

Towards Accurate Active Camera Localization

Qihang Fang^{1*}, Yingda Yin^{2*}, Qingnan Fan^{3†}, Fei Xia⁴, Siyan Dong¹
Sheng Wang⁵, Jue Wang³, Leonidas Guibas⁴, Baoquan Chen^{2†}

¹Shandong University, ²Peking University
³Tencent AI Lab, ⁴Stanford University, ⁵3vjia

Abstract. In this work, we tackle 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 restricts 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 optimizes the camera pose in the continuous pose space by establishing point-wise camera-world correspondences. The latter 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. Code and data are released at <https://github.com/qhFang/AccurateACL>.

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 [55, 56, 57, 28]. The research efforts in the past decades have been mostly devoted to camera localization in a passive manner [42, 30, 48, 11, 10, 4, 5, 53], 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 [23] and textureless regions [8].

To resolve the aforementioned issues, the ability of active camera movement has been deployed in a set of works [21, 24, 29, 15], 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,

* Equal contribution; ordered alphabetically.

† Corresponding authors.

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 unaware 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 [9], 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 [21] handcrafts greedy heuristics to minimize the camera uncertainty in the coming step, while the recent work [15] 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° [15]), 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 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

Passive localization. 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 by retrieving the most similar database image for the pose approximation of a reference image [38,2,37,44] or directly regressing the camera pose through neural networks [26,25,50,7]. The other trend 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 [42,30,48,11,10] or a convolution neural network [4,5,53]. These approaches build 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 [16,46,34,52], whose performance is however limited by the image retrieval nature of PoseNet, as pointed out by [39]. 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 [17,36,58], Markov Localization [20,22] and Monte-Carlo localization [45,18]. 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.

Active localization. The pioneering work in active localization is active Markov Localization [9], which adopts the greedy strategy for action selection to reduce the camera pose uncertainty. This work inspires a few followups [24,29]. However, as the problem of active localization is highly ambiguous, the traditional approaches mostly fall into shortsighted solutions. Thanks to the rapid development of reinforcement learning, active neural localization (ANL) [15] 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.

Navigation. Visual navigation [40,1,14] aims at reaching *a specific target* with the guidance of points (PointNav), images (ImageNav), semantics (ObjNav), etc. In contrast, active visual localization targets reaching an accurate camera pose. As there are *numerous localizable poses*, the problem becomes highly ambiguous and more difficult to solve [15]. On the other hand, localization is an essential module in a navigation system, where active localization algorithms could be adopted in a navigation pipeline to further improve its performance.

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 $\{I_{\text{basis}}^{(i)}, C_{\text{basis}}^{(i)}\}_{i=1}^m$, where m is the number of frames, following previous works [42,30,48,11,10]. Such a posed RGB-D stream can be easily obtained by the SLAM system [31] 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 obtains 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. We also refer readers to Algorithm 1 in the supplementary material for the entire procedure.

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 [21,15], 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 [11], to achieve this purpose thanks to its online adaption ability in novel scenes. We briefly describe it below¹.

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

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.

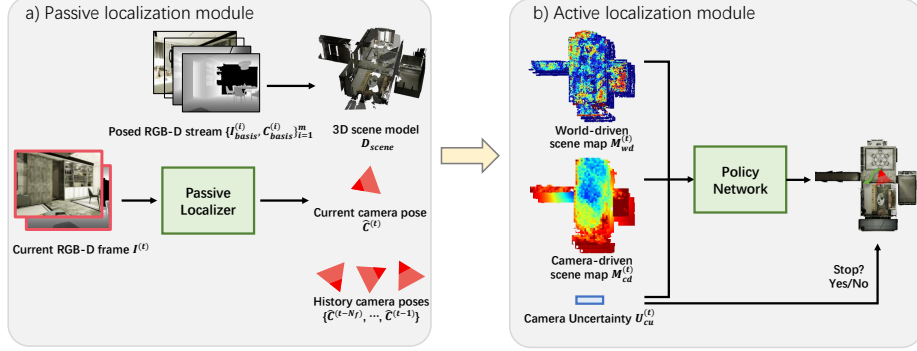


Fig. 1. The full pipeline of our algorithm. a) Given the current RGB-D frame, the passive localizer estimates its camera pose, then b) the policy network takes the scene and camera uncertainty component to estimate the next action for camera movement, and the camera uncertainty component determines when 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.

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 $\{I_{basis}^{(t)}, C_{basis}^{(t)}\}_{t=1}^m$. 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 the decision tree lie in the split node determining the routing strategy. They are pre-trained on the 7-Scenes dataset [42] and require no further finetuning. In novel scenes, only the leaf nodes are adaptively refilled online with the posed RGB-D stream². 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 [42,11,6]. The second is *scene uncertainty*, which refers to the effectiveness of each scene region for accurate localization.

² Please refer to [11] for more implementation details of the decision tree.

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 [42,47], yet underperform when there exhibit the scene regions with large uncertainties, such as textureless regions and repetitive objects [8], 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 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.

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,

$$M_{cd}^{(t)} = \text{Concat}\{U_{cd}, F_{cd_c}^{(t)}, F_{cd_h}^{(t)}\} \quad (1)$$

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.

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 λ_{cd} cm, λ_{cd} degrees) averaged over all the rendered RGB-D frames,

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

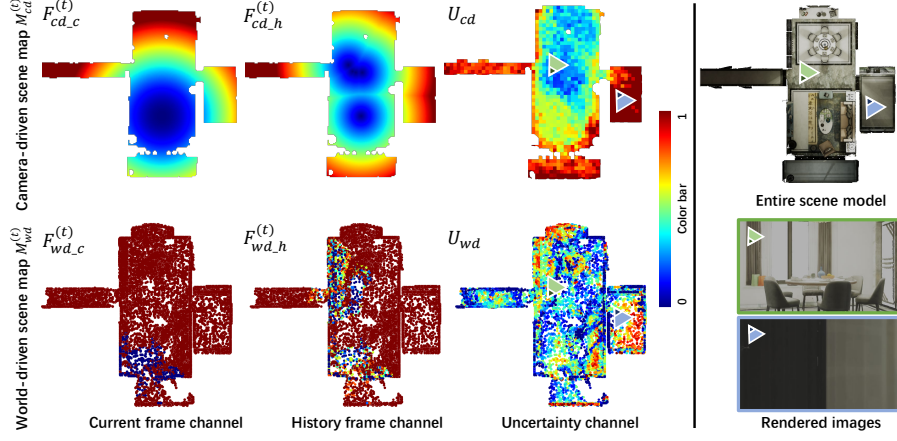


Fig. 2. Left: visualization of the different channels in both the camera-driven and world-driven scene maps. The value range of $F_{cd_c}^{(t)}$ and $F_{cd_h}^{(t)}$ is scaled into $[0, 1]$ for better visualization with the color bar. Right: we render two first-view images with rich (green camera) or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

