2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 14-18, 2024. Abu Dhabi, UAE

Deeper Introspective SLAM: How to Avoid Tracking Failures Over Longer Routes?

Kanwal Naveed[1] , Muhammad Latif Anjum[1] , Wajahat Hussain[1] and Donghwan Lee[2]

***Abstract*—Large scale active exploration has recently revealed limitations of visual SLAM's tracking ability. Active view planning methods based on reinforcement learning have been proposed to improve visual tracking robustness.**

In this work, we expose the limitations of deep reinforcement learning-based visual SLAM over longer routes. We demonstrate that additional modalities (depth, scene layout) offer little improvement. Furthermore, reward shaping is not the main reason behind the shortsightedness of the state-of-the-art visual SLAM tracker. We propose a novel video vision transformer-based architecture that improves the farsightedness of the visual tracker, which results in the completion of longer routes with efficient paths.

Out of 60 challenging routes, our approach manages to complete 56 routes, which is a three-fold improvement over the state-of-the-art active view mapping (DI-SLAM) baseline. Interestingly, ORB-SLAM3 was unable to complete a single route without tracking failure. Our code is available at https://tinyurl.com/w935spuz.

I. Introduction

There are a few well-known challenging scenarios for visual tracking, including texture-less scenes [6], motion blur [16] and dynamic content. Human-supervised datasets have led to another bias, which has resulted in the overestimation of visual tracking performance [31].

State-of-the-art SLAM algorithms successfully complete tracking on long routes and are judged by the accuracy of their performance. However, recently, multiple researchers have revealed the weakness of these visual trackers on relatively shorter routes ($<$10m) when the camera view is not human-controlled [18, 21]. Interestingly, a blind navigation agent, having a rough idea of the goal direction, performed better than the agent having visual information [21].

The advent of realistic and high fidelity simulators [26, 27], which consist of larger numbers of different environments, as compared to well-known benchmarks [29, 8], has allowed large-scale active evaluation, which has led to these findings. These simulators have made active camera view planning relatively easier to evaluate. Active navigation in the physical world has numerous constraints [1], which has stunted the growth of these algorithms. Furthermore, these simulators have also demonstrated acceptable simulation to reality performance transfer [21].

In this work, we evaluate a state-of-the-art active visual SLAM approach called Deep Introspective SLAM (DI-SLAM) [21]. In this work, Naveed et al. [21] developed a deep reinforcement learning (RL) approach to avoid tracking failures in visual SLAM. This active mapping approach managed to improve the track length of the state-of-the-art visual SLAM (ORB-SLAM [20]) by five times.

Our evaluation of DI-SLAM [21] reveals that this system performs well over shorter routes. However, this active mapping approach suffers over longer routes (Fig. 1). The path generated by this active mapping approach has a zigzag pattern (Fig. 1a), which makes the tracking relatively robust but results in inefficient paths. Additionally, this zigzag path requires a sudden change in navigating the agent's direction, which is difficult to achieve in the physical world, considering the inertia of the moving agent.

Investigating the failure cases of DI-SLAM reveals that it gets stuck in corners and closeups (Fig. 1), which ultimately results in tracking failure. Intuitively, including depth of information may improve the spatial understanding of the agent and help avoid these closeups. Surprisingly, including the raw depth information offers little improvement (Fig. 1c, Table I).

Interestingly, providing the top view (layout [25]) of the environment improves the planning of the agent. This indicates that DI-SLAM is unable to transform raw depth into a meaningful representation (layout) of the scene and requires additional help. Previously, Salas et al. [25] demonstrated that a high level of understanding of the scene improves tracking, reconstruction and map management. However, extracting the layout of the scene is limited to a few (mainly cuboid) scenes [25]. Additionally, this affects the real-time requirement of SLAM.

Crafting the reward function is another component that shares the responsibility for the under-performance of reinforcement learning. This limitation of reward may be avoided by leveraging imitation learning [6]. However, imitation learning requires human involvement, limiting its effectiveness in novel scenarios. Our results demonstrate that reward shaping is not the main reason causing short-sightedness in DI-SLAM [21].

In this work, we propose a memory-based approach, which requires only visual input to improve the short-term planning of DI-SLAM. We evaluate backbones based on long-short

[1]The authors are with Robotics & Machine Intelligence (ROMI) Lab, School of Electrical Engineering and Computer Science (SEECS), National University of Sciences and Technology, Islamabad, Pakistan. {knaveed.phdee17seecs,latif.anjum, wajahat.hussain}@seecs.edu.pk

[2]The author is with Reinforcement Learning Research Lab, School of Electrical Engineering, Korea Advanced Institute of Science and Technology (KAIST), Daejeon, South Korea. donghwan@kaist.ac.kr

