A Distance-Geometric Method for Recovering Robot Joint Angles From an RGB Image ⋆

Ivan Bilić * **Filip Marić** *, ** **Ivan Marković** * **Ivan Petrović** *

* *University of Zagreb, Faculty of Electrical Engineering and Computing, Laboratory for Autonomous Systems and Mobile Robotics (e-mail: name.surname@fer.hr).*
** *University of Toronto, Space and Terrestrial Autonomous Robotic Systems Laboratory (e-mail: name.surname@mail.utoronto.ca)*

Abstract: Autonomous manipulation systems operating in domains where human intervention is difficult or impossible (e.g., underwater, extraterrestrial or hazardous environments) require a high degree of robustness to sensing and communication failures. Crucially, motion planning and control algorithms require a stream of accurate joint angle data provided by joint encoders, the failure of which may result in an unrecoverable loss of functionality. In this paper, we present a novel method for retrieving the joint angles of a robot manipulator using only a single RGB image of its current configuration, opening up an avenue for recovering system functionality when conventional proprioceptive sensing is unavailable. Our approach, based on a distance-geometric representation of the configuration space, exploits the knowledge of a robot's kinematic model with the goal of training a shallow neural network that performs a 2D-to-3D regression of distances associated with detected structural keypoints. It is shown that the resulting Euclidean distance matrix uniquely corresponds to the observed configuration, where joint angles can be recovered via multidimensional scaling and a simple inverse kinematics procedure. We evaluate the performance of our approach on real RGB images of a Franka Emika Panda manipulator, showing that the proposed method is efficient and exhibits solid generalization ability. Furthermore, we show that our method can be easily combined with a dense refinement technique to obtain superior results.

Keywords: Manipulation; Mechatronic Systems; Robotics; Joint angle estimation

1. INTRODUCTION

Autonomous manipulation systems are ideal for performing various tasks in environments where human presence is limited, such as underwater or orbital laboratories, as well as hazardous (e.g., radioactive, toxic) environments. In addition to effective planning and control algorithms, these systems require a high degree of robustness to sensing and communication failures, as a timely intervention by humans may be impossible. We propose a method for recovering the joint angles (i.e., the configuration) of an articulated robotic manipulator using only a single RGB image, providing an alternative source of proprioceptive data that can be used when data from joint encoders is unavailable. This is a challenging task for multiple reasons: fundamentally different robot configurations may result in similar images and certain configurations may have diminished observability due to physical occlusion by other parts of the robot. However, even a rough estimate of the joint configuration may enable the use of simple control methods to steer the robot to an approximate desired state, enabling the planning and execution of critical recovery protocols (Ortenzi et al. (2018)).

⋆ This work has been supported by the European Regional Development Fund under the grant KK.01.1.1.01.0009 (DATACROSS).

The problem of recovering a robot configuration from spatial constraints such as gripper pose is known as inverse kinematics and features a variety of well known solutions (Lynch and Park (2017)). However, previous work has also explored instances of this problem where spatial constraints may result from a variety of sensing modalities, such as depth or RGB images. Widmaier et al. (2016) use synthetic depth images to train semantic classifiers for direct joint angle regression in order to estimate the robot arm pose. Bohg et al. (2014) also use depth images to train a random forest classifier for pixel-wise part classification, while using joint encoder readings to initialize an incremental update scheme. Conversely, our method aims to recover joint configuration using only a single RGB image, which necessitates first finding the appropriate set of spatial constraints using 2D-to-3D regression.

The constraints resulting from regressed 3D keypoints may be under-determined and therefore correspond to multiple configurations. Instead, we use a distance-geometric model that integrates structural data (e.g., link lengths) to remove ambiguity. Distance geometry is highly relevant for applications such as molecular conformation, sensor network localization (SNL) and statics (Liberti et al. (2014)). For instance, SNL is commonly framed as an Euclidean distance matrix (EDM) completion problem (Dokmanic

et al. (2015)) and tackled through semidefinite programming (SDP) (Biswas et al. (2006)). Marić et al. (2022) consider a large class of articulated robot manipulators and elaborate on the equivalence of the distance geometry problem and distance-based inverse kinematics. Furthermore, Moreno-Noguer (2017) tackles the problem of 3D human pose estimation from a single RGB image and demonstrates that representing human poses with EDMs instead of Cartesian coordinates results in more precise and less ambiguous pose estimates. Our work is partly inspired by these observations, and we show that the problem of recovering robot's joint angles is in general highly related to distance geometry.

