Approximate Reflection Symmetry in a Point Set: Theory and Algorithm with an Application

Rajendra Nagar* and Shanmuganathan Raman†
Electrical Engineering, Indian Institute of Technology Gandhinagar
Gujarat, India, 382355
Email: *rajendra.nagar@iitgn.ac.in, †shanmuga@iitgn.ac.in

Abstract—We propose an algorithm to detect approximate reflection symmetry present in a set of volumetrically distributed points belonging to $\mathbb{R}^d$ containing a distorted reflection symmetry pattern. We pose the problem of detecting approximate reflection symmetry as the problem of establishing the correspondences between the points which are reflections of each other and determining the reflection symmetry transformation. We formulate an optimization framework in which the problem of establishing the correspondences amounts to solving a linear assignment problem and the problem of determining the reflection symmetry transformation amounts to an optimization problem on a smooth Riemannian product manifold. The proposed approach estimates the symmetry from the distribution of the points and is descriptor independent. We evaluate the robustness of our approach by varying the amount of distortion in a perfect reflection symmetry pattern where we perturb each point by a different amount of perturbation. We demonstrate the effectiveness of the method by applying it to the problem of 2-D reflection symmetry detection along with relevant comparisons.

Index Terms—Reflection Symmetry, Optimization on Manifolds, Correspondences on a Point Set, Linear Assignment.

1 INTRODUCTION

The symmetry present in natural and man-made objects, enriches the objects to be physically balanced, beautiful, easy to recognize, and easy to understand. Characterizing and finding the symmetry has been an active topic of research in computer vision and computer graphics as physical objects form the basis for these research areas. The digitized objects are mainly represented in the form of meshes, volumes, sets of points, and images. The primary objective has been to detect symmetry in objects depicted through these different representations. We particularly aim to detect reflection symmetry present in objects represented by a set of finite number of points belonging to $\mathbb{R}^d$. In Figure 1 we present an example result of the proposed approach for illustration.

The motivation behind detecting symmetry in higher dimensional spaces ($d > 3$) is inspired from the fact that many physical data points reside in the space of dimension greater than three. For example, RGB-D image captured using a Kinect sensor, which has become a major tool of interaction of human with machines, has four dimensions at each pixel location. Another example is the embedding of feature points or shapes into a higher-dimensional space. In the scale invariant feature transform (SIFT) algorithm, each keypoint is represented in a 128 dimensional space [1]. The proposed approach not only targets data residing in 2-D (image) and 3-D (point cloud), but also develops a generic computational framework in order to detect symmetry from data residing in higher dimensions.

The problem of establishing correspondences between reflection symmetry points and determining the hyperplane of reflection symmetry has been extensively studied due to its astounding applications such as compression of objects, symmetrization, shape matching, and symmetry aware segmentation of shapes [2]. Most of the existing algorithms attempt this problem by using surface signatures such as Gaussian curvature, eigenbases of the Laplace-Beltrami operator, and Heat kernels, for the points sampled on a given surface ( [2], [3], [4]). The challenge we face is that we only have a set of discrete points in $\mathbb{R}^d$. We cannot take benefits from local surface signatures by fitting a surface over these points. For the case $d = 2$, an explanation could be that the prominent surface signatures, such as Gaussian curvatures, are meaningful only if the surface is non-linear. For the case $d \geq 3$, an explanation could be that if the points set represent a volumetric shape, fitting a surface could be hard and eigenbases of Laplace-Beltrami operator are not defined for a set of finite points since it is not a compact manifold without boundary [5]. Prominent methods such as [6] and [7] are independent of surface features and employ randomized algorithms in order to establish correspondences between the reflection symmetry points. However, they require fine tuning of a hyper-parameter in order to handle the reflection symmetry patterns perturbed by an unknown source of noise and improper choice of this parameter might lead to higher time complexity.

Both these categories of algorithms are sequential in the sense that they first establish the correspondences between the reflection symmetry points and then determine the reflection symmetry hyperplane. Therefore, many outlier correspondences could be detected along with the correct correspondences. In summary,
detecting symmetry in a set containing finite number of points is a non-trivial problem.

In this work, we propose an optimization framework where we jointly establish correspondences between reflection symmetry points and determine the reflection symmetry transformation in a set of discrete points residing in $\mathbb{R}^d$ containing a distorted reflection symmetry pattern. In order to design the cost function, we introduce an affine transformation to obtain the reflection point of a point in $\mathbb{R}^d$. The main intuition behind forming this cost function is that the reflection point of a point obtained through the optimal reflection hyperplane should be present closest to its ground-truth reflection point. The primary contributions of this work are listed below.

1) We propose an optimization based algorithm to establish correspondences between the reflection symmetry points and determine the reflection symmetry transformation in a set of discrete points residing in $\mathbb{R}^d$ containing a distorted reflection symmetry pattern.

2) We show that the proposed optimization framework is convex in translation and correspondences matrix and locally convex in each of the rotation matrices.

3) The proposed approach is shown to not use any shape descriptors and can be applied to point sets obtained by sampling any shape residing in $\mathbb{R}^d$.

We organize the remainder of the paper as follows. In §2 we present state-of-the-art works relevant to the present work. In §3.1 we formulate an energy minimization problem which is a function in rotation, translation, and mirror symmetric correspondences. In §3.2 we optimize the formulated energy function with respect to the rotation and the translation. In §3.3 we optimize the formulated energy function with respect to the mirror symmetric correspondences. In §3.4 we prove the convergence properties of the proposed alternating optimization scheme. In §4 we report the results and the evaluation of the proposed approach with a state-of-the-art approach for 2-D and 3-D. In §5 we conclude the proposed approach with limitations and future directions. We present few proofs in the supplementary file for completeness at the end of the paper.

2 Related Works

The problem of characterizing and detecting the reflection symmetry in digitized objects has been extensively studied. The works [8] and [2] provide a survey of the symmetry detection algorithms. The symmetry detection algorithms can be categorized based on either the form of the input data or whether the algorithm is dependent or independent of the surface features. General forms of the input data are: set of points, mesh, volume, and image. Most of the methods for symmetry detection in meshes first extract salient keypoints on the surface and then describe each point using local surface features. The prominent surface features are: Gaussian curvatures, slippage features, moments, geodesic distances, and extended Gaussian images.

Symmetry detection in a set of points without features. These algorithms detect reflection symmetry in a set of points without using surface features. Our work also falls in this category. In the work by Zabrodsky et al., the authors find the closest shape to a given shape represented by a set of points in $\mathbb{R}^2$ [9] and it requires point correspondences. However, our goal is different in the sense that we find reflection correspondences within the given set of points in $\mathbb{R}^d$. In the work by Lipman et al., the authors propose the concept of symmetry factored embedding where they represent pairs of points which are in the same orbit in a new space and propose the concept of symmetry factored distance to find the distance between such pairs [6]. In the work by Xu et al., the authors detect multi-scale symmetry [7]. The authors use a randomized algorithm to detect the correspondences efficiently. However, performance degrades as the perfect pattern gets perturbed due to noisy measurements. We compare the correspondences established by our method to that of this method and show that our method performs better than this method when the patterns are perturbed. It is fair to compare with this method on the perturbed patterns because most of the practical patterns are not perfectly symmetric e.g., human face and butterfly wings. In the work by Combès et al., the authors automatically detect the symmetry plane in a point cloud but do not establish correspondences [10]. However, correspondences are an important aspect as shown in the work by [7].

Symmetry detection in meshes using surface features. These algorithms either directly use surface patches described using local features or first detect the salient keypoints on the surface and describe using the local surface features. Here, we review only the salient works to give an idea of these algorithms. Mitra et al. detect partial and approximate symmetries in 3D models [4]. They start with sampling salient keypoints on the surface and pair them up using their local principal curvatures. Then using the Hough transformation, they find the pairs of reflection symmetry points. Then in the Hough transformation space, they perform the clustering of the pairs to determine all the partial symmetries. Martinet et al. detect symmetries by generalized moment functions where the shape symmetry gets inherited as symmetry in these functions [11]. Raviv et al. detect symmetry in non-rigid shapes by observing that the intrinsic geometry of a shape is invariant under non-rigid shape transformations [12]. Berner et al. start with constructing a graph based on the similarity of slippage features detected on the surface [3]. Then they detect the structural regularities by matching the subgraphs. Cohen et al. detect symmetry using geometric and image cues [13]. They use it to reconstruct accurate 3D models. We refer the reader to some of the pioneering works for more details on this category ([14], [15], [16], [17], [18], [19]). There exist algorithms which find symmetry in meshes and volume without sampling keypoints. The works described in [5], [20], [21], [22], [23], [24], [25], [26], [27] belong to this category.

