MVRackLay: Monocular Multi-View Layout Estimation for Warehouse Racks and Shelves

Pranjali Pathre¹, Anurag Sahu¹, Ashwin Rao¹, Avinash Prabhu¹, Meher Shashwat Nigam¹, Tanvi Karandikar¹, Harit Pandya², and K. Madhava Krishna¹

¹Robotics Research Center, IIIT Hyderabad, India, ²Toshiba Research, UK

Abstract—In this paper, we propose and showcase, for the first time, monocular multi-view layout estimation for warehouse racks and shelves. Unlike typical layout estimation methods, MVRackLay estimates multi-layered layouts, wherein each layer corresponds to the layout of a shelf within a rack. Given a sequence of images of a warehouse scene, a dual-headed Convolutional-LSTM architecture outputs segmented racks, the front and the top view layout of each shelf within a rack. With minimal effort, such an output is transformed into a 3D rendering of all racks, shelves and objects on the shelves, giving an accurate 3D depiction of the entire warehouse scene in terms of racks, shelves and the number of objects on each shelf. MVRackLay generalizes to a diverse set of warehouse scenes with varying number of objects on each shelf, number of shelves and in the presence of other such racks in the background. Further, MVRackLay shows superior performance vis-a-vis its single view counterpart, RackLay [1] in layout accuracy, quantized in terms of the mean IoU and mAP metrics. We also showcase a multi-view stitching of the 3D layouts resulting in a representation of the warehouse scene with respect to a global reference frame akin to a rendering of the scene from a SLAM pipeline. To the best of our knowledge, this is the first such work to portray a 3D rendering of a warehouse scene in terms of its semantic components - Racks, Shelves and Objects - all from a single monocular camera.

I. INTRODUCTION

The need for warehouse automation grows by the day, and in the future, a fleet of robots could manage an entire warehouse with little to no human intervention [2]. Yet, almost 30% of warehouses operate without their staple warehouse management systems [3].

In this paper, we address the thus far untackled problem of multi-view layout estimation for all the visible racks in the image. We propose a straightforward and effective network architecture MVRackLay¹, which outputs the top-view and front-view layouts of all shelves making up each rack, partly or wholly visible in every frame of an input sequence of monocular RGB images of a warehouse (these could be the frames of a video) (Fig. 1). Note that a rack may only be partially visible in a frame. The network learns layouts in the canonical frame centered on the shelf, called the shelf-centric layout.

An essential point to note is that the problem is not a direct application of a standard formulation of object recognition, semantic segmentation or layout estimation. While the above methods can be applied to objects on rack shelves [4], [5], present methods cannot be adapted directly to localize rack shelves, as shown in our baseline comparisons in Sec. V-D. While typical layout formulations estimate layouts with reference to a single dominant plane (such as the ground plane) [6], warehouse rack shelves take the form of disconnected planar segments, each present at different heights above the

¹Project page: https://github.com/pranjali-pathre/vRackLay
Using domain randomization, and improved version of the pipeline introduced in [1].

To alleviate the issue of obtaining sufficient training data, we develop and open-source a complete pipeline, as described in Sec. V-A. Moreover, we compile and present results of several ablations involving variations in architecture which establish the superiority of MVRackLay (Sec. V).

II. RELATED WORK

Object detection methods: A significant portion of our problem deals with localizing semantic classes like shelves and boxes/cartons in a 3D scene. There exist several approaches to detect object layouts in 3D. Some of these [8], [9] combine information from images and LiDAR, while others [6], [10] first convert images to bird’s eye view representations, followed by object detection.

Bird’s eye view (BEV) representation: Schulte et al. [11] proposed one of the first approaches to estimate an occlusion-reasoned BEV road layout from a single color image. Wang et al. [12] build on top of [11] to infer parameterized road layouts. In contrast, our approach is non-parametric and hence, more flexible than such parametric models, which may not account for all possible layouts. We take inspiration from MonoLayout [13] (which can be trained end to end on color images, reasons beyond occlusion boundaries and being non-parameterized, need not be actuated with these additional inputs) and extend it to multiple planes.

