of nonlinear optimisation problems which have to be solved and the complexity of applying a nonlinear optimisation method, ($O(\eta)$). Algorithm 1 is applicable only for "simple" objects, since the number of terminal nodes of the search tree $\Im$ increases rapidly with the numbers n_A and n_B of basic objects which form *A* and *B*.

Algorithm 2. This algorithm finds good solutions with reasonable computation times that

do not increase significantly with the complexity of the objects. In order to obtain a good starting solution $u^0 \in W$ the algorithm employs a fast and effective heuristic given in [10]. The heuristic is based on searching for an approximate solution of problem (4)-(5) provided that the placement parameters of objects *A* and *B* take discrete values. Then the algorithm applies IPOPT [27] to search for local minima. Below we give a description of the algorithm.

Let us define function $\Lambda(u) = \min\{\Phi_{AB}, \Phi_{\Omega^* A}, \Phi_{\Omega^* B}, \lambda\}$. Our aim is to extract from $\Lambda(u^0) \geq 0$ an inequality system, which describes subregion $W_s \subset W$, such that $u^0 \in W_s$.

We form the subregion W_s as follows. We realise an exhaustive search of nodes $v_s^1, s = 1, \ldots, \eta_1$, of the first level of $\Im$ (see Figure 3) sequentially and search for the number s_1 such that $f_{s_1}^1(u^0) = f^1(u^0) = \max\{f_1^1(u^0), f_2^1(u^0), \ldots, f_{\eta_1}^1(u^0)\}$. Then we realise an exhaustive search of offsprings v_s^2, $s = 1, \ldots, \eta_2$, of node $v_{s_1}^1$ and search for the number s_2 such that $f_{s_2}^2(u^0) = f^2(u^0) = \max\{f_1^2(u^0), f_2^2(u^0), \ldots, f_{\eta_1}^2(u^0)\}$. And so on.

On the *n*-th level of our solution tree $\Im$ we realise an exhaustive search of nodes $v_s^n, s = 1, \ldots, \eta_n$ which are offsprings of node $v_{s_{n-1}}^{n-1}$ and search for the number s_n such that $f_{s_n}^n(u^0) = f^n(u^0) = \max\{f_1^n(u^0), f_2^n(u^0), \ldots, f_{\eta_n}^n(u^0)\}$. Then we form inequality system which corresponds to s-*th* terminal node of our solution tree $\Im$ in the form: $W_s = \{u \in R^\sigma : f_{s_1}^1 \geq 0, f_{s_2}^2 \geq 0, \ldots, f_{s_n}^n \geq 0, \lambda \geq 0\}$. To each sequence of numbers $s_1, s_2, \ldots, s_k, \ldots, s_n$ there corresponds the number s which is derived by formula:

$$s = s_1^* = \begin{cases} s_k^* = (s_k - 1) \cdot \prod_{i=k+1}^{n} \eta_i + s_{k+1}^*, if\ k = 1, \ldots, n-1 \\ s_k^* = s_k, if\ k = n \end{cases}.$$

Finally, we solve problem $\min_{u \in W_s} F(u)$ starting from point u^0. The complexity of Algorithm 2 depends on the number *n* of basic phi-functions forming the solution space and the complexity of a single application of a nonlinear optimisation method, ($O(n_A \cdot n_B)$).

In conclusion, our approach is able to find the global minimum for problems P1 and P6 for non-

rotable objects using Algorithm 1, since in the case each problem (7) becomes a linear problem and *a good approximation to the global minimum* of problems P1-P6 for the general case using Algorithm 1 or using Algorithm 2.

6. Computational experiments

The results include a number of examples to demonstrate the effectiveness of the methodology. In all cases the input data of the example has been provided in Appendix E. Example 1.1 is the proved global optimum since all the sub-problems defined by (7) are linear and Algorithm 1 performs an exhaustive search. All other examples are non-linear optimisation problems. For local optimisation, our programs use IPOPT (https://projects.coin-or.org/Ipopt). Example 5a and Example 9 are examples of global optimum proved by known global solutions. All input and output data considered in the following examples can be download from http://www.math.tu-dresden.de/~capad/. We use computer AMD Athlon 64 X2 5200+.

Example 1. We consider two triangles A and B and problem **P2.** The task is to find the enclosing rectangle of minimal perimeter, i.e. $F(u) = a + b$.

Example 1.1. Non-rotatable case. In this example we demonstrate the approach for computing the global solution. We use Algorithm 1 to relise model (6)-(7). Figure 4 shows the optimal arrangments of A and B which coorespond to six local minima of problem (4)-(5) arising from the solution tree. Since all subproblems (7) are linear, we can find the global minima $F(u^*)$ of problem (6)-(7). In Figure 4, each solution point u^{s*} is the global minimum of subproblem (5), $s = 1, \ldots, 6$. Solution $u^* = u^{4*}$ is the point of the global minimum of problem (4)-(5). $F(u^*) = \min\{F(u^{1*}), F(u^{2*}), F(u^{3*}), F(u^{4*})\} = F(u^{4*}) = 7.6667$. Running time is 0.06 sec.

Example 1.2. Continuous rotations are allowed, $F(u^*) = a + b =$ 6.3640. Running time is 0.109 sec, see Figure 5a. We use Algorithm 1 to relise model (6)-(7).

Example 1.3. Discrete rotations are allowed. We solve problem (6)-(7) for each object orientation according to the rotation step. We use Algorithm 1. $F(u^*) = a + b =$ 7.0000, see Figure 5b. Running time is 0.72 sec, see Figure 5a.

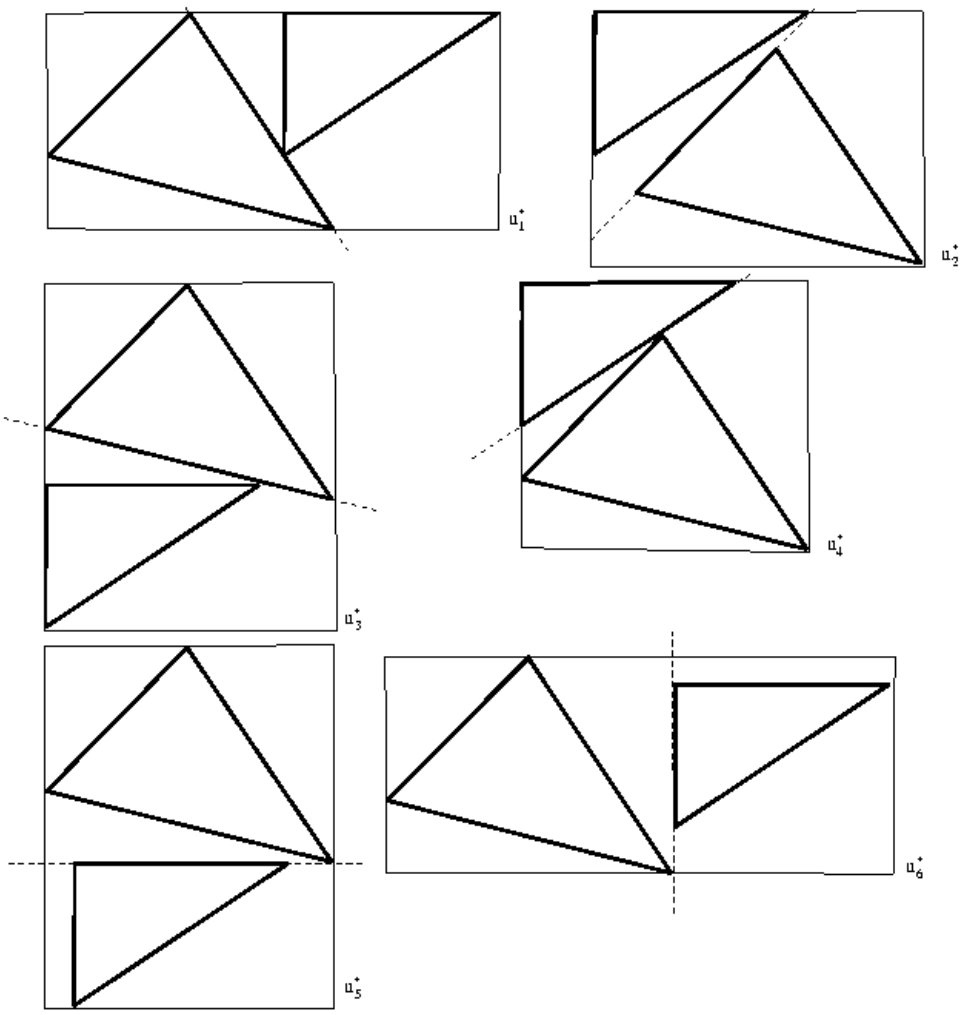
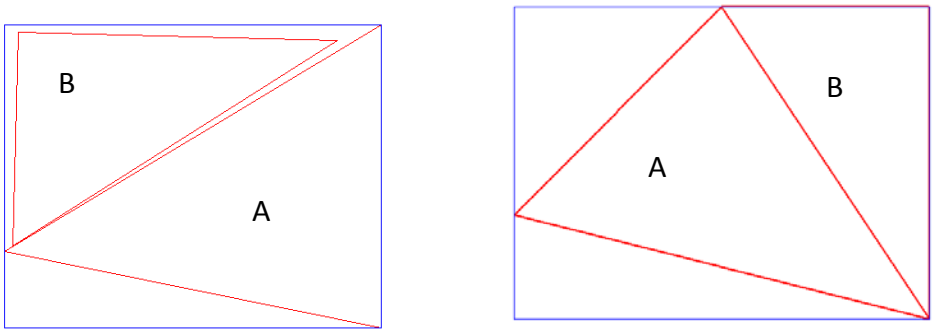


