Robust affine point matching via quadratic assignment on Grassmannians

Alexander Kolpakov, Michael Werman

Abstract

Robust Affine Matching with Grassmannians (RoAM) is a new algorithm to perform affine registration of point clouds. The algorithm is based on minimizing the Frobenius distance between two elements of the Grassmannian. For this purpose, an indefinite relaxation of the Quadratic Assignment Problem (QAP) is used, and several approaches to affine feature matching are studied and compared. Experiments demonstrate that RoAM is more robust to noise and point discrepancy than previous methods.

Keywords: Shape matching, point cloud registration, affine correspondence, Grassmann manifold, singular value decomposition (SVD), affine feature matching, quadratic assignment problem (QAP).

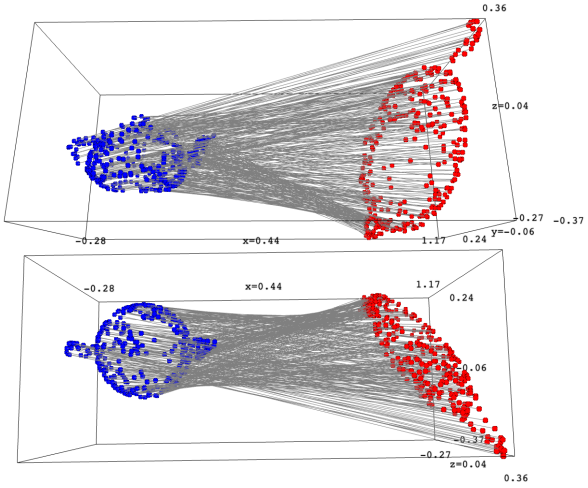

Figure 1: "Teapot" cloud: the initial point cloud X (blue) and its image Y (red) under a linear transformation L with $\operatorname{cond} L = 4.0$. The feature matching is shown: side view (top) and top view (bottom).

1. Introduction

Affine registration has a long history in computer vision literature, and extensive work was carried out for affine registration in two and three dimensions, [1, 2, 3, 4, 5]. In [6], two interesting matching problems are formulated and solved based on affine registration in higher dimensions; stereo correspondence under motion and image set matching.

This paper is related to [7] which proposed a two-stage procedure to recover correspondences between unlabelled, affinely transformed point sets. The first step finds the affine transformation, and the second is the matching between the point sets. We adopt their first step of using the Grassmannian, which is invariant under linear (affine) transformations of the point sets.

As noted in [8], it is provably better, in certain cases, to use an *indefinite relaxation* of the Quadratic Assignment Problem (QAP), which we use to find the matching, while the common convex relaxation almost always fails.

Based on these theoretical findings, we propose a relaxed quadratic programming method to identify the correct permutation (match) between the sets of points. The experimental outcomes support our notion that the new algorithm is more resistant to noise and point discrepancies than the approach presented in [7].

2. Algorithm RoAM

The algorithm is based on two main ideas, first affine invariant representations, Grassmanians, are computed for the point clouds and then a relaxation of a hard computational problem (QAP) is used to find the matching between the representative Grassmanians.

Email addresses: kolpakov.alexander@gmail.com (Alexander Kolpakov), michael.werman@mail.huji.ac.il (Michael Werman)

Let $X, Y \subset \mathbb{R}^{d\times n}$ be two n point clouds in $\mathbb{R}^d$, such that $Y = L\,X\,S + t$ for a linear transformation $L \in \mathrm{GL}(d)$, a matrix S from the group of permutation matrices $\mathrm{Sym}(n)$, and a translation vector $t \in \mathbb{R}^d$. Let us assume that $t = 0$ by translating both point clouds to have barycenters, an affine invariant, at 0. Furthermore, we assume, generically, that $\operatorname{rank} X = d$, which implies the same rank for Y.

Let $\mathrm{Gr}(k, n)$ be the Grassmannian of k–dimensional planes in $\mathbb{R}^n$. We can view the point cloud X as a map $X : \mathbb{R}^n \to \mathbb{R}^d$ defined by $X(v) = X\,v$, for $v \in \mathbb{R}^n$. Then $\ker X$ is an element of $\mathrm{G} = \mathrm{Gr}(n-d, n) \cong \mathrm{Gr}(d, n)$. Moreover, for any $L \in \mathrm{GL}(d)$ we have $\ker LX = \ker X$. Thus, X and all of its non–degenerate images define the same element of G. Let P_X be the orthogonal projection of $\mathbb{R}^n$ onto $\ker X^\perp$. The orthogonal projection P_X, which uniquely defines $\ker X$, will be used to represent an element of G. Note that P_X can be computed from the (reduced) singular value decomposition (SVD) of X,

$$X = U \cdot \Sigma \cdot V \Rightarrow P_X = V^T\,V, \tag{1}$$

where $U \in \mathbb{R}^{d\times d}$, $\Sigma \in \mathbb{R}^d$ is the diagonal of non–zero singular values, and $V \in \mathbb{R}^{d\times n}$.

