Supplementary Material
(Gravitational Approach for Point Set Registration)

Vladislav Golyanik[1,2] vladislav.golyanik@dfki.de
Sk Aziz Ali[1] saali@rhrk.uni-kl.de
Didier Stricker[1,2] didier.stricker@dfki.de

[1] University of Kaiserslautern, Germany
[2] German Research Center for Artificial Intelligence (DFKI), Germany

Further details on the Gravitational Approach going beyond the scope of the main matter are given in this supplementary material. Specifically, we provide:

A A proof of the Proposition 1
B Details on the rigorous rotation resolving
C Scale resolving through divergence of the force field
D Reasoning of the GPE expression in Eq. (17)
E An additional experiment on SLAM datasets
F A remark on the Gravitational Search Algorithm

All references to expressions from the main matter are given directly by the corresponding numbers (e.g. 8). References to the expressions introduced in the appendix are preceded by a chapter literal (e.g. C.1).

A. Proof of the Proposition 1

Proof. Eq. (14) has no exact solution, unless $\hat{\mathbf{Y}}_{t+1}$ lies in the column space of $\hat{\mathbf{Y}}_t$. Nevertheless, we can solve

$$\Upsilon_{t+1} = \hat{\mathbf{Y}}_t s, \tag{A.1}$$

where Υ_{t+1} is a projection of $\hat{\mathbf{Y}}_{t+1}$ to the column space of $\hat{\mathbf{Y}}_t$ by normal equations. After rewriting Eq. (A.1) in terms of the known variables we obtain:

$$s = (\hat{\mathbf{Y}}_t^T \hat{\mathbf{Y}}_t)^{-1} \hat{\mathbf{Y}}_t^T \hat{\mathbf{Y}}_{t+1} = \frac{\hat{\mathbf{Y}}_t^T \hat{\mathbf{Y}}_{t+1}}{\hat{\mathbf{Y}}_t^T \hat{\mathbf{Y}}_t} \tag{A.2}$$

which corresponds to the optimal in the least squares sense solution to scaling. □

B. Rigorous rotation resolving

A torque is always defined relative to a reference point (2D) or an axis (in three and higher dimensions). From Euler's rotation theorem (1776) follows: *in three and higher dimensions, rotation of a rigid body about a fixed point inside the body is equivalent to the rotation about some axis passing through this point* [4].

Having determined a torque, it is possible to compute a rotation angle around a reference point/axis and convert it to the corresponding rotation matrix. This procedure consists of several steps (see Fig. I): 1) determining the axis of rotation $\vec{a}_Y$; 2) computing the moment of inertia I_y around the axis; 3) determining the angular acceleration α; 4) obtaining the angular velocity ω; 5) updating the rotation angle φ around the axis $\vec{a}_Y$, whereupon the rotation matrix R is inferred. The sequence of steps 1) - 5) is repeated in every iteration.

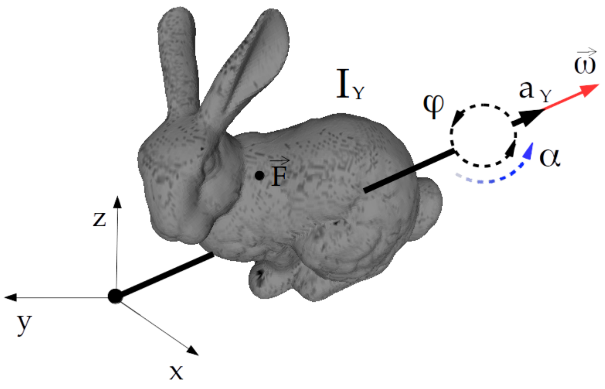


Figure I: Rotation of a rigid body in 3D can be unambiguously represented by an axis of rotation a_Y and an angle φ around the axis. An external force $\vec{F}$ (directed towards the observer) causes a torque with the moment of inertia I_y inducing angular acceleration α. From the angular acceleration and the current angular velocity $\vec{\omega}$, an update for rotation angle φ through forward integration is computed.

