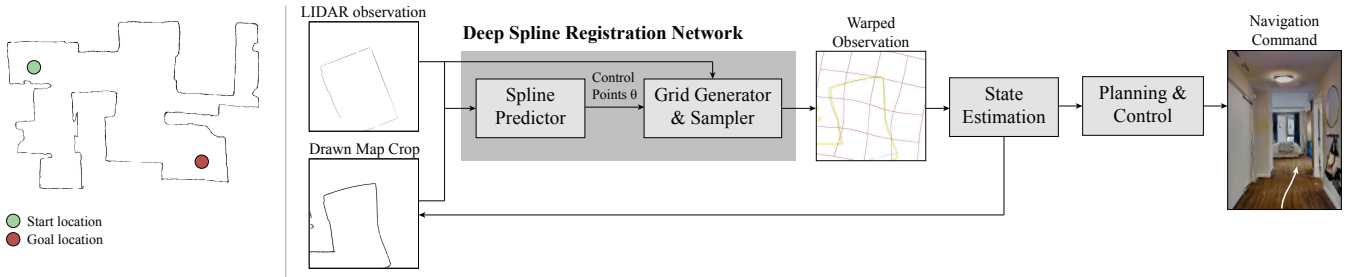
2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

# Localizing Against Drawn Maps via Spline-Based Registration

Kevin Chen[1], Marynel Vázquez[2], Silvio Savarese[1]




(a) Drawn Environment Map (b) Navigation Approach

Fig. 1: We propose a method for performing robot localization in hand-drawn maps (a) via spline-based registration (b). Our deep registration approach takes as input a LIDAR observation and a crop of the hand-drawn map, and predicts control points which are used to compute the transformation from one image into the other.

***Abstract*— We propose a method to facilitate robot navigation relative to sketched maps of human environments. Our main contribution centers around using thin plate splines for registering the robot's LIDAR observation with the hand-drawn maps. Thin plate splines are particularly effective for this task because they are able to handle many of the non-rigid deformations commonly seen in sketches of maps, which render traditional rigid transformations inappropriate. Our proposed approach uses a convolutional neural network to efficiently predict the control points which define the spline transform, from which we then compute the pose of the robot on the hand drawn map for navigation purposes. Our systematic evaluations in simulation using a synthetic dataset and real, hand-drawn sketches show that the proposed spline-based registration approach outperforms baseline methods.**

## I. Introduction

In recent times, we have seen an increased interest in enabling robots to follow navigation directions from humans. This can now be achieved through graphical user interfaces [1], [2], natural language commands [3], [4], [5], or even gestures [6]. But what about visual communication? People often convey directions to other people through drawings.

Inspired by [7], [8], [9], [10], we investigate methods to enable robots to localize against sketches of human environments, such that robots can follow navigation directions in such representations (Fig. 1a). Intuitively, if people drew precise maps, we could use existing 2D localization methods to follow drawn navigation instructions in human environments. But people often draw approximate maps, where straight lines may be slightly curvy. Likewise, angles, distances and areas may not be accurate.

[1]K. Chen and S. Savarese are with, respectively, the Department of Electrical Engineering and Department of Computer Science, Stanford University, Stanford, CA 94305, USA kevin.chen@cs.stanford.edu

[2]M. Vázquez is with the Department of Computer Science, Yale University, New Haven, CT 06520, USA

Our insight is that even though drawn maps are not precise, they are topologically consistent with the real world – otherwise, people would get lost. To enable robots to navigate relative to drawn maps, we propose a method for 2D localization via spline-based registration. In contrast to using more traditional rigid transformations, spline-based transformations allow for curved lines to be mapped to straight lines and vice-versa. Moreover, spline-based transformations do not necessarily preserve distances, areas, or angles. Unlike arbitrary displacement fields, spline-based transformations are more likely to preserve the topology of the environment.

We demonstrate how the proposed spline-based registration method enables a mobile robot equipped with a light detection and ranging (LIDAR) scanner to navigate human environments in Gibson [11], a perceptual and physics simulator often used for evaluating navigation tasks in realistic scenarios [12]. Enabling mobile navigation systems to localize against imprecise map representations opens up possibilities for new forms of human-robot interaction via pictures and drawings.

