Autonomous Robotic 3D Scanning for Smart Factory Planning

Adam Haroon[a], Anush Lakshman S[a], Micah Mundy[a], and Beiwen Li[a]

[a]Department of Mechanical Engineering, Iowa State University, Ames, Iowa, USA

ABSTRACT

3D scanning is an indispensable tool in industrial applications, enabling the precise digital replication of complex objects and environments to revolutionize crucial tasks in factory planning such as quality control, factory design, and overall efficiency. However, mainstream 3D scanning approaches such as structured light, 3D laser triangulation, and photogrammetry are often limited by their need for human intervention, operation, and oversight, leading to significant potential errors. To address this challenge, this paper proposes a mobile ground robotic scanning system to enable the quick, robust, and precise 3D scanning of large-scale industrial environments for smart factory planning. Through a simulation of our developed ground mobile robotic system, we demonstrate the potential of NVIDIA Isaac Sim to serve as a platform for the efficient development, prototyping, and testing of large-scale industrial robotic 3D scanning frameworks.

Keywords: robotic simulation, 3D scanning, industrial automation, NVIDIA Isaac Sim, large-scale meteorology

1. INTRODUCTION AND RELATED WORK

Industrial automation has witnessed remarkable progress with the integration of advanced technologies, reshaping manufacturing, production, and logistic processes. Central to this evolution is the role of 3D scanning methods, enabling the precise digital replication of complex industrial objects and environments. The 3D reconstructions obtained from these methods enable their applications towards safety-critical factory inspection tasks such as predictive machinery maintenance, equipment quality inspection, and surface health monitoring.[1,2]

However, traditional 3D scanning approaches such as structured light, 3D laser triangulation, and photogrammetry face inherent limitations, particularly in large-scale applications within dynamic settings. These methods often rely heavily on human intervention, whether through handheld devices or stationary scanning systems.[3] The reliance on human-dependent 3D scanning methods introduces constraints in terms of mobility, scalability, and accuracy, which limits their applicability in large-scale industrial environments.[4] Moreover, the need for manual operation in these systems makes them prone to human-induced operation errors, increasing the risk of inaccuracies in the obtained reconstruction and necessitating the need for skilled personnel to effectively operate the 3D scanning system.[1]

The application of mobile robotics offers a transformative solution to these challenges by employing frameworks for 3D scanning, thereby mitigating the limitations imposed by human intervention.[5] These robots enable the fast, precise, and robust scanning of multiple objects and entire large-scale environments without the constraints of being fixed in one place or relying on manual operation. The use of these robots enhances efficiency whilst expanding the scope of 3D scanning applications, particularly in complex industrial landscapes.[6]

Despite these advances, creating or modifying robotic systems for large-scale industrial environments poses significant challenges. The complexity of these environments, coupled with the critical nature of industrial processes, necessitates meticulous design, testing, and validation of robotic systems. Additionally, the conception, design, and development of custom robotic frameworks in particular require significant lead time for robot system design, control policy development, and hardware integration to ensure their safe operation without extensive human supervision.[7] Any errors in the development process or during operation could have severe consequences, including potential harm to personnel and disruption of critical industrial processes.[8]

Send correspondence to Dr. Beiwen Li: E-mail: beiwen@iastate.edu, Telephone: 1 515-294-9226

Simulation emerges as a crucial tool in addressing these challenges. By simulating robotic systems in realistic industrial environments, researchers, engineers, and decision-makers can understand the constraints required to ensure the framework's intended functionality and safety.[9] NVIDIA Isaac Sim has been widely used in prior studies for tasks such as robot manipulation training, navigation, and learning.[10–12] However, its utility for robotic 3D reconstruction has not been illustrated before. This paper aims to demonstrate the potential of NVIDIA Isaac Sim as a foundational framework for the efficient development, prototyping, and testing of robotic 3D scene reconstruction systems. By leveraging the capabilities of NVIDIA Isaac Sim, researchers can explore and refine mobile ground robotic systems for 3D scanning, paving the way for enhanced efficiency, safety, and innovation in industrial automation.

In the subsequent sections, we delve into the methodologies employed, including an overview of NVIDIA Isaac Sim as a robot simulation framework, the development of a simulated robotic system to act as a configurable digital shadow of our already developed physical system. Finally, we demonstrate NVIDIA Isaac Sim's built-in 3D scanning capabilities by reconstructing an entire simulated warehouse environment. These insights shed light on the intricate technologies driving the advancement of robust robotic 3D scanning frameworks for smart factory planning and inspection.