We also have $P_Y = S^T\,P_X\,S$. The matrix S is not unique if there is a linear transformation $T \in \mathrm{GL}(d)$ such that $T\,X = X\,S$. Generically this does not happen, which we formulate as "X has no symmetries".

Thus, we can reformulate the initial problem as

$$\|P_Y - S^T\,P_X\,S\|_F \to \min_{S\in\mathrm{Sym}(n)}, \tag{2}$$

which is equivalent to

$$\operatorname{tr}(P_Y\,S^T\,P_X\,S) \to \max_{S\in\mathrm{Sym}(n)}. \tag{3}$$

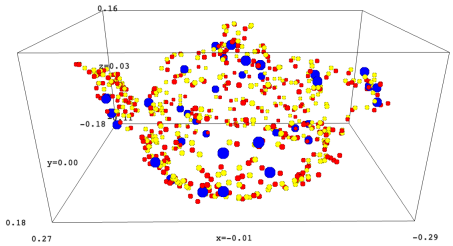

Figure 2: "Teapot" cloud: specimen X (yellow) and preimage $L_0^{-1}\,Y$ (red). Points of X having no correspondence in Y are marked (blue).

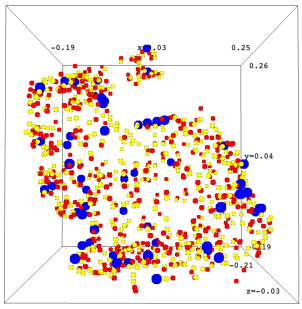

Figure 3: "Bunny" cloud: specimen X (yellow) and preimage $L_0^{-1}\,Y$ (red). Points of X having no correspondence in Y are marked (blue).

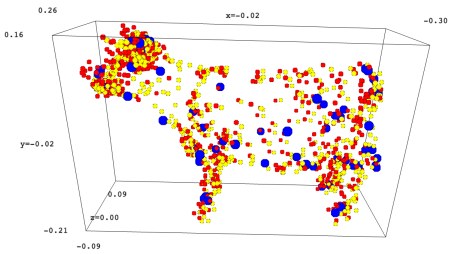

Figure 4: "Cow" cloud: specimen X (yellow) and preimage $L_0^{-1}\,Y$ (red). Points of X having no correspondence in Y are marked (blue).

This is the Quadratic Assignment Problem (QAP) [9] and is NP–complete.

A generic picture of the "Teapot" point cloud X and its image Y is shown in Figure 1. The feature matching under the linear map L between the points is marked. However, the points of Y may be randomly permuted, and determining the right matching S is necessary to recover the linear map L.

Rather than directly optimizing over the permutation matrices, we relax the constraint set to the convex hull of S, the set of doubly stochastic matrices (also known as the Birkhoff polytope),

$$\mathcal{D} = \{S \in [0,1]^{n\times n} \;\big|\; S1_n = 1_n, S^t 1_n = 1_n\}, \tag{4}$$

non–negative square matrices whose rows and columns all sum to one.

Relaxing the feasible region allows one to employ the tools of continuous optimization to search for local optima, as this is not a convex problem different initializations can give different results.

In the ideal case when $P_Y = S^T\,P_X\,S$ we have that

$$\max_{S\in\mathrm{Sym}(n)} \operatorname{tr} P_Y\,S^T\,P_X\,S = \operatorname{tr} P_Y^2 = \operatorname{tr} P_Y = d. \tag{5}$$

The penultimate equality holds because $P^2 = P$ for any projection operator P.

Thus, given a matrix S produced by rQAP, we may assign a weight $w(S)$ to it that measures the proximity of S to S_*. For example,

$$w(S) = \exp\big(-C \cdot (\mathrm{tr}(P_Y S^T P_X S) - d)^2\big), \quad (6)$$

where $C > 0$ is some large constant. The requirement is that $w(S) \approx 1$ for permutations S that are Hamming–close to S_* and $w(S) \approx 0$ for all others.

The idea is to run a sufficiently large number, N, of trials of rQAP, with different initializations obtaining the solutions $S_1, \ldots, S_N$, and then project

$$\pi : \sum_{i=1}^{N} w(S_i)\, S_i \longmapsto S_0 \in \mathrm{Sym}(n). \quad (7)$$

The projection π to the nearest permutation can be realized e.g. by the Hungarian Algorithm.

Another approach would be to take such an S_* among $S_1, \ldots, S_N$ that provides the best match between P_X and P_Y, that is $S_* = \mathrm{argmin}_{S_i} \|P_Y - S_i^T P_X S_i\|_F$. The results of [8] suggest that this indefinite relaxation of QAP proposed in [10] finds the optimal solution, S_*, with high probability. We shall refer to the algorithm proposed in [8] as rQAP (relaxed Quadratic Assignment Problem).

