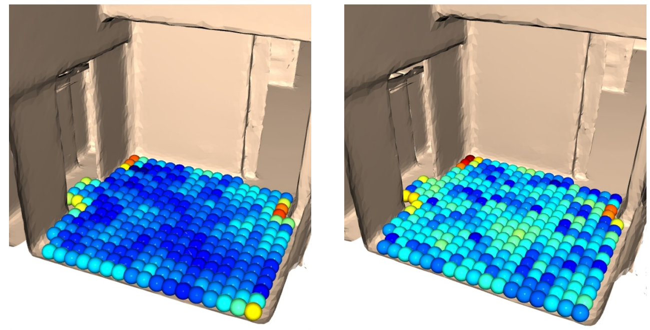
Deep Weakly Supervised Positioning for Indoor Mobile Robots

Ruoyu Wang[1*], Xuchu Xu[1*], Li Ding[2], Yang Huang[3], Chen Feng[1†]

***Abstract*—PoseNet can map a photo to the position where it is taken, which is appealing in robotics. However training PoseNet requires full supervision, where ground truth positions are non-trivial to obtain. Can we train PoseNet *without knowing the ground truth positions* for each observation? We show that *it is possible to do so via constraint-based weak-supervision*, leading to the proposed framework: DeepGPS. Particularly, using wheel-encoder-estimated distances traveled by a robot along with random straight line segments as constraints between PoseNet outputs, DeepGPS can achieve a relative positioning error of less than 2% for indoor robot positioning. Moreover, training DeepGPS can be done as auto-calibration with almost no human attendance, which is more attractive than its competing methods that typically require careful and expert-level manual calibration. We conduct various experiments on simulated and real datasets to demonstrate the general applicability, effectiveness, and accuracy of DeepGPS on indoor mobile robots and perform a comprehensive analysis of its robustness. Our code is avaible at: https://ai4ce.github.io/DeepGPS/**

***Index Terms*—Deep Learning for Visual Perception, Localization**



(a) DeepGPS (ours) (b) PoseNet

Fig. 1: Visual positioning by the proposed weakly-supervised DeepGPS (Figure 1a) and the fully-supervised PoseNet [14] (Figure 1b) on an indoor environment from the Gibson dataset [16]. Each sphere shows the testing location and the color-coded error (blue/red means small/large error). Best viewed in color.

I. Introduction

Visual localization/positioning has been a fundamental problem in robotics and computer vision which draws attention from researchers for a long time. Given a local observation, the goal of visual localization is to recover the sensor position in the scene where the local observation is captured. Depending on the modality of local observations, visual localization can be based on either image or point cloud. The ability of localizing the sensor plays a vital role in a variety of applications including virtual and augmented reality (VR/AR) [1], 3D reconstruction [2], and robotics [3]. In robotics particularly, commercial visual localization systems like Vicon are expensive yet popular options. Conventional visual localization approaches [4–6] typically rely on finding corresponding feature points/lines/planes [7, 8] between the local observation and the global scene, which are then used to determine the sensor position, typically within the RANSAC [9] framework to handle mismatched features.

Recently, deep learning methods have demonstrated compelling improvements in geometric computer vision problems, and many efforts have been made for visual localization from image [10, 11] and point cloud [12, 13]. Among them, PoseNet [14] is one of the first that formulates the visual localization as the absolute pose regression problem where the global scene is implicitly represented inside a deep neural network (DNN). Since then, this approach has been studied extensively. Despite different network architectures or loss functions, these methods follow the same supervised learning pipeline where DNNs are first trained on a large set of observations along with the corresponding ground truth sensor poses and then directly used to predict sensor poses for unseen observations. The performance of this approach strongly relies on the quantity and the quality of available training data. Benchmarking results [15] have shown that the performance would significantly drop when training data is limited. However, large-scale high-quality ground truth positions for input observations are non-trivial to obtain. The ground truth data collection requires either external devices such as the Vicon system or high-end GPS, or complex 3D reconstruction methods such as Structure-from-Motion (SfM) or Simultaneous Localization and Mapping (SLAM) that often calls for expert efforts. While a trained PoseNet can provide fast and efficient pose estimation, the requirement of ground truth data makes it less convenient for roboticists.

In this paper, we explore a different approach, DeepGPS, for visual localization using DNNs that does not require the ground truth sensor positions for each observation, yet still achieves comparable performance as PoseNet (see Figure 1). We notice that a mobile robot can often easily travel along a random straight line and accurately estimate the traveled distance using odometry from wheel encoders. This leads to

Manuscript received: September, 9, 2021; Revised November, 20, 2021; Accepted December, 6, 2021. This paper was recommended for publication by Editor Sven Behnke upon evaluation of the Associate Editor and Reviewers' comments. The research is supported by NSF CPS CMMI-1932187. * indicates equal contribution. † Chen Feng is the corresponding author.

[1]Ruoyu Wang, Xuchu Xu, and Chen Feng are with New York University, Brooklyn, NY 11201. {ruoyuwang, xuchuxu, cfeng}@nyu.edu

[2]Li Ding is with University of Rochester, Rochester, NY 14627. l.ding@rochester.edu

[3]Yang Huang is with University of California, Berkeley, CA 94720. yang_huang@berkeley.edu

Digital Object Identifier (DOI): see top of this page.

the key idea behind the proposed DeepGPS: instead of supervised learning on sensor positions for visual localization (such as PoseNet), we adopt the constraint-based weakly-supervised learning, where the constraints come from the distances between two sensor positions where the local observations are captured. The main advantage of this weakly-supervised approach is that the large-scale data collection can be performed fully automatically with almost no human attendance. The training can be considered as an auto-calibration, which is more attractive than alternative approaches, especially for indoor robot positioning, which typically involves expert-level manual calibration. To summarize, we make the following contributions:

- We find that it is feasible to train DNNs for visual positioning via constraint-based weakly supervised learning *without* ground truth positions.
- We propose the DeepGPS framework that uses distances between sensors, rather than sensor position itself, as the indirect supervision signals.
- We show by simulated and real-world experiments that DeepGPS achieves less than 2% relative error with various network architectures for different sensing modalities.
- We perform a robustness analysis of DeepGPS to improve our understanding of its training data requirements.

