The Radiance of Neural Fields: Democratizing Photorealistic and Dynamic Robotic Simulation

Georgina Nuthall[1], Richard Bowden[2] and Oscar Mendez[3]

***Abstract*— As robots increasingly coexist with humans, they must navigate complex, dynamic environments rich in visual information and implicit social dynamics, like when to yield or move through crowds. Addressing these challenges requires significant advances in vision-based sensing and a deeper understanding of socio-dynamic factors, particularly in tasks like navigation. To facilitate this, robotics researchers need advanced simulation platforms offering dynamic, photorealistic environments with realistic actors. Unfortunately, most existing simulators fall short, prioritizing geometric accuracy over visual fidelity, and employing unrealistic agents with fixed trajectories and low-quality visuals. To overcome these limitations, we developed a simulator that incorporates three essential elements: (1) photorealistic neural rendering of environments, (2) neurally animated human entities with behaviour management, and (3) an ego-centric robotic agent providing multi-sensor output. By utilizing advanced neural rendering techniques in a dual-NeRF simulator, our system produces high-fidelity, photorealistic renderings of both environments and human entities. Additionally, it integrates a state-of-the-art Social Force Model (SoFM) to model dynamic human-human and human-robot interactions, creating the first photorealistic and accessible human-robot simulation system powered by neural rendering. The code for the simulator is available at `https://gitlab.surrey.ac.uk/gn00217/radiance-of-neural-fields-simulator/`.**

I. INTRODUCTION

Robotic simulators have long been indispensable tools for design and testing, allowing researchers and engineers to safely experiment with robotic tasks without risking harm to hardware or the environment. This capability is vital across many domains but becomes especially crucial for human-centred tasks where safety and reliability are paramount. Despite their importance, however, existing simulators face significant limitations: they are often costly, lack realism, and fail to account for human factors. These shortcomings hinder the development of accurate and safe testing platforms, particularly in scenarios involving human-robot interaction.

Modelling simulation environments that closely replicate physical-world settings is both challenging and time-consuming. Traditionally, these environments have been manually created by computer-aided design specialists, a process that demands significant effort. Alternatively, 3D reconstruction techniques, which generate models from 2D images, can be used, but these often result in noisy and incomplete models that lack the fidelity required for realistic simulations. This is especially problematic when testing computer vision approaches to robotics. Widely used simulators like Gazebo [34], CARLA [11], and NVIDIA Isaac [29] either limit users to predefined environments or rely on custom meshes, further restricting flexibility. Moreover, these simulators often neglect photorealism and realistic modelling of human interactions—between people or robots—limiting their effectiveness in conducting reliable research.

To overcome these challenges, we introduce a novel simulation platform designed to democratize advanced robotic research. This platform models complex dynamics in photorealistic environments with integrated multi-sensor capabilities, enabling accurate testing of robotic tasks, including those in populated environments. By emphasizing realism, accessibility, and ease of use, it reduces the costs of developing full-scale digital twins and supports cutting-edge robotics research, particularly in computer vision. Our platform features three key features: (1) Photorealistic Neural Rendering using Neural Radiance Fields (NeRFs) to create immersive, high-fidelity environments; (2) Animated Human Entities with Behavior Management for lifelike human-robot interactions; and (3) An Ego-Centric Robotic Agent with Multi-Sensor Integration that can perform tasks like Simultaneous Localization and Mapping. These features enable researchers to generate realistic, localized environments with accurate sensor data and human behaviours for comprehensive testing.

Recent advances in 3D scene representation, particularly neural rendering, have highlighted NeRFs for their photorealistic output and implicit learning of depth and occupancy. Although some studies have explored the use of NeRFs in simulation [14], [5], their applications have been limited. To our knowledge, we are the first to present a complete human-robot simulator powered by neural rendering. In this work, we integrate neural rendering into a simulation pipeline to offer a novel, efficient approach to simulating localized environments for robotics research. The main contributions of our paper are the following:

1) We experiment with low computational cost methods to obtain high-fidelity representations of indoor scenes using the latest approaches in Neural Radiance Fields.
2) We provide a method for photorealistic rendering of human agents, along with customisable behaviour and reactions to the robotic agents in the scene.
3) We provide an indoor dataset of multiple sequences captured by a Boston Dynamics advanced quadruped Spot; which we used to benchmark our simulation results.