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 [3] 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 ($I^{(t-N_f)}, \dots, I^{(t-1)}$) 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),

$$M_{wd}^{(t)} = \text{Concat}\{P_{wd}, U_{wd}, F_{wd_c}^{(t)}, F_{wd_h}^{(t)}\} \quad (3)$$

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,

$$\Omega_{wd,i} = \{j | \forall j \in \Omega_{wd_r}, \|P_{wd_r,j} - P_{wd,i}\| < \lambda_{wd}\} \quad (4)$$

where Ω_{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 Ω_{wd_r} , and λ_{wd} is a threshold and measured in centimeters.

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 [42,10]. 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,

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

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,

$$U_{wd,i} = \frac{1}{N_{wd,i}} \sum_{j \in \Omega_{wd,i}} U_{wd_r,j} \quad (6)$$

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,

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

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,

$$F_{wd_h,i}^{(t)} = \frac{1}{N_f} \sum_{t' \in [1, N_f]} F_{wd_c,i}^{(t-t')} \quad (8)$$

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 a 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.

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 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 colored point cloud registration approach (Colored ICP) [32] 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 [9, 24, 15] 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 λ_{cu} cm, λ_{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 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 truly within 5cm, 5° compared to the ground truth (2509 samples). It means the adaptive stop condition judged by the camera uncertainty component is trustworthy.

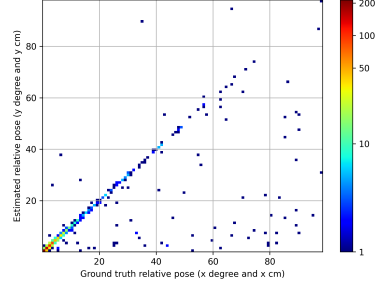


Fig. 3. Justification of the camera uncertainty component. The color bar indicates the sample number.

Reinforcement Learning Formulation
We optimize the policy with the off-policy learning method Proximal Policy Optimization (PPO) [41] 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 [54,35]. 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 currently 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 and camera uncertainty components at time step t as input $\{M_{cd}^{(t)}, M_{wd}^{(t)}, U_{cu}^{(t)}\}$.

Action space: Following the previous active localization setting [21,15], 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 with an average area of $40.9m^2$, that feature textureless walls, repetitive pillows/drawings, *etc*, by design, and provide a train/test split of the scenes (train/test: 15/20 scenes). For the scanned real-world data, we collect 5 indoor scenes with an average area of $64.8m^2$ from the public Matterpot3D dataset [12] 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 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 datasets³ 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. Following the popular camera pose accuracy measured by 5cm, 5° [42,30,48,11,10,4,5,53], 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.

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° by default, a fine-scale measurement compared to 40cm, 90° adopted in ANL [15]. 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. We limit all the approaches with a maximum step length of 100.

4.2 Compared Approaches

We detail the compared approaches below.

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.

Table 1. Numerical results evaluated with the fine-scale 5cm, 5° accuracy.

Methods	ACL-synthetic		ACL-real	
	Acc (%)	#steps	Acc (%)	#steps
ANL [15]	3.25	100	3.20	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

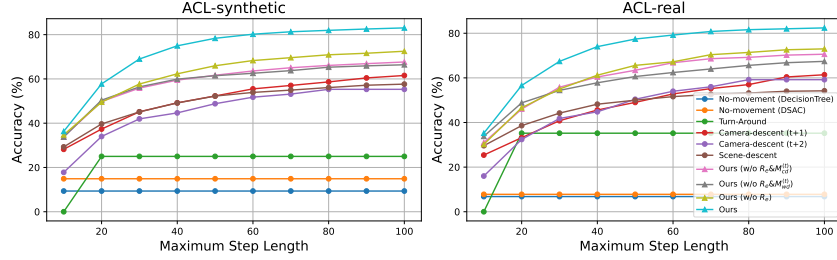
- **No-movement.** It only uses the passive localization module to estimate the camera pose for the initial test frame. We adopt two passive localizers for comparison, the default decision tree [11] (No-movement (DecisionTree)) and the popular CNN-based passive localizer [4] (No-movement (DSAC)).
- **Turn-around.** 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 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) [15] 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 [15]).

4.3 Results

Comparison with baselines: The comparison is shown in Table 1. 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

Table 2. Numerical results evaluated with the coarse-scale 20cm, 90° accuracy.

Methods	ACL-synthetic		ACL-real	
	Acc (%)	#steps	Acc (%)	#steps
Markov Loc. [20]	44.70	100	39.20	100
Active Markov Loc. [21]	44.10	100	40.00	100
ANL [15]	87.30	100	84.20	100
Ours	88.75	17.09	85.20	17.88

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

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. Our algorithm outperforms all the approaches in the camera pose accuracy (83.05%) with limited steps being taken. Similar phenomena 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 the camera pose scale it can achieve. Therefore, it performs poorly on the finer-scale accuracy (5cm, 5°) as expected. Furthermore, we evaluate both ANL and our method on the coarse-scale 20cm, 90° accuracy where ANL is good at, shown in Table 2. We can see that ANL achieves significantly better results on the coarser-scale accuracy, while our method still achieves comparable localization accuracy, with much fewer moving steps. We also compare with Markov localization [20] and Active Markov Localization [21] on the coarse-scale 20cm, 90° accuracy following [15].