II. Related Works

There has been a large body of work on positioning systems using various sensing modalities. We refer readers to [17, 18] for comprehensive surveys of non-vision-based methods and focus our discussion on the vision-based methods.

Visual localization. Prior works on visual localization generally fall into two groups. The methods in the first category are based on matching features between the local and the global scenes. The scene environment can be explicitly represented as a 3D model, which is reconstructed from images or directly captured by 3D scanners. The correspondences can be established by matching hand-crafted [19, 20] or learning-based [12, 21] features. The downside is that these approaches require an explicit representation of the 3D model or have to store all descriptors and camera poses in memory. Additionally, several works use machine learning and deep learning methods to directly regress 3D coordinates of pixels in the image [22, 23]. While learning to predict 3D coordinates from image patches yields accurate sensor poses, these approaches may fail to handle large-scale scenes [24, 25].

Rather than learning to find corresponding points between local and global scenes, another type of learning-based approach, absolute pose regression (APR), uses DNNs to model the full localization pipeline. In this scenario, the global environment is implicitly represented by the weights of DNNs. The core idea is to build DNN models to directly predict the sensor positions for the input observations. The first APR-based approach is PoseNet [14] that uses a modified GoogLeNet [26] to regress camera poses. The network is trained by minimizing the Euclidean distance between the predicted poses and the ground truth poses. Since then, the APR-based approach for visual localization has been explored extensively. For example,

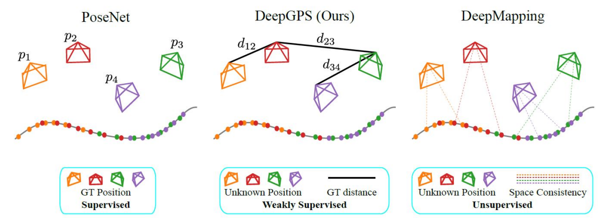


Fig. 2: Comparison of training procedures for various positioning methods: PoseNet [14] (supervised), the proposed DeepGPS (weakly supervised), and DeepMapping [42] (unsupervised). PoseNet requires ground truth (GT) poses. DeepMapping requires point clouds for the free-space consistency loss. Our DeepGPS only requires line segment traveling and distance measuring, which is easy for most of the mobile robots. DeepGPS can also use different observation modalities, including images and point clouds.

method in [27] incorporate additional loss terms such as the reprojection error and the weighted position and rotation loss. In addition to the novel loss function, different network architectures are proposed for extracting the fine-grained features [28] or for handling sequences of inputs [29]. Moreover, view synthesis techniques [30, 31] are integrated into the APR-based approach to enlarge the training dataset and to pose constraints on sensor poses. Methods in [32, 33] exploit other available supervision signals, including GPS, IMU, and VO, to improve the accuracy of localization. Despite various network architectures and loss functions, these methods follow the supervised pipeline and require a fully labeled training dataset that contains a massive amount of local observations and the associated ground truth positions. As noted in Section I, such ground truth positions are non-trivial to obtain and therefore pose a challenge for this supervised learning approach to be effectively deployed.

Learning-based visual odometry. Another related problem to visual localization is the visual odometry (VO) that incrementally estimates the ego-motion of the sensor using a sequence of observations. We briefly review the learning-based VO and refer readers to the survey paper [34] for broader context on the conventional methods. DeepVO [35] and VidLoc [36] propose to use a combined CNN and LSTM networks for monocular VO by taking the advantage that CNN and LSTM are capable to extract features from the input images and incorporating temporal information, respectively. In addition to the monocular cameras, different sensors are incorporated into the learning-based methods that allow the information from independent sensors to be fused in a complementary way, including lidar [37], GPS [38], or IMU [39]. Several works focus on unsupervised learning for the VO problem by jointly estimating the depth maps for the input RGB images [40, 41].

Beyond supervised learning. Most deep learning-based methods for visual localization are fully supervised. As noted, ground truth data collection is non-trivial. Therefore, a few alternative learning strategies have been investigated as well. In Fig. 2, we compare several training procedure for visual localization. Unsupervised learning, for example, avoids labeled data collection. The recent DeepMapping [42] introduces an unsupervised framework for multiple point clouds registration and mapping, where the supervision signals are derived from the free-space consistency among point clouds. The trained model can perform coarse re-localization for unseen point clouds that are in close vicinity to the registered point clouds. The main drawback of this method is that it is only applicable to point cloud modality.

III. DeepGPS Method

In this section, we first discuss the classic positioning problem assuming either correct measurement-landmark correspondences or known landmark positions. Then we motivate and detail our deep weakly supervised solution for positioning using images or point clouds where that assumption is often not valid. Finally, we discuss some important implementation details about our deep network and automatic data collection strategies for mobile robots.

A. Classical positioning without deep learning

In general, the positioning problem can be formulated as: given a set of local observations $\{\mathbf{x}_i\}_{i=1}^N$ collected at N different positions to a set of M fixed landmarks $\{\mathbf{m}_k\}_{k=1}^M$ in the environment, the positions $\{\mathbf{p}_i\}_{i=1}^N$ corresponding to the local observations needs to be estimated. An observation is obtained through an observation function of the sensor position, conditioned on the landmarks: $\mathbf{x}_i = h\left(\mathbf{p_i}|\{\mathbf{m}_k\}_{k=1}^M\right)$.

To solve the positioning problem, one usually *assumes that correct measurement-landmark correspondences are known*. For instance, in some RF-based beacon systems, observations are distances to RF signal transmitters installed in the environment. With the unique IDs decoded from the RF signals, the correspondence between observations and landmarks (transmitters) can be established. Under the known-correspondence assumption, the observation function for the k-th observed value in $\mathbf{x}_i$ (corresponding to the k-th landmark) is:

$$\mathbf{x}_i^{(k)} = \|\mathbf{m}_k - \mathbf{p}_i\|,\, k \in \mathbb{N}_{\leq M},\, i \in \mathbb{N}_{\leq N}. \tag{1}$$

Besides the correspondence assumption, one also needs to *know the value of landmark positions $\{\mathbf{m}_k\}_{k=1}^M$, otherwise landmarks have to be calibrated before positioning*. Once landmarks positions are known, the positioning problem can be solved by the classic least square estimation:

$$\mathbf{p}^* = \arg\min_{\mathbf{p}} \sum_{k=1}^{M} [\mathbf{x}^{(k)} - h\left(\mathbf{p_i}|\{\mathbf{m}_k\}_{k=1}^M\right)]^2. \tag{2}$$

B. Positioning from constraint-based weak supervision

However, the above two assumptions are not always valid. Especially, when $\mathbf{x}_i$ is an image or a point cloud, landmarks need to be identified and their positions are unknown, and the correspondences cannot be established without robust feature matching. Solving these issues automatically is non-trivial and often error-prone. To enable easier positioning, we develop **DeepGPS, a constraint-based weak supervision approach that does not require known landmark positions nor explicit measurement-landmark correspondences**.

