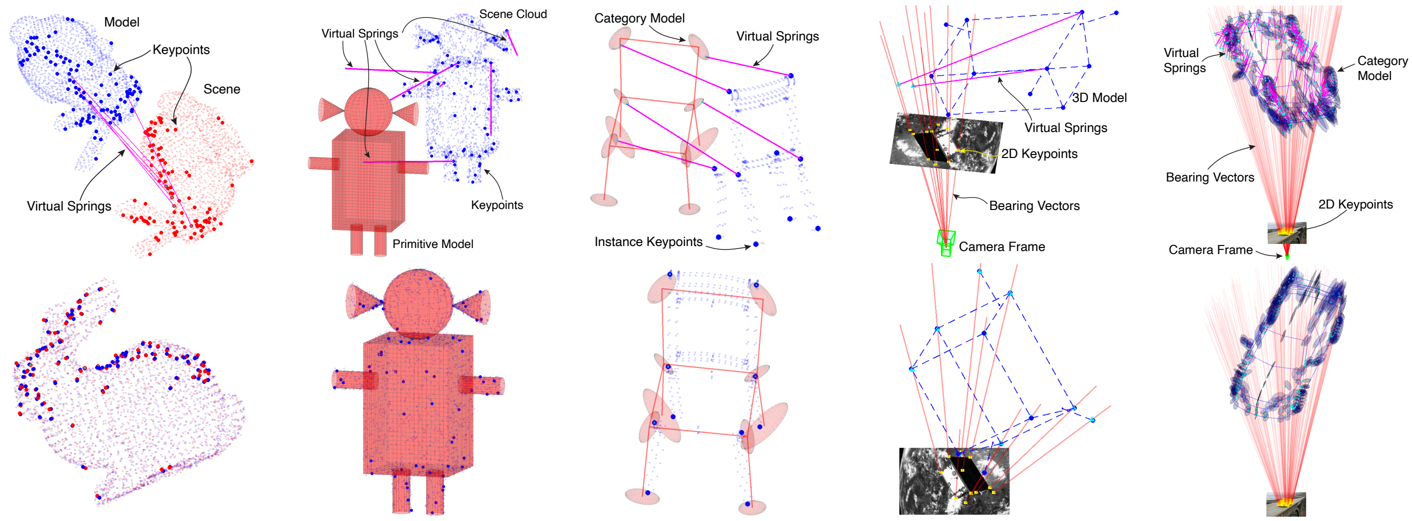
Dynamical Pose Estimation

Heng Yang
MIT

Chris Doran
University of Cambridge

Jean-Jacques Slotine
MIT



(a) Point Cloud Registration (b) Primitive Registration (c) Category Registration (d) Absolute Pose Estimation (e) Category APE

Figure 1: We propose *DynAMical Pose estimation* (DAMP), the first general and practical framework to perform pose estimation from 2D and 3D visual correspondences by simulating *rigid body dynamics* arising from *virtual springs and damping* (top row, magenta lines). DAMP almost always returns the *globally* optimal rigid transformation across five pose estimation problems (bottom row). (a) Point cloud registration using the Bunny dataset [16]; (b) Primitive registration using a robot model of spheres, planes, cylinders and cones; (c) Category registration using the chair category from the PASCAL3D+ dataset [51]; (d) Absolute pose estimation (APE) using the SPEED satellite dataset [45]; (e) Category APE using the FG3DCar dataset [34].

Abstract

We study the problem of aligning two sets of 3D geometric primitives given known correspondences. Our first contribution is to show that this primitive alignment *framework unifies five perception problems including point cloud registration, primitive (mesh) registration, category-level 3D registration, absolution pose estimation (APE), and category-level APE. Our second contribution is to propose DynAMical Pose estimation (*DAMP*), the first general and practical algorithm to solve primitive alignment problem by simulating* rigid body dynamics *arising from virtual springs and damping, where the springs span the* shortest distances *between corresponding primitives. We evaluate* DAMP *in simulated and real datasets across all five problems, and demonstrate (i)* DAMP *always converges to the* globally optimal *solution in the first three problems with 3D-3D correspondences; (ii) although* DAMP *sometimes converges to suboptimal solutions in the last two problems with 2D-3D correspondences, using a scheme for escaping local minima,* DAMP *always succeeds. Our third contribution is to demystify the surprising empirical performance of* DAMP *and formally prove a global convergence result in the case of point cloud registration by charactering local stability of the equilibrium points of the underlying dynamical system.*[1]

[1]Code: https://github.com/hankyang94/DAMP

1. Introduction

Consider the problem of finding the best *rigid* transformation (pose) to align two sets of *corresponding* 3D geometric *primitives* $\mathcal{X} = \{X_i\}_{i=1}^N$ and $\mathcal{Y} = \{Y_i\}_{i=1}^N$:

$$\min_{\boldsymbol{T} \in \mathrm{SE}(3)} \sum_{i=1}^{N} \mathrm{dist}\left(\boldsymbol{T} \otimes X_i, Y_i\right)^2, \tag{1}$$

where $\mathrm{SE}(3) \triangleq \left\{(\boldsymbol{R}, \boldsymbol{t}) : \boldsymbol{R} \in \mathrm{SO}(3), \boldsymbol{t} \in \mathbb{R}^3\right\}$[2] is the set of 3D rigid transformations (rotations and translations), $\boldsymbol{T} \otimes X$ denotes the action of a rigid transformation $\boldsymbol{T}$ on the primitive X, and $\mathrm{dist}\left(X, Y\right)$ is the *shortest* distance between two primitives X and Y. In particular, we focus on the following primitives:

1. *Point*: $P(\boldsymbol{x}) \triangleq \{\boldsymbol{x}\}$, where $\boldsymbol{x} \in \mathbb{R}^3$ is a 3D point;
2. *Line*: $L(\boldsymbol{x}, \boldsymbol{v}) \triangleq \{\boldsymbol{x} + \alpha \boldsymbol{v} : \alpha \in \mathbb{R}\}$, where $\boldsymbol{x} \in \mathbb{R}^3$ is a point on the line, and $\boldsymbol{v} \in \mathbb{S}^2$ is the unit direction;[3]
3. *Plane*: $H(\boldsymbol{x}, \boldsymbol{n}) \triangleq \{\boldsymbol{y} \in \mathbb{R}^3 : \boldsymbol{n}^\mathsf{T}(\boldsymbol{y} - \boldsymbol{x}) = 0\}$, where $\boldsymbol{x} \in \mathbb{R}^3$ is a point on the plane, and $\boldsymbol{n} \in \mathbb{S}^2$ is the unit normal that is perpendicular to the plane;

[2]$\mathrm{SO}(3) \triangleq \{\boldsymbol{R} \in \mathbb{R}^{3\times 3} : \boldsymbol{R}\boldsymbol{R}^\mathsf{T} = \boldsymbol{R}^\mathsf{T}\boldsymbol{R} = \mathbf{I}_3, \det(\boldsymbol{R}) = +1\}$ is the set of proper 3D rotations.

[3]$\mathbb{S}^{n-1} \triangleq \{\boldsymbol{v} \in \mathbb{R}^n : \|\boldsymbol{v}\| = 1\}$ is the set of n-D unit vectors.

4. *Sphere*: $S(\boldsymbol{x}, r) \triangleq \{\boldsymbol{y} \in \mathbb{R}^3 : \|\boldsymbol{y} - \boldsymbol{x}\|^2 = r^2\}$, where $\boldsymbol{x} \in \mathbb{R}^3$ is the center, and $r > 0$ is the radius;
5. *Cylinder*: $C(\boldsymbol{x}, \boldsymbol{v}, r) \triangleq \{\boldsymbol{y} \in \mathbb{R}^3 : \operatorname{dist}(\boldsymbol{y}, L(\boldsymbol{x}, \boldsymbol{v})) = r\}$, where $L(\boldsymbol{x}, \boldsymbol{v})$ (defined in 2) is the central axis of the cylinder, $r > 0$ is the radius, and $\operatorname{dist}(\boldsymbol{y}, L)$ is the orthogonal distance from point $\boldsymbol{y}$ to line L;
6. *Cone*: $K(\boldsymbol{x}, \boldsymbol{v}, \theta) \triangleq \{\boldsymbol{y} \in \mathbb{R}^3 : \boldsymbol{v}^\mathsf{T}(\boldsymbol{y} - \boldsymbol{x}) = \cos\theta \|\boldsymbol{y} - \boldsymbol{x}\|\}$, where $\boldsymbol{x} \in \mathbb{R}^3$ is the apex, $\boldsymbol{v} \in \mathbb{S}^2$ is the unit direction of the central axis pointing inside the cone, and $\theta \in (0, \frac{\pi}{2})$ is the half angle;
7. *Ellipsoid*: $E(\boldsymbol{x}, \boldsymbol{A}) \triangleq \{\boldsymbol{y} \in \mathbb{R}^3 : (\boldsymbol{y} - \boldsymbol{x})^\mathsf{T} \boldsymbol{A} (\boldsymbol{y} - \boldsymbol{x}) \leq 1\}$, where $\boldsymbol{x} \in \mathbb{R}^3$ is the center, and $\boldsymbol{A} \in \mathcal{S}^3_{++}$ is a positive definite matrix defining the principal axes.[4]

Problem (1), when specialized to the primitives 1-7, includes a broad class of fundamental perception problems concerning *pose estimation* from visual measurements, and finds extensive applications to object detection and localization [30, 39], motion estimation and 3D reconstruction [58, 56], and simultaneous localization and mapping [10, 52, 42].

