Exhaustive Interval-based 2D Shape Registration Under Similarity Transformation

Verlein Radwan[a,*], Simon Rohou[b] and Gilles Trombettoni[a]

[a]LIRMM, University of Montpellier, CNRS, 161 Rue Ada, Montpellier, 34095, France
[b]ENSTA, Lab-STICC, 2 Rue François Verny, Brest, 29200, France

ARTICLE INFO

Keywords:
Similarity transformation
Procrustes analysis
Registration
Separators
Interval analysis

ABSTRACT

Given two bounded sets of $\mathbb{R}^2$ (surfaces, shapes), we want to find all possible *similarity transformations* that map the two sets together, namely a composition of translation, rotation and uniform scaling. Due to possible shape symmetries, this problem may have several discrete solutions. This paper provides a method for approximating them all, exhaustively and rigorously.

Our approach offers a sequence of operations performed on sets, which does not apply to point clouds. A key idea is to use a Procrustes-like approach, *i.e.* to move from a four-dimensional transformation to a preliminary one-dimensional rotation by first computing the smallest bounding circles for both sets. Subsequent operations on sets allow to deduce translation and scaling parameters values.

To our knowledge, this paper proposes the first set membership algorithm, based on interval methods, that can perform these set operations in an exhaustive and rigorous way. The proposed method uses advanced mathematical interval tools called *separators* that can operate on sets and implement the whole approach in a concise way, eventually providing a reliable approximation of the feasible similarity transformations.

1. Introduction

A similarity transformation (or similitude) of a Euclidean space is a bijection $\mathbf{f} : \mathbb{R}^n \to \mathbb{R}^n$ from the space onto itself that can be expressed as:

$$\mathbf{f}(\mathbf{x}) = k\mathbf{R}\mathbf{x} + \mathbf{t} \tag{1}$$

where $k \in \mathbb{R}$ is a positive real number called the *ratio of similarity*, $\mathbf{R} \in \mathbb{R}^{n\times n}$ is a rotation matrix and $\mathbf{t} \in \mathbb{R}^n$ is a translation vector. These parameters express respectively uniform scaling, rotation, and translation transformations. Because all distances are multiplied by the same ratio k, a similarity transformation has the particularity to preserve some properties including perpendicularity, parallelism, midpoints, as well as planes, lines and angles. A *registration* problem consists in identifying the parameters k, $\mathbf{R}$, $\mathbf{t}$ depicting this transformation. Note that in this paper, we will not consider non-uniform deformations.

This paper focuses on the *shape registration* problem, that can be stated as follows. Let $\mathbb{A}$ and $\mathbb{B}$ be bounded sets of $\mathbb{R}^2$. We want to exhaustively describe all 2D similarity transformations mapping $\mathbb{A}$ and $\mathbb{B}$. In this paper, a 2D similarity transformation will be characterized by the function $\mathbf{f}_\mathbf{p}(\cdot)$: a composition of rotation $\theta \in \mathbb{R}/(2\pi\mathbb{Z})$, translation $\mathbf{t} \in \mathbb{R}^2$ and uniform scaling $k \in \mathbb{R}^+$, whose parameters will be specified in the vector $\mathbf{p} = (\theta, \mathbf{t}, k)^\intercal \in \mathbb{S}$, where $\mathbb{S} = \big(\mathbb{R}/(2\pi\mathbb{Z})\big)\times\mathbb{R}^2\times\mathbb{R}^+$ denotes the set of parameters of all similarity transformations. The shape registration problem therefore amounts to characterizing all feasible $\mathbf{p}$ in $\mathbb{S}$ such that[1]:

$$\mathbf{f}_\mathbf{p}(\mathbb{A}) = \mathbb{B} \tag{2}$$

*Corresponding author
v.radwan@hotmail.fr (V. Radwan); simon.rohou@ensta.fr (S. Rohou); gilles.trombettoni@lirmm.fr (G. Trombettoni)
https://www.simon-rohou.fr (S. Rohou); https://www.lirmm.fr/~trombetton/ (G. Trombettoni)
ORCID(s): 0000-0001-6232-9918 (S. Rohou); 0000-0002-5283-7203 (G. Trombettoni)

[1]More rigorously, Eq. (2) is equivalent to the following double inclusion: $\big(\forall\mathbf{b} \in \mathbb{B}, \exists\mathbf{a} \in \mathbb{A} \mid \mathbf{f}_\mathbf{p}(\mathbf{a}) = \mathbf{b}\big) \wedge \big(\forall\mathbf{a} \in \mathbb{A}, \exists\mathbf{b} \in \mathbb{B} \mid \mathbf{f}_\mathbf{p}(\mathbf{a}) = \mathbf{b}\big)$

Using homogeneous coordinates, the registration problem is classically formulated as the following linear expression:

$$\begin{pmatrix} \mathbb{B} \\ 1 \end{pmatrix} = \left(\begin{array}{cc|c} & k\mathbf{R}_\theta & \mathbf{t} \\ \hline 0 & 0 & 1 \end{array}\right) \begin{pmatrix} \mathbb{A} \\ 1 \end{pmatrix}, \tag{3}$$

where $\mathbf{R}_\theta$ is the 2-dimensional rotation matrix, of rotation parameter θ.

The registration problem arises in many applications across several fields, including robotics, geospatial data conflation for cartography and GIS Safra, Kanza, Sagiv and Doytsher (2013), medical imaging, and factor analysis. In *data fusion*, multiple acquisitions representing the same subject can be aligned and combined to improve image resolution Irani and Peleg (1991), enable tumor monitoring, or perform multimodal data fusion Woods, Mazziotta, Cherry et al. (1993); similar techniques are also used in stereo vision Lucas and Kanade (1981) and in *Structure from Motion* (SfM) Fard and Peña-Mora (2007). In *motion estimation*, registration provides the alignment parameters between successive frames in a video Anandan (1989), allowing motion to be estimated for tasks such as video compression Sullivan and Baker (1991).

Our approach relies on the assumption that the two sets to be matched, $\mathbb{A}$ and $\mathbb{B}$, are bounded and fully enclosed, typically to account for uncertainties. Specifically, we assume that $\mathbb{A}^- \subseteq \mathbb{A} \subseteq \mathbb{A}^+$ and $\mathbb{B}^- \subseteq \mathbb{B} \subseteq \mathbb{B}^+$, where the enclosing sets $\mathbb{A}^-$, $\mathbb{A}^+$, $\mathbb{B}^-$ and $\mathbb{B}^+$ are computer-representable.

While $\mathbb{A}$ and $\mathbb{B}$ may be only partially observed in some contexts (e.g., due to occlusions in image processing), our assumption remains valid for various practical applications, including cartography, state estimation, mechanical assembly and geometric puzzles Huang, Flöry, Gelfand, Hofer and Pottmann (2006). Several examples are detailed below.

Matching cadastral maps

A common GIS application involves matching sub-sections of two cadastral maps produced at different periods, for instance, to identify new, demolished, or preserved buildings. The first step of this process is to compute the transformation between the maps by registering two specific regions, one from each map, where the user identifies potentially identical shapes. If this initial correspondence is correct, our tool guarantees the retrieval of the exact transformation between the maps; otherwise, it formally indicates that no such transformation exists.

Terrain-based state estimation using aerial or satellite imagery

Figure 1 illustrates how registration can assist the *state estimation* of an aircraft or a drone.

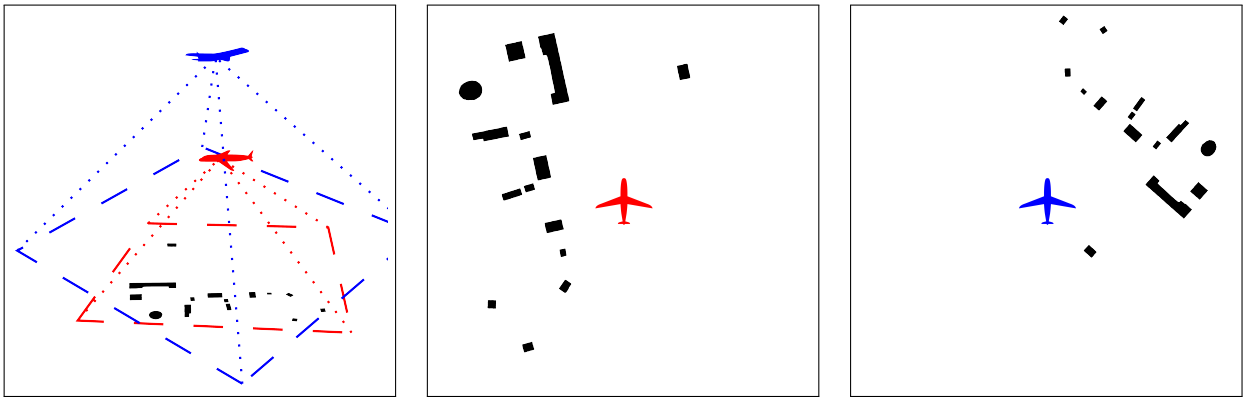


Figure 1: Two aerial views showing different perceptions of building shapes after a segmentation step. In this context, registration involves estimating the changes in position, altitude, and orientation between two timestamps. This is a fundamental task in state estimation for mobile robotics. These illustrations are based on real photographs taken at the Domaine d'O park in Montpellier.

When perception is based on camera imagery, the aircraft's state estimation can be framed as a registration problem under similarity transformations. Let $\mathbf{g}$ be an observation function and let $\mathbb{Y}_1 = \mathbf{g}\left(\mathbf{x}(t_1)\right)$, $\mathbb{Y}_2 = \mathbf{g}\left(\mathbf{x}(t_2)\right)$ be camera outputs at times t_1, t_2, respectively. The transformation $\mathbf{p}_{1\to2}$ that satisfies $\mathbb{Y}_2 = \mathbf{f}_{\mathbf{p}_{1\to2}}\left(\mathbb{Y}_1\right)$ is directly related to the evolution of the aircraft's state from t_1 to t_2: $\mathbf{g}\left(\mathbf{x}(t_2)\right) = \mathbf{f}_{\mathbf{p}_{1\to2}}\left(\mathbf{g}\left(\mathbf{x}(t_1)\right)\right)$. The transformation $\mathbf{p}_{1\to2}$ can be computed when common features are identified in both images, a process often referred to as *loop closure* Rohou, Franek, Aubry and Jaulin (2018). Given the state $\mathbf{x}(t_1)$ and the registration vector $\mathbf{p}_{1\to2}$, one can derive an estimate of $\mathbf{x}(t_2)$, thereby refining the aircraft's localization.

In this vision-based context, shape registration is particularly effective provided that bounded sets can be extracted (typically through segmentation algorithms based on color or geometric features) and that the aerial perspective does not introduce significant non-linear deformations. This assumes a downward-looking camera at a sufficient altitude, a moderate field of view, and a relatively planar scene. While restricting the problem to bounded sets may limit some applications, it remains highly relevant for a wide range of image matching and robotic tasks.

State estimation is a critical component of the Simultaneous Localization And Mapping (SLAM) problem Durrant-Whyte and Bailey (2006). SLAM frameworks combine robot dynamics with environmental perception to concurrently estimate the vehicle's trajectory, $\mathbf{x}(t)$, and reconstruct the map of the surroundings. For instance, as illustrated in Figure 1 (right), the estimated position of the aircraft (in blue) is refined by integrating its evolution equation with our interval registration approach, which processes the image acquired from a different viewpoint (in red). Consequently, the algorithm proposed in this work can serve as a robust component for integration into broader SLAM architectures.

Note that this example should be understood as an illustrative SLAM-related scenario, not as a general treatment of SLAM under arbitrary viewpoint changes. The proposed method applies when the dominant deformation between observations is well approximated by a similarity transformation, while affine or projective effects caused by substantial viewpoint changes are outside the scope of this work.

Mechanical assembly

A typical application in mechanical assembly involves the registration of components on a conveyor belt using a vision system, prior to robotic pick-and-place operations. In this scenario, each detected item must be matched against a set of reference objects stored in a database.

Notably, if an item exhibits intrinsic symmetries, our algorithm is capable of identifying the full set of valid transformations. Furthermore, in the presence of pseudo-symmetries (which often generate parasitic, non-solution, transformations), our method effectively filters out these ambiguities, as demonstrated by the *stretched cross* example in Section 6.

Finally, we emphasize that our contribution is primarily theoretical and algorithmic. Our method has been integrated into the Codac library and is intended to be used as a specialized component within complex application pipelines rather than as a standalone end-user solution.

Outline

After a review of existing methods and their limits in the next section, the contribution is outlined in Section 3. The proposed approach is mainly based on *separators* that are mathematical tools describing and operating sets, introduced in Section 4. The implementation of the contribution comes with Section 5 in which the separators used are described as well as the complete resolution algorithm. We provide some test cases in Section 6, highlighting the pros and cons of the approach, before concluding.

2. State of the art

Among the existing methods that tackle problems of similarity transformations, *Procrustes Analysis* Gower and Dijksterhuis (2004) seeks the transformation $\mathbf{M}$ that aligns two point cloud sets $\mathbf{A}$ and $\mathbf{B}$ as closely at possible, *i.e.* that minimizes the Frobenius norm of a difference derived from (2):

$$\mathbf{M} = \arg\min_{\mathbf{M}'} ||\mathbf{B} - \mathbf{M}'\mathbf{A}||_F. \tag{4}$$

where the Frobenius norm $||\mathbf{X}||_F$ is defined by $\mathrm{Tr}(\mathbf{X}^\intercal\mathbf{X})$. The matrix $\mathbf{M}$ is subject to further restrictions, leading to different variants of the problem.

2.1. Orthogonal Procrustes problem

The core variant of the Procrustes Analysis is the *Orthogonal Procrustes Problem* Green (1952); Cliff (1966); Schönemann (1966), where $\mathbf{M}$ is orthogonal, *i.e.* a rotation with or without reflection :

$$\mathbf{M}\mathbf{M}^{\intercal} = \mathbf{M}^{\intercal}\mathbf{M} = 1. \tag{5}$$

Several solving methods are based on matrix decomposition such as *Singular Value Decomposition* (SVD) Schönemann (1966) or orthonormal matrices Horn, Hilden and Negahdaripour (1988). Other methods are based on quaternions Horn (1987).

2.2. Procrustes problem under similarity

As for similarity transformations, the principle presented in Schönemann and Carroll (1970) and replicated to a certain extent in this paper, is to first perform a standardization of the two sets in two motions. This method, called *ProcrustesPair* in the following, is described in Figure 2.

1. Center the sets w.r.t. their centroid, in order to cancel out the translation part of the transformation;
2. Pre-scale the sets so that they have the same total sum-of-squares (or Frobenius norm);
3. Solve the Orthogonal Procrustes Problem on these sets;
4. Compute the scaling parameter according to Schönemann and Carroll (1970);
5. Compute the translation parameter by matching centroids.

Figure 2: Description of the `ProcrustesPair` method

The pre-scaling step 2 is optional since the Orthogonal Procrustes is invariant to set scales, but it proves to be more efficient when handling more than two sets with different magnitudes. Also note that if we ignore Step 4, this method is suited for the less complicated rigid transformation, *i.e.* a composition of rotations, reflections and translations.