2. METHODS

2.1 NVIDIA Isaac Sim

NVIDIA Isaac Sim is a robotic simulation toolkit part of the NVIDIA Omniverse platform. This simulation framework contains all the essential features required to build robust virtual worlds for robotic system design and experimentation.[13] NVIDIA Isaac Sim provides a variety of pre-defined sensors, robotic system assets, and industrial testing environments that can be applied out-of-the-box.[14] Additionally, the robotics toolkit here provides the functionality to import custom CAD models and assemblies as *.STEP* files, enabling the seamless development, prototyping, and testing of novel robotic system designs. The simulation framework is composed of the following key components.

- **Prim:** The most fundamental unit in Omniverse is known as a prim, an abbreviation for primitive. They are metadata containers, properties (physics, material, drive, etc.), and other prims. Some examples of prims include but are not limited to, cameras, sounds, lights, and meshes.
- **Universal Scene Description (USD):** Originally developed by Pixar Animation Studios, USD or OpenUSD is a scene description that is used for 3D animation.[15] It enables seamless transfer of designs, systems, and environments across various rendering software such as Blender, Unity, etc.
- **XForm:** The location reference to every object in the simulation is known as an XForm.
- **Replicator:** This is the built-in tool for generating and logging synthetic data for post-processing. The main engine behind this tool is the Synthetic Data Generation (SDG) pipeline, which contains the main visual script for recording and storing the data in the required working directory defined by the user. Moreover, the replicator has the ability to record various modalities of data from a single sensor using well-established methods.[16] For a single basic camera asset, the replicator provides options to record 2D image data in the form of frames and 3D point cloud data simultaneously. Additionally, the back-end engine performs various analyses on obtained point cloud data such as semantic and instance segmentation.
- **Action Graph:** A specialized subset within the broader OmniGraph framework, the Action Graph serves as an engine for programmatically manipulating scenes through a graph-based visual scripting interface.[17] Action graphs provide a user-friendly drag-and-drop programming approach to exert control over assets within the environment easily during simulation execution.

Due to the wide range of features provided by this software, we intend to use this software over traditional well-established robotic software such as Gazebo. With the working principles of the software established, we will further investigate the capabilities of the Synthetic Data Generation Pipeline for performing virtual 3D reconstructions of a complex industrial environment.

2.2 Synthetic Data Acquisition

Through the utilization of the synthetic data generation pipeline discussed in Section 2.1, we were able to acquire, visualize, and save a wide variety of synthetic data modalities necessary for the development of a semi-autonomous mobile robot including RGB, point cloud, depth, instance, and semantic information at predefined frame rates during simulation execution. An illustration of the many data modalities able to be acquired simultaneously from a single pre-defined RGBD camera asset is shown in Figure 1.

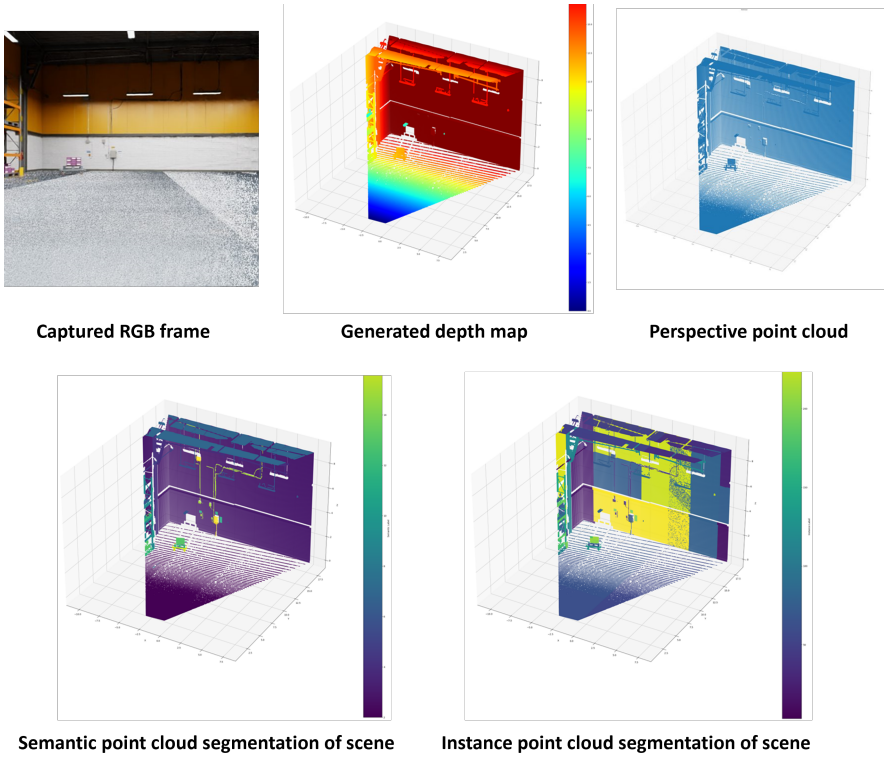

Figure 1. Synthetic data modalities extracted from NVIDIA Isaac Sim.

