# Differentiable Mapping Networks: Learning Task-Oriented Latent Maps with Spatial Structure

**Peter Karkus**[1,2], **Anelia Angelova**[1], **and Rico Jonschkowski**[1]
[1] Robotics at Google
[2] National University of Singapore
karkus@comp.nus.edu.sg, anelia@google.com, rjon@google.com

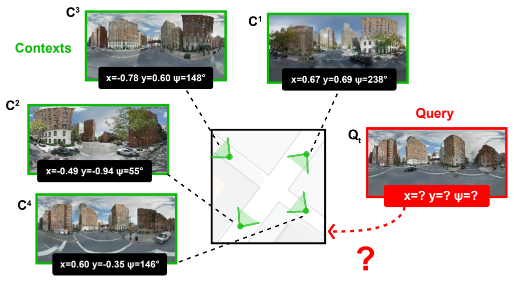
## 1 Introduction

To efficiently operate in previously unseen environments, robots must be able to build a *map* – an internal representation of the environment – even from a small number of observations. But how should that map be represented and which information should be stored in it, to enable downstream tasks, for example localization? Classic approaches use a fixed map representation with strong spatial structure, such as voxels or point clouds, which makes them applicable to wide range of robotic tasks. Data-driven approaches, on the other hand, are able to learn rich and robust representations by optimizing them directly for a downstream task. Eslami et al. [4], for example, learn to construct representations of simulated environments from a few images that allow them to generate images from novel viewpoints. The challenge for learning in complex environments is choosing suitable *inductive bias* that enable generalization while having only limited amount of data for training. A desirable approach would combine the best of both worlds: retain the spatial structure of the classic approaches, but also leverage the power of deep neural networks to learn a flexible and effective map representation for the downstream task.

In this paper we explore how structure and learning can be combined in the context of a sparse visual localization task. In the task a robot has access to a handful of visual observations from known viewpoints to build a map, and then it needs to localize with respect to this map given a new sequence of visual observations. This task is challenging due to the small number of observations in which the relevant spatial information is encoded in rich visual features. While the task setting is not widely studied in the literature (with one notable exception [25]), it has practical importance for high impact real-world applications, such as i) localizing a vehicle anywhere on earth using only an onboard camera and pre-collected visual data, e.g., Street View [16]; ii) mapping for multi-robot cooperative tasks with limited communication bandwidth, *e.g.*, for search and rescue; and iii) localization in warehouses with frequently changing visual appearance.

We propose the Differentiable Mapping Network (DMN), that learns a map representations for the sparse localization task while using the spatial structure of the problem. Given a set of image-pose pairs the DMN constructs a structured latent map representation that consists of pairs of viewpoint poses and learned image embeddings. Based on this map the model performs visual localization using a differentiable particle filter [11, 12]. Instead of generating observations as in [25], we use the map discriminatively to evaluate candidate poses for localization. Importantly, during localization the map is transformed into the coordinate frame of each particle, which provides the model with a prior for simple spatial reasoning.

We evaluate our method in synthetic 3D environments [4] and on real-world Street View data [16]. During training, we sample different environments, let the model generate a map from context image-pose pairs, run particle filter localization for query images, and optimize similarity of the predicted and true query poses. During inference, we test the model in previously unseen environments. Our results show that the DMN is effective in a variety of sparse localization settings and that the high performance critically depends on the structured map representation.



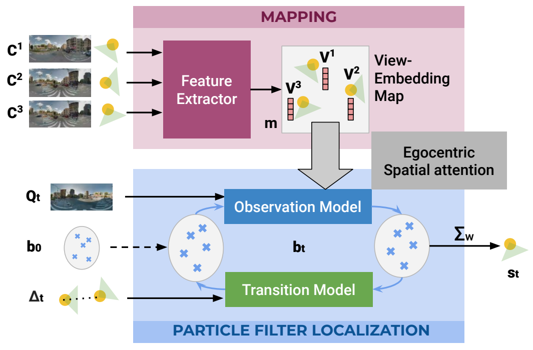


Figure 1: Sparse visual localization. The robot receives 4 context image-pose pairs and a query image; and estimates the pose of the query image.