Symmetry detection algorithm for real images. These algorithms primarily rely on the local image features such as edge orientations, curvatures, and gradients. The recent works such as (28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39) present excellent algorithms for reflection symmetry detection in images. Given the accurate detection of keypoints, the algorithm developed in this work can be used to detect reflection symmetry in images without using local features.

3 Proposed Approach

Consider a set $S = \{x_i\}_{i=1}^n$ of points, where $x_i \in \mathbb{R}^d$, containing a distorted reflection symmetry pattern. Our goal is to determine the reflection symmetry transformation and establish the correspondences between the reflection symmetry points. In Fig. [3] we show the graphical representation of our problem. We formulate an optimization framework in which both the correspondences
between reflection symmetry points and reflection symmetry transformation are variables as described below. We use the notation \([k]\) for the set \(\{1, 2, \ldots, k\}\), where \(k\) is a natural number.

### 3.1 Problem Formulation

We introduce the reflection transformation in \(\mathbb{R}^d\) in order to obtain the reflection of a point through a hyperplane \(\pi\), not necessarily passing through the origin. The intuition is based on the fact that any hyperplane in \(\mathbb{R}^d\) is a \((d-1)\) dimensional subspace. Therefore, it can be made parallel to the subspace spanned by any \((d-1)\) coordinate axes by translating the origin of the coordinate system on the hyperplane \(\pi\) and then rotating these \((d-1)\) axes sequentially (by the angle between the hyperplane \(\pi\) and the axis). In this new coordinate system, the reflection of a point through the hyperplane \(\pi\) can be obtained by multiplying the coordinate corresponding to the remaining axis of the point by \(-1\). Then the reflection in the original coordinate system is obtained by applying the inverse procedure.

**Definition 1.** The reflection point \(x_{i'} \in \mathbb{R}^d\) of a point \(x_i \in \mathbb{R}^d\) through the reflection symmetry hyper-plane \(\pi\) is determined by an affine transformation as shown in Equation (1)

\[
x_{i'} = \left( \prod_{u=1}^{d-1} R_u \right) ^\top E \left( \prod_{u=1}^{d-1} R_u \right) ^\top (x_i - t) + t.
\]

Here, \(i, i' \in [n], t \in \mathbb{R}^d\) is the translation vector which translates the origin of the coordinate system on the hyperplane \(\pi\), \(R_u\) is a rotation matrix of size \(d \times d\) that rotates \(u^{th}\) axis about the origin such that it becomes perpendicular to the normal of the hyperplane \(\pi\), and the matrix \(E\) is defined as \(E = \begin{bmatrix} I_{d-1} & 0_{d-1} \\ 0_{d-1} & -1 \end{bmatrix}\) and satisfies \(E^\top E = E^\top = I_d\). The matrix \(R_u\) is an orthogonal matrix \((R_u^\top R_u = R_u R_u^\top = I_d)\) with determinant equal to \(+1\), \(\forall u \in \{1, 2, \ldots, d-1\}\). Here, \(0_{d-1}\) is a vector of size \((d-1) \times 1\) with all coordinates equal to zero and \(I_{d-1}\) is an identity matrix of size \((d-1) \times (d-1)\).

Now, we introduce essential properties of this transformation in order to formulate the problem. We show that the rotation matrices \((R_1, \ldots, R_{d-1})\) and the translation vector \(t\) uniquely determine the reflection hyper-plane \(\pi\). We let \(T = \prod_{u=1}^{d-1} R_u\) throughout this paper and note that it is again an orthogonal matrix with determinant equal to \(+1\).

**Theorem 1.** The point \(x_{i'}\) is the reflection of the point \(x_i\) through the hyperplane \(\pi\) if and only if the point \(x_i\) is the reflection of the point \(x_{i'}\) through the hyperplane \(\pi\).

**Proof.** We prove the forward direction of the Theorem, since the backward direction can be proven in a similar way. Let us assume that the point \(x_{i'}\) is the reflection of the point \(x_i\), therefore equation (1) holds true. Now, we multiply equation (1) by \(T^\top\) from left and use the identities \(E^\top = E, EE^\top = I_d, T^\top T = TT^\top = I_d\) to achieve,

\[
T^\top x_{i'} = x_i - t + T^\top t
\]

\[
\Rightarrow x_i = T^\top (x_{i'} - t) + t. \quad \square
\]

**Theorem 2.** The normal vector of the reflection hyper-plane \(\pi\) lies in the null space of the matrix \(I_d + T^\top T\), the hyper-plane \(\pi\) passes through \(t\), and the null space of the matrix \(I_d + T^\top T\) is an one-dimensional subspace of \(\mathbb{R}^d\).

**Proof.** We subtract equation (1) from equation (2) to achieve \(x_i - x_{i'} = TET^\top (x_{i'} - x_i) = (I_d + TET^\top) (x_i - x_{i'}) = 0\).

Therefore, the normal to the reflection hyperplane \(\pi\), which is in the direction of the vector \((x_i - x_{i'})\), lies in the null space of the matrix \((I_d + TET^\top)\). It is easy to show that the reflection hyperplane \(\pi\) passes through the translation \(t\) by noting that the reflection point of the point \(t\) is \(t\). This is possible only if the point \(t\) lies on the reflection hyperplane \(\pi\). We prove that the null space of the matrix \((I_d + TET^\top)\) is an one-dimensional subspace of \(\mathbb{R}^d\) in order to show that there exists an unique hyperplane \(\pi\). The nullspace of a matrix is the space spanned by the eigenvectors corresponding to the zero eigenvalue. Let \(p = [p_1, p_2, \ldots, p_d]^\top \in \mathbb{R}^d\) be any vector. If \(p\) is an eigenvector corresponding to the zero eigenvalue of the matrix \((I_d + TET^\top)\) then we must have

\[
p^\top (I_d + TET^\top) = 0 = p^\top I_d p + (T^\top p)^\top E (T^\top p) = 0
\]

\[
\Rightarrow p^\top I_d p + b^\top E b = 0 = \sum_{u=1}^{d} p_u^2 + \sum_{u=1}^{d-1} b_u^2 - 2b_d^2 = 0
\]

\[
\Rightarrow \sum_{u=1}^{d} p_u^2 + \sum_{u=1}^{d-1} b_u^2 - 2b_d^2 = 0. \quad (3)
\]

Here, \(b = T^\top p\). We note that \(\|b\|^2 = (T^\top p)^\top (T^\top p) = p^\top p = \|p\|^2\). Therefore, from equation (3) we have:

\[
\sum_{u=1}^{d} p_u^2 + \sum_{u=1}^{d-1} b_u^2 - 2b_d^2 = 0 = \sum_{u=1}^{d-1} b_u^2 \Rightarrow b_u = 0. \quad (4)
\]

Therefore, \(b_1 = b_2 = \ldots = b_{d-1} = 0\) and \(b_d \in \mathbb{R}\). Hence, the vector \(b\) lies in the one dimensional space \(\{q_1 : q_1 = q_2 = \ldots = q_{d-1} = 0, q_d \in \mathbb{R}\}\). Since \(b = T^\top p \Rightarrow p = T b\). Hence, the vector \(p\) also lies in the one dimensional space \(\{Tq : q_1 = q_2 = \ldots = q_{d-1} = 0, q_d \in \mathbb{R}\}\).

Given the set \(S\), our goal is to find all the correct reflection correspondences \((i, i') \in [n] \times [n]\) and the matrices \((R_1, R_2, \ldots, R_{d-1}, t)\) which define the reflection symmetry hyperplane \(\pi\). We represent all the correspondences by a permutation matrix \(P \in \{0, 1\}^{n \times n}\), such that \(P_{ii'} = 1\) if the point \(x_{i'}\) is the
reflection point of the point \( x_i \) and \( P_{ji} = 0 \), otherwise. Here we note from Theorem 1 that \( P_{ji} = 1 \Leftrightarrow r_{ji} = 1 \).