Fig. 4 Arrangments of *A* and *B* of Example 1.1, corresponding to points u_s^*, $s = 1, \ldots, 6$



(a) (b)

Fig. 5 Arrangment of polygons *A* and *B* corresponding to point u^*: a) with continuous rotations, Example 1.2, b) with discrete rotations, Example 1.3.

Example 2. We consider two irregular objects A and *B*, see Figure 6. We use Algorithm 2.

Example 2.1 Clustering of objects A and B into a rectangle R, we believe this optimal but not proved**,** problem **P1,** see Figure 6a, $F(u^*) = a^* \cdot b^* = 23.2253$. Running time is 0.431 sec.

Example 2.2 Clustering of objects A and B into a circle C, we believe this optimal but not proved, problem **P3,** see Figure 6b, $F(u^*) = r^* = 3.2599$. Running time is 0.387 sec.

Example 2.3 Clustering of objects A and B into a rectangle R taking into account minimal allowable distance ρ=0.6, we believe this optimal but not proved, problem **P1**, see Figure 6c, $F(u^*) = a^* \cdot b^* = 27.685$. Running time is 0.4067sec.

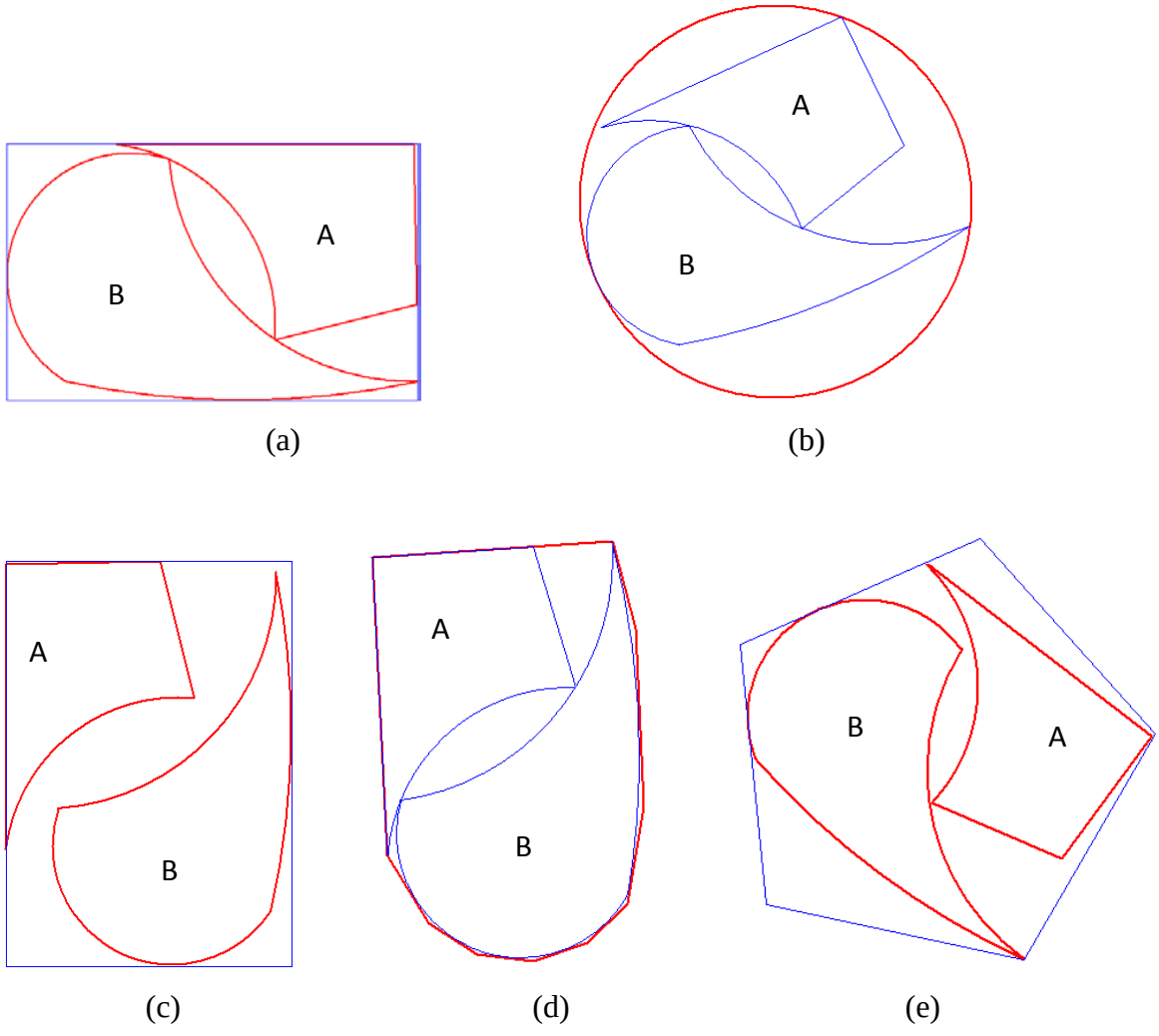


Fig.6. Arrangement of objects A and B as described in Example 2: a) minimal enclosing rectangle, b) minimal enclosing circle, c) minimal enclosing rectangle taking into account distance constraints, d) minimal enclosing m-polygon, e) minimum homothetic coefficient

Example 2.4 Polygonal approximation to the minimal convex hull of objects A and B, problem

P5, Figure 6d, $F(u^*) = S^* = 42.6835$. Running time is 0.589 sec.