To the best of our knowledge, the only similar approaches to ours are that of Zuo et al. (2019) and Labbé et al. (2021), in the sense that only a single RGB image is used as an input for joint angle estimation. Zuo et al. (2019) train a joint keypoint detector to recognize a predefined, specifically chosen set of 17 keypoints in the image displaying a 4-DoF toy robot. These keypoints are then fed to a nonlinear non-convex 2D-to-3D optimization algorithm in order to recover the 6D pose together with joint angles. Although our distance-geometric method is also keypoint-based, it only requires a number of keypoints equal to the robot's DoF, placed in a robot-invariant manner. Furthermore, the optimization proposed in Zuo et al. (2019) is complex, which diminishes its potential to improve from scaling the model and data size. On the other hand, Labbé et al. (2021) use a dense, rendering-based deep iterative matching approach (Li et al. (2018)) to jointly learn the 6D camera-to-robot pose and joint angle updates. Although primarily concerned with estimating the 6D pose, Labbé et al. (2021) also demonstrate that joint angles, if unknown, can be accurately reconstructed, at least with a sufficient number of iterations. Overall, the dense approach seems to generally outperform sparse approaches in terms of accuracy due to incorporating global information, at the cost of high computation time. We demonstrate that using our method in conjunction with a dense refinement offers the best of both worlds.

In this paper, we propose a novel method for recovering the joint angles of an articulated robotic manipulator using only a single RGB image, based on a distance-geometric representation of the configuration space and the knowledge of a robot's kinematic model [1]. Instead of training a single large model to directly predict the solution, our method divides the problem into a set of smaller subproblems in a theoretically justified manner. First, state-of-the-art keypoint detectors (Lee et al. (2020)) are used to detect joint keypoints in the image corresponding to the robot's joints, which are insufficient for configuration recovery on their own. Then, our method takes the full set of inter-point distances and uses a learned 2D-to-3D regression to produce an EDM corresponding to the associated 3D keypoints. Following the approach in (Marić et al. (2022)) this EDM is extended to include distances between auxiliary points determined by the robot's structure, removing ambiguity related to joint angle recovery. Given a complete EDM, joint angles can be computed using parameter-free transformations; classical multidimensional scaling (MDS) and kinematic transformations. The former maps an EDM to a geometrically centered set of 3D points, while the latter calculates the joint angles based on these points, forming a fully-differentiable set of kinematic transformations [2] that supports batching and can be ran on a GPU. In addition to generating the complete EDM, ground-truth joint angles are used to compute the loss in the configuration space. Finally, our method is evaluated on a large set of real images displaying a 7-DoF robot arm in various configurations. We opt for a shallow architecture for all our experiments since the primary goal of this work is to develop a geometrically meaningful learning-based framework. The proposed method exhibits solid generalization ability, while being simple and computationally efficient.

[1] https://github.com/iwhitey/distance-geometric-robot-joint-angles

2. METHODOLOGICAL BACKGROUND

Euclidean distance geometry is an important tool in several applications whose aim is to reconstruct a complete set of distances (or points that realize them) in Euclidean space, given an incomplete set of distances as an input. In addition to a small subset of distance geometry which relates to EDMs, this section describes a kinematics procedure responsible for generating a set of 3D points sufficient for recovering the robot's configuration as well as distance constraints emerging from kinematics.

2.1 Euclidean distance matrices

Let $\mathbf{P} \in \mathbb{R}^{n\times d}$ denote a matrix representing a set of n points in a d-dimensional Euclidean space. Then, the pairwise distances $d_{u,v}$ between points can be calculated using the Euclidean norm:

$$d_{u,v} = \|\mathbf{p}_u - \mathbf{p}_v\|. \tag{1}$$

For the sake of notation simplicity, EDMs and individual distances are assumed to be squared in the remaining of the paper. Expansion of (1) reveals that EDM is a function of the Gram matrix $\mathbf{G} = \mathbf{P}\mathbf{P}^\top$:

$$\operatorname{edm}(\mathbf{G}) = \operatorname{diag}(\mathbf{G})\mathbf{1}^\top + \mathbf{1}\operatorname{diag}(\mathbf{G})^\top - 2\mathbf{G}, \tag{2}$$

with $\operatorname{diag}(\mathbf{G})$ representing the diagonal entries of $\mathbf{G}$ in the form of a column vector and $\mathbf{1}$ stands for column vector filled with ones.