Now, let \( R = (R_1, R_2, \ldots, R_{d-1}) \in \mathbb{V} \). Here, \( \mathbb{V} = \mathbb{R}^{d \times d} \times \mathbb{R}^{d \times d} \times \cdots \times \mathbb{R}^{d \times d} \). Let \( X = [x_1, x_2, \ldots, x_n] \in \mathbb{R}^{d \times n} \) be the matrix containing all the points of the set \( S \) as its columns. Since the \( i \)th column of the matrix \( XP \) is the reflection point of the point \( x_i \), the reflection transformation \((R, t)\) maps the points matrix \( X \) to the reflected points matrix \( XP \). Using Equation (1) we write the reflected points in the form of the matrix T

\[
\text{TET}^T(X - te^T) + te^T = XP.
\]

In practice, a reflection symmetry pattern might have been distorted. Therefore, we would be able to find only the approximate reflection symmetry. We find the reflection transformation \((R, t)\) and the correspondences matrix \( P \). Since the tangent space at a point in an Euclidean space is again an Euclidean space

\[
\min_{R \in \mathbb{R}^{d \times d}, t \in \mathbb{R}^d} \left\{ \left\| \left( \prod_{u=1}^{d-1} R_u \right)^{T} (X - te^T) + te^T - XP \right\|_F^2 \right. \\
\left. \text{s.t. } P e = e, P^T e = e, P \in \{0,1\}^{n \times n} \right.
\]

\[
R_u^T R_u = I_d, R_u \in \mathbb{R}^{d \times d}, \det(R_u) = 1, \forall u \in [d - 1], \forall t \in \mathbb{R}^d.
\]

By imposing the constraints \( Pe = e \) and \( P^T e = e \), we ensure that each point has only one reflection point. We adopt an alternating optimization approach to solve the problem defined in Equation (5). We start with initializing the reflection transformation \((R, t)\) and solve for the optimal correspondences \( P \) and then for this optimal \( P \), we solve for optimal the \((R, t)\). We continue to alternate till convergence.

Once \( P \) is fixed, if we minimize the cost over the set \( \mathbb{V} \), then we have to make sure that the orthogonality and the unit determinant constraints hold true for the matrices \( R_u, \forall u \in [d - 1] \). One approach could be the Lagrangean augmentation which requires us to handle \( 3d - 5 \) extra Lagrange multipliers. However, we observe that the set \( M = \{ (R_1, \ldots, R_{d-1}, t) : R_u^T R_u = R_u R_u^T = I_d, \det(R_u) = 1, R_u \in \mathbb{R}^{d \times d}, \forall u \in [d - 1], \forall t \in \mathbb{R}^d \} \) of constraints is a smooth Riemannian product manifold over which the optimization algorithms are well studied.

We solve the sub-optimization problem for \((R, t)\) on a manifold which we discuss in the section 3.3. We observe that the optimization of equation (5) for \( P \) is a standard linear assignment problem for which we formulate an integer linear program which we discuss in the section 3.3.

### 3.2 Optimizing reflection transformation \((R, t)\)

In this step, we fix the correspondences matrix \( P \) and find the optimal reflection transformation \((R, t)\) by taking advantages from the differential structure of the set \( M \). We shall now briefly introduce the differential geometry of the set \( M \).

**Differential geometry of the set \( M \) of constraints.**

In order to introduce the differential essential geometry of the set \( M \), we follow [40]. The elements of the set \( M \) are of the form \((R, t) \simeq (R_1, \ldots, R_{d-1}, t)\). All the orthogonal matrices with determinant +1 form a Lie group, also known as *special orthogonal group*, and denoted as \( SO(d) \), which is a smooth Riemannian manifold. The Euclidean space \( \mathbb{R}^d \) is also a smooth Riemannian manifold. Therefore the set \( M \) is a product manifold, \( SO(d) \times \cdots \times SO(d) \times \mathbb{R}^d \), the product of \( d \)− 1 special orthogonal groups \( SO(d) \) and an Euclidean space \( \mathbb{R}^d \). The tangent space \( T_{(R,t)}M \) at the point \((R, t) \in M \) is equal to

\[
\{(R\Omega, t) : \Omega_{u} = -\Omega_{u}, \Omega \in \mathbb{R}^{d \times d}, \forall u \in [d - 1], t \in \mathbb{R}^d \}.
\]

Here, \( R\Omega = (R_1 \Omega_1, \ldots, R_{d-1} \Omega_{d-1}) \). The Riemannian metric \((\cdot, \cdot)_{(R,t)}\) on the product manifold \( M \), which gives the intrinsic distance between two elements \((R\Omega, \eta_t)\) and \((R\Omega', \eta'_t)\) of the tangent space at the point \((R, t)\) of the manifold \( M \), is defined in Equation (7).

\[
\langle (R\Omega, \eta_t), (R\Omega', \eta'_t) \rangle_{(R,t)} = \eta_t^T \eta'_t + \sum_{u=1}^{d-1} \text{trace}(\Omega_u^T \Omega_u').
\]

Let \( f : \mathbb{V} \times \mathbb{R}^d \rightarrow \mathbb{R} \) be a scalar function. Let the function \( f = f_{\mid M} \) be the restriction of the function \( f \) on the product manifold \( M \). Since the product manifold \( M \) is a submanifold of the Riemannian manifold \( \mathbb{V} \times \mathbb{R}^d \), the Riemannian gradient of the function \( f \) at the point \((R, t) \in M \) is obtained by projecting the Riemannian gradient of the function \( f \) at the point \((R, t) \in \mathbb{V} \times \mathbb{R}^d \) on the tangent space at the point \((R, t) \in M \). Therefore, the Riemannian gradient of the function \( f \) at the point \((R, t) \in M \) is defined in equation (8).

\[
\text{grad } f(R, t) = (P_{R}(\nabla_{Rf}), P_{t}(\nabla_{tf})) \in T_{(R,t)}M.
\]

Since the tangent space at a point in an Euclidean space is again an Euclidean space, the second component is given by \( P_{t}(\nabla_{tf}) = \nabla_{tf} \). The first component is defined as

\[
P_{R}(\nabla_{Rf}) = (P_{R_1}(\nabla_{R_1f}), \ldots, P_{R_{d-1}}(\nabla_{R_{d-1}f})).
\]

Here,

\[
P_{R_{i}}(\nabla_{R_{i}f}) = R_{j} \text{skew}(R_{j}^T \nabla_{R_{i}f}),
\]

where \( \text{skew}(A) = 0.5(A - A^T) \). We define \( \xi_{R_i}(R_j) = P_{R_j}(\nabla_{R_j f}) \). The Riemannian Hessian of the function \( f \) at a point \((R, t) \in M \) is a linear map. Hess \( f : T_{(R,t)}M \rightarrow T_{(R,t)}M \) and is defined as shown in equation (9).

\[
\text{Hess } f(R, t)[\eta_{R} \cdot \eta_{t}] = (P_{R}(D\xi_{R}(R)[\eta_{R}]), P_{t}(D\xi_{t}(t)[\eta_{t}])).
\]

Here, the first component \( P_{R}(D\xi_{R}(R)[\eta_{R}]) \) is equal to

\[
(P_{R_1}(D\xi_{R_1}(R_1)[\eta_{R_1}]), \ldots, P_{R_{d-1}}(D\xi_{R_{d-1}}(R_{d-1})[\eta_{R_{d-1}}])),
\]

where \( \eta_{R_i} = R_{j} \Omega_{j} \). The term

\[
D\xi_{x}(x)[\eta_{x}] = \lim_{t \to 0} \frac{\xi(x + t\eta_{x}) - \xi(x)}{t}
\]

is the classical derivative of the vector field \( \xi(x) \) in the direction \( \eta_{x} \).