[1] Georgina Nuthall is with the Center of Vision, Speech and Signal Processing, University of Surrey, Guildford, UK. galcoladonuthall@surrey.ac.uk

[2] Richard Bowden is with the Center of Vision, Speech and Signal Processing, University of Surrey, Guildford, UK. r.bowden@surrey.ac.uk

[3] Oscar Mendez is with Locus Robotics, Wilmington, Massachusetts USA. omendez@locusrobotics.com

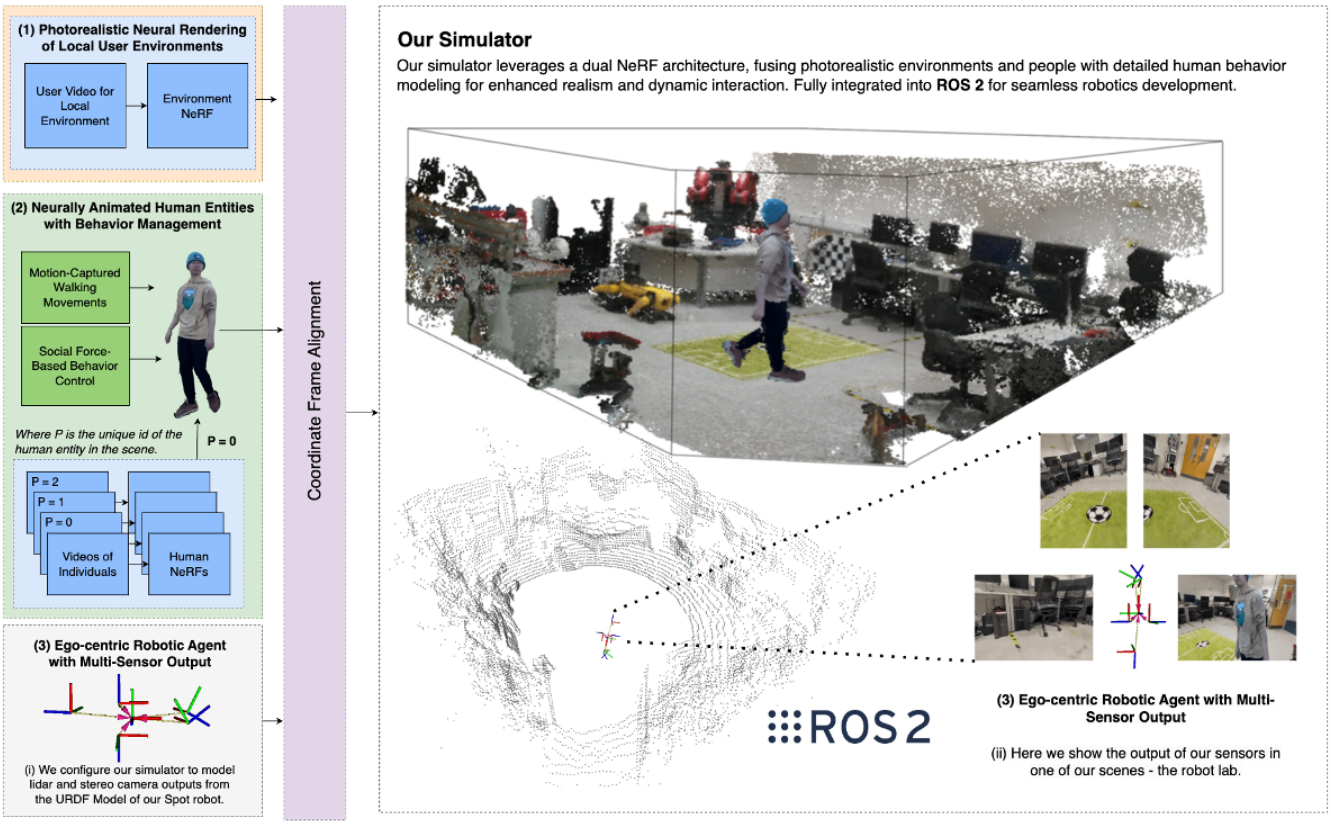

Fig. 1. System Overview – The figure illustrates the components of our simulation pipeline. (1) Shows the training of a local environment based on user-provided video footage. (2) Depicts the integration of human entity representations, trained using motion capture data and the social force model, for dynamic human behaviour simulation. (3) (i) Details the integration of a predefined robot, exemplified here by Spot's URDF, into the simulator. All elements are aligned within the same coordinate frame to enable accurate multi-sensor output rendering (3) (ii). The bottom left shows the simulated LiDAR output, while the bottom right presents the simulated RGB stereo output.

II. RELATED WORK

A. Robotic Simulators

Robotic simulators are essential for designing and testing robots safely and cost-effectively in a controlled environment. Simulators have been developed for various applications, such as mobile robotics and manipulation n [22] [26] [35] [40], but realism remains a significant challenge [1]. For example, AIRSIM [37] and CARLA [12] offer photorealistic rendering using Unreal Engine but are limited to autonomous driving and handcrafted environments [8]. Multi-purpose simulators like GAZEBO [22] also struggle with realism and complex environment setup [1], as creating 3D assets manually is time-consuming and error-prone. These errors affect both environmental representation and simulated sensor outputs [30] [31] [17]. Without realistic rendering of the physical test environment, roboticists are often left with the choice between using synthetic and approximate environments or investing in the costly process of digital twinning [29]. In our work, we address this challenge by focusing on developing an accessible, high-fidelity simulation platform for ego-centric robotic mobile agents.

B. Neural Rendering Approaches

Recent advancements in 3D scene representation, particularly through neural rendering techniques, have attracted significant attention due to their photorealistic capabilities. This interest was initially driven by the development of implicit representations using NeRFs [27].