2.3 Point Cloud Stitching

Point cloud data structures are widely applied across the field of robotics for tasks such as 3D topography-based object detection, analysis, and reconstruction.[18–20] In our experiment, point cloud data is obtained from the RGBD camera asset, via the simulation's SDG pipeline discussed in Section 2.2. The data obtained is utilized for the reconstruction of the factory environment the robot manipulation is performed. The detailed description of this process is provided in Section 3.3.

2.3.1 Registration

Point cloud registration involves merging two or more point clouds to create a unified point cloud that encompasses the features captured in each point cloud. While various methods exist for point cloud registration, each method can generally be categorized as either local or global operations based on whether they require an initial transformation to roughly align the current point cloud with a reference point cloud.[21]

2.3.2 Iterative Closest Point (ICP) Registration

Iterative closest point registration is a local method designed to achieve precise alignment between two given point clouds: the reference point cloud and the source point cloud we intend to align with the reference. As a local registration approach, ICP registration begins with an initial rough alignment of the source point cloud to the reference, based on the point-to-point disparities between them.[22]

The process involves identifying the closest corresponding points between the source and reference point clouds, establishing pairs K of corresponding points p from the source and q from the reference. The root mean squared difference between the correspondence pairs is then used to compute the initial transformation matrix T that facilitates a rough alignment of the source to the reference. This transformation matrix T is subsequently refined by minimizing an objective function $E(t)$ defined over the current set of correspondences K. While several objective functions have been proposed for ICP registration, Open3D employs the following:

$$E(t) = \sum_{(p,q)\in K} \|p - Tq\|^2 \tag{1}$$

This process iteratively refines the transformation matrix until convergence, typically achieved when the change in the objective function between successive iterations is infinitesimal.[22,23]

2.3.3 Global Registration

Global registration methods, on the other hand, aim to align point clouds on a larger scale, often requiring no initial rough alignment. Unlike local methods such as ICP, global registration considers the overall structure of point clouds, enabling the alignment of disjoint or structurally incomplete areas. Techniques such as feature-based matching are commonly employed in global registration to achieve accurate alignment across the entire point cloud.[24] However, this method typically produces less precise alignments due to the lack of an initial rough alignment between the source and reference point cloud at initialization.[25,26]

2.3.4 Down-sampling

In both the aforementioned methods, the volume of data is high which leads to a significant increase in computational time. To tackle this issue, only certain key points, required to obtain accurate structural information, are extracted from the entire point cloud. This technique, termed down-sampling, serves as both a pre-processing and post-processing step in point cloud registration that involves reducing the density of points in a cloud while retaining its essential features. This step is crucial for improving the efficiency and speed of registration algorithms, particularly when dealing with large and dense point clouds.[27] Common down-sampling techniques include random sampling, voxel grid filtering, and bilateral filtering,[28–30] all of which help to maintain the overall structure and characteristics of the point cloud while reducing computational complexity during the registration process. In our experiment, we apply the voxel grid filtering method, with a voxel size of 0.4.

3. SIMULATED ROBOTIC SYSTEM

Based on the existing CAD model of the robot, a simulation environment was setup for navigation and obtaining fringe image data from the simulation environment. The first step in creating a realistic simulation is to define all the joints accurately. Secondly, we devise a keyboard-based control policy, in order to manually manipulate the robot around a preset environment and test the scanning capabilities of the default sensor assets present in the Isaac Sim framework. The details involved in each of these processes are elucidated in the upcoming sections.

3.1 Design to Simulation of Physical Robotic System

To create a precise simulation of the developed robotic system, we first imported the *.STEP* file of the existing CAD model of the physical robotic system. The entire assembly was then split into four key sub-components including the chassis, camera carriage, body, and omniwheels highlighted in Figure 2. With the above set of key sub-components identified, the degrees of freedom for each were defined to fully constrain the entire assembly as a unified rigid body in simulation.