**The Riemannian trust region method.** Our goal is to minimize the function \( f(R, t) \) over the product manifold \( M \). There exists a generalization of the popular optimization methods on the Riemannian manifolds. Since our problem is locally convex in each variable \( R_j \), which we prove in Theorem 6, we employ the Riemannian trust region approach [41]. It requires the Riemannian gradient and the Riemannian Hessian operator for the function \( f \),
which we find as follows. Let \( f \) be a function from the set \( \mathbb{V} \times \mathbb{R}^d \) to \( \mathbb{R} \) and defined as \( f(R, t) = \| \text{TE} \hat{t} \| \mathbb{R}^T (X - te^T) + te^T - XP \| \mathbb{F}^2 \). Its classical gradients with respect to both the variables are given in the equations (10) and (11). The detailed derivation is given in the appendices A1 and A2.

\[
\nabla_t f = 2 (I_d - \text{TE} \hat{t}^T ) ( \text{Xe}^T e - Xe - XPe ) .
\]

(10)

\[
\nabla_R, \hat{f} = -2 \left( \prod_{u=1}^{d-1} R_u \right) A \left( \prod_{u=1}^{d-1} R_u \right) E \left( \prod_{u=1}^{d-1} R_u \right) .
\]

(11)

Here,

\[
A = (XP - te^T)(X - te^T)^T + (X - te^T)(XP - te^T)^T
\]

which satisfies \( A^T = A \). Now let the function \( f = \hat{f} \mid_M \) be the restriction of the function \( \hat{f} \) on the set \( M \). We obtain the Riemannian gradient of the function \( f \) at a point \((R, t)\) by projecting the Riemannian gradient of the function \( \hat{f} \) over the tangent space \( T_{(R, t)} \) at the point \((R, t)\). Since the manifold \( \mathbb{V} \times \mathbb{R}^d \) is an Euclidean space, the Riemannian gradient of the function \( f \) is equal to its classical gradient. Therefore, we apply the definition given in equation (7) in order to find the Riemannian gradient \( \nabla f(r, t) \) of the function \( f \) which we denote as \( \xi_{R_j}, (R_1), \ldots, \xi_{R_{d-1}}, (R_d - 1), \xi_t \) and define in equations (12) and (13). The detailed derivation is given in the appendices A3 and A4.

\[
\xi_t (t) = 2(I_d - \text{TE} \hat{t}^T ) ( \text{Xe}^T e - Xe - XPe ) .
\]

(12)

\[
\xi_{R_j} (R_j) = -R_j \left( \prod_{u=1}^{j-1} R_u \right) A \left( \prod_{u=1}^{d-1} R_u \right) E \left( \prod_{u=1}^{d-1} R_u \right) + R_j \left( \prod_{u=j+1}^{d-1} R_u \right) E \left( \prod_{u=1}^{d-1} R_u \right) A \left( \prod_{u=1}^{j-1} R_u \right) .
\]

(13)

We determine the Riemannian Hessian of the function \( f \) using the definition given in equation (8). In order to determine the \( j \)th component \( \text{Hess}_R, (f(R, t), [R_j \Omega_j]) \) of the Riemannian Hessian, which is equal to \( \mathbb{E}(\xi_{R_j} (R_j) [R_j \Omega_j]) \), we first find the classical derivative \( \text{D} \xi_{R_j} (R_j) [R_j \Omega_j] \) of the Riemannian gradient \( \xi_{R_j} (R_j) \) in the direction \( R_j \) and then apply the projection operator \( \mathbb{P}_R \). Therefore, the \( j \)th component \( \text{Hess}_R, (f(R, t), [R_j \Omega_j]) \) of the Riemannian Hessian is equal to

\[
\frac{1}{2} R_j ([B_1, [R_j B_2] R_j, \Omega_j)] + ([\Omega_j, B_1, R_j B_2 R_j])] .
\]

(14)

The detailed derivation is given in appendix A5. Here \( [.,.] \) is the Lie bracket and defined as \( [U, V] = UV - VU \) for any two matrices \( U \) and \( V \),

\[
B_1 = \left( \prod_{u=j+1}^{d-1} R_u \right) E \left( \prod_{u=j+1}^{d-1} R_u \right) ^T
\]

and

\[
B_2 = \left( \prod_{u=1}^{j-1} R_u \right) A \left( \prod_{u=1}^{j-1} R_u \right) .
\]

In a similar way, we determine the component, \( \mathbb{E}(\xi_t (t) [\eta_t]) \), of the Riemannian Hessian which is shown in equation (15).

\[
\text{Hess}_t (f(R, t), [\eta_t]) = 4n (I_d - \text{TE} \hat{t}^T ) \eta_t .
\]

(15)

The detailed derivation is given in appendix A6. Now, we apply the Riemannian-trust-region method using the Riemannian gradient and Hessian defined in equations (12), (13), (14) and (15) in order to obtain the optimal solution. We use the manopt toolbox [22] in order to implement the optimization problem given in equation (5) for a fixed \( P \).

### 3.3 Optimizing Correspondences \( P \)

After obtaining the current estimate of the reflection transformation \( (R, t) \), we improve the correspondences matrix \( P \) by solving the problem given in equation (5) while fixing \( (R, t) \). We show that this sub-problem is equivalent to a linear assignment problem, where an assignment is a pair \((i, i')\) of reflection symmetry points.

**Claim 1:** The optimization problem given in equation (5) is a linear assignment problem in \( P \), for a fixed \( (R, t) \).

**Proof:** Let us consider the cost function in equation (5) and let \( X_m = \text{TE} \hat{t} \) (X - te^T) + te^T. We have

\[
\| X_m - X_p \|_F^2 = \text{trace}((X_m - XP)^T (X_m - XP))
\]

\[
= \text{trace}(X_m X_m - 2X_m X_P + X^T X_P P X_P^T).
\]

Since, the first and the third terms (using the fact that the permutation matrices are orthogonal) are not the functions of \( P \), the problem of finding the point of minimum of the function \( \| X_m - XP \|_F^2 \) is identical to the problem of finding the point of maximum of the function \( \text{trace}(X_m^T XP) \). Using the identity \( \text{trace}(A^T B) = \text{vec}(A)^T \text{vec}(B) \), we have that \( \text{trace}(X_m^T XP) = \text{vec}(X^T X_m) \text{vec}(P) \), where the operator \( \text{vec} \) vectorizes a matrix by stacking all the columns successively in a column vector.

Therefore, for a fixed reflection transformation, the problem defined in equation (5) is equivalent to the problem defined in equation (16).

\[
\text{P} \in \{0, 1\}^{n \times n} \text{vec}(X_m X_m^T)^T \text{vec}(P)
\]

subject to \( \text{Pe} \leq e \), \( P^T e \leq e \),

(16)

which is a standard linear assignment problem.

**Claim 2:** The problem defined in Equation (16) is an integer linear program.

**Proof:** Let \( v_1 \) be a vector of size \( n^2 \times 1 \) with the first \( n \) coordinates equal to one and the last \( n(n - 1) \) coordinates equal to zero. Let \( e_1 \) be a vector of size \( n \times 1 \) with all the coordinates equal to zero except the first coordinate which is equal to one. Let \( v_2 = [e_1^T e_1^T \ldots e_1^T] \) be a vector of size \( n^2 \times 1 \). Now let us construct the matrices \( A_1 \) and \( A_2 \), each of size \( n \times n^2 \), such that the \( i \)th row of the matrix \( A_1 \) is equal to the row vector \( cs(v_1^T, n(i - 1)) \) and the \( i \)th row of the matrix \( A_2 \) is equal to the row vector \( cs(v_2^T, i - 1) \). Here \( cs(v^T, i) \) is a row vector obtained by circularly shifting any row vector \( v^T \) right by \( i \) coordinates.

Now, it is trivial to verify that the constraint \( P^T e \leq e \) is equivalent to \( A_1 \text{vec}(P) \leq e \) and the constraint \( \text{Pe} \leq e \) is equivalent to \( A_2 \text{vec}(P) \leq e \). Therefore, the problem defined in equation (16) is equivalent to the problem defined in equation (17).

\[
\text{max} \text{vec}(X_m X_m^T)^T a
\]

subject to \( A_1^T A_2^T a \leq [e^T e^T]^T \)

(17)
which is an integer linear program with $a = \text{vec}(P)$.

### 3.4 Convergence Analysis

We derive the essential results in order to prove that the alternating optimization framework converges.