Noticeably, a gravitational field is a conservative force field. This implies that it cannot by itself cause any rotational effects, i.e

$$\nabla \times \vec{\mathbf{F}} = 0, \tag{B.1}$$

where $\nabla\times$ denotes curl operator. If a rigid body at rest is placed into a static gravitational field, it will start accelerate; the potential energy will transform to the kinetic energy

(and vice versa), but the body will not start to spin. In other words, without external forces the point of rotation will coincide with the center of gravity — a point with the resulting zero torque. Through introducing an external force the point of rotation changes and the resulting torque becomes non-zero — in this case rotation occurs. Thus, point of rotation is set to the template's center of mass in the current formulation of GA. In an inhomogeneous vector field the center of gravity of a rigid body either with a constant or varying density does not coincide with the center of mass and a torque emerges. This torque is used in GA to resolve rotation. Indeed, the Kabsch algorithm resolves rotation relatively to the center of mass. The center of mass is fixed, which is only possible through an external force. Since the center of mass and the center of gravity do not coincide, as a result a torque emerges.
To determine the axis of rotation, the following observation can be made. If a body rotates around a fixed point/axis, rotation causes each point of the rigid body to displace depending on the distance of this point to the center/axis of rotation. The displacement vector lies in the plane perpendicular to the axis of rotation and passing through the point's position vector. Conversely, in our case point displacements $\mathbf{D}$ are available and the axis of rotation has to be determined. Considering starting and final position vectors of a point Y_i, the axis of rotation can be estimated as

$$\vec{a}_{Yi} = \vec{r}_{Yi} \times (\vec{r}_{Yi} + \vec{d}_i). \tag{B.2}$$

Accounting for all the points, we strengthen the estimate and resolve the axis of rotation of the rigid system of particles as a mean vector of individual estimates:

$$\vec{a}_Y = \frac{\sum_i \vec{a}_{Yi}}{M}. \tag{B.3}$$

For this rigid body, the moment of inertia I_Y around the axis reads:

$$I_Y = \sum_j^M m^{Yj} |r^{Yi}|^2_{\vec{a}_Y}, \tag{B.4}$$

where $|r^{Yi}|_{\vec{a}_Y}$ denotes distance of a point r^{Yi} to the axis of rotation $\vec{a}_Y$. Respectively, torque around the axis $\vec{a}_Y$ reads:

$$\tau = \sum_{j=1}^M |r^{Yi}|_{\vec{a}_Y} \times \vec{F}_{Yi}. \tag{B.5}$$

Finally, it is possible to compute an angular acceleration α of the system of particles $\mathbf{Y}$ using the relation between a torque and a moment of inertia as

$$\alpha = \frac{\tau}{I_Y}. \tag{B.6}$$

Performing forward integration in the similar manner as in the case of an unconstrained translational motion (Eqs. (7), (8)), the angular velocity ω_Y and the rotation angle φ_Y can be obtained. The latter together with the axis of rotation forms an axis-angle representation and uniquely determines the rotation matrix $\mathbf{R}$ in every iteration.

C. Scale resolving through divergence of the force field

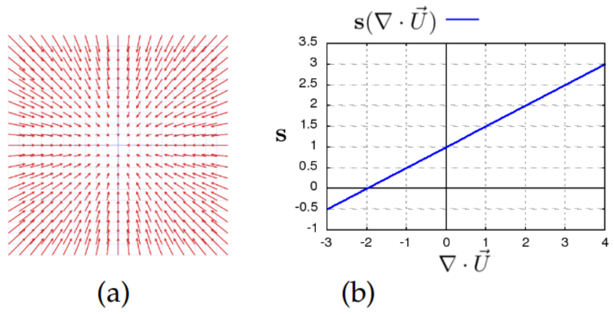


Figure II: (a): an example of a central vector field $-x\hat{\mathbf{i}} - y\hat{\mathbf{j}}$ and (b): the function relating divergence of a central vector field with the scaling factor $\mathbf{s}$ of an influenced rigid system of particles.