We compare the best match and weighted sum approaches in section 2.4 below.

Once S is recovered, we recover L as

$$L = Y\, S^T\, X^T\, (X\, X^T)^{-1}, \quad (8)$$

since L solves the least squares problem

$$\min_{L \in \mathrm{GL}(d)} \|L\, X - Y\, S^T\|_F. \quad (9)$$

Algorithm 1 RoAM: Robust Affine Matching with Grassmannians

Require: $X, Y \subset \mathbb{R}^{d \times n}$
Ensure: L_0 and S_0 so that $Y \approx L_0\, X\, S_0$

$X \leftarrow \overline{X}$, $Y \leftarrow \overline{Y}$ ▷ centering
$P_X \leftarrow \ker X$, $P_Y \leftarrow \ker Y$
for $i \leftarrow 1$ to N **do** $S_i \leftarrow \max_{S \in \mathcal{D}} \mathrm{tr}(P_Y\, S^T\, P_X\, S)$ ▷ rQAP
end for
$S_0 \leftarrow \pi\left(\sum_{i=1}^{N} w(S_i)\, S_i\right)$ ▷ Hungarian Algorithm
$L_0 := Y\, S_0^T\, X^T\, (X\, X^T)^{-1}$

2.1. Correctness

Let S be the feature matching matrix produced by RoAM. Let $\mathbb{P}(S = S_*) = 1 - \alpha$ for some constant $0 \le \alpha < 1$, and $\mathbb{P}(S \neq S_*) = \alpha$. In the setting of best matching for random Bernoulli graphs $\alpha = 0$ holds [10].

Let $w : \mathcal{D} \to [0, 1]$ be a weight function on the set of doubly stochastic matrices $\mathcal{D} \supset \mathrm{Sym}(n)$ such that $w(S_*) = 1$ while $w(S') \le \varepsilon$. Let $S_w = w(S)\, S$ be the weighted version of a permutation $S \in \mathrm{Sym}(n)$. Then

$$\begin{aligned}(1-\alpha) S_* &\le \mathbb{E}\, S_w \\ &= (1-\alpha)\, w(S_*)\, S_* + \sum_{S' \in \mathrm{Sym}(n),\ S' \neq S_*} \mathbb{P}(S = S')\, w(S)\, S \\ &\le (1-\alpha) S_* + \varepsilon\alpha \mathbf{1}^T \mathbf{1} \end{aligned} \quad (10)$$

Thus, once $\varepsilon < \frac{1-\alpha}{\alpha}$ we have that $\pi(\mathbb{E}\, S_w) = S_*$. However, we need to guarantee that the same equality holds once we replace $\mathbb{E}\, S_w$ by the sample average $\overline{S}_w$ for a large enough sample $\{S_i\}_{i=1}^N$ of permutation matrices produced by RoAM.

Let the sample $\{S_i\}_{i=1}^N$ contain $m \le N$ matrices $S_i = S_*$ and $N - m \ge 0$ matrices $S_i \neq S_*$. Then we have $\alpha = 1 - \frac{m}{N}$, and thus $\pi(\overline{S}_w) = S_*$ for any $\varepsilon < \frac{m}{N-m}$.

The Cauchy–Bunyakovsky inequality implies that

$$\begin{aligned}|\mathrm{tr}(P_Y S^T P_X S)| &\le \|P_Y S^T\|_F\, \|P_X S\|_F \\ &= \|P_Y\|_F \|P_X\|_F = d, \end{aligned} \quad (11)$$

and thus, we need to choose a sufficiently large constant

$$C \ge \frac{\log(1/\varepsilon)}{4d^2} \quad (12)$$

in the equality

$$w(S) = \exp\big(-C \cdot (\mathrm{tr}(P_Y S^t P_X S) - d)^2\big). \quad (13)$$

2.2. Robustness to noise

As follows from Wedin's theorem [11] (see also Theorem 2.5 and Corollary 2.6 in [12]), given a point clouds $Y \subset \mathbb{R}^{d \times n}$ a small perturbation of the vectors in Y results only in a small perturbation of the projection P_Y as the subspace $\ker Y^{\perp}$, corresponds to the d largest singular values of Y. This means that $\ker Y$ is only slightly perturbed in the sense that if $\widetilde{Y} = Y + N$, where N represents noise, then $\|P_Y - P_{\widetilde{Y}}\|_2 = O(\|N\|_2)$.

2.3. Point discrepancy

Let $X \subset \mathbb{R}^{d\times m}$ and $Y \subset \mathbb{R}^{d\times n}$, where we have $m \le n$. Assume that there exist two matrices $L \in \mathrm{GL}(d)$ and $S \in \{0,1\}^{n\times m}$ such that $Y' = Y\,S = L\,X$.