**Theorem 3.** The cost function $f(R, t, P)$ is convex in the variable $t$.

**Proof.** In order to prove this Theorem, we prove that the Riemannian Hessian of the function $f$ with respect to the variable $t$ is a positive semi-definite (PSD) matrix. Let $\eta_k = [\eta_1 \eta_2 \ldots \eta_d]^T \in \mathbb{R}^d$, then using the definition of Riemannian metric we have

$$\langle \eta_k, \text{Hess}_t(f)\eta_k \rangle_t = \eta_k^T \text{Hess}_t(f)\eta_k.$$  

Now using the Riemannian Hessian $\text{Hess}_t(f)\eta_k$ defined in equation (15), we have that

$$\eta_k^T \text{Hess}_t(f)\eta_k = \eta_k^T \eta_k - (T^T \eta_k)^T E (T^T \eta_k) = \|\eta_k\|^2 - \sum_{u=1}^{d-1} q_u + q_d^2 = \|\eta_k\|^2 - \|q\|^2 + 2q_d^2.$$  

Here, $q = T^T \eta_k$. Now we observe that

$$\|q\|^2 = q^T q = \eta_k^T T^T T \eta_k = \eta_k^T \eta_k = \|\eta_k\|^2.$$  

Therefore

$$\|\eta_k\|^2 - \|q\|^2 = 0 \Rightarrow \eta_k^T \text{Hess}_t(f)\eta_k = 2q_d^2 \geq 0.$$  

**Theorem 4.** At the critical point, the matrix $T^* = \prod_{u=1}^{d-1} R_u^*$ contains the eigenvectors of the matrix $A$ as columns.

**Proof.** At the critical point, the Riemannian gradient given in equation (13) vanishes. Therefore, $\xi_{R_u}(R_j) = \partial_x d. \$ Now premultiplying it with $(\prod_{u=1}^{j} R_u)R_j^T$ and then post-multiplying with $(\prod_{u=j+1}^{d-1} R_u)$ we achieve

$$AT^*E = T^*E(T^* )^T AT^* = (T^* )^T AT^* E = (T^* )^T AT^*.$$  

Now let $Q = \begin{bmatrix} Q_1 & q_2 \\ q_3^T & q_4^T \end{bmatrix} = (T^* )^T AT^* $ be a matrix. Then we have $QE = EQ$. Therefore,

$$\begin{bmatrix} Q_1 & q_2 \\ q_3^T & q_4^T \end{bmatrix} \begin{bmatrix} I_{d-1} & 0_d-1 \\ 0_{d-1} & -1 \end{bmatrix} = \begin{bmatrix} I_{d-1} & 0_d-1 \\ 0_{d-1} & -1 \end{bmatrix} \begin{bmatrix} Q_1 & q_2 \\ q_3^T & q_4^T \end{bmatrix} \Rightarrow q_3 = 0_{d-1}, q_3 = 0_{d-1}, Q_1I_{d-1} = I_{d-1}Q_1.$$  

Since $I_{d-1}$ is a diagonal matrix and the equality $Q_1I_{d-1} = I_{d-1}Q_1$ holds true, it is easy to prove that $Q_1$ is a diagonal matrix. Therefore, the matrix $Q$ is also diagonal. The spectral theorem states that every real symmetric matrix has eigenvalue decomposition with real eigenvalues and orthogonal eigenvectors. Here, we have observed that the matrix $A$ is a real symmetric matrix and satisfies $Q = (T^* )^T AT^* $, where the matrix $Q$ is a diagonal matrix and the matrix $T^* $ is an orthogonal matrix. Therefore, the matrix $T^* $ is the matrix containing the eigenvectors of the matrix $A$. In Theorem 5, we prove that the order of stacking eigenvectors of $A$ as columns of $T^*$ affects the convexity of the problem.

**Theorem 5.** The cost function $f(R, t, P)$ is locally convex in each rotation matrix $R_j$.

**Proof.** In order to show the local convexity in $R_j$, we have to show that the value $(R_j \Omega_j, \text{H}[R_j \Omega_j])_{R_j} \geq 0$ in the neighborhood of the optimal angle $\theta^*_j$. Here, $\text{H}[R_j \Omega_j] = \text{Hess}_{R_j}(f(R, t))[R_j \Omega_j]$. By using the Riemannian metric defined in equation (6), we have

$$(R_j \Omega_j, \text{H}[R_j \Omega_j])_{R_j} = \text{trace}(\Omega_j^T R_j^T \text{H}[R_j \Omega_j]).$$  

By using equation (13) the matrix $R_j^T \text{H}[R_j \Omega_j]$ is equal to

$$0.5[B_1, [R_j^T B_2 R_j, [R_j^T B_2 R_j, [R_j^T B_2 R_j, B_1]]].$$  

In appendix A7, we show that the trace$(\Omega_j^T R_j^T \text{H}[R_j \Omega_j])$ is equal to

$$4 \times \text{trace}(R_j B_2 R_j (\Omega_j^T B_1 \Omega_j - \Omega_j \Omega_j B_1)).$$ (18)  

We visualize this term for $d = 2$. For $d = 2$, the matrix $\Omega = \begin{bmatrix} 0 & -\theta \\ \theta & 0 \end{bmatrix}, E = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, and let $A = \begin{bmatrix} a_1 & a_2 \\ a_2 & a_3 \end{bmatrix}$ and $R = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix}$ we have that

$$(R \Omega, \text{H}[R \Omega])_R = 8a_2 \theta^2 \sin(\theta) + 4\theta^2 \cos(\theta)(a_1 - a_3).$$  

In Fig. 3 we plot the value $(R \Omega, \text{H}[R \Omega])_R$ against the initialization angle $\theta$ for six reflection symmetry patterns having different orientations for symmetry axis. We observe that the PSD values are positive in the proximity of the optimal angles. Therefore, it is locally convex. We further observe that this quantity is maximum at the optimal angle. We also observe that, if $\theta$ is the symmetry axis orientation, then the PSD value becomes positive in the proximity of $\theta$ and of $\theta + 180^\circ$. The reason for the second range is that, if $\theta$ is the slope of a line, then $\theta + 180^\circ$ is also the slope of the same line.

In Theorem 4, we claimed that the order in which the eigenvectors are stacked as columns of the matrix $R$ affects the local convexity. We prove it as follows. At the critical point we have that $R^T A R = \text{diag}(d_1, d_2)$. We note that $\Omega_j B_1 \Omega_j - \Omega_j \Omega_j B_1 = E$ for $d = 2$. Now from equation (18), we achieve

$$\langle R \Omega, \text{H}[R \Omega] \rangle_R = d_1 - d_2 \Rightarrow d_1 \geq d_2.$$  

Therefore, the first column of the matrix $R^*$ should be the eigenvector corresponding to the maximum eigenvalue and the second column of the matrix $R^*$ should be the eigenvector corresponding to the minimum eigenvalue of the matrix $A$.  

![Fig. 3. Illustration of the local convexity. The value $(R \Omega, \text{H}[R \Omega])_R$ against the initialization angle $\theta$ for 6 reflection symmetry patterns having different orientations, $\{0^\circ, 20^\circ, 40^\circ, 60^\circ, 80^\circ, 100^\circ\}$ for symmetry axis. The PSD value (divided by $\theta^2$) is positive in the proximity of the optimal angle.](image-url)
Theorem 6. The alternating framework converges to the global minimum if the initialization of the rotation matrices $(R_1, \ldots, R_{d-1})$ are within the proximity of the optimal rotation matrices and initialization of the translation $t$ is any random vector.

Proof. We observe that the alternating framework is basically the block coordinate descent (BCD) method, where $(R_1, \ldots, R_{d-1}, t)$ and $P$ are two blocks of coordinates. According to [43], the BCD method converges if the cost function is convex in each block of coordinates. Here, we have seen that the cost function is convex in the coordinates $t$ (Theorem 3), convex in the coordinates $P$ on the relaxed domain $[0, 1]^{n \times n}$, and locally convex in the coordinates $(R_1, \ldots, R_{d-1})$ (Theorem 5). This implies that if the initialization of $(R_1, \ldots, R_{d-1})$ is within the proximity of the optimal solution, then the alternating framework converges to the global minimum. We experimentally show this theorem for the case $d = 2$. In Fig. 4 we plot the error (averaged over all optimal angles) at the convergence point against the initialization angles for the case $d = 2$ (we shift the error vectors for different optimal angles so that the optimal angle is always 90°). We observe that the variance becomes zero for initialization angle $\theta_0 \in (90° - 12°, 90° + 9°)$ and $\theta_0 \in (270° - 12°, 270° + 9°)$. The reason for the second range is that, if $\theta$ is the slope of a line, then $\theta + 180°$ is also the slope of the same line.