With the robot now defined in simulation in the universal scene description (USD) file format, we then placed the robot in a pre-built warehouse environment asset shown in Figure 2 to test the behavior of the system in a realistic virtual setting.

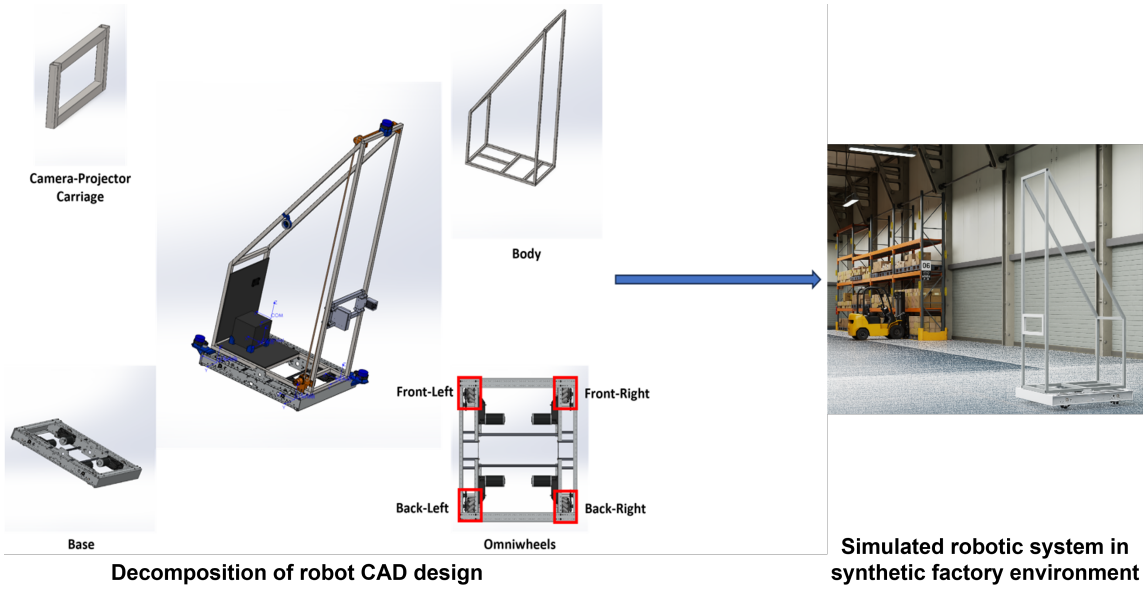

Figure 2. Robot CAD design to simulation process with four key sub-components identified.

3.2 Devised Motion Policy

With the simulated robotic system fully defined, we employed Action Graph, the visual scripting functionally within NVIDIA Isaac Sim discussed in Section 2.1, to simulate the allowed motion of the physical system. Namely, we integrated holonomic drive, a mechanism that allows the robot to turn in place, into the simulated robotic system as illustrated in Figure 4 along with constrained prismatic movement for the camera system along the body shown in Figure 2. To facilitate this movement in real-time during simulation execution, the current keyboard state fed by the user during simulation execution was used as input for the motion control framework.