## II. Related Work

**Sketch-based navigation.** Prior works have investigated the use of hand-drawn maps for robot navigation from a human-robot interaction perspective. For example, [7], [8], [9], [10] propose systems in which the user uses a PDA (e.g. PalmPilot) to draw a map of the environment along with the desired trajectory for the robot. Methods such as [10] process the trajectory and represent the navigation task as a sequence of qualitative states with corresponding navigation behaviors. A set of rules is then used for computing the robot's state along the path and a pre-programmed or pre-learned controller executes the selected behavior. Our work is inspired by these approaches but seeks to estimate a precise

location of the robot in the map and does not rely on a hand-drawn trajectory.

More similarly, [13], [14] tackle navigation in sketched maps by extending the Monte Carlo Localization [15], [16], [17] algorithm. However, they assume sketch deformations are limited to scale and rotation.

**2D and 3D registration.** Registration is a well studied topic in 2D and 3D computer vision. A common approach for registration is to iteratively minimize an alignment objective. This has been demonstrated using keypoint detection and building hand-engineered feature descriptors [18], [19], [20], which can in turn be used for correspondence matching via methods such as RANSAC [21].

Recent success in deep learning has led researchers to investigate learned feature descriptors, which have shown to have great success for feature extraction [22], [23] and feature matching [24], [25], [26]. End-to-end models [27], [28], [29], [30] have also been proposed for predicting geometric transformations (e.g. affine, thin plate spline). Our work falls under this class of approaches, but focuses on spline-based registration of LIDAR scans for state estimation.

**Scan matching and localization in robotics.** Scan matching and localization have been studied extensively in robotics. For point set registration, iterative closest point (ICP) [31] is a commonly used approach for finding the best rotation and translation to align two point sets. The point-to-plane objective [32] has been shown to have faster convergence speed [33] compared to the point-to-point variant. However, this method requires strong initialization and takes a long time to converge. Fast global registration [34] was proposed to alleviate this issue and can be used in conjunction with ICP to perform accurate and fast registration. We compare with this as a baseline.

Monte Carlo Localization (MCL) [15], [16], [17] is one of the most widespread methods for robot localization. This approach can very accurately estimate the robot state but relies on a known metric map as well as motion and observation models, which we do not depend on. However, it would be interesting to investigate methods for combining our proposed method together with particle filters to smooth out localization predictions over time.

## III. Problem Statement

Our aim is to enable robots to navigate human environments depicted by 2D sketches. In this work, we assume that the sketches are contours of free space, which approximately describe the layout of the environment of interest and serve as a reference for navigation, e.g., as in Fig. 1(a). Further, we assume that the initial pose and desired destination for the robot on the sketch are given. Then, the main challenge to enable navigation in this setup consists of localizing the robot against the drawn map, given its current observations of the world. This is an interesting problem because sketches are inaccurate maps. People abstract spatial information [35] and often focus on drawing familiar and salient elements [36] rather than drawing precise outlines.

## IV. Approach

A summary of our navigation approach is depicted in Fig. 1b. During navigation, we repeatedly (1) register LIDAR scans with range observations of the world against the drawn map, (2) compute the state of the robot in the drawn map from the registration, (3) plan motion commands for the robot based on the current state and the desired goal, and (4) execute the motion commands.

For the first step, we propose to use the powerful approximation capabilities of neural networks to predict thin plate spline control points [37] which can then be used to register LIDAR observations of the world against sketches. Splines are advantageous for this problem because they allow for significant deformation while preserving the topology of the environment in most cases. For example, in Fig. 2, the ordering of elements in a 2D map is preserved under the transformation. More details about this registration approach, which is one of our main contributions, are provided in the next section.

We compute the robot's state as its 2D pose, $s = [p_x, p_y, \alpha]$, in the drawn map using the sampling grid from the spline registration network. The position $(p_x, p_y)$ of the robot in the drawn map is its warped location from the LIDAR image. The orientation $(\alpha)$ is computed using the sampling grid as well. In this case, we warp a horizontal line in the direction of the robot's orientation in the observation image. We then perform linearization on the warped line around the robot's position to calculate its direction in the sketch.

