Viewpoint Push Planning for Mapping of Unknown Confined Spaces

Nils Dengler Sicong Pan Vamsi Kalagaturu Rohit Menon Murad Dawood Maren Bennewitz

***Abstract*— Viewpoint planning is an important task in any application where objects or scenes need to be viewed from different angles to achieve sufficient coverage. The mapping of confined spaces such as shelves is an especially challenging task since objects occlude each other and the scene can only be observed from the front, thus with limited possible viewpoints. In this paper, we propose a deep reinforcement learning framework that generates promising views aiming at reducing the map entropy. Additionally, the pipeline extends standard viewpoint planning by predicting adequate minimally invasive push actions to uncover occluded objects and increase the visible space. Using a 2.5D occupancy height map as state representation that can be efficiently updated, our system decides whether to plan a new viewpoint or perform a push. To learn feasible pushes, we use a neural network to sample push candidates on the map and have human experts manually label them to indicate whether the sampled push is a good action to perform. As simulated and real-world experimental results with a robotic arm show, our system is able to significantly increase the mapped space compared to different baselines, while the executed push actions highly benefit the viewpoint planner with only minor changes to the object configuration.**

I. Introduction

Viewpoint planning (VPP) is crucial in robotic applications for understanding the environment, such as identifying occluded objects [1], estimating fruit yields [2], or determining optimal grasping positions [3]. However, VPP performance varies with the scenario, as environments can constrain actions and reduce possible coverage. Especially, confined spaces, such as shelves or refrigerators, limit robots to only a few viewpoints, making mapping difficult when multiple objects are present.

In particular, without the top-down view from above a scene the identification and mapping of objects can be difficult, especially for objects hidden behind others or closer to the back of a shelf. While manipulation actions can unveil such occluded space, human preferences in indoor environment arrangement must be considered [4]. Therefore, minimally invasive pushing actions are preferred to avoid unwanted layout changes. Additionally, identifying promising non-prehensile push positions is easier than grasping poses for object rearrangement, due to the limited number of possible actions.

All authors are with the Humanoid Robots Lab, University of Bonn, Germany. Murad Dawood and Maren Bennewitz are additionally with the Lamarr Institute for Machine Learning and Artificial Intelligence, Germany. This work has partially been funded by the European Commission under grant agreement number 964854 –RePAIR – H2020-FETOPEN-2018-2020 and by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy, EXC-2070 – 390732324 – Phenorob and under the grant number BE 4420/4-1 (FOR 5351: KI-FOR Automation and Artificial Intelligence for Monitoring and Decision Making in Horticultural Crops, AID4Crops).

Fig. 1: Motivation of our approach. The robot aims to create a complete 2.5D occupancy height map of the shelf board, which is challenging due to large objects blocking the view. However, by manipulating the right box, the information value of the viewpoint increases significantly due to the newly uncovered object (white circle).

Fig.1 shows an example scenario where a packed shelf board needs to be mapped. As can be seen, a short minimally invasive push is sufficient to move one of the larger boxes, exposing the space behind it and unveil the previously occluded can.

In this paper, we present a novel framework called viewpoint push planning, which uses deep reinforcement learning (DRL) to perform VPP in confined scenarios and predict suitable non-prehensile push actions to improve shelf coverage. As map representation our framework generates a 2.5D occupancy height map from 3D point clouds obtained from an RGB-D image. Furthermore, We use a push prediction network trained in a supervised manner to output the best pushing action. An action selection method is used to decide whether VPP or a pushing action is best given the current situation. DRL can learn the best viewpoints from experience and generalize them to different configurations, avoiding unnecessary push actions that alter the environment too much.

Regarding related work, we differ from Zeng *et al.* [5], since we use no computationally expensive octomap ray casting. Besides, previous work in the area of interaction with objects in confined spaces has concentrated on reasoning about individual objects [6]–[8] or their retrieval [9]–[13] but not on the complete apriori mapping of the shelf. Furthermore we differ from Miao *et al.* [14], since we perform no time intensive 3d reconstruction of the objects.