Equation (2) establishes a one-way connection between an EDM and a Gram matrix. Consider an inverse problem, i.e. recovering the set of points that generated the distance matrix $\mathbf{D}$. Let $\mathbf{D}$ be a squared EDM. Then, a Gram matrix that satisfies (2) can be determined via

$$\mathbf{G} = -\frac{1}{2}\mathbf{J}\mathbf{D}\mathbf{J}, \tag{3}$$

where

$$\mathbf{J} = \mathbf{I} - \frac{1}{N}\mathbf{1}\mathbf{1}^\mathbf{T} \tag{4}$$

denotes a geometric centering matrix. Moreover, $\mathbf{G}$ is a real symmetric matrix, hence it can be factored into a canonical form via eigenvalue decomposition:

[2] We refer to this set of transformations as an IK layer in the rest of the paper.

$$\mathbf{G} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^{\mathbf{T}} \tag{5}$$

where $\mathbf{\Lambda} = \operatorname{diag}(\lambda_0, \lambda_1, \ldots, \lambda_{n-1})$ contains the non-negative eigenvalues λ_i and U is an orthonormal matrix. Now, assuming the eigenvalues are sorted in the descending order, the point set $\hat{\mathbf{P}} \in \mathbb{R}^{n \times d}$ can finally be recovered by taking:

$$\hat{\mathbf{P}}^\top = \left[\operatorname{diag}\left(\sqrt{\lambda_0}, \sqrt{\lambda_1}, \ldots, \sqrt{\lambda_{d-1}}\right), \mathbf{0}_{d \times N-d}\right] \mathbf{U}^\top. \tag{6}$$

Computing the collection of points $\hat{\mathbf{P}}$ from a distance matrix $\mathbf{D}$ using (3), (4), (5) and (6) is also known as classical multidimensional scaling (cMDS). Note how in (6), all but the d largest eigenvalues are discarded. Assuming that $\mathbf{G}$ is generated by a d-dimensional set of points, all but the d largest eigenvalues will be zeros. If this is not the case, we can assume the presence of noise which is handled by the truncation (Dokmanic et al. (2015)). Additionally, plugging the estimated $\hat{\mathbf{P}}$ in (2) yields an EDM that equals $\mathbf{D}$. However, distances are preserved under rigid transformations, thus $\hat{\mathbf{P}}$ and the original $\mathbf{P}$ are not generally equal. Absolute position and orientation of the point set can be recovered through Procrustes analysis (Schönemann (1966)), assuming that a set of at least d fixed points (i.e. anchors) is known beforehand. Then, a rigid transformation that aligns the anchors in $\mathbf{P}$ to their corresponding points in $\hat{\mathbf{P}}$ can be found. Finally, the original set of points $\mathbf{P}$ can be recovered by applying the obtained rigid transformation to all the points in $\hat{\mathbf{P}}$.

2.2 Distance-based kinematics

Consider an n-DoF robotic manipulator comprised of single-axis revolute joints, forming a kinematic chain. The procedure introduced in Marić et al. (2022) defines a sparse set of points whose positions are sufficient for recovering the full set of joint angles determining the robot's configuration. As shown in Figure 1, a set of points $\mathbf{p}_i$ centered at the joint coordinate frames are introduced, which we associate with the keypoints detected in the RGB image of the robot by our network. Then, "virtual" auxiliary points $\mathbf{q}_i$ are placed at a unit distance along the joints' rotation axes $\tilde{\mathbf{z}}$ using joints' orientation $\mathbf{R}_i$

$$\mathbf{q}_i = \mathbf{p}_i + \mathbf{R}_i \tilde{\mathbf{z}}, \tag{7}$$

adding information on the relative orientation of neighbouring joints required for joint angle recovery. Finally, the model is completed with the addition of three points, $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{z}$, corresponding to the root coordinate frame, defined by distances

$$\begin{aligned} \|\mathbf{p}_0 - \mathbf{x}\| = \|\mathbf{p}_0 - \mathbf{y}\| = \|\mathbf{p}_0 - \mathbf{z}\| = 1, \\ \|\mathbf{x} - \mathbf{y}\| = \|\mathbf{x} - \mathbf{z}\| = \|\mathbf{y} - \mathbf{z}\| = \sqrt{2}. \end{aligned} \tag{8}$$

The proof in Marić et al. (2022) shows that the distances between these points are sufficient for recovering the full set of joint angles for a large variety of manipulator structures.