2.3. Registration between unbalanced sets: the ICP algorithm

The problem considers a relaxed version of the registration problem, where we have an inclusion relationship rather than the equality of (2):

$$\mathbf{f}_{\mathbf{p}}(\mathbf{A}) \subset \mathbf{B}. \tag{6}$$

The Iterative Closest Point (ICP) Besl and McKay (1992); Chen and Medioni (1992); Zhang (1994) algorithm tackles this variant for a rigid transformation. The algorithm takes two 2D or 3D point clouds as input: the source $\mathbf{A}$ and the reference (or model) $\mathbf{B}$. The general outline of this method is then as follows:

1. initialization: we are given an initial transformation $\mathbf{M}$,
2. choose $\mathbf{B}'$ as a subset of $\mathbf{B}$ of the points closest to $\mathbf{MA}$,
3. apply `ProcrustesPair` to $(\mathbf{A}, \mathbf{B}')$ to estimate $\mathbf{M}$,
4. return to step 2 until convergence.

The main drawback of this method is that it often falls into local minima due to its iterative process. `Go-ICP` Yang, Li, Campbell and Jia (2016) is a global variant of ICP that does not handle similarity transformations, while Fast Global Registration (`FGR`) Zhou, Park and Koltun (2016) provides a faster global alternative by computing the correspondences once. More recent methods, such as `TEASER` Yang, Shi and Carlone (2021), achieve robust and certifiable estimation of the transformation even in the presence of a large number of outliers.

2.4. Generalized Procrustes Analysis (GPA)

Generalized Procrustes Analysis (GPA) Gower (1975) is another variant of the problem where we want to align several configurations at once, and get a consensus form $\mathbf{B}$, *i.e.* an average of all configurations $\mathbf{A_1}, \dots, \mathbf{A_m}$.

Moreover, we want to compute all the similarity transformations $\mathbf{M_1}, \dots, \mathbf{M_m}$ such that:

$$\mathbf{B}, \mathbf{M_1}, \dots, \mathbf{M_m} = \underset{\mathbf{B},\mathbf{M_1},\dots,\mathbf{M_m}}{\arg\min} \sum_{i=1}^{m} ||\mathbf{B} - \mathbf{M_i}\mathbf{A_i}||_{\mathbf{F}}. \tag{7}$$

Classical methods Gower (1975); Ten Berge (1977); Rohlf and Slice (1990) uses an approach similar to ICP:

1. initialization: we are given a set of initial transformation, *i.e.* $(\mathbf{M_1}, \mathbf{M_2}, \dots, \mathbf{M_m}) = (\mathbf{I}, \mathbf{0}, \dots, \mathbf{0})$,
2. choose $\mathbf{B}$ as the mean of $\mathbf{M_i}\mathbf{A_i}$,
3. apply `ProcrustesPair` to $(\mathbf{A_i}, \mathbf{B})$ to actualize all $\mathbf{M_i}$,
4. return to step 2 until convergence.

2.5. Learning-based approaches for registration

Deep learning techniques have recently emerged as a powerful tool for tackling registration problems. A first line of work consists in learning descriptive features from point clouds, which can then be used within classical registration pipelines seen above. Architectures such as PointNet and PointNet++ Qi, Su, Mo and Guibas (2017) and DGCNN Wang, Sun, Liu, Sarma, Bronstein and Solomon (2019) have been proposed to extract global and local geometric features, enabling more robust correspondence estimation.

Building upon these feature representations, DCP Wang and Solomon (2019) embeds the input point clouds into a learned feature space. These embeddings are then updated through an attention-based module so as to encode contextual information between the two point clouds. The final rigid transformation is extracted from this soft matching with SVD. In particular, DCP can provide a good initialization for ICP, enabling it to converge to a global optimum in cases where it would otherwise fail.

A comprehensive review of deep learning approaches for point cloud registration can be found in Chen, Feng, Ma, Zhao and Wang (2023).

2.6. Discussion about different variants of statistical methods and limitations

In practical applications, the sets $\mathbb{A}$ and $\mathbb{B}$ are seldom observed directly; instead, they are represented by finite samples, often corrupted by noise. Consequently, a perfect mapping between two datasets is rarely achievable. In this context, the best-fit alignment obtained from (4) is generally considered as a suitable approximation of the underlying registration problem (2).

However, conventional statistical or optimization-based methods exhibit several critical shortcomings:

- **False convergence:** They invariably produce a single "best" solution even when no exact mapping exists.
- **Ambiguity and symmetries:** They typically converge to a unique solution, failing to account for cases where multiple global minima exist due to symmetries (see Figure 3).
- **Lack of reliability:** Being non-guaranteed, these methods may converge to local minima, leading to incorrect matches.

While orthogonal and similarity Procrustes methods may suffer from local numerical issues, the quaternion-based closed-form solution proposed by Horn (1987) can compute a global optimum. However, even a globally optimal alignment may fail to reflect the true registration when data provide a noisy or non-exhaustive observations of the underlying sets.

In common with many optimization-based frameworks, allowing scaling in this context can cause the method to diverge towards degenerate solutions where $k = 0$. This prevents meaningful solutions from being found (even for global versions such as Go-ICP Yang et al. (2016)), a phenomenon documented in Bonneel and Coeurjolly (2019).

In contrast, the approach proposed in this paper leverages set-theoretic and set-membership (interval) methods. This paradigm ensures that the search is both exhaustive and guaranteed, providing a rigorous characterization of all feasible transformations.

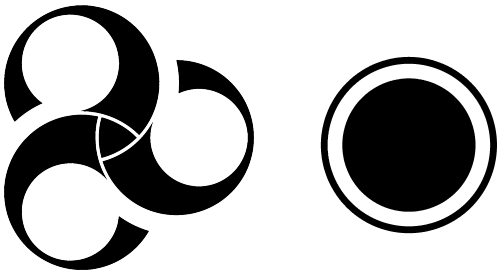


Figure 3: Examples of registration problems with non-unique solutions: a Triskelion (left) exhibiting discrete rotational symmetry, resulting in three distinct solutions, and a disk/ring configuration (right) where any rotation constitutes a valid solution.

2.7. Existing set-theoretic approaches for registration

To develop an exhaustive framework capable of capturing the full multiplicity of solutions, our methodology leverages set-theoretic tools. These techniques are particularly effective for characterizing feasible solution sets by systematically pruning regions of the search space that are guaranteed to contain no valid solutions. In the context of registration, this yields a rigorous and reliable approximation of the set $\mathbb{P} = \left\{\mathbf{p} \in \mathbb{S} \mid \mathbf{f}_{\mathbf{p}}(\mathbb{A}) = \mathbb{B}\right\}$.

A formal foundation for registration within the set-theoretic paradigm is provided in Desrochers (2018), which introduces several operators adapted to this problem. Our work builds upon these same fundamentals, specifically the concept of *separators* (detailed in Section 4).

However, a direct application of these methods to the registration problem often proves computationally intractable due to the high dimensionality of the parameter space $\mathbb{P}$. The core contribution of this paper is the introduction of a Procrustes-like decomposition: by breaking down the original problem into a sequence of lower-dimensional sub-problems, we circumvent the "curse of dimensionality" and enable the efficient use of set-theoretic tools.

The following section formalizes the set-theoretic approach used to characterize $\mathbb{P}$, while the numerical tools and interval-based algorithms required for its computation are detailed in Sections 4 and 5.

3. A Procrustes-like set-theoretic approach

As introduced in Section 1, 2D shapes registration under similarity transformations involves a four-dimensional mapping. The problem addressed in this paper consists of a set-inversion task to approximate the set of feasible transformations $\mathbb{P}$, given two bounded sets $\mathbb{A}$ and $\mathbb{B}$. Set-inversion is notoriously challenging in high-dimensional spaces, primarily due to the conventional reliance on branch-and-bound methods that perform bisections across every dimension Jaulin and Walter (1993). In our four-dimensional parameter space, bisections pose a significant bottleneck for real-time applications such as SLAM, in robotics. Consequently, we treat the branching strategy as a last resort: by adopting a Procrustes-like approach, we reformulate the problem into a rotational mapping, thereby restricting the branching process to a single dimension.

An overview of this methodology is illustrated in Figure 4, and its core stages are detailed below.

Step 1: Minimum Bounding Circle (MBC)

The cornerstone of this method lies in the determination of the unique MBC for $\mathbb{A}$ and $\mathbb{B}$. This task relates to the well-established "minimum covering circle" or "smallest enclosing circle" problems Sylvester (1857), which aim to identify the minimal circle containing a given set of points. In other contexts, this is recognized as the unweighted case of the 1-center optimization problem. While the smallest circle enclosing a finite set of points can be computed in linear time Megiddo (1983), classical algorithms are not directly applicable here due to the continuous set-based nature of our inputs and the requirement for guaranteed, exhaustive outputs.

When extended to continuous sets, the problem involves identifying the bounding circle of the infinite points in a set $\mathbb{X}$ with the smallest radius. The first contribution of this paper is to provide a set-theoretic solution for computing such a circle for bounded sets. This approach is further developed in Section 5 through the use of separators and projections. Consequently, for any two bounded sets $\mathbb{A}$ and $\mathbb{B}$, we can identify two unique vectors $\mathbf{c}^{\mathbb{A}}$ and $\mathbf{c}^{\mathbb{B}}$ in $\mathbb{R}^2 \times \mathbb{R}^+$, representing their respective centers $\mathbf{c}_{1,2}$ and radii c_3.

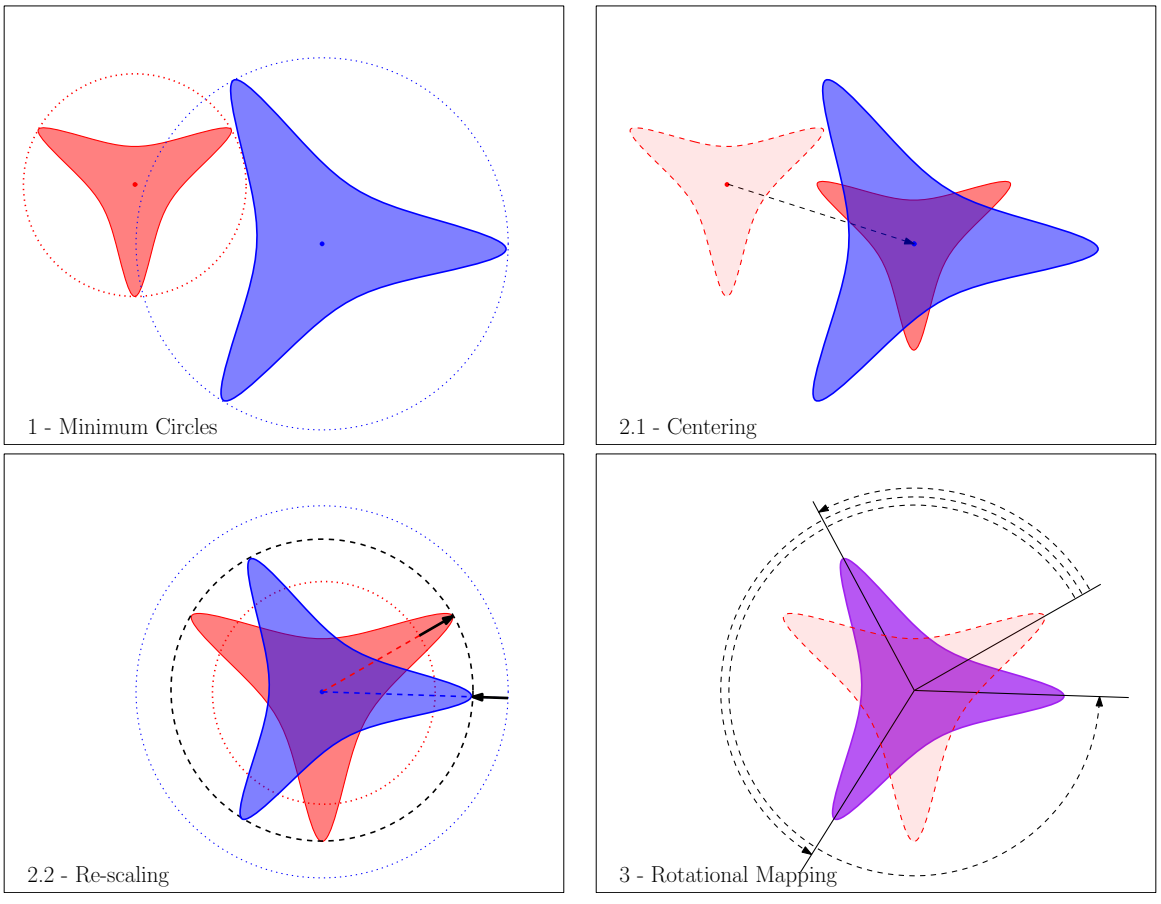


Figure 4: Overview of the Procrustes-like approach. (1) Minimum Bounding Circles (MBC) identification: the unique MBC is computed for each input set, $\mathbb{A}$ and $\mathbb{B}$. (2) Normalization: (2.1) Centering: both sets are translated to a common origin based on their MBC centers; (2.2) Scaling: sets are rescaled according to the ratio of their radii. (3) Rotational mapping: the feasible rotation parameters are determined by rotating the normalized sets. In this instance, three valid rotations are identified, reflecting the intrinsic symmetries of the initial shapes.

The MBC possesses several advantageous properties, notably existence and uniqueness for any bounded set. The resulting vectors, $\mathbf{c}^{\mathbb{A}}$ and $\mathbf{c}^{\mathbb{B}}$, serve as normalization parameters for the subsequent stage of the algorithm. In the remainder of this paper, the function $\texttt{MinCircle} : \mathcal{P}(\mathbb{R}^2) \to \mathbb{R}^3$ denotes the operation that returns the minimum enclosing circle of a bounded set.

Step 2: Normalization (Centering and Scaling)

The normalization step enables the subsequent rotational mapping by ensuring that the sets are compared within a canonical frame. Specifically, the sets $\mathbb{A}$ and $\mathbb{B}$ are transformed into their normalized counterparts, $\mathbb{A}_N$ and $\mathbb{B}_N$, through a sequential process of translation and scaling defined as follows:

$$\mathbb{A}_N := \left(\mathbb{A} - \mathbf{c}^{\mathbb{A}}_{1,2}\right)/c^{\mathbb{A}}_3 \quad \text{and} \quad \mathbb{B}_N := \left(\mathbb{B} - \mathbf{c}^{\mathbb{B}}_{1,2}\right)/c^{\mathbb{B}}_3. \tag{8}$$

Step 3: Rotational mapping

This stage aims to determine the exhaustive set of rotations that align the two normalized sets. This process characterizes the feasible rotation space Θ, which is formally defined as follows:

$$\Theta := \left\{\theta \in \mathbb{R}/(2\pi\mathbb{Z}) \mid \mathbb{B}_N = \mathbf{R}_\theta \cdot \mathbb{A}_N\right\} \tag{9}$$

The practical implementation of this phase, utilizing set-membership methods to provide a guaranteed approximation of the rotation parameters, is detailed in Section 5.

3.1. Correctness of the approach

The normalization procedure introduced above relies on the assumption that the parameters of the MBC effectively decouple the translation and scaling factors from the similarity transformation. We hereafter provide a formal justification for this approach.