The experimental evaluation of our approach shows that our learned VPP significantly reduces map entropy compared to two baselines and two VPP systems from the literature [15], [16]. Our complete framework, including push predic-

tions, further reduces entropy. The main contributions of our work are as follows:

- A 2.5D heightmap representation for confined space mapping
- A deep reinforcement learning-based viewpoint planning framework with pushing actions
- A qualitative and quantitative evaluation on a real robot and in simulation, including the usability of our map representation for object reasoning and

II. Our Approach

In the following, we first give an overview of our system and introduce our framework in detail afterwards.

A. Overview

We consider the problem of mapping a confined shelf environment using a stationary robotic arm to capture as much of the shelf board as possible, despite occlusions from various objects. We consider the following problem. Our system receives a depth map from an RGB-D camera on the arm's end-effector and back-projects it into a 2.5D occupancy height map, as shown in Fig. 2. To facilitate faster convergence of the DRL-agent for viewpoint planning, we use a variational auto-encoder [17] to encode the height map into a 32-dimensional latent space. Furthermore, we sample push proposals to predict the best minimally invasive push action for unveiling additional space on the height map and to address the occlusion issue. Depending on the change in mapped space, an action selection method decides whether the best action is a push, moving to a new viewpoint, or if the mapping process is complete.

B. Environment Representation

We use a 2.5D representation of the environment since it can be efficiently updated. From a depth image, we compute the corresponding point cloud and orthographically back-project it into the top-down view of the scene to calculate a 2.5D occupancy height map as shown in Fig.3. To update the map, we implemented log-odds for occupancy updates [18] and update only the cells in the current field of view. For the height component of the map, we use the maximum measured height value for each cell.

In case grasping requires a more accurate representation of the objects, our map can be used to plan a path to the object, while the actual grasping action can be performed on a local higher dimensional representation. As previous approaches have shown, 2.5D representations can be used to compute grasping points and manipulate objects [19]–[21].

C. Viewpoint Planning (VPP)

In this paper, we use DRL to generate the next best view for VPP, aiming to map the environment as completely as possible. We generate three fixed viewpoints at the beginning of each episode, as in [22], to generate an initial map with at least 50% coverage, reducing the need for the agent to learn initial viewpoints that stay the same for each configuration. To evaluate the agent's quality, we use entropy $h(M)$, information gain $IG(M)$, and motion cost $c(p_t)$ as defined below:

$$h(M) = \frac{m_u^t}{cells_M} \tag{1}$$

$$IG(M) = \frac{m_u^{t-1} - m_u^t}{m_u^{t-1}} \tag{2}$$

$$c(p_t) = d(p_{t-1}, p_t) \tag{3}$$

$h(M)$ is the ratio of unknown cells m_u^t to total cells of map M at time t. We consider cells with occupancy probability $occ > 0.5 + \tau_{unknown}$ as occupied, $occ < 0.5 - \tau_{unknown}$ as free, and unknown otherwise. $IG(M)$ indicates the percentage decrease of unknown cells m_u from time step $t-1$ to t. $c(p_t)$ is the Euclidean distance between two viewpoints p_{t-1} and p_t.

During training, an episode ends if the change of $h(M)$ for the last three viewpoints is below threshold τ_{single} and the sum of changes is below threshold τ_{sum}. We terminate the episode if any part of the arm has contact with an object or an object drops in or out of the shelf.

Next, we define the actions, observations, and the reward function of our agent.

*1) **Action Space**:* As action we use the 5D Cartesian pose $(x, y, z, pitch, yaw)$ of the camera on the end effector.

*2) **Observation Space**:* The observation space consists of the latent space of the state representation (32), the last action in Cartesian coordinates (5), the information gain (1) the agent receives from this viewpoint, as well as the motion cost (1) to move there. Furthermore, we also included the collision indication (1) and the center point of the largest unknown area in the map (3) to give the agent an indication of a promising region of interest.

*3) **Reward**:* We keep the reward function as simple as possible to aid the training. The reward consists of two parts:

$$r_{sparse} = \begin{cases} -25, & \text{if collision or drop from shelf} \\ 0, & \text{else} \end{cases} \tag{4}$$

$$r_{cont} = \quad \alpha * -c(p_t) + \beta * IG(M) \tag{5}$$

r_{sparse} penalizes each contact with the object. Since the goal is to map the environment in as few time steps as possible, we penalize each step in r_{cont} according to the move cost $c(p_t)$. The higher $IG(M)$, the more space has been unveiled during one step, which is positively rewarded. α and β are used to weight closer viewpoints with the effectiveness of the viewpoint. Finally, the total reward can be expressed as $r = r_{sparse} + r_{cont}$.