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 three components.

Time analysis and intelligent behavior: It takes only 9.59s to adapt the passive localizer in a novel scene, and 0.87s to evaluate our entire algorithm for

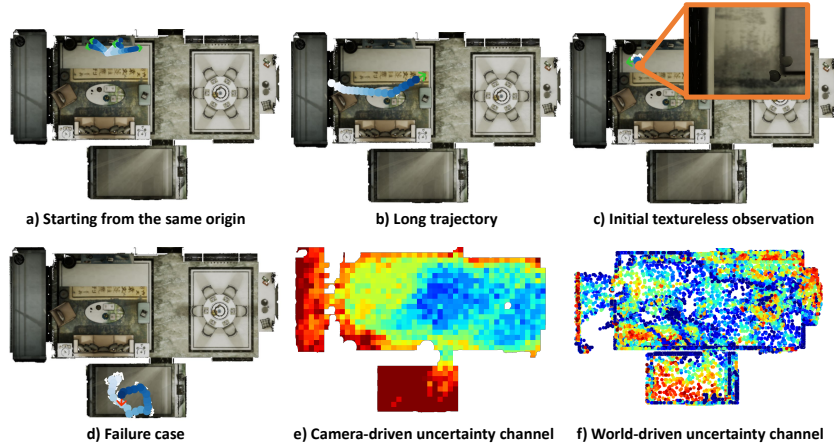


Fig. 5. Qualitative results. White arrow: start position; Green arrow: end position (successfully localized); The dots with color gradient indicate the path the agent takes. Intelligent behaviors: **a)** Starting from the same location, the agent travels to various regions for localization. **b)** The 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.

a single step, where the bottleneck comes from the CPU-based implementation of ICP [59] (0.59s), which can be further improved with more efficient GPU implementation. 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 [19] and long-term goal planning [13], can be incorporated in the future for further improvement.

Acknowledgments We thank the anonymous reviewers for the insightful feedback. This work was supported in part by NSFC Projects of International Cooperation and Exchanges (62161146002), NSF grant IIS-1763268, a Vannevar Bush Faculty Fellowship, and a gift from the Amazon Research Awards program.

References

1. Anderson, P., Chang, A., Chaplot, D.S., Dosovitskiy, A., Gupta, S., Koltun, V., Kosecka, J., Malik, J., Mottaghi, R., Savva, M., et al.: On evaluation of embodied navigation agents. arXiv preprint arXiv:1807.06757 (2018) [4](#)
2. Arandjelovic, R., Gronat, P., Torii, A., Pajdla, T., Sivic, J.: Netvlad: Cnn architecture for weakly supervised place recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 5297–5307 (2016) [3](#)
3. Borgefors, G.: Distance transformations in digital images. Computer vision, graphics, and image processing **34**(3), 344–371 (1986) [7](#)
4. Brachmann, E., Krull, A., Nowozin, S., Shotton, J., Michel, F., Gumhold, S., Rother, C.: Dsac-differentiable ransac for camera localization. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 6684–6692 (2017) [1](#), [3](#), [11](#), [12](#)
5. Brachmann, E., Rother, C.: Learning less is more-6d camera localization via 3d surface regression. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4654–4662 (2018) [1](#), [3](#), [11](#)
6. Brachmann, E., Rother, C.: Neural-guided ransac: Learning where to sample model hypotheses. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 4322–4331 (2019) [5](#)
7. Brahmbhatt, S., Gu, J., Kim, K., Hays, J., Kautz, J.: Geometry-aware learning of maps for camera localization. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2616–2625 (2018) [3](#)
8. Bui, M., Birdal, T., Deng, H., Albarqouni, S., Guibas, L., Ilic, S., Navab, N.: 6d camera relocalization in ambiguous scenes via continuous multimodal inference. In: European Conference on Computer Vision. pp. 139–157. Springer (2020) [1](#), [6](#)
9. Cassandra, A.R., Kaelbling, L.P., Kurien, J.A.: Acting under uncertainty: Discrete bayesian models for mobile-robot navigation. In: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems. IROS’96. vol. 2, pp. 963–972. IEEE (1996) [2](#), [3](#), [9](#)
10. Cavallari, T., Golodetz, S., Lord, N., Valentin, J., Prisacariu, V., Di Stefano, L., Torr, P.H.: Real-time rgb-d camera pose estimation in novel scenes using a relocalisation cascade. IEEE transactions on pattern analysis and machine intelligence (2019) [1](#), [3](#), [4](#), [8](#), [11](#)
11. Cavallari, T., Golodetz, S., Lord, N.A., Valentin, J., Di Stefano, L., Torr, P.H.: On-the-fly adaptation of regression forests for online camera relocalisation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4457–4466 (2017) [1](#), [3](#), [4](#), [5](#), [11](#), [12](#)
12. Chang, A., Dai, A., Funkhouser, T., Halber, M., Niessner, M., Savva, M., Song, S., Zeng, A., Zhang, Y.: Matterport3d: Learning from rgb-d data in indoor environments. arXiv preprint arXiv:1709.06158 (2017) [11](#)
13. Chaplot, D.S., Gandhi, D., Gupta, S., Gupta, A., Salakhutdinov, R.: Learning to explore using active neural slam. In: International Conference on Learning Representations (ICLR) (2020) [14](#)
14. Chaplot, D.S., Gandhi, D.P., Gupta, A., Salakhutdinov, R.R.: Object goal navigation using goal-oriented semantic exploration. Advances in Neural Information Processing Systems **33**, 4247–4258 (2020) [4](#)
15. Chaplot, D.S., Parisotto, E., Salakhutdinov, R.: Active neural localization. In: International Conference on Learning Representations (2018) [1](#), [2](#), [3](#), [4](#), [9](#), [10](#), [11](#), [12](#), [13](#)