Proposition 3.1. *Let $\mathbf{f_p}(\cdot)$ be a similarity transformation characterized by a scaling factor k, a rotation matrix $\mathbf{R}_\theta$, and a translation vector $\mathbf{t}$. If there exists a parameter vector $\mathbf{p}$ such that $\mathbb{B} = \mathbf{f_p}(\mathbb{A})$, then the normalized sets $\mathbb{A}_N$ and $\mathbb{B}_N$ are related by only a pure rotation $\mathbf{R}_\theta$.*

Proof. The MBC of set $\mathbb{A}$ is uniquely defined by its center $\mathbf{c}^{\mathbb{A}}_{\mathbf{1,2}}$ and radius $c^{\mathbb{A}}_3$ as follows:

$$\mathbf{c}^{\mathbb{A}}_{\mathbf{1,2}} = \arg \min_{\mathbf{x}\in\mathbb{R}^2} \max_{\mathbf{a}\in\mathbb{A}} \|\mathbf{a}-\mathbf{x}\| \tag{10}$$

$$c^{\mathbb{A}}_3 = \min_{\mathbf{x}\in\mathbb{R}^2} \max_{\mathbf{a}\in\mathbb{A}} \|\mathbf{a}-\mathbf{x}\| \tag{11}$$

Considering the set $\mathbb{B} = \mathbf{f_p}(\mathbb{A})$, its MBC radius $c^{\mathbb{B}}_3$ satisfies:

$$\begin{aligned}
c^{\mathbb{B}}_3 &= \min_{\mathbf{y}\in\mathbb{R}^2} \max_{\mathbf{b}\in\mathbb{B}} \|\mathbf{b}-\mathbf{y}\| \\
&= \min_{\mathbf{x}\in\mathbb{R}^2} \max_{\mathbf{a}\in\mathbb{A}} \|\mathbf{f_p}(\mathbf{a}) - \mathbf{f_p}(\mathbf{x})\| \quad \text{since } \mathbf{f_p} \text{ is bijective} \\
&= \min_{\mathbf{x}\in\mathbb{R}^2} \max_{\mathbf{a}\in\mathbb{A}} \|(k\mathbf{R}_\theta\mathbf{a}+\mathbf{t}) - (k\mathbf{R}_\theta\mathbf{x}+\mathbf{t})\| \\
&= \min_{\mathbf{x}\in\mathbb{R}^2} \max_{\mathbf{a}\in\mathbb{A}} k\|\mathbf{R}_\theta(\mathbf{a}-\mathbf{x})\| \\
&= k \min_{\mathbf{x}\in\mathbb{R}^2} \max_{\mathbf{a}\in\mathbb{A}} \|\mathbf{a}-\mathbf{x}\| \quad \text{as } \mathbf{R}_\theta \text{ is orthogonal} \\
&= k\, c^{\mathbb{A}}_3
\end{aligned}$$

Using the same reasoning, and paying attention to the change of variables in the second line, the center of the MBC for $\mathbb{B}$ is given by:

$$\mathbf{c}^{\mathbb{B}}_{\mathbf{1,2}} = \mathbf{f_p}(\mathbf{c}^{\mathbb{A}}_{\mathbf{1,2}}). \tag{12}$$

Finally, substituting $\mathbf{c}^{\mathbb{B}}_{\mathbf{1,2}}$ and $c^{\mathbb{B}}_3$ into the definition of the normalized set $\mathbb{B}_N$:

$$\begin{aligned}
\mathbb{B}_N &= (\mathbb{B} - \mathbf{c}^{\mathbb{B}}_{\mathbf{1,2}})/c^{\mathbb{B}}_3 && (13)\\
&= (\mathbf{f_p}(\mathbb{A}) - \mathbf{f_p}(\mathbf{c}^{\mathbb{A}}_{\mathbf{1,2}}))/(k\, c^{\mathbb{A}}_3) && (14)\\
&= (k\mathbf{R}_\theta\mathbb{A} + \mathbf{t} - (k\mathbf{R}_\theta\mathbf{c}^{\mathbb{A}}_{\mathbf{1,2}} + \mathbf{t}))/(k\, c^{\mathbb{A}}_3) && (15)\\
&= \mathbf{R}_\theta(\mathbb{A} - \mathbf{c}^{\mathbb{A}}_{\mathbf{1,2}})/c^{\mathbb{A}}_3 && (16)\\
&= \mathbf{R}_\theta\mathbb{A}_N && (17)
\end{aligned}$$

The normalized sets thus differ only by the rotation $\mathbf{R}_\theta$, concluding the proof. □

3.2. Link with the similarity transformation

By substituting the definition of normalized sets from (8) into the rotation space Θ, we can express the relationship directly in terms of the initial sets $\mathbb{A}$ and $\mathbb{B}$. Specifically, the set of feasible rotations Θ is characterized as:

$$\Theta = \left\{ \theta \in \mathbb{R} \;\middle|\; \mathbb{B} = \frac{c^{\mathbb{B}}_3}{c^{\mathbb{A}}_3}\mathbf{R}_\theta\mathbb{A} + \mathbf{c}^{\mathbb{B}}_{\mathbf{1,2}} - \frac{c^{\mathbb{B}}_3}{c^{\mathbb{A}}_3}\mathbf{R}_\theta \cdot \mathbf{c}^{\mathbb{A}}_{\mathbf{1,2}} \right\} \tag{18}$$

By comparing (18) with the general similarity transformation defined in (1) and (2), we can deduce the full set of feasible transformation parameters $\mathbb{P}$. Each rotation $\theta \in \Theta$ uniquely determines a scale factor k and a translation vector $\mathbf{t}$ as follows:

$$\mathbb{P} := \left\{ (\theta, \mathbf{t}, k)^\intercal \;\middle|\; (\theta \in \Theta) \wedge (k = \frac{c_3^{\mathbb{B}}}{c_3^{\mathbb{A}}}) \wedge (\mathbf{t} = \mathbf{c}_{\mathbf{1,2}}^{\mathbb{B}} - k\mathbf{R}_\theta \cdot \mathbf{c}_{\mathbf{1,2}}^{\mathbb{A}}) \right\} \tag{19}$$

3.3. Summary of operations

To summarize the proposed approach, the system of equations presented in (20) formally characterizes the set $\mathbb{P} = \left\{ \mathbf{p} \in \mathbb{S} \mid \mathbf{h}_\mathbf{p}(\mathbb{A}) = \mathbb{B} \right\}$ of all valid transformation parameters. This formulation integrates the fundamental problem variables with necessary intermediate variables, all of which will be estimated simultaneously by the set-membership method.

$$\begin{cases} 1.\ \mathbf{c}^{\mathbb{A}} = \texttt{MinCircle}(\mathbb{A}) \\ 2.\ \mathbf{c}^{\mathbb{B}} = \texttt{MinCircle}(\mathbb{B}) \\ 3.\ \mathbb{A}_N = \left(\mathbb{A} - \mathbf{c}_{\mathbf{1,2}}^{\mathbb{A}} \right) / c_3^{\mathbb{A}} \\ 4.\ \mathbb{B}_N = \left(\mathbb{B} - \mathbf{c}_{\mathbf{1,2}}^{\mathbb{B}} \right) / c_3^{\mathbb{B}} \\ 5.\ \mathbb{B}_N = \mathbf{R}_\theta \cdot \mathbb{A}_N \\ 6.\ k = c_3^{\mathbb{B}} / c_3^{\mathbb{A}} \\ 7.\ \mathbf{t} = \mathbf{c}_{\mathbf{1,2}}^{\mathbb{B}} - k\mathbf{R}_\theta \cdot \mathbf{c}_{\mathbf{1,2}}^{\mathbb{A}} \end{cases} \tag{20}$$

Having outlined the general methodology, the following section introduces the mathematical tools required for its practical implementation. These set-membership techniques provide a robust paradigm that inherently ensures the properties of completeness and reliability in the computed solutions.

4. Set-membership interval tools

In this work, the inputs for the registration problem are defined as bounded sets $\mathbb{A}$ and $\mathbb{B}$ in $\mathbb{R}^2$. Our objective is to estimate the set of feasible similarity transformations, previously defined as $\mathbb{P} = \left\{ \mathbf{p} \in \mathbb{S} \mid \mathbf{f}_\mathbf{p}(\mathbb{A}) = \mathbb{B} \right\}$. As noted in Section 2.7, while $\mathbb{P}$ cannot be computed exactly, a rigorous over-approximation $\mathbb{P}^+ \supset \mathbb{P}$ is computationally tractable.

However, computing $\mathbb{P}^+ \in \mathcal{P}(\mathbb{R}^4)$ using a conventional interval branch-and-bound algorithm is prohibitively expensive. This is due to the exponential complexity inherent in such bisection-based methods when applied to a four-dimensional parameter space. To address this, the Procrustes-like decomposition introduced in Section 3 reformulates the problem into a sequence of set operations, effectively reducing the computational bottleneck to a one-dimensional branch-and-bound task.

This section details the algorithmic framework used to represent these sets and perform the corresponding operations. The fundamental building blocks of this approach relies on interval analysis.

4.1. Interval analysis

An interval $[x] = [x^-, x^+]$ is a closed, connected subset of $\mathbb{R}$. The set of all real intervals is denoted by $\mathbb{IR}$. By extension, an interval vector (or box) $[\mathbf{x}] \in \mathbb{IR}^n$ is defined as the Cartesian product of n intervals, representing an axis-aligned, closed, and connected subset of $\mathbb{R}^n$. These sets are particularly well-suited for computer representation: a box $[\mathbf{x}]$ is uniquely defined by its lower and upper bounds, $\mathbf{x}^-$ and $\mathbf{x}^+$, which are representable vectors in $\mathbb{R}^n$.

Interval analysis provides a reliable arithmetic Moore (1966) based on the inclusion-preserving extension of classical real operators. Given a function $f : \mathbb{R} \to \mathbb{R}$, its interval extension $[f] : \mathbb{IR} \to \mathbb{IR}$ is defined such that it computes an enclosure of the image of $[x]$ under f:

$$[f]([x]) \supseteq \{f(x) \mid x \in [x]\} \tag{21}$$

For many elementary operations, the resulting interval is the narrowest possible (*i.e.*, the interval hull of the image). For example, the subtraction of two intervals is defined as $[x] - [y] = [x^- - y^+, x^+ - y^-]$. For more complex functions, various algorithms and specialized libraries Goualard provide efficient and reliable implementations of these extensions.

Elementary interval functions (such as +, cos, exp, *etc.*) are also equipped with their *backward* (or *reverse* Revol (2017)) counterparts. These allow for the reliable approximation of pre-images given an interval output and a prior enclosure of the domain. For instance, while the forward evaluation $\cos([-\pi/4, \pi/3])$ yields $[0.5, 1]$, the reverse evaluation reverse_cos($[y], [x]$) with $[y] = [0.5, 1]$ and a prior domain $[x] = [-\pi, -\pi/4]$ enables the contraction of $[x]$ to $[-\pi/3, -\pi/4]$.

Interval analysis and reverse functions are fundamental to interval constraint satisfaction problems, as they allow for the bidirectional propagation of information between the inputs and outputs of a function. These tools form the basis for constructing contractors Jaulin, Kieffer, Didrit and Walter (2001).

4.2. Contractors and separators

Given a set $\mathbb{X} \subset \mathbb{R}^n$, a contractor $\mathcal{C}_{\mathbb{X}} : \mathbb{IR}^n \to \mathbb{IR}^n$ is an operator associated with $\mathbb{X}$ and designed to reduce a box $[\mathbf{x}] \in \mathbb{IR}^n$ without losing any vector of $\mathbb{X}$ enclosed in $[\mathbf{x}]$ prior to the contraction. Conceptually, a contractor is a mathematical operator (and in the most complicated cases, an algorithm) that narrows the bounds of an interval domain in a reliable manner, ensuring that no element of $\mathbb{X}$ initially contained in $[\mathbf{x}]$ is lost during the process. The formal properties of a contractor are defined as follows:

Definition 1. *A contractor $\mathcal{C}_{\mathbb{X}}$ associated with a set $\mathbb{X} \subset \mathbb{R}^n$ is a mapping from $\mathbb{IR}^n$ to $\mathbb{IR}^n$ satisfying:*

$$(i)\ \ \forall [\mathbf{x}] \in \mathbb{IR}^n,\ \mathcal{C}_{\mathbb{X}}([\mathbf{x}]) \subseteq [\mathbf{x}] \qquad \textit{(contraction)}$$

$$(ii)\ \ (\mathbf{x} \in \mathbb{X} \cap [\mathbf{x}]) \implies \mathbf{x} \in \mathcal{C}_{\mathbb{X}}([\mathbf{x}]) \qquad \textit{(consistency)}$$

Example 4.1. *Consider a quadratic constraint $ax_1^2 + bx_2^2 + cx_1x_2 + dx_1 + ex_2 + f = 0$ defining the boundary $\partial\mathbb{E}$ of an ellipse $\mathbb{E}$. Robust interval-based algorithms, such as the HC4 Benhamou, Goualard, Granvilliers and Puget (1999), enable the construction of contractors derived from such analytical constraints. Thus, an operator $\mathcal{C}_{\partial\mathbb{E}}$ can be implemented to contract any box $[\mathbf{x}] \in \mathbb{IR}^2$ with respect to the elliptical boundary.*

Figure 5 illustrates various contraction scenarios for $\mathcal{C}_{\partial\mathbb{E}}$. Notably, if a box does not intersect the set $\partial\mathbb{E}$, the contractor may reduce it to the empty set, as shown for the box $[\mathbf{y}]$ where $\mathcal{C}_{\partial\mathbb{E}}([\mathbf{y}]) = \emptyset$.

A contractor is not necessarily minimal; that is, it may satisfy the strict inclusion $\mathcal{C}_{\mathbb{X}}([\mathbf{x}]) \supset [[\mathbf{x}] \cap \mathbb{X}]$, where the brackets $[\cdot]$ denote the interval hull. This potential over-approximation often arises from the dependency problem (multi-occurrence of variables) in mathematical expressions or from the specific performance of the interval extensions employed. Nevertheless, the contraction remains rigorously reliable, as all underlying operations rely on interval arithmetic.

Contractors are typically integrated into a *paver* to characterize (to *reveal*) the underlying set through a recursive bisect-and-contract algorithm. Figure 6 illustrates the results of this paving process for $\mathcal{C}_{\partial\mathbb{E}}$ at varying levels of precision.

The use of contractors enables the encapsulation of complex algorithms into modular "black boxes", which serve solely to contract interval vectors with respect to specific constraints or sets. This reliability, formally expressed by the *consistency* property in Definition 1, is fundamental as it allows for the seamless composition of operators Chabert and Jaulin (2009). Specifically, contractors can be invoked in any sequence or frequency without the risk of discarding valid solutions. This modularity facilitates the resolution of intricate problems, provided that a sufficient suite of contractors is available to address the elementary constraints derived from the problem's decomposition. Ultimately, these contractors can be integrated into a paver to generate a rigorous outer-approximation of the solution set.