![Fig. 4. We plot the error at the convergence point against the initialization angles for the case $d = 2$ (we shift the error vectors for different optimal angles so that the optimal angle is always 90°). We observe that the variance becomes zero for initialization angle $\theta_0 \in (90° - 12°, 90° + 9°)$ and $\theta_0 \in (270° - 12°, 270° + 9°)$.](image)

In summary, in order to obtain the optimal $(R^*, t^*, P^*)$, we follow the Algorithm 1.

**Initialization Strategy:** We have shown experimentally that if the initialization angle $\theta_0 \in (\theta^* - 12°, \theta^* + 9°)$, then the Algorithm 1 converges to the global minimum. Hence, we split the range $[-90°, 90°]$ into 10 bins. We randomly choose a bin and randomly choose an angle in this bin and run the Algorithm 1 till convergence. If this solution satisfies Theorems 3, 4, and 5, we accept it else we consider the next bin. Theorem 6 guarantees that it does not take more than $\sim 10$ bins to find the global optimum.

4 Evaluation and Results

We compare our method with [7]. In [7], the authors detect pairs of reflection symmetry points by a randomized algorithm as follows. They start by randomly selecting two pairs $(x_i, x_j)$ and $(x_i', x_j')$ which are symmetrically consistent, that is the condition $(||x_i - x_j||_2 - ||x_i' - x_j'||_2 < \epsilon_d) \land (||x_i - x_j||_2 - ||x_i' - x_j'||_2 < \epsilon_d)$ is true. If the number of pairs from the remaining pairs, which are symmetrically consistent with both the pairs, is above some threshold (0.003% of all possible pairs), they keep all such pairs. The same procedure is repeated for a new pair of symmetrically consistent pairs and if the number of pairs which are symmetrically consistent with the new pair, is above the threshold, then they union these pairs with those of the previous step. They repeat until the number of selected pairs is above some threshold (0.1% of all possible pairs).

We observe that the choice of the threshold $\epsilon_d$ is totally dependent on how much perturbation is present in the reflection symmetry pattern. If the perturbation is different for each point in the set, then the problem of choosing $\epsilon_d$ becomes more hard and it may be a computationally complex process. We experimentally show that our method is robust to significant perturbations even for the case where the perturbation is different for each point in the set.

4.1 Dataset preparation

There does not exist any dataset containing sets of points with volumetric distribution and with ground truth mirror symmetric correspondences. Therefore, we synthetically create such a dataset. We compare proposed approach with [7] for the cases $d = 2$ and $d = 3$. Let $\{x_1, x_2, \ldots, x_{2g}\}$ be the randomly chosen $2g$ points. Given the reflection $\{R_1, \ldots, R_{d-1}, t\}$, we reflect these points using the Definition 1 in order to get the set $\hat{S} = \{x_1, x_2, \ldots, x_{2g}, x_1, x_2, \ldots, x_{2g}\}$. Then we perturb each point with random noise as $x \leftarrow x + N(0, d, diag(\sigma^2, \sigma^2, \ldots, \sigma^2)) \forall x \in \hat{S}$. Here, $\sigma^2$ is the perturbation radius and we note that the perturbation is different for each point. For the case $d = 2$, we create sets containing reflection symmetry patterns with $n = 50, 100, 150, 200, 250, 300$ with $0 \leq x, y \leq 1$. For each $n$, we take 19 different symmetry axis orientations in the range from $-90°$ to $90°$ with step size of $10°$. We choose $\sigma^2 = 0, 0.01, 0.02, \ldots, 0.1$ to get 11 different perturbations. In total, we have 1254 sets for the evaluation. For the case $d = 3$, we create sets containing reflection symmetry patterns with $n = 50, 100, 150, 200, 250, 300$ with $0 \leq x, y \leq 1$. For each $n$, we take 16 different symmetry plane orientations by considering $\theta_1 \in \{-30°, 0°, 35°, 80°\}$ and $\theta_2 \in \{-30°, 0°, 35°, 80°\}$. We choose $\sigma^2 = 0, 0.01, 0.02, \ldots, 0.1$. In total, we have 1056 point sets.

4.2 Evaluation Methods

(A) Correspondences rate. Let $(i, i')$ be an estimated correspondence and let $(i, i')$ be the ground-truth correspondence. Then

**Algorithm 1**

1. Input: Set of points $\mathcal{S} = \{x_i\}_{i=1}^n$.
2. Initialize angels $\theta_0$ and translation $t$.
3. Solve the ILP defined in equation (17) for $P$.
4. For this $P$ solve for $(R, t)$ using the Riemannian-trust-region method using the Riemannian gradient and Hessian defined in equations (12), (13), (14), and (15).
5. Keep iterating steps 3 and 4 till convergence.
6. Output: The optimal $R_1^*, R_2^*, \ldots, R_{d-1}^*$ and $t^*$.
we decide if the correspondence \((i, i'_g)\) is correct based on a distance threshold \(\tau\). If the distance \(|x_{i}-x_{i'}|\) between the points \(x_{i}\) and \(x_{i'}\) is less than the distance threshold \(\tau\), then the correspondence \((i, i'_g)\) is correct and otherwise, incorrect. For a given threshold \(\tau\), we count the correspondences \((i, i'_g)\) for which the condition \(|x_{i}-x_{i'}|\) holds true. In Fig. 5, we show the correspondences rate against the distance threshold \(\tau\). We choose the same set of point sets as created in the [4.1]. We observe that for the zero perturbation \(\sigma^2 = 0\), the correspondences rate is 100% for \(d = 3\) and nearly 100% for \(d = 2\) for all the distance thresholds for the proposed approach. For the method [7], it remains around 85%. As we increase the perturbation radius, the correspondences rate decrease for the method [7] more rapidly than the proposed approach. This is due to the fact that, if we choose low \(\epsilon_d\) (such as 0.001) it becomes computationally complex and for high values \(\epsilon_d\) (such as 0.06) it return many false positive correspondences.

(B) Precision and recall rates. Let \(P_{\bar{g}}, P_{\bar{e}}\) be the correspondences matrices for the cases: ground truth and estimated \([\epsilon_d]\) or the proposed approach), respectively. Then the number of true positive correspondences is \(|P_{\bar{g}} \cap P_{\bar{e}}|\), true negative correspondences is \(|P_{\bar{g}} \cap P_{\bar{e}}|\), false positive correspondences is \(|\neg P_{\bar{g}} \cap P_{\bar{e}}|\), and false negative correspondences is \(|P_{\bar{g}} \cap \neg P_{\bar{e}}|\). Here, \(|\ .\ |\) denote returns the sum of all elements of the argument matrix. We average the precision and recall values over all the point sets. In the Fig. 6 and Fig. 7 we show the precision and recall values against the perturbation radius \(\sigma^2\) for the method Xu et al. [7] and the proposed method for the case \(d = 2\), and \(d = 3\), respectively.
We observe that the precision and recall values for the method Xu et al. [7] start decreasing as the perturbation radius start increasing. However, for our method, the precision and recall values remain comparatively high. We also observe that for $\sigma^2 = 0$, we achieve 100% accuracy. The precision and the recall values remain same for our method. The reason is that while doing the experiment, we choose the even number of points in each set. Therefore, the number of pairs returned by our method is equal to half of the number points in the set. Therefore, the number of false positive pairs is equal to the number of false negative pairs. Whereas, for the method in [7], the number of pairs returned may not be equal to the half of the number of points in the set. The error bars represent the variance in the precision and recall for a fixed perturbation radius over all the sets.