Early implementations of NeRFs boasted high fidelity results but were costly in both their need for computational resources and training time [27]. These costs have been reduced significantly through multi-resolution hash encoding [28] permitting models to be trained in seconds using only a single GPU. In our dual-NeRF simulator, we train custom simulation environments on our background NeRF that use this approach to speed up the waiting time needed to reconstruct the simulated environment.

Research on NeRFs has covered a wide range of topics, including few-shot training [7], [19], [10], editing [50], [45], and semantics [51], [44]. More recently NeRFs has been extended to represent scenes but also humans. ST-NeRF [20] is able to render dynamic humans and change their position based on the time sequence video it was trained on. However, this means it requires videos of specific actions in order to pose the human models. NeuMan [21], on the other hand, is able to render unseen human poses by training the NeRF using a body skeleton and therefore able to leverage motion capture data to dynamically change the position of a person. Building on these advancements, we integrate this sophisticated human modelling technique into our simulator.

C. Robot Navigation in Crowds

There is a growing need in robotics research to develop systems that work seamlessly with humans, especially in assistive and industrial settings. To meet this demand, it is essential to create detailed simulation environments that accurately represent both human behaviour and the interactions of people around robotic agents. Such simulations are crucial for ensuring that robots can navigate and operate safely in crowded environments, thereby reducing the risk of human injury and improving the effectiveness of collaborative robotics. It is therefore important to simulate not only photorealistic humans but also realistic behaviours. PedSim-ROS[1] uses the SoFM [18] to determine crowd movement and uses Gazebo and RVIZ to render humans with simple non-realistic markers. MengeROS[2] [2], although it provides no advanced rendering for crowd simulation, offers more advanced features by allowing different collision-avoidance strategies such as Optimal Reciprocal Collision Avoidance (ORCA) and Pedestrian Velocity Obstacle (PedVO) [4] [9]. Crowdbot[3] and SEAN[4] both boast photorealistic human rendering as they utilize the Unity game engine [16] [41]. Inspired by advancements in NeRF-based methods, which can render photorealistic humans in various poses once trained on specific individuals [21] [48], and the capabilities of HuNavSim[5] [32], which uses the SOFM model to simulate group behaviours and dynamic reactions to robot interactions, we leverage these approaches to enhance our simulator. While HuNavSim effectively models group dynamics and individual behaviours based on emotional responses, it lacks photorealistic human renderings. To address this, we applied photorealistic rendering techniques and utilized a skeleton-conditioned NeRF [21] to achieve greater flexibility and realism in human modelling. By integrating these capabilities, in addition to photorealistic neural rendering of environments and an ego-centric robotic agent providing multi-sensor output, we are able to create the first photorealistic and accessible human-robot simulation system powered by neural rendering. This system offers a high level of realism and interactivity that is crucial for effective testing and development.

III. METHODOLOGY

Central to our simulator is the use of a dual-NeRF photorealistic rendering framework which provides lifelike, localized environments with dynamic human entities. It incorporates neurally animated human entities and manages a wide range of human actions and interactions for adaptive responses. An ego-centric robotic agent with optional sensors provides detailed multi-sensor outputs for accurate testing of interactions with humans and the environment. Together, these elements form a robust, cost-effective platform that balances high realism with accessibility, supporting advanced research in robotics.

A. Neural Rendering of Local Environments

A primary objective of this work is to create a photorealistic simulator that is accessible to the broader robotics community. To achieve this, we utilize a state-of-the-art NeRF framework [39] for modelling local environments. This framework leverages handheld video data captured by users and metrically scaled camera poses, which can be easily acquired using readily available tools such as PolyCam[6].

1) Neural Radiance Fields: In recent years, significant attention has been paid to methods capable of achieving photorealistic view synthesis. NeRF models use a 5D function using neural networks. The five dimensions are a 3D point in space and a 2D viewing direction often represented as the following.

$$F(x, \theta, \phi) \rightarrow (\mathbf{c}, \sigma) \tag{1}$$

where $x \in \mathbb{R}^3$ is the point in space and θ, ϕ represent the azimuthal and polar viewing angles. The radiance field describes the colour $\mathbf{c}$ represented as (r, g, b) and volume density σ for every point and viewing direction of a captured scene. It uses ReLU activations and consists of two branches. One branch is used to estimate the volume density independent of the viewing direction, as the line of sight towards an object should not impact its occupancy. Meanwhile, the other branch of the network is used to estimate the colour dependent on the viewing direction and the 3D point in space, which supports viewing specularities within a scene.

2) Our NeRF: The model utilizes techniques that reduce training times and memory requirements [28], ensuring that our simulator enables quick and efficient environment setup. Furthermore, our method uses a proposal network sampler [3] concentrating samples that provide the most value to the final representation, providing higher quality results. The background model, based on NeRFacto [39], also leverages features from NeRF– [47], NeRF-W [25] and Ref-NeRF [43]. We added further inference approaches to speed up the sensor renderings from our model to support real-time capable robotic simulation (see SECTION III-D).