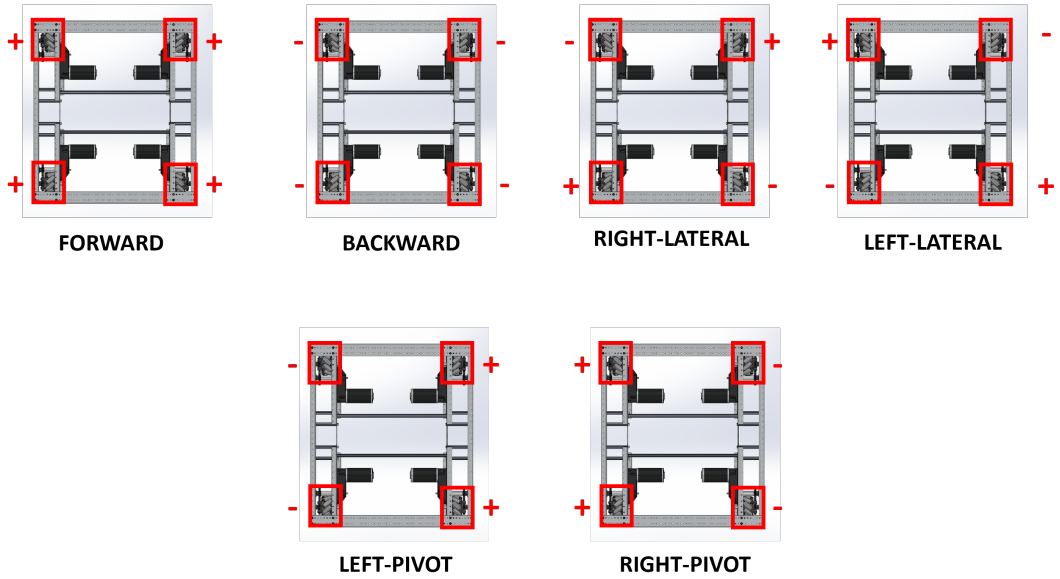

Figure 3. Holonomic motion principles with degrees of freedom illustrated.

The framework for real-time keyboard holonomic control begins with the parameter of a constant positive

value for linear velocity which is then fed into one 4-element array corresponding to 4 wheels illustrated in Figure 3. The six *Read Keyboard State* nodes correspond to the six possible directions of movement of the robot. Let us analyze the motion structures in detail.

The first direction is linear where forward motion is facilitated with all assignment of positive linear velocity values for each wheel and backward motion is facilitated with the assignment of negative velocity values for each wheel. In the case of lateral movement, diagonal wheels possess the same sign in their velocity values. Finally, in the case of pivoting motion, same-side wheels exhibit identical velocity values. A pictorial representation of these motion concepts is presented in Figure 3.

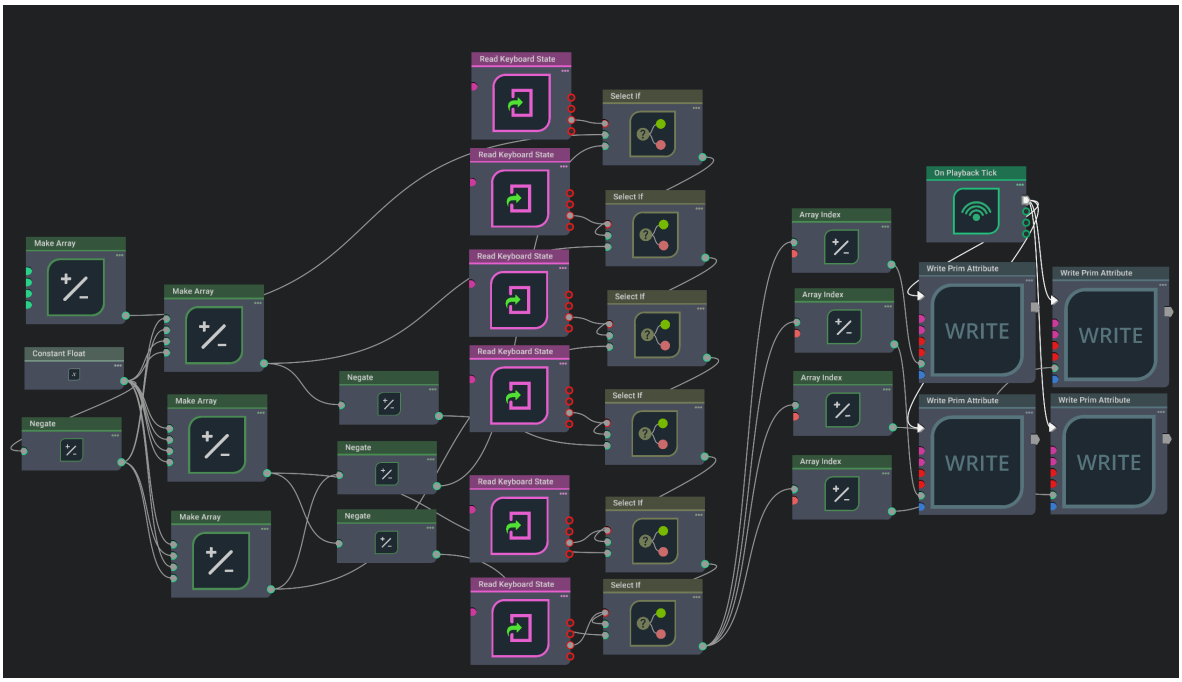

Figure 4. Action Graph for holonomic motion with user keyboard input.

3.3 Out-of-the-box 3d Scene Reconstruction Demonstration

To demonstrate the built-in 3D scene reconstruction capabilities of NVIDIA Isaac Sim, we applied the devised motion policy discussed in Section 3.2 to maneuver the simulated robotic system around a cube within an existing warehouse environment asset within simulation, capturing point cloud data at each frame of motion with a single pre-defined depth camera asset to reconstruct the entire cube with its surrounding industrial environment as a singular point cloud as displayed in Figure 5.