Single-view layout estimation: RackLay [1] proposed a layout estimation technique that is able to predict shelf layouts of one rack at a time, which must be in focus and completely visible in a single monocular image. Our network MVRackLay is more flexible in that it predicts shelf layouts of all fully-visible and partially-visible racks in a monocular image sequence and more accurate due to the incorporation of spatial data of warehouse racks from the consecutive frames of the input video.

Warehouse Datasets: Publicly available datasets for warehouse settings are far and in between. Real-world datasets like LOCO [14] exist for scene understanding in warehouse logistics, but they provide a limited number of images, along with corresponding 2D annotations. Furthermore, there are only a handful of general-purpose synthetic data simulators for generating photo-realistic images, like NVIDIA Isaac [15], which provide warehouse scenes. However, there is no straightforward way to modify them to generate annotations needed for the task at hand.

Domain Randomization: We integrate domain randomization techniques [16] in our dataset generation pipeline, as described in Sec. V-A.
III. METHOD

A. Problem Formulation

Given a sequence of RGB images $I_1, I_2, \ldots, I_n$ of racks in warehouses in perspective view, we aim to predict the top-view (bird’s eye view) and front-view layout for each rack present in each frame of the input video sequence.

We consider $R$ to be a rectangular area in a top-down orthographic view of the scene. The camera is placed at the mid-point of the lower side of the rectangle, directly facing the racks such that the image plane is orthogonal to the ground plane. (Fig. 2). Concretely, we want our network to generate top-view and front-view layouts for all the racks visible in each frame $I_t$, within a region of interest $\Omega$. Our network predicts shelf-centric layouts where we map $\Omega$ to a rectangular area. In shelf-centric layout representation, we consider $\Omega$ to be a rectangular area, positioned such that its center coincides with the center of the shelves spanning across all the racks visible in the image, as shown in Fig. 2. This layout is hence with respect to the rack and is viewpoint agnostic.

Fig. 2: Top-view representation of the shelf-centric (ii) layout for a given position of a shelf (i), and the reference coordinate frames for the same.

Our model predicts top-view and front-view layout representations for each frame in the sequence. Top-view layouts predict the bird’s eye view occupancy of each shelf on the rack. Each pixel in the layout can either be classified as occupied, unoccupied, or background. A pixel is said to be occupied when it is a part of the object on the shelf, unoccupied when it represents the empty space on the rack, and background when it denotes the region which is not occupied by the shelf.

Consider a right-handed coordinate frame where the X axis points to the right, the Y axis points downwards, and the Z axis points into the plane. For top-view layouts, the coordinate frame is positioned at the center of the shelves spanning across all racks for layouts. Hence, the top-view layouts are parallel to the X-Z plane, as in fig. 2 (and the ground plane) at corresponding shelf heights. In the case of front-view layouts, the center of the coordinate frame is positioned at the center of shelves in X and Y directions. Front view layouts are, therefore, orthogonal to the ground plane, as in fig. 3.

As an additional task, we demonstrate Multi-view Stitching - combining the representation from the top-view and front-view layouts to obtain a 3D reconstruction of all racks in the warehouse in the global shelf frame (Fig. 9). This can be further used for 3D spatial reasoning tasks. We infer the X and Z coordinates from the top-view and the Y coordinate from the front-view. We further explore these applications in Sec. V-F.

B. MVRackLay Architecture

We build a double-decoder MVRackLay architecture (Fig. 4) which takes as input a sequence of RGB images $I_1, I_2, \ldots, I_n$ and predicts the top-view and front-view layouts. The components of the model are described in detail below.

1) A context encoder which uses a ResNet-18 backbone pre-trained on ImageNet [17] to retrieve relevant 3D scene and semantic components from the monocular input $I_t$. This feature extractor learns low-level features $C_1, C_2, \ldots, C_n$ that help reason about the occupancy of the scene points.

2) A stacked Convolutional LSTM submodule uses the encoder extracted features $C_1, C_2, \ldots, C_n$ and in turn encodes a temporal representation to capture motion across input frames. We use this Spatio-temporal prediction to estimate consistent layouts by consolidating the information from the past frames to predict the current frame. The number of previous frames used in this prediction is a hyperparameter, the value of which is varied in our ablation studies (Refer to Sec. V-E). The output of this block is an encoded representation that better reasons the scene points as occupied, unoccupied, and background.
3) A top-view decoder and front-view decoder that generates layouts respectively for top-view and front-view from the temporal representation learned by the Convolutional LSTM submodule. It consists of downsampling layers to output an $R \times D \times D$ grid which represents the layout, where $R$ is the number of output channels and $D$ is the resolution of the output layouts.