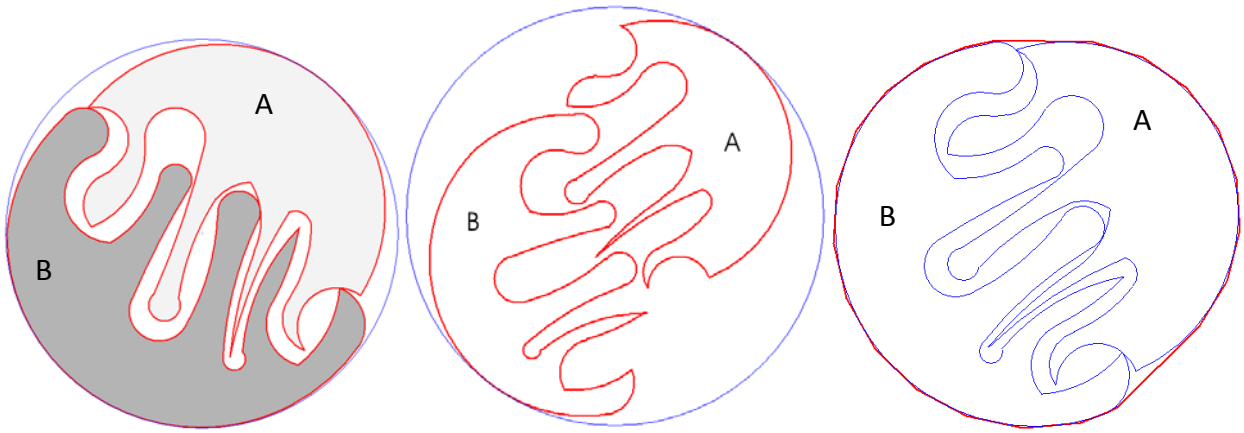
Example 2.5 Clustering of objects A and B into a convex pentagon $\Omega \equiv \alpha K$ of minimal homothetic coefficient, we believe this optimal but not proved, problem **P6**, **s**ee Figure 6e, $F(u^*) = \alpha^* = 0.5259$. Running time is 0.401 sec.

Example 3. We consider two irregular objects A and B, see Figure 7. We use Algorithm 2.

Example 3.1 . Clustering of objects A and B in a circle of minimal radius, we believe this optimal but not proved, problem **P3**, see Figure 7a, $F(u^*)$=8.5826.Running time is 0.531 sec.

Example 3.2. Clustering of objects A and B in a circle of minimal radius with allowable distance $\rho = 0.3$, we believe this optimal but not proved, problem **P3**, see Figure 7b, $F(u^*)=r^*$=11.3709. Running time is 1.235 sec.

Example 3.3. Clustering of objects A and B in a convex polygon of minimal area, we believe this optimal but not proved, problem **P5**, see Figure 7c, $F(u^*)=S^*$= 439.8638. Running time is 5.06 sec.



(a) (b) (c)

Fig 7. Minimal enclosing container of objects A and B as described in Example 3: a) circle, without distance constraints , b) circle, with distance constraints, c) convex m-polygon

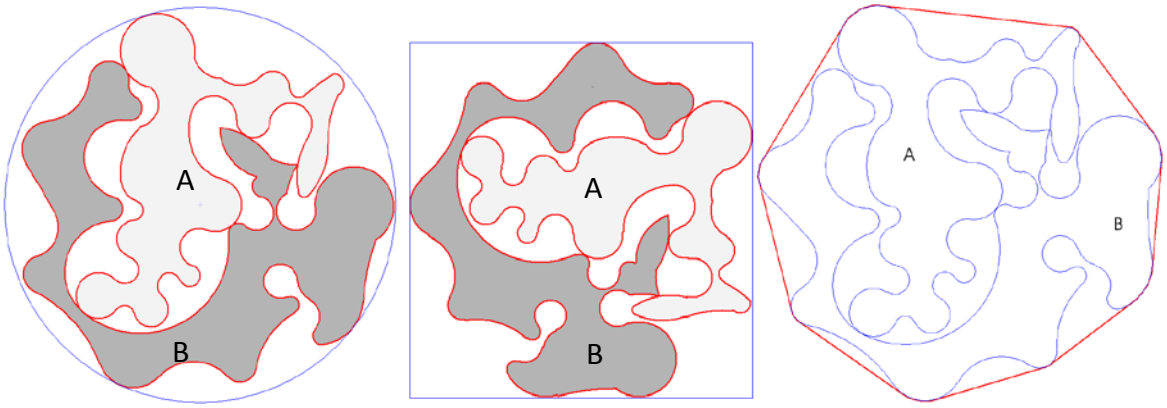
Exanple 4. We consider two irregular objects A and B, see Figure 8. We use Algorithm 2.

Example 4.1 Clustering of objects A and B in a circle of minimal radius we believe this optimal but not proved, problem **P3**, see Figure 8a, F(u^*)=17.7674. Running time is 5.031 sec.

Example 4.2 Clustering of objects A and B in a rectangle of minimal area, we believe this optimal but not proved, problem **P1**, see Figure 8b, $F(u^*)$=1121.6867. Running time is 2.938

sec.

Example 4.3 Clustering of objects *A* and *B* in a convex m-polygon, we believe this optimal but not proved, problem **P5**, see Figure 8c**,** $F(u^*)$=1736.6091. Running time is 10.375 sec.

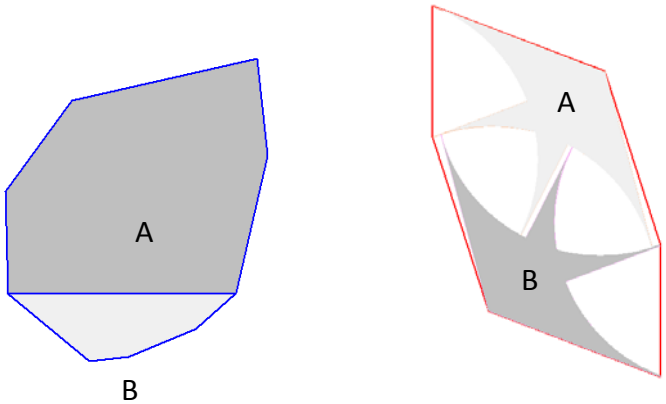


(a) (b) (c)

Fig 8. Minimal enclosing regions for objects *A* and *B* of Example 4**:** a) rectangle, b) circle, c) convex m-polygon

Example 5. The convex hull of two convex polygons A and B, the optimal solution**,** problem **P5**, see Figure 9a, $F(u^*)$ =387.5215. Running time is 5.14 sec. We use Algorithm 1.

Example 6. The convex hull for two rotated objects A and B, we believe this optimal but not proved problem **P5**, see Figure 9b, $F(u^*)$ =51.0228. Running time is 0.484 sec. We use Algorithm 2.



(a) (b)

Fig. 9 The convex hull for objects A and B: (a) two convex polygons, Example 5, (b) two non-convex objects, Example 6.

Example 7. An approximation of the convex hull for two convex polygons considering minimal allowable distance $\rho = 0.2$ between objects *A* and B, as well as, between the frontier of Ω and

each object**,** problem **P5**, see Figure 10. We consider polygons A and B, given in Appendix E, for Example 1. $F(u^*)$ =11.3211. Running time is 0.283 sec. We use Algorithm 2.

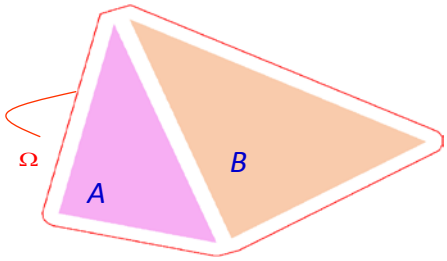
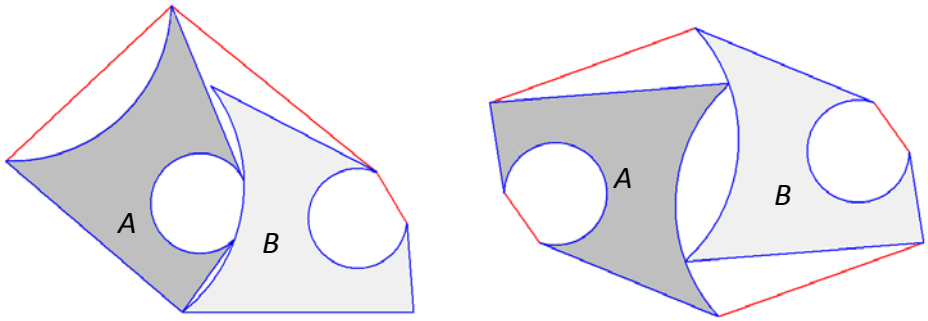


Fig.10. An approximation of the m-polygonal convex hull for two convex polygons of Example1

Example 8. An approximation of the convex hull for two non-convex objects, see Figure 11. We use Algorithm 2.

Example 8. 1. An approximation of the convex hull of minimal area for two non-convex objects, we believe this optimal but not proved, problem **P5**, see Figure 11a, $F(u^*)$ =373.5249. Running time is 0.562 sec.