B. Animated Human Entities with Behavior Management

Our model utilizes two NeRF networks—one for human entities and one for the surrounding scene—integrated within a behaviour management framework controlled by a SoFM pipeline. The human NeRF [21] is responsible for encoding the appearance and geometry of individuals in the environment, with their poses dynamically generated by the SoFM. Meanwhile, the scene NeRF captures the background and environment details, ensuring photorealistic renderings. The training process is sequential, starting with the scene NeRF to establish an accurate environmental representation, followed

[1] https://github.com/srl-freiburg/pedsim_ros
[2] https://github.com/ml-lab-cuny/menge_ros
[3] https://crowdbot.eu/CrowdBot-challenge/
[4] https://sean.interactive-machines.com/
[5] https://github.com/robotics-upo/hunav_sim
[6] https://poly.cam/

by the human NeRF, which is conditioned on the previously trained scene.

The positioning of individuals within the environment is dictated by the SoFM alongside predefined emotional states of the entities [32]. Human actions, such as walking, are simulated through cycling skeletal poses derived from motion capture data, including sequences from the AMASS dataset [24]. By combining neural rendering with human behaviour modelling, our framework offers the ability to simulate realistic human interactions within dynamic, photorealistic environments. This enables advanced robotic task testing in scenarios that closely mimic the complexities of real-world settings. The integration of neural rendering and socio-dynamic modelling provides an essential level of realism, crucial for the development and validation of robotics systems intended to operate in environments rich with human interactions and real-world dynamics.

C. Ego-centric Robot Agent with Multi-Sensor Integration

Our simulator models various ego-centric sensors for robots, including Spot's[7], which features both appearance sensors like stereo cameras and spatial sensors such as lidar.

1) Appearance Sensors: From our model representing the 3D background simulation environment, we are able to generate RGB cameras using volume rendering. Given the position of the camera to be rendered within the simulation environment, we can cast rays through each pixel of an (n by m) image from the camera origin **o**. We use the same sampling method as above and obtain the colour and density of each sample point. The colour of the ray is the weighted average, represented by a transparency α_i of each sample along the ray, the probability that the sample is not impeded i.e. its transmittance T_i and the evaluated colour for the that sample c_i. For each sampled point and viewing direction, a colour can be approximated as:

$$\hat{C}(\mathbf{r}) = \sum_{i=1}^{N} \alpha_i T_i \mathbf{c}_i \tag{2}$$

where

$$\alpha_i = 1 - \exp(\sigma_i \delta i) \tag{3}$$

and

$$T_i = \exp(-\sum_{j=1}^{i-1} \sigma_j \delta_j) \tag{4}$$

The transparency factor of a sample point is the result of an activation function of its estimated density and distance from itself *i* to the unimpeded next sample *i+1* represented by δ_i. Thereby obtaining an RGB value for every pixel in the image allowing the simulator to accurately replicate the output of real-world camera sensors in the robot's perception system.

[7]https://bostondynamics.com/products/spot/

2) Spatial Sensors: We offer users the option to output data from multiple depth sensors. Building on the methodology outlined in SECTION **??**, an approximate depth can be calculated for a single ray by using the accumulated transmittance of samples, which can then generate depth images. This ray-based depth estimation approach can also be extended to simulate LiDAR. In our simulator, users can choose from various LiDAR configurations based on the number of channels and the vertical field of view (FoV). By default, we model a 16-beam LiDAR with a vertical FoV ranging from -15 to 15 degrees. Leveraging the trained environment model, the simulator can render a point cloud by casting rays in a full 360-degree sweep around the LiDAR, with the rays originating from the LiDAR'scentrer **w** and direction **d**

$$\mathbf{d} = (\cos\theta \sin\phi, \sin\theta \sin\phi, \cos\phi)^{\mathsf{T}} \tag{5}$$

Where θ is the horizontal angle, which sweeps around the LiDAR in a full 360-degree circle and ϕ is the vertical angle of the ray which controls the up-and-down direction of the rays. In our case, the 16 beams are spread evenly from -15° to 15°. We cast N rays every 2π/N radians to complete a 360-degree loop within the desired number of samples allowing the simulator to generate a detailed and accurate point cloud that replicates the output of physical-world LiDAR sensors in diverse environments.

D. System Architecture and Components

Our simulator integrates three core components: (1) photo-realistic neural rendering for environment visualization, (2) neurally animated human models with behaviour management, and (3) an ego-centric robotic agent equipped with multi-sensor outputs. FIGURE 1 presents an overview of the system and its integration into ROS2. The simulation process begins with the user providing video data and camera poses to build a local environment model, which is then integrated into the simulator. Human entities are added based on predefined social force parameters, with our human entity models pre-loaded. Accurate interaction between the environment and human NeRFs is ensured by aligning their coordinate frames. This allows for the ego-centric agent sensors to accurately output the state of the dynamic scene.