4) Identical discriminators following top-view and front-view decoders, respectively, are adversarial regularizers that rectify the layouts further by homogenizing their distributions to be similar to the true distribution of plausible layouts. The layout predicted by the decoder is the input to this submodule which outputs the final refined predictions.

C. Loss function

We describe here the loss function of our MV RackLay architecture for top-view layout estimation. We use stochastic gradient descent to optimize over the network parameters $\phi, \psi, \theta$ of the context encoder, convolutional LSTM, and the decoder.

$$L_{sup}(\hat{T}; \phi, \psi) = \sum_{j=1}^{N} \sum_{i=1}^{R} f\left(\hat{T}_i^j, T_i^j\right)$$

$$L_{adv}(\hat{T}; \phi, \psi, \theta) = \mathbb{E}_{\theta \sim p_{fake}} \left[\left(\hat{T}(\theta) - 1\right)^2\right]$$

$$L_{short}(\hat{T}; \phi, \psi) = \sum_{i=1}^{N} \sum_{j=1}^{\text{seqlen}-1} f\left(\hat{T}_i^{j}, \hat{T}_i^{j+1}\right)$$

$$L_{long}(\hat{T}; \phi, \psi) = \sum_{i=1}^{N} \sum_{j=1}^{\text{seqlen} - 1} \sum_{k=j+2}^{\text{seqlen}} f\left(\hat{T}_i^{j}, \hat{T}_i^{k}\right)$$

$$L_{discr}(\hat{T}; \theta) = \mathbb{E}_{\theta \sim p_{true}} \left[\left(\hat{T}(\theta) - 1\right)^2\right]$$

$$L_{total} = L_{sup} + L_{short} + L_{long} + L_{adv} + L_{discr}$$

Here $\hat{T}$ is the predicted top-view layout of each shelf, $T$ is the ground truth top-view layout of each shelf, $R$ is the maximum number of shelves considered, and $N$ is the mini-batch size.

$L_{sup}$ is the per-pixel cross-entropy loss which penalizes variation of the predicted output labels ($\hat{T}$) from corresponding ground-truth values ($T$). The adversarial loss $L_{adv}$ enables the distribution of layout estimates from the top-view decoder ($p_{fake}$) to be similar to the actual data distribution ($p_{true}$). $L_{discr}$ enforces the discriminator to accurately classify the network-generated top-view layouts sampled from the true data distribution [18]. $L_{short}$ is the short-range consistency loss, and $L_{long}$ is the long-range consistency loss. Finally, we minimize the total loss over the network parameters $\phi, \psi, \theta$ and use it to back-propagate gradients through the network. Equivalent expressions are defined for front-view layout prediction as well.

### IV. DATASET GENERATION PIPELINE

In this section, we introduce WareSynth, a robust pipeline for synthetic data generation, inspired by the more simplistic pipeline in [1]. WareSynth is an end-to-end pipeline that can be used to generate 3D warehouse scenes, capture the relevant data and output the annotations. It is developed in the 3D graphics framework Unity [19] in order to generate more realistic images compared to the original pipeline created in Blender [20]. By using the WareSynth pipeline on an NVIDIA RTX 2080Ti, we are able to generate 500 images per minute. Our pipeline is characterized by the following:

1) Highly customizable as per user requirements.
2) Ability to export to popular annotation formats such as COCO, YOLO, Pix3D, KITTI and BOP.
3) Completely free and open source.
4) Extensive variation and domain randomization.

Although only a handful of warehouse datasets/simulators are currently available such as LOCO and NVIDIA Isaac (discussed in Sec. II), to the best of our knowledge, there is no such pipeline that satisfies all four of the above criteria. WareSynth allows users to customize the number of racks and boxes in the scene and also the box and rack models used. The ability to export to not only our annotation format but also popular annotation formats is important for additional benchmarks against new approaches being developed in these settings. Section V-A enumerates the randomizations enabled by WareSynth. Our data generation and capture methods are efficient, flexible, and easily customizable based on user requirements. Hence, WareSynth can prove very useful for large-scale data annotation generation.