Our training data is generated from ground truth joint angle vectors $\mathbf{\Theta} \in \mathcal{C}$ corresponding to the robot observed in the image $\mathcal{I}$, where $\mathcal{C} \subseteq \mathbb{R}^n$ represents the configuration space. This allows us to simply generate the full distance-geometric robot description. Each data point is constructed by sequentially taking the position $\mathbf{p}_i$ and orientation $\mathbf{R}_i$ of a parent joint i, as well as its joint angle θ_i, giving the position and orientation of the child joint j as

$$\begin{aligned} \mathbf{R}_j &= \mathbf{R}_i \mathbf{R}_z(\theta_i)\, \mathbf{R}_{i,j}, \\ \mathbf{p}_j &= \mathbf{p}_i + \mathbf{R}_i \mathbf{R}_z(\theta_i)\, \mathbf{p}_{i,j}. \end{aligned} \tag{9}$$

The relative positions and orientations $\mathbf{p}_{i,j}$ and $\mathbf{R}_{i,j}$ of neighbouring joints, are completely defined by the robot's structure. Thus, the distances between the set of the four neighbouring points can also be defined from structural parameters as

$$\begin{aligned} \|\mathbf{p}_i - \mathbf{q}_i\| &= \|\mathbf{p}_j - \mathbf{q}_j\| = 1 \\ \|\mathbf{p}_i - \mathbf{p}_j\| &= \|\mathbf{p}_{i,j}\| \\ \|\mathbf{p}_i - \mathbf{q}_j\| &= \|\mathbf{p}_{i,j} + \mathbf{R}_{i,j}\tilde{\mathbf{z}}\| \\ \|\mathbf{q}_i - \mathbf{p}_j\| &= \|\mathbf{p}_{i,j} - \tilde{\mathbf{z}}\| \\ \|\mathbf{q}_i - \mathbf{q}_j\| &= \|\mathbf{p}_{i,j} - \tilde{\mathbf{z}} + \mathbf{R}_{i,j}\tilde{\mathbf{z}}\|, \end{aligned} \tag{10}$$

indicating their invariance to changes in joint angles. Conversely, the distances corresponding to the remaining non-neighbouring pairs exist as a function of the configuration $\mathbf{\Theta}$. Using the constructive procedure in (7) and (9), together with distance constraints stated in (8) and (10), we construct a complete EDM $\mathbf{D} \in \mathbb{R}^{\hat{n} \times \hat{n}}$ that uniquely (up to a rigid transformation) represents a set of 3D points associated with the configuration $\mathbf{\Theta}$ observed in the RGB image. Concretely, for an n-DoF articulated robot comprised of single-axis revolute joints - one auxiliary point per joint to determine the axis of rotation, and two additional points are required to fully define the base coordinate frame. Finally, a single data sample used for training the model can thus be formalized as $\mathcal{D} := (\mathbf{\Theta}, \mathbf{D})$.

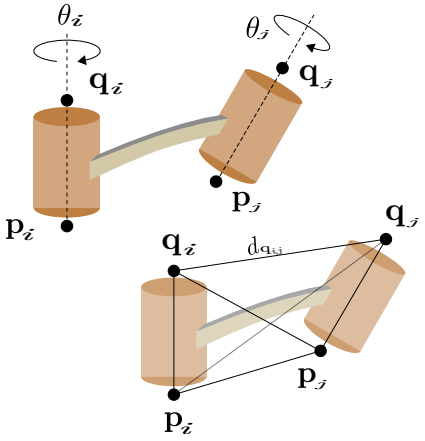


Fig. 1. Visualization of neighbouring revolute joints. Each joint is rotated by an angle θ around its rotation axis (dashed lines). Auxiliary points $\mathbf{q}$ are placed along these axis. Points $\mathbf{p}$ correspond to the position of coordinate frames defined in the center of their respective joints (placed below for better visibility).