Figure 2: The Differentiable Mapping Network (DMN) schematic.

## 2 Related work

Classic approaches to mapping and localization rely on fixed, structured map representations, such as grids [27], point clouds [9], feature points [18], and landmarks [17]. With the emergence of data-driven methods, especially deep learning, there has been increasing interest in learning to represent the map in a latent memory [14, 29, 31, 4, 25]. Since generic neural networks require large amounts of training data [4], encoding spatial structure in the latent memory is a promising direction that has been explored in the context of mapping [8, 7, 10], localization [24, 22], and SLAM [21, 8, 3]. A neural network can even incorporate structure by encoding modules of *differentiable algorithms* [13]. Examples include robot localization [11, 12], planning [26, 19, 5, 6], and control [20, 2, 23].

Our work is aligned with these methods, but focuses on differentiable mapping and learns a latent map optimized for a downstream task (in our case sparse localization). Instead of using the map to generate observations as in [25], we use it discriminatively to evaluate candidate poses for localization: when querying the map with a query observation and coordinate, viewpoints are transformed into the query coordinate frame and embeddings are combined with the query image features though a spatial attention mechanism. Compared to unstructured map representations [4, 25], our structured map allows for generalization from limited training data, while the differentiable representation allows for powerful task-oriented learning.

## 3 Differentiable Mapping Network (DMN)

**Sparse visual localization.** In the sparse visual localization task (Fig. 1) a robot receives a handful of RGB-image and pose pairs from a previously unseen environment, which we will refer to as *context*, $C^i$, $i = 1 : N_c$. From this sparse context the robot constructs a *map*, $m$, and uses the map for subsequent localization, i.e. to estimate the pose $s_t$ for a sequence of *query* observations from novel viewpoints, $Q_t$. The robot is given the relative ego motion between time steps, $\Delta_t$, and an initial belief, $b_0$, that is potentially uninformed uniform. Query observations are assumed to be of the same environment as the contexts, but they can be taken potentially much later in time, making the task especially challenging in dynamic environments, *e.g.*, city streets.

**The DMN Architecture.** We introduce DMN, a novel neural network architecture that learns a structured view-embedding map for sparse localization (Fig. 1). For each context $C^i$ the network computes an embedding (top left of Fig. 1). This set of embeddings and poses taken together make up the map representation (top right), which is used for a particle filter localization (bottom). We use a differentiable particle filter (PF), similar to [11, 12]. Starting with a set of particles based on the initial belief $b_0$, the observation model updates the particle weights by comparing the query observation $Q_t$ to the map using an egocentric spatial attention, i.e. attention based on the relative poses of the context views and the particle. The transition model updates the particle poses using odometry $\Delta_t$, i.e., known relative motion between time steps. At each time step the pose prediction $s_t$ is computed by the weighted mean of particles poses. For simplicity we refer to the full approach of building a differentiable map and its use in particle filter localization as DMN.

**Mapping.** The map in DMN, $m$, is represented by a set of $N_l$ latent view embeddings, $m = \langle f^i, s^i \rangle_{i=1:N_c}$, where each view embedding consists of a latent feature tensor $f^i$ and a planar pose $s^i = (x, y, \psi)$, where $\psi$ is the yaw. The features are extracted from the corresponding context image with a learned CNN (with shared weights across contexts).

**Egocentric spatial attention.** To allow reading the map from different query viewpoints, we introduce an *egocentric spatial attention* mechanism (Fig. 3). The attention mechanism computes a weighted sum of latent view-embedding features, using the scalar product of a *query key* and a *view key* as the weight, similarly to standard attention [28]. Our attention is *spatial*: it extracts view keys jointly from the view-embedding features and viewpoint coordinates. Specifically, we concatenate view-embedding features with viewpoint coordinates in the channels dimension, and pass them through a CNN. The attention is also *egocentric*: we transform viewpoint coordinates of the map into an egocentric coordinate frame where the query viewpoint is the origin. The egocentric spatial attention leverages the spatial structure of the map, and it is expected to reduce the difficulty of extracting the useful features of the map for a particular query viewpoint.