### V. EXPERIMENTS AND ANALYSIS

#### A. MV RackLay Dataset

For training and testing our network, we generated a diverse dataset with 20k images spanning 400 sequences with 50 images per sequence, using WareSynth. It is split into 360/20/20 for train/test/validation. All the results discussed are on the test set of this dataset. The dataset is highly varied and spans multiple warehouse scenes. The variety demonstrates the generalizability of MV RackLay and is useful to evaluate the performance of our model in varied warehouse settings. The video sequences captured using WareSynth

---

For more information on WareSynth and code: https://pranjali-pathre.github.io/MVRackLay/

Download RackLay dataset: https://tinyurl.com/yxmu5t64

---

TABLE I: Quantitative results: We benchmark the 3 different versions of our network - MVRackLay-Disc-4, MVRackLay-Disc-8 and MVRackLay-4, along with three baselines - RackLay-D-disc [1] PseudoLidar-PointRCNN [10], [7] and MaskRCNN-GTdepth [5] (as described in Sec. V-B).
resemble the data captured by a manual camera movement or a mechanized system performing the task in an actual warehouse.

Various scene elements were diversified during data generation to impart assortment in generated scenes so that the synthetically developed warehouses mimic their real-world counterparts. We describe these randomizations below.

**Domain Randomization:** We show the randomizations we introduce using 3 randomly selected images from our dataset through Fig. 5 (referred throughout this section):

- Boxes have random sizes, textures, rotation about the vertical axis, colors, and reflective properties.
- Box placement varies from dense to moderate to sparse.
- Color and texture of racks are randomized.
- Height to which boxes are stacked vertically is randomized.
- Background is either a wall or a busy warehouse.
- Color and textures of floors and walls are randomized.
- The camera’s position concerning the rack varies within a range to capture different numbers of shelves in the sequence. For our dataset, we set $R=3$.
- Camera is moved such that different number of racks are visible across frames.

We find that this large diversity in the dataset has enabled the network to not overfit on the domain of the synthetic data, but rather learn features that emulate real-world scenes.

**B. Evaluated Methods and Metrics**

We compare the following variants of MVRackLay:

- **MVRackLay-Disc-4:** Double decoder architecture with discriminators for both front-view and top-view, with a sequence length of 4 used in the ConvLSTM module.
- **MVRackLay-Disc-8:** Double decoder architecture with discriminators for both front-view and top-view, with a sequence length of 8 used in the ConvLSTM module.
- **MVRackLay-4:** Double decoder architecture for both front-view and top-view without discriminators, with a sequence length of 4 used in the ConvLSTM module.

We report Mean Intersection-Over-Union (mIoU) and Mean Average-Precision (mAP) scores in this task as the previously discussed metrics for the bottommost shelf (row 2). Our model not only extends RackLay's functionality to image sequences but also improves upon its liabilities. We predict sharper and more accurate shelf and object boundaries and reduce false box predictions.

**PseudoLidar-PointRCNN:** PointRCNN [7] is a 3D object detector that uses the raw point cloud as input. Hence we use the PseudoLidar[10] information to detect 3D objects using PointRCNN. As this method considers a single dominant plane and is employed for birds-eye view prediction, we report metrics for the bottommost shelf (refer to Table I) and the top-view prediction only. Accounting for the single-dominant layer assumption, it is clear from Table I that our model performs better as it performs multilayer, front-view, and top-view layout predictions that can also reason about the inter-shelf distance.

**Mask R-CNN:** We select Mask R-CNN[5] as one of our baselines to test the instance segmentation method for multilayer layout prediction in the warehouse setting for the sequential data. We subsequently integrate the Mask R-CNN segmented instances with the depth maps and project on a horizontal plane to segment the boxes shelf-wise, using the fact that a set of boxes on a particular shelf will be situated on the same plane located at some elevation from the ground plane. From the experiment, it is observed that Mask R-CNN fails to detect the precise boundary of the rack due
Fig. 5: MV Rack Lay-Disc-4 Results: Here, we present the results of our network tested on domain randomized data. The bottom-most shelf layout is shown in the left-most column, followed by the middle and top shelf (if visible). Observe the diversity of warehouse scenes captured (detailed in V-A) and the top-view and front-view layouts predicted for the same.