D. Push Prediction

Standard viewpoint planning performs actions aiming at minimizing the entropy without altering the environment. In contrast, interactive perception [23] as proposed in this paper performs additional actions to change the current object configuration to support perception and coverage of the environment. However, any manipulation action constitutes a trade-off between reducing the entropy of the environment and disrupting the current configuration of the scene. Thus,

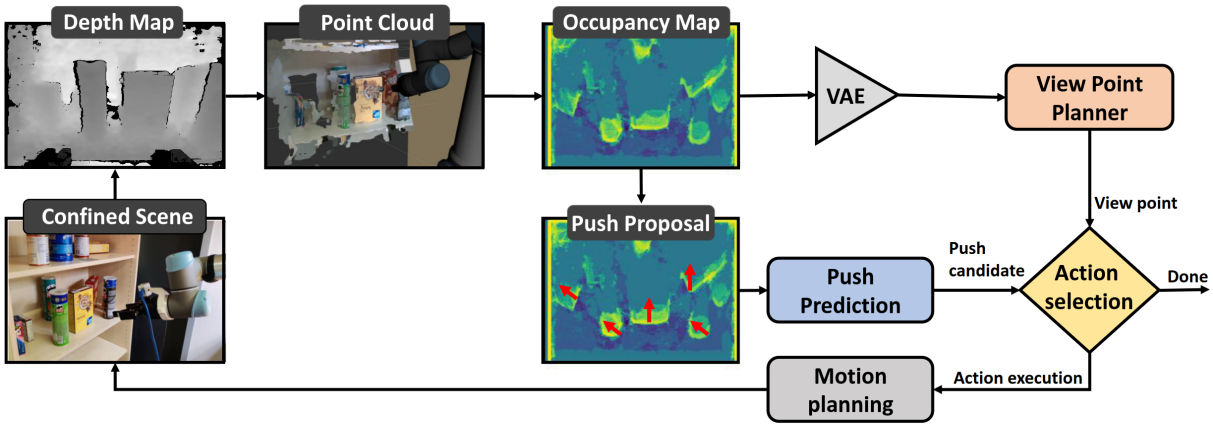
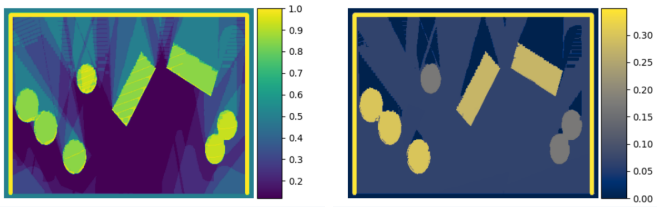


Fig. 2: Framework Overview: Our system converts a depth map from an RGB-D camera into a point cloud. This cloud is backprojected into a 2.5D occupancy height map and fed into a variational auto-encoder to obtain the latent space. Push proposals are then sampled from this space. The latent space is used for the viewpoint planner to select the next best view, while the push proposals predict the best push candidate. An action selection method decides whether to map or perform a predicted push or next viewpoint action.



(a) Occupancy map (b) Height map

Fig. 3: Example of 2.5D state representation after mapping. The left image displays occupancy probability, while the right image shows cell height. Brighter colors indicate higher values.

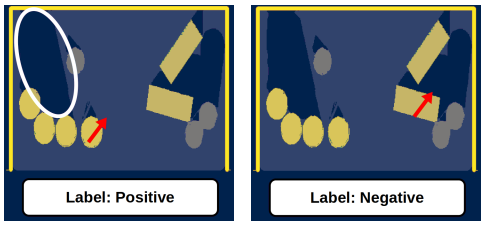


Fig. 4: Two object configurations with a sampled push candidate for each (red arrow). Brighter colors indicate higher object height. Unknown space is marked in dark blue. Labels indicate expert evaluation of push candidate quality in terms of invasiveness and additional uncovered space.

we focus on minimally invasive actions to achieve a balance between these competing requirements. In particular, we use non-prehensile push actions since they are easy to compute and carry out in confined spaces, while other types of manipulation actions would increase the computational efforts due to the constraints of the environment.