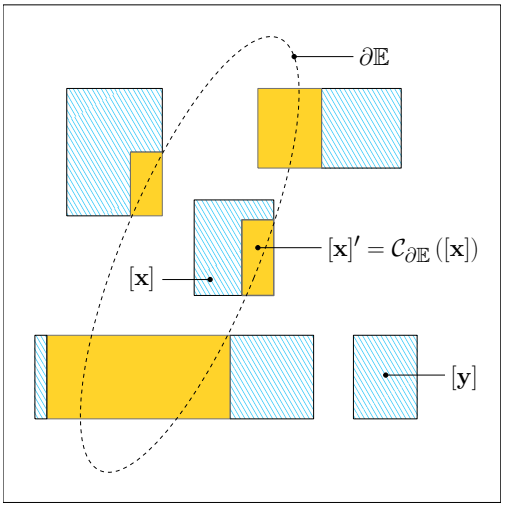


Figure 5: Illustration of contractions associated with the set $\partial\mathbb{E}$ (dashed line). The blue regions represent the portions of the boxes removed by the operator, while the remaining boxes satisfy the consistency property relative to the ellipse boundary.

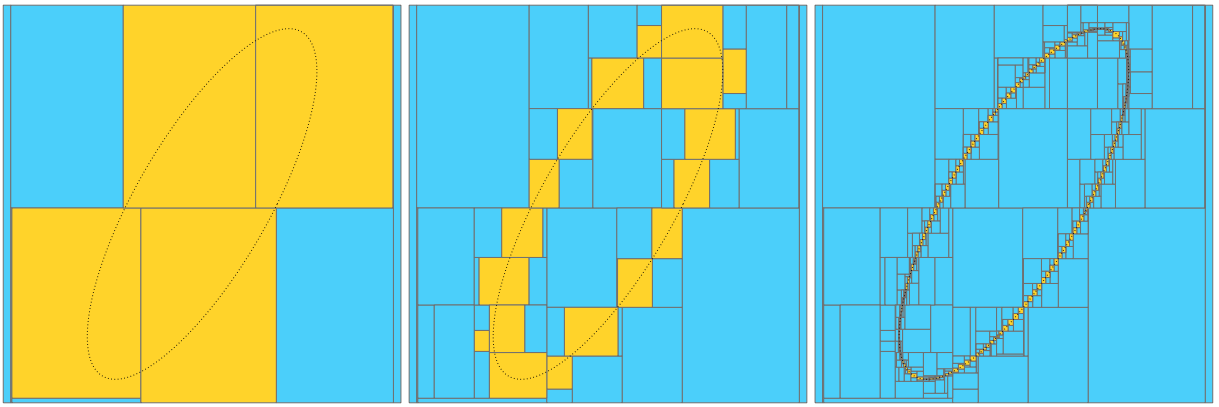


Figure 6: Paving results using the contractor $\mathcal{C}_{\partial\mathbb{E}}$ for various precision thresholds $\epsilon \in \{4, 1, 0.1\}$, where ϵ denotes the maximum diameter of the boxes. Blue boxes represent regions guaranteed to contain no solutions, while yellow boxes indicate regions that may contain vectors satisfying the constraint.

When both inner and outer approximations are required, contractors alone are insufficient. Recent advancements have addressed this by introducing *separators* Jaulin and Desrochers (2014). A separator $\mathcal{S}_{\mathbb{X}}$ is an operator from $\mathbb{IR}^n$ to $\mathbb{IR}^n \times \mathbb{IR}^n$ designed to partition ("separate") a box into two sub-boxes representing the outer and inner approximations of a set $\mathbb{X}$, respectively. In practice, a separator is typically implemented as a pair of complementary contractors: one for the set itself ($\mathcal{C}_{\mathbb{X}}$) and one for its complement ($\mathcal{C}_{\overline{\mathbb{X}}}$). This dual approach allows for a rigorous classification of the search space into regions that are guaranteed to be inside, outside, or on the boundary of the target set. A conceptual illustration of this principle is provided in Figure 7.

4.3. Contractor Programming

Following the framework introduced in Chabert and Jaulin (2009), a contractor can be formally identified with a set. Let $\text{set}(\mathcal{L})$ denote the solution set associated with a constraint $\mathcal{L}$: $\text{set}(\mathcal{L}) = \{\mathbf{x} \in \mathbb{R}^n \mid \mathcal{L}(\mathbf{x}) \text{ is true}\}$. Similarly, we define the set associated with a contractor $\mathcal{C}$ as: $\text{set}(\mathcal{C}) = \{\mathbf{x} \in \mathbb{R}^n \mid \mathcal{C}(\{\mathbf{x}\}) = \{\mathbf{x}\}\}$, where $\{\mathbf{x}\}$ represents a singleton (or degenerate box) that remains invariant under the operator.

According to Definition 1, if $\mathcal{C}$ is a contractor for $\mathcal{L}$, then $\text{set}(\mathcal{L}) \subseteq \text{set}(\mathcal{C})$. In the case of a minimal contractor, this inclusion becomes an equality: $\text{set}(\mathcal{L}) = \text{set}(\mathcal{C})$. This equivalence allows us to treat a contractor as a functional

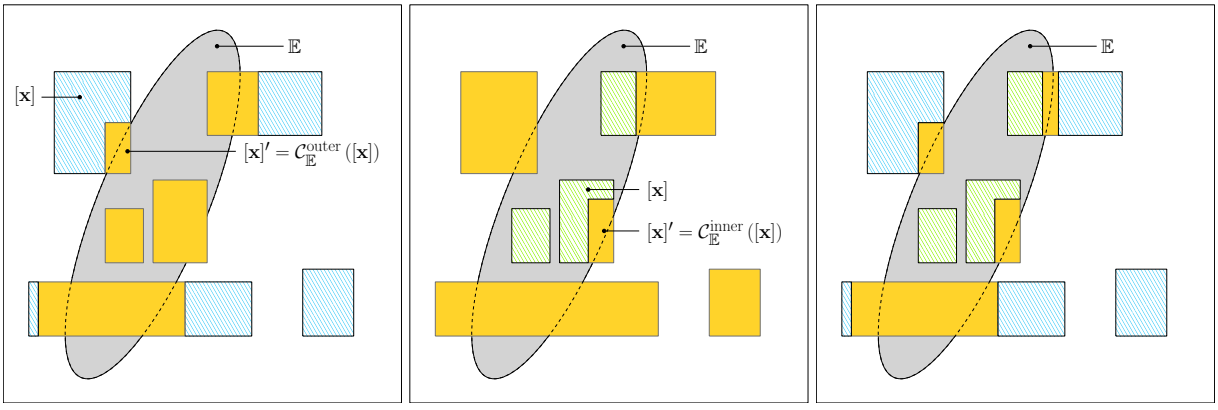


Figure 7: Principle of a separator $\mathcal{S}_{\mathbb{E}} = \{\mathcal{C}_{\mathbb{E}}^{\text{out}}, \mathcal{C}_{\mathbb{E}}^{\text{in}}\}$ associated with a set $\mathbb{E}$. Left: outer contractions using $\mathcal{C}_{\mathbb{E}}^{\text{out}}$, which prunes the space outside $\mathbb{E}$. Center: inner contractions using $\mathcal{C}_{\mathbb{E}}^{\text{in}}$, which prunes the space inside $\mathbb{E}$ to reveal the complement. Right: the combined effect of the separator, illustrating the resulting partition of the boxes.

representation of its underlying constraint: $\text{set}(\mathcal{L}) \sim \mathcal{C}$. Consequently, logical operations on constraints can be directly implemented using set operations on their corresponding contractors. For instance, the conjunction $\mathcal{L}_3 = \mathcal{L}_1 \wedge \mathcal{L}_2$ corresponds to the intersection of their sets, and thus to the composition of their contractors:

$$\text{set}(\mathcal{L}_3) = \text{set}(\mathcal{L}_1 \wedge \mathcal{L}_2) = \text{set}(\mathcal{L}_1) \cap \text{set}(\mathcal{L}_2) \sim \mathcal{C}_1 \cap \mathcal{C}_2 \tag{22}$$

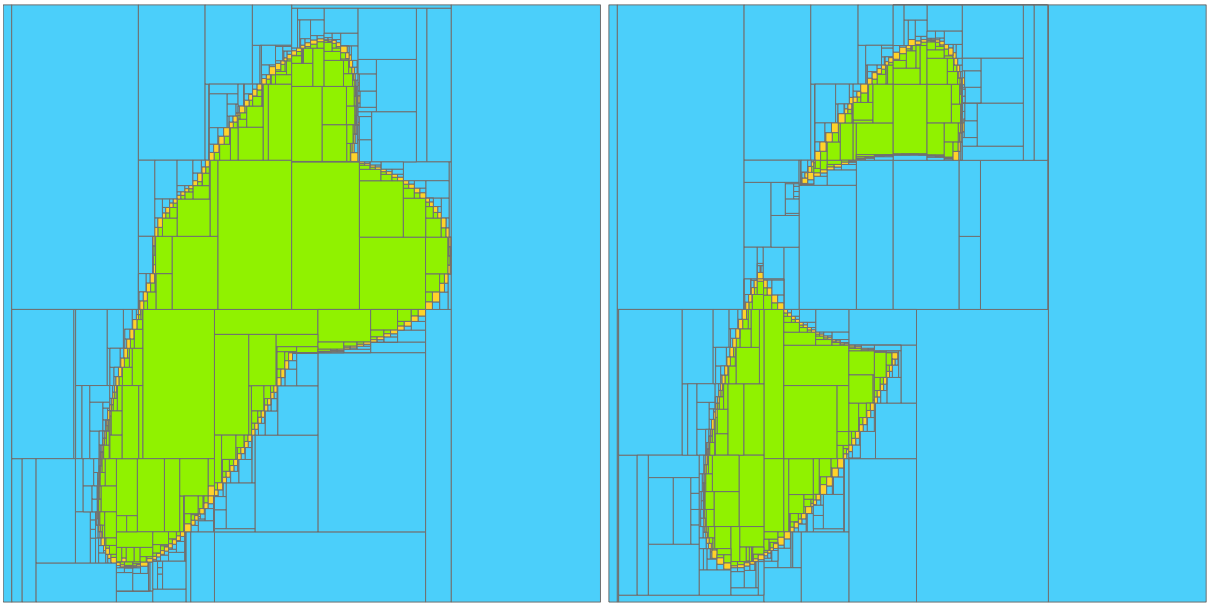
This modeling paradigm called *contractor programming* in Chabert and Jaulin (2009), enables the manipulation of contractors through standard set-theoretic operations, including intersection, union, projection, and Cartesian products. This algebraic approach extends naturally to separators Jaulin and Desrochers (2014) and so to *separator programming*. Figure 8 illustrates how the combination of elementary separators allows for the representation of complex geometric sets with guaranteed inner and outer approximations.

For practical implementation, the Codac library Rohou, Desrochers and Le Bars (2024) provides a comprehensive catalog of pre-defined contractors and separators, facilitating the characterization of various constraints. An example of code using this library is provided in A.

4.4. Reformulation of the problem in separator programming

By leveraging this algebraic paradigm, the system of equations in (23) provides a direct translation of the geometric problem into the space of separators. The left side recalls the set-based operations from (20), while the right side defines their corresponding functional implementation based on separators.

$$\left\{\begin{array}{l} 1.\ \mathbf{c}^{\mathbb{A}} = \texttt{MinCircle}(\mathbb{A}) \\ 2.\ \mathbf{c}^{\mathbb{B}} = \texttt{MinCircle}(\mathbb{B}) \\ 3.\ \mathbb{A}_N = \left(\mathbb{A} - \mathbf{c}_{\mathbf{12}}^{\mathbb{A}}\right)/c_3^{\mathbb{A}} \\ 4.\ \mathbb{B}_N = \left(\mathbb{B} - \mathbf{c}_{\mathbf{12}}^{\mathbb{B}}\right)/c_3^{\mathbb{B}} \\ 5.\ \mathbb{B}_N = \mathbf{R}_\theta \cdot \mathbb{A}_N \\ 6.\ k = c_3^{\mathbb{B}}/c_3^{\mathbb{A}} \\ 7.\ \mathbf{t} = \mathbf{c}_{\mathbf{12}}^{\mathbb{B}} - k\mathbf{R}_\theta \cdot \mathbf{c}_{\mathbf{12}}^{\mathbb{A}} \end{array}\right. \sim \left\{\begin{array}{l} 1.\ \mathcal{S}_{\{\mathbf{c}^{\mathbb{A}}\}} = \texttt{SepMinCircle}(\mathcal{S}_{\mathbb{A}}) \\ 2.\ \mathcal{S}_{\{\mathbf{c}^{\mathbb{B}}\}} = \texttt{SepMinCircle}(\mathcal{S}_{\mathbb{B}}) \\ 3.\ \mathcal{S}_{\mathbb{A}_N} = \left(\mathcal{S}_{\mathbb{A}} - \mathcal{S}_{\{\mathbf{c}^{\mathbb{A}}\}_{12}}\right)/\mathcal{S}_{\{\mathbf{c}^{\mathbb{A}}\}_3} \\ 4.\ \mathcal{S}_{\mathbb{B}_N} = \left(\mathcal{S}_{\mathbb{B}} - \mathcal{S}_{\{\mathbf{c}^{\mathbb{B}}\}_{12}}\right)/\mathcal{S}_{\{\mathbf{c}^{\mathbb{B}}\}_3} \\ 5.\ \mathcal{S}_{\mathbb{B}_N} = \mathbf{R}_{\mathcal{S}_\Theta} \cdot \mathcal{S}_{\mathbb{A}_N} \\ 6.\ \mathcal{S}_{\mathbb{K}} = \mathcal{S}_{\{\mathbf{c}^{\mathbb{B}}\}_3}/\mathcal{S}_{\{\mathbf{c}^{\mathbb{A}}\}_3} \\ 7.\ \mathcal{S}_{\mathbb{T}} = \mathcal{S}_{\{\mathbf{c}^{\mathbb{B}}\}_{12}} - k\mathbf{R}_{\mathcal{S}_\Theta} \cdot \mathcal{S}_{\{\mathbf{c}^{\mathbb{A}}\}_{12}} \end{array}\right. \tag{23}$$



(a) Paving approximation of the disjunction $\mathcal{E}_1 \vee \mathcal{E}_2$ implemented via the union of separators ($\mathcal{S}_{\mathbb{E}_1} \cup \mathcal{S}_{\mathbb{E}_2}$).

(b) Paving approximation of the complementary intersection $\mathcal{E}_1 \wedge \neg\mathcal{E}_2$ implemented via the separator ($\mathcal{S}_{\mathbb{E}_1} \cap \overline{\mathcal{S}_{\mathbb{E}_2}}$).

Figure 8: Algebraic combination of two separators $\mathcal{S}_{\mathbb{E}_1}$ and $\mathcal{S}_{\mathbb{E}_2}$ associated with ellipsoidal constraints $\mathcal{E}_1$ and $\mathcal{E}_2$, defined by analytical expressions of the form $f(\mathbf{x}) < 0$. The resulting composite separators are processed by a paver to characterize the solution sets with guaranteed inner (in green) and outer (in blue) approximations.

The operators involved in this formulation, specifically the `SepMinCircle` separator, the rotational mapping (relation 5.), and the algebraic arithmetic (relations 3., 4., 6. and 7.) between separators, are detailed in Section 5. Their implementation relies on specific separators described hereafter.

4.5. Separators for parameterized functions

Let $\mathbf{h} : \mathbb{R}^l \times \mathbb{R}^m \to \mathbb{R}^n$ be a function and $\mathbb{X} \subset \mathbb{R}^l$ and $\mathbb{Z} \subset \mathbb{R}^n$ be two given sets. Desrochers (2018) has proposed a way to infer a separator determining the parameter set $\mathbb{Y} \subset \mathbb{R}^m$ of parameters that map $\mathbb{X}$ and $\mathbb{Z}$ onto each other.