**Particle filter localization.** Based on the map, DMN performs subsequent sequential localization with a differentiable PF. The PF represents the belief as a set of weighted particles, $b_t(s) \approx \langle s_t^k, \log w_t^k \rangle_{k=1:K}$ where each particle $s_t^k$ correspond to a candidate pose of the robot ($x$, $y$, $\psi$). In the first step particles are sampled from the input initial belief $b_0$. In each following step, particles are updated given new observations and the latent map. Particle poses are updated with a fixed, analytic transition model, $s_t^k = f_T(s_{t-1}^k, \Delta_t)$, given the odometry input $\Delta_t$. Particle weights are updated with a learned observation model, $\log w_t^k = \log l_\theta(o_t, s_t^k, m) + \log w_{t-1}^k + \eta$, given the query image observation $o_t$, the particle pose $s_t^k$, the latent map $m$ and a normalization factor $\eta$.

The observation model connects the latent map with the downstream localization task with egocentric spatial attention (Fig. 4). By definition, the observation model estimates the conditional log-probability of observing $o_t$, given a pose $s_t^k$ and the map $m$, that is, $l_\theta(o_t, s_t^k, m) \approx \log p(o_t | s_t^k, m)$. Our observation map is discriminative: it takes in $o_t$, $s_t^k$, and $m$, and it outputs a real value, $l_t^k$, a direct estimate of the particle log-likelihood. The network first extracts features from $o_t$, the query image, using a CNN. It then uses the query features together with the particle pose to read the map with egocentric spatial attention. The attention outputs features of the latent map specific to the particle pose and the query image. The map features are then concatenated with query image features (extracted using a similar CNN as for view embeddings) and passed through a convolutional–dense component with a single output unit. The output unit defines $l_t^k$, the log-likelihood estimate of the particle. Log-likelihoods are estimated for each particle this way (learnable parameters are shared across particles) and used to update particle weights.

We define an output pose estimate at time $t$ as the mean of the particle poses weighted by the particle weights, $\overline{s}_t = \sum_k w_t^k s_t^k$. One could alternatively use a weighted kernel density estimate, e.g., a mixture of Gaussians as in [11]. Note that $s_t$ depends on all DMN inputs $(C^i, Q_t, \Delta_t, b_0)$ through the learned feature extractor for view-embeddings and the learned observation model. Because of the differentiable architecture gradients can propagate from $s_t$ to all learnable parameters.

**End-to-end training.** Since the DMN is end-to-end differentiable, we can train it to optimize mapping for the task of localization. We use a training loss of the mean-squared-error (MSE) between the pose estimate $\overline{s}$ and true query pose $s^*$: $\mathcal{L} = ||\overline{s} - s^*||^2 = (\overline{x} - x^*)^2 + (\overline{y} - y^*)^2 + \alpha(\overline{\psi} - \psi^*)^2$, where $\alpha$ is a constant parameter. We use $\alpha = 0.5$ in experiments.

## 4 Experiments

**Domains and datasets.** We evaluate our approach in two established simulated domains from prior work [4, 25], Rooms and Mazes, as well as in the real-world StreetLearn dataset [16], which consists of StreetView images from urban environments such as New York City's Manhattan. Localization in this dataset is particularly challenging because the data is collected at different times in a highly dynamic environment.

**Baselines.** We compare DMN to the following baselines: a latent map representations without explicit spatial structure (*image map*, *vector map*), which are similar to related work [4, 25]; regression instead of particle filtering; the *closest context*, which is an upper bound on image similarity based approaches; and an *uninformed estimate*, the mean belief updated with odometry inputs.