16. Clark, R., Wang, S., Markham, A., Trigoni, N., Wen, H.: Vidloc: A deep spatio-temporal model for 6-dof video-clip relocalization. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pp. 6856–6864 (2017) [3](#)
17. Cox, I.J., Leonard, J.J.: Modeling a dynamic environment using a bayesian multiple hypothesis approach. *Artificial Intelligence* **66**(2), 311–344 (1994) [3](#)
18. Dellaert, F., Fox, D., Burgard, W., Thrun, S.: Monte carlo localization for mobile robots. In: *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No. 99CH36288C)*. vol. 2, pp. 1322–1328. IEEE (1999) [3](#)
19. Dornhege, C., Kleiner, A.: A frontier-void-based approach for autonomous exploration in 3d. *Advanced Robotics* **27**(6), 459–468 (2013) [14](#)
20. Fox, D.: Markov localization—a probabilistic framework for mobile robot localization and navigation. Ph.D. thesis, Universität Bonn (1998) [3](#), [13](#)
21. Fox, D., Burgard, W., Thrun, S.: Active markov localization for mobile robots. *Robotics and Autonomous Systems* **25**(3), 195–208 (1998) [1](#), [2](#), [4](#), [10](#), [13](#)
22. Fox, D., Burgard, W., Thrun, S.: Markov localization for mobile robots in dynamic environments. *Journal of artificial intelligence research* **11**, 391–427 (1999) [3](#)
23. Halber, M., Shi, Y., Xu, K., Funkhouser, T.: Rescan: Inductive instance segmentation for indoor rgbd scans. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. pp. 2541–2550 (2019) [1](#)
24. Jensfelt, P., Kristensen, S.: Active global localization for a mobile robot using multiple hypothesis tracking. *IEEE Transactions on Robotics and Automation* **17**(5), 748–760 (2001) [1](#), [3](#), [9](#)
25. Kendall, A., Cipolla, R.: Geometric loss functions for camera pose regression with deep learning. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pp. 5974–5983 (2017) [3](#)
26. Kendall, A., Grimes, M., Cipolla, R.: Posenet: A convolutional network for real-time 6-dof camera relocalization. In: *Proceedings of the IEEE international conference on computer vision*. pp. 2938–2946 (2015) [3](#)
27. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014) [21](#)
28. Luo, W., Sun, P., Zhong, F., Liu, W., Zhang, T., Wang, Y.: End-to-end active object tracking and its real-world deployment via reinforcement learning. *IEEE transactions on pattern analysis and machine intelligence* **42**(6), 1317–1332 (2019) [1](#)
29. Mariottini, G.L., Roumeliotis, S.I.: Active vision-based robot localization and navigation in a visual memory. In: *2011 IEEE International Conference on Robotics and Automation*. pp. 6192–6198. IEEE (2011) [1](#), [3](#)
30. Meng, L., Tung, F., Little, J.J., Valentin, J., de Silva, C.W.: Exploiting points and lines in regression forests for rgb-d camera relocalization. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. pp. 6827–6834. IEEE (2018) [1](#), [3](#), [4](#), [11](#)
31. Mur-Artal, R., Tardós, J.D.: Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE transactions on robotics* **33**(5), 1255–1262 (2017) [4](#)
32. Park, J., Zhou, Q.Y., Koltun, V.: Colored point cloud registration revisited. In: *Proceedings of the IEEE international conference on computer vision*. pp. 143–152 (2017) [9](#)
33. Qi, C.R., Yi, L., Su, H., Guibas, L.J.: Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In: *Advances in neural information processing systems*. pp. 5099–5108 (2017) [21](#)

34. Radwan, N., Valada, A., Burgard, W.: Vlocnet++: Deep multitask learning for semantic visual localization and odometry. *IEEE Robotics and Automation Letters* **3**(4), 4407–4414 (2018) [3](#)
35. Ramakrishnan, S.K., Jayaraman, D., Grauman, K.: An exploration of embodied visual exploration. *International Journal of Computer Vision* **129**(5), 1616–1649 (2021) [10](#)
36. Roumeliotis, S.I., Bekey, G.A.: 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)*. vol. 3, pp. 2985–2992. IEEE (2000) [3](#)
37. Sarlin, P.E., Cadena, C., Siegwart, R., Dymczyk, M.: From coarse to fine: Robust hierarchical localization at large scale. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pp. 12716–12725 (2019) [3](#)
38. Sattler, T., Leibe, B., Kobbelt, L.: Fast image-based localization using direct 2d-to-3d matching. In: *2011 International Conference on Computer Vision*. pp. 667–674. IEEE (2011) [3](#)
39. Sattler, T., Zhou, Q., Pollefeys, M., Leal-Taixe, L.: Understanding the limitations of cnn-based absolute camera pose regression. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pp. 3302–3312 (2019) [3](#)
40. Savva, M., Kadian, A., Maksymets, O., Zhao, Y., Wijmans, E., Jain, B., Straub, J., Liu, J., Koltun, V., Malik, J., et al.: Habitat: A platform for embodied ai research. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. pp. 9339–9347 (2019) [4](#)
41. Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O.: Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347* (2017) [10](#)
42. Shotton, J., Glocker, B., Zach, C., Izadi, S., Criminisi, A., Fitzgibbon, A.: Scene coordinate regression forests for camera relocalization in rgb-d images. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pp. 2930–2937 (2013) [1](#), [3](#), [4](#), [5](#), [6](#), [8](#), [11](#), [22](#)
43. Straub, J., Whelan, T., Ma, L., Chen, Y., Wijmans, E., Green, S., Engel, J.J., Mur-Artal, R., Ren, C., Verma, S., et al.: The replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797* (2019) [19](#)
44. Taira, H., Okutomi, M., Sattler, T., Cimpoi, M., Pollefeys, M., Sivic, J., Pajdla, T., Torii, A.: Inloc: Indoor visual localization with dense matching and view synthesis. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pp. 7199–7209 (2018) [3](#)
45. Thrun, S., Fox, D., Burgard, W., Dellaert, F.: Robust monte carlo localization for mobile robots. *Artificial intelligence* **128**(1-2), 99–141 (2001) [3](#)
46. Valada, A., Radwan, N., Burgard, W.: Deep auxiliary learning for visual localization and odometry. In: *2018 IEEE international conference on robotics and automation (ICRA)*. pp. 6939–6946. IEEE (2018) [3](#)
47. Valentin, J., Dai, A., Nießner, M., Kohli, P., Torr, P., Izadi, S., Keskin, C.: Learning to navigate the energy landscape. In: *2016 Fourth International Conference on 3D Vision (3DV)*. pp. 323–332. IEEE (2016) [6](#), [22](#)
48. Valentin, J., Nießner, M., Shotton, J., Fitzgibbon, A., Izadi, S., Torr, P.H.: Exploiting uncertainty in regression forests for accurate camera relocalization. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. pp. 4400–4408 (2015) [1](#), [3](#), [4](#), [11](#)
49. Wald, J., Sattler, T., Golodetz, S., Cavallari, T., Tombari, F.: Beyond controlled environments: 3d camera re-localization in changing indoor scenes. *arXiv preprint arXiv:2008.02004* (2020) [22](#)