In this paper, we consider five examples of problem (1), with graphical illustrations given in Fig. 1. Note that we restrict ourselves to the case when all correspondences $X_i \leftrightarrow Y_i, i = 1, \ldots, N$, are *known and correct*, for two reasons: (i) there are general-purpose algorithmic frameworks, such as RANSAC [19] and GNC [52, 3] that re-gain robustness to incorrect correspondences (*i.e. outliers*) once we have efficient solvers for the outlier-free problem (1); (ii) even when all correspondences are correct, problem (1) can be difficult to solve due to the non-convexity of the feasible set $\mathrm{SE}(3)$.

Example 1 (Point Cloud Registration [27, 53])**.** *Let* $X_i = P(\boldsymbol{x}_i)$ *and* $Y_i = P(\boldsymbol{y}_i)$ *in problem* (1)*, with* $\boldsymbol{x}_i, \boldsymbol{y}_i \in \mathbb{R}^3$*, point cloud registration seeks the best rigid transformation to align two sets of 3D points.*

Fig. 1(a) shows an instance of point cloud registration using the Bunny dataset [16], with bold blue and red dots being the *keypoints* $P(\boldsymbol{x}_i)$ and $P(\boldsymbol{y}_i)$, respectively. Point cloud registration commonly appears when one needs to align two or more Lidar or RGB-D scans acquired at different space and time [58], and in practice either hand-crafted [43] or deep-learned [13, 21, 57] feature descriptors are adopted to generate point-to-point correspondences.

However, in many cases it is challenging to obtain (in run time), or annotate (in training time), point-to-point correspondences (*e.g.*, it is much easier to tell a point lies on a plane than to precisely localize where it lies on the plane as in Fig. 1(b)). Moreover, it is well known that correspondences such as point-to-line and point-to-plane ones can lead to better convergence in algorithms such as ICP [7].

Recently, a growing body of research seeks to represent and approximate complicated 3D shapes using simple primitives such as cubes, cones, cylinders etc. to gain efficiency in storage and capability in assigning semantic meanings to different parts of a 3D shape [50, 20, 33]. These factors motivate the following primitive registration problem.

Example 2 (Primitive Registration [9, 33])**.** *Let* $X_i = P(\boldsymbol{x}_i), \boldsymbol{x} \in \mathbb{R}^3$*, be a 3D point, and let* Y_i *be any type of primitives among 1-7 in problem* (1)*, primitive registration seeks the best rigid transformation to align a set of 3D points to a set of 3D primitives.*

Fig. 1(b) shows an example where a semantically meaningful robot model is compactly represented as a collection of planes, cylinders, spheres and cones, while a noisy point cloud observation is aligned to it by solving problem (1).

Both Examples 1 and 2 require a known 3D model, either in the form of a clean point cloud or a collection of fixed primitives, which can be quite restricted. For example, in Fig. 1(c), imagine a robot has seen multiple *instances* of a chair and only stored a *deformable* model (shown in red) of the category "*chair*" in the form of a collection of *semantic uncertainty ellipsoids* (SUE), where the center of each ellipsoid keeps the *average* location of a semantic keypoint (*e.g.*, legs of a chair) while the orientation and size of the ellipsoid represent *intra-class variations* of that keypoint within the category (see Supplementary Material for details about how SUEs are computed from data). Now the robot sees an instance of a chair (shown in blue) that either it has never seen before, or it has seen but does not have access to a precise 3D model, and has to estimate the pose of the instance w.r.t. itself. In this situation, we formulate a *category-level 3D registration* using SUEs.

Example 3 (Category Registration [36, 11, 46])**.** *Let* $X_i = P(\boldsymbol{x}_i), \boldsymbol{x}_i \in \mathbb{R}^3$*, be a 3D point, and* $Y_i = E(\boldsymbol{y}_i, \boldsymbol{A}_i), \boldsymbol{y}_i \in \mathbb{R}^3, \boldsymbol{A}_i \in \mathcal{S}^3_{++}$*, be a SUE of a semantic keypoint, category registration seeks the best rigid transformation to align a point cloud to a set of category-level semantic keypoints.*

The above three Examples 1-3 demonstrate the flexibility of problem (1) in modeling pose estimation problems given 3D-3D correspondences. The next two examples show that pose estimation given 2D-3D correspondences (*i.e.*, *absolute pose estimation* (APE) or *perspective-n-points* (PnP)) can also be formulated in the form of problem (1). The crux is the insight that a 2D image keypoint is uniquely determined (assume camera intrinsics are known) by a so-called *bearing vector* that originates from the camera center and goes through the 2D keypoint on the imaging plane (*c.f.* Fig. 1(d)) [26].[5] Consequently, APE can be formulated as aligning the 3D model to a set of 3D bearing vectors.

[4] $\mathcal{S}^n, \mathcal{S}^n_+, \mathcal{S}^n_{++}$ denote the set of real $n \times n$ symmetric, positive semidefinite, and positive definite matrices, respectively.

[5] Similarly, a 2D line on the imaging plane can be uniquely determined by a 3D plane containing two bearing vectors that intersects two 2D points

Example 4 (Absolute Pose Estimation [30, 1]). *Let* $X_i = P(\boldsymbol{x}_i), \boldsymbol{x}_i \in \mathbb{R}^3$*, be a 3D point, and* $Y_i = L(\mathbf{0}, \boldsymbol{v}_i)$*,* $\boldsymbol{v}_i \in \mathbb{S}^2$*, be the bearing vector of a 2D keypoint (the camera center is* $\mathbf{0} \in \mathbb{R}^3$*), APE seeks to find the best rigid transformation to align a 3D point cloud to a set of bearing vectors.*

Fig. 1(d) shows an example of aligning a satellite wire-frame model to a set of 2D keypoint detections. Similarly, by allowing the 3D model to be a collection of SUEs, we can generalize Example 4 to category-level APE.

Example 5 (Category Absolute Pose Estimation [55, 34]). *Let* $X_i = E(\boldsymbol{x}_i, \boldsymbol{A}_i)$*,* $\boldsymbol{x}_i \in \mathbb{R}^3, \boldsymbol{A}_i \in \mathcal{S}^3_{++}$*, be a SUE of a category-level semantic keypoint, and* $Y_i = L(\mathbf{0}, \boldsymbol{v}_i)$*,* $\boldsymbol{v}_i \in \mathbb{S}^2$*, be the bearing vector of a 2D keypoint, category APE seeks to find the best rigid transformation to align a 3D category to the 2D keypoints of an instance.*

Fig. 1(e) shows an example of estimating the pose of a car using a category-level collection of SUEs. Strictly speaking, Example 2 contains Examples 1, 3 and 4, but we separate them because they have different applications.

Related Work. To the best of our knowledge, this is the first time that the five seemingly different examples are formulated under the same framework. We shall briefly discuss existing methods for solving them. Point cloud registration (Example 1) can be solved in closed form using singular value decomposition [27, 5]. A comprehensive review of recent advances in point cloud registration, especially on dealing with outliers, can be found in [58]. The other four examples, however, do not admit closed-form solutions. Primitive registration (Example 2) in the case of point-to-point, point-to-line and point-to-plane correspondences (referred to as *mesh registration* [56]) can be solved globally using branch-and-bound [38] and semidefinite relaxations [9], hence, is relatively slow. Further, there are no solvers that can solve primitive registration including point-to-sphere, point-to-cylinder and point-to-cone correspondences with global optimality guarantees. The absolute pose estimation problem (Example 4) has been a major line of research in computer vision, and there are several global solvers based on Grobner bases [30] and convex relaxations [1, 44]. For category-level registration and APE (Example 3 and 5), most existing methods formulate them as simultaneously estimating the *shape coefficients* and the camera pose, *i.e.*, they treat the unknown instance model as a *linear combination* of category templates (known as the *active shape model* [15]) and seek to estimate the linear coefficients as well as the camera pose. Works in [25, 40, 34] solve the joint optimization by alternating the estimation of the shape coefficients and the estimation of the camera pose, thus requiring a good initial guess for convergence. Zhou *et al*. [59, 60] developed a convex relaxation technique to solve category APE with a *weak perspective* camera model and showed efficient and accurate results. Yang and Carlone [55] later showed that the convex relaxation in [59, 60] is less tight than the one they developed based on sums-of-squares (SOS) relaxations. However, the SOS relaxation in [55] leads to large semidefinite programs (SDP) that cannot be solved efficiently at present time. Very recently, with the advent of machine learning, many researchers resort to deep networks that regress the 3D shape and the camera pose directly from 2D images [11, 31, 49]. We refer the interested reader to [49, 31, 29, 32] and references therein for details of this line of research.