Then we replace the projection matrix $P_X \in \mathbb{R}^{m\times m}$ with $P_X \leftarrow P_X \oplus \mathbf{0}_{n-m}$ and apply RoAM to the new pair $P_X, P_Y \in \mathbb{R}^{n\times n}$. Once an output S_0 is produced, we pick only the first m columns to produce a matching between the points of X and Y. Indeed, matching any of the extra $n-m$ rows/columns of P_Y to a zero row/column in P_X will be heavily penalized, and thus rQAP will target the closest possible match for the first m rows/columns of P_X and some m rows/columns of P_Y. Based on our experiments, this approach works well with missing/added points in X and Y, even for relatively large discrepancies.

2.4. Finding feature matching

In Figure 19 we compare the relative errors δ_L, δ_Y and δ_X as defined above for the two approaches:

1. Taking the best match S_0 solving $\|P_Y - S^T P_X S\| \to \min_{S\in\{S_1,\dots,S_N\}}$;
2. Taking the weighted sum $S_0 = \pi\left(\sum_{i=1}^N w(S_i)\,S_i\right)$.

The relative Hamming distance

$$\delta_H = \frac{1}{2n}\|S - S_0\|_F^2$$

between the recovered S_0 and ground truth S feature matching of n points is measured: for this purpose we assume that X and Y have no point discrepancy.

The initial point cloud X is a random point cloud of n points distributed uniformly in $[0,1]^d$, and the image point cloud $Y = L\,X\,S$ is obtained by applying a random linear map L with given condition number, and a random permutation matrix $S \in \mathrm{Sym}(n)$. We also apply Gaussian multiplicative noise $\mathcal{N}(1,\sigma^2)$ to each entry of Y.

In Figure 19 we have $d = 3$, $n = 100$, and $\mathrm{cond}\,L = 3.0$. The noise magnitude (determined by σ) and the number of iterations N in the feature matching search vary: namely $\sigma \in [0.05, 0.25]$ (with a step of 0.05) and $N = 2^k$ for $k = 5$ up to $k = 10$ with unit step. For each pair of parameters (σ, N) we perform 100 tests and report the average results.

As evidenced in Figure 19 neither approach shows acceptable accuracy for $N \le 2^8$. For $N \ge 2^9$ the best match approach shows equally good or better accuracy while the noise is relatively small ($0.0 < \sigma < 0.15$), while the weighted sum approach appears to be more robust for noise of larger magnitudes ($0.15 < \sigma < 0.25$).

3. Experiments

The following quantities were computed for each experiment:

- the error in recovering the affine transformation L, $\delta_L = \|L - L_0\|_2/\|L\|_2$;
- the error in matching the preimages defined as $\delta_X = \|[L_0^{-1}\,Y\,S^T]_X - X\|_2/\|X\|_2$, where $[A]_B$ is the matrix formed from the columns of $A = L_0^{-1}\,Y\,S^T \in \mathbb{R}^{d\times n}$ indexed with the same indexes as the columns of $B = X \in \mathbb{R}^{d\times m}$ inside $X' \in \mathbb{R}^{d\times n}$;
- the error in matching the images, $\delta_Y = \|L\,X - L_0\,X\|_2/\|L\,X\|_2$.

3.1. Random point clouds

In our numerical experiments, RoAM was applied to pairs of synthetic point clouds with the following specifications: one specimen point cloud $X \subset \mathbb{R}^{3\times m}$ and another point cloud $Y \subset \mathbb{R}^{3\times n}$ such that

- X is a random subset of $X' \subset \mathbb{R}^{3\times n}$ with $m = \lfloor \lambda\, n \rfloor$ points with $\lambda \in (0,1]$ being the level of similarity (with $\lambda = 1$ if all of X coincides with X');
- $Y = Y' \odot N$, with $Y' = L\,X'\,S$, $L \in \mathrm{GL}(3)$ a random linear map such that $L = P\,O$ with P a positive definite matrix having condition number $c \in [1, +\infty)$ and O a uniformly distributed orthogonal matrix; $S \in \mathrm{Sym}(n)$ a random permutation; and N, multiplicative noise such that $N_{ij} \in \mathcal{N}(1,\sigma^2)$, $i = 1,\dots,3$, $j = 1,\dots,n$, are i.i.d. Gaussian variables. Here $\odot$ is the Hadamard (entry–wise) matrix product.

3.2. Caerbannog point clouds

We used the (unoccluded) Caerbannog clouds [13] as the primary source of specimens X, while the corresponding Y clouds were synthesized.

The following values for σ (noise level) and λ (similarity level to create point discrepancy) were used:

- $\sigma \in \{0.0, 0.01, 0.05, 0.1, 0.15, 0.2\}$,
- $\lambda \in \{1.0, 0.95, 0.90, 0.85, 0.8, 0.7, 0.6, 0.5\}$.