50. Wang, B., Chen, C., Lu, C.X., Zhao, P., Trigoni, N., Markham, A.: Atloc: Attention guided camera localization (2020) [3](#)
51. Xia, F., Zamir, A.R., He, Z., Sax, A., Malik, J., Savarese, S.: Gibson env: Real-world perception for embodied agents. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 9068–9079 (2018) [19](#)
52. Xue, F., Wang, X., Yan, Z., Wang, Q., Wang, J., Zha, H.: Local supports global: Deep camera relocalization with sequence enhancement. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 2841–2850 (2019) [3](#)
53. Yang, L., Bai, Z., Tang, C., Li, H., Furukawa, Y., Tan, P.: Sanet: Scene agnostic network for camera localization. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 42–51 (2019) [1](#), [3](#), [11](#)
54. Ye, J., Batra, D., Das, A., Wijmans, E.: Auxiliary tasks and exploration enable objectgoal navigation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 16117–16126 (2021) [10](#)
55. Ye, K., Dong, S., Fan, Q., Wang, H., Yi, L., Xia, F., Wang, J., Chen, B.: Multi-robot active mapping via neural bipartite graph matching. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 14839–14848 (2022) [1](#)
56. Zhong, F., Sun, P., Luo, W., Yan, T., Wang, Y.: Ad-vat+: An asymmetric dueling mechanism for learning and understanding visual active tracking. *IEEE transactions on pattern analysis and machine intelligence* **43**(5), 1467–1482 (2019) [1](#)
57. Zhong, F., Sun, P., Luo, W., Yan, T., Wang, Y.: Towards distraction-robust active visual tracking. In: International Conference on Machine Learning. pp. 12782–12792. PMLR (2021) [1](#)
58. Zhou, L., Luo, Z., Shen, T., Zhang, J., Zhen, M., Yao, Y., Fang, T., Quan, L.: Kfnet: Learning temporal camera relocalization using kalman filtering. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4919–4928 (2020) [3](#)
59. Zhou, Q.Y., Park, J., Koltun, V.: Open3d: A modern library for 3d data processing. arXiv preprint arXiv:1801.09847 (2018) [14](#)

Appendix

The appendix provides the additional supplemental material that cannot be included in the main paper due to its page limit:

- Algorithm illustration
- More results.
- More analysis.
- More implementation details.
- More details of the ACL-synthetic/real datasets.

A Algorithm illustration

We summarize our proposed algorithm in Algorithm [1](#).

Algorithm 1 The full pipeline of our algorithm

```

function PASSIVE LOC. MODULE(observation  $I^{(t)}$ , posed RGB-D stream  $\{I_{basis}^{(i)}, C_{basis}^{(i)}\}_{i=1}^m$ )
  if initialization then
    initialization  $\leftarrow$  false
    Adapt the passive localizer by posed RGB-D stream  $\{I_{basis}^{(i)}, C_{basis}^{(i)}\}_{i=1}^m$ 
    Construct the scene model  $D_{scene}$  by fusing posed RGB-D stream  $\{I_{basis}^{(i)}, C_{basis}^{(i)}\}_{i=1}^m$ 
    Current pose estimation  $\hat{C}^{(t)} \leftarrow$  Passive localizer( $I^{(t)}$ )
  return  $\hat{C}^{(t)}$ 

function ACTIVE LOC. MODULE(pose estimation  $\hat{C}^{(t)}$ , scene model  $D_{scene}$ )
   $M_{wd}^{(t)}, M_{cd}^{(t)} \leftarrow$  Scene uncertainty computation( $\{\hat{C}^{(t)}, D_{scene}\}$ )
   $U_{cu}^{(t)} \leftarrow$  Camera uncertainty computation( $\{\hat{C}^{(t)}, D_{scene}\}$ )
  Action  $a^{(t)} \leftarrow$  Policy network( $\{M_{wd}^{(t)}, M_{cd}^{(t)}\}$ )
  return  $U_{cu}^{(t)}, a^{(t)}$ 

procedure ENTIRE PIPELINE(posed RGB-D stream  $\{I_{basis}^{(i)}, C_{basis}^{(i)}\}_{i=1}^m$ , accuracy threshold  $\lambda_{cu}$ )
   $t \leftarrow 0$ 
   $D_{scene} \leftarrow NULL$ 
  initialization  $\leftarrow$  true
  while  $t <$  maximum step length do
    Obtain the current observation  $I^{(t)}$ 
     $\hat{C}^{(t)} \leftarrow$  PASSIVE LOC. MODULE( $I^{(t)}, \{I_{basis}^{(i)}, C_{basis}^{(i)}\}_{i=1}^m$ )
     $U_{cu}^{(t)}, a^{(t)} \leftarrow$  ACTIVE LOC. MODULE( $\hat{C}^{(t)}, D_{scene}$ )
    if  $U_{cu}^{(t)}$  is within  $\lambda_{cu}$  cm,  $\lambda_{cu}$  degrees then
      break
    Execute the action  $a^{(t)}$ 
     $t \leftarrow t + 1$ 
  return  $\hat{C}^{(t)}$ 

```

B More results

B.1 Comparison on the sparse data

The posed RGB-D stream is the basis for both passive and active localization. To further validate the robustness of our proposed algorithm, we discard half of the posed RGB-D stream as the sparse data for evaluation. The numerical comparisons with the best baselines (Camera-descent/Scene-descent) on both the sparse data and dense data (default setting in the main paper) are shown in Table 3. We observe that all the methods achieve worse results on the sparse data as expected, yet our approach still outperforms the other competitors.