1) Learning of the Push Prediction Network: To predict the best push to aid the VPP, we use an approach similar to Eitel et al. [24] to learn a prediction network that outputs the best push of the current scene, modified to predict the best minimally invasive push candidate that increases the mapped area while balancing the trade-off mentioned above. Therefore, three human experts labeled 5,000 simulated map configurations and an additional 850 configurations of the real-world scene with binary labels "positive" or "negative" for the starting point and direction of the push. Example images including the labels are shown in Fig. 4. Note that we used manual rather than automatically generated labels because without computationally expensive physics simulations, it is difficult to automatically calculate the effect that pushing of an object may have on the object itself as well as on its neighbours. Instead, we rely on human expertise.

2) Generation of Push Proposals: To generate push proposals, we ray cast from three positions outside the shelf to the back of the shelf, sampling the 3D Cartesian coordinate of the first occupied cell along the ray as a push candidate, along with eight push angles and a predefined push length. Push maps are generated, as input to the network, by translating the starting point to the center of the map and rotating the image. Our prediction network consists of five convolution layers followed by max-pooling and ReLU activation. We train with the labeled push maps using Adam optimizer and learning rate of $1e-4$. In inference, we feed the push proposal maps into the network and execute the push with the highest output of the network..

E. Action Selection

To decide whether another push is needed, we assume the VPP agent and push prediction network are both optimal. Therefore, VPP is always performed until the VPP agent terminates (II-C). If the previous push did not lead to an entropy reduction of $\Delta h(M) > \tau_{push}$, we terminate the mapping process without performing any further pushes.

III. Experimental Evaluation

The experiments aim to demonstrate the improved mapping result of our framework by evaluating entropy reduction, number of steps needed, planning time, and object displacement of our complete pipeline. We compare the performance to different baselines in simulation and real-world experiments. For the simulation, we use Pybullet [25] and for the real-world experiments ROS [26]. For all experiments we use a UR5 with a Robotiq 2f85 gripper and a Realsense

Simulation	Entropy Reduction
Ours	$\mathbf{30.0}\% \pm 0.9\%$
Random	$15.5\% \pm 0.6\%$
RSE [16]	$16.5\% \pm 0.5\%$
GMC [15]	$21.8\% \pm 0.8\%$
Real World	**entropy reduction**
Ours	$\mathbf{26.1}\% \pm 1.0\%$
Random	$20.3\% \pm 1.0\%$

TABLE I: Evaluation of the viewpoint planner: Reduced entropy compared to the initial entropy of the map computed with 3P (three fixed viewpoints). Each metric is shown with its standard error. The experiments were carried out in 100 simulated and 15 real-world scenarios. As can be seen, our approach lead to the highest reduction in entropy in comparison to all baselines. The reduction is significant according to the paired t-test with a p-value of 0.05.

D405 camera mounted on top of the end effector. For the reinforcement learning agent, we use the stable-baselines3 framework [27] with the TQC algorithm [28]. We set the following thresholds for the experiments: $\tau_{unknown} = 0.2$, $\tau_{single} = 0.01$, $\tau_{sum} = 0.05$, and $\tau_{push} = 0.01$. For our experiments we use a shelf with size of 40 cm x 80 cm x 40 cm. The implementation of our learning pipeline is available on Github[1].

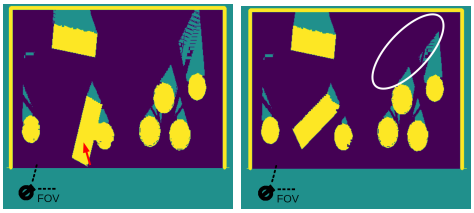
A. Evaluation of the Viewpoint Planner

To evaluate our viewpoint planner individually, we executed the 3-point fixed planner (3P) used for training and evaluated how much our learned planner improves the resulting map. Furthermore, we evaluated our viewpoint planner (VPP) against a random method, a greedy planner by Delmerico *et al.* [16] (RSE), and a global planner by Pan *et al.* [15] (GMC), which we adapted to work in our confined-space multi-object scenario. We compared against RSE to show the limitation of greedy planners, and against GMC to demonstrate the overall benefit of a learned approach over a sample-based method.