For complete reconstruction of the factory from the synthetic point clouds obtained, the method of Global Registration outlined in Section 2.3.3 was applied. We initially employed the traditional ICP approach discussed in Section 2.3.2 for point cloud stitching. However, the computational time elapsed by this algorithm was significantly higher compared to the global registration method. As a result, we employed global registration to stitch the combined 4,000 point clouds out of the total 4,800 point clouds acquired during simulation execution into a single point cloud, yielding a final stitched point cloud of the entire warehouse environment shown in Figure 5. Due to the large computational resources needed for point cloud registration, down-sampling was applied before and after the successful stitching of each neighbor point cloud. The entire synthetic data acquisition, visualization, and post-processing procedure is illustrated in Figure 5.

While this process of point cloud reconstruction took a significant amount of time, approximately 35 minutes for the full collection of 4,000 point clouds, the end result gave a highly detailed reconstruction of the factory

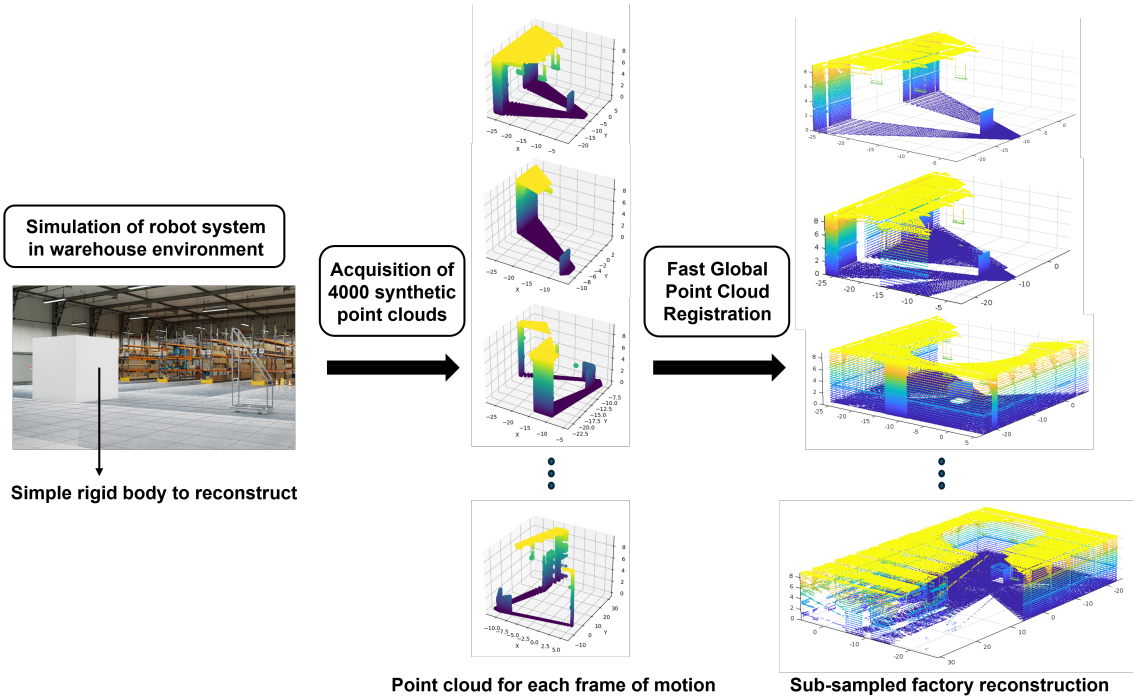

Figure 5. Synthetic point cloud data acquisition of simulated factory environment for full synthetic warehouse reconstruction process visualized.

environment with minute details such as the ceiling light fixtures also captured all through the built-in functionality of the utilized camera asset. Despite the ease in 3D scene reconstruction made possible through the robotic simulation framework, we observe significant quantities of missing features of the factory and the simple rigid cube we intended to fully reconstruct in Figure 6. Missing parts of the ceiling and corner can be attributed to the limited field of view of the camera asset along with the intentional omission of the final 800 point clouds to obtain a sectioned view of the reconstructed pint cloud comprised of 4000 of the 4800 total point clouds acquired.