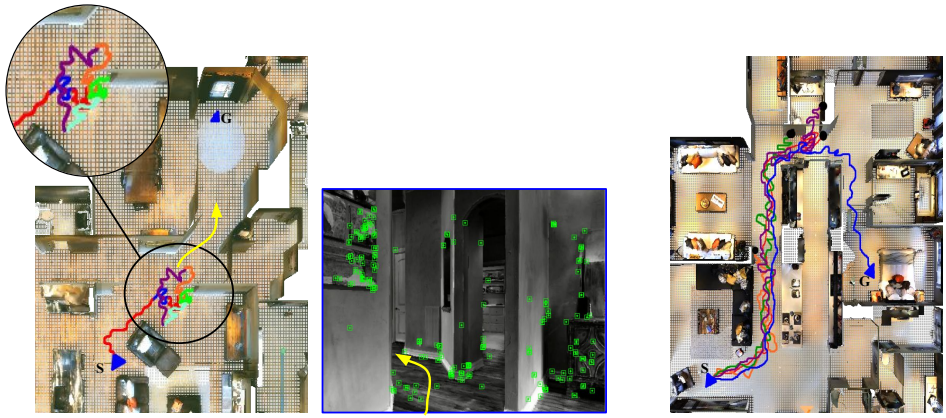


Fig. 1: **Visual Tracking Over Longer Routes:** (a) We evaluate the tracking performance of the state-of-the-art active view mapping (DI-SLAM [21]) over a route that spans multiple rooms. The DI-SLAM agent starts well; however, when it is about to enter alley (b), which requires multiple turns in quick succession, the agent gets confused and starts making random back-and-forth moments. This indicates the short-term planning nature of DI-SLAM. This limitation is again visible in another episode (c), where the DI-SLAM agent is unable to make a sharp turn (red line). Interestingly, providing the raw depth (orange line) results in tracking failure earlier along the route. Providing the scene layout information improves the performance (green line) but fails around the sharp turn. Our proposed approach (blue line) manages to reach the goal. S: Start, G: Goal.

term memory (LSTM) [10] and transformer architecture [3]. Both these pipelines are known to improve long-term reasoning.

Our proposed approach improves the performance of active mapping 3 times. Out of 60 episodes, ORB-SLAM3 [4] was unable to complete a single route, DI-SLAM [21] completed the route 19 times, while our proposed approach managed to complete the challenge on 56 scenes (Table I). Furthermore, our approach takes 20 % fewer steps to reach the goal than the DI-SLAM [21], indicating a reduced change in path directions, making it more practical.

Contributions

- In this work, we reveal the short-term planning of DI-SLAM [21], which results in shorter navigation routes due to tracking failures. Furthermore, the aforementioned short-sightedness results in a zigzag motion, resulting in inefficient path planning.
- We introduce long-term reasoning in active view mapping, which increases the success rate of tracking over longer routes, as well as achieves more efficient path planning using only a monocular camera.

II. Related Work

ALVINN [22] is the pioneering work which incorporated active planning for navigation agents. In another seminal work, Michels et al. [17] demonstrated navigation using a single camera. The aim of these agents was to complete the navigation by either staying on the road and/or avoiding obstacles. Maintaining visual tracking was not the focus of these approaches.

There is a long history of improving the robustness of visual tracking. The classic approaches fall into two categories. The first approach consists of planning safe paths [12], which usually means paths having loops, which improves the quality of the map and ultimately results in robust tracking. There are a few approaches [19, 5] that try to choose a feature-rich path a.k.a. next-best-view (NBV), leveraging the available 3D map ([19]) or the image texture [5]. However, these approaches have limited utility, as shown in [11], especially for indoor scenarios.

The second approach consists of post-failure tracking recovery methods. PTAM [14] incorporated a relocalization module to recover from tracking failures. ORB-SLAM [20] has built an elaborated loop closing pipeline which has been adopted by multiple SLAM pipelines and has become the de facto recovery method. However, there are no guarantees that the agent itself will be able to complete the loop and recover from failures.

Standard benchmarks [8, 29] include long persistent revisits which make it easier for loop closing methods to recover from the tracking loss [13]. Recently. Qureshi et al. [11] demonstrated an active loop closing approach. However, this approach requires recovery action to be initiated instantly, which is not possible due to the inertia of the navigation agent.

Above mentioned post failure recovery mechanisms offer no guarantees of recovery. If the agent manages to revisit the place only then these methods will come into effect. Our focus is predicting the tracking failures beforehand thereby avoiding them.