For the experiments, we sampled 100 shelf configurations in simulation and 15 in real-world, each with 8 to 10 objects. As can be seen in Tab. I, our agent outperformed all baselines in simulation, significantly according to the paired t-test, with an on average 30% reduced entropy compared to the 3P, while needing 4.94 viewpoints. Furhtermore, it outperforms the random agent in simulation as well as during real-world experiments, confirming its capabilities. RSE performed worse or equal to the random agent, due to its greedy properties, while GMC had an 8% less entropy reduction than our VPP and took 7.29,s per step, whereas our approach planned the next viewpoint in 0.040,s.

B. Evaluation of the Viewpoint Push Planner

We evaluated our complete pipeline by calculating entropy before and after push actions, and comparing a method using randomly selected push candidates to our push prediction network. Each pipeline iteration terminates according to the criteria in Sec. II-E, and we evaluated it on the same 100 simulated environments as in III-A. In Tab. II, both pushing methods reduced the map entropy resulting from VPP alone, with an average of 3.5 and 3.7 pipeline iterations. Our method improved over the random method by reducing the likelihood of object drops and large object displacements. In 100 simulated trials, our learned pipeline reduced the number of object drops by 80% and reduced the displacement by 29.4% compared to the random method. We tested different push lengths and found that 5 cm is optimal in terms of the trade-off between entropy reduction and configuration disruption. Fig.5 shows a qualitative comparison of a map before and after a push action. On average, our method takes 0.31 s to sample the push candidates, while the network takes 0.1 s for inference. To summarize, our pipeline aids VPP while maintaining the overall structure of the scene, and can be easily transferred to a real-robot system without loss of performance, as demonstrated in the accompanying video[2].



(a) Before pushing (b) After pushing

Fig. 5: Qualitative example of an occupancy map update in simulation after a minimally invasive push. (a) shows the map before the push (green indicates unknown space), with the arrow indicating the planned push action. (b) shows the map after the push, where the cells in the marked area became visible due to the push.

Simulation	Entropy Reduction	Displacement	#iterations	#object drops
Learned Push	$27.62\% \pm 2.2$	$^{*}\mathbf{0.24\,cm} \pm 0.03$	3.5 ± 0.24	**4**
Random Push	$\mathbf{37.05}\% \pm 2.6$	$0.34\,cm \pm 0.03$	3.7 ± 0.21	21

TABLE II: Evaluation of the viewpoint push planner pipeline: Reduced entropy, object displacement, pipeline iterations, and drops shown with standard error. Experiments conducted in 100 simulated scenarios. Complete pipeline reduced entropy further. Trained push proposal network led to less object displacement compared to random selection method. Paired t-test shows significant difference in object displacement (p=0.05).

IV. Conclusion

We presented a novel pipeline for interactive viewpoint planning in confined spaces, which was tested in simulated and real-world scenarios using a robotic arm with an RGB-D camera. Our system outperformed several baselines and demonstrated its ability to to increase the mapped space, due to minimally invasive push actions. Furthermore, we evaluated the real-time performance of our system and made our code available on Github[1].

References

[1] M. Li, C. Weber, M. Kerzel, J. H. Lee, Z. Zeng, Z. Liu, and S. Wermter, "Robotic occlusion reasoning for efficient object existence

[1] https://github.com/NilsDengler/view-point-pushing

[2] https://youtu.be/6C3Q2UFKq2Q

prediction," in *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2021.

[2] T. Zaenker, C. Smitt, C. McCool, and M. Bennewitz, "Viewpoint planning for fruit size and position estimation," in *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2021.

[3] M. Breyer, L. Ott, R. Siegwart, and J. J. Chung, "Closed-loop next-best-view planning for target-driven grasping," in *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2022.

[4] N. Abdo, C. Stachniss, L. Spinello, and W. Burgard, "Robot, organize my shelves! tidying up objects by predicting user preferences," in *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*. IEEE, 2015.

[5] X. Zeng, T. Zaenker, and M. Bennewitz, "Deep reinforcement learning for next-best-view planning in agricultural applications," in *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2022.

[6] H. Huang, M. Dominguez-Kuhne, V. Satish, M. Danielczuk, K. Sanders, J. Ichnowski, A. Lee, A. Angelova, V. Vanhoucke, and K. Goldberg, "Mechanical search on shelves using lateral access x-ray," in *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2021.