| Approach |  | Sequential loc. (↑) |  | One-step loc. (↑) |  |
|---|---|---|---|---|---|
| Map | Algorithm | Rooms | Mazes | Rooms | Mazes |
| View-embedding | PF-DMN (ours) | 95.6% | **75.9%** | **93.5%** | **27.0%** |
| View-embedding | Regr. (ours) | **97.2%** | 39.9% | 60.5% | 19.0% |
| Latent image | PF | 94.0% | 69.4% | 90.5% | 21.1% |
| Latent image | Regression | 9.6% | 10.1% | 5.1% | 5.5% |
| Latent vector | PF | 92.9% | 40.4% | 88.1% | 13.2% |
| Latent vector | Regression | 9.8% | 16.8% | 5.6% | 7.3% |
| *Closest context* |  | 11.0% | 39.3% | 11.4% | 34.5% |
| *Uninformed estimate* |  | 1.7% | 3.3% | 2.9% | 3.7% |

Table 1: Global localization results for the Rooms and Mazes datasets.

| Approach |  | Sequential | One-step |
|---|---|---|---|
| Map | Algorithm | loc. (↑) | loc. (↑) |
| View-embedding | PF (ours) | **73.2%** | **28.4%** |
| View-embedding | Regression (ours) | 42.3% | 13.4% |
| Latent image | PF | 37.8% | 9.8% |
| Latent image | Regression | 11.1% | 10.1% |
| Latent vector | PF | 15.8% | 5.7% |
| Latent vector | Regression | 10.4% | 7.1% |
| *Closest context* |  | 8.6% | 8.7% |
| *Uninformed estimate* |  | 2.7% | 7.1% |

Table 2: Global localization results for the StreetView dataset.

**Evaluation.** We consider two uncertainty settings similar to [12]: *global localization*, where $b_0$ is uniform over the environment; and *tracking*, where initial belief is a local Gaussian distribution around the true initial pose. We train all models in the tracking setting (with $t = 1$, $N_c = 4$, $K = 32$). We evaluate trained model for one-step global localization ($t = 1$), sequential global localization ($t = 5$), and tracking ($t = 5$). The evaluation metric for global localization is success rate (SR) defined as the percentage of trials where the MSE is under $0.2$ square units (equivalent to 8.94m radius for Street view); for tracking it is the average MSE at time $t$ computed for the $x$-$y$ dimensions. During evaluation we use $K = 2048$ for global localization and $K = 256$ for tracking.

**Main results.** Table 1 shows results for global localization in the Mazes and Rooms domains. Models with view-embedding maps consistently outperform the other map representations. Particle filtering is also useful: while regression can do well in the simple Rooms domain, we see a large benefit for PF-DMN in the complex Mazes domain (75.9% vs 39.9% at $t = 5$). PF can propagate a potentially multimodal belief distribution over time, in contrast with the regression network that only propagates a point estimate. Results for *Closest context* verify that image-similarity based approaches cannot do well in these domains. Table 2 shows results for the StreetView dataset. This is challenging real-world dataset. Baselines perform poorly and generally achieve less than 16% or 11% success. Sequential localization with PF is the only baseline with more reasonable success rates of 37.8%. In contrast, our method is able to achieve about 2x higher accuracy of 73.2%. This experiment shows more distinctly the importance of using view-based learned map (73.2% vs 37.8%) as well as the importance of particle filtering (73.2% vs 42.3%).

**Additional experiments.** Preliminary results for a set of additional experiments, that did not fit in the page limit, suggest that the benefit of the view-embedding maps over unstructured alternatives is larger 1) when training with smaller amounts of data, which can be attributed to the strong spatial prior; 2) when more contexts are available for mapping, which can be attributed to our maps dynamic representation capacity; 3) when the environment to map is much larger, which can to attributed to the egocentric spatial attention mechanism used to query the view-embedding map. When comparing PF with regression, we observed increasing benefit for longer trajectories, due to PF's ability to propagate a probabilistic estimate through time. Further, PF performance improves with increasing number of particles at test time, allowing a trade-off between performance and computation cost. We also considered a different downstream task for mapping, and found that optimizing the map for the downstream task can be critically important. Finally, preliminary results for ablations of DMN suggest that using an egocentric coordinate frame for viewpoints is important, while the specific attention only provides a small added advantage.

