# Towards Accurate Active Camera Localization

Qihang Fang[1*] Yingda Yin[2*] Qingnan Fan[3] Fei Xia[4] Siyan Dong[1] Sheng Wang[5]
Jue Wang[3] Leonidas Guibas[4] Baoquan Chen[2]
[1]Shandong University [2]Peking University [3]Tencent AI Lab [4]Stanford University [5]3vjia
{qihfang, fqnchina, siyandong.3, arphid}@gmail.com {yingda.yin, baoquan}@pku.edu.cn
feixia@stanford.edu wangsh@3vjia.com guibas@cs.stanford.edu

## Abstract

*In this work, we solve the problem of active camera localization, which controls the camera movements actively to achieve an accurate camera pose. The past solutions are mostly based on Markov Localization, which reduces the position-wise camera uncertainty for localization. These approaches localize the camera in the discrete pose space and are agnostic to the localization-driven scene property, which restrict the camera pose accuracy in the coarse scale. We propose to overcome these limitations via a novel active camera localization algorithm, composed of a passive and an active localization module. The former one optimizes the camera pose in the continuous pose space by establishing the point-wise camera-world correspondences. The latter one explicitly models the scene and camera uncertainty components to plan the right path for accurate camera pose estimation. We validate our algorithm on the challenging localization scenarios from both synthetic and scanned real-world indoor scenes. Experimental results demonstrate that our algorithm outperforms both the state-of-the-art Markov Localization based approach and other compared approaches on the fine-scale camera pose accuracy.*

## 1. Introduction

The problem of camera localization is to estimate the accurate camera pose in a known environment. Such a problem is of great importance in many computer vision and robotics applications. The research efforts in the past decades have been mostly devoted to camera localization in a passive manner [3, 4, 9, 10, 27, 37, 42, 45], which predicts the camera pose from the provided RGB/RGB-D frame. However, the passive localization approaches become unstable and fragile when they run into many well-known localization challenges, such as repetitive objects [21] and textureless regions [7].

To resolve the aforementioned issues, the ability of active camera movement has been deployed in a set of works [13, 19, 22, 26], also known as *active camera localization*. Three critical questions need to be answered to solve such a problem: 1) How to locate: how to localize the camera for the most accurate camera pose. 2) Where to go: the camera is initialized at an unknown position in the environment, where it should move for accurate active localization. As there are numerous localizable positions in the continuous camera pose space, the problem of active localization becomes highly ambiguous and difficult to solve. 3) When to stop: the agent is unconscious of its ground truth camera pose, hence when it should decide to stop the camera movement.

Due to the difficulties raised by these questions, there has been very little research in this field. Most active localization works are inspired by Markov Localization [8], a passive localization approach that takes random actions to reduce camera uncertainty within a 2D discrete belief map by Bayesian filtering. To decide camera movements, the early research of active localization [19] handcrafts greedy heuristics to minimize the camera uncertainty in the coming step, while the recent work [13] deploys a policy network to directly estimate the camera movement for higher localization accuracy via reinforcement learning. These approaches have dominated the active localization field in the past few decades. However, they still suffer from a few drawbacks that make them prohibitive for practical applications: 1) *Camera localization in the coarse-scale discrete pose space*. The localization accuracy relies on the predefined resolution of the 2D discrete belief map (40cm, 90° [13]), which is usually unsatisfactory for many practical applications. Pursuing fine-scale accuracy (5cm, 5°) would result in significantly increased state space, which is both memory and computation inefficient, and not scalable to large environments and continuous camera pose space. 2) *Camera movement agnostic to localization-driven scene uncertainty*. The past approaches control the actions mainly based on the camera uncertainty, without considering the localization-driven scene uncertainty information much. Scene uncertainty is an intrinsic scene

[*] Equal Contribution

property, which is small for geometry- and texture- rich regions and large for repetitive and textureless regions (common localization challenges). Scene uncertainty serves as the important guidance for camera movements towards the localizable scene region, and ignorance of such information limits the localization accuracy.

To overcome the limitations exhibited in the existing approaches, we propose a novel active camera localization algorithm solved by reinforcement learning for accurate camera localization. Our algorithm consists of two functional modules, the *passive localization* module and the *active localization* module. The former passive module answers the "How to locate" question, and estimates the step-wise camera pose in the entire episode. It abandons localization in the discrete pose space, instead learns to predict the world coordinates from the single RGB-D frame, and optimizes the instant camera pose in the continuous pose space via the established camera-world coordinate correspondences. The latter active module consists of the scene uncertainty and camera uncertainty components that answer the "Where to go" and "When to stop" questions separately. The scene uncertainty component explicitly models the localization-driven scene properties and instant localization estimations in the scene, hence it aims to guide the camera movement towards the localizable region. The camera uncertainty component explicitly models the quality of camera pose estimations, and determines the adaptive stop condition for the camera movement.

We validate our algorithm on both the synthetic and scanned real-world indoor scenes. Experimental results demonstrate that our proposed algorithm is able to achieve very high fine-scale camera pose accuracy (5cm, 5°) compared to the Markov Localization based approach and other baselines. Benefited from the proposed scene uncertainty and camera uncertainty components, our algorithm learns various intelligent behaviors.

## 2. Related Work

The past camera localization approaches are mostly passive. They can be separated into two categories, which mainly differ in the input that comes from a single frame or a sequence of frames.

For single-frame camera localization, one trend focuses on direct camera pose estimations. The early works explore various image features to retrieve the most similar database image for the pose approximation of a reference image. The traditional retrieval-based approaches mostly rely on handcrafted features [34], which are replaced with the recent deep learning features [1,33,38]. Besides image retrieval, a different popular solution is to learn a deep neural network to directly regress the camera pose [6,23,24,43]. The other trend for camera localization is indirect pose estimation that employs a two-step procedure, where the first step is to regress the 3D scene coordinates from the input RGB/RGB-D observation, and the second step takes a RANSAC based optimization to produce the final camera pose. The popular scene coordinate regression approaches are implemented as a decision tree [9,10,27,37,42] or a convolution neural network [3,4,45]. These approaches builds structure-based knowledge in a more explicit way, and performs better than image retrieval on small- or middle- scale environments.