It appears to us that there is no need to increment σ uniformly as we need to test only for reasonably small levels of noise (to observe that the algorithm works robustly) and for relatively high ones (to understand the limits of the algorithm's applicability). It also appears that we do not need to use lower levels of similarity than 0.5 as more than half of the points missing constitutes a fairly large discrepancy. As in the case of noise, we prefer to vary the discrepancy non–uniformly.

For each value of the noise σ and point discrepancy λ, a batch of 10 tests was run and the mean values of δ_L, δ_Y, and δ_X were computed. Every run of RoAM used $N = 2^{10}$ calls to rQAP.

The Python code to perform the tests and the respective output stored is accessible on GitHub [14]. The code requires SageMath [15] and runs in the Jupiter environment. The output is saved in `csv` format and can be accessed without running the code.

For different levels of noise and occlusion, we got 69.43 seconds on average per batch of 100 trials for the "choose best" approach and 57.07 seconds on average per batch of 100 trials for the "weighted sum" approach (see [14]). Thus, a single run of RoAM took approximately between 0.57 to 0.69 seconds on a MacBook Pro (M1, 2020).

Below we provide a graphical interpretation of the results, where for each value of σ, the values of δ_*, $* \in \{L, X, Y\}$ are provided for different values of λ. The higher the point discrepancy (i.e. the lower λ), the longer the horizontal bar over the respective value of δ_*, $* \in \{L, X, Y\}$.

In Figures 5 – 12, we provide only values $\delta_* \leq 1.5$, and sometimes cannot avoid the visual merging and overlaps of some of the horizontal bars. The exact values of δ_*, $* \in \{L, X, Y\}$ can be found in the GitHub repository [14].

As part of the data available in [14], we also measured the quantities

1. $d_\sigma = \|Y' - Y\|_2 / \|Y\|_2$ (the relative noise compared to the ground truth image of X under L); and
2. $d_\lambda = \|X' \setminus X\|_2 / \|X\|_2$ (the relative point discrepancy compared to the initial specimen image).

The `csv` files of the test data contain the following values in each line in this order: σ, λ, d_σ, d_λ, δ_L, δ_Y, δ_X. We are mostly interested in measuring δ_L and δ_Y to understand how well we can recover L and how well the recovered linear map L_0 approximates the original image of X under the action of L. The distance between the preimage of Y under L_0 and the specimen image X is used for comparison, and we use preimages for illustrative purposes (as the direct images under L can be fairly deformed and thus less suitable for visualization).

The linear transformation L in all tests had condition number $\operatorname{cond} L = \|L\|_2 \, \|L^{-1}\|_2$ equal to 3.

In Figures 2, 3 and 4, we show the specimen cloud X (yellow), the preimage $[L^{-1} Y S^T]_X$ (red), and also the point discrepancy consisting of points in X' that do not belong to X (blue). For all three point clouds we have $\sigma = 0.05$, $\lambda = 0.90$, $\operatorname{cond} L = 3$. The number of trials in each case is $N = 2^{10}$.

In Figure 2 we illustrate a test with X having 315 points and Y having 351 points. The test parameters are $d_\sigma \approx 0.045$, $d_\lambda \approx 0.328$, with the output $\delta_L \approx 0.045$, $\delta_Y \approx 0.042$, and $\delta_X \approx 0.140$.

In Figure 3 we illustrate a test with X having 475 points and Y having 528 points. The test parameters are $d_\sigma \approx 0.044$, $d_\lambda \approx 0.306$, with the output $\delta_L \approx 0.053$, $\delta_Y \approx 0.044$, and $\delta_X \approx 0.127$.

In Figure 4 we illustrate a test with X having 541 points and Y having 602 points. The test parameters are $d_\sigma \approx 0.048$, $d_\lambda \approx 0.348$, with the output $\delta_L \approx 0.036$, $\delta_Y \approx 0.036$, and $\delta_X \approx 0.141$.

3.3. Partial scans of the same object

We also use two point clouds sampled from two scanned images of the same sculpture before and after restoration. Namely, our specimen cloud X is sampled from the mesh representing a scan of the damaged sculpture in Figure 14 (left). The other cloud Y is sampled from the mesh of a scan of the restored statue in Figure 14 (right). More precisely, a random linear transformation with a given condition number is applied to the mesh, and only then Y is sampled from it.

In Figure 14 (center) the ground truth of matching the damaged and restored sculptures is shown. In Figure 15–16, we show the matching of the broken sculpture (from which X is sampled) to the restored one with a linear transform L applied (from which Y is sampled). In order not to produce a confusing image of the sculpture distorted by L, we instead show the image of the restored sculpture under $L_0^{-1} L$ (where L is the original linear transform, and L_0 is recovered by running RoAM) followed by

a translation. The translation adjusts the barycenters of the two meshes.