## 5 Conclusion

We introduced a differentiable mapping network optimized for a downstream sparse visual localization and demonstrated strong performance across different domains. Our results on structured and task-oriented map representation learning may provide valuable insights for key unsolved questions in designing embodied AI systems. Future work may explore extensions of our view-based latent map structure, *e.g.*, multiple landmarks placed at learned relative locations. A particularly interesting direction is to extend the proposed work to visual SLAM, where new observations could be treated as additional context, and uncertain context poses could be encoded in particles.

## Acknowledgement

We thank Dan Rosenbaum for fruitful discussions and for suggesting the Street View dataset.

## References

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, et al. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015.

[2] Brandon Amos, Ivan Jimenez, Jacob Sacks, Byron Boots, and J Zico Kolter. Differentiable MPC for end-to-end planning and control. In *Neurips*, 2018.

[3] Gil Avraham, Yan Zuo, Thanuja Dharmasiri, and Tom Drummond. Empnet: Neural localisation and mapping using embedded memory points. *arXiv preprint arXiv:1907.13268*, 2019.

[4] SM Ali Eslami, Danilo Jimenez Rezende, Frederic Besse, Fabio Viola, Ari S Morcos, Marta Garnelo, Avraham Ruderman, Andrei A Rusu, Ivo Danihelka, Karol Gregor, et al. Neural scene representation and rendering. *Science*, 360(6394):1204–1210, 2018.

[5] Gregory Farquhar, Tim Rocktäschel, Maximilian Igl, and Shimon Whiteson. TreeQN and ATreeC: Differentiable tree planning for deep reinforcement learning. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018.

[6] Arthur Guez, Théophane Weber, Ioannis Antonoglou, Karen Simonyan, Oriol Vinyals, Daan Wierstra, Rémi Munos, and David Silver. Learning to search with MCTSnets. In *International Conference on Machine Learning*, pages 1822–1831, 2018.

[7] Saurabh Gupta, James Davidson, Sergey Levine, Rahul Sukthankar, and Jitendra Malik. Cognitive mapping and planning for visual navigation. In *CVPR*, pages 2616–2625, 2017.

[8] Joao F Henriques and Andrea Vedaldi. Mapnet: An allocentric spatial memory for mapping environments. In *CVPR*, pages 8476–8484, 2018.

[9] Peter Henry, Michael Krainin, Evan Herbst, Xiaofeng Ren, and Dieter Fox. Rgb-d mapping: Using kinect-style depth cameras for dense 3d modeling of indoor environments. *The International Journal of Robotics Research*, 31(5):647–663, 2012.

[10] Tri Huynh, Michael Maire, and Matthew R Walter. Multigrid neural memory. *arXiv preprint arXiv:1906.05948*, 2019.

[11] Rico Jonschkowski, Divyam Rastogi, and Oliver Brock. Differentiable particle filters: End-to-end learning with algorithmic priors. *Robotics: Science and Systems*, 2018.

[12] Peter Karkus, David Hsu, and Wee Sun Lee. Particle filter networks with application to visual localization. In *CoRL*, pages 169–178, 2018.

[13] Peter Karkus, Xiao Ma, David Hsu, Leslie Pack Kaelbling, Wee Sun Lee, and Tomás Lozano-Pérez. Differentiable algorithm networks for composable robot learning. *Proceedings of Robotics: Science and Systems*, 2019.

[14] Alex Kendall, Matthew Grimes, and Roberto Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *CVPR*, pages 2938–2946, 2015.

[15] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[16] Piotr Mirowski, Matt Grimes, Mateusz Malinowski, Karl Moritz Hermann, Keith Anderson, Denis Teplyashin, Karen Simonyan, Andrew Zisserman, Raia Hadsell, et al. Learning to navigate in cities without a map. In *Neurips*, 2018.

[17] Michael Montemerlo, Sebastian Thrun, Daphne Koller, Ben Wegbreit, et al. Fastslam: A factored solution to the simultaneous localization and mapping problem. *AAAI*, 2002.

[18] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. ORB-SLAM: a versatile and accurate monocular SLAM system. *IEEE transactions on robotics*, 31(5):1147–1163, 2015.