For temporal camera localization, one trend focuses on extending PoseNet to the time domain [14,30,40,44], whose performance is however limited by the image retrieval nature of PoseNet, as pointed out by [35]. The other more popular trend assumes a uniform belief of the current camera pose, and leverages Bayesian filtering to iteratively maximize the belief until a certain stop condition is reached. According to the representations of the belief, these approaches can be separated into Kalman Filter [15,32,47], Markov Localization [18,20] and Monte-Carlo localization [16,39]. Most active localization approaches are developed based on Markov Localization, which characterizes the belief as a 2D discrete map grid and the belief is maximized when the camera randomly navigates in the environment. However, Markov Localization suffers from expensive computation due to the huge state space for step-wise comparison.

The pioneering work in active localization is active Markov Localization [8], which adopts the greedy strategy for action selection to reduce the camera pose uncertainty. This work inspires a few followups [22,26]. However, as the problem of active localization is highly ambiguous, the traditional approaches mostly fall into the shortsighted solutions. Thanks to the rapid development of reinforcement learning, active neural localization (ANL) [13] firstly learns a policy model to seek a more accurate solution from visual observations. All the above approaches benefit from Markov Localization, yet also suffer from the limited discrete camera pose space and ignorance of scene-specific localization knowledge, as discussed in the Introduction session.

## 3. Approach

### 3.1. Task Setup

Initializing the camera at an unknown position and orientation in an environment, the problem of *active camera localization* is to control the camera movement actively towards a better place to obtain an accurate camera pose. Such a task provides us with two inputs. 1) A sequence of RGB-D frames along with the corresponding ground truth camera poses, denoted as $\{I_{\text{basis}}^{(t)}, C_{\text{basis}}^{(t)}\}_{t=1}^{m}$, where $m$ is the number of frames, following previous works [9,10,27,37,42]. Such a posed RGB-D stream can be easily obtained by the SLAM system [28] with visual odometry and loop closure and roughly covers the scene. It provides the basis for both passive and active localization. 2) The instant RGB-D frame

$I^{(t)}$ obtained during active localization.

The entire procedure of our framework is as follows. With the initial RGB-D frame $I^{(0)}$, the passive localization module estimates the current camera pose $\hat{C}^{(0)}$, and the active localization module estimates the next action for camera movement and then obtain a new RGB-D frame. Such a process is iterated until the active localization module decides to stop the movement, and the final camera pose is chosen as the estimated camera pose at the last step. The entire framework is shown in Figure 1, and elaborated below.

## 3.2. Passive Localization Module

The passive localization module answers the "How to locate" question. Instead of localizing the camera in the discrete pose space within a grid-based map as previous approaches [13, 19], we propose to optimize the camera pose in the continuous pose space through a passive localizer. We adopt the state-of-the-art approach, decision tree [10], to achieve this purpose thanks to its online adaption ability in novel scenes. We briefly describe it below[1].

A decision tree, denoted as $DT$, takes a 2D image pixel $I_j^{(t)}$ sampled from the captured RGB-D frame $I^{(t)}$ as input, and performs hierarchical routing to estimate the index of one leaf node $DT(i)$, which consists of a set of 3D scene points $\{P_{dt,k}\}_{k\in\Omega_{dt,DT(i)}}$, where $\Omega_{dt,DT(i)}$ is the index set of 3D points belonging to the leaf node $DT(i)$ and $P_{dt,k}$ is back-projected in the world space with the posed RGB-D stream $\{I^{(t)}_{\text{basis}}, C^{(t)}_{\text{basis}}\}_{t=1}^{m}$. Then it randomly samples a 3D scene point from the distribution fitted from $\{P_{dt,k}\}_{k\in\Omega_{dt,DT(i)}}$ to establish the 2D-3D correspondence between the camera and world space. With correspondences obtained for many such input pixels, it infers the ranked camera pose hypotheses via pose optimization over the correspondences, and determines the camera pose $\hat{C}^{(t)}$ for the input frame $I^{(t)}$ by iteratively discarding the worse pose hypotheses until the last one left. The parameters of decision tree lie in the split node determining the routing strategy. They are pre-trained on the 7-Scenes dataset [37] and require no further finetuning. In the novel scene, only the leaf nodes are adaptively refilled online with the posed RGB-D stream[2]. The 3D scene model $D_{scene}$ is further constructed by fusing the posed RGB-D stream and the basis to generate the camera and scene uncertainty component for the active localization module.

## 3.3. Active Localization Module

In the vast literature of passive camera localization, two important factors have been studied widely for accurate localization. The first is *camera uncertainty*, which indicates the confidence of camera pose estimations, and determines which camera pose to keep for localization [5, 10, 37]. The second is *scene uncertainty*, which refers to the effectiveness of each scene region for accurate localization. For example, the passive localization approaches are able to achieve almost 100% camera pose accuracy (5cm, 5°) in scenes with small uncertainties, such as the texture- and geometry- rich scenes [37, 41], yet underperform when there exhibit the scene regions with large uncertainties, such as textureless regions and repetitive objects [7], which are all the common localization challenges. We consider that both camera uncertainty and scene uncertainty are also necessary for accurate active localization, while the focus of most active localization works lies in the camera uncertainty. Our active localization module consists of the scene uncertainty and camera uncertainty components, which answer the "Where to go" and "When to stop" questions separately.

### 3.3.1 Scene Uncertainty Component

Scene uncertainty is an intrinsic localization-driven scene property, and we describe such property from two perspectives, where the camera is located and what underlying part of the scene is observed are more effective for accurate localization. To model the above information, we propose the camera-driven scene map and world-driven scene map. They answer the "Where to go" question, and guide the camera movement towards scene regions with smaller uncertainties by combining the scene uncertainty property and the estimated camera properties (pose/world coordinate). The scene uncertainty property is purely determined by the scene model $D_{scene}$ and the passive localization module, hence pre-computed and invariant to the active localization process, while the estimated camera properties are instantly computed from the captured RGB-D frame during the camera movements.

**Camera-driven scene map:** The camera-driven scene map $M_{cd}^{(t)}$ at time step $t$ is represented in the form of the 2D top-view orthographic projection of the 3D scene model $D_{scene}$, and visualized in Figure 2. It consists of three components, position-wise uncertainty value $U_{cd}$, camera pose estimations of the current and history frames $F_{cd\_c}^{(t)}, F_{cd\_h}^{(t)}$. The scene map $M_{cd}^{(t)}$ is computed as the position-wise concatenation of the three components and thus of size $X \times Y \times 3$, where $X, Y$ are the map size,

$$M_{cd}^{(t)} = \text{Concat}\{U_{cd}, F_{cd\_c}^{(t)}, F_{cd\_h}^{(t)}\} \tag{1}$$

To filter out the invalid camera positions, we initialize all the map channels as the binary traversable map where the traversable and obstacle positions are filled with $0$ and $-1$ separately, and only update the values at traversable positions.