To explain our approach, we first briefly introduce positioning from supervised learning. The goal of learning-based visual positioning is to find a function $f_\theta\left(\mathbf{x}\right)$, modeled as a DNN with learnable parameters θ, to estimate sensor position $\mathbf{p}$ without explicitly modeling landmarks nor solving equation (2). Given a mostly static environment, f is a bijection if the local observation $\mathbf{x}$ is independent with sensor orientation, and otherwise a surjection. The training dataset $\mathcal{D}_s = \{(\mathbf{x}_i, \mathbf{p}_i)\}_{i=1}^N$ contains a set of local observations $\mathbf{x}_i$ associated with the ground truth positions $\mathbf{p}_i$. The network parameters are obtained by minimizing the loss function:

$$\ell_s\left(\theta\right) = \sum_i |f_\theta\left(\mathbf{x}_i\right) - \mathbf{p}_i|, \tag{3}$$

which penalizes the disagreement between network predictions and the ground truth positions.

Differently, the proposed DeepGPS tackles the visual positioning problem as constraint-based weakly supervised learning, by specifying certain constraints over the network output [43]. Our training dataset is a graph $\mathcal{D}_{ws} = (\mathcal{V}, \mathcal{E})$ where nodes $\mathcal{V} = \{\mathbf{x}_i\}_{i=1}^N$ are local observations, and edges $\mathcal{E} = \{c_{ij}\}$ are constraints between sensor positions $\mathbf{p}_i$ and $\mathbf{p}_j$ defined by the constraint function $C\left(\cdot,\cdot\right)$. Note that this graph does not need to be fully connected, i.e., $\mathcal{E}$ does not necessarily contain constraints between all pairs of $\mathbf{p}_i$ and $\mathbf{p}_j$. A constraint c_{ij} exists in $\mathcal{E}$ only if it is accessible, i.e., when measured physically. In general, we minimize the following loss to penalize constraint violations,

$$\ell_{ws}\left(\theta\right) = \sum_{c_{ij}\in\mathcal{E}} |C\left(f_\theta\left(\mathbf{x}_i\right), f_\theta\left(\mathbf{x}_j\right)\right) - c_{ij}|. \tag{4}$$

As mentioned, the constraint we adopted is the Euclidean distance between two (unknown) sensor positions i and j where local observations are captured, i.e., $c_{ij} = \|\mathbf{p}_i - \mathbf{p}_j\|_2$, and the constraint function $C\left(\cdot,\cdot\right)$ calculates the distance between two network-predicted positions. Since wheel encoders or IMU is commonly available for mobile robots, the value of c_{ij} can be measured with high accuracy if the robot moves along the straight line between $\mathbf{p}_i$ and $\mathbf{p}_j$ without blocked by obstacles.

In practice, we found that the training would converge faster if we normalized each loss term with the sum of the estimated and the ground truth distances. Thus we use the following loss function throughout our experiments:

$$\ell_{ws}\left(\theta\right) = \sum_{c_{ij}\in\mathcal{E}} \frac{|\,\|f_\theta\left(\mathbf{x}_i\right) - f_\theta\left(\mathbf{x}_j\right)\| - c_{ij}|}{\|f_\theta\left(\mathbf{x}_i\right) - f_\theta\left(\mathbf{x}_j\right)\| + c_{ij}}. \tag{5}$$

It is worth stressing that in DeepGPS training, only scalar constraints, e.g, distance, are required. Neither absolute nor relative ground truth position vectors are needed.

Relationship to classical solution. Note that in the situation where the measurement-landmark correspondences are known, and the constraints are provided, the classical solution is also applicable, by solving the following equations:

$$\begin{cases} \|\mathbf{m}_k - \mathbf{p}_i\| = x_{ki},\, k \in \mathbb{N}_{\leq M}\,, i \in \mathbb{N}_{\leq N} \\ \|\mathbf{p}_i - \mathbf{p}_j\| = c_{ij},\, j \in \mathbb{N}_{\leq N}, \end{cases} \tag{6}$$

where all $\mathbf{m}$ and $\mathbf{p}$ are unknowns. Once they are solved, a new position $\mathbf{p}'$ can be triangulated via $\mathbf{m}$ and $\mathbf{x}'$. Such a situation is turned into our toy example experiment in section IV-A, which is used as a sanity check for our solution.

Why does it work? The effectiveness of DeepGPS has a connection with Multi-Dimensional Scaling (MDS) [44], a well-known technique in data mining and visualization. In effect, Eq. (4) provides a direct supervision for a function $C_\theta(\mathbf{x}_i, \mathbf{x}_j)$ that predicts the Euclidean distance between the positions i and j using the two observations, and this function has a Siamese structure denoted by f_θ. After the full supervision on C_θ, we can use it to compute a Euclidean Distance Matrix (EDM) for a set of positions inside that environment. From MDS, the coordinates of those positions can be computed by eigendecomposition on a centered version of EDM. If we choose this set of positions to be the densely sampled grid locations inside the environment, then in effect, we get a "GPS sensor" for that environment. *Interestingly, DeepGPS is more appealing because we do not need to actually go through the above MDS process: the "PoseNet" f_θ implicitly achieves it more efficiently*.

Global transformation ambiguity. Unlike PoseNet which is supervised directly by global positions, the DeepGPS output positions could be subject to an unknown global Euclidean transformation. So before evaluation, we need to align the DeepGPS output with the ground truth by solving for this transformation, which is a standard operation when computing the Absolute Trajectory Error [45] in SLAM.