There are a few learning-based approaches proposed for outdoor scenarios to avoid tracking failures. Hartmann et al. [9] learned the match-ability of features, which allowed maintaining only those features which will result in long-term tracking. This approach is best suited for environments with repeating textures (trees, leaves). Saxena et al. [28] learned maneuvers to recover from tracking failures under sun glare, large shadows, and closeups. Similarly, Rabiee et al. [24] leveraged depth along with images to manage sun glare. The focus of our work is reliable tracking in indoor scenarios,

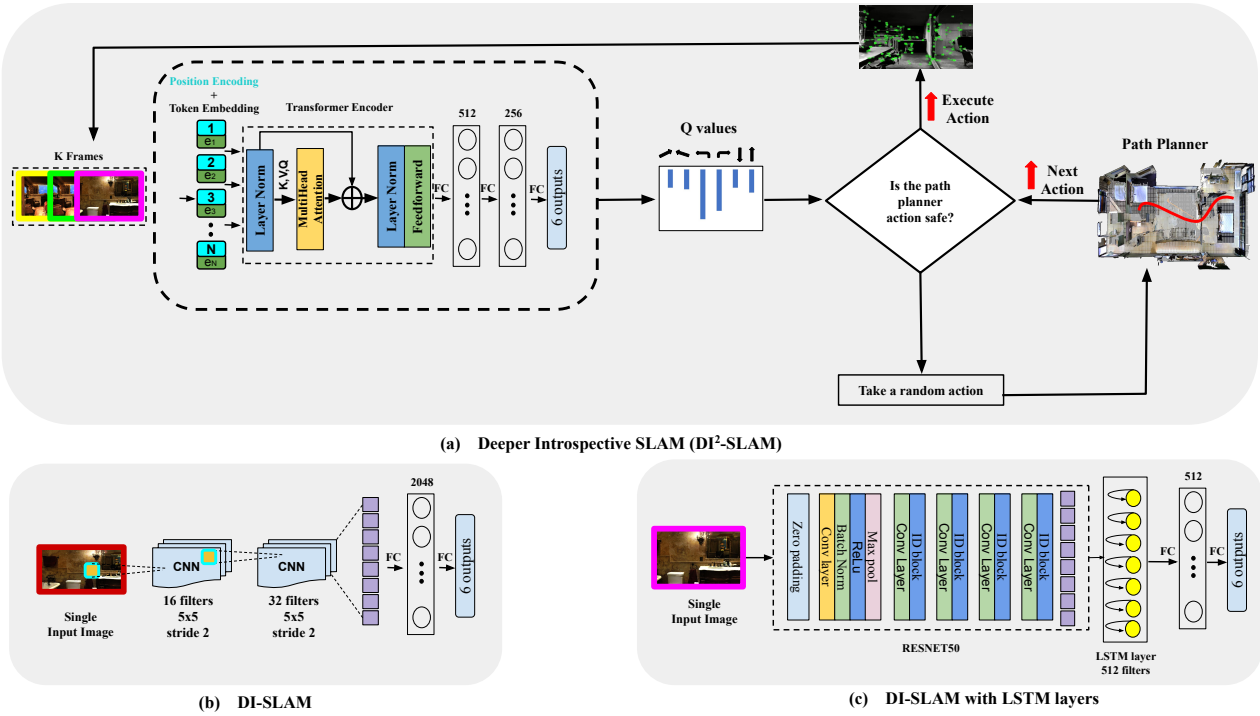


Fig. 2: **Overview**: Our Deeper Introspective SLAM (D^2I-SLAM) leverages the large receptive field of the video transformer (dashed box) to evaluate the safety of the action proposed by the path planner by considering a large batch (32) of images at a time. The previous active mapping approach (DI-SLAM [21]) resulted in a reactive policy due to the limited receptive field of CNN architecture (b) deployed, which considers a single image at a time. Deploying a much deeper network (c) having a combination of convolutional and long short-term memory (LSTM) layers improves the farsightedness of the active mapping agent; however, the proposed video transformer-based approach outperforms all the baselines.

which are mostly textureless, and generating recovery manoeuvres is slightly more challenging.

There are a few custom solutions designed to avoid tracking failures in (texture-less) indoor scenarios. Prasad et al. [23] proposed a Q learning-based approach to avoid tracking failures around corners and sharp turns. Naveed et al. [21] fused this Q-learning approach with deep networks. They evaluated their deep reinforcement learning DI-SLAM on a larger scale, leveraging the recently made available realistic simulators which offer a large variety of indoor scenes. Dai et al. [6] proposed an imitation learning approach to avoid tracking failures indoors. The imitation learning approach requires human input, which limits extending it to novel scenarios. Therefore, similar to DI-SLAM [21], our focus is an automatic reinforcement learning approach to improve tracking over longer paths.

III. Deeper Introspective SLAM (D^2I-SLAM)

In this section, we describe in detail our novel approach to improve long-term reasoning in active visual SLAM, leading to robust tracking performance over longer routes. The training and experimental environment during deployment are inspired by Naveed et al.[21] using ORB-SLAM3.