Rendering humans from the robot's perspective is unnecessary if they are out of view of the robot's cameras. In such cases, we reduce rendering times by excluding these humans. We do this by performing an efficient bounding box calculation to determine if the human is visible to the robot and subsequently render only the necessary rays for the human (see ALGORITHM 1). This approach allows for an optimized simulator.

IV. EXPERIMENTS AND RESULTS

To rigorously evaluate the capabilities and performance of our simulator, a series of experiments were conducted focusing on three critical aspects: photorealistic neural rendering of local scenes, neurally animated human entities, and the ego-centric robotic agent.

Algorithm 1 Visibility Determination via Bounding Box

Input: Bounding box coordinates in human coordinates (Min, Max)
Input: Transformation matrix $\mathbf{T}_{robot \to human}$
Input: Camera intrinsic matrix $\mathbf{K}$
Input: Camera image dimensions (W, H)
For each corner of the bounding box
Transform the corner point to robot coordinates:

$$\mathbf{p}_{robot} = \mathbf{T}_{robot \to human} \cdot \mathbf{p}_{human}$$

Project the point to the camera image plane:

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \mathbf{K} \cdot \begin{pmatrix} X \\ Y \\ Z \end{pmatrix}$$

If $0 \leq u < W$ **and** $0 \leq v < H$
Mark the bounding box as visible
Else
Skip rendering of this human
End If
End For

A. Evaluation of Neural Rendering of Local Scenes

To assess the quality of neural rendering for local scenes, we captured short videos of diverse environments and obtained camera poses using Structure from Motions (SfMs) [36]. As detailed in TABLE I, our approach to neurally rendered environments shows competitive performance across common environments for robotic agents, including a Robot Lab, Kitchen, and Living Room.

Environment	PSNR ↑	SSIM ↑	LPIPS ↓
Robot Lab	23.1	0.78	0.26
Common Room	22.1	0.80	0.25
Kitchen	20.7	0.79	0.29
Living Room	21.8	0.77	0.35
Seminar Room	21.2	0.79	0.33

TABLE I
RGB IMAGE QUALITY FOR MULTIPLE SIMULATED ENVIRONMENTS

B. Evaluation of Neurally Animated Human Entities

FIGURE 2 shows an example of one of the available photorealistic humans in our simulator. The agents can be controlled using poses from a motion capture (or similar system). For most of our work, we use a standard gait, shown in FIGURE 2 which allows the agents to perform realistic motion between the different configured waypoints. However, our work can be articulated with any human skeleton, allowing us to represent the different emotions showcased by the SoFM model, creating dynamic reactions to robots in the scene.

C. Ego-Centric Robotic Agent Performance

To evaluate our simulator's ability to accurately model the sensor outputs of ego-centric robotic agents, we test its performance using the Spot robot by Boston Dynamics. This evaluation focuses on three key aspects: modelling sensor outputs, implementing a vision-based SLAM (Simultaneous Localization and Mapping) method, and performing an object detection task.

Fig. 2. Sample Gait Cycle using Motion Capture

1) Evaluating Simulator Sensor Output: To validate our simulator, we collected data from the Boston Dynamics Spot platform, including RGB, Depth, and LiDAR outputs. We captured data from all five stereo cameras for a 360-degree view and used Graph Navigation for initial position estimates, refined with SfM techniques. The spot was manually controlled to map the environment and then autonomously followed the recorded trajectory. In the simulation, we retraced these trajectories to compare the appearance and spatial sensor outputs of both the simulated and real Spot.

In order to evaluate the performance of our depth estimation, we follow the evaluation criteria of [38]. TABLE II shows results on the depth cameras mounted on the Boston Dynamics Spot robot. FIGURE 3 shows the qualitative results of the depth renders compared to the ground-truth RGB-D estimates from Spot's depth sensors. As can be seen, the data is both accurate and high fidelity, allowing the user to use the depth images as part of a development pipeline.

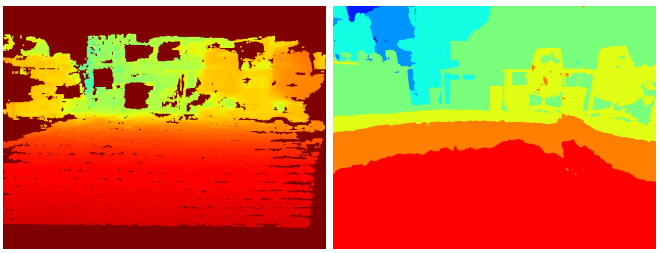

Fig. 3. Depth Image Comparison: (Left) Real Back Depth Image and (Right) Simulated Back Depth Image for Comparison

Spots Depth Cameras	AbsRel (%)	$\delta_{.05}$	$\delta_{.1}$	$\delta_{.25}$	$\delta_{.25^2}$	$\delta_{.25^3}$
left fisheye	0.25	0.26	0.29	0.58	0.86	0.93
right fisheye	0.24	0.258	0.29	0.581	0.88	0.95
back fisheye	0.27	0.24	0.28	0.56	0.84	0.92
frontleft fisheye	0.6	0.215	0.23	0.3	0.698	0.74
frontright fisheye	0.67	0.21	0.21	0.31	0.63	0.7