Finally, the robot uses the A* search algorithm [38] to generate a shortest path plan to the goal in the form of a sequence of positions. The plan is computed on the sketched map, using the current estimated pose of the agent and its goal position in the drawing. To move, the robot takes a discrete, fixed size step in the direction of the next position along the planned trajectory. This process of registration, state estimation, planning, and execution is repeated until the agent reaches the goal.

### A. Registration via Thin Plate Splines

The inputs to our registration model are two images, as depicted in Fig. 1b. The first image represents a LIDAR scan gathered by the robot in the environment and projected into a top-down view. The second image is a cropped section of the hand-drawn map of the environment. This cropped region is centered around the most recently predicted robot pose. We use a neural network to predict the control points of thin plate splines which can be used to warp the observation image to the hand-drawn map. This network is composed of differentiable components: a spline predictor network, and a grid generator and sampler, described in the next paragraphs.

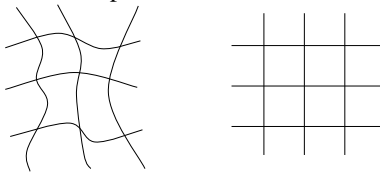


Fig. 2: The two diagrams are topologically equivalent.

**Spline Predictor.** Thin plate splines are a method for smooth interpolation along a fixed set of control points [37]. In the case of a 2D function $f(x,y)$, we can use thin plate splines to define a smooth surface that passes through a set of $N$ control points at positions $(x_i,y_i)$ with values $f(x_i,y_i)$. Formally, the surface is defined as:

$$f(x,y) = a_1 + a_2 x + a_3 y + \sum_{i=1}^{N} w_i U(|(x_i,y_i) - (x,y)|) \quad (1)$$

where $U(r) = r^2 \log r^2$, and $a_1$, $a_2$, $a_3$, and $w_i$ are parameters of the spline function. Note that eq. (1) can be written as $\mathbf{v} = A\mathbf{h}$, where:

$$\mathbf{v} = \begin{bmatrix} f(x_1,y_1) \\ f(x_2,y_2) \\ \vdots \\ f(x_N,y_N) \end{bmatrix}, \quad A = \begin{bmatrix} K & P \\ P^T & 0 \end{bmatrix}, \quad \mathbf{h} = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_N \\ a_1 \\ a_2 \\ a_3 \end{bmatrix} \quad (2)$$

$$K = \begin{bmatrix} U_{11} & U_{12} & \dots & U_{1N} \\ U_{21} & U_{22} & \dots & U_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ U_{N1} & U_{N2} & \dots & U_{NN} \end{bmatrix}, \quad P = \begin{bmatrix} 1 & x_1 & y_1 \\ \vdots & \vdots & \vdots \\ 1 & x_N & y_N \end{bmatrix} \quad (3)$$

with $U_{jk} = U(|(x_k,y_k) - (x_j,y_j)|)$. If the positions $(x_i,y_i)$ and values $f(x_i,y_i)$ of the control points are known, then the parameters of the spline can be computed as $\mathbf{h} = A^{-1}\mathbf{v}$.

To apply the above interpolation method to image registration, we can consider each pixel in the *warped* image to have a coordinate $(x_T,y_T)$. Our goal is then to compute where each of these pixels is located in the *pre-warped* image. For every pair of images to be registered, we compute two spline functions $f_1(\cdot)$ and $f_2(\cdot)$ to map pixel coordinates in the target image $(x_T,y_T)$ to their corresponding positions in the source image: $(x_S,y_S) = (f_1(x_T,y_T), f_2(x_T,y_T))$.

To compute the function parameters $\mathbf{h}_1$, $\mathbf{h}_2$ for $f_1(\cdot)$, $f_2(\cdot)$, we use a Convolutional Neural Network (CNN). Doing so allows the parameters to be quickly estimated in a single forward pass rather than iteratively.