Contribution. Our first contribution, as described in the previous paragraphs, is to *unify* five pose estimation problems under the general framework of aligning two sets of geometric primitives. While such proposition has been presented in [9, 38] for point-to-point, point-to-line and point-to-plane correspondences, generalizing it to a broader class of primitives such as cylinders, cones, spheres, and ellipsoids, and showing its modeling capability in category-level registration (using the idea of SUEs) and pose estimation given 2D-3D correspondences has never been done. Our second contribution is to develop a simple, general, intuitive, yet effective and efficient framework to solve all five examples by simulating *rigid body dynamics*. As we will detail in Section 3, the general formulation (1) allows us to model $\mathcal{Y}$ as a *fixed* rigid body and $\mathcal{X}$ as a *moving* rigid body with $\boldsymbol{T}$ representing the relative pose of $\mathcal{X}$ w.r.t. $\mathcal{Y}$. We then place *virtual* springs between points in X_i and Y_i that attain the shortest distance $\mathrm{dist}(\boldsymbol{T} \otimes X_i, Y_i)$ given $\boldsymbol{T}$. The virtual springs naturally exert forces under which $\mathcal{X}$ is pulled towards $\mathcal{Y}$ with motion governed by Newton-Euler rigid body dynamics, and moreover, the *potential energy* of the dynamical system coincides with the objective function of problem (1). By assuming $\mathcal{X}$ moves in an environment with constant damping, the dynamical system will eventually arrive at an *equilibrium* point, from which a solution to problem (1) can be obtained. Our construction of such a dynamical system is inspired by recent work on physics-based registration [22, 23, 28], but goes much beyond them in showing that simulating dynamics can solve broader and more challenging pose estimation problems other than just point cloud registration. We name our approach *DynAMical Pose estimation* (DAMP), which we hope to stimulate the connection between computer vision and dynamical systems. We evaluate DAMP on both simulated and real datasets (Section 4) and demonstrate (i) DAMP always returns the *globally optimal* solution to Examples 1-3 with 3D-3D correspondences; (ii) although DAMP converges to suboptimal solutions given 2D-3D correspondences (Examples 4-5) with very low probability ($< 1\%$), using a sim-

on the imaging plane. Therefore, problem (1) can also accommodate point-to-line correspondences commonly seen in the literature of perspective-n-points-and-lines (PnPL) [1, 35].

ple scheme for escaping local minima, DAMP almost always succeeds. Our last contribution (Section 3.2) is to (partially) demystify the surprisingly good empirical performance of DAMP and prove a nontrivial global convergence result in the case of point cloud registration, by charactering the local stability of equilibrium points. Extending the analysis to other examples remains open.

2. Geometry and Dynamics

In this section, we present two key results underpinning the DAMP algorithm. One is geometric and concerns computing the shortest distance between two geometric primitives, the other is dynamical and concerns simulating Newton-Euler dynamics of an N-primitive system.

2.1. Geometry

In view of Black-Box Optimization [37], the question that needs to be answered before solving problem (1) is to *evaluate* the cost function at a given $\boldsymbol{T} \in \mathrm{SE}(3)$, because the $\mathrm{dist}(X, Y)$ function is itself a minimization. Although in the simplest case of point cloud registration, $\mathrm{dist}(X, Y) = \|\boldsymbol{x} - \boldsymbol{y}\|$ can be written analytically, the following theorem states that in general $\mathrm{dist}(\cdot, \cdot)$ may require nontrivial computation.

Theorem 6 (Shortest Distance Pair)**.** *Let X and Y be two primitives of types 1-7, define $(X, Y)_{\mathrm{p}}$ as the set of points that attain the shortest distance between X and Y,* i.e.*,*

$$(X, Y)_{\mathrm{p}} \triangleq \underset{(\boldsymbol{x},\boldsymbol{y})\in X\times Y}{\arg\min} \|\boldsymbol{x} - \boldsymbol{y}\|. \tag{2}$$

In the following cases, $(X, Y)_{\mathrm{p}}$ (and hence $\mathrm{dist}(X, Y)$) can be computed either analytically or numerically.

1. *Point-Point* (PP)*, $X = P(\boldsymbol{x})$, $Y = P(\boldsymbol{y})$:*

$$(X, Y)_{\mathrm{p}} = \{(\boldsymbol{x}, \boldsymbol{y})\}. \tag{3}$$

2. *Point-Line* (PL)*, $X = P(\boldsymbol{x})$, $Y = L(\boldsymbol{y}, \boldsymbol{v})$:*

$$(X, Y)_{\mathrm{p}} = \{(\boldsymbol{x}, \boldsymbol{y} + \alpha\boldsymbol{v}) : \alpha = \boldsymbol{v}^{\mathsf{T}}(\boldsymbol{x} - \boldsymbol{y})\}, \tag{4}$$

where $\boldsymbol{y} + \alpha\boldsymbol{v}$ is the projection of $\boldsymbol{x}$ onto the line.

3. *Point-Plane* (PH)*, $X = P(\boldsymbol{x})$, $Y = H(\boldsymbol{y}, \boldsymbol{n})$:*

$$(X, Y)_{\mathrm{p}} = \{(\boldsymbol{x}, \boldsymbol{x} + \alpha\boldsymbol{n}) : \alpha = \boldsymbol{n}^{\mathsf{T}}(\boldsymbol{y} - \boldsymbol{x})\}, \tag{5}$$

where $\boldsymbol{x} + \alpha\boldsymbol{n}$ is the projection of $\boldsymbol{x}$ onto the plane.

4. *Point-Sphere* (PS)*, $X = P(\boldsymbol{x})$, $Y = S(\boldsymbol{y}, r)$:*

$$(X, Y)_{\mathrm{p}} = \begin{cases} \{(\boldsymbol{x}, \boldsymbol{z}) : \boldsymbol{z} \in S(\boldsymbol{y}, r)\} & \textit{if } \boldsymbol{x} = \boldsymbol{y} \\ \{(\boldsymbol{x}, \boldsymbol{y} + r\boldsymbol{v}) : \boldsymbol{v} = \frac{\boldsymbol{x}-\boldsymbol{y}}{\|\boldsymbol{x}-\boldsymbol{y}\|}\} & \textit{otherwise} \end{cases}, \tag{6}$$

where if $\boldsymbol{x}$ coincides with the center of the sphere, then the entire sphere achieves the shortest distance, while otherwise $\boldsymbol{y} + r\boldsymbol{v}$, the projection of $\boldsymbol{x}$ onto the sphere, achieves the shortest distance.

5. *Point-Cylinder* (PC)*, $X = P(\boldsymbol{x})$, $Y = C(\boldsymbol{y}, \boldsymbol{v}, r)$:*

$$(X, Y)_{\mathrm{p}} = \begin{cases} \{(\boldsymbol{x}, \hat{\boldsymbol{y}} + r\boldsymbol{u}) : \boldsymbol{u} \in \mathbb{S}^2, \boldsymbol{u} \perp \boldsymbol{v}\} & \textit{if } \boldsymbol{x} = \hat{\boldsymbol{y}} \\ \{(\boldsymbol{x}, \hat{\boldsymbol{y}} + r\frac{\boldsymbol{x}-\hat{\boldsymbol{y}}}{\|\boldsymbol{x}-\hat{\boldsymbol{y}}\|})\} & \textit{otherwise} \end{cases}, \tag{7}$$

where $\hat{\boldsymbol{y}} \triangleq \boldsymbol{y} + \alpha\boldsymbol{v}, \alpha = \boldsymbol{v}^{\mathsf{T}}(\boldsymbol{x} - \boldsymbol{y})$, is the projection of $\boldsymbol{x}$ onto the central axis $L(\boldsymbol{y}, \boldsymbol{v})$. If $\boldsymbol{x}$ lies on the central axis, then any point on the circle that passes through $\boldsymbol{x}$ and is orthogonal to $\boldsymbol{v}$ achieves the shortest distance, otherwise, the projection of $\boldsymbol{x} - \hat{\boldsymbol{y}}$ onto the cylinder achieves the shortest distance.