TABLE II
DEPTH ERROR FOR DIFFERENT SPOT ROBOT SIMULATED CAMERAS

Fig. 4. (Left) Front-Left and (Right) Front-Right Simulated Camera Renders

Robot Lab RGB Cameras	PSNR ↑	SSIM ↑	LPIPS ↓
left fisheye	9.5	0.28	0.55
right fisheye	9.37	0.28	0.59
back fisheye	11.8	0.49	0.42
frontleft fisheye	10.1	0.34	0.53
frontright fisheye	10.9	0.34	0.53

TABLE III

RGB IMAGE QUALITY FOR SIMULATED CAMERAS

Furthermore, we evaluate the RGB stereo camera output from our simulated Spot using the same metrics as outlined in SECTION IV-A (see TABLE III). While some variation in results is expected due to training our background environments with different camera models, this variability does not detract from the overall quality of the simulation. In fact, FIGURE 4 highlights the impressive level of photorealism achieved within our custom NeRF-based simulation environments. This not only underscores the robustness of our approach but also showcases the versatility of our rendering technique, which consistently produces realistic and visually compelling outputs across varying conditions.

2) SLAM Performance Analysis: The vision-based SLAM method is essential for robots navigating dynamic environments, enabling them to construct and update maps while tracking their own position. This capability is particularly important for robots operating in complex, unstructured spaces, such as those found in assistive applications.

We ran ORBSLAM3 [6] on matching trajectories of RGB data from each of Spot's five cameras, both simulated and real. The results showed that the median average trajectory error for Spot's real cameras was 0.14 meters, compared to 0.24 meters in our simulated environment. This indicates a difference of just 0.1 meters between the simulated and real-world testing, highlighting the close alignment between our simulation and actual performance. In further ablations we found that we could enhance the accuracy by incorporating a proportion of the real-world camera data into the training of our simulation background environment, underscoring the potential for continued improvement and refinement of our simulation pipeline.

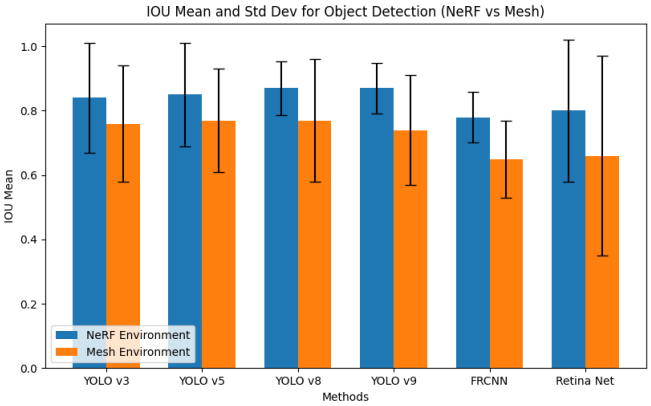

Fig. 5. Object Detection Evaluation – Comparison of object detection performance across simulated environments and 3D mesh reconstructions

3) Human-Robot Task: Object Detection: Object detection is crucial for assistive robotics, as robots must identify and interact with objects to support users effectively. Evaluating our simulator's performance ensures it can model real-world challenges, supporting computer vision-based tasks like item retrieval, obstacle avoidance, and daily assistance. To evaluate our simulator's object detection capabilities, we test on 50 diverse indoor environments from the ScanNet++ [49] dataset. We compare object detection performance on multiple models [33], [13], [42], [46], [15], [23] by analyzing RGB images from both the simulated environment and a 3D mesh reconstruction. By comparing bounding boxes from images in both environments, we find that the neurally rendered scenes consistently achieve higher Intersection over Union (IoU) and show less variation across scenes compared to the 3D mesh reconstruction (see FIGURE 5). This demonstrates that NeRF-based simulations offer superior performance for object detection tasks in robotics.

V. CONCLUSIONS

In this paper, we have demonstrated the capabilities of using Neural Radiance Fields as a robotic simulator that provides multi-sensor readings to validate robotic tasks. High-fidelity simulation with implicitly learnt sensor measurements will be critical for advancing the field of robotics, as it allows researchers and developers to test and refine their algorithms in a safe and controlled environment that is not only representative of the real world, but that allows learning-based vision approaches to be trained in simulation.

Our simulator provides a strong foundation for robotics research, and future enhancements—such as integrating a robust physics engine for realistic interactions and enabling real-time capability via Gaussian splatting will further expand its potential.

REFERENCES

[1] Afsoon Afzal, Deborah S. Katz, Claire Le Goues, and Christopher S. Timperley. A study on the challenges of using robotics simulators for testing. 4 2020.

[2] Anoop Aroor, Susan L. Epstein, and Raj Korpan. Mengeros: a crowd simulation tool for autonomous robot navigation. *AAAI Fall Symposium - Technical Report*, FS-17-01 - FS-17-05:123–125, 1 2018.