A. D^2I-SLAM Deployment Policy

Our aim is to learn a $Q_\phi(\mathbf{s}, a)$ function which evaluates the quality of action a given the current state $\mathbf{s}$. At each step, the selected action must avoid the tracking failure (Fig. 2a). Prior work on active mapping [21] considers an individual current frame as the current state, and the action is selected from a set of six actions/moves, i.e., $\mathscr{A} = \{$forward/backwards, clockwise/anticlockwise rotation, lateral left/right translation$\}$. Given the start and the goal position, the in-built path planner of simulator [26] provides the navigation path (Fig. 2a). The agent uses the monocular visual input to track its motion using ORBSLAM3 [4]. At each step along the planned path, the agent evaluates the safety of the recommended action a using the $Q_\phi(\mathbf{s}, a)$ function. If the corresponding Q value of action a is higher than the threshold value (-28), the agent will perform the recommended action a. Otherwise, the navigating agent takes a random action. The path is re-planned at this stage from the current location to the provided goal. The process is repeated until the agent reaches the goal or loses tracking.

B. D^2I-SLAM's Leveraging Video Transformer For Farsightedness

As mentioned earlier, DI-SLAM [21] manages to improve tracking of the state-of-the-art visual SLAM (ORB-SLAM3 [4]) over short paths; however, it has difficulty maintaining tracking over longer paths. DI-SLAM deploys a convolutional neural network (CNN) based architecture (Fig. 2b) to learn the $Q_\phi(\mathbf{s}, a)$ function. However, CNN is unable to see the big picture (limited receptive field) as compared to recurrent neural networks (RNN) and visual transformers [30, 3].

Therefore, we plan to overcome this shortsightedness by leveraging the visual transformer-based architecture (ViViT [3]) that takes a video segment as input (Fig. 2a) as compared to a single image. As shown in the results, our proposed approach outperforms even the architecture based on a combination of convolutional and long short-term memory (LSTM) layers (Fig. 2c).

More formally, at any given instant, our tracking agent has access to the current frame and k-1 previous frames. This set of k (32) frames will form the current state $\mathbf{s}_t = \{\mathbf{I}_t, \mathbf{I}_{t-1}, \ldots, \mathbf{I}_{t-k+1}\}$. Each frame $\mathbf{I}$ is divided into 16 $\times$ 16 patches (as per [7]) leading to a set $\mathscr{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{16\times16\times k}\}$ for a frame of size 256x256. Embedding $\mathbf{e}$ is generated for each patch $\mathbf{x}$, leading to a set $\mathscr{E} = \{\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_{16\times16\times k}\}$. A position encoding vector is added to each embedding. The resulting embeddings are concatenated before feeding into the transformer encoder. Following the transformer encoder, we add three fully connected layers (Fig. 2a, Algorithm 1).

C. D^2I-SLAM Training

Algorithm 1 Deeper Introspective SLAM

Initialize the main and target Q network with random weight values ϕ and ϕ', respectively
Initialize experience replay buffer $\mathbf{D}$
while *max iteration is not reached* **do**
for $t = 0, 1, \ldots, \tau - 1$ **do**
for *each state* $\mathbf{s_t}$ **do**
$\mathscr{X} \Leftarrow$ getPatches($\mathbf{s}_t$) $\triangleright$ $\mathscr{X} = \{\mathbf{x}_1, .., \mathbf{x}_{16\times16\times k}\}$
$\mathscr{E} \Leftarrow$ getEmbeddings($\mathscr{X}$) $\triangleright$ $\mathscr{E} = \{\mathbf{e}_1, .., \mathbf{e}_{256\times k}\}$
$\mathscr{E}_p \Leftarrow$ addPositionEncodings($\mathscr{E}$)
$\mathbf{X} \Leftarrow$ transformerEncoder($\mathscr{E}_p$) $\triangleright$ $\mathbf{X} \in \mathbb{R}^{192\times1}$
$\mathbf{X} \Leftarrow$ FC_Layer($\mathbf{X}$) $\triangleright$ $\mathbf{X} \in \mathbb{R}^{512\times1}$
$\mathbf{X} \Leftarrow$ FC_Layer($\mathbf{X}$) $\triangleright$ $\mathbf{X} \in \mathbb{R}^{256\times1}$
$Q_\phi(\mathbf{s}_t, a) \Leftarrow$ FC_Layer($\mathbf{X}$)
With probability ε select an action a_t
Otherwise a_t = $argmax_a$ Q($\mathbf{s_t}$,a_t)
Observe the next frame $\mathbf{I}_{t+1}$ and reward r_t
From $\mathbf{I}_{t+1}$ create $\mathbf{s}_{t+1}$
Store $(\mathbf{s_t}, a_t, r_t, \mathbf{s_{t+1}})$ in the replay buffer $\mathbf{D}$
Uniformly sample a random mini-batch $\mathbf{B}$ from $\mathbf{D}$
Set the target

$$y = \begin{cases} r_t & \text{if the episode terminates at } \mathbf{s}_{t+1} \\ r_t + \gamma \max_{a_{t+1}\in\mathscr{A}} Q_{\phi'}(\mathbf{s}_{t+1}, a_{t+1}) & \text{otherwise} \end{cases}$$

Minimize the mean square error loss

$$L(\phi) = \frac{1}{2}\frac{1}{|\mathbf{B}|} \sum_{(\mathbf{s}_t, a_t, r_t, \mathbf{s}_{t+1})\in\mathbf{B}} (y - Q_\phi(\mathbf{s}_t, a_t))^2$$

Every C steps, update the target variable: $\phi' \leftarrow \phi$
Set $\mathbf{s}_{t+1} \leftarrow \mathbf{s}_t$

Our visual navigation agent will explore the (simulated) environment to gather experience. During exploration, at any given instant, the action for the current state $\mathbf{s_t}$ is selected via ε greedy policy (Algorithm 1), after which the reward r_t and the next frame $\mathbf{I}_{t+1}$ are observed.