6. *Point-Cone* (PK)*, $X = P(\boldsymbol{x})$, $Y = K(\boldsymbol{y}, \boldsymbol{v}, \theta)$:*

$$(X, Y)_{\mathrm{p}} = \begin{cases} \{(\boldsymbol{x}, \boldsymbol{y})\} & \textit{if } \boldsymbol{v}^{\mathsf{T}}\boldsymbol{x}_y \leq -\|\boldsymbol{x}_y\| \sin\theta \\ \{(\boldsymbol{x}, \boldsymbol{y} + \|\boldsymbol{x}_y\| \cos\theta\boldsymbol{u}) : {}^{\boldsymbol{u}\in\mathbb{S}^2,}_{\boldsymbol{u}^{\mathsf{T}}\boldsymbol{v}=\cos\theta}\} & \textit{if } \frac{\boldsymbol{x}_y}{\|\boldsymbol{x}_y\|} = \boldsymbol{v} \\ \{(\boldsymbol{x}, \boldsymbol{y} + \alpha\boldsymbol{w}) : \alpha = \boldsymbol{w}^{\mathsf{T}}\boldsymbol{x}_y\} & \textit{otherwise} \end{cases}, \tag{8}$$

where $\boldsymbol{x}_y \triangleq \boldsymbol{x} - \boldsymbol{y}$, $\boldsymbol{w} \triangleq \boldsymbol{R}_\theta\boldsymbol{v}$, with $\boldsymbol{R}_\theta \in \mathrm{SO}(3)$ being the 3D rotation matrix of axis $\boldsymbol{v} \times \frac{\boldsymbol{x}_y}{\|\boldsymbol{x}_y\|}$ and angle θ.[6] *The first condition $\boldsymbol{v}^{\mathsf{T}}\boldsymbol{x}_y \leq -\|\boldsymbol{x}_y\| \sin\theta$ corresponds to $\boldsymbol{x}$ in the dual cone of K and the apex $\boldsymbol{y}$ achieves the shortest distance. The second condition corresponds to $\boldsymbol{x}$ lies on the central axis and inside the cone, in which case an entire circle on the surface of the cone achieves the shortest distance. Under the last condition, a unique projection of $\boldsymbol{x}$ onto (an extreme ray of) the cone achieves the shortest distance.*

7. *Point-Ellipsoid* (PE)*, $X = P(\boldsymbol{x})$, $Y = E(\boldsymbol{y}, \boldsymbol{A})$:*

$$(X, Y)_{\mathrm{p}} = \begin{cases} \{(\boldsymbol{x}, \boldsymbol{x})\} & \textit{if } \boldsymbol{x} \in E \\ \{(\boldsymbol{x}, (\lambda\boldsymbol{A} + \mathbf{I})^{-1}\boldsymbol{x}_y + \boldsymbol{y}) : {}^{g(\lambda)=0,}_{\lambda>0}\} & \textit{otherwise} \end{cases}, \tag{9}$$

where $\boldsymbol{x}_y \triangleq \boldsymbol{x} - \boldsymbol{y}$, and $g(\lambda)$ is a univariate function whose expression is given in Supplementary Material. If $\boldsymbol{x}$ belongs to the ellipsoid, then the shortest distance is zero. Otherwise, there is a unique point on the surface of the ellipsoid that achieves the shortest distance, obtained by finding the root of the function $g(\lambda)$.

8. *Ellipsoid-Line* (EL)*, $X = E(\boldsymbol{x}, \boldsymbol{A})$, $Y = L(\boldsymbol{y}, \boldsymbol{v})$:*

$$(X, Y)_{\mathrm{p}} = \begin{cases} \{(\boldsymbol{y} + \alpha\boldsymbol{v}, \boldsymbol{y} + \alpha\boldsymbol{v}) : \alpha \in [\alpha_1, \alpha_2]\} & \textit{if } \Delta \geq 0 \\ \{(\boldsymbol{z}(\lambda), \boldsymbol{y} + \alpha(\lambda)\boldsymbol{v}) : {}^{g(\lambda)=0,}_{\lambda>0}\} & \textit{otherwise} \end{cases}, \tag{10}$$

where $\boldsymbol{y}_x \triangleq \boldsymbol{y} - \boldsymbol{x}$, and the expressions of $\Delta, \alpha_{1,2}, \boldsymbol{z}(\lambda), \alpha(\lambda), g(\lambda)$ are given in Supplementary Material. Intuitively, the discriminant Δ decides when the line intersects with the ellipsoid. If there is nonempty intersection, then an entire line segment

[6] $\boldsymbol{a} \times \boldsymbol{b}$ denotes the cross product of $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^3$. Given an axis-angle representation $(\boldsymbol{v}, \theta)$ of a 3D rotation, the rotation matrix can be computed as $\boldsymbol{R} = \cos\theta\mathbf{I}_3 + \sin\theta[\boldsymbol{v}]_\times + (1 - \cos\theta)\boldsymbol{v}\boldsymbol{v}^{\mathsf{T}}$, where $[\boldsymbol{v}]_\times$ is the skew-symmetric matrix associated with $\boldsymbol{v}$ such that $\boldsymbol{v} \times \boldsymbol{a} \equiv [\boldsymbol{v}]_\times\boldsymbol{a}$ [54].

(determined by $\alpha_{1,2}$) achieves shortest distance zero. Otherwise, the unique shortest distance pair can be obtained by first finding the root λ of a univariate function $g(\lambda)$ and then substituting λ into $\boldsymbol{z}(\lambda)$ and $\alpha(\lambda)$.

A detailed proof of Theorem 6 is in Supplementary Material, with numerical methods for finding roots of $g(\lambda)$.

Remark 7 (Distance)**.** *The* $\operatorname{dist}(\cdot,\cdot)$ *function defined in (2) is inherited from convex analysis [17] and is appropriate for problems in this paper. However, it can be ill-defined for,* e.g., *aligning a pyramid to a sphere. A potentially better distance function would be the* Hausdorff distance *[41], but it is much more complicated to compute.*

2.2. N-Primitive Rigid Body Dynamics

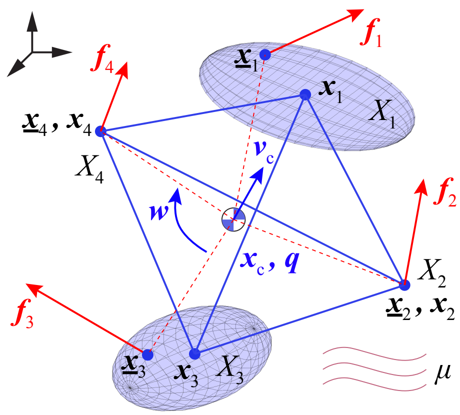


Figure 2: Example of an N-primitive rigid-body dynamical system with $N=4$. $X_{1,3}$ are ellipsoids, $X_{2,4}$ are points.

In this paper we consider a rigid body consisting of N primitives $\{X_i\}_{i=1}^N$ moving in an environment with constant damping coefficient $\mu > 0$, and each primitive X_i has a *pointed* mass located at $\boldsymbol{x}_i \in \mathbb{R}^3$ w.r.t. a *global* coordinate frame (Fig. 2). Assume there is an external force $\boldsymbol{f}_i \in \mathbb{R}^3$ acting on each primitive at location $\underline{\boldsymbol{x}}_i \in \mathbb{R}^3, i = 1,\dots,N$. Note that we do not restrict $\boldsymbol{x}_i = \underline{\boldsymbol{x}}_i$, *i.e.* the external force is not required to act at the location of the pointed mass. For example, when X_i is an ellipsoid, $\boldsymbol{x}_i$ is the center of the ellipsoid, but $\underline{\boldsymbol{x}}_i$ can be any point on the surface of or inside the ellipsoid (*c.f.* Fig. 2). We assume each primitive has equal mass $m_i = m, i = 1,\dots,N$, such that the *center of mass* of the N-primitive system is at $\bar{\boldsymbol{x}} \triangleq \frac{1}{N}\sum_{i=1}^N \boldsymbol{x}_i$ (in the global frame). The next proposition states the system of equations governing the motion of the N-primitive system.

Proposition 8 (N-Primitive Dynamics)**.** *Let* $\boldsymbol{s}(t) \triangleq [\boldsymbol{x}_c^\mathsf{T}, \boldsymbol{q}^\mathsf{T}, \boldsymbol{v}_c^\mathsf{T}, \boldsymbol{\omega}^\mathsf{T}]^\mathsf{T} \in \mathbb{R}^{13}$ *be the state space of the N-primitive rigid body in Fig. 2, where* $\boldsymbol{x}_c \in \mathbb{R}^3$ *denotes the position of the center of mass in the global coordinate frame,* $\boldsymbol{q} \in \mathbb{S}^3$ *denotes the unit quaternion representing the rotation from the body frame to the global frame,* $\boldsymbol{v}_c \in \mathbb{R}^3$ *denotes the translational velocity of the center of mass, and* $\boldsymbol{\omega} \in \mathbb{R}^3$ *denotes the angular velocity of the rigid body w.r.t. the center of mass. At $t=0$, assume*

$$\boldsymbol{x}_c(0) = \bar{\boldsymbol{x}},\ \boldsymbol{q}(0) = [0,0,0,1]^\mathsf{T},\ \boldsymbol{v}_c(0) = \mathbf{0},\ \boldsymbol{\omega}(0) = \mathbf{0}, \tag{11}$$

so that the body frame coincides with the global frame ($\boldsymbol{q}(0)$ is the identity rotation). Call $\boldsymbol{x}_{r_i} \triangleq \boldsymbol{x}_i - \bar{\boldsymbol{x}}$ *the relative position of* $\boldsymbol{x}_i$ *w.r.t. the center of mass expressed in the body frame (a constant value w.r.t. time), then under the external forces* $\boldsymbol{f}_i$ *acted at locations* $\underline{\boldsymbol{x}}_i$*, expressed in global frame, the equations of motion of the dynamical system are*