Proposition 4.1. *Let us consider the parameter set $\mathbb{Y}$ defined by:*

$$\mathbb{Y} = \{\mathbf{y} \in \mathbb{R}^m \mid \mathbf{h}(\mathbb{X}, \mathbf{y}) \subset \mathbb{Z}\} \tag{24}$$

Also consider $proj_{[l+1,l+m]}(\mathbb{X})$ denoted as the projection of $\mathbb{X}$ onto its components indexed by $i \in \{l+1, \dots, l+m\}$. Then, the separator

$$\mathcal{S}_{\mathbb{Y}} = \overline{proj_{[l+1,l+m]}\left\{(\mathcal{S}_{\mathbb{X}} \times \mathcal{S}_{\mathbb{R}^m}) \cap \mathbf{h}^{-1}(\overline{\mathcal{S}_{\mathbb{Z}}})\right\}}, \tag{25}$$

is a separator for $\mathbb{Y}$.

Proof. Based on (24), we reformulate $\mathbb{Y}$:

$$\begin{aligned}
\mathbb{Y} &= \{\mathbf{y} \in \mathbb{R}^m \mid \mathbf{h}(\mathbb{X}, \mathbf{y}) \subset \mathbb{Z}\} && (26)\\
&= \{\mathbf{y} \in \mathbb{R}^m \mid \forall \mathbf{x} \in \mathbb{X},\ \mathbf{h}(\mathbf{x}, \mathbf{y}) \in \mathbb{Z}\} && (27)\\
&= \overline{\left\{\mathbf{y} \in \mathbb{R}^m \mid \exists \mathbf{x} \in \mathbb{X},\ \mathbf{h}(\mathbf{x}, \mathbf{y}) \in \overline{\mathbb{Z}}\right\}} && (28)\\
&= \overline{\left\{\mathbf{y} \in \mathbb{R}^m \mid \exists \mathbf{x} \in \mathbb{R}^l,\ (\mathbf{x}, \mathbf{y}) \in (\mathbb{X} \times \mathbb{R}^m) \cap \mathbf{h}^{-1}(\overline{\mathbb{Z}})\right\}} && (29)
\end{aligned}$$

$$= \overline{\text{proj}_{[l+1,l+m]}\left\{(\mathbb{X}\times\mathbb{R}^m)\cap\mathbf{h^{-1}}(\overline{\mathbb{Z}})\right\}} \tag{30}$$

$$\tag{31}$$

Using separator algebra, this proves that $\mathcal{S}_{\mathbb{Y}}$ is a separator for $\mathbb{Y}$. □

Finally, Figure 9 illustrates the expressiveness of the separator programming paradigm. By leveraging the Codac library, the formal definition in (25) is translated into a concise and readable implementation, bridging the gap between set-theoretic theory and practical computation.

$$\begin{cases} \mathcal{S}_1 = \mathcal{S}_{\text{inv}}(\mathbf{h}, \overline{\mathcal{S}_{\mathbb{Z}}}) \\ \mathcal{S}_2 = (\mathcal{S}_{\mathbb{X}} \times \mathcal{S}_{\mathbb{R}^m}) \cap \mathcal{S}_1 \\ \mathcal{S}_3 = \mathcal{S}_{\text{proj}}(\mathcal{S}_2, [l+1, l+m]) \\ \mathcal{S}_{\mathbb{Y}} = \overline{\mathcal{S}_3} \end{cases} \rightarrow$$

```
k = VectorVar(l+m)
f_proj = AnalyticFunction([k], [k[l]])

S1 = SepInverse(h, ~Sz)
S2 = SepInverse(f_proj, Sx) & S1
S3 = SepProj(S2, [l+1,...,l+m])
SY = ~S3
```

Figure 9: Left: Decomposition of the separator $\mathcal{S}_{\mathbb{Y}}$ as defined in (25). Right: Equivalent implementation using the Codac library. Without detailing this code, we can appreciate that the syntax closely mirrors the mathematical operations, where operators `SepInverse` and `SepProj` are elementary separators in the Codac library.

4.6. Separator for Cartesian-Polar coordinate transformation

The transformation between Cartesian coordinates (x, y) and polar coordinates (ρ, θ) is a classical challenge for interval arithmetic. Due to the non-monotonicity and periodicity of the trigonometric functions involved, standard methods often yield significant over-approximations. Set-membership operators, however, provide a rigorous way to handle these transformations. The coordinate change is governed by the following relations:

$$\rho = \sqrt{x^2 + y^2} \qquad x = \rho\cos(\theta)$$
$$\theta = \text{atan2}(y, x) \qquad y = \rho\sin(\theta)$$

In the literature, several operators have been developed to manage these constraints efficiently:

- $\mathcal{C}_{\text{Polar}}$ Desrochers and Jaulin (2016): a minimal contractor that maintains consistency between the polar and Cartesian representations of a point in $\mathbb{R}^2$.
- $\mathcal{S}_{\text{Polar}}$ Desrochers (2018): the separator counterpart of $\mathcal{C}_{\text{Polar}}$, enabling the characterization of both the interior and exterior of polar-defined sets.
- $\mathcal{S}_{\text{PolarXY}}([\rho], [\theta])$: a separator in Codac Rohou et al. (2024) parameterized by a polar box $[\rho]\times[\theta]$. This operator defines a set in the Cartesian plane consistent with $[\rho]\times[\theta]$, allowing for its paving using Cartesian boxes.

In what follows, we introduce the separator `SepCartPol`, which enables the conversion of the representation of a set representations from a Cartesian coordinate system to a polar one. This contribution is implemented in Algorithm 1. The operators `CartHull` and `PolarHull` compute the tightest Cartesian (resp. polar) bounding box for a given polar (resp. Cartesian) input box. These hulls are efficiently implemented using the contractor $\mathcal{C}_{\text{Polar}}$, ensuring that the resulting enclosures are as small as possible while remaining guaranteed.

5. Separators in our registration approach

Since the solution to the registration system (20) is intrinsically a set, the algebra of separators provides a natural framework for both its formal definition and its computational characterization. This section details each separator required for the implementation of our method, following the sequence of operations introduced in the previous sections.

Algorithm 1: SepCartPolar separator

Data: $\mathcal{S}_0$
Input: $[\mathbf{u}^{\mathbf{P}}]$
Output: $[\mathbf{u}_{\mathbf{i}}^{\mathbf{P}}], [\mathbf{u}_{\mathbf{o}}^{\mathbf{P}}]$
$[\mathbf{u}^{\mathbf{C}}] \leftarrow \texttt{CartHull}([\mathbf{u}^{\mathbf{P}}])$
$([\mathbf{u}_{\mathbf{i}}^{\mathbf{C}}], [\mathbf{u}_{\mathbf{o}}^{\mathbf{C}}]) \leftarrow \mathcal{S}_0([\mathbf{u}^{\mathbf{C}}])$
$[\mathbf{u}_{\mathbf{i}}^{\mathbf{P}}] \leftarrow [\mathbf{u}^{\mathbf{P}}] \cap \texttt{PolarHull}([\mathbf{u}_{\mathbf{i}}^{\mathbf{C}}])$
$[\mathbf{u}_{\mathbf{o}}^{\mathbf{P}}] \leftarrow [\mathbf{u}^{\mathbf{P}}] \cap \texttt{PolarHull}([\mathbf{u}_{\mathbf{o}}^{\mathbf{C}}])$
return $[\mathbf{u}_{\mathbf{i}}^{\mathbf{P}}], [\mathbf{u}_{\mathbf{o}}^{\mathbf{P}}]$

5.1. Minimum bounding circle separator

As illustrated in Figure 4 (p.7), the first stage of our approach is achieved through these two equations:

$$\begin{cases} \mathbf{c}^{\mathbb{A}} = \texttt{MinCircle}(\mathbb{A}) \\ \mathbf{c}^{\mathbb{B}} = \texttt{MinCircle}(\mathbb{B}) \end{cases} \tag{32}$$

Let $\mathbb{C}^{\mathbb{A}} \subset \mathbb{R}^3$ denote the set of all circles enclosing the set $\mathbb{A}$. The minimum bounding circle $\mathbf{c}^{\mathbb{A}}$ is then obtained by solving a constrained minimization problem. We define the optimal subset $\mathbb{C}^{\mathbb{A}\star} \subset \mathbb{C}^{\mathbb{A}}$ as the set of enclosing circles with minimal radius:

$$\mathbb{C}^{\mathbb{A}\star} = \left\{ \mathbf{c} \in \mathbb{C}^{\mathbb{A}} \mid c_3 = \min_{\mathbf{c}' \in \mathbb{C}^{\mathbb{A}}} c_3' \right\}. \tag{33}$$

This set characterizes the smallest enclosing circle of $\mathbb{A}$. Due to the strict convexity of the problem, this minimum is unique, and the set $\mathbb{C}^{\mathbb{A}\star}$ is therefore a singleton:

$$\mathbb{C}^{\mathbb{A}\star} = \left\{ \mathbf{c}^{\mathbb{A}} \right\}. \tag{34}$$

To compute $\mathbb{C}^{\mathbb{A}}$, we first define the algebraic distance function h_0:

$$\begin{aligned} h_0 : \mathbb{R}^2 \times \mathbb{R}^3 &\longrightarrow \mathbb{R} \\ (\mathbf{u}, \mathbf{c}) &\longmapsto (u_1 - c_1)^2 + (u_2 - c_2)^2 - c_3^2 \end{aligned} \tag{35}$$

where $\mathbf{u} = (u_1, u_2)^\top$ represents the Cartesian coordinates of a point in $\mathbb{R}^2$, and $\mathbf{c} = (c_1, c_2, c_3)^\top$ denotes the parameters of the circle, specifically its center (c_1, c_2) and its radius c_3. The condition $h_0(\mathbf{u}, \mathbf{c}) \leq 0$ implies that the point $\mathbf{u}$ is contained within the disk defined by $\mathbf{c}$ or lies on its boundary.

The set of all circles enclosing $\mathbb{A}$ can thus be defined as:

$$\mathbb{C}^{\mathbb{A}} \triangleq \left\{ \mathbf{c} \in \mathbb{R}^3 \mid h_0(\mathbb{A}, \mathbf{c}) \subset (-\infty, 0] \right\}. \tag{36}$$

Applying Proposition 4.1, we can directly derive a separator $\mathcal{S}_{\mathbb{C}^{\mathbb{A}}}$ for this set.

The projection of $\mathbb{C}^{\mathbb{A}}$ onto its third component (the radius c_3) yields the interval $[c_3^{\mathbb{A}}, +\infty]$, as any circle concentric with the minimum bounding circle but with a larger radius also satisfies the enclosure constraint. To isolate the unique minimal radius $c_3^{\mathbb{A}}$, we observe that it resides exactly at the lower boundary of this set. In the framework of set-membership and closed sets, $\{c_3^{\mathbb{A}}\}$ can be formally characterized as the intersection of the projection and the closure of its complement:

$$\{c_3^{\mathbb{A}}\} = \text{proj}_3(\mathbb{C}^{\mathbb{A}}) \cap \overline{\text{proj}_3(\mathbb{C}^{\mathbb{A}})}. \tag{37}$$

This topological definition naturally induces a separator $\mathcal{S}_{\{c_3^{\mathbb{A}}\}}$ for the MBC radius through separator programming. Once this minimal radius is identified, the MBC itself is obtained by restricting the feasible set $\mathbb{C}^{\mathbb{A}}$ to the specific slice where the radius equals $c_3^{\mathbb{A}}$:

$$\left\{\mathbf{c}^{\mathbb{A}}\right\} = \mathbb{C}^{\mathbb{A}} \cap \left(\mathbb{R}^2 \times \{c_3^{\mathbb{A}}\}\right). \tag{38}$$

The separator $\mathcal{S}_{\{\mathbf{c}^{\mathbb{A}}\}}$ for the complete MBC follows directly from the intersection of the previously defined separators $\mathcal{S}_{\mathbb{C}^{\mathbb{A}}}$ and $\mathcal{S}_{\{c_3^{\mathbb{A}}\}}$. A practical implementation of this process is detailed in Algorithm 3 (Section 5.5).

5.2. Normalized set separator

The normalization of the datasets, corresponding to Step 2 of our method (see Figure 4, p.7), is an important stage. It ensures that the point clouds are centered and scaled based on their respective minimum bounding circles. The normalized sets $\mathbb{A}_N$ and $\mathbb{B}_N$ are defined as follows:

$$\begin{cases} 3.\ \mathbb{A}_N = \left(\mathbb{A} - \mathbf{c}_{\mathbf{12}}^{\mathbb{A}}\right)/c_3^{\mathbb{A}}, \\ 4.\ \mathbb{B}_N = \left(\mathbb{B} - \mathbf{c}_{\mathbf{12}}^{\mathbb{B}}\right)/c_3^{\mathbb{B}} \end{cases} \tag{39}$$

To derive a separator for these sets, consider the transformation function $\mathbf{h}_1$:

$$\begin{aligned} \mathbf{h}_1 \,:\, & \mathbb{R}^3 \times \mathbb{R}^2 \longrightarrow \mathbb{R}^2 \\ & (\mathbf{c}, \mathbf{a}_n) \longmapsto \mathbf{a}_n \cdot c_3 + \mathbf{c}_{1,2} \end{aligned} \tag{40}$$

where $\mathbf{a}_n$ is a point in the normalized space. This allows us to redefine $\mathbb{A}_N$ as the set of points that, when mapped back by $\mathbf{h}_1$ using the parameters of the minimum bounding circle $\mathbf{c}^{\mathbb{A}}$, fall within the original set $\mathbb{A}$:

$$\mathbb{A}_N = \left\{\mathbf{a}_n \in \mathbb{R}^2 \mid \mathbf{h}_1(\mathbf{c}^{\mathbb{A}}, \mathbf{a}_n) \in \mathbb{A}\right\}. \tag{41}$$

Since $\{\mathbf{c}^{\mathbb{A}}\}$ is a singleton and the mapping $\mathbf{a}_n \mapsto \mathbf{h}_1(\mathbf{c}^{\mathbb{A}}, \mathbf{a}_n)$ is a bijection, we can characterize $\mathbb{A}_N$ and its complement $\overline{\mathbb{A}_N}$ using the inclusion properties:

$$\mathbb{A}_N = \left\{\mathbf{a_n} \in \mathbb{R}^2 \mid \mathbf{h_1}\left(\left\{\mathbf{c}^{\mathbb{A}}\right\}, \mathbf{a_n}\right) \subset \mathbb{A}\right\} \tag{42}$$

$$\overline{\mathbb{A}_N} = \left\{\mathbf{a_n} \in \mathbb{R}^2 \mid \mathbf{h_1}\left(\left\{\mathbf{c}^{\mathbb{A}}\right\}, \mathbf{a_n}\right) \subset \overline{\mathbb{A}}\right\} \tag{43}$$

Applying Proposition 4.1, we construct two separators for $\mathbb{A}_N$:

$$\mathcal{S}^i_{\mathbb{A}_N} = \overline{\mathrm{proj}_{\mathbf{45}}\left\{(\mathcal{S}_{\{\mathbf{c}^{\mathbb{A}}\}} \times \mathcal{S}_{\mathbb{R}^2}) \cap \mathbf{h_1^{-1}}(\overline{\mathcal{S}_{\mathbb{A}}})\right\}} \tag{44}$$

$$\mathcal{S}^o_{\mathbb{A}_N} = \mathrm{proj}_{\mathbf{45}}\left\{(\mathcal{S}_{\{\mathbf{c}^{\mathbb{A}}\}} \times \mathcal{S}_{\mathbb{R}^2}) \cap \mathbf{h_1^{-1}}(\mathcal{S}_{\mathbb{A}})\right\} \tag{45}$$

Remark 1. *The simultaneous use of the separator pair $(\mathcal{S}^i_{\mathbb{A}_N}, \mathcal{S}^o_{\mathbb{A}_N})$ is fundamental for the convergence of the paving algorithm. Since $\{\mathbf{c}^{\mathbb{A}}\}$ is a singleton, its associated separator $\mathcal{S}_{\{\mathbf{c}^{\mathbb{A}}\}}$ possesses no interior; it can only prune regions strictly outside the point. Consequently, as derived from (44), $\mathcal{S}^i_{\mathbb{A}_N}$ is only capable of characterizing the inner approximation of $\mathbb{A}_N$. Conversely, following (45), $\mathcal{S}^o_{\mathbb{A}_N}$ serves to prune infeasible regions, thereby characterizing the outer approximation (the complement $\overline{\mathbb{A}_N}$).*

In practical terms, the separation process utilizes the inner contractor of $\mathcal{S}^i_{\mathbb{A}_N}$ and the outer contractor of $\mathcal{S}^o_{\mathbb{A}_N}$ in complementary roles. Therefore, the branch-and-contract paving algorithm must integrate both separators to rigorously characterize the normalized sets $\mathbb{A}_N$ and $\mathbb{B}_N$.