C. Orientational invariant network architecture

The specific network architecture f_θ depends on the input modality $\mathbf{x}$. In our toy example experiment, the $\mathbf{x}$ is a distance-vector to each landmark, and f_θ is a Multi-layer Perceptron (MLP). When $\mathbf{x}$ is an image, we use ResNet-18 [46] as the backbone network. In such a situation, f_θ is usually not a bijection, since $\mathbf{x}$ depends not only on the position but also the orientation. To achieve orientation-invariant positioning, we use upward-viewing omnidirectional images as $\mathbf{x}$. Rotating an image taken at a position with a certain orientation is equivalent to taking another image at the same location with a corresponding orientation. Then we warp the observed images $\mathbf{x}(u, v)$ from the Cartesian coordinate system to the polar coordinate system, i.e., $\mathbf{x}(r, \phi)$. In this case, the problem is converted to ensure the translational invariance along the ϕ dimension.

To improve the network robustness and make the training easier, We employed the following two techniques to increase the rotational invariance of the network:

One is circular CNN. According to [47], a convolutional layer is translation-invariant when the output has the same height and width as the input via circular padding. Therefore, we modify the original ResNet-18 [46], by changing the kernel size of the first convolution layer from 7 to 3 and replacing all zero paddings with circular paddings.

The other technique is data augmentation. We randomly and circularly shift the warped image along the ϕ-dimension and feed them into the network during the training to simulate multiple observations at the same position with different orientations.

D. Observation and distance constraint collection.

We adopt two strategies to collect the traveling distance as the weak supervisions to train DeepGPS, i.e., the end-point strategy and the dense-sampling strategy.

In the end-point strategy, a robot sequentially visits random positions $\{\mathbf{p}_i\}_{i=1}^N$ in the target environment, collect local observation $\mathbf{x_i}$ at each position $\mathbf{p_i}$, and measure the travelling distance $c_{i-1,i}$ along a straight line in between.

While the end-point strategy allows robot to freely explore the environment, this data collection strategy is inefficient because no data is collected between two consecutive points $\mathbf{p}_{i-1}$ and $\mathbf{p}_i$. As a result, the total traveling distance is inevitably large to collect sufficient data. To overcome this issue, we further propose a dense-sampling strategy that is a line-segment-based data collection. Specifically, the trajectory of the robot contains L random sampled line segments $\{l_k\}_{k=1}^L$. When traveling along each line segment l_k, observations $\{\mathbf{x}_i^{(k)}\}_{i=1}^{N_k}$ are collected at a sequence of N_k points $\{\mathbf{p}_i^{(k)}\}_{i=1}^{N_k}$ along the line segments. The order of $\mathbf{p}_i^{(k)}$ follows the movement direction of the robot. The starting point on line segment l_k is $\mathbf{p}_1^{(k)}$ and $c_{i1}^{(k)}$ is recorded by the wheel encoder for all the sampling locations i traveled by the robot on the k-th line segment. More generally, $c_{ij}^{(k)} = |c_{i1}^{(k)} - c_{j1}^{(k)}|$ means the distance between any two sampling location i and j on the k-th line segment. Using the dense-sampling strategy, Eq. (5) becomes:

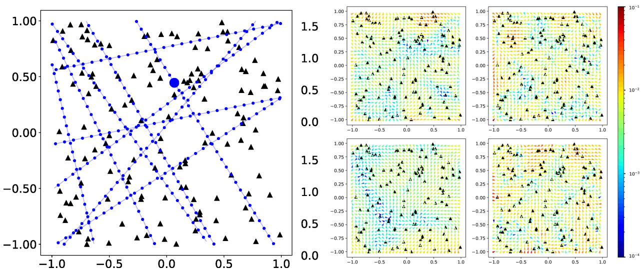
$$\ell_{ws}(\theta) = \sum_{k=1}^{L} \sum_{i=1}^{N_k-1} \sum_{j=i+1}^{N_k} \frac{\left|\left\|f_\theta(\mathbf{x}_i^{(k)}) - f_\theta(\mathbf{x}_j^{(k)})\right\| - c_{ij}^{(k)}\right|}{\left\|f_\theta(\mathbf{x}_i^{(k)}) - f_\theta(\mathbf{x}_j^{(k)})\right\| + c_{ij}^{(k)}}. \quad (7)$$

The end-point strategy can be viewed as a special case of the dense-sampling strategy where $N_k = 2, \forall k \in [1, L]$.

IV. EXPERIMENTS

In this section, we report the performance of our method on different modalities of observations. The experiments were performed under 3 types of environments with different levels of reality: *numerically simulated environment*, *virtualized photo-realistic environment*, and *real-world environment*. We evaluated our method for observations with correspondences in the *numerically simulated environment* as a toy example, and for observations without correspondences in the *virtualized photo-realistic environment* and the *real-world environment* as more realistic testings. We use the Absolute Trajectory Error (ATE) [45] as the evaluation metric. We use PoseNet [14] in all the environments as a baseline for DeepGPS. The settings of all baseline methods are the same as DeepGPS. In our project website, we also show experimental results on the simulated 2D lidar point cloud.

Baseline selection. We choose two baselines: PoseNet[14] representing learning-based ones, and MASFM [48] for SFM-/SLAM-based ones. For our real-world experiment, since ground truth positions with high accuracy are non-trivial to collect, we use the positions from MASFM as our "ground truth" positions to compare DeepGPS and PoseNet.



(a) Data generation (b) Qualitative results

Fig. 3: **Data generation and qualitative results of the toy example experiment.** Figure 3a illustrates the sensor positions (blue points) on 10 random line segments (blue lines) and the landmarks (black triangles). Figure 3b demonstrates the qualitative results. Top: PoseNet [14], Bottom: DeepGPS. Left: complete observation, Right: incomplete observation. The arrows indicate the error directions between the predicted and the ground truth positions, where the error magnitudes are color-coded.

Since the original PoseNet uses the SFM result as the ground truth poses, in our photo-realistic simulation experiment, in addition to training the PoseNet using accurate ground truth positions, we also train the PoseNet with noisy ground truth positions to simulate the noise introduced by the SFM. The noise follows $\mathcal{N}([0,0]^T, \sigma\mathbf{I})$, $\sigma = 0.01\text{m}$.