Example 8. 2. An approximation of the convex hull of minimal perimeter for two non-convex objects, which looks like the optimal, problem **P4**, see Figure11b, $F(u^*)$ =55.0508 . Running time is 0.5710 sec.



(a) (b)

Fig 11. An approximation of the m-polygonal convex hull for two non-convex objects of Example 8: (a) area of the convex hull (b) perimeter of the convex hull

Example 9. Containment of two convex polygons A and B into the given polygonal container K taken from [16]. We consider the containment problem as problem **P6,** assuming that $\Omega \equiv \alpha K$. We say that the problem is solved, if $\alpha^* \le 1$. In the example we got the global minimum: $F(u^*) = \alpha^*$ =1, see Figure 12. We use Algorithm 1. Running time is 0.95 sec.

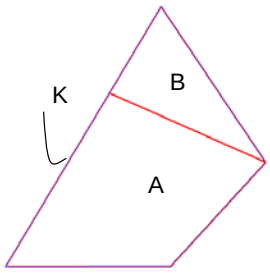


Fig. 12 Containment of two polygons A and B into container K of Example 9 with respect to point of the global minimum $u^* = (\lambda^*, u_A^*, u_B^*)$

7. Conclusions

In the paper a basic approach is presented to handle placement problems with irregular shapes, whose frontiers formed by circular-arc and/or line segments. We investigated the problem of enclosing two such objects by a rectangle or circle or convex polygon of minimal area or perimeter or homothetic coefficient by means of phi-function technique. The solution methodology can be applied to a wide range of problems in cutting and packing. The extension of the approach to the case of more than two objects, the problem of filling holes of arbitrary shapes and other forms of objectve functions is onging work for the near future publication.

References

1. Adamowicz, M., Albano, A.: Nesting two-dimensional shapes in rectanglular modules. Computer Aided Design. 8, 1, 27-33 (1976)
2. Avnaim, F., Boissonnat, J.D.: Polygon placement under translation and rotation. Informatique Theorique et Applications. 23(1), 5-28 (1989)
3. Avnaim, F., Boissonnat, J.D.: Simultaneous containment of several polygons. In Symposium on Computational Geometry. 242-247 (1987)
4. Bennell, J.A., Oliveira J.F.: The geometry of nesting problems: A tutorial. European Journal of Operational Research. 184:397-415 (2008)
5. Bennell, J.A., Oliveira J.F, A tutorial in irregular shape packing problem, Journal of the Operational Research Society, 60:s93-s105 (2009)
6. Bennell, J., Scheithauer, G., Stoyan, Y., Romanova, T.: Tools of mathematical modeling of arbitrary object packing problems. Annals of OR, 179:343-368 (2008)
7. Blazewicz, J., Drozdowski, M., Soniewicki, B., Walkowiak, R.: Two-dimensional cutting problem. basic complexity results and algorithms for irregular shapes. Found. Cont. Eng., 14(4), 137-60 (1989)
8. Burke, E.K., Hellier, R., Kendall, G., Whitwell, G.: Irregular packing using the line and arc no-fit polygon. Operations Research. 58(4), 948-970 (2010)
9. Chazelle, B., Edelsbrunner, H., Guibas, L.J.: The complexity of cutting complexes. Discrete & Computational Geometry. 4(2), 139-81 (1989)

10. Chernov, N., Stoyan, Yu., Romanova, T.: Mathematical model and efficient algorithms for object packing problem. Comput. Geometry: Theory & Appl. 43, 535-553 (2010)
11. Chernov, N., Stoyan, Yu., Romanova, T., Pankratov, A.: Phi-functions for 2D objects formed by line segments and circular arcs. Advances in Operations Research (2012). doi:10.1155/2012/346358
12. Chlebik, M., Chlebikova, J.: Hardness of approximation for orthogonal rectangle packing and covering problems. Journal of Discrete Algorithms. 7, 291-305 (2009)
13. Dori, D., Ben-Bassat, M.: Circumscribing a convex polygon by a polygon of fewer sides with minimal area additions. Computer Vision, Graphics and Image Processing.24,131-159(1983)
14. Dowsland, K.A., Dowsland, W.B.: Solution approaches to irregular nesting problems. European Journal of Operational Research. 84, 506-521 (1995)
15. Grinde, R. B., Cavalier, T. M.: Containment of a single polygon using mathematical programming. European Journal of Operational Research. 92, 368-386 (1996)
16. Grinde, R. B., Cavalier, T. M.: A new algorithm for the two-polygon containment problem. Computers & Oper. Res. 24, 231--251 (1997)
17. Grinde, R. B., Cavalier, T. M.: A new algorithm for the minimum-area convex enclosing problem. European Journal of Operational Research. 84, 522-538 (1995)
18. Han, W., Bennell, J.A., Song, X., Zhao, X.: Construction heuristics for two-dimensional irregular shape bin packing with guillotine constraints. European Journal of Operational Research. (2013) doi:10.1016/j.ejor.2013.04.048
19. Kallrath, J.: Cutting circles and polygons from area-minimizing rectangles: J Glob Optim (2009) 43:299-328
20. Li, Z., Milenkovic, V.: The complexity of the compaction problem. In 5th Canadian Conf. On Comp. Geom., Univ. Waterloo (1993)
21. Martin, R. R., Stephenson, P. C.: Putting objects into boxes. Computer Aided Design. 20(9), 506-514 (1988)
22. Milenkovic, V.: Multiple translational containment part ii: Exact algorithms. Algorithmica, 19(9):183-218 (1997)
23. Milenkovic, V.: Rotational polygon containment and minimum enclosure using only robust 2D constructions. Computational Geometry. 13(1), 3-19 (1999)
24. Milenkovic, V.: Rotational polygon overlap minimization and compaction. Computational Geometry. 10(4), 305-318 (1998)
25. Milenkovic, V., Daniels, K.: Translational polygon containment and minimal enclosure using mathematical programming. International Transactions in Operational Research. 6(5), 525-554 (1999)
26. Milenkovic, V., Sacks, E.: Two approximate Minkowski sum algorithms. Int. J. Comp. Geometry & App. 20, 485-509 (2010)
27. Wachter, A., Biegler, L. T.: On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Mathematical Programming.106, 1, 25-57 (2006)
28. Wascher, G., Hauner, H., Schuma, H.: An improved typology of cutting and packing problems. European Journal of Operational Research. 183, 1109-1130 (2007)

Acknowledgement. T. Romanova, Yu. Stoyan and A. Pankratov acknowledge the support of the Science and Technology Center in Ukraine and the National Academy of Sciences of Ukraine, grant 5710.