[7] H. Huang, M. Danielczuk, C. M. Kim, L. Fu, Z. Tam, J. Ichnowski, A. Angelova, B. Ichter, and K. Goldberg, "Mechanical search on shelves using a novel "bluction" tool," in *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2022.

[8] H. Huang, L. Fu, M. Danielczuk, C. M. Kim, Z. Tam, J. Ichnowski, A. Angelova, B. Ichter, and K. Goldberg, "Mechanical search on shelves with efficient stacking and destacking of objects," *arXiv preprint arXiv:2207.02347*, 2022.

[9] W. Bejjani, W. C. Agboh, M. R. Dogar, and M. Leonetti, "Occlusion-aware search for object retrieval in clutter," in *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2021.

[10] M. Kang, H. Kee, J. Kim, and S. Oh, "Grasp planning for occluded objects in a confined space with lateral view using monte carlo tree search," in *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2022.

[11] J. Ahn, C. Kim, and C. Nam, "Coordination of two robotic manipulators for object retrieval in clutter," in *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2022.

[12] C. Nam, J. Lee, S. H. Cheong, B. Y. Cho, and C. Kim, "Fast and resilient manipulation planning for target retrieval in clutter," in *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 2020.

[13] J. K. Li, D. Hsu, and W. S. Lee, "Act to see and see to act: Pomdp planning for objects search in clutter," in *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2016.

[14] Y. Miao, R. Wang, and K. Bekris, "Safe, occlusion-aware manipulation for online object reconstruction in confined spaces," *Proc. of the Intl. Symposium on Robotic Research (ISRR)*, 2022.

[15] S. Pan and H. Wei, "A global generalized maximum coverage-based solution to the non-model-based view planning problem for object reconstruction," *Computer Vision and Image Understanding*, 2023.

[16] J. Delmerico, S. Isler, R. Sabzevari, and D. Scaramuzza, "A comparison of volumetric information gain metrics for active 3d object reconstruction," *Autonomous Robots*, 2018.

[17] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[18] H. Moravec and A. Elfes, "High resolution maps from wide angle sonar," in *Proc. of the IEEE Intl. Conf. on Robotics & Automation (ICRA)*, 1985.

[19] A. Zeng, S. Song, S. Welker, J. Lee, A. Rodriguez, and T. Funkhouser, "Learning synergies between pushing and grasping with self-supervised deep reinforcement learning," in *Proc. of the IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*.

[20] M. Q. Mohammed, K. L. Chung, and C. S. Chyi, "Pick and place objects in a cluttered scene using deep reinforcement learning," *Intl. Journal of Mechanical Engineering and Mechatronics (IJMEM)*, 2020.

[21] A. Zeng, S. Song, J. Lee, A. Rodriguez, and T. Funkhouser, "Tossingbot: Learning to throw arbitrary objects with residual physics," *IEEE Transactions on Robotics*, 2020.

[22] B. Hepp, M. Nießner, and O. Hilliges, "Plan3d: Viewpoint and trajectory optimization for aerial multi-view stereo reconstruction," *ACM Transactions on Graphics*, 2018.

[23] J. Bohg, K. Hausman, B. Sankaran, O. Brock, D. Kragic, S. Schaal, and G. S. Sukhatme, "Interactive perception: Leveraging action in perception and perception in action," *IEEE Transactions on Robotics*, 2017.

[24] A. Eitel, N. Hauff, and W. Burgard, "Learning to singulate objects using a push proposal network," in *Proc. of the Intl. Symposium on Robotic Research (ISRR)*, 2020.

[25] E. Coumans and Y. Bai, "Pybullet, a python module for physics simulation for games, robotics and machine learning," http://pybullet.org, 2016–2023.

[26] Stanford Artificial Intelligence Laboratory et al., "Robotic operating system." [Online]. Available: https://www.ros.org

[27] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, "Stable-baselines3: Reliable reinforcement learning implementations," *Journal of Machine Learning Research*, 2021.

[28] A. Kuznetsov, P. Shvechikov, A. Grishin, and D. Vetrov, "Controlling overestimation bias with truncated mixture of continuous distributional quantile critics," in *International Conference on Machine Learning (ICML)*, 2020.