$$\dot{\boldsymbol{s}}(t) = \mathcal{F}(\boldsymbol{s}; \boldsymbol{f}_i, \underline{\boldsymbol{x}}_i, \mu) = \begin{cases} \dot{\boldsymbol{x}}_c = \boldsymbol{v}_c \\ \dot{\boldsymbol{q}} = \frac{1}{2}\boldsymbol{q} \odot \tilde{\boldsymbol{\omega}} \\ \dot{\boldsymbol{v}}_c := \boldsymbol{a}_c = \frac{1}{M}\boldsymbol{f} \\ \dot{\boldsymbol{\omega}} := \boldsymbol{\alpha} = \boldsymbol{J}^{-1}(\boldsymbol{\tau} - \boldsymbol{\omega} \times \boldsymbol{J}\boldsymbol{\omega}) \end{cases}, \tag{12}$$

where $\tilde{\boldsymbol{\omega}} \triangleq [\boldsymbol{\omega}^\mathsf{T}, 0]^\mathsf{T} \in \mathbb{R}^4$ *is the homogenization of* $\boldsymbol{\omega}$*, "$\odot$" denotes the quaternion product [54],* $M \triangleq Nm$ *is the total mass of the system,* $\boldsymbol{f}$ *is the total external force*

$$\boldsymbol{f} = \sum_{i=1}^N \overbrace{\boldsymbol{f}_i - \mu m(\boldsymbol{v}_c + \boldsymbol{R}_q(\boldsymbol{\omega} \times \boldsymbol{x}_{r_i}))}^{:=\boldsymbol{f}_i'}, \tag{13}$$

with $\boldsymbol{R}_q \in \mathrm{SO}(3)$ *being the unique rotation matrix associated with the quaternion* $\boldsymbol{q}$*,* $\boldsymbol{J}$ *is the moment of inertia* $\boldsymbol{J} \triangleq -m\sum_{i=1}^N [\boldsymbol{x}_{r_i}]_\times^2 \in \mathcal{S}_{++}^3$ *expressed in the body frame, and* $\boldsymbol{\tau}$ *is the total torque*

$$\boldsymbol{\tau} = \sum_{i=1}^N \boldsymbol{R}_q^\mathsf{T}(\underline{\boldsymbol{x}}_i - \boldsymbol{x}_c) \times (\boldsymbol{R}_q^\mathsf{T}\boldsymbol{f}_i'), \tag{14}$$

in the body frame ($\boldsymbol{R}_q^\mathsf{T}$ rotates vectors to body frame).

The proof of Proposition 8 follows directly from [6].

Remark 9 (Unbounded Primitives)**.** *In this paper, it suffices to consider bounded primitives (ellipsoids, points) in the N-primitive system. For an unbounded primitive (*e.g.*, lines, planes), it remains open how to distribute its mass. A simple idea is to place all its mass m_i at the point of contact* $\underline{\boldsymbol{x}}_i$*.*

3. Dynamical Pose Estimation

3.1. Overview of DAMP

The idea in DAMP is to treat $\mathcal{X}$ as the N-primitive rigid body in Fig. 2, and treat $\mathcal{Y}$ as a set of primitives in the *global frame* that stay fixed and generate external forces to $\mathcal{X}$, *i.e.*, each primitive Y_i applies an external force $\boldsymbol{f}_i$ on X_i at location $\underline{\boldsymbol{x}}_i$ (red arrows in Fig. 2). Although this idea is inspired by related works [22, 23, 28], our construction of the forces significantly differ from them in two aspects: (i) we place a *virtual spring*, with coefficient k, between each pair of corresponding primitives (X_i, Y_i);[7] (ii) the two endpoints of the virtual spring are found using Theorem 6 so that the virtual spring spans the *shortest distance* between X_i and Y_i. With this, we have the following lemma.

Lemma 10 (Potential Energy)**.** *If the virtual spring has its two endpoints located at the shortest distance pair* $(\boldsymbol{T} \otimes X_i, Y_i)_{\mathrm{p}}$ *for any* $\boldsymbol{T}$*, and the spring has constant coefficient* $k = 2$*, then the cost function of problem* (1) *is equal to the potential energy of the dynamical system.*

We now state the DAMP algorithm (Algorithm 1). The input to DAMP is two sets of geometric primitives as in problem (1). In particular, we require the (X_i, Y_i) pair to be one of the seven types listed in Theorem 6, which encapsulate Examples 1-5. DAMP starts by computing the center of mass $\bar{\boldsymbol{x}}$, the relative positions $\boldsymbol{x}_{\mathrm{r}_i}$, and the moment of inertia $\boldsymbol{J}$ (line 4) using the location of the pointed mass $\boldsymbol{x}_i$ of each primitive in $\mathcal{X}$ (since X_i is either a point or an ellipsoid among Examples 1-5, $\boldsymbol{x}_i$ is well defined as in Fig. 2). Then DAMP computes the Cholesky factorization of $\boldsymbol{J}$ and stores the lower-triangular Cholesky factor $\boldsymbol{L}$ (line 5), which will later be used to compute the angular acceleration $\boldsymbol{\alpha}$ in eq. (12).[8] In line 7, the simulation is initialized at $\boldsymbol{s}_0$ as in (11), which basically states that $\mathcal{X}$ starts at rest without any initial speed. At each iteration of the main loop, DAMP first computes a shortest distance pair $(\underline{\boldsymbol{x}}_i, \underline{\boldsymbol{y}}_i)$ between the fixed Y_i and the X_i at current state $\boldsymbol{s}$, denoted as $X_i(\boldsymbol{s})$ (line 12). With the shortest distance pair $(\underline{\boldsymbol{x}}_i, \underline{\boldsymbol{y}}_i)$, DAMP spawns an instantaneous virtual spring between X_i and Y_i with endpoints at $\underline{\boldsymbol{x}}_i$ and $\underline{\boldsymbol{y}}_i$, leading to a virtual spring force $\boldsymbol{f}_i = k(\underline{\boldsymbol{y}}_i - \underline{\boldsymbol{x}}_i)$ (line 13). Then DAMP computes the time derivative of the state $\dot{\boldsymbol{s}}$ using eqs. (12)-(14) (line 15). If $\|\dot{\boldsymbol{s}}\|$ is smaller than the predefined threshold ε, then the dynamical system has reached an equilibrium point and the simulation stops (line 23). Otherwise, DAMP updates the state of the dynamical system, with proper *renormalization* on $\boldsymbol{q}$ to ensure a valid 3D rotation (line 27). The initial pose of $\mathcal{X}$ is $(\bar{\boldsymbol{x}}, \mathbf{I}_3)$, and the final pose of $\mathcal{X}$ is $(\boldsymbol{x}_{\mathrm{c}}, \boldsymbol{R}_q)$, therefore, DAMP returns the alignment $\boldsymbol{T}$ that transforms $\mathcal{X}$ from the initial state to the final state (line 30): $\boldsymbol{R} = \boldsymbol{R}_q$, $\boldsymbol{t} = \boldsymbol{x}_{\mathrm{c}} - \boldsymbol{R}_q \bar{\boldsymbol{x}}$.

Escape local minima. The DAMP framework allows a simple scheme for escaping suboptimal solutions. If the boolean flag EscapeMinimum is True, then each time the system reaches an equilibrium point, DAMP computes the potential energy of the system (which is the cost function of (1) by Lemma 10), stores the energy and state in $\mathcal{C}$, $\mathcal{S}$, and *randomly perturbs* the derivative of the state (imagine a virtual "hammering" on $\mathcal{X}$, line 20). After executing the EscapeMinimum scheme for a number of $T_{\max}$ trials, DAMP uses the state with *minimum* potential energy (line 29) to compute the final solution $\boldsymbol{T}$.

[7] Previous works [22, 23, 28] use gravitational and electrostatic forces between two point clouds, under which the potential energy of the dynamical system is not equivalent to the objective function of (1).

[8] One can also invert $\boldsymbol{J}$ directly since $\boldsymbol{J}$ is a 3×3 small matrix.