Scaling can be resolved considering divergence of the *central force field component* $\vec{U}$ of the displacement field $\mathbf{D}$. We refer to the central force field component as a curl-free component of the vector field, so that

$$\vec{U} = ax\hat{\mathbf{i}} + ay\hat{\mathbf{j}} + az\hat{\mathbf{k}}, \tag{C.1}$$

where a is a constant. Ultimately, it is possible to relate $\nabla \cdot \vec{U}$ and scaling. Fig. II shows an example of a central vector field and the function relating divergence of a central vector field and the scaling factor of a rigid system of particles placed into it. The relation $s(\nabla \cdot \vec{U})$ is linear. When $\nabla \cdot \vec{U} = 0$, there is no dilatational component in the vector field and the scaling is 1. If $\nabla \cdot \vec{U} = 2$, the force field will tend to place every point twice as far from the origin of the coordinate system per unit of time. This corresponds to a twofold scaling. In the case of $\nabla \cdot \vec{U} = -1$, every coordinate is halved which corresponds to the scaling value of 0.5. Divergence -2 implies that wherever the point is located, it is translated to the origin of the coordinate system. This corresponds to an immediate shrinkage of a point set to a single point. Values less than -2 would shrink and cause reflection of a point set. For other divergence values the linear relation holds likewise. Finding $\vec{U}$ requires Helmholtz vector field decomposition with extraction of the central force field component from the curl-free component. For more details on the Helmholtz decomposition an interested reader may refer to [1].

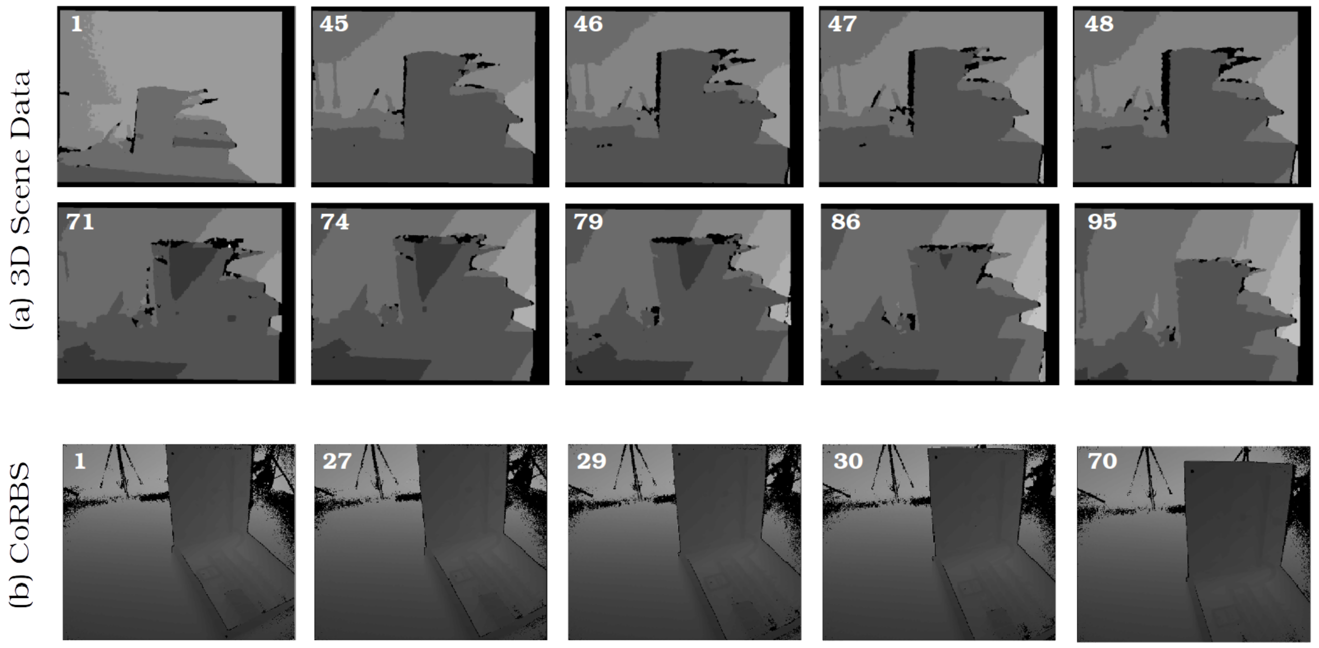


Figure III: Depth maps involved in the experiment: (a) selected frames from the *copyroom* dataset [8]; (b) selected frames from the *electrical cabinet* dataset [7]; the contrast of the depth maps is enhanced for better perceptibility; corresponding frame numbers are given in the top left corners of the images.