B.2 Comparison on more real-world datasets

To further evaluate the compatibility of our method, we compare our approach and its best competitors on 10 scenes of the real-world Gibson V2 [51] and Replica [43] datasets besides ACL-synthetic/-real datasets in the main paper. Shown in Table 4, our results consistently outperforms the others.

B.3 More qualitative results

In Figure 6, we show the qualitative results of the intelligent behaviors of our algorithm on more test scenes.

Table 3. Numerical results on both the dense and sparse data.

Data	Methods	ACL-synthetic		ACL-real	
		Acc (%)	#steps	Acc (%)	#steps
Dense	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	83.05	17.33	82.40	17.90
Sparse	Camera-descent (t+1)	55.45	24.17	49.60	31.62
	Camera-descent (t+2)	55.05	28.15	52.40	37.35
	Scene-descent	19.90	6.12	41.40	22.71
	Ours	82.00	20.52	76.40	22.54

Table 4. Numerical results on GibsonV2 and Replica datasets.

Methods	GibsonV2		Replica	
	Acc (%)	#steps	Acc (%)	#steps
Camera-descent (t+1)	57.60	23.51	67.80	19.04
Camera-descent (t+2)	51.60	25.42	69.80	26.13
Scene-descent	56.20	16.16	62.80	14.60
Ours	75.00	15.27	86.80	13.30

C More analysis

We provide more analysis of the camera uncertainty component below.

The iterative closest point (ICP) approach is based on the general assumption that the two input point clouds are roughly aligned. When the estimated camera pose of the current frame is far from its ground truth, such as 20cm, 20°, the camera uncertainty component generated by ICP becomes unstable and not reliable to determine the adaptive stop condition. To be specific, following the experiment of “Analysis of camera uncertainty” in the main paper, we further summarize that when the estimated relative pose is within 20cm, 20° ($\lambda_{cu} = 20$), about 83.57% (3220/3853) samples are truly within 20cm, 20° compared to the ground truth, which is much smaller than 94.14% for 5cm, 5° ($\lambda_{cu} = 5$).

Therefore, a natural question to ask is, when evaluating the camera pose in a coarse level, such as 20cm, 20°, what is the best parameter value (λ_{cu}) to determine the adaptive stop condition for the highest camera pose accuracy? In Table 5, we compare the numerical results of our algorithm trained with different parameter values ($\lambda_{cu} = 5/20$) and evaluated on the coarse-scale accuracy (20cm, 20°). We observe that the camera pose accuracy is much worse with $\lambda_{cu} = 20$, which validates the parameter selection λ_{cu} for the camera uncertainty component.

Table 5. Numerical results of our algorithm trained with different parameter values ($\lambda_{cu} = 5, 20$) on the ACL-synthetic dataset.

Uncertainty parameters	Accuracy (20cm, 20°)
$\lambda_{cu} = 5$	85.92
$\lambda_{cu} = 20$	49.40

D More implementation details

D.1 Policy network

The policy network takes the scene uncertainty component as input and generates the probability of the three actions defined by the action space. The camera-driven scene map is represented as a 3-channel 2D map $M_{cd}^{(t)}$, which can be easily consumed by the convolution operation. We employ a convolution neural network of 6 convolution layers (32-64-128-128-256-256) and 1 linear layer (64) to extract the global feature ($\mathbb{R}^{64}$). Each convolution layer is of kernel size 3x3 and followed by a batch normalization layer and a max pooling layer of stride 2. The world-driven scene map is represented as a 6-channel point cloud $M_{wd}^{(t)}$. Inspired by the popular point cloud processing network PointNet [33], we employ a three-layer pointwise MLP (64-128-64) followed by a max pooling layer to extract its global feature ($\mathbb{R}^{64}$). Finally, by concatenating all the extracted features, we use a three-layer MLP (64-16-3) to predict the probability of the three predefined actions.

D.2 Noise perturbation on the action space

To simulate robotic agents in a real-world condition, the action does not lead to perfect execution, hence we add the Gaussian noise to each action. To be specific, if the agent turns left or right, the Gaussian noise of standard deviation (6) will be added to the rotation angle $\theta^{(t)}$ of mean value (20); if the agent moves forward, the Gaussian noise of standard deviation (5) will be added on the 2D positions $x^{(t)}, y^{(t)}$ of mean value (30). The positions are measured in centimeters, and the rotation angle is measured in degrees. Note the standard deviation (σ) is actually very large considering 31.74% of sampled noises are beyond σ for the Gaussian distribution.

D.3 Implimentation details

In our experiment, we employ the Adam [27] to optimize the network weights with the initial learning rate of 3×10^{-4} . Some hyper-parameters: $N_{cd} = 12$, $N_{wd_r} = 1000$, $N_{wd_p} = 2^{14} = 16384$, $N_f = 5$, $X = 256$, $Y = 256$.

Table 6. Scene statistics of the ACL-synthetic and ACL-real dataset. We summarize the number of scenes, scene area, max scene area, min scene area and the number of frames in the RGB-D sequences. The unit for all areas is m^2 . The Area and #frames metrics are averaged over all the scenes involved.

Scene		#scenes	Area	Max area	Min area	#frames
ACL-synthetic	Train split	15	37.89	49.40	25	58.00
	Test split	20	43.17	75.00	26.9	54.45
ACL-real	Test split	5	64.82	98.28	23.62	88.40
All		40	43.90	98.28	23.62	60.03

E More details of the ACL-synthetic/real datasets

The posed RGB-D stream in the existing camera localization datasets [42,47,49] is usually obtained by scanning the environment with handheld sensors by human operators, hence does not always cover the complete scene model. We design the posed RGB-D stream in our dataset to simulate this effect. Directly visualizing the trajectory of the posed RGB-D stream in the scene is not intuitive as it would lose the orientation information of the camera pose, instead we choose to visualize the scene model reconstructed from the posed RGB-D stream to showcase how much scene region is covered by the posed RGB-D stream.