[1] Note we do not consider the implementation of passive localizer as our technical contribution, yet focus on how to make the best use of it for the entire task.

[2] Please refer to [10] for more implementation details of the decision tree.

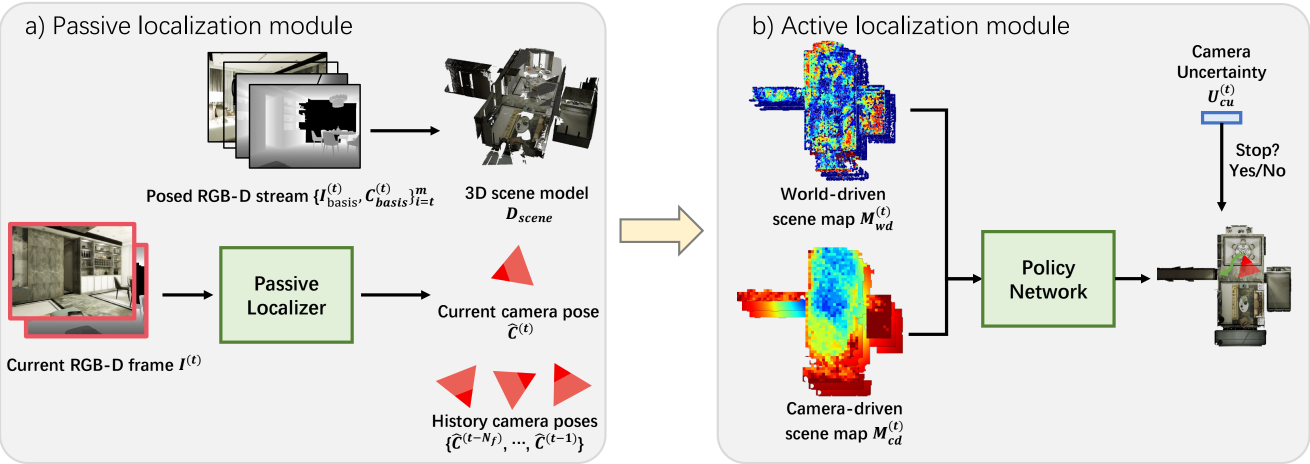
a) Passive localization module
Posed RGB-D stream $\{I_{\text{basis}}^{(t)}, C_{basis}^{(t)}\}_{i=t}^{m}$
3D scene model $D_{scene}$
Current RGB-D frame $I^{(t)}$
Passive Localizer
Current camera pose $\hat{C}^{(t)}$
History camera poses $\{\hat{C}^{(t-N_f)}, \cdots, \hat{C}^{(t-1)}\}$
b) Active localization module
World-driven scene map $M_{wd}^{(t)}$
Camera-driven scene map $M_{cd}^{(t)}$
Camera Uncertainty $U_{cu}^{(t)}$
Stop? Yes/No
Policy Network

Figure 1. The full pipeline of our algorithm. a) Given the current RGB-D frame, the passive localizer estimates its camera pose, then b) the policy network takes the scene uncertainty component (scene map) to estimate the next action for camera movement, and the camera uncertainty component determines when to stop the movement. The 3D scene model is fused from the posed RGB-D stream, and further combined with the estimated current and history camera poses to construct the camera and scene uncertainty components.

The uncertainty channel $U_{cd}$ describes the probability of successful passive localization at each valid camera position in the scene map. To be specific, for each valid camera position, we render RGB-D frames along $N_{cd}$ uniformly sampled camera directions with the scene model $D_{scene}$, and estimate the corresponding camera poses via the passive localization module. The position-wise uncertainty value $U_{cd,i}$ is inversely proportional to the camera pose accuracy (within $\lambda_{cd}$ cm, $\lambda_{cd}$ degrees) averaged over all the rendered RGB-D frames,

$$U_{cd,i} = 1 - \frac{1}{N_{cd}} \sum_{j \in [1, N_{cd}]} A^{(j)} \tag{2}$$

where $A^{(j)}$ is the binary camera pose accuracy for the $j$th frame.

The current camera pose estimation channel $F_{cd\_c}^{(t)}$ indicates where the camera is located in the scene map estimated from the current RGB-D frame $I^{(t)}$. As the camera pose is estimated in the orientation-aware continuous space, and not compatible with the orientation-agnostic discrete scene map, to minimize this gap, we simply discretize the camera pose and project it onto the 2D scene map by only considering its translation on the horizontal plane. However, the estimated camera pose formulated in this way is nothing but a single point shown in the scene map, and tends to be overwhelmed by its blank neighborhood via the common convolution operations. To highlight the importance of the camera pose information in the 2D map, we draw a distance map centered on the discretized camera position via distance transform [2] as $F_{cd\_c}^{(t)}$. For the history camera pose estimation channel, we obtain the estimated camera positions in the 2D scene map for the last $N_f$ frames $(I^{(t-N_f)}, ..., I^{(t-1)})$ same as the current channel, and draw a distance map centered on the history camera positions via distance transform as $F_{cd\_h}^{(t)}$.

**World-driven scene map:** The world-driven scene map $M_{wd}^{(t)}$ at time step $t$ is represented in the form of the 3D point cloud sampled from the scene model $D_{scene}$, and visualized in Figure 2 from the top view for better comparison with the camera-driven scene map. It consists of four components, the $x$, $y$, $z$ world coordinates of the scene points $P_{wd}$, point-wise uncertainty value $U_{wd}$, world coordinate estimations of the current and history frames $F_{wd\_c}^{(t)}$, $F_{wd\_h}^{(t)}$. The scene map $M_{wd}^{(t)}$ is computed as the point-wise concatenation of the four components and thus of size $N_{wd\_p} \times 6$ (with $N_{wd\_p}$ points and 6 channels),

$$M_{wd}^{(t)} = \text{Concat}\{P_{wd}, U_{wd}, F_{wd\_c}^{(t)}, F_{wd\_h}^{(t)}\} \tag{3}$$

The uncertainty channel $U_{wd}$ describes the effectiveness of each observable scene point to the successful passive localization, and the point-wise uncertainty value is highly related to the viewpoint where the scene point is observed. To compute the uncertainty value, we first render $N_{wd\_r}$ RGB-D frames that are randomly positioned and oriented within the traversable region. We associate each 3D scene point $P_{wd,i}$ with an index set of 2D image pixels $\Omega_{wd,i}$ that can be back-projected to it as follows,

$$\Omega_{wd,i} = \{j | \forall j \in \Omega_{wd\_r}, \|P_{wd\_r,j} - P_{wd,i}\| < \lambda_{wd}\} \tag{4}$$

where $\Omega_{wd\_r}$ is the index set of all the image pixels in the $N_{wd\_r}$ rendered frames, $P_{wd\_r,j}$ is the 3D point in the world space back-projected from the pixel $j$ in $\Omega_{wd\_r}$, and $\lambda_{wd}$ is a threshold and measured in centimeters.