3. METHOD

This section is dedicated to describing the system displayed in Figure 2 as a pipeline through which the proposed method is implemented. First, given the input RGB image and 2D joint keypoint detections, one needs to bridge the gap between 2D and 3D information. Theoretically, infinitely many sets of $\hat{n}$ points which are equal up to

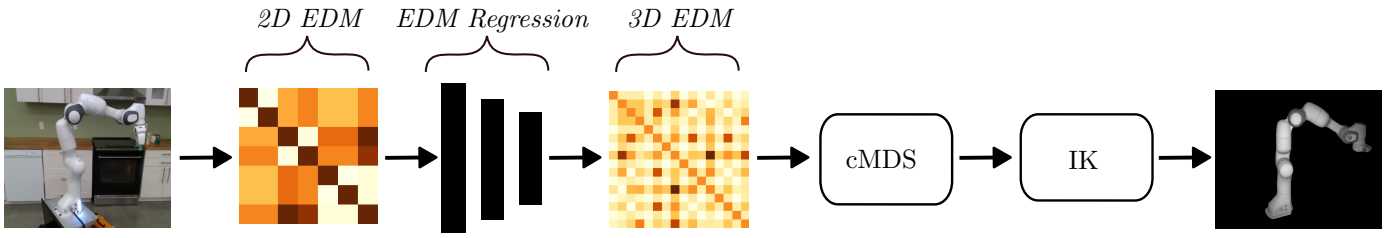


Fig. 2. System overview. In the image space, EDM is constructed from detected 2D joint keypoints and fed to a 2D-to-3D EDM regression network (black), which outputs the complete EDM in the 3D space. The cMDS layer maps the complete EDM to a set of geometrically centered points, from which an IK layer computes the joint angles. Estimated joint angles are shown as an image of the robot rendered in the respective configuration.

a rigid transformation, and generated as described in Section 2.2, can map to a single feasible robot configuration $\mathbf{\Theta}$. As inter-point distances remain identical irrespective of such point set transformation, we associate a configuration $\mathbf{\Theta}$ with a single unique EDM. Therefore, instead of directly predicting the points, we predict the corresponding distance matrix. We frame the distance matrix regression problem as learning a mapping $\zeta : \mathbb{R}_+^{n\times n} \mapsto \mathbb{R}_+^{\hat{n}\times\hat{n}}$, where $\mathbb{R}^+ = \{x \in \mathbb{R} : x \geq 0\}$, n equals the number of joints, and $\hat{n} = 2n + 2$. Note that we don't use auxiliary points in the image space, only keypoints directly corresponding to the robot's joints. The mapping is implemented as a shallow feedforward neural network that takes an EDM computed from 2D joint keypoints as input and outputs a complete EDM which is treated as if it was generated by a set of 3D points comprised of joint correspondences and auxiliary points. We define a distance-based loss as:

$$L_d = \|\hat{\mathbf{D}} - \mathbf{D}\|_F, \tag{11}$$

where $\|\cdot\|_F$ denotes Frobenius norm, while $\hat{\mathbf{D}}$ and $\mathbf{D}$ are the predicted and ground-truth EDMs, respectively. Due to the fact that EDMs are symmetric matrices, the actual implementation works with upper-triangular elements from which the full EDM is computed afterwards. This amounts to using $n(n-1)/2$ elements instead of n^2 with no loss of information. The ground-truth EDMs are computed from ground-truth configurations $\mathbf{\Theta}$ via a function $f : \mathbf{\Theta}^n \mapsto \mathbf{D}^{\hat{n}\times\hat{n}}$ based on equations described in Section 2.2. Figure 3 demonstrates that EDM regression and joint angle recovery are highly related, by depicting a mean absolute joint angle error as a function of mean absolute EDM error, using the Kinect dataset (unseen during the learning process). Note that the model used for this figure is trained for EDM regression, i.e. using the loss function (11), while the joint angles are only computed during inference via cMDS and IK layers. We use our library [3] for most of the distance-geometry and kinematics-related computation.