Appendix A: Primitive and basic objects

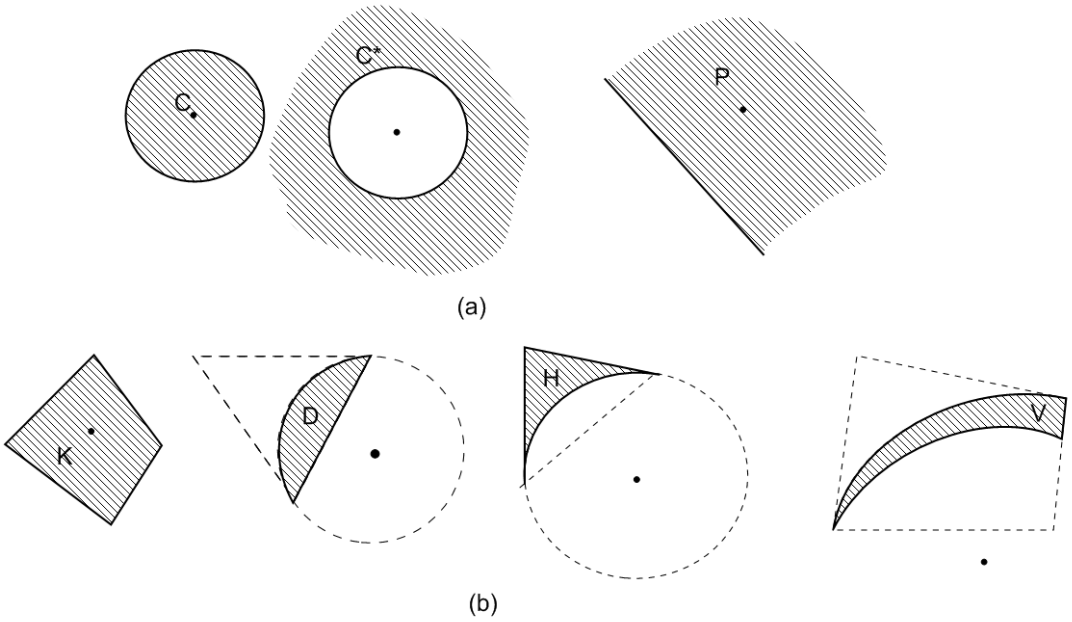


Fig. A. (a) 3 types of primitive objects (b) 4 types of basic objects

Appendix B: Description of irregular objects in an analytical form

We define phi-object T, given in Figure B, as follows: $T = T_1 \bigcup T_2$, where T_1 is a circle of radius r_1, $T_2 = T_{21} \bigcap T_{22} \bigcap T_{23}$, where T_{21} and T_{23} are half-planes, T_{22} is the complement to the interior of a circle of radius r_2 with center point (x_{22}, y_{22}). Primitive objects T_1, T_{21}, T_{22}, T_{23} are defined as follows: $T_1 = \{t \in R^2 : f_1(t) \le 0\}$, $f_1(t) = f_{11}(t) = x_t^2 + y_t^2 - r_1^2$, $T_{21} = \{t \in R^2 : f_{21}(t) \le 0\}$, $f_{21}(t) = \alpha' x_t + \beta' y_t + \gamma'$, $T_{22} = \{t \in R^2 : f_{22}(t) \le 0\}$, $f_{22}(t) = -(x_t - x_{22})^2 - (y_t - y_{22})^2 + r_2^2$, $T_{23} = \{t \in R^2 : f_{23}(t) \le 0\}$, $f_{23}(t) = \alpha'' x_t + \beta'' y_t + \gamma''$.

Now we may conclude, that $T_2 = \{t \in R^2 : f_2(t) \le 0\}$, where $f_2(t) = \max\{f_{21}(t), f_{22}(t), f_{23}(t)\}$. Thus, $T = \{t \in R^2 : f(t) \le 0\}$, where $f(t) = \min\{f_1(t), f_2(t)\}$. Note, that $\operatorname{int} T = \{t \in R^2 : f(t) < 0\}$, $frT = \{t \in R^2 : f(t) = 0\}$.

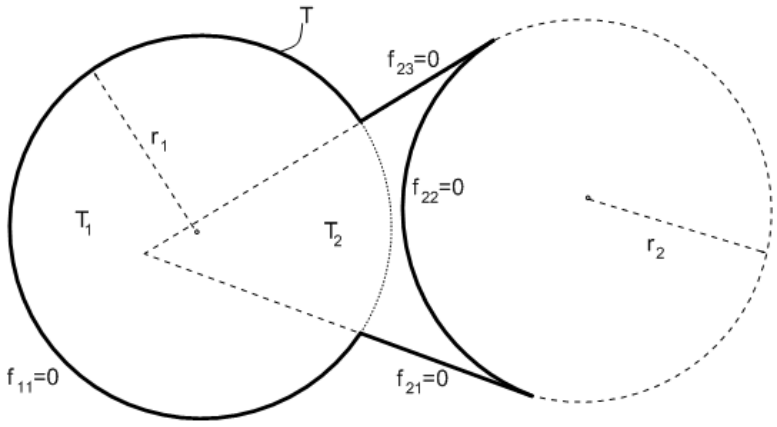
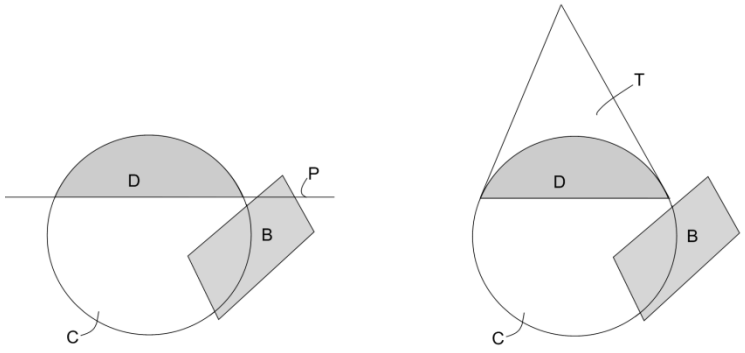


Fig. B. Definition of object T

Appendix C: Definition of circular segment D for deriving phi-functions

Given $A = P \bigcap C$ and $B = K$ convex polygon, where P is a half-plane and C is a circle. We arrange A and B as it is shown in Figure Ca. One can see that $P \cap K \neq \varnothing$ (resulting in $\Phi^{PK} < 0$) and $C \cap K \neq \varnothing$ (resulting in $\Phi^{CK} < 0$), while $A \cap K = \varnothing$ (meaning that Φ^{AB} should be positive). Therefore, $\Phi^{AB} \neq \max\{\Phi^{RK}, \Phi^{C^*K}\}$. If we take $A = T \bigcap C$, where T is a triangle with two tangent sides to circle C (see Figure Cb), then we have $\Phi^{AB} = \max\{\Phi^{TK}, \Phi^{CK}\}$.



(a) (b)

Fig. C. Definition of circular segment D for deriving phi-functions: (a) D is an intersection of circle *C* and half-plane *P*, (b) D is an intersection of circle *C* and triangle *T*