A. Toy example: numerically simulated environment

Data generation. In this experiment, we use the dense-sampling strategy to collect data. The left figure in Fig. 3a illustrates the data collection process. The whole environment is a 2D space bounded by $[-1, 1]$ on both dimensions. We place 128 landmarks in the environment (black triangles in Fig. 3a). In the training dataset, a simulated robot starts at a random position and moves along a straight line segment (blue line in Fig. 3a). Once the robot hits the boundary, it will choose another random orientation and move along a new line segment. The sampling distance is 0.02 between $\mathbf{p}_i^{(k)}$ and $\mathbf{p}_{i+1}^{(k)}$ on each line segment l_k. At each position, the robot measures the distance between sensor position and the landmarks forms a 128-dimensional vector. The entire training dataset contains 14,413 observations collected on 128 line segments. For testing, we uniformly sample a 128×128 grid in the environment as testing positions.

Completeness of observations. In practice, the robot may not be able to observe all landmarks at a certain location. To simulate such a situation, we set a maximum observing distance d_{max}. If the distance between the landmark and the observing position is larger than d_{max}, then the distance observation to that landmark is set to d_{max}. The top-right and bottom-right figures in Fig. 3a show the complete and the incomplete observations, respectively. We evaluated our method with both complete observation (all landmarks can be observed) and incomplete observation (not all landmarks can be observed). We set $d_{max} = 0.6$ for incomplete observation.

Training details. We use an MLP with 128, 512, 512, 512, 256, 256, 128, 64, 2 neurons in each layer. The total number of training epochs is 1500 with a batch size of 800. We use the Adam optimizer [49] with a learning rate of 0.001 in the first 300 epochs and 0.0001 in the remaining epochs.

TABLE I: Quantitative results for the toy example experiment.

ATE	Complete Observation			Incomplete Observation		
	RMS	Median	Max	RMS	Median	Max
PoseNet[14]	0.006	0.004	0.024	**0.007**	**0.006**	**0.069**
DeepGPS	**0.004**	**0.004**	**0.021**	0.009	0.007	0.111

Results. Table I lists the results of DeepGPS and the baseline methods. Our method performs better than PoseNet [14] with complete observations. The results of incomplete observations, while slightly worse than PoseNet, still demonstrate the effectiveness of DeepGPS. It is worth noting that DeepGPS only requires weak supervision, i.e., the distance between two observations, which are readily available in robot applications. Figure 3b illustrates the qualitative results. From the figure, we can see that most of the errors are less than 0.02 (1% relative to the side length of the environment).

B. Virtualized photo-realistic environment experiments

Data generation. We randomly loaded 30 rooms in the Gibson [16] dataset into Habitat-Sim [50]. For each room, we collected about 8,000 images for training using the dense-sampling strategy and the testing images were collected on grids with 0.08m intervals in unoccupied areas. On each testing position, we placed the robot towards 4 different orientations: $+x, -x, +y, -y$. The maximum error among the 4 orientations is used as the ATE at that location. Our project website shows sample original omnidirectional images taken in the experiment environment and the ones after the linear-polar warping (section III-C).

Training details. We use a modified ResNet18 as f_θ (section III-C). We train it for 1,500 epochs with a batch size of 100. We use an Adam optimizer with a learning rate of $1e-4$.

Results. Table II and Figure 4 show our quantitative and qualitative experiment results on selected scenes. Complete results are reported in our project website. Our method shows comparable performance with PoseNet. However, the ground truth is more convenient to collect for our method. The relative error of our method is about 2% (relative to the shorter side of the bounding box of the experiment environment). The main reason that our method has larger errors on some environments (e.g. the fourth column in Figure 4, Islandton) is that our random-walk strategy may not be able to cover enough area, which also influences PoseNet. It is reasonable to expect better performance by a better sampling strategy. Note that although the experiments are done in simulated environments, the results are still valuable since the images of scenes are photo-realistic.

C. Real-world environment experiments

Data generation. We mounted two cameras on a TurtleBot: a fisheye camera that looks upwards and a perspective

TABLE II: ATE for experiments in virtualized photo-realistic environment (Gibson/Habitat-Sim).

ATE (m)	Aloha	Brevort	Cokeville	Delton	Euharlee	Germfask	Islandton	Montreal	Sodaville	**Overall**
PoseNet [14]	**0.061**	0.060	**0.040**	0.060	**0.161**	0.037	0.338	0.072	0.211	**0.090**
	0.037	0.043	**0.032**	0.035	**0.043**	0.029	0.121	0.048	0.173	0.051
	0.931	0.434	**0.388**	0.500	**2.135**	0.235	**2.368**	0.582	0.860	**0.828**
PoseNet [14] (noisy GT)	0.133	1.003	1.108	1.096	1.076	0.932	1.176	1.025	1.255	1.054
	0.093	0.922	1.098	1.005	0.981	0.869	1.131	0.919	1.226	0.998
	0.984	2.354	2.447	2.613	2.325	2.208	2.472	2.213	2.376	2.465
DeepGPS	0.103	**0.048**	0.056	**0.041**	0.217	**0.022**	**0.289**	**0.044**	**0.095**	0.101
	0.053	**0.026**	0.036	**0.023**	0.044	**0.016**	**0.090**	**0.034**	**0.070**	**0.045**
	0.726	**0.393**	0.470	**0.290**	2.599	**0.103**	2.573	**0.440**	**0.576**	0.870

For each method, the top/middle/bottom row shows the root-mean-square/ median/max error. All errors are measured in meter. Note that the overall result is the average over all 30 scenes (see our project website), not only the scenes in this table.

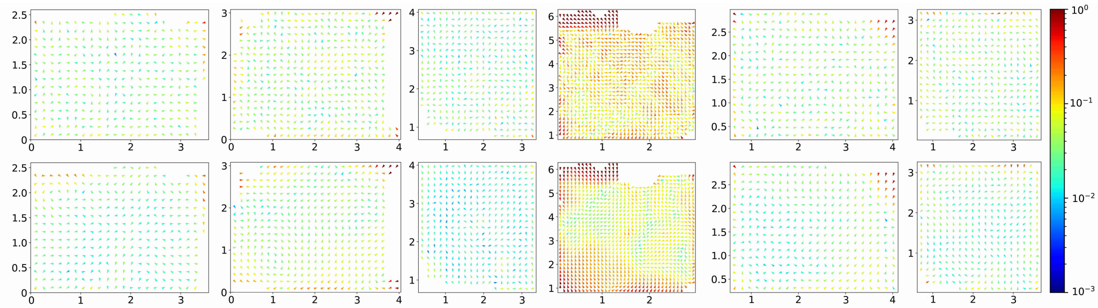


Fig. 4: **Sample visual results of positioning error in the virtualized photo-realistic environment experiments.** Top: PoseNet; bottom: DeepGPS. We show the localization errors computed from a discrete grid. The arrows indicate the error directions between the predicted positions and the ground truth, where the error magnitudes are color-coded. The unit is in meter.