[19] Junhyuk Oh, Satinder Singh, and Honglak Lee. Value prediction network. In *Neurips*, 2017.

[20] Masashi Okada, Luca Rigazio, and Takenobu Aoshima. Path integral networks: End-to-end differentiable optimal control. *arXiv preprint arXiv:1706.09597*, 2017.

[21] Emilio Parisotto and Ruslan Salakhutdinov. Neural map: Structured memory for deep reinforcement learning. In *ICLR*, 2018.

[22] Emilio Parisotto, Devendra Singh Chaplot, Jian Zhang, and Ruslan Salakhutdinov. Global pose estimation with an attention-based recurrent network. In *CVPR Workshop on Deep Learning for Visual SLAM*, 2018.

[23] Marcus Pereira, David D Fan, Gabriel Nakajima An, and Evangelos Theodorou. MPC-inspired neural network policies for sequential decision making. *arXiv preprint arXiv:1802.05803*, 2018.

[24] Noha Radwan, Abhinav Valada, and Wolfram Burgard. Vlocnet++: Deep multitask learning for semantic visual localization and odometry. *IEEE Robotics and Automation Letters*, 3(4):4407–4414, 2018.

[25] Dan Rosenbaum, Frederic Besse, Fabio Viola, Danilo J Rezende, and SM Eslami. Learning models for visual 3d localization with implicit mapping. *arXiv preprint arXiv:1807.03149*, 2018.

[26] Aviv Tamar, Sergey Levine, Pieter Abbeel, Yi Wu, and Garrett Thomas. Value iteration networks. In *Neurips*, 2016.

[27] Sebastian Thrun. Learning occupancy grid maps with forward sensor models. *Autonomous robots*, 15(2):111–127, 2003.

[28] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, pages 5998–6008, 2017.

[29] Amir R Zamir, Tilman Wekel, Pulkit Agrawal, Colin Wei, Jitendra Malik, and Silvio Savarese. Generic 3d representation via pose estimation and matching. In *ECCV*. Springer, 2016.

[30] Amir Roshan Zamir and Mubarak Shah. Image geo-localization based on multiplenearest neighbor feature matching usinggeneralized graphs. *IEEE Transactions on PAMI*, 36(8):1546–1558, 2014.

[31] Jingwei Zhang, Lei Tai, Joschka Boedecker, Wolfram Burgard, and Ming Liu. Neural slam: Learning to explore with external memory. *arXiv preprint arXiv:1706.09520*, 2017.

## A Illustrations of the DMN architecture

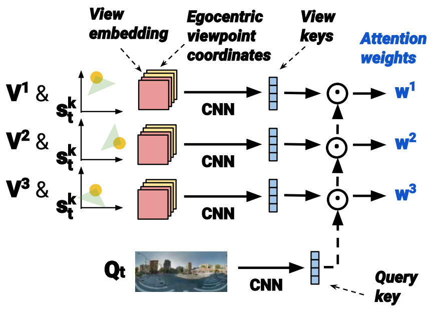


Figure 3: Egocentric spatial attention

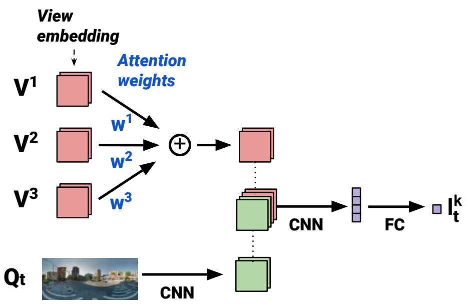


Figure 4: Discriminative observation model.

We illustrate the discriminative observation model in Fig. 4, and the egocentric spatial attention mechanism in Fig. 3.