This next frame $\mathbf{I}_{t+1}$ will in return create the next state $\mathbf{s}_{t+1} = \{\mathbf{I}_{t+1}, \mathbf{I}_t, \ldots, \mathbf{I}_{t-k}\}$. The current state, the action performed, the reward gained, and the next state observed are stored as a tuple memory $\mathbf{D}$ (Algorithm 1).

Reward shaping is the next challenge in reinforcement learning. We have adopted the reward (Eq. 1) provided by [23, 21].

$$r(\mathbf{s}, a) = \alpha_1 T_{ov} + \alpha_2 \psi + \alpha_3, \tag{1}$$

where T_{ov} represents the number of overlapping tracking features between two consecutive images. α_1 is the regulation constant for T_{ov} with a value of 10/400, where 400 represents the upper limit of overlapping tracking features. α_3 is a constant term with a value -10 so that the reward remains negative at all times. ψ is the SLAM failure indicator, with the usual value of zero. Its value is raised to 1 when the agent faces a SLAM tracking failure. It is then multiplied with a constant α_2 = -10, thereby making the reward function highly negative. This high negative reward helps in negative reinforcement, which, in turn, ensures SLAM failure avoidance.

As shown in the results, even with the adopted award, our approach manages to maintain tracking over longer routes. Therefore, we conclude that reward shaping is not the hurdle behind the short-term reasoning of DI-SLAM [21].

To balance out the exploration of new actions that have instantaneous rewards with the exploitation of known high-reward actions, we need to keep in mind the role of future rewards. This balancing act is achieved by setting the target y using Eq. 2

$$y = r_t + \gamma \max_{a_{t+1}\in\mathscr{A}} Q_{\phi'}(\mathbf{s}_{t+1}, a_{t+1}), \tag{2}$$

where γ is the discount factor with a typical value ranging between 0 and 1. It decides the weightage given to future rewards.

In order to train our transformer-based Q-network (broken line in Fig. 2a), we sample $\mathbf{B}$ tuples from Memory $\mathbf{D}$. For each experience (episode), the hidden states of the target network are updated as per algorithm 1 by optimizing the loss function $L(\phi)$ (Eq. 3). After every C steps, Q_ϕ replaces $Q_{\phi'}$.

$$L(\phi) = \frac{1}{2}\frac{1}{|\mathbf{B}|} \sum_{(\mathbf{s}_t, a_t, r_t, \mathbf{s}_{t+1})\in\mathbf{B}} (y - Q_\phi(\mathbf{s}_t, a_t))^2 \tag{3}$$

IV. Experiments

A. Baselines

1) ORB-SLAM3 [4]: We evaluate the tracking performance of the state-of-the-art visual SLAM (ORB-SLAM3 [4]) over a variety of challenging scenarios, including longer distances and sharp turns.
2) Safe-SLAM [23]: The Safe-SLAM approach avoided tracking failures in ORBSLAM [20] .We have implemented and tested Safe-SLAM in MINOS [26] using

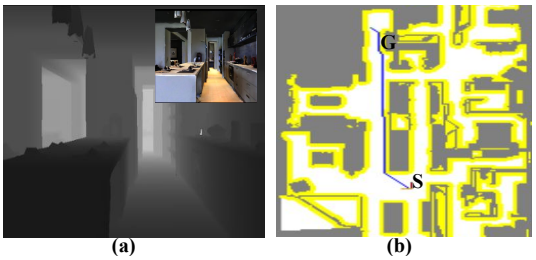


Fig. 3: **Additional Modalities for DI-SLAM [21]**. Depth image (a) and the 2D map (b) provided by MINOS [26].

ORBSLAM3 [4] with a fixed rotation angle of 0.40 rad.

3) DI-SLAM [21]: This approach incorporates a deep RL-based network to predict dangerous actions. This approach utilized ORB-SLAM [20]. We have re-evaluated DI-SLAM with ORB-SLAM3 [4] for a fair comparison.
4) DI-SLAM [21] with RGB-D input: We augmented the RGB image with depth image (Fig. 3a) at the input of the Deep RL network. We retrained the updated depth-conscious approach for approximately ∼60 hours, which is equivalent to the training time of DI-SLAM [21].
5) DI-SLAM [21] with 2D map: In this method, we updated the DI-SLAM [21] by augmenting the RGB image, with a top view (layout) of the scene (Fig. 3b), at the input. We retrained the updated agent for approximately ∼60 hours.
6) DI-SLAM [21] with LSTM layer: In this method, we updated the backbone of DI-SLAM [21] by replacing the shallow network (Fig. 2b) with a much deeper network RESNET50 ([15]). In order to improve the long-term reasoning, we used the long short-term memory (LSTM) layer after RESNET50 (Fig. 2c).