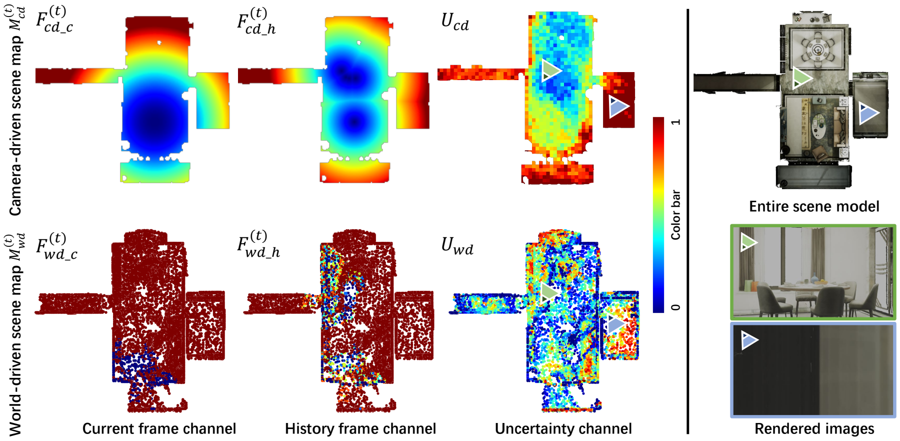

Figure 2. Left: visualization of the different channels in both the camera-driven and world-driven scene maps. The value range of $F_{cd\_c}^{(t)}$ and $F_{cd\_h}^{(t)}$ is scaled into $[0, 1]$ for better visualization with the color bar. Right: we render two first-view images with rich (green camera) or poor (blue camera) geometry and texture details, which are consistent with the uncertainty values shown in $U_{cd}$ and $U_{wd}$.

Then for each 2D pixel, we evaluate its uncertainty value $U_{wd\_r,j}$ as the estimation quality of the passive localizer, which in our case is the routing quality of the decision tree and adapted from the common measurement for the camera pose evaluation [9, 37]. To be specific, $U_{wd\_r,j}$ is computed as a binary value that judges if its back-projected 3D point $P_{wd\_r,j}$ is close to any 3D point in its routed leaf node of the decision tree,

$$U_{wd\_r,j} = \begin{cases} 0 & (\min_{k\in\Omega_{dt,DT(j)}} \|P_{wd\_r,j} - P_{dt,k}\|) < \lambda_{wd} \\ 1 & \text{otherwise} \end{cases} \tag{5}$$

where $\Omega_{dt,DT(j)}$ is the index set of the 3D points $P_{dt,k}$ in the leaf node $DT(j)$ where the pixel $j$ is routed. Then the uncertainty value of each 3D scene point $U_{wd,i}$ is averaged over the ones of its associated 2D pixels,

$$U_{wd,i} = \frac{1}{N_{wd,i}} \sum_{j\in\Omega_{wd,i}} U_{wd\_r,j} \tag{6}$$

where $N_{wd,i}$ is the size of the index set $\Omega_{wd,i}$.

The current world coordinate estimation channel indicates where the world coordinates back-projected from the current RGB-D frame using the estimated camera pose are located on the scene point cloud, hence is computed as the point-wise binary value that describes if each scene point is occupied by at least one back-projected world coordinates. To be specific, for each scene point $P_{wd,i}$, its binary value $F_{wd\_c,i}^{(t)}$ is outputted by an indicator function based on the unidirectional Chamfer distance from the estimated world coordinates to the scene point,

$$F_{wd\_c,i}^{(t)} = \begin{cases} 0 & (\min_{l\in\Omega_f^{(t)}} \|P_{wd,i} - P_{f,l}^{(t)}\|_2^2) < \lambda_{wd} \\ 1 & \text{otherwise} \end{cases} \tag{7}$$

where $\Omega_f^{(t)}$ is the index set of 3D points $P_{f,l}^{(t)}$ back-projected from the current frame $I^{(t)}$ with the estimated camera pose $\hat{C}^{(t)}$.

The history world coordinate estimation channel is simply averaged over the last $N_f$ frames. Specifically, $F_{wd\_h,i}^{(t)}$ is computed as,

$$F_{wd\_h,i}^{(t)} = \frac{1}{N_f} \sum_{t'\in[1,N_f]} F_{wd\_c,i}^{(t-t')} \tag{8}$$

**Analysis of scene uncertainty:** We visualize the computed uncertainty channel in both the camera-driven and world-driven scene maps in Figure 2. The uncertainty value denotes how much the valid camera positions and observable scene points are uncertain to successful camera localization. For better understanding of the computed uncertainty values, we also render two first-view images with the green and blue cameras separately in the scene. The blue camera captures an image with poor texture and geometry, which is a common localization challenge, correspondingly, its camera position and observed scene points in the uncertainty channel all contain very large uncertainties. On the other hand, The green camera observes an image with rich texture and geometry, which is usually easy for accurate localization, correspondingly, its camera position and observed scene points mostly contain small uncertainties. The above observation further validates the design of the proposed scene uncertainty component.