Fig. 6: Rack Lay vs. MV Rack Lay-Disc-4: Above, we compare qualitatively the results of Rack Lay and our MV Rack Lay-Disc-4. The shelf in focus is highlighted with a red border. Observe that our model removes the false positive in row 1, removes noise in row 2, and increases the sharpness of both box boundaries (both rows) and shelf edges (row 2).

Fig. 7: MV Rack Lay-Disc-4 vs. MV Rack Lay-Disc-8: The shelf in focus is highlighted with a red border. Better demarcations between adjoining boxes and less joining of abreast layouts are observed in the output of MV Rack Lay-Disc-4 compared to its counterpart.

to its thin structures. If multiple racks are present in the image, Mask R-CNN also fails to mark a clear distinction between them. The results summarized in Table I prove that our model performs better quantitatively too. Since Mask R-CNN can only claim regarding the points visible in the image, it is evident that our model accomplishes better results with amodal estimation to reason beyond the visual elements that structurally characterize racks and objects.

E. Ablation studies

To thoroughly comprehend the underlying effect of different components, we perform ablation studies over the model’s architecture and examine its effect on performance.

1) Convolutional LSTM sequence length: We varied the time steps used in the stacked Convolutional LSTM sub-module. We observed that MV Rack Lay-Disc-4 was able to converge faster and improve qualitatively over MV Rack Lay-Disc-8. Although quantitatively MV Rack Lay-Disc-4 and MV Rack Lay-Disc-8 perform alike (refer to Table I), from fig. 7 considerable qualitative improvements can be observed. MV Rack Lay-Disc-4 performs better in identifying precise object boundaries and distinguishing between the closely spaced objects on the rack. A lower sequence length enabled the model to compile only relevant details from past frames and avoid spurious noise.

2) Adversarial learning: In MV Rack Lay-Disc-4, we add discriminators after decoders in MV Rack Lay-4 to capture the distribution of plausible layouts. We observed a substantial improvement both quantitatively (refer to Table I) and qualitatively (Fig. 8). Layouts have become sharper, and most notably, using a discriminator diminished the stray pixels wrongly categorized as boxes. The actual distance between the boxes positioned near the end of the shelf is difficult to estimate as they are imaged obliquely. In such cases, MV Rack Lay-Disc-4 remarkably improved the prediction of the boxes and generated cleaner layouts.

F. Applications

Multi View Stitching: From the layout prediction of a particular frame $I_t$, we first obtain the 2D bounding boxes of all shelves and boxes detected in the front-view and top-view layout. The detections from the top-view and front-view layouts are corresponded to identify the matching. Once we
have a map, using the dimension information from front-view and top-view layout predictions, we generate the 3D bounding box for all the mapped racks and objects. Finally, we combine these representations of each shelf to get a 3D reconstruction \( f_t \) of all the racks in the frame. This process is repeated for all the frames in the sequence.

Given two consecutive 3D reconstructions \( f_t \) and \( f_{t+1} \), we initially find all the corresponding matching boxes. We then calculate the shift between them; using this shift, we discern the direction of the motion. Finally, we consider the last box in frame \( f_t \) in the direction of motion and check its corresponding box in frame \( f_{t+1} \). If the size of the box in \( f_{t+1} \) is larger, we increase the size of the shelf and boxes in \( f_t \) accordingly. If there is an addition of a new box or shelf in \( f_{t+1} \), the same is included in the \( f_t \) reconstruction. Eventually, we obtain the merged layouts of \( f_t \) and \( f_{t+1} \) in \( f_t \) frame.

If \( F_t \) denotes the merged 3D representation from \( f_t \) to \( f_{t+1} \), \( F_{t+1} \) is merged into \( F_t \) using the method described above. Fig. 9 shows the 3D reconstruction of a single warehouse with 4 racks, obtained from the multi-view stitching of predicted layouts of 4 sequences with 70 frames per sequence.