Affine registration cannot recover the initial map as well as rigid registration in the presence of occlusion and noise: the affine group has more degrees of freedom, and thus it will necessarily register some bias produced by the noise and occlusion points. Note that in Figure 15 with more sample points and a smaller condition number of the random linear map, we have a better match than in Figure 16 where the number of sample points is reduced and the condition number of the random linear map is larger.

The code and mesh data are available on GitHub [14].

4. Comparison to other algorithms

In this section, we compare RoAM to another algorithm, GrassGraf [7], which appears to be the state–of–the–art algorithm to recover affine correspondences.

We implemented GrassGraph as described by its authors in [7] and applied it in the simplest case of two point clouds $X, Y \subset \mathbb{R}^{d\times n}$ where X is the specimen cloud to be compared to $Y = L\,X\,S$ with $L \in \mathrm{GL}(d)$, $S \in \mathrm{Sym}(n)$.

The main finding is that GrassGraf loses precision very quickly if Y is affected even by relatively small noise. The test point clouds used are the unoccluded Caerbannog clouds [13]. We measured the values δ_L and δ_Y, just as in the case of RoAM. These values are enough to conclude that GrassGraph does not appear to recover L robustly.

In Figures 20 – 18, we show the graphical depiction of our test results (as in the case of RoAM) for GrassGraph with 3 Laplacian eigenvectors used. The analogous dataset with test results for GrassGraph with 10 Laplacian eigenvectors used is available on GitHub [14]. We do not reproduce it here as it shows only a marginal improvement.

Also, some images of the test clouds are available on GitHub [14]: we do not reproduce them here, note that the "Teapot" cloud shows slightly better robustness to noise and point discrepancy while, say, the "Cow" cloud already demonstrates a great difference between the recovered L_0 and the initial L even for relatively small noise and point discrepancy.

5. Conclusions

Feature matching is at the heart of many applications and requires robust methods for recovery of correspondences and estimation of geometric transformations between domains.

This paper provides a robust and efficient algorithm to find correspondences between point clouds up to an affine transformation and gives bounds on the algorithm's robustness in the presence of noise and point discrepancies.

This paper also continues the line of research started in [16], and aims to generalize it from recovering orthogonal transformations (that are "rigid" in nature) to the case of much "softer" affine correspondences.

6. Acknowledgements

This research is supported by the University of Austin (UATX) and by the Israel Science Foundation (ISF).

References

[1] P. Barrios, V. Guzman, M. Adams, Pso-cola: A robust solution for correspondence-free point set registration, in: 2022 11th International Conference on Control, Automation and Information Sciences (ICCAIS), IEEE, 2022, pp. 223–230.

[2] L. Tang, G. Hamarneh, K. Iniewski, Medical image registration: A review, Medical imaging: technology and applications 1 (2013) 619–660.

[3] P. Besl, N. D. McKay, A method for registration of 3-D shapes, IEEE Transactions on Pattern Analysis and Machine Intelligence 14 (2) (1992) 239–256. doi:10.1109/34.121791.

[4] X. Huang, G. Mei, J. Zhang, R. Abbas, A comprehensive survey on point cloud registration, arXiv preprint arXiv:2103.02690 (2021).

[5] E. Begelfor, M. Werman, Affine invariance revisited, in: 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), Vol. 2, 2006, pp. 2087–2094. doi:10.1109/CVPR.2006.50.

[6] Y.-T. Chi, S. M. N. Shahed, J. Ho, M.-H. Yang, Higher dimensional affine registration and vision applications, in: D. Forsyth, P. Torr, A. Zisserman (Eds.), Computer Vision – ECCV 2008, Springer Berlin Heidelberg, Berlin, Heidelberg, 2008, pp. 256–269.

[7] M. Moyou, A. Rangarajan, J. Corring, A. M. Peter, A grassmannian graph approach to affine invariant feature matching, IEEE Transactions on Image Processing 29 (2020) 3374–3387. doi:10.1109/TIP.2019.2959722.

[8] V. Lyzinski, D. E. Fishkind, M. Fiori, J. T. Vogelstein, C. E. Priebe, G. Sapiro, Graph matching: Relax at your own risk, IEEE Transactions on Pattern Analysis and Machine Intelligence 38 (1) (2016) 60–73. doi:10.1109/TPAMI.2015.2424894.