B. D^2I-SLAM Qualitative Analysis

We evaluate our D^2I-SLAM on a long route spanning multiple rooms, containing multiple sharp turns, with a distance of 15.62m between the start (S) and the goal (G) (Fig. 4). In this experiment, the agent needs to enter a living area from outside, cross the living area and enter a room via a wide corridor.

The path provided by the inbuilt path planner (blue line in Fig. 4) of the simulator has multiple sharp turns. Our agent largely follows the path (red line in Fig. 4) closely except at sharp turns (circles highlighted in Fig. 4). We investigate the visual input at these crucial junctures.

Visual input from ORB-SLAM3 [4] shows that the number of tracked features is limited in the region of the recommended next step. Opting for the recommended action might lead to tracking failure. The path planner recommends the right step (purple, green, and yellow frames); however, the number of tracked features is negligible in the right half of the key frame. Hence, our agent avoided taking that action.

Our approach manages to reach the goal in longer routes (∼28 meters) spanning over multiple turns (Fig. 5). Previously active mapping [21] was attempted on shorter routes (∼17 meters).

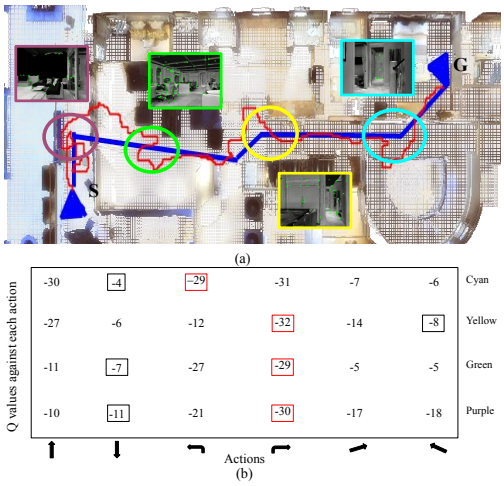
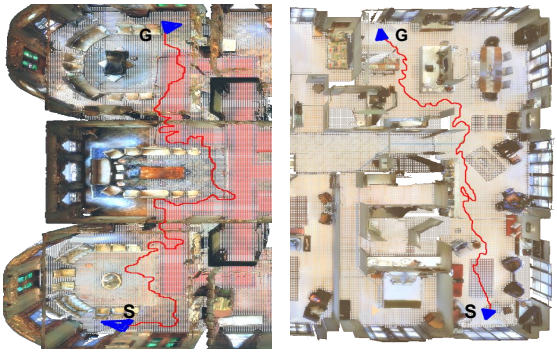


Fig. 4: **D^2I-SLAM in action**. The blue line (a) represents the recommended path from the start (S) towards a goal (G), while the red line represents the path followed by our D^2I-SLAM. Our agent closely follows the optimal path except at sharp turns (circle highlighted) where the recommended action is deemed dangerous (highlighted by red box in (b)) by our approach, in which case the agent makes a random manoeuvre (highlighted by black box in (b)).



(a) Path Length: 20.03m (b) Path Length: 27.58m

Fig. 5: Our D^2I-SLAM manages to successfully complete longer routes (red line) extended over several rooms and multiple turns.

Our approach is real-time and processes the input image with an average inference time of 21 milliseconds. Our system details include the CPU: Intel(R) Core(TM) i5-9300H CPU @ 2.40GHz and GPU: GeForce GTX 1650.

We also analyse the cases in which our approach failed to reach the goal (Fig. 6). A closer look at the failure frames (frontal view of failure point in Fig. 6) shows that the agent fails when it is faced with tight alleys/entrances. These situations have little room for recovery manoeuvres. Additionally, it's difficult to plan the path if the scene is not visible from afar. An agent *can only cross the bridge once it reaches one*.

C. Results Comparison

We collected 60 navigation episodes containing longer distances (with a total path length of ∼ 585 meters for all episodes), sharp turns and texture-less scenes. The re-

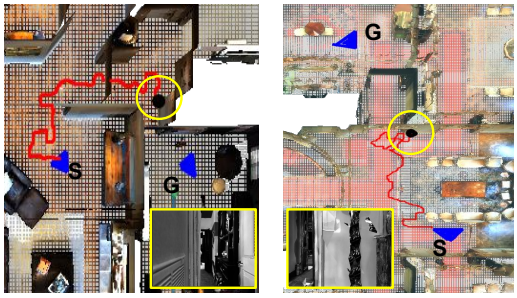


Fig. 6: **Understanding limitations of our approach**. Our agent (red track) fails in scenarios where the prospective scene is not visible from far away, as is the case in scene transitions with narrow entrances.