## B Visualizations

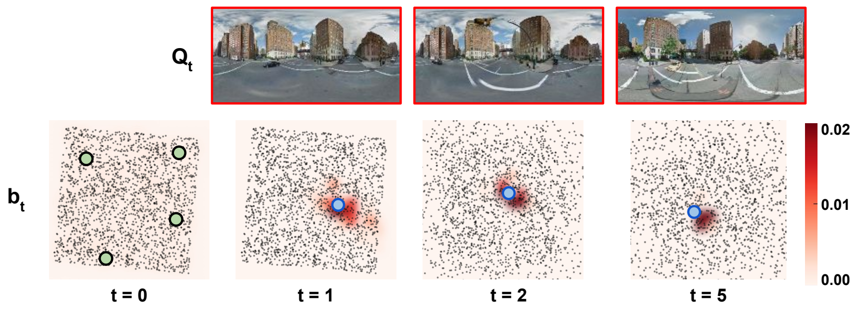


Figure 5: Sequential global localization example. The figure shows particle beliefs in a trained DMN for context inputs shown in Fig. 1. Particle weights are visualized as a heat map aggregated over all yaw values. Context poses are shown in green. The unknown query pose is shown in blue.

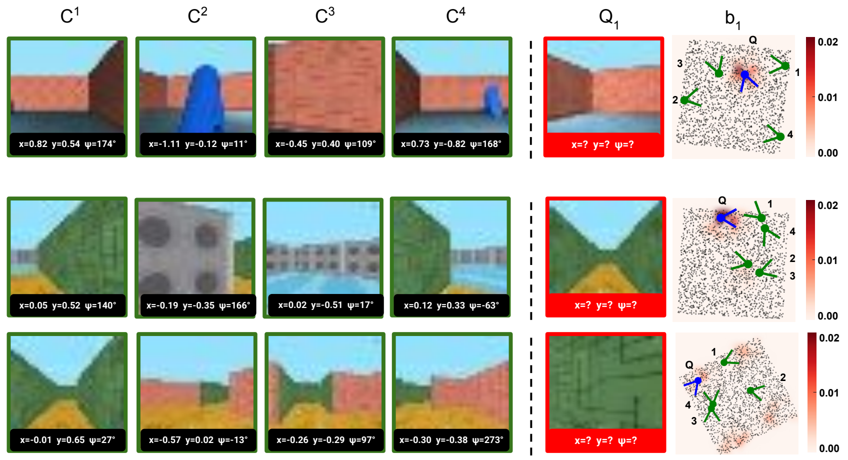


Figure 6: Example for one-step global localization in the Rooms and Mazes domains. The last column shows the DMN belief distribution as a 2D histogram of weighted particles. Darker red corresponds to larger probability density.

We visualize examples for global localization with DMN. Fig. 1 shows an example for sequential localization for the StreetView dataset. Fig. 6 shows examples for one-step localization for the Rooms and Mazes domains.

## C Experiment details

### C.1 Domains and datasets

**3D simulated environments.** We use the *Rooms* (with object rotation) and *Mazes* domains from the dataset [4]. In the Rooms domain each environment is a single square shaped room with randomly generated appearance and 1-3 random objects at a random location. The data contains 10 RGB images from each environment along with the camera poses. In the Mazes domain each environment is a large maze with randomly generated layout and appearance, giving rise to highly ambiguous observations. The data contains 300 images per environment taken at random locations. To train DMN we sample $N_c$ context image-pose pairs and $N_q$ query image-pose pairs. For sequential localization we sample multiple query image-pose pairs and form a trajectory by randomly ordering them. We use 1% of the original training set for training, i.e., 100k environments for Rooms and 960 environments for Mazes; and 5000 environments from the original test set for evaluation. We resize RGB images to $32\times32\times3$. Pose coordinates are normalized between $-1\ldots1$.

**Street view dataset.** We define a sparse localization task using the Street view dataset, StreetLearn [16]. The dataset contains panorama images taken in Manhattan, New York City. The dataset is split into a training set with 38,746 data points west from longitude $-73.9857$, and a test set with 16,359 data points on the east. We randomly sample areas from the dataset for sparse localization, each area of size $40m \times 40m$ and with at least 10 image-pose pairs. Within an area we sample contexts and query trajectories for training and testing. Panorama images are resized to $104\times208\times3$. Pose coordinates are normalized between $-1\ldots1$.