Appendix D: Phi-tree diagram

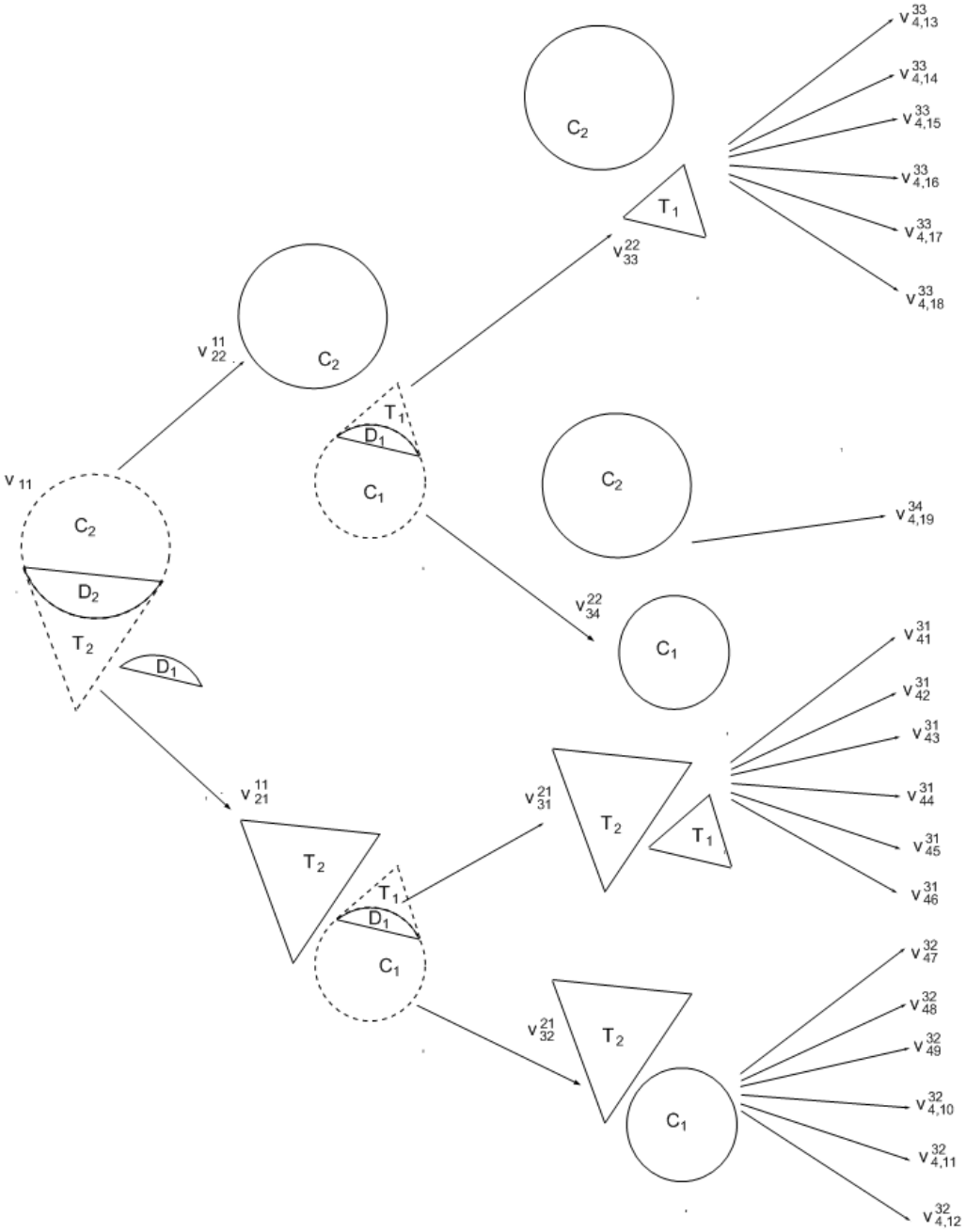


Fig. D. Diagram of phi-tree for $\Phi_{D_1 D_2} \geq 0$

APPENDIX E: Input and output data for examples

Example_1.
INPUT DATA
OBJECT A EX_1
$l_A = (2, -1, 0, 0, 2, 0, -2, 0, 0)$
OBJECT B EX_1
$l_B = (0, 0, 0, 3, 2, 0, 0, 2, 0)$
Example_1 OUTPUT DATA
Example 1.1. $u^* = (a^*, b^*, x_1^*, y_1^*, x_2^*, y_2^*) = (4.0, 3.6667, 2.0, 1.0, 0.0, 1.6667)$.
Example 1.2. $u^* = (a^*, b^*, x_1^*, y_1^*, \theta_1^*, x_2^*, y_2^*, \theta_2^*) =$ (3.5355, 2.8284, 2.1213, 1.4142, 2.3562, 0.0791, 0.7591, 6.3087)
Example 1.3. Rotation step is 30°,
$u^* = (a^*, b^*, x_1^*, y_1^*, \theta_1^*, x_2^*, y_2^*, \theta_2^*) = (4.0, 3.0, 2.0, 1.0, 0.0, 2.0, 3.0, 1.5708)$
Example_2. INPUT DATA
OBJECT A EX_2
$l_A =$ (-1.605, -2.125, -2.693, 0.829, -3.278, 1.892, -0.804, 0, 2.039, 1.369, 0, -0.2372, 2.0661, 0)
OBJECT B EX_2
$l_B =$ (2.022, -1.281, 1.843, 1.1539, 0.3449, 0.708, 2.133, 12.743, 7.836, -8.429, -2.934, -1.619, -3.632, -0.276, -4.0936).
Example_2. OUTPUT DATA
Example 2.1.
$u^* = (a^*, b^*, x_1^*, y_1^*, \theta_1^*, x_2^*, y_2^*, \theta_2^*) =$ (6.0977, 3.8089, 4.1637, 2.9426, 1.2554, 2.8937, 1.2500, -2.3398)
Example 2.2.
$u^* = (r^*, x_1^*, y_1^*, \theta_1^*, x_2^*, y_2^*, \theta_2^*) =$ (3.2599, -0.2514, 1.4905, -5.4582,-0.1020, -0.7134, -9.01344)
Example 2.3.
$u^* = (a^*, b^*, x_1^*, y_1^*, \theta_1^*, x_2^*, y_2^*, \theta_2^*) =$ (4.4249, 6.2566, 0.8663, 4.3227, 5.9678, 3.1659, 2.8858, 2.3858)
Example 2.4. m=11, $u^* = (x_1^*, y_1^*, \theta_1^*, t_1^*, \ldots, x_{11}^*, y_{11}^*, \theta_{11}^*, t_{11}^*, x_A^*, y_A^*, \theta_A^*, x_B^*, y_B^*, \theta_B^*) =$(3.7724, 0.0000, 2.3683, 0.8404, 3.1711, -0.5870, 2.8165, 0.8404, 2.3747, -0.8554, -3.0185, 0.8404, 1.5408, -0.7522, -2.5702, 0.8404, 0.8339, -0.2977, -2.1220, 1.1793, 0.2163, 0.70686, -1.6199, 4.4084, 0.0000, 5.1099, -0.0673, 3.5617, 3.5537, 5.3494, 0.5327, 0.0000, 3.5537, 5.3494, 1.3184, 1.3214, 3.8837, 4.0698, 1.5251, 2.6428, 4.0045, 1.4298, 1.7318, 1.4485, 0.9580 , 3.2641, 5.9187, 2.7233, 2.1026, 2.3254)
Example 2.5.
Pentagon K is given by a vector of coordinates of its vertices:
(7.0190, 1.4637, 1.8053, 7.2809, -5.3382, 4.1200, -4.5396 -3.6507, 3.0977, -5.2924)
m=5, $u^* = (\alpha^*, x_A^*, y_A^*, \theta_A^*, x_B^*, y_B^*, \theta_B^*) = (0.5259,$ 1.6035, 1.1955, 8.1947, -0.3486, -0.0779,

10.8685)

Example_3 INPUT DATA

OBJECT A EX_3

l_A = (4.326, 6.395, -1.433, 2.914, 6.639, 1.56, 6.169, -2.405, 3.738, 7.189, 1.333, 7.212, 1.507, -0.143, 6.908, -0.553, 8.358, 3.78, 3.226, 8.266, 2.139, 4.645, -1.335, 1.574, 3.436, 0.749, 2.385, -41.479, 26.033, 35.267, -4.337, 7.015, 0.69, -4.999, 6.826, -4.974, 6.137, -11.293, -10.967, -3.435, -1.594, 2.865, -2.1027, -3.484, 1.944, -1.523, 1.186, -3.278, -1.925, 4.439, -4.36, 2.245, 18.437, -17.211, -10.975, -8.242, 5.133, -19.365, -19.496, -10.626, -3.431, 0.187, -0.727, -4.157, 0.218, -4.551, -0.393, -6.038, -1.879, 5.022, -6.914, 1.689, 1.485, -8.213, 0.971, -9.274, 2.01, -1.738, -8.923, 0.308, -8.055, -1.197, 7.273, -2.074, 2.942)

OBJECT B EX_3

l_B = (2.493, 6.764, 1.771, 2.143, 5.028, 0.38, 5.191, -4.788, 1.364, 0.506, 4.149, 4.399, -1.876, 2.274, 4.368, 1.795, 2.554, 10.681, -0.594, -7.857, -1.702, 2.767, 8.905, 2.509, 10.614, 4.229, 1.877, -0.496, 4.671, 1.651, 4.781, 1.167, -16.555, 1.111, 17.31, -1.293, 0.931, 0.944, -1.623, 0.047, -1.214, -0.804, -10.767, 1.31, -11.271, 3.834, -0.804, -1.324, 3.524, -2.091, 3.361, -3.405, -40.094, 8.293, 36.385, -2.239, -2.301, 0.723, -2.193, -3.023, -2.003, -3.72, -4.687, -1.968, -8.407, 1.231, -4.981, -2.155, -0.867, -5.474, 0.837, -6.794, -1.946, -0.701, -5.602, -2.239, -6.794, 1.103, -3.112, -7.469, -3.423, -8.528, 7.762, 0.541, -1.854, 2.0200 -9.4743 8.4740 -0.1576 -1.2849).