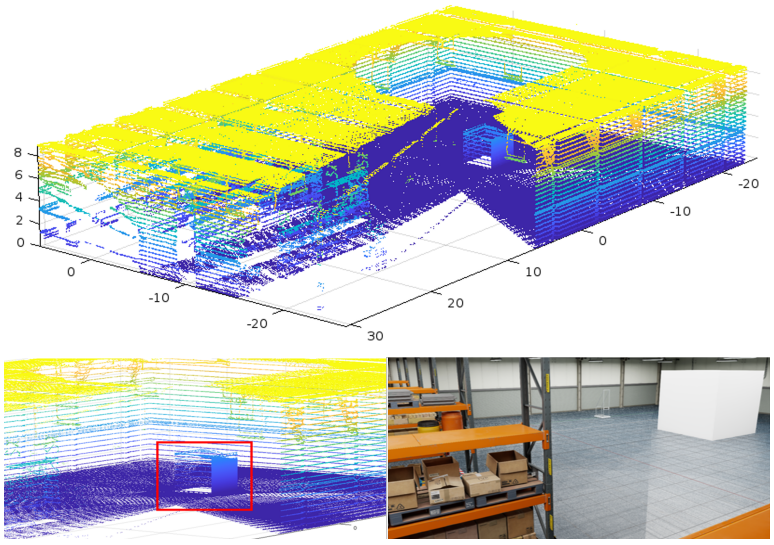

Figure 6. Complete factory point cloud reconstruction with perspective RGB view and cube structure reconstruction.

4. CONCLUSION AND FUTURE WORK

Through the framework we established in Section 3, we demonstrated the potential of NVIDIA Isaac Sim as an efficient simulation platform for the prototyping of complex robotic 3D scanning and industrial scene reconstruction frameworks. We showcased the ease of control over the traversal of custom robotic systems during simulation execution with the developed keyboard control policy shown in Figure 4 and the ability to capture meaningful multi-modal information about complex industrial environments through a single pre-configured RGBD camera asset. We acquire a full reconstruction of a simulated factory environment with the basic camera asset to collect a point cloud of the environment at each frame of motion, then applying Open3D's Global Registration method to stitch each point cloud together for the full reconstruction of the factory displayed in Figure 6.

While a successful demonstration of NVIDIA Isaac Sim's 3D scanning capabilities was done, the current experiment has yet to utilize the full extent of the simulation's features. One of these features is the ability to perform ROS2 integration for control and data acquisition, which can be accomplished by utilization of the ROS2 Bridge part of this framework. Our future work is focused on integrating ROS2 along with pre-existing autonomous navigation libraries such as Simultaneous Localization and Mapping (SLAM), Particle Filters, Dijkstra's Algorithm, etc. Additionally, the other aspect of future work would be focused on obtaining more precise, quick, and accurate 3D reconstruction of dynamic factory environments by the use of custom advanced sensors with a higher resolution than the basic pre-defined depth camera asset.

ACKNOWLEDGMENTS

This research was funded in part by a First-Year Honors Mentor Program Grant made available through the Iowa State University Foundation in conjunction with the University Honors Program.

REFERENCES

[1] Haleem, A., Javaid, M., Singh, R. P., Rab, S., Suman, R., Kumar, L., and Khan, I. H., "Exploring the potential of 3d scanning in industry 4.0: An overview," *International Journal of Cognitive Computing in Engineering* **3**, 161–171 (2022).

[2] Javaid, M., Haleem, A., Pratap Singh, R., and Suman, R., "Industrial perspectives of 3d scanning: Features, roles and it's analytical applications," *Sensors International* **2**, 100114 (2021).

[3] Daneshmand, M., Helmi, A., Avots, E., Noroozi, F., Alisinanoglu, F., Arslan, H. S., Gorbova, J., Haamer, R. E., Ozcinar, C., and Anbarjafari, G., "3d scanning: A comprehensive survey," *arXiv preprint arXiv:1801.08863* (2018).

[4] Helle, R. H. and Lemu, H. G., "A case study on use of 3d scanning for reverse engineering and quality control," *Materials Today: Proceedings* **45**, 5255–5262 (2021). Second International Conference on Aspects of Materials Science and Engineering (ICAMSE 2021).

[5] Callieri, M., Fasano, A., Impoco, G., Cignoni, P., Scopigno, R., Parrini, G., and Biagini, G., "Roboscan: an automatic system for accurate and unattended 3d scanning," in [*Proceedings. 2nd International Symposium on 3D Data Processing, Visualization and Transmission, 2004. 3DPVT 2004.*], 805–812 (2004).