After the complete EDM $\mathbf{D} \in \mathbb{R}^{\hat{n}\times\hat{n}}$ is estimated, the cMDS layer (a set of fixed, differentiable transformations described in Section 2) is used to obtain the geometrically centered set of points which generate the respective EDM. This mapping can be formally defined as $\Omega : \mathbb{R}_+^{\hat{n}\times\hat{n}} \mapsto \mathbf{P}^{\hat{n}\times d}$. The set of points is then fed to an IK layer which computes the joint angles $\mathbf{\Theta}$. Note that what we refer to as an IK layer is not an inverse kinematics solver; it is a sequence of differentiable kinematics transformations that, given the estimated set of points, compute the joint angles by computing the respective coordinate frame positions and orientations together with the axis of rotation for each joint (Marić et al. (2022)). The default configuration (corresponding to zero joint angles) of the robot is known from its model (an Unified Robotics Description Format file), hence the joint angles can be computed. This allows us to define a loss in the configuration space between the predicted and ground-truth configuration:

$$L_c = |\hat{\mathbf{\Theta}} - \mathbf{\Theta}|, \tag{12}$$

with $|\cdot|$ denoting the L1 norm. We use a linear combination of the two losses as a final loss to train the model:

$$L = L_c + \lambda L_d, \tag{13}$$

with λ set to 0.5. Note that L_c causes the gradients to be propagated through IK and cMDS layers, while L_d is applied directly at the output of the EDM regression network and serves to provide the model with additional information.

The EDM regression network is comprised of two $Dense \rightarrow BatchNorm \rightarrow ReLU \rightarrow Dropout$ layers, and an output $Dense \rightarrow ReLU$ layer. The respective hidden and output layer sizes are $512 \rightarrow 512 \rightarrow 120$, which amounts to $\sim$ 350k parameters. The output size is determined by [4] $\hat{n}(\hat{n}-1)/2$, where $\hat{n} = 2n + 2$ and $n = 7$ for the Franka Emika Panda robot. Among other necessary conditons, EDM elements must be positive, which is enforced by the last ReLU (Nair

[3] https://github.com/utiasSTARS/graphIK

[4] The number of non-zero elements in strictly upper triangular EDM.

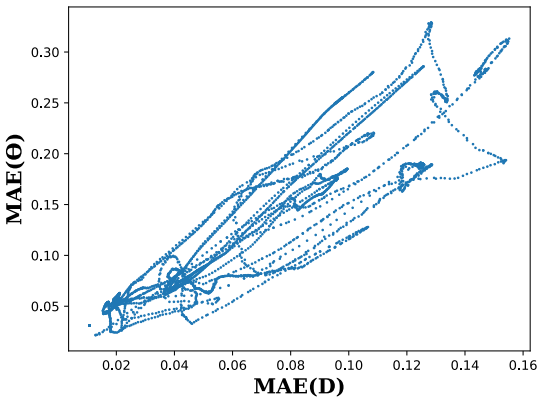


Fig. 3. Joint angles vs EDMs, using mean absolute error (MAE). The model is trained for EDM regression only. The errors are highly correlated; Pearson correlation coefficient is 0.94.

and Hinton (2010)) activation. We use Adam (Kingma and Ba (2014)) for optimization, with initial learning rate $\alpha = 1e-3$, linear warmup (Ma and Yarats (2021)) over 2k iterations, a batch size of 64 EDMs, and a dropout (Srivastava et al. (2014)) ratio of 0.5. The training is carried out for 100 epochs and learning rate is reduced by a factor of 2 after 50 epochs. The model is initialized as proposed by He et al. (2015).

4. EXPERIMENTS

All our experiments were conducted on images of a 7-DoF Franka Emika Panda robot observed in various configurations, using three different datasets. We report mean absolute error as a joint angle error metric. Training and evaluation were carried out on a single NVIDIA RTX A5000. Using the setup described in Section 3, inference takes 1.6ms and requires 1.5GB of GPU memory. In reality, it would be limited by the running time of the chosen keypoint detector.