D. Gravitational Potential Energy

Between two bodies of masses m_1 and m_2 the gravitational potential energy (GPE) U_p emerges:

$$U_p = -G\frac{m_1 m_2}{r}, \tag{D.1}$$

where G is the gravitational constant and r is the distance between centers of mass of the bodies. Accordingly, the energy function for two interacting rigid systems of particles can be defined as

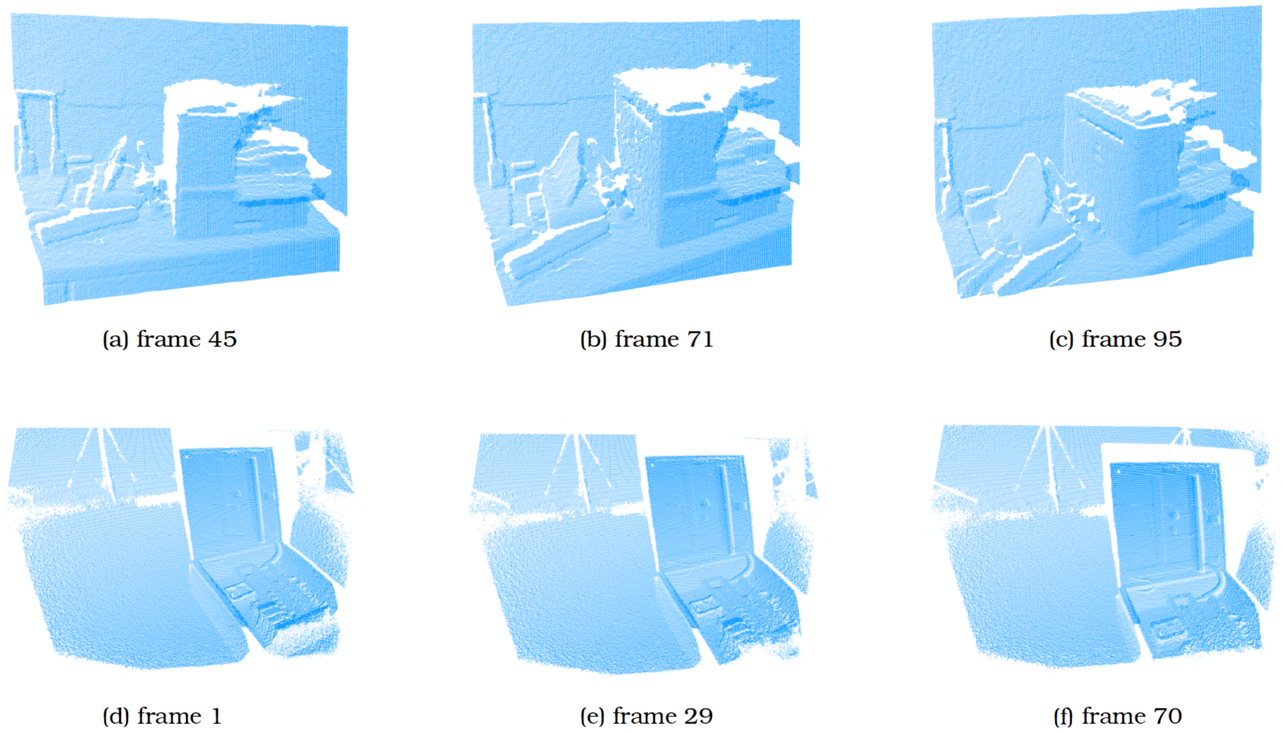
$$E(\mathbf{R}, \mathbf{t}, \mathbf{s}, \epsilon) = -G \sum_{i}^{M} \sum_{j}^{N} \frac{m^{Yi}\, m^{Xj}}{\|\mathbf{R}\; c^{Yi}\; \mathbf{s} + \mathbf{t} - c^{Xj}\| + \epsilon}. \tag{D.2}$$