In Figure 8 we present the results of the method [7] and the
proposed approach for the case \( d = 3 \). We choose \( n = 300 \) and \( \sigma^2 = 0, 0.01, 0.03, 0.05 \). Plots in even columns represent the correspondences matrix \( \mathbf{P} \). The green pixels represent the true positive correspondences, yellow pixels represent the false negative correspondences, and the red pixels represent the false negative correspondences. For better visualization of the correspondences between the mirror symmetric points, we generate the set \( \mathcal{S} \) such that \( y^b \) and \( (i + \frac{1}{2}) \mathbf{y}^b \) points are mirror reflections of each other. Similarly, in Figure 9 we present the results of the method [31] and the proposed approach for different perturbations to a perfect reflection symmetry pattern containing 200 points with \( 0 \leq x, y \leq 1 \). We choose \( \sigma^2 = 0, 0.04 \). We observe that our method is able to detect symmetry with significant accuracy even when \( \sigma^2 = 0.04 \).

### 4.3 Qualitative analysis

In order to measure the qualitative performance of the proposed approach, we investigate the following two errors which are functions of the perturbation radius \( \sigma^2 \):

\[
ed = \frac{1}{n} \sum_{i=1}^{n} | \langle \hat{z}_i, \hat{v} \rangle | \quad \text{and} \quad \em = \frac{1}{n} \sum_{i=1}^{n} | \hat{v}^T x_i^m + w_0 |
\]

The error \( e_d \) represents how well the vectors, along the line segments joining the estimated reflection symmetry points, align with the normal to the hyperplane \( \pi \) at convergence. The error \( e_m \) represents how well the mid-points of line segments joining reflection symmetry points lie on the estimated hyperplane \( \pi \). Here, \( \hat{z}_i = \frac{x_i - \bar{x}_i}{|x_i - \bar{x}_i|^2} \), \( \hat{v} \) is the unit normal to the hyperplane \( \pi \), \( x_i^m = \frac{x_i + \bar{x}_i}{2} \), and \( w_0 \) is the distance of the hyperplane \( \pi \) from the origin. In Figure 10 we show the errors \( e_d \) and \( e_m \) against the perturbation radius \( \sigma^2 \). We observe that the values \( e_m \) and \( e_d \) for the proposed approach are close to that of the ground-truth reflection symmetry even as the value of \( \sigma^2 \) increases. We do not report these errors for the method [31], since it does not determine the reflection symmetry hyperplane. The average computation time for \( n = 500 \) using MATLAB on a machine with 16 GB RAM and 3.4 GHz processor, is \( \sim 413 \) seconds with the proposed initialization scheme.

### 4.4 Results

In Fig. 11 and Fig. 12, we present the results for the cases \( d = 2, 3 \). For \( d = 2 \), we detect reflection symmetry in the set of corner points in a real image. In order to determine the symmetry axis, we use Theorem 2. For \( d = 3 \), we use exiting 3D models dataset [46]. In order to calculate the symmetry axis in an image using the proposed approach, we first find the set of corner points [47]. This set may contain the corners not lying on the symmetric object, therefore we apply the proposed approach with RANSAC [48]. We compare the proposed results with results of two descriptor based methods [31] and [38]. We evaluate on real and synthetic images containing single symmetric object from the dataset [49]. In Table 1 we present the precision and the recall rates. We observe that, for synthetic images the precision rate is very high for the proposed approach due to the fact that most of the corner points lie on the symmetric object. Whereas, in real images the set of corner points contains many outlier corners which leads to the degraded performance. Precision rates for the proposed approach are higher than that for the methods [31] and [38]. The recall rates are better than that of the method [31] and comparable to that of the method [38]. This leads to the conclusion that symmetry detection can be performed independent of feature descriptors.

### 5 Conclusion

In this work, we have developed the theory for establishing the correspondences between the points in \( \mathbb{R}^d \) which are reflections of each other. We, further, determine the reflection symmetry transformation in a volumetric set of points in \( \mathbb{R}^d \) containing a perturbed reflection symmetry pattern using optimization on Riemannian manifold. We have shown that our method is robust to a significant amount of perturbation and achieves 100% accuracy for no perturbation. We have further shown the significance of this work by detecting reflection symmetry in real images and comparing with state-of-the-art methods. The proposed approach is particularly suitable for detecting reflection symmetry of objects in applications where obtaining a robust local descriptor is challenging due to the discrete nature of the obtained data. Our method works for the cases where a single pattern is present in the set as the linear reflection assignment problem is a time consuming step for a large point set.

We believe that the fundamental theory and algorithm developed in this work will pave way for computer vision researchers to exploit them for scenarios where estimating feature descriptors is extremely challenging. The algorithm can be extended to other relevant problems in computer vision which rely on the processing of a discrete point set residing in a high dimensional space. As a future work, we would like to extend this work to detect the reflection symmetry in large volumetric point sets containing multiple reflection symmetry patterns.

### APPENDIX

#### A1. Euclidean gradient of the function \( \tilde{f} \) with respect to the variable \( t \) (Equation (10))

We write the cost function as follows.

\[
\tilde{f}(\mathbf{R}, t, \mathbf{P}) = \| \text{TET}^T (\mathbf{X} - \mathbf{te}^T) - (\mathbf{X}P - \mathbf{te}^T) \|^2 \\
= \| \langle \text{TET}^T \mathbf{X} - \mathbf{XP}, (\mathbf{I}_d - \text{TET}^T)\mathbf{te}^T \rangle \|_F^2.
\]

We note that

\[
(\mathbf{I}_d - \text{TET}^T)^T (\mathbf{I}_d - \text{TET}^T) = 2(\mathbf{I}_d - \text{TET}^T).
\]

Therefore, we have (the terms which are not functions of \( t \) are not shown)

\[
\tilde{f}(\mathbf{R}, t, \mathbf{P}) = \text{trace}(2\mathbf{te}^T (\mathbf{I}_d - \text{TET}^T)\mathbf{te}^T) + 2(\mathbf{X}^T (\mathbf{I}_d - \text{TET}^T)\mathbf{te}^T).
\]

Now taking the derivative with respect to \( t \) we have,

\[
\nabla_t \tilde{f} = 2(\mathbf{I}_d - \text{TET}^T)\mathbf{te}^T e + 2(e^T \mathbf{te}^T (\mathbf{I}_d - \text{TET}^T))^T.
\]
Fig. 11. Results of symmetry detection in real images from the dataset [44, 45]. We show the set $S$ using green points, the reflection symmetry axis by a red line, and the correspondences between the mirror symmetric points by the blue lines.

Fig. 12. Results of symmetry detection in the 3D object models from the dataset [46]. The correspondences are shown by joining the mirror symmetric points by the black colored lines. The Reflection symmetry plane is shown in light brown color. The mid-points of the mirror symmetric points are show in blue color. We do not provide precision values as the ground-truth symmetry is unavailable for in this dataset.
+ 2(e^T (X^T TET^T - P^T X^T)(I_d - TET^T))e^T

= 4(I_d - TET^T)te^T + 2(I_d - TET^T)(TET^T X - XP)e^T

Here we have that (I_d - TET^T)TET^T = -(I_d - TET^T).

Therefore,

\nabla_t f = 4(I_d - TET^T)te^T e - 2(I_d - TET^T)(X + XP)e^T

= 2(I_d - TET^T)(2te^T - Xe - XP)e^T.

A2. Euclidean gradient of the function \( f \) with respect to the variable \( R_j \) (Equation [11])

Let us consider the cost function as defined in equation (5):

\[ R = \text{TET}^T U - V \]

Then the cost function becomes.

\[ f = \text{TET}^TI_d^T \text{TET}^T = -(I_d - TET^T). \]

Now, let us define \( T = \prod_{u=1}^{d-1} R_u \). We have that \( U = X - te^T \) and \( V = XP - te^T \).

Then the cost function becomes.

\[ f(R, t, P) = \| TET^T U - V \|_F^2. \]

Now, let us define \( T = \prod_{u=1}^{d-1} R_u \). We have that \( U = X - te^T \) and \( V = XP - te^T \).

Then the cost function becomes.

\[ f(R, t, P) = \| TET^T U - V \|_F^2. \]

Next, we determine the Riemannian Hessian of the function \( f \) with respect to the variable \( R_j \) (Equation [13]).