We use a fixed $4 \times 4$ grid of target control points uniformly spread across the *warped* image and use the neural network to predict their corresponding positions in the source (unwarped) image. Thus, the neural network takes as input the two images $I_1$, $I_2$, each of shape $H \times W$ pixels, and predicts a tensor of shape $16 \times 2$ representing the source coordinates of the control points in the input to be warped:

$$\theta = \begin{bmatrix} \mathbf{v}_1 & \mathbf{v}_2 \end{bmatrix} = g(I_1, I_2) \quad (4)$$

The predicted source control points $\mathbf{v}_1$, $\mathbf{v}_2$ along with the matrix $A$ can be used to compute the spline parameters $\mathbf{h}_1$, $\mathbf{h}_2$. Note that $A$ can be calculated from the known target control points.

We use a sequential neural network architecture to implement eq. (4). The input images are concatenated along the channel dimension to produce a tensor of shape $H \times W \times 2$. This is passed through five convolution layers with stride 2, followed by two convolution layers with stride 1, and finally through two fully connected layers. Each layer uses batch normalization [39] and the ReLU activation function.

**Grid generation and sampler.** We use the two thin plate splines to generate a sampling grid which is used to compute the final warped image, similar to [27]. The sampling grid is generated by first normalizing the pixel coordinates in the target image $(x_T,y_T)$ to be in the range $[-1,1]$ and applying the functions $f_1(\cdot)$, $f_2(\cdot)$ to each coordinate. Therefore, each coordinate in the target image can be mapped to a corresponding spatial location in the input image. The grid is then used for sampling the input observation image and warp it based on the splines.

We use the integer sampling kernel and perform nearest neighbor sampling to preserve the structure of the input image.[1] In summary, the input LIDAR observation image is sampled according to the splines to generate the warped LIDAR image, which should ideally align with the hand-drawn map. As stated in the beginning of Sec. IV, the registration is then used for performing state estimation.

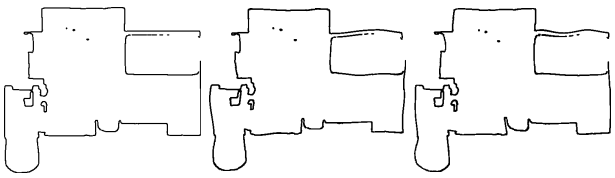
**Training objective.** We trained our model in a supervised manner with mean squared error loss on the control point positions: $J_{mse} = \frac{1}{M}\sum_{i=1}^{M} ||\theta_i - \hat{\theta}_i||^2$. To this end, we first collected a synthetic dataset consisting of $(I_1, I_2, \theta)$ triplets, where $I_1$ is the LIDAR observation and $I_2$ is a crop of the floor map perturbed using thin plate spline parameters $\theta$. The dataset was collected by capturing LIDAR observations at different poses in multiple Gibson environments [40]. We used the ADAM optimizer [41] with a fixed learning rate of 1e-4 to train the registration network end-to-end.

## V. Registration Evaluation with Synthetic Data

We evaluate the proposed registration approach using the Gibson simulator [40] and the Gibson dataset [40]. We process Gibson environments to generate a collection of ground truth floor maps, which are warped to create synthetic sketches. Warping is achieved by randomly sampling thin plate spline control points and applying them to the ground truth layouts. For each map in the environment, we sample and evaluate on 5 different perturbations. Figures 3b and 3c show examples of synthetic sketches from Fig. 3a.

In total, we train and evaluate on 573 and 10 different environments, respectively. The evaluation is performed on previously unseen environments, considering two registration errors: the average L2 error in translation, and the average angular (yaw) error. Errors are evaluated as the robot moves through pre-defined trajectories in the environments.

[1] In theory, any sampling kernel can be used [27].



(a) Ground truth (b) Synthetic 1 (c) Synthetic 2

Fig. 3: Original floorplan of the environment (a) and synthetically generated maps using thin plate splines (b, c).

### A. Baselines

We compare our proposed registration approach, as described in Sec. IV-A, with the following baselines:

**ICP.** Point-to-Plane Iterative Closest Point (ICP) method with fast global registration by Zhou et al. [34]. This registration method computes an initial global registration using features from fast point feature histograms (FPFH) [42]. The estimated initial transform is then used to initialize the point-to-plane ICP algorithm [32], which outputs a final rotation and translation to register the observation with the sketch.