4.1 Dataset

For our experiments, we use the DREAM dataset introduced by Lee et al. (2020) in their recently proposed state-of-the-art method for single-view camera-to-robot pose estimation. The dataset is comprised of real and synthetic parts. The synthetic part is photorealistic and generated with domain randomization. We focus on the real part of the dataset, which is split into 4 different Panda-3CAM datasets - Realsense, Azure, Kinect, and Orb which contain 5944, 6394, 4966 and 32315 samples, respectively. Each of these datasets is comprised of RGB images of the 7-DoF Franka Emika Panda robotic manipulator, captured using different cameras with different intrinsic parameters. The robot is observed in various configurations, including images with joint occlusions and even out-of-view joints. Besides RGB images, the datasets contain 2D joint keypoint annotations together with their 3D correspondences and ground-truth robot configuration. The camera-to-robot pose is different and fixed for each dataset, except for Orb which is captured from 27 different viewpoints. For all our experiments we used 8x subsampled version of the Orb dataset for training, which we refer to as Orb in the rest of the section. We automatically adjusted all the 2D joint keypoint annotations in all the datasets so that they match the Panda's coordinate frame definitions exactly.

4.2 Results

The results of applying our method on the Panda-3CAM datasets are displayed in Table 1. For evaluating on Kinect and Azure datasets the model is trained on Realsense and Orb, while for Realsense evaluation we trained it on the Kinect and Orb datasets. The results show that our method gives solid joint angle approximations on unseen data, while being simple and computationally efficient. The top 50% predictions mostly correspond to images which display configurations relatively close to coplanar with respect to the image plane, for a given robot-camera pose. However, the datasets also contain images with joint occlusions and highly non-coplanar configurations with respect to the image plane, making the task more difficult for a sparse, keypoint-based method such as ours. By manual inspection, we detected that Azure dataset contains more such images compared to the other two datasets, which reflects on the results. The more thorough error analysis is left for future work.

Table 1. Results on the Panda-3CAM datasets. The mean absolute joint angle error and standard deviation are reported. The error corresponds to predictions on all test images, or top 50% predictions closest to the ground-truth joint angles.

Dataset	num. images	all [10°]	top 50% [10°]
Realsense	5944	1.261 ± 0.218	0.562 ± 0.08
Kinect	4966	1.061 ± 0.2	0.344 ± 0.04
Azure	6394	1.433 ± 0.264	0.733 ± 0.092

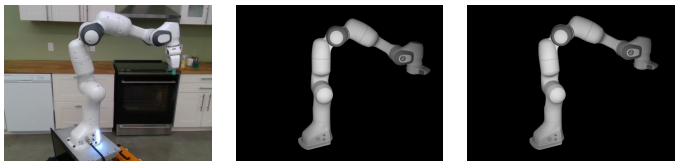


Fig. 4. Input image (left) and rendered joint angle predictions - our method (middle), our method with refinement (right)

4.3 Deep refinement

The proposed distance-geometric method exhibits solid generalization in addition to being lightweight, thus it can be easily used in conjunction with a refinement procedure without introducing noticeable computational complexity. To this end, we use the publicly available pretrained RoboPose model, introduced by Labbé et al. (2021) and trained using a deep iterative matching procedure (Li et al. (2018)) on the DREAM dataset. This procedure can be briefly described as follows. First, the joint angles are initialized randomly within joint angle limits and used to render an RGB image of the 3D robot model in this configuration. Then, the rendered and input RGB images are fed to a convolutional backbone. The backbone outputs the 6D pose update together with a joint angle residual which are used to update the input and the process is repeated iteratively.

We combine our method with RoboPose by using it to initialize the deep refinement procedure. The results are shown in Table 2. Clearly, the combined approach outperforms both our method (Table 1) and RoboPose applied independently. This is because our model provides a good initial guess, thereby making the refinement task much easier in contrast to using a feasible random configuration as an initial guess. An exemplary robot configuration estimated by our method and the combined approach is displayed in Figure 4 in the form of a rendered image. Furthermore, the goal is to obtain an accurate estimation using as few iterations as possible, hence the results were generated using 3 iterations, where each iteration requires a rendering operation and a forward pass of a deep

CNN backbone. On the contrary, RoboPose, when applied independently, requires at least 10 iterations to achieve similar accuracy. Note that we have not trained the model from scratch in order to adjust it to our initialization - it is pretrained using a feasible random configuration. Since our method provides an estimate much closer to the actual solution, we would expect retraining the model to introduce further improvements.

Table 2. Results on the Panda-3CAM datasets before and after combining the deep refinement model with our method. The mean absolute joint angle error is reported on all images.