Algorithm 1: DAMP

1 **Input:** Primitives $\mathcal{X} = \{X_i\}_{i=1}^N$ and $\mathcal{Y} = \{Y_i\}_{i=1}^N$; damping $\mu > 0$ (default: $\mu = 2$); mass $m > 0$ (default: $m = 1$); spring coefficient $k > 0$ (default: $k = 2$); boolean: EscapeMinimum (default: False); number of trials $T_{\max} > 0$ (default: $T_{\max} = 5$); equilibrium threshold $\varepsilon > 0$ (default: $\varepsilon = 10^{-6}$); step size $dt > 0$ (default: $dt = 0.3$); maximum number of steps $K_{\max}$ (default: $K_{\max} = 10^3$)

2 **Output:** an estimate $\boldsymbol{T} \in \mathrm{SE}(3)$ to problem (1)

3 % Compute N-primitive quantities of $\mathcal{X}$

4 $\bar{\boldsymbol{x}} = \frac{\sum_{i=1}^N \boldsymbol{x}_i}{N}, \boldsymbol{x}_{\mathrm{r}_i} = \boldsymbol{x}_i - \bar{\boldsymbol{x}}, \boldsymbol{J} = -m \sum_{i=1}^{N} [\boldsymbol{x}_{\mathrm{r}_i}]_\times^2$

5 $\boldsymbol{J} = \boldsymbol{L}\boldsymbol{L}^\mathsf{T}$, with $\boldsymbol{L}$ lower triangular

6 % Initialization

7 $\boldsymbol{s} = \boldsymbol{s}_0$ as in eq. (11)

8 % Simulate dynamics

9 **if** EscapeMinimum **then** $T = 0, \mathcal{S} = \emptyset, \mathcal{C} = \emptyset$

10 **for** $j = 1, \ldots, K_{\max}$ **do**

11   % Compute shortest distance pair (Theorem 6)

12     $(\underline{\boldsymbol{x}}_i, \underline{\boldsymbol{y}}_i) \in (X_i(\boldsymbol{s}), Y_i)_{\mathrm{p}}, i = 1, \ldots, N$

13     $\boldsymbol{f}_i = k(\underline{\boldsymbol{y}}_i - \underline{\boldsymbol{x}}_i), i = 1, \ldots, N$

14   % Compute derivative of state (Proposition 8)

15     $\dot{\boldsymbol{s}} = \mathcal{F}(\boldsymbol{s}; \boldsymbol{f}_i, \underline{\boldsymbol{x}}_i, \mu)$, with $\boldsymbol{J} = \boldsymbol{L}\boldsymbol{L}^\mathsf{T}$ in (12)

16   % Check equilibrium

17   **if** $\|\dot{\boldsymbol{s}}\| < \varepsilon$ **then**

18   **if** EscapeMinimum *and* $T \leq T_{\max}$ **then**

19             $\mathcal{S} = \mathcal{S} \cup \boldsymbol{s}, \mathcal{C} = \mathcal{C} \cup \frac{k}{2} \sum_{i=1}^N \|\underline{\boldsymbol{y}}_i - \underline{\boldsymbol{x}}_i\|^2$

20             $\dot{\boldsymbol{s}} \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{13})$ % Random perturbation

21             $T = T + 1$

22   **else**

23   **break**

24   **end**

25   **end**

26   % Update state with quaternion correction

27     $\boldsymbol{s} = \boldsymbol{s} + dt \cdot \dot{\boldsymbol{s}}, \quad \boldsymbol{s}(\boldsymbol{q}) \leftarrow \frac{\boldsymbol{s}(\boldsymbol{q})}{\|\boldsymbol{s}(\boldsymbol{q})\|}$

28 **end**

29 **if** EscapeMinimum **then** $\boldsymbol{s} = \mathcal{S}(\arg\min \mathcal{C})$

30 **Return:** $\boldsymbol{T} = (\boldsymbol{R}_q, \boldsymbol{x}_{\mathrm{c}} - \boldsymbol{R}_q \bar{\boldsymbol{x}})$

3.2. Global Convergence: Point Cloud Registration

Due to the external damping μ, DAMP is guaranteed to converge to an equilibrium point with $\dot{\boldsymbol{s}} = \mathbf{0}$, a result that is well-known from Lyapunov theory [47]. However, the system (12) may have many (even infinite) equilibrium points. Therefore, a natural question is: *Does* DAMP *converge to an* *equilibrium point that minimizes the potential energy of the system?* If the answer is affirmative, then by Lemma 10, we can guarantee that DAMP finds the global minimizer of problem (1). The next theorem establishes the global convergence of DAMP for point cloud registration.

Theorem 11 (Global Convergence)**.** *In problem* (1)*, let $\mathcal{X}$ and $\mathcal{Y}$ be two sets of 3D points under* generic configuration*.*

(i) The system (12) *has four equilibrium points ($\dot{\boldsymbol{s}} = \mathbf{0}$);*
(ii) One of the (optimal) equilibrium point minimizes the potential energy;
(iii) Three other spurious *equilibrium points differ from the optimal equilibrium point by a rotation of π;*
(iv) The spurious equilibrium points are locally unstable*.*

Therefore, DAMP *(Algorithm 1 with* EscapeMinimum = False*) is guaranteed to converge to the optimal equilibrium point.*

The proof of Theorem 11 is algebraically involved and is presented in the Supplementary Material. The condition "generic configuration" helps remove pathological cases such as when the 3D points are collinear and coplanar (examples given in Supplementary Material).

4. Experiments

We first show that DAMP always converges to the optimal solution given 3D-3D correspondences (Section 4.1), then we show the EscapeMinimum scheme helps escape suboptimal solutions given 2D-3D correspondences (Section 4.2).

4.1. 3D-3D: Empirical Global Convergence

Point Cloud Registration. We randomly sample $N = 100$ 3D points from $\mathcal{N}(\mathbf{0}, \mathbf{I}_3)$ to be $\mathcal{X}$, then generate $\mathcal{Y}$ by applying a random rigid transformation $(\boldsymbol{R}, \boldsymbol{t})$ to $\mathcal{X}$, followed by adding Gaussian noise $\mathcal{N}(\mathbf{0}, 0.01^2\mathbf{I}_3)$. We run DAMP without EscapeMinimum, and compare its estimated pose w.r.t. the groundtruth pose, as well as the *optimal* pose returned by Horn's method [27] (label: SVD). Table 1 shows the rotation (e_R) and translation (e_t) estimation errors of DAMP and SVD w.r.t. groundtruth, as well as the difference between DAMP and SVD estimates ($\bar{e}_R$ and $\bar{e}_t$), under 1000 Monte Carlo runs. The statistics show that (i) DAMP always converges to the globally optimal solution ($\bar{e}_R, \bar{e}_t$ are numerically zero), empirically proving the correctness of Theorem 11; (ii) DAMP returns accurate pose estimations. On average, DAMP converges to the optimal equilibrium point in 27 iterations ($\|\dot{\boldsymbol{s}}\| < 10^{-6}$), and runs in 6.3 milliseconds. Although DAMP is slower than SVD in 3D, it opens up a new method to perform high-dimensional point cloud registration by using *geometric algebra* [18] to simulate rigid body dynamics [8], when SVD becomes expensive. We also use the Bunny dataset for point cloud registration and DAMP always returns the correct solution, shown in Fig. 1(a).

	DAMP	SVD [27]
e_R (°)	(0.065/0.011/0.188)	(0.065/0.011/0.188)
e_t (m)	(1.6e−3/1.4e−4/4.3e−3)	(1.6e−3/1.4e−4/4.3e−3)
$\bar{e}_R$ (°)	(2.9e−5/0/5.1e−5)	
$\bar{e}_t$ (m)	(2.3e−7/6.1e−9/6.9e−7)	

Table 1: Point cloud registration: DAMP converges to the globally optimal solution. Errors in (mean/ min / max).

Primitive Registration. In order to test DAMP's performance on primitive registration and verify its global convergence, we follow the test setup in [9] using random registration problems with point-to-point, point-to-line and point-to-plane correspondences, and compare DAMP with the state-of-the-art *certifiably optimal* solver in [9] based on semidefinite relaxation (label: SDR). In particular, we randomly sample 50 points, 50 lines and 50 planes (150 primitives in total) within a scene with radius 10, randomly sample a point on each primitive, and transform the sampled points by a random $(\boldsymbol{R}, \boldsymbol{t})$, followed by adding Gaussian noise $\mathcal{N}(\mathbf{0}, \sigma^2\mathbf{I}_3)$. We increase the noise level σ from 0.01 to 2, and perform 1000 Monte Carlo runs at each noise level. Fig. 3 boxplots the rotation estimation error and runtime of DAMP and SDR (SDP solved by SeDuMi [48] with CVX interface [24]). We observe that (i) DAMP always returns the same solution as SDR, which is certified to be the globally optimal solution (Supplementary Material plots the relative duality gap of SDR is always zero); (ii) DAMP is about 10 times faster than SDR, despite being implemented in Matlab using for loops. The translation error looks similar as rotation error and is shown in Supplementary Material. This experiment shows that the same global convergence Theorem 11 is very likely to hold in the case of general primitive registration with line and plane correspondences. In fact, Supplementary Material also performs the same set of experiments using the robot primitive model in Fig. 1(b) with spheres, cylinders and cones, demonstrating that DAMP also *always* converges to an accurate (most likely optimal) pose estimate (note that we cannot claim global optimality because there is no guaranteed globally optimal solver, such as SDR [9], in that case to verify DAMP).

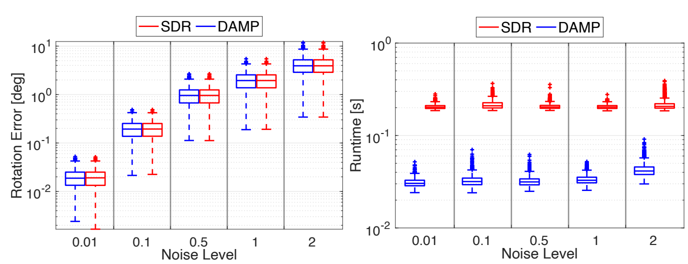


Figure 3: Rotation error and runtime of DAMP compared with SDR [9] on random primitive registration with increasing noise levels. DAMP always converges to the globally optimal solution while being 10 times faster.