5.3. Rotational mapping separator

The rotational mapping, corresponding to Step 3 of our approach (see Figure 4, p.7), is derived from the fifth relation of system (20):

$$5.\ \mathbb{B}_N = \mathbf{R}_\theta \cdot \mathbb{A}_N \tag{46}$$

Let Θ be the set of all rotation angles θ satisfying this relation:

$$\Theta := \left\{\theta \in \mathbb{R}/(2\pi\mathbb{Z}) \mid \mathbb{B}_N = \mathbf{R}_\theta \cdot \mathbb{A}_N\right\} \tag{9}$$

To characterize Θ efficiently, we transform the Cartesian coordinates into the polar domain (ρ, ϕ). A rotation $\mathbf{R}_\theta$ acting on a polar vector is expressed as a simple shift in the angular component:

$$\mathbf{R}_\theta \cdot \begin{pmatrix}\rho\\ \phi\end{pmatrix} = \begin{pmatrix}\rho\\ \phi+\theta\end{pmatrix}. \tag{47}$$

The primary advantage of this polar representation is that it linearizes the rotation into a Minkowski sum (or a simple addition of intervals), which is significantly easier to handle with contractors than the trigonometric couplings of the Cartesian rotation matrix.

A new specialized separator, described in the appendix, is employed to perform the transformation between Cartesian and polar coordinates. Let $\mathbb{A}_N^P$ and $\mathbb{B}_N^P$ denote the polar representations of the normalized sets $\mathbb{A}_N$ and $\mathbb{B}_N$, respectively. The rotation constraint can then be strictly reformulated as a translation in the polar plane:

$$(\mathbb{B}_N = \mathbf{R}_\theta \cdot \mathbb{A}_N) \iff (\mathbb{A}_N^P + (0,\theta)^\intercal = \mathbb{B}_N^P). \tag{48}$$

This shift property allows us to handle the sets $\mathbb{A}_N^P$ and $\mathbb{B}_N^P$ in polar coordinates.

Let us define the functions $\mathbf{h}_2^+$ and $\mathbf{h}_2^-$:

$$\mathbf{h}_2^+(\mathbf{u},\theta) = \mathbf{u} + (0,\theta)^\intercal, \tag{49}$$

$$\mathbf{h}_2^-(\mathbf{u},\theta) = \mathbf{u} - (0,\theta)^\intercal, \tag{50}$$

where $\mathbf{u} \in \mathbb{R}^2$ is a point in the polar plane. While $\mathbf{h}_2^+$ maps the normalized set $\mathbb{A}_N^P$ toward $\mathbb{B}_N^P$ via a positive angular shift, $\mathbf{h}_2^-$ provides the inverse mapping.

The set Θ can then be formally characterized by the equivalence of the polar sets under the angular shift. We decompose the set equality into two reciprocal inclusions:

$$\Theta = \left\{\theta \in \mathbb{R}/(2\pi\mathbb{Z}) \mid \mathbf{h}_2^+(\mathbb{A}_N^P,\theta) = \mathbb{B}_N^P\right\} \tag{51}$$

$$= \left\{\theta \in \mathbb{R}/(2\pi\mathbb{Z}) \mid \left(\mathbf{h}_2^+(\mathbb{A}_N^P,\theta) \subseteq \mathbb{B}_N^P\right) \wedge \left(\mathbf{h}_2^-(\mathbb{B}_N^P,\theta) \subseteq \mathbb{A}_N^P\right)\right\}. \tag{52}$$

By applying Proposition 4.1 to each inclusion, we derive a global separator $\mathcal{S}_\Theta$. This separator, implemented in lines 17 and 18 of Algorithm 3, is defined by the intersection of two specialized separators:

$$\mathcal{S}_\Theta = \overline{\mathrm{proj}_3\left\{(\mathcal{S}_{\mathbb{A}_N^P} \times \mathcal{S}_\mathbb{R}) \cap \mathbf{h}_2^{+-1}(\overline{\mathcal{S}_{\mathbb{B}_N^P}})\right\}} \cap \overline{\mathrm{proj}_3\left\{(\mathcal{S}_{\mathbb{B}_N^P} \times \mathcal{S}_\mathbb{R}) \cap \mathbf{h}_2^{--1}(\overline{\mathcal{S}_{\mathbb{A}_N^P}})\right\}} \tag{53}$$

The first term of the intersection prunes rotations that fail to map $\mathbb{A}_N^P$ into $\mathbb{B}_N^P$, while the second term prunes those that do not satisfy the reciprocal mapping, effectively narrowing Θ down to its consistent solutions.

5.4. Separators for standard algebraic constraints

The final parameters of the registration, namely the scaling factor k and the translation vector $\mathbf{t}$, are determined by equations 6. and 7. of system (20):

$$\begin{cases} 6.\ k = c_3^{\mathbb{B}} / c_3^{\mathbb{A}} \\ 7.\ \mathbf{t} = \mathbf{c}_{\mathbf{12}}^{\mathbb{B}} - k\mathbf{R}_\theta \cdot \mathbf{c}_{\mathbf{12}}^{\mathbb{A}} \end{cases} \tag{54}$$

These equations consist of standard algebraic operations and trigonometric functions. In our implementation, the corresponding separators are derived from the natural interval extensions of these functions. These operators can be constructed using off-the-shelf libraries that provide interval-analysis algorithms, such as `Codac` Rohou et al. (2024) and `IBEX` Chabert (2012). In this work, we rely on `Codac` to build these operators.

5.5. Complete Resolution Algorithm

The global resolution strategy is summarized in Algorithm 2. This sequence follows the conceptual flow illustrated in Figure 4 and directly implements the set-membership requirements of system (20).

A noteworthy feature of this implementation is its modularity: each high-level operator (e.g., `SepMinCircle`, `SepRotation`) encapsulates a complex sub-routine of constraint propagation and box contractions. As mentioned in lines 6 and 7, the separators $\mathcal{S}_{c_3}$ and $\mathcal{S}_{\mathbf{c}_{1,2}}$ are extracted from the minimum bounding circle separator via projections. While not detailed here for the sake of brevity, these projections maintain the rigorous enclosure of the parameters throughout the process.

Algorithm 2: MinCircleRegistration

Data: Separators $\mathcal{S}_{\mathbb{A}}$ and $\mathcal{S}_{\mathbb{B}}$ representing input datasets.
Result: Separator $\mathcal{S}_{\mathbb{P}}$ for the transformation parameters $(\theta, \mathbf{t}, k)$.

```
// 1. MBC of 𝔸 and 𝔹
1 S_c𝔸 ← SepMinCircle(S_𝔸)
2 S_c𝔹 ← SepMinCircle(S_𝔹)
// 2. Normalization
3 S_𝔸N ← SepNormalized(S_𝔸, S_c𝔸)
4 S_𝔹N ← SepNormalized(S_𝔹, S_c𝔹)
// 3. Parameter Identification
5 S_Θ ← SepRotation(S_𝔸N, S_𝔹N)
6 S_𝕂 ← SepScaling(S_c3𝔸, S_c3𝔹)
7 S_𝕋 ← SepTranslation(S_c12𝔸, S_c12𝔹, S_𝕂, S_Θ)
// 4. Composition of the final Parameter Separator
8 S_ℙ ← S_Θ × S_𝕋 × S_𝕂
9 return S_ℙ
```

Algorithm 3 formalizes each individual separator derived throughout this section. It serves as the low-level library of operators called by the main resolution routine.

Notably, the $\mathcal{S}_{\text{pair}}$ operator introduced in line 11 facilitates the construction of a hybrid separator: it utilizes the first argument to perform inner contractions and the second for outer contractions. This implementation directly fulfills the requirements discussed in Remark 1 for handling sets derived from singletons. Finally, the `SepCartPol` operator, which handles the transition to the polar domain, is detailed separately in Algorithm 1 (Section 4.6).

Algorithm 3: Separators definition

Function `SepMinCircle`$(\mathcal{S}_{\mathbb{X}})$**:**
  $h_0 \leftarrow (\mathbf{x}, \mathbf{c}) \mapsto (x_1 - c_1)^2 + (x_2 - c_2)^2 - c_3^2$
  $\mathcal{S}_1 \leftarrow \neg\text{proj}_{\mathbf{345}}\left\{(\mathcal{S}_{\mathbb{X}} \times \mathcal{S}_{\mathbb{R}^3}) \cap \mathbf{h_0^{-1}}([0, \infty])\right\}$
  $\mathcal{S}_2 \leftarrow \text{proj}_3\left(\mathcal{S}_1\right)$
  $\mathcal{S}_3 \leftarrow \mathcal{S}_2 \cap \neg\mathcal{S}_2$
  return $\mathcal{S}_1 \cap (\mathcal{S}_{\mathbb{R}^2} \times \mathcal{S}_3)$
Function `SepNormalized`$(\mathcal{S}_{\mathbb{X}}, \mathcal{S}_{\mathbf{c}^{\mathbb{X}}})$**:**
  $\mathbf{h_1} \leftarrow (\mathbf{c}, \mathbf{x_n}) \mapsto \mathbf{x_n} \cdot c_3 + \mathbf{c_{12}}$
  $\mathcal{S}_1 \leftarrow \neg\text{proj}_{\mathbf{45}}\left\{(\mathcal{S}_{\mathbf{c}^{\mathbb{X}}} \times \mathcal{S}_{\mathbb{R}^2}) \cap \mathbf{h_1^{-1}}(\neg\mathcal{S}_{\mathbb{X}})\right\}$
  $\mathcal{S}_2 \leftarrow \text{proj}_{\mathbf{45}}\left\{(\mathcal{S}_{\mathbf{c}^{\mathbb{X}}} \times \mathcal{S}_{\mathbb{R}^2}) \cap \mathbf{h_1^{-1}}(\mathcal{S}_{\mathbb{X}})\right\}$
  return $\mathcal{S}_{\text{pair}}(\mathcal{S}_1, \mathcal{S}_2)$
Function `SepRotation`$(\mathcal{S}_{\mathbb{A}_N}, \mathcal{S}_{\mathbb{B}_N})$**:**
  $\mathcal{S}_1 \leftarrow$ `SepCartPol`$(\mathcal{S}_{\mathbb{A}_N})$
  $\mathcal{S}_2 \leftarrow$ `SepCartPol`$(\mathcal{S}_{\mathbb{B}_N})$
  $\mathbf{h_2^+} \leftarrow (\mathbf{y}, \theta) \mapsto \mathbf{y} + (0, \theta)^\intercal$
  $\mathbf{h_2^-} \leftarrow (\mathbf{y}, \theta) \mapsto \mathbf{y} - (0, \theta)^\intercal$
  $\mathcal{S}_3 \leftarrow \neg\text{proj}_3\left\{(\mathcal{S}_1 \times \mathcal{S}_{\mathbb{R}}) \cap \mathbf{h_2^{+-1}}(\neg\mathcal{S}_2)\right\}$
  $\mathcal{S}_4 \leftarrow \neg\text{proj}_3\left\{(\mathcal{S}_2 \times \mathcal{S}_{\mathbb{R}}) \cap \mathbf{h_2^{--1}}(\neg\mathcal{S}_1)\right\}$
  return $\mathcal{S}_3 \cap \mathcal{S}_4$
Function `SepScaling`$(\mathcal{S}_{c_3^{\mathbb{A}}}, \mathcal{S}_{c_3^{\mathbb{B}}})$**:**
  $h_3 \leftarrow (c_3^{\mathbb{A}}, c_3^{\mathbb{B}}) \mapsto c_3^{\mathbb{B}} / c_3^{\mathbb{A}}$
  return $h_3(\mathcal{S}_{c_3^{\mathbb{A}}}, \mathcal{S}_{c_3^{\mathbb{B}}})$
Function `SepTranslation`$(\mathcal{S}_{\mathbf{c_{12}^{\mathbb{A}}}}, \mathcal{S}_{\mathbf{c_{12}^{\mathbb{B}}}}, \mathcal{S}_{\mathbb{K}}, \mathcal{S}_{\Theta})$**:**
  $\mathbf{h_4} \leftarrow (\mathbf{c_{12}^{\mathbb{A}}}, \mathbf{c_{12}^{\mathbb{B}}}, k, \theta) \mapsto \mathbf{c_{12}^{\mathbb{B}}} - k \cdot \mathbf{R}_\theta\, \mathbf{c_{12}^{\mathbb{A}}}$
  return $\mathbf{h_4}(\mathcal{S}_{\mathbf{c_{12}^{\mathbb{A}}}}, \mathcal{S}_{\mathbf{c_{12}^{\mathbb{B}}}}, \mathcal{S}_{\mathbb{K}}, \mathcal{S}_{\Theta})$

5.6. Integration of $\mathcal{S}_{\mathbb{P}}$ within a paving algorithm

The separator $\mathcal{S}_{\mathbb{P}}$ derived in Algorithms 2 and 3 provides the fundamental contraction power to characterize the four transformation parameters $(\theta, \mathbf{t}, k)$. However, embedding this separator within a naive branch-and-separate paving algorithm over the full four-dimensional search space $\mathbb{P}$ leads to significant computational overhead due to the curse of dimensionality. To overcome this, a first improvement involved proposing a specialized branching strategy to enhance performance.

Alternatively, instead of performing bisections over the parameter domain, in the experiments presented hereafter, we utilize a more efficient "combinatorial" strategy that mirrors the Procrustes-like decomposition of the problem. This approach breaks down the 4D search into a coordinated sequence of lower-dimensional sub-problems:

1. **Descriptor characterization:** A dedicated paving process first approximates the descriptors of the input sets (radii and centers of the MBCs) by invoking $\mathcal{S}_{\texttt{MinCircle}}$.