Sparse localization is challenging, because the environment can differ significantly at the time of mapping and localization. Data points from an area typically belong to either a single vehicle trajectory, or trajectories collected months or years apart. The environment is dynamic in both cases: moving vehicles and pedestrians in the former case; changed season, weather conditions, and even newly built roads and buildings in the latter case. Street view based data has been used, *e.g.*, for similarity-based localization [30], and generic feature extraction [29]; however, to the best of our knowledge, we are the first to use the StreetLearn data for localization and mapping with sparse information.

### C.2 Alternative methods

**Image latent map.** The image map aims to capture the spatial environment in latent features prior to querying the map, in contrast to view-embedding maps that retains spatial structure and can be queried from an egocentric coordinate frame. The image map is obtained by extracting features from context image-pose pairs with a CNN and taking the mean of the latent features over contexts. For a fair comparison, we use the same CNN structure as in DMN for obtaining view embedding features. Context poses enter the feature extractor similarly to how context poses enter the view-embedding key extractor. In the observation model the latent image map is queried similarly to DMN but without spatial attention. The map features are simply concatenated with query features and passed to the convolutional–dense network structure to estimate the particle likelihood. Unlike in egocentric spatial attention, context poses are represented in their original global coordinate frame, and the candidate pose is concatenated with the query image when extracting query features. Because of the same network structures, the number of learnable parameters is approximately the same as for the view-based map. The image map can be also thought of as a 2D grid where each cell holds a learned latent vector.

**Vector latent map.** The latent vector map is similar to the latent image map, but it uses a flat vector instead of an image, again similar to [4]. Map features are obtained by a convolutional–dense network with 3 convolutional and a dense layer. We choose hidden unit sizes such that the number of learnable parameters is comparable to the view-based map.

**Regression.** We compare DMN to a regression network that directly regresses to a pose instead of using particle filter localization. The regression network has a similar structure to a DMN with a single particle, but the observation model directly regresses to a pose instead of estimating the log-likelihood of a particle. More specifically, the network has 3 output units that define an estimate of the $x$, $y$, $\psi$ coordinates of the query pose, interpreted relative to the mean initial belief. The mean initial beliefs enters the regression network similarly to how a candidate pose enters the DMN observation model: for view-based maps the mean initial belief is used for egocentric spatial attention; for unstructured maps its coordinates are input to the query feature extractor. Because of the similar structures the regression model has approximately the same number of learnable parameters as the DMN. When used in a sequence, the pose output at $t$ becomes the mean initial belief input at $t + 1$.

**Closest context.** We report evaluation metrics the context pose closest to the ground-truth target query pose. The closest context provides an upper bound on the performance of any image similarity-based approach that would compare the query and context images.

**Uninformed estimate.** We include an uninformed estimate to help calibrating the task difficulty. The uninformed estimate ignores contexts and query images, and simply outputs the mean belief, sequentially updated with ego-motion inputs.

### C.3 Uncertainty settings

**Global localization.** The initial belief is uniform over the entire state space of the environment. That is, in the global localization mode, the agent has no information of where it can be in the environment.

**Tracking.** The initial belief is a local Gaussian distribution around the true initial pose with $\sigma_x = 0.3$, $\sigma_y = 0.3$ and $\sigma_\psi = 30°$.

### C.4 Training and evaluation

We train all DMN models and alternatives in the single step tracking setting, with $N_c = 4$ contexts. PF approaches use $K = 32$ particles. The models are implemented in Tensorflow [1] and trained with the Adam optimizer [15]. We used early stopping with patience, i.e., we terminated training if the validation error did not improve for 100k training iterations. Validation error is computed every 20k iterations using 1000 samples. We applied weight decay only for the Street view dataset, using a scaler of $10^{-4}$. We performed a simple grid search over learning rates independently for each approach and dataset pair, and chose the best performing parameter based on validation error.

We evaluate each trained model for one-step global localization ($t = 1$) and sequential global localization ($t = 5$). All reported results are mean evaluation metrics for 3 models trained with different seeds. We omitted standard errors which were small and would not have affected our conclusions.