At a local minimizer $(\mathbf{R}_{opt}, \mathbf{t}_{opt}, \mathbf{s}_{opt})$ the total GPE of the system and the value of the energy function E are locally minimal. Thus, it is possible to express the GA stopping criterion by a difference of GPEs in several consecutive iterations.
Note that direct minimization of the energy function in Eq. (D.2) is not trivial. For instance, applying a non-linear optimization algorithm such as Levenberg-Marquardt [5] with quaternion parametrization of rotations is problematic due to instability. Nevertheless, it might be possible to employ a genetic algorithm to optimize it.

E. An experiment on SLAM datasets

In this section, we describe an additional experiment on real-world data. We use point clouds from two RGB-D datasets, i.e. the Stanford 3D Scene Dataset [8] (captured by a pattern projection sensor) and CoRBS [7] (captured by a time-of-flight camera). These datasets are primarily designed to benchmark SLAM methods. One of the goals of SLAM is to reconstruct a scene given multiple depth maps (which can be unambiguously converted into point clouds) and color images captured by an RGB-D sensor such as Kinect. An RGB-D sensor outputs RGB images as well as depth maps for discrete moments of time.
The course of the experiment is equal for both datasets. The difference concerns resolution of the depth maps and, as a consequence, the number of points in the point clouds. In Fig. III depth maps involved in the experiments are shown. Every depth map corresponds to a frame in a recorded RGB-D sequence. The resolutions of the depth maps are 640×480 and 512×424 pixels for the *copyroom* (from the Stanford Scene Dataset) and the *electrical cabinet* dataset (from CoRBS) respectively. The depth maps are converted to the corresponding point clouds using the parameters provided by the authors of the datasets, i.e. intrinsic camera parameters and scaling factors for the depth maps. In Fig. IV, several examples of point clouds are shown.

For the registration we choose several frame combinations with different frame intervals between references and templates. If a frame interval is reasonably large, rigid reg-


(a) frame 45 (b) frame 71 (c) frame 95

(d) frame 1 (e) frame 29 (f) frame 70

Figure IV: Selected point clouds converted from the depth maps corresponding to the frames form the Stanford 3D Scene Dataset (a)-(c) and CoRBS (d)-(f).

istration can become challenging because of clustered outliers, missing parts, effects of the depth sensor distortion and noise presented in both point sets. The goal of the experiment is to demonstrate that GA can potentially be used in a SLAM system to register point clouds. Therefore, we compare cloud-to-cloud distances between point clouds on the initialization step and after the registration is finished. Thus, we can qualitatively observe if GA can improve the template's pose based at least on a single criterion.

For several challenging frame pairs, rigid registration with GA is performed. Point clouds contain $\approx 2.67 \cdot 10^5$ and $\approx 2 \cdot 10^5$ points for the Stanford 3D Scene Dataset and CoRBS respectively. All point clouds are subsampled so that every of them contains 2000 points. The algorithm converges at the latest after 100 iterations when oscillations attenuate. The runtime per iteration amounts to ≈ 0.5 sec. $G = 1.27 \cdot 10^{-4}$ for the Stanford 3D Scene Dataset and $G = 8.27 \cdot 10^{-4}$ for CoRBS is set. In all experiments, $\eta = 0.2, \epsilon = 0.1$ are set and the scaling is fixed, since the point clouds do not differ in scale significantly in this experiment. Note that the color information presented in the color images is not used — all points are of equal masses. This makes the scenario more challenging for GA. Color information provided by the RGB images can be used to assign different point weights both in a reference and in a template, similar to the Orion experiment (see Fig. 5, Sec. 4).