[6] Yin, S., Ren, Y., Guo, Y., Zhu, J., Yang, S., and Ye, S., "Development and calibration of an integrated 3d scanning system for high-accuracy large-scale metrology," *Measurement* **54**, 65–76 (2014).

[7] Afzal, A., Goues, C. L., Hilton, M., and Timperley, C. S., "A study on challenges of testing robotic systems," in [*2020 IEEE 13th International Conference on Software Testing, Validation and Verification (ICST)*], 96–107 (2020).

[8] Berx, N., Decré, W., Morag, I., Chemweno, P., and Pintelon, L., "Identification and classification of risk factors for human-robot collaboration from a system-wide perspective," *Computers Industrial Engineering* **163**, 107827 (2022).

[9] Collins, J., Chand, S., Vanderkop, A., and Howard, D., "A review of physics simulators for robotic applications," *IEEE Access* **9**, 51416–51431 (2021).

[10] Mittal, M., Yu, C., Yu, Q., Liu, J., Rudin, N., Hoeller, D., Yuan, J. L., Singh, R., Guo, Y., Mazhar, H., Mandlekar, A., Babich, B., State, G., Hutter, M., and Garg, A., "Orbit: A unified simulation framework for interactive robot learning environments," *IEEE Robotics and Automation Letters* **8**(6), 3740–3747 (2023).

[11] Ellis, K., Zhang, H., Stoyanov, D., and Kanoulas, D., "Navigation among movable obstacles with object localization using photorealistic simulation," in [*2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*], 1711–1716 (2022).

[12] Son, Y., Han, H., and Cho, J., "Usefulness of using nvidia isaacsim and isaacgym for ai robot manipulation training," in [*2023 14th International Conference on Information and Communication Technology Convergence (ICTC)*], 1725–1728 (2023).

[13] NVIDIA, "Nvidia omniverse isaac sim documentation."

[14] NVIDIA, "Nvidia omniverse isaac sim python documentation."

[15] NVIDIA, "Nvidia omniverse usd overview."

[16] NVIDIA, "Nvidia isaac sim replicator tutorials."

[17] NVIDIA, "Nvidia omniverse isaac sim glossary of terms."

[18] Fernandes, D., Silva, A., Névoa, R., Simões, C., Gonzalez, D., Guevara, M., Novais, P., Monteiro, J., and Melo-Pinto, P., "Point-cloud based 3d object detection and classification methods for self-driving applications: A survey and taxonomy," *Information Fusion* **68**, 161–191 (2021).

[19] Kim, P., Chen, J., and Cho, Y. K., "Slam-driven robotic mapping and registration of 3d point clouds," *Automation in Construction* **89**, 38–48 (2018).

[20] Berger, M., Tagliasacchi, A., Seversky, L. M., Alliez, P., Guennebaud, G., Levine, J. A., Sharf, A., and Silva, C. T., "A survey of surface reconstruction from point clouds," in [*Computer graphics forum*], **36**(1), 301–329, Wiley Online Library (2017).

[21] Pomerleau, F., Colas, F., and Siegwart, R., "A review of point cloud registration algorithms for mobile robotics," *Foundations and Trends® in Robotics* **4**(1), 1–104 (2015).

[22] Besl, P. J. and McKay, N. D., "Method for registration of 3-d shapes," in [*Sensor Fusion IV: Control Paradigms and Data Structures*], **1611**, SPIE (1992).

[23] Team, O. D., "Open3d documentation - icp registration tutorial."

[24] Lei, R. et al., "Deep global feature-based template matching for fast multi-modal image registration," in [*2021 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*], IEEE (2021).

[25] Zhou, Q.-Y., Park, J., and Koltun, V., "Fast global registration," in [*Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II*], **14**, Springer International Publishing (2016).

[26] Team, O. D., "Open3d documentation - global registration tutorial."

[27] Garrote, L. et al., "3d point cloud downsampling for 2d indoor scene modelling in mobile robotics," in [*2017 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*], IEEE (2017).

[28] Hu, Q. et al., "Learning semantic segmentation of large-scale point clouds with random sampling," *IEEE Transactions on Pattern Analysis and Machine Intelligence* **44**(11), 8338–8354 (2021).

[29] Nezhadarya, E. et al., "Adaptive hierarchical down-sampling for point cloud classification," in [*Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*], (2020).

[30] Zou, B., Qiu, H., and Lu, Y., "Point cloud reduction and denoising based on optimized downsampling and bilateral filtering," *IEEE Access* **8**, 136316–136326 (2020).