We illustrate the textured meshes of both the complete and reconstructed scene models for the ACL-synthetic and ACL-real datasets in Figure 7, 8 and 9. Their related statistics are shown in Table 6.

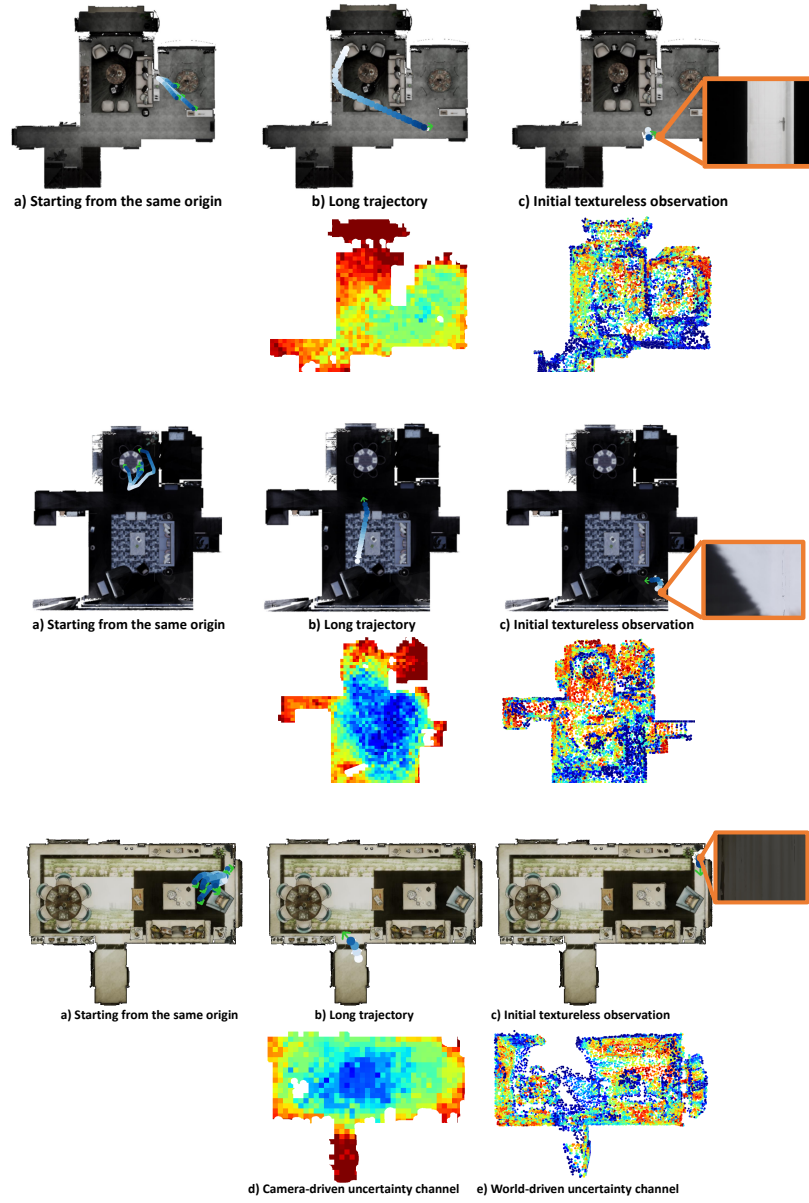


Fig. 6. Qualitative results of the intelligent behaviors learned by our algorithm.