Dataset	#images	RoboPose [10°]	Ours + RoboPose [10°]
Realsense	5944	1.221	0.585
Kinect	4966	1.368	0.613
Azure	6394	0.93	0.544

5. CONCLUSION

In this paper, we have proposed a novel distance-geometric framework for recovering the joint angles determining the configuration of the robot from a single RGB image. In addition to being geometrically meaningful, our method is computationally efficient and exhibits solid generalization ability when tested on a large set of images displaying a state-of-the-art 7-DoF robot. We also show that, due to its computational efficiency, it can be easily used in conjuction with a dense refinement approach to obtain superior results. We believe that a modular approach is promising in the long-term, i.e. tackling the larger problem through a set of smaller, simpler problems. If one can detect the joint keypoints and recover the respective EDM in the 3D space accurately, the joint angles can accurately be recovered since the leap from EDMs to joint angles is done via deterministic, parameter-free transformations. This is valuable since interpreting and analyzing deep models is often hard due to their nature; thus, if possible, it is appealing to design models that aim to solve smaller pieces of the problem. Finally, due to the sensitivity of sparse methods, achieving robustness requires incorporating global, dense information, but in a computationally efficient way. As future work, we intend to explore the adequacy of different architectures for this task, including scaling the data size and model capacity in order to investigate the potential of our approach more thoroughly.

REFERENCES

Biswas, P., Liang, T.C., Toh, K.C., Ye, Y., and Wang, T.C. (2006). Semidefinite programming approaches for sensor network localization with noisy distance measurements. *IEEE Transactions on Automation Science and Engineering*, 3(4), 360–371.

Bohg, J., Romero, J., Herzog, A., and Schaal, S. (2014). Robot arm pose estimation through pixel-wise part classification. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, 3143–3150.

Dokmanic, I., Parhizkar, R., Ranieri, J., and Vetterli, M. (2015). Euclidean distance matrices: Essential theory, algorithms, and applications. *IEEE Signal Processing Magazine*, 32(6), 12–30.

He, K., Zhang, X., Ren, S., and Sun, J. (2015). Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *2015 IEEE International Conference on Computer Vision (ICCV)*, 1026–1034.

Kingma, D.P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Labbé, Y., Carpentier, J., Aubry, M., and Sivic, J. (2021). Single-view robot pose and joint angle estimation via render & compare. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 1654–1663.

Lee, T.E., Tremblay, J., To, T., Cheng, J., Mosier, T., Kroemer, O., Fox, D., and Birchfield, S. (2020). Camera-to-robot pose estimation from a single image. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, 9426–9432.

Li, Y., Wang, G., Ji, X., Xiang, Y., and Fox, D. (2018). Deepim: Deep iterative matching for 6d pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 683–698.

Liberti, L., Lavor, C., Maculan, N., and Mucherino, A. (2014). Euclidean distance geometry and applications. *SIAM review*, 56(1), 3–69.

Lynch, K.M. and Park, F.C. (2017). *Modern Robotics*. Cambridge University Press.

Ma, J. and Yarats, D. (2021). On the adequacy of untuned warmup for adaptive optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, 8828–8836.

Marić, F., Giamou, M., Hall, A., Khoubyarian, S., Petrović, I., and J.Kelly (2022). Riemannian optimization for distance-geometric inverse kinematics. *IEEE Transactions on Robotics*, 38(3), 1703–1722.

Moreno-Noguer, F. (2017). 3d human pose estimation from a single image via distance matrix regression. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1561–1570.

Nair, V. and Hinton, G.E. (2010). Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, 807–814.

Ortenzi, V., Marturi, N., Mistry, M., Kuo, J., and Stolkin, R. (2018). Vision-based framework to estimate robot configuration and kinematic constraints. *IEEE/ASME Transactions on Mechatronics*, 23(5), 2402–2412.

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

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1), 1929–1958.

Widmaier, F., Kappler, D., Schaal, S., and Bohg, J. (2016). Robot arm pose estimation by pixel-wise regression of joint angles. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, 616–623.

Zuo, Y., Qiu, W., Xie, L., Zhong, F., Wang, Y., and Yuille, A. (2019). Craves: Controlling robotic arm with a vision-based economic system. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 4209–4218.