2. **Angular search:** The rotation parameter θ is then solved via a one-dimensional paving. This stage recursively utilizes the normalization separators $\mathcal{S}_{\mathbb{A}_N}$ and $\mathcal{S}_{\mathbb{B}_N}$ (where the normalized sets can be viewed as two-dimensional intermediate variables) to maintain consistency with the previously bounded descriptors.

3. **Analytical extension:** Finally, the scaling k and translation $\mathbf{t}$ are computed through direct interval evaluations, benefiting from the sharp enclosures obtained in the preceding steps.

This hierarchical decomposition significantly reduces the number of required bisections, ensuring that the global separator $\mathcal{S}_{\mathbb{P}}$ remains computationally tractable while preserving the guarantee of capturing all consistent solutions.

The use of a paving serves another purpose. Indeed, interval-based methods are inherently susceptible to over-approximations. One source of over-approximation is the *wrapping effect*, which arises when sets are enclosed by simple geometric shapes such as boxes or polar sectors (pies). In particular, the Cartesian–polar conversion induces a wrapping effect: a Cartesian box will necessarily be over-approximated in a polar representation.

Nevertheless, wrapping effect is mitigated when a paving is employed, as the subdivision allows for a tighter approximation of sets boundaries. In our implementation, this effect is reduced during the paving procedure performed in the projection step of equation (53), as well as the final paving of $\mathcal{S}_{\mathbb{P}}$.

The second source of over-approximation is pessimism in interval computations. However, we note that the $\mathcal{C}_{\text{polar}}$ does not introduce additional pessimism. Our Cartesian–polar separator `SepCartPolar`, described in Algorithm 1, is minimal and does not itself introduce pessimism.

6. Experiments

This section evaluates the performance and robustness of the proposed set-membership registration approach across various configurations. We specifically focus on the algorithm's ability to handle different levels of symmetry and degenerate cases, which are often problematic for local optimization methods.

6.1. The benchmark

The separators for the input sets $\mathbb{A}$ and $\mathbb{B}$ are constructed from high-resolution binary images (4000×4000 pixels). The dataset, illustrated in Figure 10, comprises five distinct pairs of shapes designed to test specific geometric challenges:

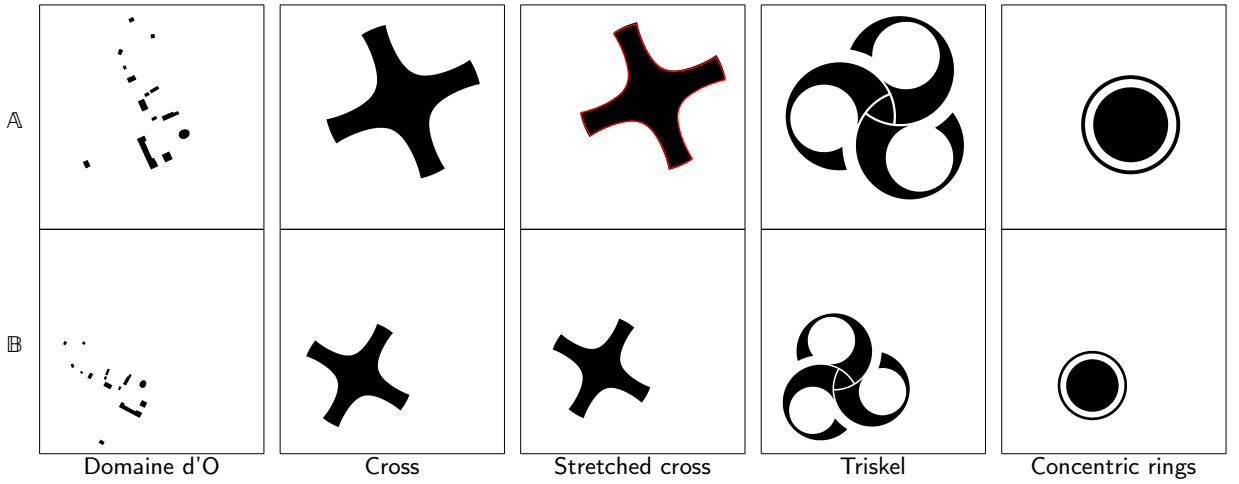


Figure 10: Benchmark image pairs used for the registration evaluation. Each column represents a specific geometric challenge: the first row displays the reference sets $\mathbb{A}$, and the second row shows the target sets $\mathbb{B}$ to be registered. For the stretched cross, we overlay the contour of the perfect-cross case in red, highlighting the subtle geometric difference that reduces the number of symmetry-related solutions from four to two.

- **Domaine d'O**: This case represents a realistic, asymmetrical scenario based on a satellite view of a park in Montpellier, France. It is directly motivated by robotics, in particular state-estimation from aerial imagery. Its elongated morphology is particularly demanding for the `SepMinCircle` operator, as the center of the minimum bounding circle is sensitive to the set's extremities, making its estimation less accurate.
- **Cross**: A symmetric geometric shape possessing four distinct valid rotations. This test assesses the algorithm's capacity to identify all isolated solutions within the search space.
- **Stretched cross**: A modified version of the previous cross, stretched along the horizontal axis by a factor of 3%. While the symmetry is reduced to two solutions, this case tests the separator's ability to rigorously dismiss the two "near-solutions" that appear visually plausible but are geometrically inconsistent. This property is illustrated

in Section 6.2.3. This example will serve as a basis for comparison with the ICP method in Section 6.2.4, highlighting situations in which such approaches may converge to incorrect alignments, and motivating the use of our exhaustive and guaranteed method.

- **Triskel**: A non-convex and non-connected shape. This complex topology leads to three discrete solutions, challenging the robustness of the rotational mapping $\mathcal{S}_\Theta$.
- **Concentric rings**: Representing a degenerate limit case, these shapes of revolution possess an infinite number of solutions. This benchmark evaluates how the paving algorithm behaves when the solution set Θ is a continuous interval rather than a set of discrete points.

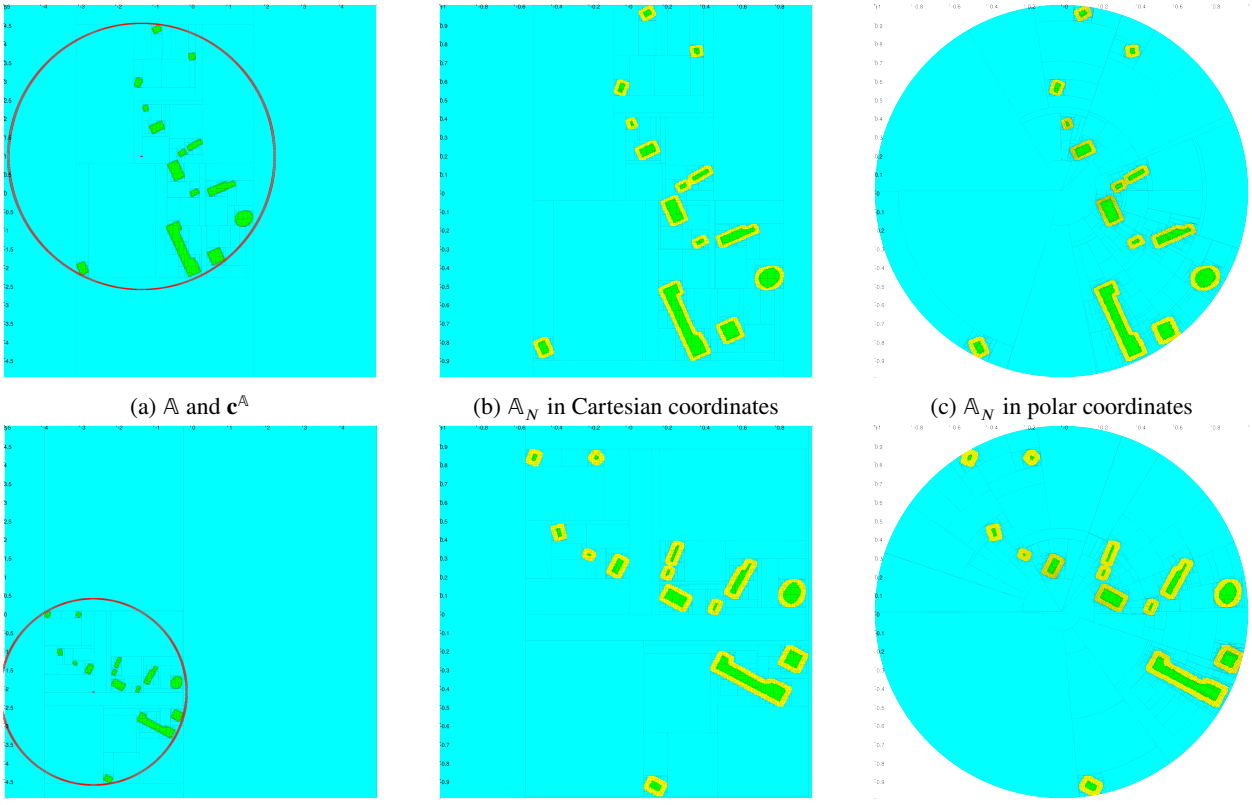
6.2. Experimental results

The benchmark was processed using the hierarchical sequence of paving algorithms described in Section 5.6, implementing the separators from Algorithms 2 and 3.

6.2.1. Precision parameter setting and intermediate set visualization

A global precision parameter $\epsilon = 10^{-3}$ was defined as the stopping criterion for all sub-paving processes. This threshold is applied both to the main branching loops and to the internal mechanisms of the projection separators, ensuring a consistent resolution across all transformation parameters.

Figure 11 illustrates approximations of the key intermediate sets for the *Domaine d'O* case: $\mathbb{A}$, $\mathbb{B}$, $\mathbb{A}_N$ and $\mathbb{B}_N$.



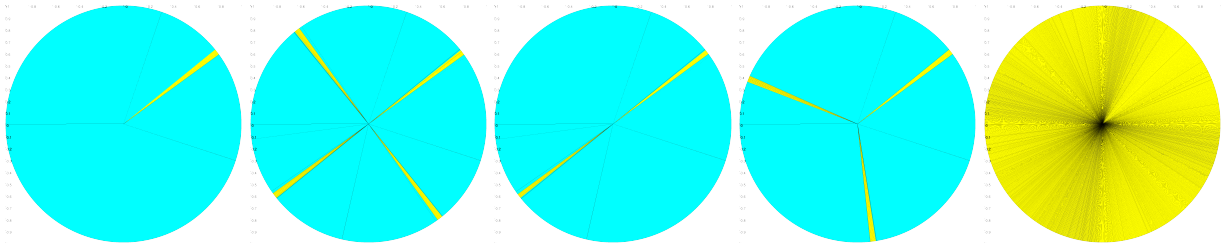
(a) $\mathbb{A}$ and $\mathbf{c}^{\mathbb{A}}$ (b) $\mathbb{A}_N$ in Cartesian coordinates (c) $\mathbb{A}_N$ in polar coordinates

(d) $\mathbb{B}$ and $\mathbf{c}^{\mathbb{B}}$ (e) $\mathbb{B}_N$ in Cartesian coordinates (f) $\mathbb{B}_N$ in polar coordinates

Figure 11: Visualization of the key intermediate steps for the *Domaine d'O* benchmark. The reference $(\mathbb{A}, \mathbb{B})$ and normalized $(\mathbb{A}_N, \mathbb{B}_N)$ sets are approximated by evaluating their respective separators within a paving algorithm.

6.2.2. Experimental results and analysis

The results of the angular search are presented in Figure 12. Each blue region represents a set of angles rigorously proved to contain no valid rotation for the registration problem.



(a) Domaine d'O (b) Cross (c) Stretched cross (d) Triskel (e) Conc. Rings

Figure 12: Rotation(s) solutions, isolated in outer-approximations drawn as yellow pies.

Bench	$c_3^{\mathbb{A}}$	$\mathbf{c}_{12}^{\mathbb{A}}$	$c_3^{\mathbb{B}}$	$\mathbf{c}_{12}^{\mathbb{B}}$	θ	$\mathbf{t}$	Time (s)
Domaine d'O	0.905	5.54	0.968	4.65	7.39	11.5	1.30
Cross	0.994	2.81	1.00	2.98	5.50 5.30 5.56 6.57	7.54 7.78 7.57 8.24	13.6
Stretched cross	0.999	3.23	0.995	3.87	6.16 6.08	13.3 13.2	3.69
Triskel	0.998	3.46	1.00	3.11	6.59 6.42 6.61	8.58 8.59 8.95	13.1
Conc. rings	1.00	2.00	0.974	2.07	–	–	21.7

Table 1
Precision and computational performance. Measured uncertainties on the transformation parameters for $\varepsilon = 10^{-3}$. Linear uncertainties (radii, centers, and translation) are expressed in thousandths of the image size; angular uncertainty θ is in 10^{-3} of 2π. Multiple rows per benchmark indicate distinct isolated solution groups found by the paving algorithm.

Note that in the *Cross* and *Triskel* cases, the figure clearly reveals four and three distinct solution boxes, respectively, perfectly capturing the discrete rotational symmetries of these shapes.

The *Concentric rings* case results in a continuous band of non-pruned space, correctly reflecting the infinite number of solutions inherent to shapes of revolution.

These results highlight the exhaustive nature of the approach: while local search methods might get trapped in one of the multiple minima of the *Triskel*, our separator-based paving recovers the entire set of consistent rotations.

Table 1 summarizes the numerical performance of the algorithm for a fixed precision $\varepsilon = 10^{-3}$. It details the total computation time and the resulting uncertainties for each estimated parameter: the MBC descriptors (radius and center), the rotation angle θ, and the translation vector $\mathbf{t}$.

The results demonstrate that the hierarchical strategy maintains a sharp enclosure of the solution even for complex shapes. For symmetrical cases like the *Cross* or *Triskel*, each row represents an isolated consistent region in the parameter space, confirming the algorithm's ability to distinguish between all global minima. In the case of *Concentric rings*, the rotation and translation uncertainties are marked as "–" since the solution set is a continuous manifold rather than a discrete set of points.

Remark 2. *Also note that there is no green pies for the rotation solutions in Figure 12: they are either blue or yellow. Let us recall that any separator can identify the following:*

- *blue pies that are rejected regions, i.e., that contain no solutions,*
- *green pies that correspond to entirely feasible regions (all points are solutions),*
- *yellow pies that contain points that are not yet classified: subsequent subdivision of the yellow region will allow one to classify some subregions in green or blue. Therefore, a yellow pie may still contain no solution: it only means that the separator could not prove its exclusion at the chosen precision.*

Precision ε	Threshold on stretching factor
1/400	1.0573
1/1000	1.0275
1/4000	1.0065

Table 2
Threshold values of the stretching factor below which near-solutions can no longer be ruled out, for different precision parameters.

As for rotation solutions, it is not possible to obtain green pies. A good example to understand this is the case of Concentric rings, for which the ideal shape admits a continuum of rotational symmetries. Imagine now a variant with a small bump on only the set $\mathbb{A}$ and invisible at the chosen precision. This bump removes the exact equality between the two sets and explains that no green region can be computed and yellow regions persist as shown in Figure 12e.