### 3.3.2 Camera Uncertainty Component

Camera uncertainty is an intrinsic camera property, which represents the quality of the current camera pose estimation during camera movements. The camera uncertainty component answers the "When to stop" question, and hence

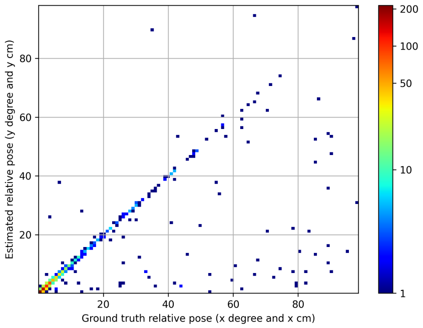

Figure 3. Justification of the camera uncertainty component. The color bar indicates the sample number.

determines the adaptive stop condition for active movements. Ideally, the camera uncertainty value should be computed by directly comparing the estimated camera pose with the ground truth camera pose, which is however absent during active movements. To alleviate the above difficulty, instead of directly dealing with the camera pose, we propose to calculate the camera uncertainty value by comparing the captured depth observation that represents the ground truth camera pose and the depth image projected from the 3D scene model $D_{scene}$ with the estimated camera pose $\hat{C}^{(t)}$. To be specific, given the observed depth and projected depth images, we first back-project the two images into the point clouds in the camera space with the known camera intrinsic parameters. Then we leverage the recent iterative closest point (ICP) approach [29] to register the two point clouds and estimate the relative camera pose between them. When the two point clouds are roughly aligned, the adopted ICP approach is able to achieve very tight point cloud alignment. Therefore, the estimated relative pose indicates how far the current camera pose estimation $\hat{C}^{(t)}$ is to the ground truth, and is treated as the camera uncertainty component $U_{cu}^{(t)} \in \mathbb{R}^2$.

To ease policy learning, many previous works fix the episode length [8, 13, 22] for camera movements, which is inefficient in implementation. In this work, we propose to adaptively stop the camera movement based on the proposed camera uncertainty component. To be specific, we consider a successful localization to stop the camera movement when the camera uncertainty component is within $\lambda_{cu}$ cm, $\lambda_{cu}$ degrees.

**Analysis of camera uncertainty:** To justify the effectiveness of the camera uncertainty component, we evaluate how close the estimated relative pose is to the ground truth in Figure 3, which contains 4500 samples randomly collected in the indoor scenes introduced in Section 4.1. We can observe that most samples lie on the diagonal lines, which means the relative pose estimations are accurate in general. To be specific, when the estimated relative poses are within 5cm, 5° (2362 samples), most samples (94.14% = 2362/2509) are truely within 5cm, 5° compared to the ground truth (2509 samples). It means the adaptive stop condition judged by the camera uncertainty component is trustworthy.

### 3.3.3 Reinforcement Learning Formulation

We optimize the policy with the state-of-the-art off-policy learning method Proximal Policy Optimization (PPO) [36] by maximizing the accumulated reward in the entire episode. The policy network is detailed in the supplementary material.

**Reward function:** We design the reward $\mathcal{R}$, consisting of a slack reward $\mathcal{R}_s$ and an exploration reward $\mathcal{R}_e$. The slack reward punishes unnecessary steps and is defined as $\mathcal{R}_s = -0.1$, which gives a negative reward for every action performed. The exploration reward $\mathcal{R}_e$ awards the agent for visiting the unseen cells to avoid repeated traversal among the same region following [31, 46]. To achieve this, we maintain a 2D occupancy map with the same map size as the camera-driven scene map, and each cell is filled with the visit count from the episode initialization. Then $\mathcal{R}_e = 0.1/v$, where $v$ is the visit count in the current occupied cell, whose position is obtained from the ground truth as the reward is only employed during training. The final reward is the summation of both rewards, $\mathcal{R} = \mathcal{R}_s + \mathcal{R}_e$.

**Policy input:** The input of the policy should encode the knowledge of the sensor input and the scene, and have positive guidance for the agents to move towards more localizable regions acknowledged by the passive localization module. In order to achieve this goal, the policy takes the scene uncertainty component at time step $t$ as input $\{M_{cd}^{(t)}, M_{wd}^{(t)}\}$.

**Action space:** Following the previous active localization setting [13, 19], we assume that the agent (camera) moves with the 3-DoF (Degree of Freedom) action space within the 1-meter high 2D plane parallel to the ground. The agent is capable of performing three actions, *move forward*, *turn left* and *turn right*. The agent moves forward by 20cm, and turns left/right by 30°. We further disturb the actions with Gaussian noises as introduced in the supplementary material.

## 4. Experiments

### 4.1. Experimental Setup

**Data processing:** We evaluate our algorithm on both the synthetic and scanned real-world indoor scenes. To alleviate the difficulty of creating the common localization challenges in the synthetic data, we collect 35 high-quality indoor scenes of average area $40.91m^2$, that feature textureless walls, repetitive pillows/drawings, *etc*, by design, and provide a train/test split of the scenes (train/test: 15/20 scenes). To prepare for the scanned real-world data, we collect 5 indoor scenes of average area $64.82m^2$ from the public Matterpot3D dataset [11] only for evaluation. For each indoor scene, we provide a list of data as follows:

- A sequence of <RGB-D image, camera pose> pairs $\{I_{\text{basis}}^{(t)}, C_{\text{basis}}^{(t)}\}_{t=1}^{m}$ that provides the basis for localization

| | ACL-synthetic | | ACL-real | |
|---|---|---|---|---|
| Methods | Accuracy (%) | #steps | Accuracy (%) | #steps |
| ANL [13] | 1.26 | 100 | 0.98 | 100 |
| No-movement (DecisionTree) | 9.35 | 0 | 6.80 | 0 |
| No-movement (DSAC) | 14.90 | 0 | 7.80 | 0 |
| Turn-around | 25.00 | 12 | 35.20 | 12 |
| Camera-descent (t+1) | 61.55 | 22.90 | 61.40 | 26.85 |
| Camera-descent (t+2) | 55.30 | 22.60 | 59.20 | 25.78 |
| Scene-descent | 57.65 | 18.56 | 54.20 | 16.87 |
| Ours (w/o $\mathcal{R}_e \& M_{cd}^{(t)}$) | 67.65 | 17.40 | 70.60 | 19.71 |
| Ours (w/o $\mathcal{R}_e \& M_{wd}^{(t)}$) | 66.40 | 16.27 | 67.40 | 18.63 |
| Ours (w/o $\mathcal{R}_e$) | 72.50 | 18.57 | 73.00 | 20.72 |
| Ours | **83.05** | 17.33 | **82.40** | 17.90 |

Table 1. Numerical results on the synthetic and scanned real-world indoor scenes.