TABLE I: Tracking Robustness: D^2I-SLAM agent shows resilience over all other baselines. The data is presented for 60 extensive experiments.

Method	Success rate	Number of steps	Short routes	Textureless routes	Sharp turns	Longer routes
ORB-SLAM3 [4]	0/60	1978	✓	✗	✗	✗
Safe-SLAM [23]	9/60	2677	✗	✗	✓	✗
DI-SLAM [21]	19/60	12049	✓	✓	✓	✗
DI-SLAM+Depth (ours)	18/60	11243	✓	✓	✓	✗
DI-SLAM+2D map (ours)	28/60	11585	✓	✓	✓	✗
DI-SLAM+LSTM (ours)	45/60	12330	✓	✓	✓	✓
D^2I-SLAM (ours)	56/60	9756	✓	✓	✓	✓✓

sults presented (Table I) show an average of all navigation episodes.

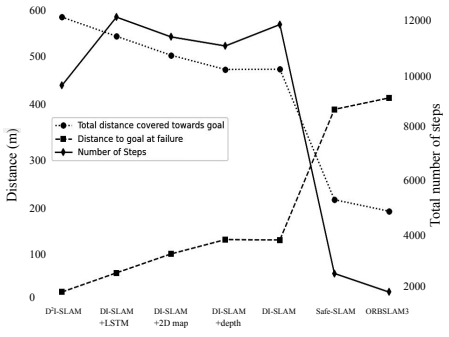


Fig. 7: **Comparison with Baselines.** Compared to previous DQN variants, our Deeper Introspective-SLAM takes less steps and achieves the goal in 56 out of 60 episodes, as evidenced by the largest total distance covered towards the goal.

1) Tracking Robustness

We present an overall quantitative comparison of our approach with other baselines in Fig. 7. We performed 60 experiments out of which our approach managed to reach the goal in 56 experiments. Our approach (Fig. 7) manages to complete episodes while taking lesser number of steps as compared to other baselines. Moreover, Fig. 7 further emphasizes our agent is actually moving towards the goal instead of just roaming around because it has the least distance left towards the goal as compared to all other approaches.

TABLE II: Path Optimality: Our Deeper Introspective SLAM achieves the most optimal path as indicated by SPL [2] value.

Method	ORB-SLAM3 [4]	Safe-SLAM [23]	DI-SLAM [21]	DI-SLAM + Depth (ours)	DI-SLAM + 2Dmap (ours)	DI-SLAM + LSTM (ours)	D^2I-SLAM (ours)
SPL	0.00	0.04	0.08	0.08	0.13	0.23	0.30

2) Path Optimality

According to Anderson et al. [2] SPL i.e. Success Weighted by Path Length is an important parameter to gauge the path optimality of any mobile agent, The metric is given in Eq. 4.

$$SPL = \frac{1}{M}\sum_{i=1}^{M} S_i \frac{l_i}{\max(p_i, l_i)} \tag{4}$$

where M is the total number of episodes, S_i is the binary indicator of success or failure (0,1) in episode i, l_i is the total distance (optimal) to goal in episode i, and p_i is the distance (optimal or otherwise) taken by the agent to reach the goal. According to the SPL equation, it is important for an agent to not only reach its goal but also attain it via an optimal path. As evident from Table II, our approach yields the maximum SPL out of all baselines.

V. Conclusion

We investigated the tracking limitations of deep reinforcement learning-based visual SLAM (DI-SLAM) over longer routes. We found that the existing methods, even when augmented with additional modalities, do not significantly improve tracking performance when the agent is faced with longer distances. To address this issue, we focused on enhancing the farsightedness of the visual tracker and proposed a novel video vision transformer-based deep reinforcement learning architecture. Experimental results demonstrate that our novel approach considerably improves the tracking robustness of Visual SLAM while outperforming other baselines on path optimality. We release our code to assist the community in developing active solutions for robust visual SLAM.

References

[1] Phil Ammirato, Alexander C Berg, and Jana Kosecka. Active vision dataset benchmark. In *CVPR Workshops*, 2018.

[2] Peter Anderson, Angel Chang, Devendra Singh Chaplot, Alexey Dosovitskiy, Saurabh Gupta, Vladlen Koltun, Jana Kosecka, Jitendra Malik, Roozbeh Mottaghi, Manolis Savva, et al. On evaluation of embodied navigation agents. *arXiv preprint arXiv:1807.06757*, 2018.

[3] Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. Vivit: A video vision transformer. In *ICCV*, 2021.

[4] Carlos Campos, Richard Elvira, Juan J Gómez Rodríguez, José MM Montiel, and Juan D Tardós. Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam. *IEEE Transactions on Robotics*, 37(6):1874–1890, 2021.

[5] Gabriele Costante, Christian Forster, Jeffrey Delmerico, Paolo Valigi, and Davide Scaramuzza. Perception-aware path planning. *arXiv preprint arXiv:1605.04151*, 2016.