Example 3. OUTPUT DATA

Example 3.1. $u^* = (r^*, x_1^*, y_1^*, \theta_1^*, x_2^*, y_2^*, \theta_2^*)$ = (8.5826, 2.6036, 4.1595, -1.9849, 1.1292, -0.5965 -4.4319)

Example 3.2. $u^* = (r^*, x_1^*, y_1^*, \theta_1^*, x_2^*, y_2^*, \theta_2^*)$ = (5.6452, 4.7846, -1.3617, -1.1846, -3.5867, -3.6332)

Example 3.3.

m=24, $u^* = (x_1^*, y_1^*, \theta_1^*, t_1^*, \ldots, x_{24}^*, y_{24}^*, \theta_{24}^*, t_{24}^*, x_A^*, y_A^*, \theta_A^*, x_B^*, y_B^*, \theta_B^*)$ = (-5.3369, -8.5664, -2.6532, 2.4302, -7.4829, -7.4261, -2.3683, 2.4302, -9.2220, -5.7287, -2.0835, 2.4302, -10.4141, -3.6120, -1.7987, 2.4302, -10.9631, -1.2436, -1.5138, 2.4302, -10.8247, 1.1826, -1.2290, 2.4302, -10.0101, 3.4722, -0.9441, 1.2151, -9.2975, 4.4564, -0.8760, 0.9460, -8.6918, 5.1831, -0.6335, 1.8920, -7.1669, 6.3030, -0.3909, 1.8920, -5.4177, 7.0239, -0.2271, 1.3731, -4.0798, 7.3330, 0.0182, 4.0874, 0.00688, 7.2587, 0.3034, 2.2215, 2.1269, 6.5950, 0.6065, 2.2215, 3.9522, 5.3288, 0.9096, 2.2215, 5.3164, 3.5755, 1.2127, 2.2215, 6.0950, 1.4949, 1.5158, 2.22150, 6.2171, -0.7232, 1.8189, 2.2215, 5.6716, -2.8767, 2.1220, 2.2215, 4.5081, -4.7695, 2.3619, 4.8634, 1.0496, -8.1884, 2.6565, 0.5084, 0.5998, -8.4255, 2.7755, 1.2151, -0.5346, -8.8605, 3.0603, 2.4302, -2.9569, -9.0577, -2.9380, 2.4302, 2.1774, 1.4609, 4.9044, -1.7436, -1.6084, 2.4573)

Example_4 INPUT DATA

OBJECT A EX_4

l_A = (0.916, -3.2835, -2.1141, -1.1925, -3.4331, -3.1599, -4.2069, 1.5176, -4.5722, -4.7624, -6.0118, -5.243, -1.9194, -7.8323, -5.8509, -9.7247, -5.5302, 6.5362, -16.1691, -4.4383, -10.9427, -0.5133, 0.531, -11.3672, -0.8321, -11.8691, -0.6587, 6.4935, -5.7318, -2.7797, -11.614, -5.5302, -10.3158, -20.9587, -9.8998, -10.6432, -9.9873, 1.0287, -9.8552, -10.6486, -9.5622, -11.6347, 12.1386, -19.1497, -4.1899, -7.9948, -8.9767, -0.8426, -7.1675, -9.1368, -6.3926, -8.8058, 1.509, -5.0049, -8.213, -3.5952, -8.7513, -2.7401, -1.0353, -9.7286, -0.3511, -7.0753, -2.5557, 0.6609, -

9.4221, 2.6318, -7.7951, 3.383, 5.924, -7.0167, 5.1053 ,-3.7343, -2.0191, 5.7366, -1.8164, 4.5707, -0.1679, 5.8141, -0.784, 2.0974, 4.2101, 5.0745, 2.1588, 2.3558, 3.9691, 2.1899, 6.1215, -0.8636, 2.1236, 6.9825, 1.721, 7.7466, 1.7119, 0.9229, 9.2611, 0.7867, 10.9676,-1.2333, 0.6886,12.1970,1.1388,13.3452,2.0740, 1.8957, 15.2761, 3.2381, 16.8571, 2.0329, 1.9223, 15.3074, -0.0714, 15.7046 -0.8439,-0.8991,15.8695,-1.5973,15.3956,1.5839,-2.9079, 14.5062, -3.3626, 12.9889, -1.2291, -3.7155, 11.8116, -2.5921, 11.3130,-1.0093,-3.5146, 11.7225, -4.2379, 11.0185, 0.9691, -4.9323, 10.3426, -4.8298, 9.3789, -1.7325, -4.6465, 7.6562, -4.5749, 5.9252, 2.3942, -4.4760, 3.5330, -4.0691, 1.1736, -3.8135, -3.4210, -2.5844, -0.6502, 0.0357, -5.8961, -4.9343, -4.0153)

OBJECT B EX_4

l_B = (-3.9051, 15.6754, 3.6105, -6.3238, 12.9949, -5.4522, 9.4912, -1.7667, -4.4782, 8.0172, -5.7334, 6.7739, -7.4716, -6.5223, 14.2038, -8.5183, 7.0037, -5.0616, -13.3461, 5.4831, -8.3588, 4.6188, 6.9000, -2.9414, 0.3454, -9.7608, -0.7063, 4.6404, -11.0770, 3.7435, -6.4367, 3.7287, 1.0321, -5.8343, 4.5668, -4.8427, 4.8531, -1.5625, -3.2958, 4.6331, -1.7954, 4.1972, 0.9511, -1.0711, 3.5808, -0.7171, 2.6980, -8.0493, 2.2783, -4.7732, 9.5968, -8.1244, -4.6532, 5.3660, -6.1871, 1.0175, -7.8433, 1.0906, -0.0017, -8.2314, -1.0922, -8.2181, -3.8299, -4.9218, -8.1712, -3.1081, -11.5444, -3.3853, -4.7112, -8.5628, -8.0775, -8.9208, 1.1200, -9.1913, -9.0393, -10.2810, -8.7803, -1.2462, -11.4934, -8.4921, -11.2655, -9.7173, 1.8333, -10.9302, -11.5197, -10.6091, -13.3247, -4.1852, -9.8762, -17.4452, -5.9678, -15.9483, 2.5006, -3.6327, -15.0539, -1.7485, -16.6979, -71.089, 51.8177, -63.4345 1.6738 -13.0436 -2.7890, 3.6411, -15.0206, 5.9870, -13.5121, 1.6693, 7.3910, -12.6092, 9.0343, -12.9031, -12.2743, 21.1169, -15.0634, 13.7224, -5.2665, 3.2491, 11.7650, -2.6732, 13.2536, 0.2149, -4.2809, 15.2149, 4.0201, 11.5659, 6.2586, 1.4523, 10.3279, 7.0180, 10.1594, 8.4605, -7.0745, 9.3387, 15.4872, 4.3461, 10.4751, 0.7968, 3.7838, 9.9105, 3.1741, 9.3975, -1.5498, 1.9882, 8.3997, 1.4863, 9.866, 2.0727, 0.8151, 11.8271, 2.8459, 11.4121, -1.7315, 4.5423, 11.0654, 5.2837, 12.6302, 0.8337, 5.6407, 13.3836, 6.4089, 13.7077, 4.1508, 2.5845, 12.0942, 1.6738, 16.1439, -20.4365, -2.8097, 36.0825)

Example 4. OUTPUT DATA

Example 4.1. $u^* = (r^*, x_1^*, y_1^*, \theta_1^*, x_2^*, y_2^*, \theta_2^*)$ = (17.7674, -1.2785, 5.0441, -2.7258, -0.2362, -2.6272, 1.8515)

Example 4.2. $u^* = (a^*, b^*, x_1^*, y_1^*, \theta_1^*, x_2^*, y_2^*, \theta_2^*)$ =(32.8975, 34.0964, 22.0452, 19.8732, 4.7927, 15.1417, 16.3673, 3.0874)

Example 4.3. u^*= $(x_1^*, y_1^*, \theta_1^*, t_1^*, \ldots, x_m^*, y_m^*, \theta_m^*, t_m^*, u_A^*, u_B^*)$, m=22, where