6.2.3. Detailed analysis of the Stretched cross

For the *Stretched cross*, the algorithm successfully identifies exactly two isolated solution boxes. Despite the geometric proximity to the four-fold symmetry of the standard *Cross*, the separator-based approach rigorously dismisses the two 'parasitic' orientations, proving they are inconsistent with the prescribed precision ε.

To investigate this behavior, we conduct a sensitivity analysis with respect to the stretching factor. Starting from the *Stretched cross*, we progressively reduce the deformation toward the *Cross* by decreasing the stretching factor toward 1.

For each configuration, our method is applied and the resulting solution set is analyzed in terms of the number of distinct clusters. While the *Stretched cross* yields exactly two clusters corresponding to the valid solutions, reducing the deformation eventually leads to the appearance of additional clusters associated with inconsistent near-solutions.

This allows us to identify a threshold value of the stretching factor below which the method is no longer able to rule out all near-solutions at the prescribed precision ε, as well as other precision parameters in order to assess the robustness of this transition. To ensure accuracy at finer precisions, we rely on an analytic characterization of the sets rather than the bitmap approximation seen in Figure 10. The threshold values identified for different precision parameters are reported in Table 2.

At finer precisions, the method correctly identifies that the *Stretched cross* admits only two solutions, even for very small deformations. This suggests that the approach is able to discriminate any difference between the two shapes, provided that the precision is sufficiently small.

6.2.4. Comparison with ICP on a representative case

The stretched cross configuration, involving near-solutions, is a challenging scenario where classical methods such as ICP may fail. To this end, we compare our approach, which is able to reject the two near-solutions, with the ICP algorithm.

The following experimental protocol is used to evaluate the behavior of the ICP algorithm. The sets are first sampled into point clouds of 1000 points each. The ICP algorithm is then run 1000 times, each time with a different initial configuration: the translation is fixed at the origin, while the orientation is uniformly sampled over the rotation space, with increments of one thousandth of a full rotation.

The resulting transformations are grouped into clusters corresponding to the different alignment configurations. Out of the 1000 runs, 469 converge to valid solutions (177 and 292, respectively), while 531 converge to near-solutions (301 and 230).

Notably, more than half of the runs converge to geometrically inconsistent near-solutions, illustrating the tendency of ICP to be attracted by visually plausible but incorrect alignments.

6.2.5. Concluding remarks

We conclude the experimental section with several remarks.

Uncertainty propagation As outlined in the hierarchical resolution strategy, parameters are estimated sequentially: starting with the MBC radii (scaling), followed by the centers, the rotation θ, and finally the translation $\mathbf{t}$. Table 1 highlights a gradual accumulation of uncertainty: each stage inherits and expands the bounding boxes from the previous ones. This dependency reflects the natural coupling of the Procrustes decomposition within an interval framework.

Geometric sensitivity of the center estimate The *Domaine d'O* case illustrates the geometric sensitivity of the `SepMinCircle` operator. The center's precision depends on the spatial distribution of the points lying on the MBC boundary. In cases where a set is elongated, the center is essentially determined by the intersection of two almost tangent circles (defined by the diametrically opposed points and the radius uncertainty). As shown in Figure 11, although three points constrain the circle for the *Domaine d'O*, their poor angular distribution leads to a higher uncertainty in the center's coordinates. Conversely, the *Concentric rings* provide an optimal configuration where the entire outer boundary constrains the MBC, allowing for a much sharper triangulation of the center despite the shape's complexity.

Coupling of transformation parameters Figure 12 visually characterizes the valid rotation set Θ. The strength of the `SepTranslation` operator (Algorithm 3) lies in its ability to maintain the functional link between θ and $\mathbf{t}$. Rather than merely providing independent intervals, the separator identifies specific translation vectors compatible with given rotation candidates. This approach yields a much more informed representation of the solution manifold than a simple Cartesian product $\Theta \times \mathbb{T}$.

Handling the continuum of solutions The *Concentric rings* represent the most challenging scenario for a paving algorithm. Since every angle $\theta \in [0, 2\pi]$ is a valid solution, the separators cannot prune any part of the angular domain. This results in the maximum computation time observed (21.7s), as the algorithm must exhaustively tile the entire solution manifold. This "worst-case" performance remains tractable, demonstrating the robustness of the set-membership approach.

7. Conclusion

We have proposed a set-theoretical approach to the registration problem that achieves guaranteed and exhaustive results within reasonable computation time. By adapting the Procrustes framework to the interval analysis paradigm, the method distinguishes itself from classical probabilistic techniques by its ability to rigorously identify and prune non-solutions without omitting any potential matches.

The strength of this approach lies in the expressive power of separators. These tools allow for a direct translation of the sets' analytical expressions into efficient contractors. Our method mirrors the standardization steps of the Procrustes method (normalization, rotation, and scaling), but performs them in a guaranteed way. This hierarchical decomposition reduces a high-dimensional combinatorial problem into a sequence of smaller, lower-dimensional, and less computationally intensive sub-problems, fulfilling the objectives of completeness and efficiency.

Perspectives

While the current work focuses on similarity transformations in 2D space, several avenues exist to enhance and extend the proposed framework.

Algorithmic enhancements The efficiency of the resolution can be further improved by moving toward a more "declarative" Constraint Satisfaction Problem (CSP) approach, utilizing a single branch-and-contract algorithm with an optimized bisection strategy. Alternatively, within the current sequential strategy, specific separators could be refined. For instance, the $\mathcal{S}_{\texttt{MinCircle}}$ could be re-engineered by moving away from a purely equational definition toward one inspired by classical geometric algorithms to gain performance.

Reflectivity Reflectivity can be addressed using the existing tools. A straightforward extension would involve performing the rotational mapping twice: once for the original set and once for its mirrored version, thereby accounting for all possible reflection-based matches.

Extension to 3D space Transitioning to 3D space introduces two primary challenges. First, the Special Orthogonal group SO(3) is three-dimensional, significantly increasing the search space complexity compared to the one-dimensional SO(2). Second, 3D rotations do not benefit from the simple additive property of planar angles. Consequently, the rotational mapping step presented here must be adapted to handle the more complex geometry of 3D rotations while maintaining the set-membership guarantees.

References

Anandan, P., 1989. A computational framework and an algorithm for the measurement of visual motion. International Journal of Computer Vision 2, 283–310.

Benhamou, F., Goualard, F., Granvilliers, L., Puget, J., 1999. Revising hull and box consistency, in: Schreye, D.D. (Ed.), Logic Programming: The 1999 International Conference, Las Cruces, New Mexico, USA, November 29 - December 4, 1999, MIT Press. pp. 230–244.

Besl, P.J., McKay, N.D., 1992. Method for registration of 3-D shapes, in: Sensor Fusion IV: Control Paradigms and Data Structures, SPIE. pp. 586–606.

Bonneel, N., Coeurjolly, D., 2019. SPOT: sliced partial optimal transport. ACM Transactions on Graphics 38, 89:1–89:13.

Chabert, G., 2012. Ibex documentation.

Chabert, G., Jaulin, L., 2009. Contractor programming. Artificial Intelligence 173, 1079–1100.

Chen, L., Feng, C., Ma, Y., Zhao, Y., Wang, C., 2023. A review of rigid point cloud registration based on deep learning. Frontiers Neurorobotics 17. URL: https://doi.org/10.3389/fnbot.2023.1281332, doi:10.3389/FNBOT.2023.1281332.

Chen, Y., Medioni, G., 1992. Object modelling by registration of multiple range images. Image and Vision Computing 10, 145–155.

Cliff, N., 1966. Orthogonal rotation to congruence. Psychometrika 31, 33–42.

Desrochers, B., 2018. Simultaneous localization and mapping in unstructured environments : a set-membership approach. phdthesis. ENSTA Bretagne - École nationale supérieure de techniques avancées Bretagne.

Desrochers, B., Jaulin, L., 2016. A minimal contractor for the polar equation: Application to robot localization. Eng. Appl. Artif. Intell. 55, 83–92. URL: https://doi.org/10.1016/j.engappai.2016.06.005, doi:10.1016/J.ENGAPPAI.2016.06.005.

Durrant-Whyte, H., Bailey, T., 2006. Simultaneous localization and mapping: part I. IEEE Robotics & Automation Magazine 13, 99–110.

Fard, M.G., Peña-Mora, F., 2007. Application of Visualization Techniques for Construction Progress Monitoring, in: Computing in Civil Engineering (2007), American Society of Civil Engineers, Pittsburgh, Pennsylvania, United States. pp. 216–223.

Goualard, F., . GAOL: a C++ interval arithmetic library that strives to offer fast and reliable operators for constraint solvers.

Gower, J.C., 1975. Generalized procrustes analysis. Psychometrika 40, 33–51.

Gower, J.C., Dijksterhuis, G.B., 2004. Procrustes Problems. OUP Oxford.

Green, B.F., 1952. The orthogonal approximation of an oblique structure in factor analysis. Psychometrika 17, 429–440.

Horn, B.K., Hilden, H.M., Negahdaripour, S., 1988. Closed-form solution of absolute orientation using orthonormal matrices. Journal of the Optical Society of America A 5, 1127–1135.

Horn, B.K.P., 1987. Closed-form solution of absolute orientation using unit quaternions. JOSA A 4, 629–642.

Huang, Q., Flöry, S., Gelfand, N., Hofer, M., Pottmann, H., 2006. Reassembling fractured objects by geometric matching. ACM Trans. Graphics (TOG) 25, 569–578.

Irani, M., Peleg, S., 1991. Improving resolution by image registration. CVGIP: Graphical models and image processing 53, 231–239.

Jaulin, L., Desrochers, B., 2014. Introduction to the Algebra of Separators with Application to Path Planning. Engineering Applications of Artificial Intelligence 33.

Jaulin, L., Kieffer, M., Didrit, O., Walter, E., 2001. Applied Interval Analysis, with examples in parameter and state estimation, robust control and robotics. Springer-Verlag, London.

Jaulin, L., Walter, E., 1993. Set inversion via interval analysis for nonlinear bounded-error estimation. Automatica 29, 1053–1064. URL: https://www.sciencedirect.com/science/article/pii/0005109893901064, doi:https://doi.org/10.1016/0005-1098(93)90106-4.

Lucas, B.D., Kanade, T., 1981. An iterative image registration technique with an application to stereo vision, in: IJCAI'81: 7th international joint conference on Artificial intelligence, pp. 674–679.

Megiddo, N., 1983. Linear-time algorithms for linear programming in R^3 and related problems. SIAM journal on computing 12, 759–776.

Moore, R.E., 1966. Interval analysis. volume 4. Prentice-Hall Englewood Cliffs.

Qi, C.R., Su, H., Mo, K., Guibas, L.J., 2017. Pointnet: Deep learning on point sets for 3d classification and segmentation, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017, IEEE Computer Society. pp. 77–85. URL: https://doi.org/10.1109/CVPR.2017.16, doi:10.1109/CVPR.2017.16.

Revol, N., 2017. Introduction to the ieee 1788-2015 standard for interval arithmetic, in: Abate, A., Boldo, S. (Eds.), Numerical Software Verification, Springer International Publishing, Cham. pp. 14–21.

Rohlf, F.J., Slice, D., 1990. Extensions of the procrustes method for the optimal superimposition of landmarks. Systematic zoology 39, 40–59.

Rohou, S., Desrochers, B., Le Bars, F., 2024. The Codac library. Acta Cybernetica 26, 871–887. URL: https://cyber.bibl.u-szeged.hu/index.php/actcybern/article/view/4388, doi:10.14232/actacyb.302772.

Rohou, S., Franek, P., Aubry, C., Jaulin, L., 2018. Proving the existence of loops in robot trajectories. The International Journal of Robotics Research 37, 1500–1516. URL: journals.sagepub.com/doi/full/10.1177/0278364918808367, doi:10.1177/0278364918808367.

Safra, E., Kanza, Y., Sagiv, Y., Doytsher, Y., 2013. Ad hoc matching of vectorial road networks. Int. J. Geographical Information Science 27, 114–153.

Schönemann, P.H., 1966. A generalized solution of the orthogonal procrustes problem. Psychometrika 31, 1–10.

Schönemann, P.H., Carroll, R.M., 1970. Fitting one matrix to another under choice of a central dilation and a rigid motion. Psychometrika 35, 245–255.

Sullivan, G.J., Baker, R.L., 1991. Motion compensation for video compression using control grid interpolation, in: Acoustics, Speech, and Signal Processing, IEEE International Conference on, IEEE Computer Society. pp. 2713–2714.

Sylvester, J., 1857. A question in the geometry of situation. Quarterly Journal of Pure and Applied Mathematics 1:79.

Ten Berge, J.M., 1977. Orthogonal procrustes rotation for two or more matrices. Psychometrika 42, 267–276.

Wang, Y., Solomon, J., 2019. Deep closest point: Learning representations for point cloud registration, in: 2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019, IEEE. pp. 3522–3531. URL: https://doi.org/10.1109/ICCV.2019.00362, doi:10.1109/ICCV.2019.00362.

Wang, Y., Sun, Y., Liu, Z., Sarma, S.E., Bronstein, M.M., Solomon, J.M., 2019. Dynamic graph CNN for learning on point clouds. ACM Trans. Graph. 38, 146:1–146:12. URL: https://doi.org/10.1145/3326362, doi:10.1145/3326362.

Woods, R.P., Mazziotta, J.C., Cherry, S.R., et al., 1993. MRI-PET Registration with Automated Algorithm. Journal of Computer Assisted Tomography 17, 536–546.

Yang, H., Shi, J., Carlone, L., 2021. TEASER: fast and certifiable point cloud registration. IEEE Trans. Robotics 37, 314–333. URL: https://doi.org/10.1109/TRO.2020.3033695, doi:10.1109/TRO.2020.3033695.

Yang, J., Li, H., Campbell, D., Jia, Y., 2016. Go-ICP: A Globally Optimal Solution to 3D ICP Point-Set Registration. IEEE Transactions on Pattern Analysis and Machine Intelligence 38, 2241–2254.

Zhang, Z., 1994. Iterative point matching for registration of free-form curves and surfaces. International Journal of Computer Vision 13, 119–152.

Zhou, Q., Park, J., Koltun, V., 2016. Fast global registration, in: Leibe, B., Matas, J., Sebe, N., Welling, M. (Eds.), Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II, Springer. pp. 766–782. URL: https://doi.org/10.1007/978-3-319-46475-6_47, doi:10.1007/978-3-319-46475-6_47.

A. Codac Python code associated with Figure 8b

The following `Codac` Python snippet reproduces the paving shown in Figure 8b. To visualize the result, the `VIBes` viewer must be installed and started prior to running the script. More information on `https://codac.io`.

```
from codac import * # Codac v2.0.0

x,p = VectorVar(2), VectorVar(5)

e = AnalyticFunction([x,p],
  sqr(((x[0]-p[2])*cos(p[4])+(x[1]-p[3])*sin(p[4]))/p[0])
  + sqr(((x[0]-p[2])*sin(p[4])-(x[1]-p[3])*cos(p[4]))/p[1]))
e1 = AnalyticFunction([x], e(x,Vector([3,1,-1,0,PI/3.])))
e2 = AnalyticFunction([x], e(x,Vector([2,1,0,0.5,0])))

se1 = SepInverse(e1,[-oo,1])
se2 = SepInverse(e2,[-oo,1])

DefaultFigure.pave([[-4,3],[-3,3]], se1 & ~se2, 0.05)
```