**Ablation (Affine).** This is an ablation of our approach to evaluate performance using a rigid, isometric transform rather than the proposed spline-based transform. We use a Convolutional Neural Network to predict affine transformation parameters for image registration.

### B. Results

As shown in Fig. 5, our spline-based registration is able to handle non-rigid deformations that are infeasible to generate from traditional rigid transform approaches or even affine transforms. For example, in the second row of Fig. 5, the LIDAR observation shows a rectangular room with straight walls and right angles. However, the corresponding walls and corners on the perturbed floor map are geometrically very different from the original – straight lines now have curves, and the right angles are no longer 90 degrees. Nonetheless, the registration network is able to compute a transform which allows the images to be aligned well. Furthermore, our end-to-end trained network does not utilize explicit correspondence matching but is still able to accommodate mismatches between the LIDAR observations and floor maps. For example, in the bottom row of the Fig. 5, the LIDAR sensor gets returns from an object not visible in the floor map. However, the model is still able to successfully register the images.

Table I provides quantitative registration results. The learning approaches outperform ICP. From our observations, many of the large errors arise due to inaccurate transforms computed by the fast global registration algorithm. These likely occur due to mismatches between the synthetic sketch map and the LIDAR observation, such as rooms which are not visible from the LIDAR sensor or furniture seen in the observation but not captured in the floor map. Because ICP relies on an accurate initialization, the final predicted transformation is inaccurate and the algorithm tends to fail.

TABLE I: Registration results on the synthetic dataset.

| **Metric** | ICP | Ablation (Affine) | Ours |
|---|---|---|---|
| Translation Error (cm) | 235.95 | 168.39 | **46.25** |
| Angular Error (deg) | 95.09 | 57.11 | **18.16** |

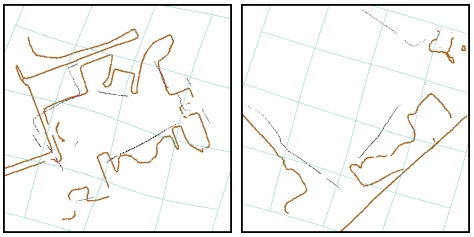


Fig. 4: Example failure cases with sketch (brown), overlay of warped observation (black), and sampling grid (light blue).

There is a natural limit in the capacity of the proposed model to align LIDAR observations with synthetic maps. If the images differ significantly, alignment becomes difficult. Fig. 4 shows two failure examples. In these cases, the objects in the map look very different in the LIDAR image.

## VI. Navigation Evaluation with Drawn Maps

To assess complete system performance, we evaluate the proposed approach on navigation tasks in the Gibson environments from Sec. V. For this evaluation, we created a second dataset of maps hand-drawn by human annotators. As shown in Fig. 6, many of the details in the ground truth map are lost in the drawings.

As before, we consider the ICP and Ablation (Affine) registration baselines for comparison. We use these baselines within our full navigation pipeline (Sec. IV) and compare navigation success rate against our full approach with the proposed spline registration method. A navigation attempt is considered successful if the robot reaches the goal location within 20 cm. Note that we do not evaluate using translational and rotational error since it is difficult to determine ground truth poses for real, hand-drawn maps.

### A. Results

Our model achieved a 31.67% success rate on navigation using hand-drawn floor maps (Table II), which is significantly higher than the 0% and 18.33% success rates achieved by the baseline approaches. In particular, the comparison with the affine ablation model illustrates the effect of choosing an appropriate transformation type for this task. Although the network architectures and training procedures for the two models are very similar, there is a significant performance gap, as the affine network struggled to predict suitable transformation parameters.

The model was trained on the synthetic dataset and tested on the hand-drawn dataset, and the data distribution and statistics differ between the two datasets. We suspect that this caused the networks to underperform. Leveraging self-supervised or semi-supervised learning methods could help address this problem in the future.