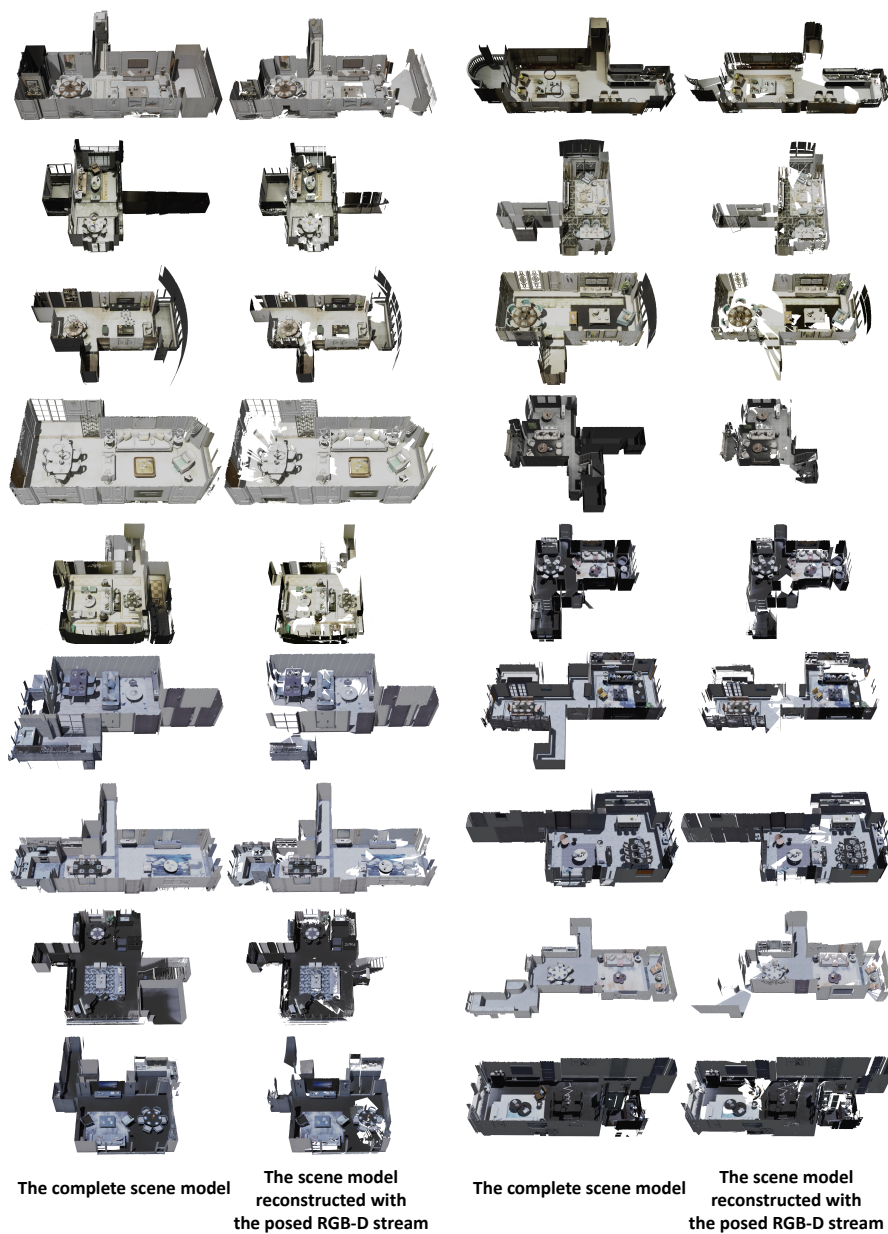


Fig. 7. Visualization of the ACL-synthetic dataset.

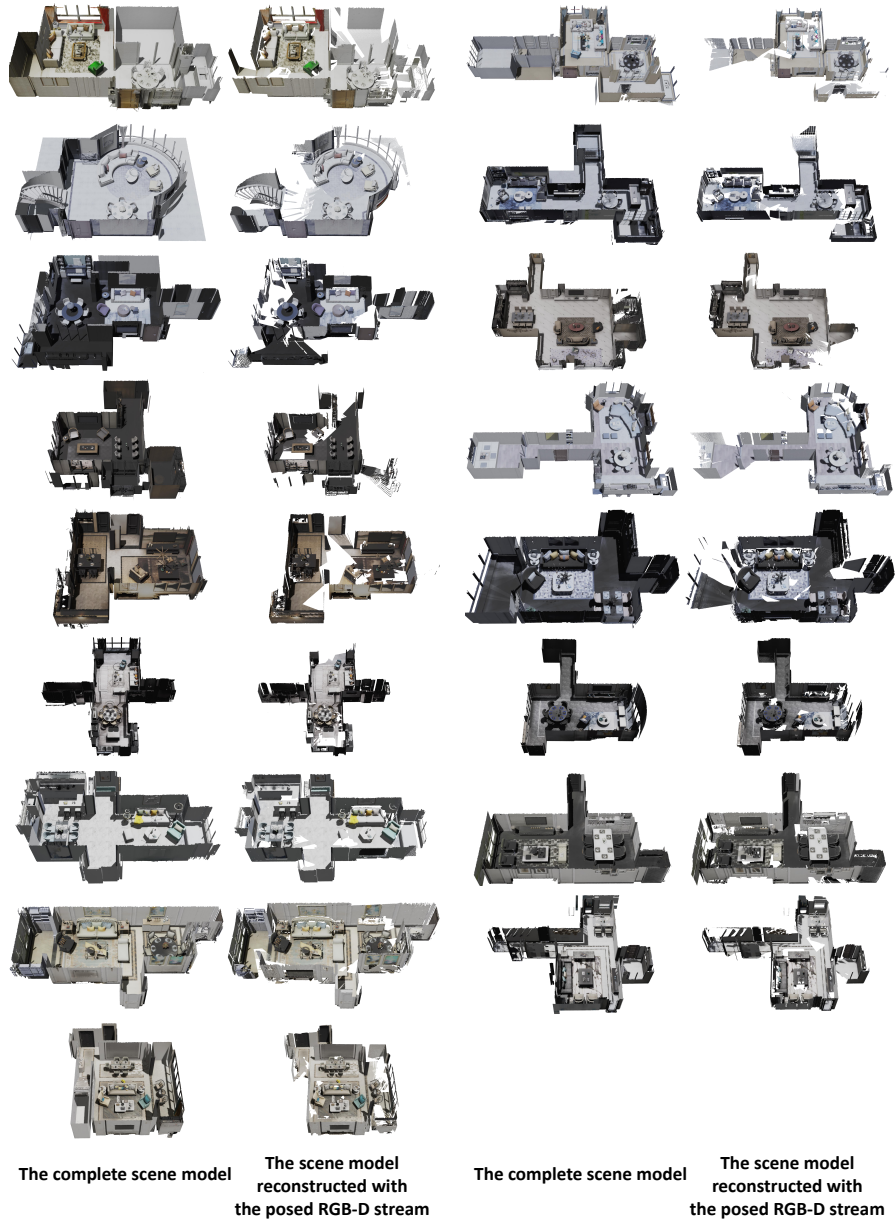


Fig. 8. Visualization of the ACL-synthetic dataset.

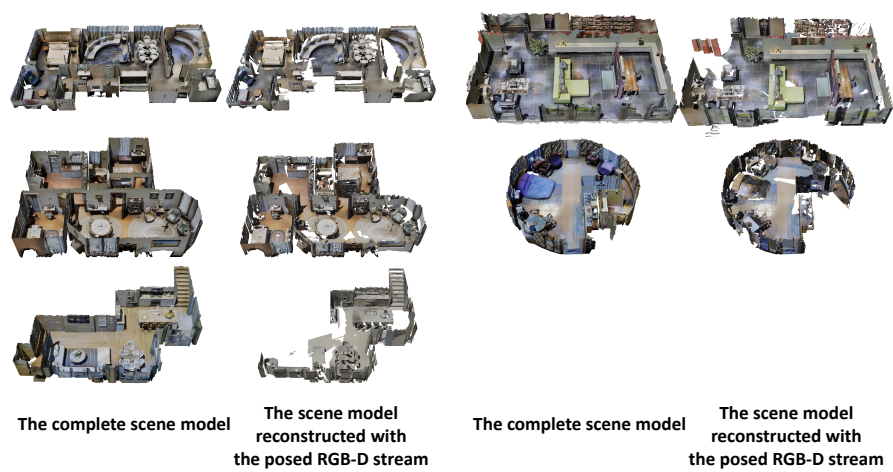


Fig. 9. Visualization of the ACL-real dataset.