[3] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. 2022-June:5460–5469, 11 2021.

[4] Jur Van Den Berg, Stephen J. Guy, Ming Lin, and Dinesh Manocha. Reciprocal n-body collision avoidance. *Springer Tracts in Advanced Robotics*, 70:3–19, 2011.

[5] Arunkumar Byravan, Jan Humplik, Leonard Hasenclever, Arthur Brussee, Francesco Nori, Tuomas Haarnoja, Ben Moran, Steven Bohez, Fereshteh Sadeghi, Bojan Vujatovic, and Nicolas Heess Deepmind. Nerf2real: Sim2real transfer of vision-guided bipedal motion skills using neural radiance fields. 10 2022.

[6] Carlos Campos, Richard Elvira, Juan J.Gomez Rodriguez, Jose M.M. Montiel, and Juan D. Tardos. Orb-slam3: An accurate open-source library for visual, visual-inertial, and multimap slam. *IEEE Transactions on Robotics*, 37:1874–1890, 12 2021.

[7] Anpei Chen, Zexiang Xu, Fuqiang Zhao, Xiaoshuai Zhang, Fanbo Xiang, Jingyi Yu, and Hao Su. Mvsnerf: Fast generalizable radiance field reconstruction from multi-view stereo, 2021.

[8] Jack Collins, Shelvin Chand, Anthony Vanderkop, and David Howard. A review of physics simulators for robotic applications. *IEEE Access*, 9:51416–51431, 2021.

[9] Sean Curtis and Dinesh Manocha. Pedestrian simulation using geometric reasoning in velocity space. *Pedestrian and Evacuation Dynamics 2012*, pages 875–890, 2014.

[10] Kangle Deng, Andrew Liu, Jun-Yan Zhu, and Deva Ramanan. Depth-supervised nerf: Fewer views and faster training for free, 2022.

[11] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. Carla: An open urban driving simulator. pages 1–16. PMLR, 10 2017.

[12] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In Sergey Levine, Vincent Vanhoucke, and Ken Goldberg, editors, *Proceedings of the 1st Annual Conference on Robot Learning*, volume 78 of *Proceedings of Machine Learning Research*, pages 1–16. PMLR, 13–15 Nov 2017.

[13] Glenn Jocher et. al. ultralytics/yolov5: v6.0 - YOLOv5n 'Nano' models, Roboflow integration, TensorFlow export, OpenCV DNN support, October 2021.

[14] Yunhao Ge, Harkirat Behl, Jiashu Xu, Suriya Gunasekar, Neel Joshi, Yale Song, Xin Wang, Laurent Itti, and Vibhav Vineet. Neural-sim: Learning to generate training data with nerf. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 13683 LNCS:477–493, 2022.

[15] Ross Girshick. Fast r-cnn. In *International Conference on Computer Vision (ICCV)*, 2015.

[16] Fabien Grzeskowiak, David Gonon, Daniel Dugas, Diego Paez-Granados, Jen Jen Chung, Juan Nieto, Roland Siegwart, Aude Billard, Marie Babel, and Julien Pettré. Crowd against the machine: A simulation-based benchmark tool to evaluate and compare robot capabilities to navigate a human crowd. *Proceedings - IEEE International Conference on Robotics and Automation*, 2021-May:3879–3885, 4 2021.

[17] Michael Gschwandtner, Roland Kwitt, Andreas Uhl, and Wolfgang Pree. Blensor: Blender sensor simulation toolbox. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 6939 LNCS:199–208, 2011.

[18] Dirk Helbing and Peter Molnar. Social force model for pedestrian dynamics. *Physical Review E*, 51:4282–4286, 5 1998.

[19] Ajay Jain, Matthew Tancik, and Pieter Abbeel. Putting nerf on a diet: Semantically consistent few-shot view synthesis, 2021.

[20] Zhang Jiakai, Liu Xinhang, Ye Xinyi, Zhao Fuqiang, Zhang Yanshun, Wu Minye, Zhang Yingliang, Xu Lan, and Yu Jingyi. Editable free-viewpoint video using a layered neural representation. In *ACM SIGGRAPH*, 2021.

[21] Wei Jiang, Kwang Moo Yi, Golnoosh Samei, Oncel Tuzel, and Anurag Ranjan. Neuman: Neural human radiance field from a single video, 2022.

[22] Nathan Koenig and Andrew Howard. Design and use paradigms for gazebo, an open-source multi-robot simulator. *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 3:2149–2154, 2004.

[23] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection, 2018.

[24] Naureen Mahmood, Nima Ghorbani, Nikolaus F. Troje, Gerard Pons-Moll, and Michael J. Black. AMASS: Archive of motion capture as surface shapes. In *International Conference on Computer Vision*, pages 5442–5451, October 2019.

[25] Ricardo Martin-Brualla, Noha Radwan, Mehdi S.M. Sajjadi, Jonathan T. Barron, Alexey Dosovitskiy, and Daniel Duckworth. Nerf in the wild: Neural radiance fields for unconstrained photo collections. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 7206–7215, 8 2020.