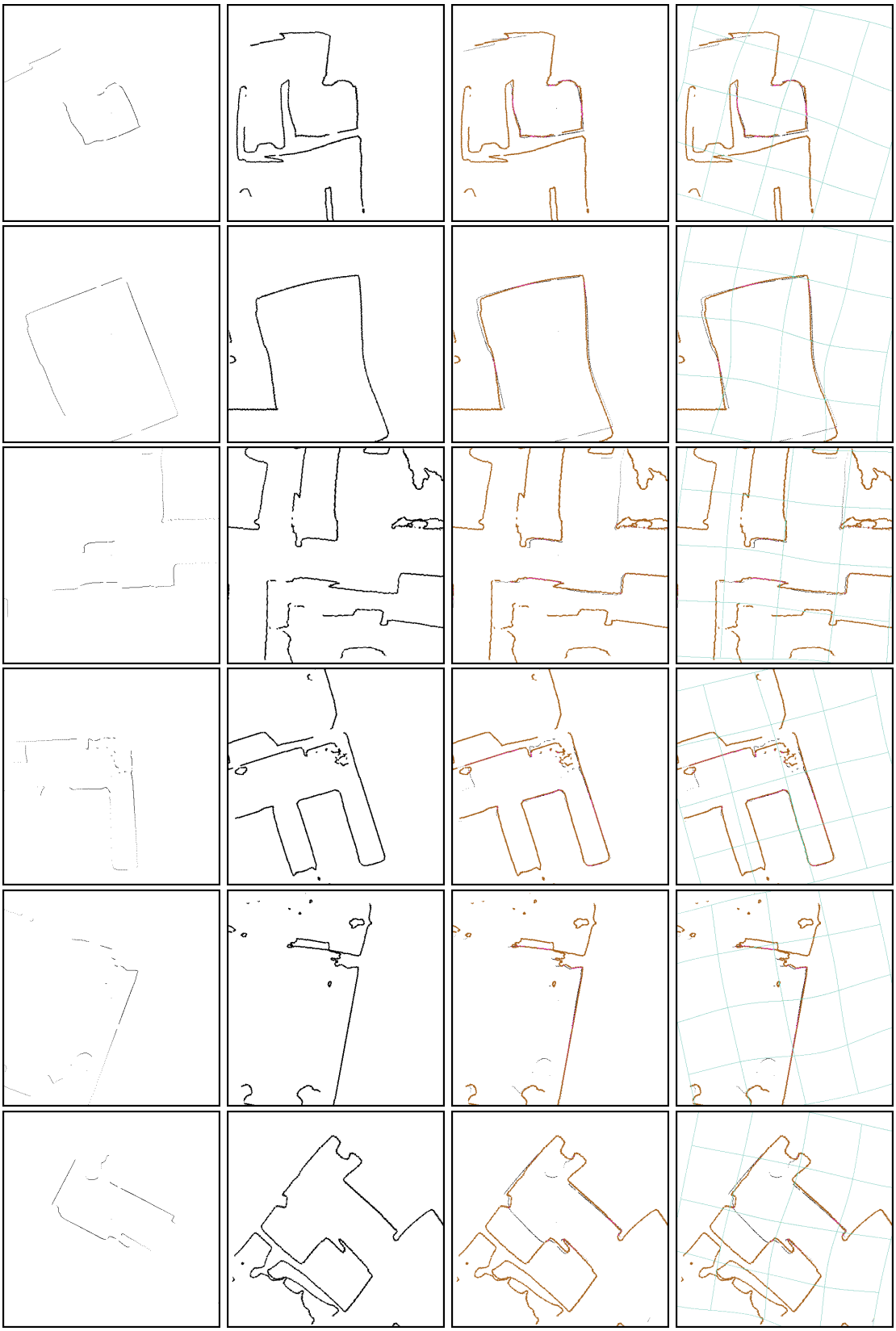
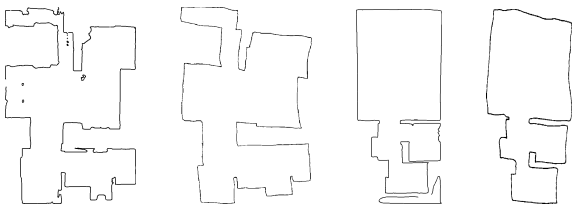


Fig. 5: Qualitative results of our spline-based registration pipeline. From left to right: LIDAR observation, synthetic sketched map, overlay of warped observation (black) and sketch (brown), and a grid (light blue) to visualize the warping.