Category Registration. We use three categories, *aeroplane*, *car*, and *chair*, from the PASCAL3D+ dataset [51] to test DAMP for category registration. In particular, given a list of K instances in a category, where each instance has N semantic keypoints $\mathcal{B}_k \in \mathbb{R}^{3\times N}, k = 1, \dots, K$. We first build a category model of the K instances into N SUEs (see Supplementary Material) and use it as $\mathcal{Y}$ in problem (1). Then we randomly generate an unknown instance of this category by following the active shape model [60, 55], *i.e.* $\mathcal{S} = \sum_{k=1}^{K} c_k \mathcal{B}_k$ with $c_k \geq 0, \sum_{k=1}^{K} c_k = 1$. After this, we apply a random transformation $(\boldsymbol{R}, \boldsymbol{t})$ to $\mathcal{S}$ to obtain $\mathcal{X}$ in problem (1). We have $N = 8, K = 8$ for aeroplane, $N = 12, K = 9$ for car, and $N = 10, K = 8$ for chair. For each category, we perform 1000 Monte Carlo runs and Fig. 4 summarizes the rotation and translation estimation errors. We can see that DAMP returns accurate rotation and translation estimates for all 1000 Monte Carlo runs of each category. Because a globally optimal solver is not available for the case of registering a point cloud to a set of ellipsoids, we cannot claim the global convergence of DAMP, although the results highly suggest the global convergence. An example of registering the chair category is shown in Fig. 1(c).

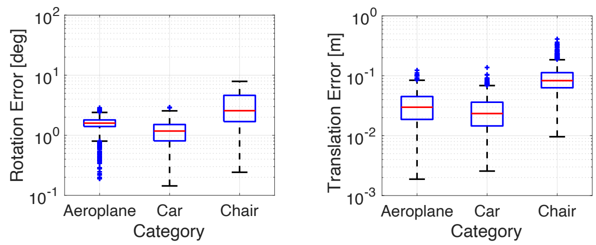


Figure 4: Rotation and translation estimation error of DAMP on category registration using the aeroplane, car, and chair categories from PASCAL3D+ dataset [51].

4.2. 2D-3D: Escape Local Minima

Absolute Pose Estimation. We follow the protocol in [30] for absolute pose estimation. We first generate N groundtruth 3D points within the $[-2, 2] \times [-2, 2] \times [4, 8]$ box inside the camera frame, then project the 3D points onto the image plane and add random Gaussian noise $\mathcal{N}(\mathbf{0}, 0.01^2\mathbf{I}_2)$ to the 2D projections. N bearing vectors are then formed from the 2D projections to be the set $\mathcal{Y}$ in problem (1). We apply a random $(\boldsymbol{R}, \boldsymbol{t})$ to the groundtruth 3D points to convert them into the world frame as the set $\mathcal{X}$ in problem (1). We apply DAMP to solve 1000 Monte Carlo runs of this problem for $N = 50, 100, 200$, with both EscapeMinimum = False and EscapeMinimum = True ($T_{\max} = 5$). Table 2 shows the success rate of DAMP, where we say a pose estimation is successful if rotation error is below $5°$ and translation error is below 0.5. One can see that, (i) even without the EscapeMinimum scheme, DAMP already has a very high success rate and it only failed twice when $N = 100$; (ii) with the EscapeMinimum scheme, DAMP achieves a 100% success rate. This experiment indicates that the special configuration of the bearing vectors (*i.e.*, they form a "cone" pointed at the camera center) is more challenging for DAMP to converge. We also apply DAMP to satellite pose estimation from 2D landmarks detected by a neural network [12] using the SPEED dataset [45] and a successful example is provided in Fig. 1(d).

N	50		100		200	
Escape	False	True	False	True	False	True
Success (%)	100	100	99.8	100	100	100

Table 2: Success rate of DAMP on absolute pose estimation with EscapeMinimum flag set to True and False.

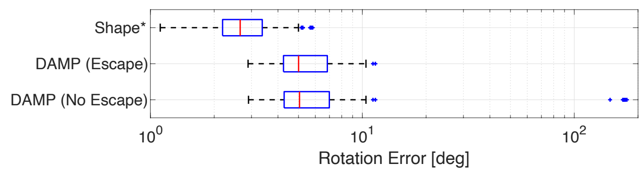


Figure 5: Rotation estimation error of DAMP (both with and without EscapeMinimum) and Shape⋆ on FG3DCar [34].

Category APE. We test DAMP on FG3DCar [34] for category APE, which contains 300 images of cars each with $N = 256$ 2D landmark detections. DAMP performs pose estimation by aligning the category model of SUEs (*c.f.* Fig. 1(e)) to the set of bearing vectors. Fig. 5 compares the rotation estimation error of DAMP with Shape⋆ [55], a state-of-the-art certifiably optimal solver for joint shape and pose estimation from 2D landmarks. We can see that DAMP without EscapeMinimum fails on 6 out of the 300 images, but DAMP with EscapeMinimum succeeds on all 300 images, and return rotation estimates that are similar to Shape⋆ (note that the difference is due to Shape⋆ using a weak perspective camera model). We do notice that this is a challenging case for DAMP because it takes more than 1000 iterations to converge, and the average runtime is 20 seconds. However, DAMP is still faster than Shape⋆ (about 1 minute runtime), and we believe there is significant room for speedup by using parallelization [28, 2].

5. Conclusion

We proposed DAMP, the first general meta-algorithm for solving five pose estimation problems by simulating rigid body dynamics. We demonstrated surprising global convergence of DAMP: it always converges given 3D-3D correspondences, and effectively escapes suboptimal solutions given 2D-3D correspondences. We proved a global convergence result in the case of point cloud registration.

Future work can be done to (i) extend the global convergence to general primitive registration; (ii) explore GPU parallelization [2] to enable a fast implementation; (iii) generalize DAMP to high-dimensional registration for applications such as unsupervised language translation [14, 4]. Geometric algebra (GA) [18] can describe rigid body dynamics in any dimension, but computational challenges remain in high-dimensional GA and deserve further investigation.

References

[1] Sérgio Agostinho, João Gomes, and Alessio Del Bue. Cvx-PnPL: A unified convex solution to the absolute pose estimation problem from point and line correspondences. *arXiv preprint arXiv:1907.10545*, 2019. 3

[2] Sk Aziz Ali, Kerem Kahraman, Christian Theobalt, Didier Stricker, and Vladislav Golyanik. Fast gravitational approach for rigid point set registration with ordinary differential equations. *arXiv preprint arXiv:2009.14005*, 2020. 8

[3] Pasquale Antonante, Vasileios Tzoumas, Heng Yang, and Luca Carlone. Outlier-robust estimation: Hardness, minimally-tuned algorithms, and applications. *arXiv preprint arXiv: 2007.15109*, 2020. 2

[4] Mikel Artetxe, Gorka Labaka, and Eneko Agirre. A robust self-learning method for fully unsupervised cross-lingual mappings of word embeddings. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018. 8

[5] K.S. Arun, T.S. Huang, and S.D. Blostein. Least-squares fitting of two 3-D point sets. *IEEE Trans. Pattern Anal. Machine Intell.*, 9(5):698–700, sept. 1987. 3

[6] David Baraff. An introduction to physically based modeling: rigid body simulation i—unconstrained rigid body dynamics. *SIGGRAPH course notes*, 82, 1997. 5

[7] P. J. Besl and N. D. McKay. A method for registration of 3-D shapes. *IEEE Trans. Pattern Anal. Machine Intell.*, 14(2), 1992. 2

[8] Marc Ten Bosch. N-dimensional rigid body dynamics. *ACM Transactions on Graphics (TOG)*, 39(4):55–1, 2020. 7

[9] Jesus Briales and Javier Gonzalez-Jimenez. Convex Global 3D Registration with Lagrangian Duality. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2, 3, 7

[10] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J.J. Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Trans. Robotics*, 32(6):1309–1332, 2016. 2

[11] Florian Chabot, Mohamed Chaouch, Jaonary Rabarisoa, Céline Teuliere, and Thierry Chateau. Deep MANTA: A coarse-to-fine many-task network for joint 2D and 3D vehicle analysis from monocular image. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2040–2049, 2017. 2, 3

[12] Bo Chen, Jiewei Cao, Alvaro Parra, and Tat-Jun Chin. Satellite pose estimation with deep landmark regression and non-linear pose refinement. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2019. 8

[13] Christopher Choy, Jaesik Park, and Vladlen Koltun. Fully convolutional geometric features. In *Intl. Conf. on Computer Vision (ICCV)*, pages 8958–8966, 2019. 2

[14] Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. Word translation without parallel data. In *International Conference on Learning Representations*, 2018. 8