TABLE III: ATE for real-world experiments

ATE (m)	RMS		Median		Max	
	Mean	Std.	Mean	Std.	Mean	Std.
PoseNet[14]	0.105	0.053	**0.032**	0.005	0.762	0.457
DeepGPS	**0.053**	0.021	0.036	0.004	**0.175**	0.026

The unit is in meter.

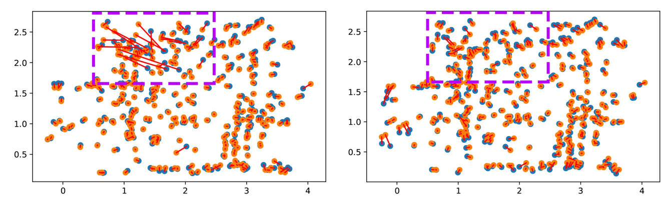


Fig. 6: **Visual results of positioning error in the real-world environment experiments.** Left: PoseNet [14], Right: DeepGPS. Blue dots are predicted positions and orange dots are ground truth positions. Red lines indicate the error magnitude between them. The purple box shows the area where PoseNet has larger error. The unit is in meter.

camera that looks horizontally. The images from the fisheye camera are used for providing visual localization input and the images from the perspective camera are used for obtaining the ground truth positions only for evaluation. We placed 82 AprilTags [51] in the environment. The 6-D poses of the tags were reconstructed using MASFM [48]. Then the groud truth positions were obtained by the perspective camera via solving the PnP problem. We collected 6 datasets in the environment using the end-point strategy and the train-test split for each dataset is 1,570/314. The experiment environment is shown in figure 5.

Training details. The network architecture and training setup are the same as we use in section IV-B.

Results. Table III lists the quantitative result of our method in the real-world environment. The mean and standard deviation of the ATEs are calculated among the 6 experiments. Interestingly, we found that the maximum ATE of our method is significantly smaller than PoseNet [14], while the median ATE is very close between the two methods. Figure 6 is one example of our experiments, which shows DeepGPS has fewer large errors than PoseNet. We believe it is because PoseNet suffers more from overfitting the positioning errors in the real-world ground truth.

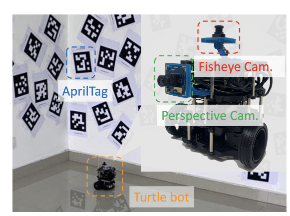


Fig. 5: Real-world experiments setup. A turtlebot (mounted with perspective and fisheye cameras) moves inside the environment with pre-calibrated AprilTags.

V. Robustness Analysis

We perform a robustness analysis on our method to obtain better insights on it. Here, we study how the level of noise in the distance measurement and the number of training samples influence the positioning accuracy. We evaluate the influence of these two factors using both the toy example environment that has observations with correspondences and the virtualized photo-realistic environment.

A. Level of noise

We add random noise to each distance measurement between two consecutive positions. The noise-contaminated distance measurement $\tilde{c}_{i,i+1}$ between two consecutive positions $\mathbf{p}_i$ and $\mathbf{p}_{i+1}$ can be modeled as:

$$\tilde{c}_{i,i+1} = c_{i,i+1} + n_{i,i+1}, \quad (8)$$

where $c_{i,i+1}$ is the noise-free distance measurement, and $n_{i,i+1}$ is the random Gaussian noise $\mathcal{N}(0, \sigma_{i,i+1}^2)$, where the standard deviation $\sigma_{i,i+1}$ is proportional to noise-free distance, controlled by a factor w i.e., $\sigma_{i,i+1} = wc_{i,i+1}$. In this experiment, we evaluate under $w = 0, 0.02, 0.04, 0.08, 0.10$.

Figure 7 depicts the results. Although the max error grows faster with the increasing level of noise, the RMS and median are all less than 0.04 (2% relative to the side length of the experiment environment), even if at 10% of the noise level. The results indicate that only few testing instances are sensitive to the noise. In practice, the noise level of a wheel encoder is less than 0.1%, which is far less than the evaluation range (2.0%-10.0%). For example, the resolution of the wheel encoder of the DYNAMIXEL-X series motor is 4096 pulse/rev[1], which is equivalent to 0.02% relative error for distance measurement. Therefore, our method is robust to the noise in distance measurement.

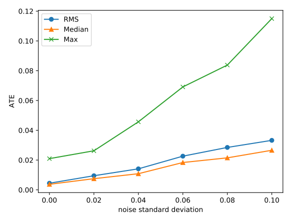


Fig. 7: DeepGPS ATE vs. noise level.

B. Number of training samples

Figure 8 shows performance under the different number of training samples. The dimension of the room is $7\,\mathrm{m} \times 4.3\,\mathrm{m}$. Our method can achieve $0.04\,\mathrm{m}$ accuracy with small number of training samples (1,000 samples, 33.2 samples/m^2). We notice that the ATE has a significant drop between $4,000$ to $5,000$ training samples, and after that, the ATE almost remains at the same level. Therefore, for this room, there is an optimal number of training samples, which is around 5,000. To increase the efficiency of the data collection, one of our future work is to make our method training on the fly, i.e., the training and data collection process happen at the same time, and the data collection terminates when the optimal number of the training sample is reached.

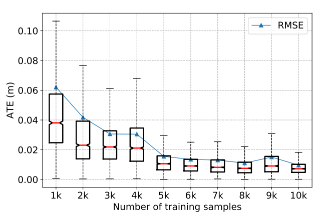


Fig. 8: DeepGPS ATE vs. number of training samples.