(a) (b) (c) (d)

Fig. 6: Example of environments (a, c) and hand drawn maps (b, d) collected for our navigation evaluation.

TABLE II: Quantitative results on the hand drawing dataset.

| **Metric** | ICP | Ablation (Affine) | Ours |
|---|---|---|---|
| Success Rate (%) | 0.0 | 18.33 | **31.67** |

## VII. Conclusion & Future Work

We presented an approach for localization against imprecise environment representations using splines. The method is able to perform registration in settings that traditional rigid transformations cannot handle by design. Our experiments show that the proposed alignment approach is promising for localizing relative to hand-drawn maps. Future directions include reducing supervision and real-world evaluation.

## References


[1] C. W. Nielsen, M. A. Goodrich, and R. W. Ricks, "Ecological interfaces for improving mobile robot teleoperation," *IEEE Trans. on Robotics*, vol. 23, no. 5, pp. 927–941, 2007.

[2] A. Martinez and E. Fernández, *Learning ROS for robotics programming*. Packt Publishing Ltd, 2013.

[3] S. Tellex, T. Kollar, S. Dickerson, M. R. Walter, A. G. Banerjee, S. Teller, and N. Roy, "Understanding natural language commands for robotic navigation and mobile manipulation," in *AAAI*, 2011.

[4] C. Matuszek, E. Herbst, L. Zettlemoyer, and D. Fox, "Learning to parse natural language commands to a robot control system," in *Experimental Robotics*. Springer, 2013, pp. 403–415.

[5] X. Zang, A. Pokle, M. Vázquez, K. Chen, J. C. Niebles, A. Soto, and S. Savarese, "Translating navigation instructions in natural language to a high-level plan for behavioral robot navigation," in *EMNLP*, 2018.

[6] M. Obaid, M. Häring, F. Kistler, R. Bühling, and E. André, "User-defined body gestures for navigational control of a humanoid robot," in *ICSR*, 2012.

[7] G. Chronis and M. Skubic, "Sketch-based navigation for mobile robots," in *FUZZ*, vol. 1. IEEE, 2003, pp. 284–289.

[8] G. Chronis and M. Skubic, "Robot navigation using qualitative landmark states from sketched route maps," in *ICRA*, vol. 2. IEEE, 2004, pp. 1530–1535.

[9] M. Skubic, S. Blisard, A. Carle, and P. Matsakis, "Hand-drawn maps for robot navigation," in *AAAI Spring Symposium, Sketch Understanding Session*, 2002, p. 23.

[10] M. Skubic, P. Matsakis, B. Forrester, and G. Chronis, "Extracting navigation states from a hand-drawn map," in *ICRA*, vol. 1. IEEE, 2001, pp. 259–264.

[11] F. Xia, C. Li, K. Chen, W. B. Shen, R. Martín-Martín, N. Hirose, A. R. Zamir, L. Fei-Fei, and S. Savarese, "Gibson env v2: Embodied simulation environments for interactive navigation," Stanford University, Tech. Rep., 6 2019.

[12] P. Anderson, A. Chang, D. S. Chaplot, A. Dosovitskiy, S. Gupta, V. Koltun, J. Kosecka, J. Malik, R. Mottaghi, M. Savva, *et al.*, "On evaluation of embodied navigation agents," *arXiv preprint arXiv:1807.06757*, 2018.

[13] B. Behzadian, P. Agarwal, W. Burgard, and G. D. Tipaldi, "Monte carlo localization in hand-drawn maps," in *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 4291–4296.

[14] F. Boniardi, A. Valada, W. Burgard, and G. D. Tipaldi, "Autonomous indoor robot navigation using a sketch interface for drawing maps and routes," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2016, pp. 2896–2901.

[15] F. Dellaert, D. Fox, W. Burgard, and S. Thrun, "Monte carlo localization for mobile robots," in *ICRA*, vol. 2. IEEE, 1999, pp. 1322–1328.

[16] D. Fox, S. Thrun, W. Burgard, and F. Dellaert, "Particle filters for mobile robot localization," in *Sequential Monte Carlo methods in practice*. Springer, 2001, pp. 401–428.