[9] T. C. Koopmans, M. Beckmann, Assignment problems and the location of economic activities, Econometrica: journal of the Econometric Society (1957) 53–76.
[10] J. T. Vogelstein, J. M. Conroy, V. Lyzinski, L. J. Podrazik, S. G. Kratzer, E. T. Harley, D. E. Fishkind, R. J. Vogelstein, C. E. Priebe, Fast approximate quadratic programming for graph matching, PLOS ONE 10 (4) (2015) e0121002. doi:10.1371/journal.pone.0121002. URL https://doi.org/10.1371%2Fjournal.pone.0121002
[11] P.-Å. Wedin, Perturbation bounds in connection with singular value decomposition, BIT Numerical Mathematics 12 (1) (1972) 99–111.
[12] G. W. Stewart, Error and perturbation bounds for subspaces associated with certain eigenvalue problems, SIAM Review 15 (4) (1973) 727–764. doi:10.1137/1015095.
[13] S. Raghupathi, N. Brunhart-Lupo, K. Gruchalla, Caerbannog point clouds. National renewable energy laboratory, https://data.nrel.gov/submissions/153 (2020). doi:10.7799/1729892.
[14] A. Kolpakov, M. Werman, SageMath worksheets for RoAM: Python code, numerical tests, and test statistics, https://github.com/sashakolpakov/roam (2024).
[15] The Sage Developers, Sagemath, the Sage Mathematics Software System (Version 9.7.1), https://www.sagemath.org (2022).
[16] A. Kolpakov, M. Werman, An approach to robust icp initialization, IEEE Transactions on Pattern Analysis and Machine Intelligence 45 (2023) 1–9. doi:10.1109/TPAMI.2023.3287468.
[17] O. Laric, Three D Scans, https://threedscans.com/ (2012).

7. Appendix: numerical tests

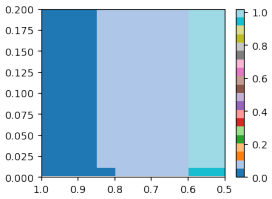

Figure 5: "Teapot" cloud: δ_L as a function of λ (discrepancy level, horizontal) and σ (noise level, vertical)

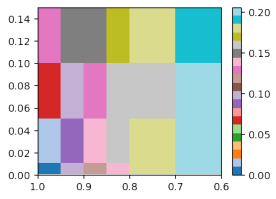

Figure 6: "Teapot" cloud: δ_X as a function of λ (discrepancy level, horizontal) and σ (noise level, vertical)

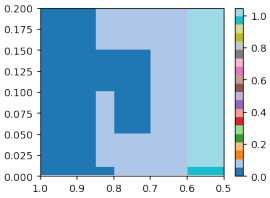

Figure 7: "Teapot" cloud: δ_Y as a function of λ (discrepancy level, horizontal) and σ (noise level, vertical)

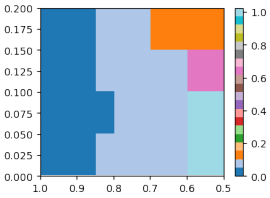

Figure 8: "Bunny" cloud: δ_L as a function of λ (discrepancy level, horizontal) and σ (noise level, vertical)

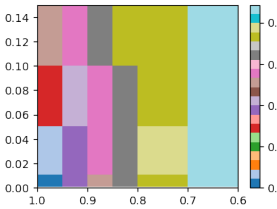

Figure 9: "Bunny" cloud: δ_X as a function of λ (discrepancy level, horizontal) and σ (noise level, vertical)

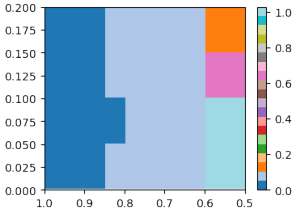

Figure 10: "Bunny" cloud: δ_Y as a function of λ (discrepancy level, horizontal) and σ (noise level, vertical)

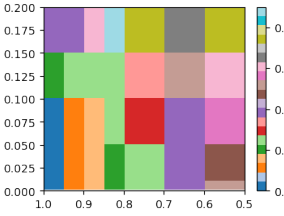

Figure 11: "Cow" cloud: δ_L as a function of λ (discrepancy level, horizontal) and σ (noise level, vertical)

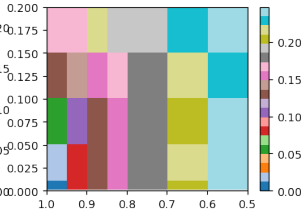

Figure 12: "Cow" cloud: δ_X as a function of λ (discrepancy level, horizontal) and σ (noise level, vertical)

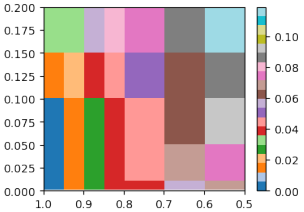

Figure 13: "Cow" cloud: δ_Y as a function of λ (discrepancy level, horizontal) and σ (noise level, vertical)

Figure 14: "Enfant au Chien" before (left) and after (right) restoration (available from [17]). The ground truth of matching the sculptures (center).