[15] Timothy F. Cootes, Christopher J. Taylor, David H. Cooper, and Jim Graham. Active shape models - their training and application. *Comput. Vis. Image Underst.*, 61(1):38–59, January 1995. 3

[16] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *SIGGRAPH*, pages 303–312, 1996. 1, 2

[17] Achiya Dax. The distance between two convex sets. *Linear Algebra and its Applications*, 416(1):184–213, 2006. 5

[18] Chris Doran, Steven R Gullans, Anthony Lasenby, Joan Lasenby, and William Fitzgerald. *Geometric algebra for physicists*. Cambridge University Press, 2003. 7, 8

[19] M. Fischler and R. Bolles. Random sample consensus: a paradigm for model fitting with application to image analysis and automated cartography. *Commun. ACM*, 24:381–395, 1981. 2

[20] Kyle Genova, Forrester Cole, Daniel Vlasic, Aaron Sarna, William T Freeman, and Thomas Funkhouser. Learning shape templates with structured implicit functions. In *Intl. Conf. on Computer Vision (ICCV)*, pages 7154–7164, 2019. 2

[21] Zan Gojcic, Caifa Zhou, Jan D Wegner, and Andreas Wieser. The perfect match: 3d point cloud matching with smoothed densities. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5545–5554, 2019. 2

[22] Vladislav Golyanik, Sk Aziz Ali, and Didier Stricker. Gravitational approach for point set registration. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 5802–5810, 2016. 3, 5, 6

[23] Vladislav Golyanik, Christian Theobalt, and Didier Stricker. Accelerated gravitational point set alignment with altered physical laws. In *Intl. Conf. on Computer Vision (ICCV)*, pages 2080–2089, 2019. 3, 5, 6

[24] M. Grant and S. Boyd. CVX: Matlab software for disciplined convex programming. 7

[25] Lie Gu and Takeo Kanade. 3D alignment of face in a single image. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 1305–1312, 2006. 3

[26] R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2004. 2

[27] Berthold K. P. Horn. Closed-form solution of absolute orientation using unit quaternions. *J. Opt. Soc. Amer.*, 4(4):629–642, Apr 1987. 2, 3, 7

[28] Philipp Jauer, Ivo Kuhlemann, Ralf Bruder, Achim Schweikard, and Floris Ernst. Efficient registration of high-resolution feature enhanced point clouds. *IEEE Trans. Pattern Anal. Machine Intell.*, 41(5):1102–1115, 2018. 3, 5, 6, 8

[29] Lei Ke, Shichao Li, Yanan Sun, Yu-Wing Tai, and Chi-Keung Tang. Gsnet: Joint vehicle pose and shape reconstruction with geometrical and scene-aware supervision. In *European Conf. on Computer Vision (ECCV)*, pages 515–532. Springer, 2020. 3

[30] Laurent Kneip, Hongdong Li, and Yongduek Seo. UPnP: An optimal o(n) solution to the absolute pose problem with uni-

versal applicability. In *European Conf. on Computer Vision (ECCV)*, pages 127–142. Springer, 2014. 2, 3, 8

[31] Nikos Kolotouros, Georgios Pavlakos, Michael J Black, and Kostas Daniilidis. Learning to reconstruct 3d human pose and shape via model-fitting in the loop. In *Intl. Conf. on Computer Vision (ICCV)*, pages 2252–2261, 2019. 3

[32] Abhijit Kundu, Yin Li, and James M Rehg. 3d-rcnn: Instance-level 3d object reconstruction via render-and-compare. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 3559–3568, 2018. 3

[33] Lingxiao Li, Minhyuk Sung, Anastasia Dubrovina, Li Yi, and Leonidas J Guibas. Supervised fitting of geometric primitives to 3d point clouds. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2652–2660, 2019. 2

[34] Yen-Liang Lin, Vlad I. Morariu, Winston H. Hsu, and Larry S. Davis. Jointly optimizing 3D model fitting and fine-grained classification. In *European Conf. on Computer Vision (ECCV)*, 2014. 1, 3, 8

[35] Yinlong Liu, Guang Chen, and Alois Knoll. Globally optimal camera orientation estimation from line correspondences by bnb algorithm. *IEEE Robotics and Automation Letters*, 6(1):215–222, 2020. 3

[36] Lucas Manuelli, Wei Gao, Peter Florence, and Russ Tedrake. kPAM: Keypoint affordances for category-level robotic manipulation. In *Proc. of the Intl. Symp. of Robotics Research (ISRR)*, 2019. 2

[37] Yurii Nesterov et al. *Lectures on convex optimization*, volume 137. Springer, 2018. 4

[38] Carl Olsson, Fredrik Kahl, and Magnus Oskarsson. Branch-and-bound methods for euclidean registration problems. *IEEE Trans. Pattern Anal. Machine Intell.*, 31(5):783–794, 2009. 3

[39] Sida Peng, Yuan Liu, Qixing Huang, Xiaowei Zhou, and Hujun Bao. PVNet: Pixel-wise Voting Network for 6DoF Pose Estimation. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 4561–4570, 2019. 2

[40] Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. Reconstructing 3D human pose from 2D image landmarks. In *European Conf. on Computer Vision (ECCV)*, 2012. 3

[41] R Tyrrell Rockafellar and Roger J-B Wets. *Variational analysis*, volume 317. Springer Science & Business Media, 2009. 5

[42] A. Rosinol, M. Abate, Y. Chang, and L. Carlone. Kimera: an open-source library for real-time metric-semantic localization and mapping. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2020. 2

[43] R.B. Rusu, N. Blodow, and M. Beetz. Fast point feature histograms (fpfh) for 3d registration. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 3212–3217. Citeseer, 2009. 2

[44] Gerald Schweighofer and Axel Pinz. Globally optimal O(n) solution to the PnP problem for general camera models. In *British Machine Vision Conf. (BMVC)*, pages 1–10, 2008. 3

[45] Sumant Sharma and Simone D'Amico. Pose estimation for non-cooperative rendezvous using neural networks. *arXiv preprint arXiv:1906.09868*, 2019. 1, 8

[46] Jingnan Shi, Heng Yang, and Luca Carlone. Optimal pose and shape estimation for category-level 3d object perception. In *Robotics: Science and Systems (RSS)*, 2021. 2

[47] Jean-Jacques E Slotine, Weiping Li, et al. *Applied nonlinear control*, volume 199. Prentice hall Englewood Cliffs, NJ, 1991. 6

[48] Jos F Sturm. Using sedumi 1.02, a matlab toolbox for optimization over symmetric cones. *Optimization methods and software*, 11(1-4):625–653, 1999. 7

[49] Maxim Tatarchenko, Stephan R Richter, René Ranftl, Zhuwen Li, Vladlen Koltun, and Thomas Brox. What Do Single-view 3D Reconstruction Networks Learn? In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 3405–3414, 2019. 3

[50] Shubham Tulsiani, Hao Su, Leonidas J Guibas, Alexei A Efros, and Jitendra Malik. Learning shape abstractions by assembling volumetric primitives. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2635–2643, 2017. 2

[51] Yu Xiang, Roozbeh Mottaghi, and Silvio Savarese. Beyond PASCAL: A benchmark for 3d object detection in the wild. In *IEEE Winter Conference on Applications of Computer Vision*, pages 75–82. IEEE, 2014. 1, 7, 8

[52] H. Yang, P. Antonante, V. Tzoumas, and L. Carlone. Graduated non-convexity for robust spatial perception: From non-minimal solvers to global outlier rejection. *IEEE Robotics and Automation Letters*, 5(2):1127–1134, 2020. 2

[53] H. Yang and L. Carlone. A polynomial-time solution for robust registration with extreme outlier rates. In *Robotics: Science and Systems (RSS)*, 2019. 2

[54] H. Yang and L. Carlone. A quaternion-based certifiably optimal solution to the Wahba problem with outliers. In *Intl. Conf. on Computer Vision (ICCV)*, 2019. 4, 5

[55] Heng Yang and Luca Carlone. In Perfect Shape: Certifiably Optimal 3D Shape Reconstruction from 2D Landmarks. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020. 3, 8

[56] H. Yang and L. Carlone. One ring to rule them all: Certifiably robust geometric perception with outliers. In *Advances in Neural Information Processing Systems (NIPS)*, 2020. 2, 3

[57] Heng Yang, Wei Dong, Luca Carlone, and Vladlen Koltun. Self-supervised Geometric Perception. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021. 2

[58] H. Yang, J. Shi, and L. Carlone. TEASER: Fast and Certifiable Point Cloud Registration. *IEEE Trans. Robotics*, 2020. 2, 3

[59] Xiaowei Zhou, Spyridon Leonardos, Xiaoyan Hu, and Kostas Daniilidis. 3D shape reconstruction from 2D landmarks: A convex formulation. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2015. 3

[60] Xiaowei Zhou, Menglong Zhu, Spyridon Leonardos, and Kostas Daniilidis. Sparse representation for 3D shape estimation: A convex relaxation approach. *IEEE Trans. Pattern Anal. Machine Intell.*, 39(8):1648–1661, 2017. 3, 8