VI. CONCLUSION

In this paper, we present MVRackLay to perform multi-view and multi-layered layout estimation for all racks partly or fully visible in each frame of the input monocular image sequence. Distinct from existing methods, it utilizes temporal information across the frames of a image sequence to enhance the quality of layouts. We also present a pipeline to 3D reconstruct the entire warehouse from the predicted shelf-centric layouts. Further, we introduce a versatile synthetic data generation pipeline, WareSynth, that is capable of producing domain randomized data which can emulate a wide variety of warehouse scenes. MVRackLay’s versatility is demonstrated by the experimental results over diverse warehouse scenes, and is vastly superior to previous baselines adapted for the same task.

REFERENCES

[1] M. S. Nigam, A. Prabhu, et al., Monocular Multi-Layer Layout Estimation for Warehouse Racks. New York, NY, USA: Association for Computing Machinery, 2021. [Online]. Available: https://doi.org/10.1145/3490035.3490263

[2] P. Buxbaum, “Many warehouses don’t have wms,” Aug 2018. [Online]. Available: https://www.globaltrademag.com/many-warehouses-dont-have-wms/

[3] “Trends in warehouse automation-2021,” Oct 2020. [Online]. Available: https://www.nitco-lift.com/blog/warehouse-automation-trends/

[4] O. Ronneberger and Fischer, “U-net: Convolutional networks for biomedical image segmentation,” in International Conference on Medical image computing and computer-assisted intervention, 2015.

[5] K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Maskrcnn,” in Proceedings of the IEEE international conference on computer vision, 2017, pp. 2961–2969.

[6] T. Roddick, A. Kendall, and R. Cipolla, “Orthographic feature transform for monocular 3d object detection,” arXiv preprint, 2018.

[7] S. Shi, X. Wang, and H. Li, “Pointrcnn: 3d object proposal generation and detection from point cloud,” in CVPR, 2019.

[8] J. Ku, M. Mozifian, et al., “Joint 3d proposal generation and object detection from view aggregation,” in IROS, 2018.

[9] M. Liang, B. Yang, S. Wang, and R. Urtasun, “Deep continuous fusion for multi-sensor 3d object detection,” in ECCV, 2018.

[10] Y. Wang, W.-L. Chao, et al., “Pseudo-lidar from visual depth estimation: Bridging the gap in 3d object detection for autonomous driving,” in CVPR, 2019.

[11] S. Schulte, M. Zhai, N. Jacobs, and M. Chandraker, “Learning to look around objects for top-view representations of outdoor scenes,” in ECCV, 2018.

[12] Z. Wang, B. Liu, S. Schulte, and M. Chandraker, “A parametric top-view representation of complex road scenes,” in CVPR, 2019.

[13] K. Mani, S. Daga, et al., “Monolayout: Amodal layout estimation from a single image,” IEEE Winter Conference on Applications of Computer Vision (WACV), 2020.

[14] C. Mayershofer, D. M. Holm, B. Molter, and J. Fottner, “Loco: Logistics objects in context,” in 2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA), 2020, pp. 612–617.

[15] “NVIDIA Isaac SDK,” 2019. [Online]. Available: https://developer.nvidia.com/isaac-sdk

[16] J. Tremblay, A. Prakash, et al., “Training deep networks with synthetic data: Bridging the reality gap by domain randomization,” in Proceedings of the IEEE conference on computer vision and pattern recognition workshops, 2018, pp. 969–977.

[17] J. Deng, W. Dong, et al., “Imagenet: A large-scale hierarchical image database,” in CVPR, 2009.

[18] I. Goodfellow, J. Pouget-Abadie, et al., “Generative adversarial nets,” in Advances in Neural Information Processing Systems 27, 2014.

[19] “Unity real-time development platform | 3d, 2d vr | ar engine,” 2021, accessed: 13-09-2021. [Online]. Available: https://unity.com/

[20] B. O. Community, “Blender - a 3d modelling and rendering package,” Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. [Online]. Available: http://www.blender.org

Fig. 8: MVRackLay-4 vs. MVRackLay-Disc-4: The shelf in focus is highlighted with a red border. Observe how using a discriminator fixes the false negative in row 1 and improves predicted box boundaries and shelf boundaries in row 2.

Fig. 9: Multi View Reconstruction of the entire warehouse using four sequences covering 280 frames (70 frames each), using the layouts predicted by MVRackLay-Disc-4.