VI. Conclusion

In this paper, we propose a new weakly-supervised positioning system: DeepGPS. This method uses distances between positions as supervision signals instead of directly using positions. The proposed method can significantly decrease the cost and difficulty of ground truth data collection while still maintaining high accuracy. The proposed method does not heavily depend on the modality of sensors, thus can be generalized to many conditions and environments. Besides, the proposed method is fully end-to-end, without explicit models of the sensor and the environment, which saves the setup time and avoids data association, which is a non-trivial problem in traditional visual localization. One limitation of this method is that the performance will drop when the layout of the environment is highly concave. This problem can be alleviated by introducing sensors that can measure distances through obstacles.

In the future, we will extend our method to 3D cases and apply it to large-scale environment.

[1]https://emanual.robotis.com/docs/en/dxl/x//#dynamixel-x/

Acknowledgment

The authors would like to thank S. Farokh Atashzar, Wilko Schwarting, and anonymous reviewers for helpful suggestions.

References

[1] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison, "DTAM: Dense tracking and mapping in real-time," in *Proc. IEEE Int'l Conf. Computer Vision (ICCV)*, Nov 2011, pp. 2320–2327.

[2] P. Tanskanen, K. Kolev, L. Meier, F. Camposeco, O. Saurer, and M. Pollefeys, "Live metric 3D reconstruction on mobile phones," in *Proc. IEEE Int'l Conf. Computer Vision (ICCV)*, Dec 2013, pp. 65–72.

[3] H. Lim, S. Sinha, M. Cohen, M. Uyttendaele, and H. J. Kim, "Real-time monocular image-based 6-DoF localization," *Int'l J. Robotics Research*, vol. 34, pp. 476–492, April 2015.

[4] M. Donoser and D. Schmalstieg, "Discriminative feature-to-point matching in image-based localization," in *Proc. IEEE Conf. Comp. Vision and Pattern Recog. (CVPR)*, 2014, pp. 516–523.

[5] L. Liu, H. Li, and Y. Dai, "Efficient global 2D-3D matching for camera localization in a large-scale 3D map," in *Proc. IEEE Int'l Conf. Computer Vision (ICCV)*, 2017, pp. 2372–2381.

[6] L. Ding and G. Sharma, "Fusing structure from motion and lidar for dense accurate depth map estimation," in *Proc. IEEE Int'l Conf. Acoustics, Speech, and Signal Processing (ICASSP)*, 2017, pp. 1283–1287.

[7] Y. Taguchi, Y.-D. Jian, S. Ramalingam, and C. Feng, "Point-plane slam for hand-held 3D sensors," in *Proc. IEEE Int'l Conf. Robotics and Auto. (ICRA)*, 2013, pp. 5182–5189.

[8] A. Pumarola, A. Vakhitov, A. Agudo, A. Sanfeliu, and F. Moreno-Noguer, "PL-SLAM: Real-time monocular visual slam with points and lines," in *2017 IEEE international conference on robotics and automation (ICRA)*, 2017, pp. 4503–4508.

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

[10] P.-E. Sarlin, C. Cadena, R. Siegwart, and M. Dymczyk, "From coarse to fine: Robust hierarchical localization at large scale," in *Proc. IEEE Conf. Comp. Vision and Pattern Recog. (CVPR)*, 2019, pp. 12 716–12 725.

[11] G. Avraham, Y. Zuo, T. Dharmasiri, and T. Drummond, "EMP-Net: Neural localisation and mapping using embedded memory points," in *Proc. IEEE Int'l Conf. Computer Vision (ICCV)*, 2019, pp. 8120–8129.

[12] R. Dubè, A. Cramariuc, J. Dugas, Daniel andNieto, R. Siegwart, and C. Cadena, "SegMap: 3D segment mapping using data-driven descriptors," in *Proc. Robotics: Science and Systems (RSS)*, 2018.

[13] W. Wang, M. R. U. Saputra, P. Zhao, P. Gusmao, B. Yang, C. Chen, A. Markham, and N. Trigoni, "DeepPCO: End-to-end point cloud odometry through deep parallel neural network,"

in *Proc. IEEE/RSJ Int'l Conf. Intelligent Robots and Systems (IROS)*, 2019, pp. 3248–3254.

[14] A. Kendall, M. Grimes, and R. Cipolla, "PoseNet: A convolutional network for real-time 6-DOF camera relocalization," in *Proc. IEEE Int'l Conf. Computer Vision (ICCV)*, 2015, pp. 2938–2946.

[15] T. Sattler, Q. Zhou, M. Pollefeys, and L. Leal-Taixe, "Understanding the limitations of CNN-based absolute camera pose regression," in *Proc. IEEE Conf. Comp. Vision and Pattern Recog. (CVPR)*, June 2019.

[16] F. Xia, A. R. Zamir, Z. He, A. Sax, J. Malik, and S. Savarese, "Gibson Env: real-world perception for embodied agents," in *Proc. IEEE Conf. Comp. Vision and Pattern Recog. (CVPR)*. IEEE, 2018.

[17] R. F. Brena, J. P. García-Vázquez, C. E. Galván-Tejada, D. Muñoz-Rodriguez, C. Vargas-Rosales, and J. Fangmeyer, "Evolution of indoor positioning technologies: A survey," *Journal of Sensors*, vol. 2017, 2017.

[18] F. Zafari, A. Gkelias, and K. K. Leung, "A survey of indoor localization systems and technologies," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, pp. 2568–2599, 2019.

[19] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int'l J. Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[20] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *Proc. IEEE Int'l Conf. Computer Vision (ICCV)*, 2011.

[21] R. Dubè, D. Dugas, E. Stumm, J. Nieto, R. Siegwart, and C. Cadena, "SegMatch: Segment based place recognition in 3D point clouds," in *Proc. IEEE Int'l Conf. Robotics and Auto. (ICRA)*, 2017, pp. 5266–5272.

[22] E. Brachmann and C. Rother, "Learning less is more - 6D camera localization via 3D surface regression," in *Proc. IEEE Conf. Comp. Vision and Pattern Recog. (CVPR)*, 2018.

[23] D. Massiceti, A. Krull, E. Brachmann, C. Rother, and P. H. Torr, "Random forests versus neural networks—what's best for camera localization?" in *Proc. IEEE Int'l Conf. Robotics and Auto. (ICRA)*, 2017, pp. 5118–5125.