[17] S. Thrun, D. Fox, W. Burgard, and F. Dellaert, "Robust monte carlo localization for mobile robots," *Artificial intelligence*, vol. 128, no. 1-2, pp. 99–141, 2001.

[18] D. G. Lowe, "Object recognition from local scale-invariant features," in *ICCV*, vol. 2. Ieee, 1999, pp. 1150–1157.

[19] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *CVPR*, vol. 1. IEEE, 2005, pp. 886–893.

[20] H. Bay, T. Tuytelaars, and L. Van Gool, "Surf: Speeded up robust features," in *ECCV*. Springer, 2006, pp. 404–417.

[21] M. A. Fischler and R. C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381–395, 1981.

[22] S. Kim, D. Min, B. Ham, S. Jeon, S. Lin, and K. Sohn, "Fcss: Fully convolutional self-similarity for dense semantic correspondence," in *CVPR*, 2017, pp. 6560–6569.

[23] S. Kim, D. Min, S. Lin, and K. Sohn, "Dctm: Discrete-continuous transformation matching for semantic flow," in *CVPR*, 2017, pp. 4529–4538.

[24] X. Han, T. Leung, Y. Jia, R. Sukthankar, and A. C. Berg, "Matchnet: Unifying feature and metric learning for patch-based matching," in *CVPR*, 2015, pp. 3279–3286.

[25] S. Zagoruyko and N. Komodakis, "Learning to compare image patches via convolutional neural networks," in *CVPR*, 2015, pp. 4353–4361.

[26] C. Choy, J. Park, and V. Koltun, "Fully convolutional geometric features," in *CVPR*, 2019, pp. 8958–8966.

[27] M. Jaderberg, K. Simonyan, A. Zisserman, *et al.*, "Spatial transformer networks," in *NeurIPS*, 2015, pp. 2017–2025.

[28] I. Rocco, R. Arandjelovic, and J. Sivic, "Convolutional neural network architecture for geometric matching," in *CVPR*, 2017, pp. 6148–6157.

[29] I. Rocco, R. Arandjelović, and J. Sivic, "End-to-end weakly-supervised semantic alignment," in *CVPR*, 2018, pp. 6917–6925.

[30] J. Chen, L. Wang, X. Li, and Y. Fang, "Arbicon-net: Arbitrary continuous geometric transformation networks for image registration," in *NeurIPS*, 2019, pp. 3410–3420.

[31] P. J. Besl and N. D. McKay, "Method for registration of 3-d shapes," in *Sensor Fusion IV: Control Paradigms and Data Structures*, vol. 1611, 1992, pp. 586–606.

[32] Y. Chen and G. G. Medioni, "Object modeling by registration of multiple range images." *Image Vision Comput.*, vol. 10, no. 3, pp. 145–155, 1992.

[33] S. Rusinkiewicz and M. Levoy, "Efficient variants of the icp algorithm," in *3DIM*, 2001, pp. 145–152.

[34] Q.-Y. Zhou, J. Park, and V. Koltun, "Fast global registration," in *ECCV*. Springer, 2016, pp. 766–782.

[35] Y.-f. Tuan, "Images and mental maps," *Annals of the Association of American geographers*, vol. 65, no. 2, pp. 205–212, 1975.

[36] N. T. Huynh, G. B. Hall, S. Doherty, and W. W. Smith, "Interpreting urban space through cognitive map sketching and sequence analysis," *The Canadian Geographer/Le Géographe canadien*, vol. 52, no. 2, pp. 222–240, 2008.

[37] F. L. Bookstein, "Principal warps: Thin-plate splines and the decomposition of deformations," *IEEE TPAMI*, vol. 11, no. 6, pp. 567–585, 1989.

[38] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.

[39] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[40] F. Xia, A. R. Zamir, Z. He, A. Sax, J. Malik, and S. Savarese, "Gibson env: Real-world perception for embodied agents," in *CVPR*, 2018.

[41] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[42] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (fpfh) for 3d registration," in *ICRA*. IEEE, 2009, pp. 3212–3217.