[6] Xu-Yang Dai, Qing-Hao Meng, Sheng Jin, and Yin-Bo Liu. Camera view planning based on generative adversarial imitation learning in indoor active exploration. *Applied Soft Computing*, 129, 2022.

[7] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[8] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*. IEEE, 2012.

[9] Wilfried Hartmann, Michal Havlena, and Konrad Schindler. Predicting matchability. In *CVPR*, 2014.

[10] Matthew Hausknecht and Peter Stone. Deep recurrent q-learning for partially observable mdps. In *AAAI fall symposium series*, 2015.

[11] Ans Hussain Qureshi, Muhammmad Latif Anjum, Wajahat Hussain, Usama Muddassar, and Sohail Abbasi. One step back, two steps forward: learning moves to recover from SLAM tracking failures. *Advanced Robotics*, pages 1–16, 2024.

[12] Vadim Indelman, Luca Carlone, and Frank Dellaert. Planning in the continuous domain: A generalized belief space approach for autonomous navigation in unknown environments. *The International Journal of Robotics Research*, 34(7):849–882, 2015.

[13] Saran Khaliq, Muhammad Latif Anjum, Wajahat Hussain, Muhammad Uzair Khattak, and Momen Rasool. Why ORB-SLAM is missing commonly occurring loop closures? *Autonomous Robots*, 47(8):1519–1535, 2023.

[14] Georg Klein and David Murray. Parallel tracking and mapping on a camera phone. In *ISMAR*. IEEE, 2009.

[15] Brett Koonce and Brett Koonce. ResNet 50. *Convolutional neural networks with swift for tensorflow: image recognition and dataset categorization*, pages 63–72, 2021.

[16] Peidong Liu, Xingxing Zuo, Viktor Larsson, and Marc Pollefeys. MBA-VO: Motion blur aware visual odometry. In *ICCV*, 2021.

[17] Jeff Michels, Ashutosh Saxena, and Andrew Y Ng. High speed obstacle avoidance using monocular vision and reinforcement learning. In *ICML*, 2005.

[18] Dmytro Mishkin, Alexey Dosovitskiy, and Vladlen Koltun. Benchmarking classic and learned navigation in complex 3d environments. *arXiv preprint arXiv:1901.10915*, 2019.

[19] Christian Mostegel, Andreas Wendel, and Horst Bischof. Active monocular localization: Towards autonomous monocular exploration for multirotor mavs. In *ICRA*. IEEE, 2014.

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

[21] Kanwal Naveed, Muhammad Latif Anjum, Wajahat Hussain, and Donghwan Lee. Deep introspective SLAM: Deep reinforcement learning based approach to avoid tracking failure in visual SLAM. *Autonomous Robots*, 46(6):705–724, 2022.

[22] Dean A Pomerleau. Alvinn: An autonomous land vehicle in a neural network. Technical report, CARNEGIE-MELLON UNIV PITTSBURGH PA ARTIFICIAL INTELLIGENCE AND PSYCHOLOGY ..., 1989.

[23] Vignesh Prasad, Karmesh Yadav, Rohitashva Singh Saurabh, Swapnil Daga, Nahas Pareekutty, K Madhava Krishna, Balaraman Ravindran, and Brojeshwar Bhowmick. Learning to Prevent Monocular SLAM Failure using Reinforcement Learning. In *Proceedings of the 11th Indian Conference on Computer Vision, Graphics and Image Processing*, pages 1–9, 2018.

[24] Sadegh Rabiee and Joydeep Biswas. IVOA: Introspective vision for obstacle avoidance. *arXiv preprint arXiv:1903.01028*, 2019.

[25] Marta Salas, Wajahat Hussain, Alejo Concha, Luis Montano, Javier Civera, and JMM Montiel. Layout aware visual tracking and mapping. In *IROS*. IEEE, 2015.

[26] Manolis Savva, Angel X Chang, Alexey Dosovitskiy, Thomas Funkhouser, and Vladlen Koltun. MINOS: Multimodal indoor simulator for navigation in complex environments. *arXiv preprint arXiv:1712.03931*, 2017.

[27] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, et al. Habitat: A platform for embodied ai research. In *ICCV*, 2019.

[28] Dhruv Mauria Saxena, Vince Kurtz, and Martial Hebert. Learning robust failure response for autonomous vision based flight. In *ICRA*. IEEE, 2017.

[29] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A Benchmark for the Evaluation of RGB-D SLAM Systems. In *IROS*. IEEE, 2012.

[30] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *NIPS*, 2017.

[31] Wenshan Wang, Delong Zhu, Xiangwei Wang, Yaoyu Hu, Yuheng Qiu, Chen Wang, Yafei Hu, Ashish Kapoor, and Sebastian Scherer. Tartanair: A dataset to push the limits of visual slam. In *IROS*. IEEE, 2020.