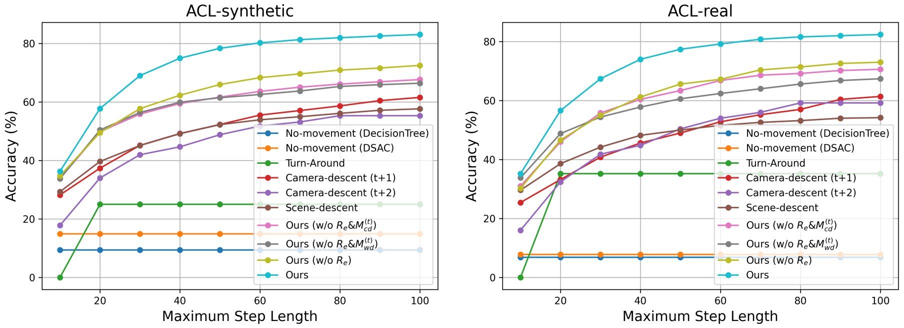

Figure 4. Plot of the localization accuracy that varies with different maximum step lengths.

and roughly covers the scene.

- Instant RGB-D frame $I^{(t)}$ obtained during active localization.
- 100 test images in each test scene. They are randomly sampled in the scene region of large uncertainties to increase the localization difficulty (1 meter away from the positions of $U_{cd,i} \leq 0.5$).

We name the synthetic dataset ACL-synthetic, and the real-world dataset ACL-real. Our algorithm is trained only on the train split of the ACL-synthetic dataset, and evaluated on both the test split of the ACL-synthetic dataset and the entire ACL-real dataset. During training, the camera is initialized randomly in the scene. During evaluation, the camera is initialized with one of the 100 test images. More details about both datasets[3] are in the supplementary material.

**Training setting:** The passive localizer is adapted online in novel scenes with the posed RGB-D stream as mentioned in Section 3.2, hence only the policy network needs to be trained in our algorithm. In our experiment, we employ the Adam [25] to optimize the network weights with the initial learning rate of $3 \times 10^{-4}$. Some hyper-parameters: $N_{cd} = 12, N_{wd\_r} = 1000, N_{wd\_p} = 2^{14} = 16384, N_f = 5, X = 256, Y = 256$. Following the popular camera pose accuracy measured by 5cm, 5° [3, 4, 9, 10, 27, 37, 42, 45], we set $\lambda_{cd} = \lambda_{wd} = \lambda_{cu} = 5$. It means we encourage the agent to move to the scene region where the camera pose estimated from the passive localization module is within 5cm, 5° to the ground truth, and stop the camera movement when it believes the estimated camera pose is within 5cm, 5° to the ground truth.

## 4.2. Compared Approaches

We detail the compared approaches below.

- **No-movement**. It only takes use of the passive localization module to estimate the camera pose for the initial test frame. We adopt two passive localizers for comparison, the default decision tree [10] (No-movement (DecisionTree)) and the popular CNN-based passive localizer [3] (No-movement (DSAC)).
- **Turn-around**. This baseline works by turning a circle along the vertical axis for 12 uniformly-sampled directions without any forward movement, and stopping at the camera pose with the smallest camera uncertainty value.
- **Camera-descent**. It iterates over all the possible actions in the future steps and selects the one with the smallest camera uncertainty value as the following path, hence it moves along the camera uncertainty descent direction. It

[3] Note we do not claim the contribution of the collected indoor scenes, which can be replaced with any ones in public indoor scene datasets.

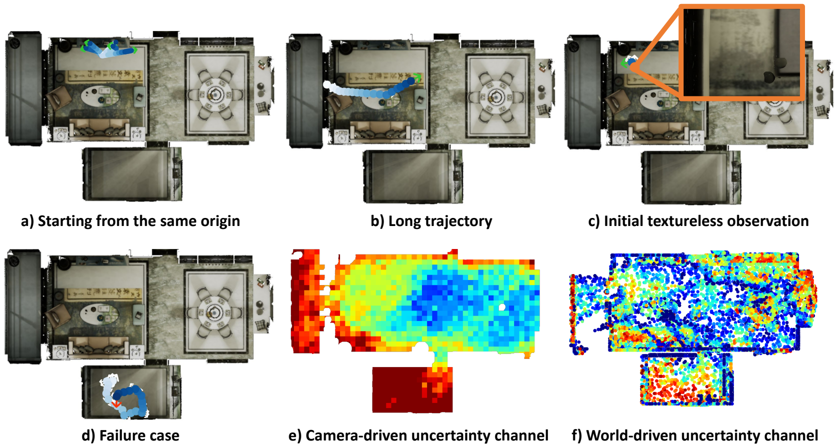
a) Starting from the same origin
b) Long trajectory
c) Initial textureless observation
d) Failure case
e) Camera-driven uncertainty channel
f) World-driven uncertainty channel

Figure 5. Qualitative results. White arrow: start position; Green arrow: end position (successfully localized); The dots with color gradient indicate the path the agent takes. Intelligent behaviors: **a)** Starting from the same location, the agent travels to various regions for localization. **b)** The agent is able to travel along a long trajectory for accurate localization. **c)** Initialized with a textureless image, the agent emerges the turn around behavior for localization. Failure case: **d)** The agent fail to get out of a small room. Uncertainty visualization: **e)** The camera-driven uncertainty channel. **f)** The world-driven uncertainty channel.

stops when it triggers our adaptive stop condition. Depending on the number of explored future steps (1/2 steps), we derive two baselines, Camera-descent (t+1/t+2). We adopt beam search to implement Camera-descent (t+2) for memory efficiency.

- **Scene-descent**. It assumes the estimated camera pose is roughly correct, and computes the shortest path from the estimated camera pose to the more localizable region ($U_{cd,i} \leq 0.5$) in the camera-driven uncertainty channel. Therefore, it moves along the scene uncertainty descent direction. It stops when it finishes the traversal over the shortest path.
- **ANL**. Active neural localization (ANL) [13] is a state-of-the-art active localization approach derived from the Markov Localization. Due to the significant requirement of memory and computation resources, its camera pose is limited at the resolution of 20cm, 90° with Nvidia Tesla V100 of 32G memory in our implementation (40cm, 90° in [13]).

### 4.3. Evaluation Metrics

The major goal of active camera localization lies in achieving higher camera pose accuracy. We evaluate the accuracy (%) as the proportion of successful localization episodes whose translation and rotation error for the final camera pose is within 5cm, 5°, a fine-scale measurement compared to 40cm, 90° adopted in ANL [13]. We further compute the number of steps (#steps) taken to finish the successful localization acknowledged by the accuracy measure. It is only a complementary metric, while we value the accuracy most.

### 4.4. Results

Starting from a bad location, it may take a very long episode to stop the camera movement for the compared approaches. To avoid such corner cases, we limit all the compared approaches with a maximum step length, which is 100 for valuation. The numerical results are shown in Table 1. For more results, please refer to the supplementary material.