$(x_1^*, y_1^*, \theta_1^*, t_1^*, \ldots, x_m^*, y_m^*, \theta_m^*, t_m^*)$ = (32.1154, -1.3741, 2.4936, 9.2842, 24.7132, -6.9780, 2.6752, 0.2644, 24.4770, -7.0969, 2.8567, 10.1271, 14.7581, -9.9431, 3.0798, 0.7279, 14.0316, -9.9880, -2.9803, 0.7279, 13.3131, -9.8711, -2.7572, 0.7279, 12.6384, -9.5981, -2.5341, 11.4076, 3.2719, -3.0864, -2.17501, 0.6059, 2.9277, -2.5878, -1.8159, 12.0635, 0.0000, 9.1151, -1.4215, 0.9994, 0.14867, 10.1034, -1.0270, 14.4742, 7.6372, 22.4899, -0.6472, 1.3006, 8.6747, 23.2741, -0.2674, 1.3006, 9.9291, 23.6177, 0.1125, 16.8834, 26.7059, 21.7236, 0.5216, 0.4268, 27.0760, 21.5110, 0.9306, 11.0851, 33.6978, 12.6210, 1.0337, 0.8197, 34.1172, 11.9168, 1.3810, 1.1366, 34.3316, 10.8006, 1.6580, 9.9226, 33.4677, 0.9157, 1.8892, 0.9642, 33.1660, 0.0000, 2.1204, 0.9641, 32.6624,-0.82219,2.3517,0.7771), (u_A^*, u_B^*) =(14.6660,12.1616, 3.334, 17.4147, 4.9045, 1.6176).

Example 5. INPUT DATA

OBJECT A EX_5

l_A=(-7.2662, 1.5935, 0, -5.9413, -6.8803, 0, -3.2915, -8.7340, 0, 2.3109, -10.6633, 0 6.7020, -10.6633, 0)

OBJECT B EX_5

l_B=(-1.443, -4.819, 0. 2.67, -1.107, 0. 2.089,7 4.41, -4.991, -2.885, 3.884, 0.83, 0.55, 0, -1.443, 2.605, -5.001, -4.7927, -1.107, 8 0.11, -0.12, -5.0, -2.885, 3.884, -2.879, -1.116, 0. 0.21, -1.116, -5.0, -4.7927, -1.109)

Example 5. OUTPUT DATA

m=10, $u^* = (x_1^*, y_1^*, \theta_1^*, t_1^*, \ldots, x_m^*, y_m^*, \theta_m^*, t_m^*, u_A^*, u_B^*)$=(10.0238, -2.2864, 3.0242, 3.2338, 6.8123, -2.6651, -2.4537, 8.5767, 0.186, 2.7804, -1.5931, 8.3482, 0, 11.1266, -0.9437, 9.1846, 5.3899, 18.5634, -0.2207, 15.5024, 20.5161, 21.9577, 1.463, 7.9482, 21.3712, 14.0556, 1.8004, 11.4346, 18.7688, 2.9211, 2.4138, 0.0001, 18.7688, 2.921, 2.4138, 4.391, 15.4903, 0, 2.7454, 5.925, 6.6713, 6.4244, 5.5554, 6.6713, 6.4244, 5.5554)

Example 6. INPUT DATA

OBJECT A EX_6

l_A=(-1.4427, -4.8186, 0, 2.6700, -1.1068, 0, 2.089, 4.4005, -1, 0.830000, 0.5500, 0, -1.4427, 2.6050, 0, -0.1100, -0.1100, -1, -2.8787, -1.1163, -1, 0.2100, -1.1163, 0, -1.4427, -4.8186, -1)

OBJECT B EX_6

l_B=(-1.4427, -4.8186, 0, 2.6700, -1.1068, 0, 2.0887, 4.4005, -1, 0.8300, 0.5500, 0, -1.4427, 2.6050, 0, -0.1100, -0.1100, -1, -2.8787, -1.1163, -1, 0.2100, -1.1163, 0, -1.4427, -4.8186, -1)

Example 6. OUTPUT DATA

m=6, $u^* = (x_1^*, y_1^*, \theta_1^*, t_1^*, \ldots, x_m^*, y_m^*, \theta_m^*, t_m^*, u_A^*, u_B^*)$= (0.0889, 5.3038, -1.5932, 3.9616, 0, 9.2643, 0.3428, 5.5377, 5.2157, 7.4031, 1.2794, 5.5372, 6.8065, 2.099, 1.5484, 3.9615, 6.896, -1.8612, -2.7988, 5.5379, 1.6797, 0, -1.8622, 5.5372, 3.0618, 5.4759, 5.1604, 3.8337, 1.9273, 2.019)

Example 7.

INPUT DATA

$\rho = 0.2$ is allowable distance between polygons *A* and *B* (see input data of example 1).

OUTPUT DATA

m=16, $u^* = (x_1^*, y_1^*, \theta_1^*, t_1^*, \ldots, x_m^*, y_m^*, \theta_m^*, t_m^*, u_A^*, u_B^*)$ = (5.3365, -0.1823, 2.5860, 2.9023, 2.8708, -1.7132, 2.7833, 0.2494, 2.6372, -1.8006, 3.0773, 0.0592, 2.5782, -1.8045, 3.3713, 2.0522, 0.5799, -1.3371, -2.6874, 0.045, 0.5394, -1.3173, -2.463, 0.0451, 0.5043, -1.289, -2.2386, 0.0451, 0.4764, -1.2536, -2.0142, 0.045, 0.4571, -1.2129, -1.7898, 0.045, 0.4473, -1.169, -1.5655, 0.0451, 0.4475, -1.1239, -1.341, 3.1296, 1.1603, 1.9235, -0.3583, 0.3955, 1.5307, 2.0621, 0.4747, 4.2656, 5.3246, 0.1123, 1.0025, 0.1081, 5.3827, 0.0212, 1.5303, 0.1081, 5.3871, -0.0868, 2.0581, 0.1081, 3.2376, 0.4146, 3.3713, 2.5949, -1.6030, 17.5085)

Example 8. INPUT DATA

OBJECT A EX_8 and OBJECT B EX_8

$l_A = l_B$=(2.0, 7.0, 0, -4.0, -3.0, 0, 0, -5.0, -2.5, 1.5, -3.0, 3.0, -5.0, 0, 11.0, 0, -8.0623, 10.0, 8.0)

Example 8. OUTPUT DATA

Example 8.1. m=6, $u^* = (x_1^*, y_1^*, \theta_1^*, t_1^*, \ldots, x_m^*, y_m^*, \theta_m^*, t_m^*, u_A^*, u_B^*)$= (20.2289, -2.57832, 1.4978 4.4721, 20.5550, -7.0386, 3.1454, 11.6619, 8.8932, -6.9943, -2.4381, 11.6619, 0.0000, 0.5497, -0.7467, 11.4018, 8.3684, 8.2938, 0.6766, 13.2450, 18.6952, 0.0000, 1.0342, 3.0000, 15.9317, -5.1345, 4.1758, 6.5824, -2.5603, 4.8755)

Example 8.2. m=8, $u^* = (x_1^*, y_1^*, \theta_1^*, t_1^*, \ldots, x_m^*, y_m^*, \theta_m^*, t_m^*, u_A^*, u_B^*)$ = (0, 4.4189, -0.3471, 10.6193, 9.9858, 8.0318, 0.3941, 9.4340, 18.696, 4.4093, 0.9527, 3, 20.435, 1.9643, 1.4164, 4.4721, 21.123, -2.4546, 2.794, 10.6193, 11.1372, -6.0674, -2.7475, 9.434, 2.4264, -2.445, -2.1889, 3, 0.688, 0, -1.7252, 4.4721, 16.3601, -0.9331, 4.0943, 4.7629, 2.8974, 0.9527)

Example 9. INPUT DATA

Polygonal container K is given by a vector of coordinates of its vertices:

K: (0, 0, 19, 0, 30, 12, 18, 30)

OBJECT A EX_9

l_A =(30.5, 16, 0, 11.5, 16, 0, 0.5, 4, 0, 18.5, -4, 0)

OBJECT B EX_9

l_B =(10, 29.3333, 0, 22, 11.3333, 0, 28, 21.3333, 0)

OUTPUT DATA

$u^* = (\alpha^*, u_A^*, u_B^*)$ =(1.0, 30.5, 16.0, -3.141593, 40.0, 41.333334, -9.4247779602)