[26] Olivier Michel. Cyberbotics ltd. webots™: Professional mobile robot simulation. *International Journal of Advanced Robotic Systems*, 1:39–42, 3 2004.

[27] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf. *Communications of the ACM*, 65:99–106, 12 2021.

[28] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics*, 41:102, 1 2022.

[29] NVIDIA. Isaac platform for robotic. `https://www.nvidia.com/en-gb/deep-learning-ai/industries/robotics/`. (accessed: 29.04.2024).

[30] OpenGL. The opengl programming guide. `http://www.opengl-redbook.com/`. (accessed: 29.04.2024).

[31] Steven G. Parker, James Bigler, Andreas Dietrich, Heiko Friedrich, Jared Hoberock, David Luebke, David McAllister, Morgan McGuire, Keith Morley, Austin Robison, and Martin Stich. Optix. *ACM Transactions on Graphics (TOG)*, 29, 7 2010.

[32] Noé Pérez-Higueras, Roberto Otero, Fernando Caballero, and Luis Merino. Hunavsim: A ros 2 human navigation simulator for benchmarking human-aware robot navigation. 5 2023.

[33] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement, 2018.

[34] Open Robotics. Gazebo. `https://gazebosim.org/home`. (accessed: 29.04.2024).

[35] Eric Rohmer, Surya P.N. Singh, and Marc Freese. V-rep: A versatile and scalable robot simulation framework. *IEEE International Conference on Intelligent Robots and Systems*, pages 1321–1326, 2013.

[36] Johannes L. Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[37] Shital Shah, Debadeepta Dey, Chris Lovett, and Ashish Kapoor. Airsim: High-fidelity visual and physical simulation for autonomous vehicles. *Springer Proceedings in Advanced Robotics*, 5:621–635, 2018.

[38] Jaime Spencer, Chris Russell, Simon Hadfield, and Richard Bowden. Kick back & relax: Learning to reconstruct the world by watching slowtv, 2023.

[39] Matthew Tancik, Ethan Weber, Evonne Ng, Ruilong Li, Brent Yi, Justin Kerr, Terrance Wang, Alexander Kristoffersen, Jake Austin, Kamyar Salahi, Abhik Ahuja, David McAllister, and Angjoo Kanazawa. Nerfstudio: A modular framework for neural radiance field development. 2 2023.

[40] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. *IEEE International Conference on Intelligent Robots and Systems*, pages 5026–5033, 2012.

[41] Nathan Tsoi, Alec Xiang, Peter Yu, Samuel S. Sohn, Greg Schwartz, Subashri Ramesh, Mohamed Hussein, Anjali W. Gupta, Mubbasir Kapadia, and Marynel Vazquez. Sean 2.0: Formalizing and generating social situations for robot navigation. *IEEE Robotics and Automation Letters*, 7:11047–11054, 10 2022.

[42] Rejin Varghese and Sambath M. Yolov8: A novel object detection algorithm with enhanced performance and robustness. In *2024 International Conference on Advances in Data Engineering and Intelligent Computing Systems (ADICS)*, pages 1–6, 2024.

[43] Dor Verbin, Peter Hedman, Ben Mildenhall, Todd Zickler, Jonathan T. Barron, and Pratul P. Srinivasan. Ref-nerf: Structured view-dependent appearance for neural radiance fields. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2022-June:5481–5490, 12 2021.

[44] Suhani Vora, Noha Radwan, Klaus Greff, Henning Meyer, Kyle Genova, Mehdi S. M. Sajjadi, Etienne Pot, Andrea Tagliasacchi, and Daniel Duckworth. Nesf: Neural semantic fields for generalizable semantic segmentation of 3d scenes. 11 2021.

[45] Can Wang, Menglei Chai, Mingming He, Dongdong Chen, and

Jing Liao. Clip-nerf: Text-and-image driven manipulation of neural radiance fields, 2022.

[46] Chien-Yao Wang and Hong-Yuan Mark Liao. Yolov9: Learning what you want to learn using programmable gradient information. 2024.

[47] Zirui Wang, Shangzhe Wu, Weidi Xie, Min Chen, and Victor Adrian Prisacariu. Nerf–: Neural radiance fields without known camera parameters. 2 2021.

[48] Chung-Yi Weng, Brian Curless, Pratul P. Srinivasan, Jonathan T. Barron, and Ira Kemelmacher-Shlizerman. HumanNeRF: Free-viewpoint rendering of moving people from monocular video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16210–16220, June 2022.

[49] Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. Scannet++: A high-fidelity dataset of 3d indoor scenes. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2023.

[50] Yu-Jie Yuan, Yang-Tian Sun, Yu-Kun Lai, Yuewen Ma, Rongfei Jia, and Lin Gao. Nerf-editing: Geometry editing of neural radiance fields, 2022.

[51] Shuaifeng Zhi, Tristan Laidlow, Stefan Leutenegger, and Andrew J. Davison. In-place scene labelling and understanding with implicit scene representation, 2021.