**Comparison with baselines:** We analyze the results in the synthetic indoor scenes (ACL-synthetic) first. The No-movement baselines achieve upmost 14.90% accuracy, indicating the fact that passive localization is not sufficient in our challenging localization scenarios. By enabling the rotation actions, the accuracy of the Turn-around heuristic is only 25.00% at most, which suggests the importance of active camera movements. The Camera-descent and Scene-descent baselines contain smarter designs based on our proposed camera uncertainty and scene uncertainty components, and also significantly improve the accuracy. The Camera-descent baseline decides its next action by foreseeing all the possible actions in the future steps (t+1/t+2). Such a strategy costs additional time steps (back and forth traversal over all the actions), which cannot be conducted in background due to the fact that the ground truth camera pose is unknown and hence real actions need to be performed to compute the camera uncertainty. The additional cost limits the accuracy within the limited number of steps on the other hand, therefore Camera-descent (t+2) shows degradation on accuracy (55.30% *v.s.* 61.55%) compared to Camera-descent (t+1). The Scene-descent baseline performs less satisfactorily in accuracy (57.65%) due to its strong assumption that the initial camera pose estimation is roughly accurate, which could be wrong and lead the agent to a completely wrong position. Our algorithm outperforms all the approaches in the camera pose accuracy (83.05%) with limited steps being taken. Similar phenomenon can also be observed in the scanned real-world indoor scenes (ACL-real). We further visualize the accuracy that progresses along the increasing maximum step length in Figure 4, where our algorithm is consistently better than all the others.

**Comparison with ANL:** ANL is trained on the discrete belief map of resolution 20cm, 90°, which is almost the upper bound of camera pose scale it can achieve. Therefore, it performs poorly on the finer-scale accuracy (5cm, 5°) as expected. In ANL, the passive localization module is implemented as the image similarity computation, performed between the current agent observation and each memory image sampled uniformly within the complete scene. The obtained position-wise image similarity (camera uncertainty) forms a belief map, which is absorbed by the active localization module (policy network) to determine the action. In our approach, the passive localization module is implemented as the camera pose estimator for the current observation, and the scene uncertainty component is taken by the active localization module for action selection. The two approaches are theoretically different, hence it is non-trivial to deploy one in the other for further improvement.

**Ablation study:** We justify our algorithm by ablating three components, the exploration reward $\mathcal{R}_e$, camera-driven scene map $M_{cd}^{(t)}$ and world-driven scene map $M_{wd}^{(t)}$. Experimentally, we observe that our algorithm benefits from all the three components.

**Time analysis and intelligent behavior:** It takes only 9.59s to adapt the passive localizer in a novel scene, and 1.31s to evaluate our entire algorithm for a single step. Our learned intelligent behaviors are visualized in Figure 5.

## 5. Conclusion

In this paper, we propose a novel active camera localization algorithm, consisting of a passive and an active localization module. The former one estimates the accurate camera pose in the continuous pose space. The latter one learns a reinforcement learning policy from the explicitly modeled camera and scene uncertainty component for accurate camera localization.

**Limitation and future work:** Figure 5 e) demonstrates a failure case, where the agent is initialized in a room with a small exit and large scene uncertainties. It fails to leave the room before reaching the maximum step length. Although we already employ a naive exploration reward to avoid repeated traversal in the same region, a smarter design, such as frontier-based exploration [17] and long-term goal planning [12], can be incorporated in the future for further improvement.

## References

[1] Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. Netvlad: Cnn architecture for weakly supervised place recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5297–5307, 2016. 2

[2] Gunilla Borgefors. Distance transformations in digital images. *Computer vision, graphics, and image processing*, 34(3):344–371, 1986. 4

[3] Eric Brachmann, Alexander Krull, Sebastian Nowozin, Jamie Shotton, Frank Michel, Stefan Gumhold, and Carsten Rother. Dsac-differentiable ransac for camera localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6684–6692, 2017. 1, 2, 7

[4] Eric Brachmann and Carsten Rother. Learning less is more-6d camera localization via 3d surface regression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4654–4662, 2018. 1, 2, 7

[5] Eric Brachmann and Carsten Rother. Neural-guided ransac: Learning where to sample model hypotheses. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4322–4331, 2019. 3

[6] Samarth Brahmbhatt, Jinwei Gu, Kihwan Kim, James Hays, and Jan Kautz. Geometry-aware learning of maps for camera localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2616–2625, 2018. 2

[7] Mai Bui, Tolga Birdal, Haowen Deng, Shadi Albarqouni, Leonidas Guibas, Slobodan Ilic, and Nassir Navab. 6d camera relocalization in ambiguous scenes via continuous multimodal inference. In *European Conference on Computer Vision*, pages 139–157. Springer, 2020. 1, 3

[8] Anthony R Cassandra, Leslie Pack Kaelbling, and James A Kurien. Acting under uncertainty: Discrete bayesian models for mobile-robot navigation. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems. IROS'96*, volume 2, pages 963–972. IEEE, 1996. 1, 2, 6

[9] Tommaso Cavallari, Stuart Golodetz, Nicholas Lord, Julien Valentin, Victor Prisacariu, Luigi Di Stefano, and Philip HS Torr. Real-time rgb-d camera pose estimation in novel scenes using a relocalisation cascade. *IEEE transactions on pattern analysis and machine intelligence*, 2019. 1, 2, 5, 7

[10] Tommaso Cavallari, Stuart Golodetz, Nicholas A Lord, Julien Valentin, Luigi Di Stefano, and Philip HS Torr. On-the-fly adaptation of regression forests for online camera relocalisation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4457–4466, 2017. 1, 2, 3, 7

[11] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3d: Learning from rgb-d data in indoor environments. *arXiv preprint arXiv:1709.06158*, 2017. 6

[12] Devendra Singh Chaplot, Dhiraj Gandhi, Saurabh Gupta, Abhinav Gupta, and Ruslan Salakhutdinov. Learning to explore using active neural slam. In *International Conference on Learning Representations (ICLR)*, 2020. 9

[13] Devendra Singh Chaplot, Emilio Parisotto, and Ruslan Salakhutdinov. Active neural localization. In *International Conference on Learning Representations*, 2018. 1, 2, 3, 6, 7, 8

[14] Ronald Clark, Sen Wang, Andrew Markham, Niki Trigoni, and Hongkai Wen. Vidloc: A deep spatio-temporal model for 6-dof video-clip relocalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6856–6864, 2017. 2