[24] H. Taira, M. Okutomi, T. Sattler, M. Cimpoi, M. Pollefeys, J. Sivic, T. Pajdla, and A. Torii, "InLoc: Indoor visual localization with dense matching and view synthesis," in *Proc. IEEE Conf. Comp. Vision and Pattern Recog. (CVPR)*, 2018.

[25] J. L. Schònberger, M. Pollefeys, A. Geiger, and T. Sattler, "Semantic visual localization," in *Proc. IEEE Conf. Comp. Vision and Pattern Recog. (CVPR)*, 2018, pp. 6896–6906.

[26] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comp. Vision and Pattern Recog. (CVPR)*, 2015.

[27] A. Kendall and R. Cipolla, "Geometric loss functions for camera pose regression with deep learning," in *Proc. IEEE Conf. Comp. Vision and Pattern Recog. (CVPR)*, July 2017.

[28] I. Melekhov, J. Ylioinas, J. Kannala, and E. Rahtu, "Image-based localization using hourglass networks," in *Proc. IEEE Int'l Conf. Computer Vision Wksp. (ICCV Wksp.)*, 2017, pp. 879–886.

[29] J. F. Henriques and A. Vedaldi, "MapNet: An allocentric spatial memory for mapping environments," in *Proc. IEEE Int'l Conf. Computer Vision (ICCV)*, 2018.

[30] J. Wu, L. Ma, and X. Hu, "Delving deeper into convolutional neural networks for camera relocalization," in *Proc. IEEE Int'l Conf. Robotics and Auto. (ICRA)*, 2017, pp. 5644–5651.

[31] T. Naseer and W. Burgard, "Deep regression for monocular camera-based 6-DoF global localization in outdoor environments," in *Proc. IEEE/RSJ Int'l Conf. Intelligent Robots and Systems (IROS)*, 2017, pp. 1525–1530.

[32] S. Brahmbhatt, J. Gu, K. Kim, J. Hays, and J. Kautz, "Geometry-aware learning of maps for camera localization," in *Proc. IEEE Conf. Comp. Vision and Pattern Recog. (CVPR)*, June 2018.

[33] N. Radwan, A. Valada, and W. Burgard, "VLocNet++: Deep multitask learning for semantic visual localization and odometry," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 4407–4414, Oct 2018.

[34] F. Fraundorfer and D. Scaramuzza, "Visual odometry: Part II: Matching, robustness, optimization, and applications," *IEEE Robotics Automation Mag.*, vol. 19, no. 2, pp. 78–90, June 2012.

[35] S. Wang, R. Clark, H. Wen, and N. Trigoni, "DeepVO: Towards end-to-end visual odometry with deep recurrent convolutional neural networks," in *Proc. IEEE Int'l Conf. Robotics and Auto. (ICRA)*, 2017, pp. 2043–2050.

[36] R. Clark, S. Wang, A. Markham, N. Trigoni, and H. Wen, "VidLoc: A deep spatio-temporal model for 6-dof video-clip relocalization," in *Proc. IEEE Conf. Comp. Vision and Pattern Recog. (CVPR)*, 2017, pp. 6856–6864.

[37] F. Ma, G. V. Cavalheiro, and S. Karaman, "Self-supervised sparse-to-dense: Self-supervised depth completion from lidar and monocular camera," in *Proc. IEEE Int'l Conf. Robotics and Auto. (ICRA)*, 2019, pp. 3288–3295.

[38] S. Pillai and J. J. Leonard, "Towards visual ego-motion learning in robots," in *Proc. IEEE/RSJ Int'l Conf. Intelligent Robots and Systems (IROS)*, 2017, pp. 5533–5540.

[39] C. Wang, Y. Yuan, and Q. Wang, "Learning by inertia: Self-supervised monocular visual odometry for road vehicles," in *Proc. IEEE Int'l Conf. Acoustics, Speech, and Signal Processing (ICASSP)*, 2019, pp. 2252–2256.

[40] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe, "Unsupervised learning of depth and ego-motion from video," in *Proc. IEEE Conf. Comp. Vision and Pattern Recog. (CVPR)*, 2017, pp. 1851–1858.

[41] R. Li, S. Wang, Z. Long, and D. Gu, "UnDeepVO: Monocular visual odometry through unsupervised deep learning," in *Proc. IEEE Int'l Conf. Robotics and Auto. (ICRA)*, 2018, pp. 7286–7291.

[42] L. Ding and C. Feng, "DeepMapping: Unsupervised map estimation from multiple point clouds," in *Proc. IEEE Conf. Comp. Vision and Pattern Recog. (CVPR)*, June 2019.

[43] R. Stewart and S. Ermon, "Label-free supervision of neural networks with physics and domain knowledge," in *Proc. the Thirty-First AAAI Conference on Artificial Intelligence (AAAI)*, 2017, pp. 2576–2582.

[44] J. B. Kruskal, *Multidimensional scaling*. Sage, 1978, no. 11.

[45] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of RGB-D SLAM systems," in *Proc. IEEE/RSJ Int'l Conf. Intelligent Robots and Systems (IROS)*, 2012, pp. 573–580.

[46] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comp. Vision and Pattern Recog. (CVPR)*, 2016, pp. 770–778.

[47] O. S. Kayhan and J. C. v. Gemert, "On translation invariance in cnns: Convolutional layers can exploit absolute spatial location," in *Proc. IEEE Conf. Comp. Vision and Pattern Recog. (CVPR)*, 2020, pp. 14 274–14 285.

[48] C. Feng, V. R. Kamat, and C. C. Menassa, "Marker-assisted structure from motion for 3d environment modeling and object pose estimation," in *Construction Research Congress 2016*, 2016, pp. 2604–2613.

[49] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Int'l Conf. Learning Representations (ICLR)*, 2015.

[50] M. Savva, A. Kadian, O. Maksymets, Y. Zhao, E. Wijmans, B. Jain, J. Straub, J. Liu, V. Koltun, J. Malik, D. Parikh, and D. Batra, "Habitat: A Platform for Embodied AI Research," in *Proc. IEEE Int'l Conf. Computer Vision (ICCV)*, 2019.

[51] E. Olson, "AprilTag: A robust and flexible visual fiducial system," in *Proc. IEEE Int'l Conf. Robotics and Auto. (ICRA)*, 2011, pp. 3400–3407.