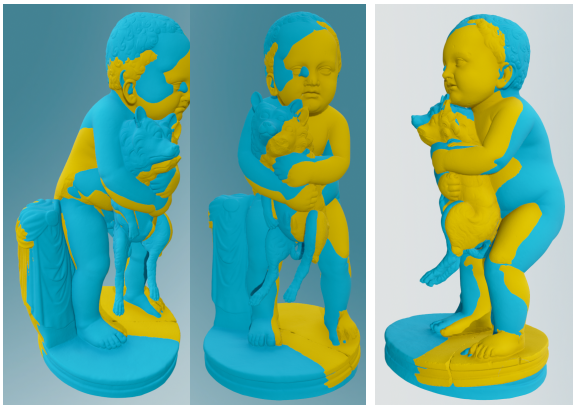

Figure 16: Testing RoAM: sample 400 points from the broken statue mesh for point cloud X, apply a random linear map L with $\operatorname{cond} L = 5.0$ to the restored statue mesh and sample another 400 points from it for point cloud Y. Showing the source and the image of the target under $L_0^{-1} L$. Frontal and side views.

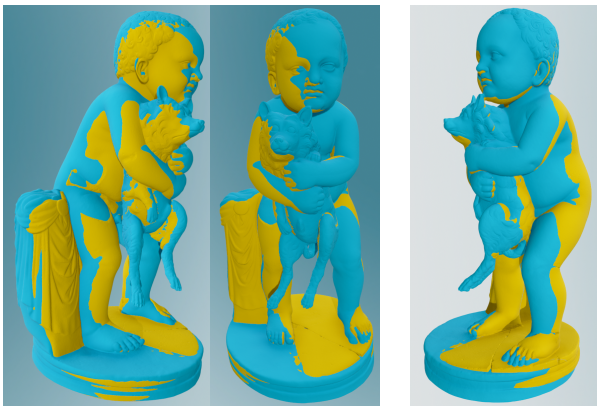

Figure 15: Testing RoAM: sample 600 points from the broken statue mesh (source) for point cloud X, apply a random linear map L with $\operatorname{cond} L = 3.0$ to the restored statue mesh (target) and sample another 600 points from it for point cloud Y. Showing the source and the image of the target under $L_0^{-1} L$. Frontal and side views.

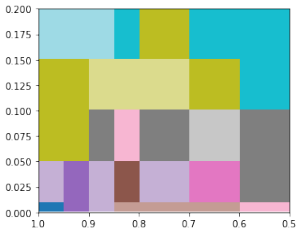

Figure 17: "Cow" cloud for GrassGraph: δ_L as a function of λ (discrepancy level, horizontal) and σ (noise level, vertical)

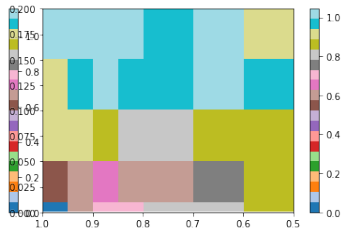

Figure 18: "Cow" cloud for GrassGraph: δ_Y as a function of λ (discrepancy level, horizontal) and σ (noise level, vertical)

Figure 19: Best matching (left) vs. weighted sum (right) approach for restoring the feature correspondence. The relative errors δ_L, δ_Y, and δ_X are shown (top to bottom). The relative Hamming distance between the recovered and ground truth feature matching is measured (bottom row). The horizontal axis of each heatmap is the number of iterations N, the vertical axis is σ for the multiplicative Gaussian noise $\mathcal{N}(1, \sigma^2)$ affecting Y.

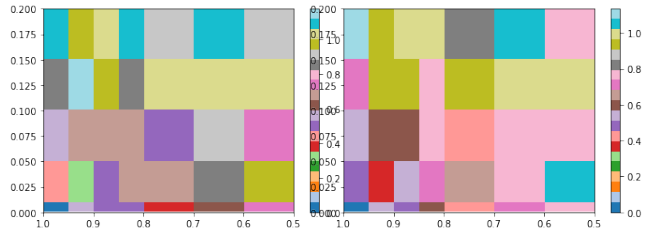

Figure 20: "Teapot" cloud for GrassGraph: δ_L as a function of λ (discrepancy level, horizontal) and σ (noise level, vertical)

Figure 21: "Teapot" cloud for GrassGraph: δ_Y as a function of λ (discrepancy level, horizontal) and σ (noise level, vertical)

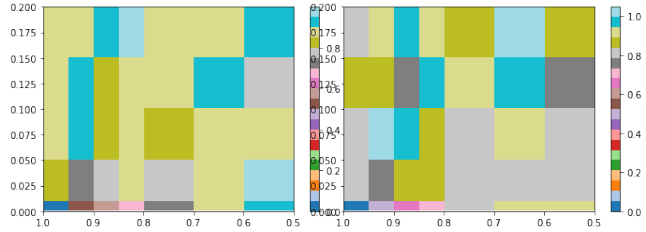

Figure 22: "Bunny" cloud for GrassGraph: δ_L as a function of λ (discrepancy level, horizontal) and σ (noise level, vertical)

Figure 23: "Bunny" cloud for GrassGraph: δ_Y as a function of λ (discrepancy level, horizontal) and σ (noise level, vertical)