[15] Ingemar J Cox and John J Leonard. Modeling a dynamic environment using a bayesian multiple hypothesis approach. *Artificial Intelligence*, 66(2):311–344, 1994. 2

[16] Frank Dellaert, Dieter Fox, Wolfram Burgard, and Sebastian Thrun. Monte carlo localization for mobile robots. In *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No. 99CH36288C)*, volume 2, pages 1322–1328. IEEE, 1999. 2

[17] Christian Dornhege and Alexander Kleiner. A frontier-void-based approach for autonomous exploration in 3d. *Advanced Robotics*, 27(6):459–468, 2013. 9

[18] Dieter Fox. *Markov localization-a probabilistic framework for mobile robot localization and navigation.* PhD thesis, Universität Bonn, 1998. 2

[19] Dieter Fox, Wolfram Burgard, and Sebastian Thrun. Active markov localization for mobile robots. *Robotics and Autonomous Systems*, 25(3):195–208, 1998. 1, 3, 6

[20] Dieter Fox, Wolfram Burgard, and Sebastian Thrun. Markov localization for mobile robots in dynamic environments. *Journal of artificial intelligence research*, 11:391–427, 1999. 2

[21] Maciej Halber, Yifei Shi, Kai Xu, and Thomas Funkhouser. Rescan: Inductive instance segmentation for indoor rgbd scans. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2541–2550, 2019. 1

[22] Patric Jensfelt and Steen Kristensen. Active global localization for a mobile robot using multiple hypothesis tracking. *IEEE Transactions on Robotics and Automation*, 17(5):748–760, 2001. 1, 2, 6

[23] Alex Kendall and Roberto Cipolla. Geometric loss functions for camera pose regression with deep learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5974–5983, 2017. 2

[24] Alex Kendall, Matthew Grimes, and Roberto Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *Proceedings of the IEEE international conference on computer vision*, pages 2938–2946, 2015. 2

[25] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 7

[26] Gian Luca Mariottini and Stergios I Roumeliotis. Active vision-based robot localization and navigation in a visual memory. In *2011 IEEE International Conference on Robotics and Automation*, pages 6192–6198. IEEE, 2011. 1, 2

[27] Lili Meng, Frederick Tung, James J Little, Julien Valentin, and Clarence W de Silva. Exploiting points and lines in regression forests for rgb-d camera relocalization. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6827–6834. IEEE, 2018. 1, 2, 7

[28] Raul Mur-Artal and Juan D Tardós. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE transactions on robotics*, 33(5):1255–1262, 2017. 2

[29] Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Colored point cloud registration revisited. In *Proceedings of the IEEE international conference on computer vision*, pages 143–152, 2017. 6

[30] 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. 2

[31] Santhosh K Ramakrishnan, Dinesh Jayaraman, and Kristen Grauman. An exploration of embodied visual exploration. *International Journal of Computer Vision*, 129(5):1616–1649, 2021. 6

[32] Stergios I Roumeliotis and George A Bekey. Bayesian estimation and kalman filtering: A unified framework for mobile robot localization. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, volume 3, pages 2985–2992. IEEE, 2000. 2

[33] Paul-Edouard Sarlin, Cesar Cadena, Roland Siegwart, and Marcin Dymczyk. From coarse to fine: Robust hierarchical localization at large scale. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 12716–12725, 2019. 2

[34] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Fast image-based localization using direct 2d-to-3d matching. In *2011 International Conference on Computer Vision*, pages 667–674. IEEE, 2011. 2

[35] Torsten Sattler, Qunjie Zhou, Marc Pollefeys, and Laura Leal-Taixe. Understanding the limitations of cnn-based absolute camera pose regression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3302–3312, 2019. 2

[36] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. 6

[37] Jamie Shotton, Ben Glocker, Christopher Zach, Shahram Izadi, Antonio Criminisi, and Andrew Fitzgibbon. Scene coordinate regression forests for camera relocalization in rgb-d images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2930–2937, 2013. 1, 2, 3, 5, 7

[38] Hajime Taira, Masatoshi Okutomi, Torsten Sattler, Mircea Cimpoi, Marc Pollefeys, Josef Sivic, Tomas Pajdla, and Akihiko Torii. Inloc: Indoor visual localization with dense matching and view synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7199–7209, 2018. 2

[39] Sebastian Thrun, Dieter Fox, Wolfram Burgard, and Frank Dellaert. Robust monte carlo localization for mobile robots. *Artificial intelligence*, 128(1-2):99–141, 2001. 2

[40] Abhinav Valada, Noha Radwan, and Wolfram Burgard. Deep auxiliary learning for visual localization and odometry. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 6939–6946. IEEE, 2018. 2

[41] Julien Valentin, Angela Dai, Matthias Nießner, Pushmeet Kohli, Philip Torr, Shahram Izadi, and Cem Keskin. Learning to navigate the energy landscape. In *2016 Fourth International Conference on 3D Vision (3DV)*, pages 323–332. IEEE, 2016. 3

[42] Julien Valentin, Matthias Nießner, Jamie Shotton, Andrew Fitzgibbon, Shahram Izadi, and Philip HS Torr. Exploiting uncertainty in regression forests for accurate camera relocalization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4400–4408, 2015. 1, 2, 7

[43] Bing Wang, Changhao Chen, Chris Xiaoxuan Lu, Peijun Zhao, Niki Trigoni, and Andrew Markham. Atloc: Attention guided camera localization. 2020. 2

[44] Fei Xue, Xin Wang, Zike Yan, Qiuyuan Wang, Junqiu Wang, and Hongbin Zha. Local supports global: Deep camera relocalization with sequence enhancement. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2841–2850, 2019. 2

[45] Luwei Yang, Ziqian Bai, Chengzhou Tang, Honghua Li, Yasutaka Furukawa, and Ping Tan. Sanet: Scene agnostic network for camera localization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 42–51, 2019. 1, 2, 7

[46] Joel Ye, Dhruv Batra, Abhishek Das, and Erik Wijmans. Auxiliary tasks and exploration enable objectgoal navigation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16117–16126, 2021. 6

[47] Lei Zhou, Zixin Luo, Tianwei Shen, Jiahui Zhang, Mingmin Zhen, Yao Yao, Tian Fang, and Long Quan. Kfnet: Learning temporal camera relocalization using kalman filtering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4919–4928, 2020. 2