Using the definition, as defined in main paper, of Riemannian gradient \( \xi_t(t) \) of the function \( f \) with respect to the variable \( t \) we have

\[ \xi_t(t) = \mathbb{P}_f(\nabla_t f) = \nabla_t f. \]

A3. The Riemannian gradient of the function \( f \) with respect to the variable \( t \) (Equation [12])

Using the definition, as defined in main paper, of Riemannian gradient \( \xi_t(t) \) of the function \( f \) with respect to the variable \( t \) on the right-hand side we have

\[ \xi_t(t) = \mathbb{P}_f(\nabla_t f) = \nabla_t f. \]

A4. The Riemannian gradient of the function \( f \) with respect to the variable \( R_j \) (Equation [13])

Using the definition, as defined in main paper, of Riemannian gradient \( \xi_{R_j}(R_j) \) of the function \( f \) with respect to the variable \( R_j \) we have

\[ \xi_{R_j}(R_j) = \mathbb{P}_{\mathbb{T}}(\nabla_{R_j} f) = R_j \text{ skew}(R_j \nabla_{R_j} f). \]

\[ \xi_{R_j}(R_j) = R_j \text{ skew}(R_j \nabla_{R_j} f). \]

\[ \nabla_{R_j} f \nabla_{R_j} R_j = -2 \left( \prod_{u=1}^{d-1} R_u \right) A \left( \prod_{u=1}^{d-1} R_u \right) \left( \prod_{u=j+1}^{d-1} R_u \right)^T \]

\[ + R_j \left( \prod_{u=1}^{d-1} R_u \right) A \left( \prod_{u=1}^{d-1} R_u \right) \left( \prod_{u=j+1}^{d-1} R_u \right)^T. \]

Therefore,

\[ \xi_{R_j}(R_j) = \frac{R_j \nabla_{R_j} f - \nabla_{R_j} f \nabla_{R_j} R_j}{2}. \]

A5. The Riemannian Hessian of the function \( f \) with respect to \( R_j \) (Equation [14])

Next, we determine the Riemannian Hessian of the function \( f \). In order to determine the \( j \)-th component \( \text{Hess}_{R_j}(f(R, t)) \), we have

\[ \text{Hess}_{R_j}(f(R, t)) = \mathbb{P}_R(\text{D} \xi_{R_j}(R_j)(R_j \Omega_j)), \]

of the Riemannian Hessian, we first find the classical derivative \( \text{D} \xi_{R_j}(R_j)(R_j \Omega_j) \) of the Riemannian gradient \( \xi_{R_j}(R_j) \) in the direction \( R_j \Omega_j \) and then we apply the projection operator \( \mathbb{P}_R \). Now using Equation [20] we have

\[ \xi_{R_j}(R_j) = -R_j \left( \prod_{u=1}^{d-1} R_u \right) A \left( \prod_{u=1}^{d-1} R_u \right) \left( \prod_{u=j+1}^{d-1} R_u \right)^T \]

\[ + R_j \left( \prod_{u=1}^{d-1} R_u \right) A \left( \prod_{u=1}^{d-1} R_u \right) \left( \prod_{u=j+1}^{d-1} R_u \right)^T. \]
\[
-R_jR_j^\top \left( \prod_{u=1}^{j-1} R_u \right)^\top A \left( \prod_{u=1}^{j-1} R_u \right) R_j \left( \prod_{u=1}^{j-1} R_u \right) E \left( \prod_{u=1}^{d-1} R_u \right) ^\top j+1
\]
\[
+R_j \left( \prod_{u=j+1}^{d-1} R_u \right) E \left( \prod_{u=1}^{d-1} R_u \right) R_j \left( \prod_{u=1}^{d-1} R_u \right) A \left( \prod_{u=1}^{d-1} R_u \right) ^\top j+1
\]
\[
= -R_jR_j^\top B_2R_jB_1 + R_jB_1R_j^\top B_2R_j.
\]

Here, 
\[
B_1 = \left( \prod_{u=j+1}^{d-1} R_u \right) E \left( \prod_{u=1}^{d-1} R_u \right) ^\top j+1
\]
\[
B_2 = \left( \prod_{u=1}^{d-1} R_u \right) ^\top A \left( \prod_{u=1}^{d-1} R_u \right)
\]

Now
\[
\mathcal{D} \mathbf{R}_j \left( R_j R_j^\top \Omega R_j \right) = \frac{d}{dt} \mathcal{D} \mathbf{R}_j \left( R_j + t R_j \Omega R_j \right) |_{t=0}
\]
\[
= \frac{d}{dt} \left( (R_j + t R_j \Omega R_j) \left( R_j + t R_j \Omega R_j \right) ^\top B_2 (R_j + t R_j \Omega R_j) B_1 \right) |_{t=0}
\]
\[
+ \frac{d}{dt} \left( B_2 (R_j + t R_j \Omega R_j) \left( R_j + t R_j \Omega R_j \right) ^\top B_2 (R_j + t R_j \Omega R_j) \right) |_{t=0}.
\]

The first term is equal to 
\[
-R_jR_j^\top B_2 R_j \Omega R_j B_1.
\]

The second term is equal to 
\[
-R_j R_j^\top B_2 R_j \Omega R_j B_1 + (R_j B_1 R_j^\top B_2 R_j \Omega R_j B_1)
\]

Therefore,
\[
\mathcal{D} \mathbf{R}_j \left( R_j R_j^\top \Omega R_j \right) = \mathcal{D} \mathbf{R}_j \left( R_j R_j^\top \Omega R_j \right) - \mathcal{D} \mathbf{R}_j \left( R_j R_j^\top \Omega R_j \right) B_1 + (R_j B_1 R_j^\top B_2 R_j \Omega R_j B_1)
\]

A6. The Riemannian Hessian of the function \( f \) with respect to \( t \) (Equation 15)

In a similar way, we determine the second component, \( \mathbb{R}^d \), of the Riemannian Hessian. Since \( \mathbb{R}^d \) is a vector space we have \( \mathbb{R}^d = \mathbb{R}^d \)

\[
\mathcal{D} \mathbf{R}^\top \left( t \right) \| \eta_t \| = \frac{d}{dt} \mathcal{D} \mathbf{R}^\top \left( t + q \eta_t \right) |_{q=0}
\]
\[
= 4 n \left( I_d - \left( \prod_{u=1}^{d-1} R_u \right) E \left( \prod_{u=1}^{d-1} R_u \right) ^\top \right) \eta_t.
\]

Therefore
\[
\mathcal{H} \mathbf{R}^\top \left( t \right) \| \eta_t \| = 4 n \left( I_d - \left( \prod_{u=1}^{d-1} R_u \right) E \left( \prod_{u=1}^{d-1} R_u \right) ^\top \right) \eta_t.
\]

(21)

A7. Steps of Theorem 5

Showing the fact
\[
\text{trace}(\Omega_j R_j^\top H[R_j, \Omega_j]) = 4 \text{trace}(R_j^\top B_2 R_j \Omega_j (B_1 \Omega_j - \Omega_j B_1)).
\]

\[
\text{trace}(\Omega_j R_j^\top H[R_j \Omega_j]) = \text{trace}(\Omega_j R_j^\top R_j R_j^\top R_j \Omega_j - \Omega_j R_j R_j^\top R_j \Omega_j B_1 + (R_j B_1 R_j^\top R_j R_j^\top R_j \Omega_j B_1)
\]

\[
= \text{trace}(\Omega_j R_j^\top R_j R_j^\top R_j \Omega_j - \Omega_j R_j R_j^\top R_j \Omega_j B_1 + (R_j B_1 R_j^\top R_j R_j^\top R_j \Omega_j B_1)
\]

\[
= \text{trace}(\Omega_j R_j^\top R_j R_j^\top R_j \Omega_j - \Omega_j R_j R_j^\top R_j \Omega_j B_1 + (R_j B_1 R_j^\top R_j R_j^\top R_j \Omega_j B_1)
\]

Since,
\[
\text{trace}(R_j^\top B_2 R_j B_1 \Omega_j B_1) = \text{trace}(B_2 R_j B_1 \Omega_j B_1)
\]

\[
= \text{trace}(B_2 R_j B_1 \Omega_j B_1)
\]

we have
\[
\text{trace}(\Omega_j R_j^\top H[R_j \Omega_j]) = 4 \text{trace}(R_j^\top B_2 R_j \Omega_j B_1)
\]

\[
= 4 \text{trace}(R_j^\top B_2 R_j \Omega_j B_1).
\]
[50] K. B. Petersen, M. S. Pedersen et al., “The matrix cookbook,” *Technical University of Denmark*, vol. 7, p. 15, 2008.