In Fig. V, results on the Stanford 3D Scene Dataset are summarized. In the first column (Fig. V-a), initializations are shown — the point clouds are taken directly after conversion from the depth maps. The references are shown in cyan and the templates are shown in orange. On the left, frame numbers of the reference frames (cyan) and the template frames (orange) are provided. The second column (Fig. V-b) contains color-coded cloud-to-cloud distances between the templates and references. In the third column (Fig. V-c), the GA registration results and, similarly, in the fourth column (Fig. V-d) the corresponding color-coded cloud-to-cloud distances between the registered templates and references are provided. Analogously to the Stanford 3D Scene Dataset, the results on the CoRBS Dataset are summarized in Fig. VI. Note that every cloud-to-cloud distance plot evinces the same color saturation for a given data set. This simplifies the visual comparability of the results.

In all attempts, GA is able to significantly improve cloud-to-cloud distances between the point clouds. This demonstrates robustness of GA against the typical real-world point cloud artefacts occurring in different combinations. Thus, the experiment shows that GA may potentially be used in challenging real-world scenarios such as scene completion for SLAM. Accordingly, an accelerated GA will be tested in the SLAM scenario in future work. Moreover, additional information such as initial velocity and colors will be used in future experiments.

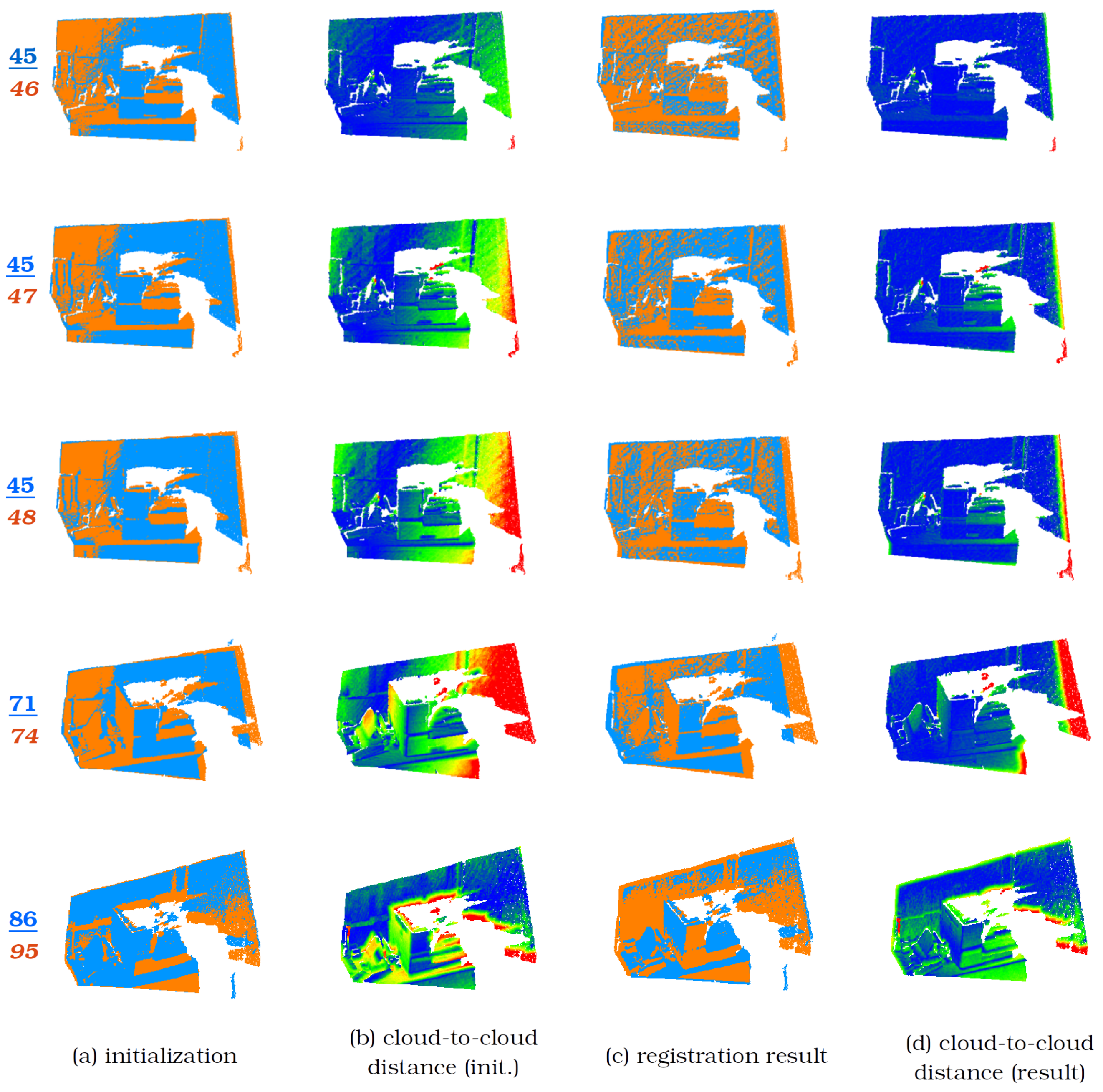

45
46
45
47
45
48
71
74
86
95
(a) initialization
(b) cloud-to-cloud distance (init.)
(c) registration result
(d) cloud-to-cloud distance (result)

Figure V: Results of the experiment on the Stanford 3D Datasets [8]: (a) initial misalignments (initialization); references are shown in cyan and templates in orange; on the left, corresponding frame numbers are provided (in cyan for the reference frames and in orange for the template frames). (b) cloud-to-cloud distance corresponding to the initial misalignment; (c) GA registration results; (d) cloud-to-cloud distance corresponding to the registration result; cloud-to-cloud distances are coded with a Blue<Green<Yellow<Red color scale; the saturation point (red value) amounts to 0.08 distance units.

F. A remark on the Gravitational Search Algorithm

An optimization algorithm known as Gravitational Search Algorithm (GSA) [6] was once applied to optimize the ICP objective function [3]. Despite the distantly similar denotation, our formulation has nothing in common with GSA. GSA is a general purpose heuristic multi-agent optimization algorithm belonging to the class of evolutionary algorithms. Noticeably, it has more discrepancies with gravitational motion than similarities: positions of the agents are initialized randomly, masses are altered based on the fitness of the agents, velocity updates occur randomly and the distances between solutions are not taken into ac-

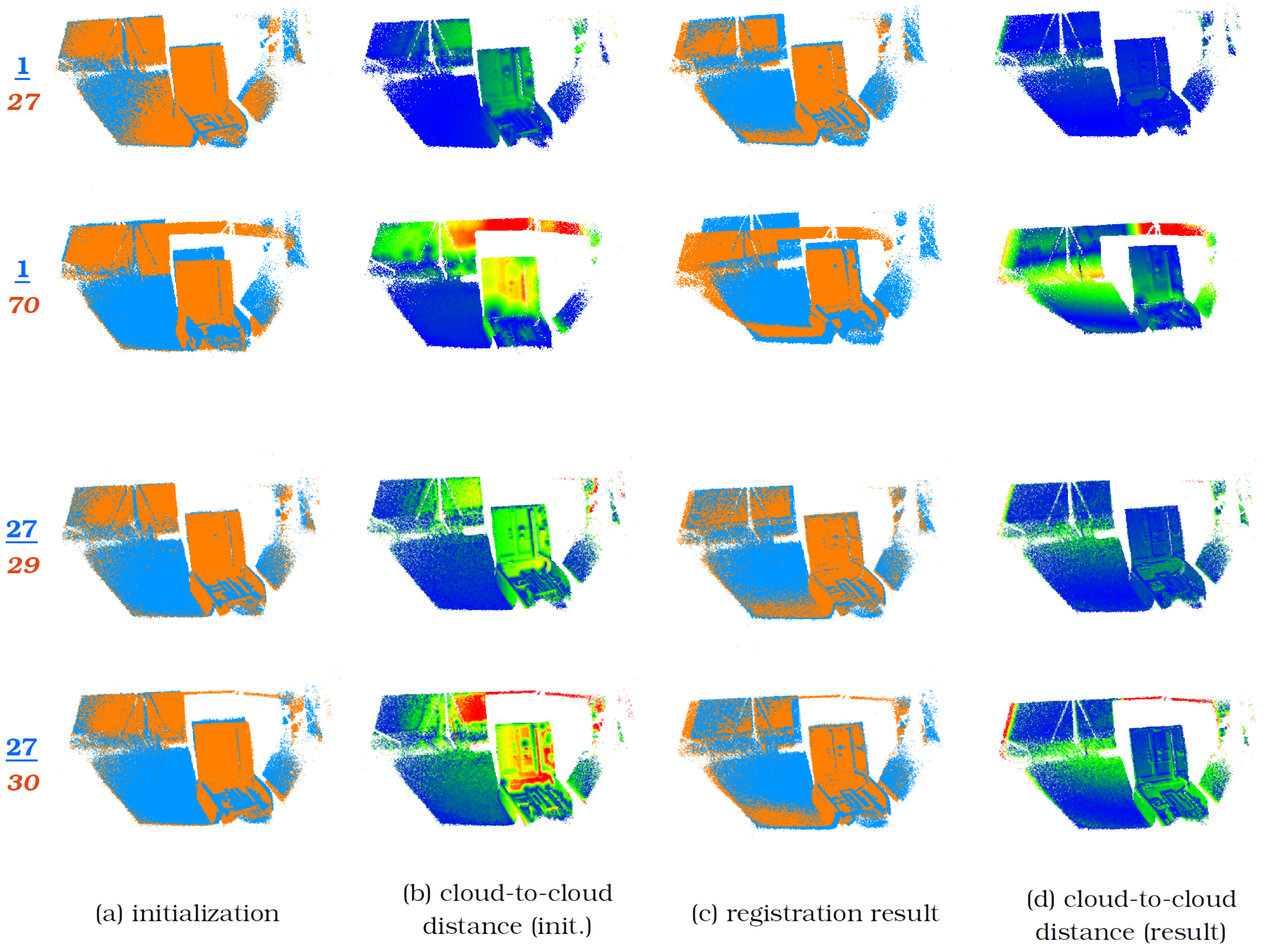

1
27
1
70
27
29
27
30
(a) initialization
(b) cloud-to-cloud distance (init.)
(c) registration result
(d) cloud-to-cloud distance (result)

Figure VI: Results of the experiment on the CoRBS dataset [7]: (a) initial misalignments (initialization); (b) cloud-to-cloud distance corresponding to the initial misalignment; references are shown in cyan and templates in orange; on the left, corresponding frame numbers are provided (in cyan for the reference frames and in orange for the template frames). (c) GA registration results; (d) cloud-to-cloud distance corresponding to the registration result; cloud-to-cloud distances are coded with a Blue<Green<Yellow<Red color scale; the saturation point (red value) amounts to 0.1 distance units.

count (the latter revealed in [2]). Other characteristic is that an optimal solution is represented by a position of one particular agent in the solution space.

References

[1] H. Bhatia, G. Norgard, V. Pascucci, and P.-T. Bremer. The Helmholtz-Hodge decomposition — a survey. *Transactions on Visualization and Computer Graphics*, 19(8):1386–1404, 2013. 2

[2] M. Gauci, T. Dodd, and R. Gro. Why GSA: a gravitational search algorithm is not genuinely based on the law of gravity. *Natural Computing*, 11(4):719–720, 2012. 6

[3] M. Khosravi, K. Khalili, and H. Amirabadi. Employing nelder-mead and gravitational search algorithm methods in point clouds registration process. *Procedia Technology*, 19:112–119, 2015. 5

[4] D. Koks. *Explorations in mathematical physics: the concepts behinds an elegant language*. Springer, 2006. 1

[5] D. W. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *SIAM Journal on Applied Mathematics*, 11(2):431–441, 1963. 3

[6] E. Rashedi, H. Nezamabadi-pour, and S. Saryazdi. GSA: A gravitational search algorithm. *Information Sciences*, 179(13):2232–2248, 2009. 5

[7] O. Wasenmüller, M. Meyer, and D. Stricker. CoRBS: Comprehensive rgb-d benchmark for slam using kinect v2. In *Winter Conference on Applications of Computer Vision (WACV)*, 2016. 3, 6

[8] Q.-Y. Zhou and V. Koltun. Dense scene reconstruction with points of interest. *ACM Transactions on Graphics (TOG)*, 32(4):112:1–112:8, 2013. 3, 5