SORDI.ai: Large-Scale Synthetic Object Recognition Dataset Generation for Industries

Chafic Abou Akar[1,2*], Jimmy Tekli[1], Joe Khalil[1,2], Anthony Yaghi[1,2], Youssef Haddad[1], Abdallah Makhoul[2†], Marc Kamradt[1†]

[1]BMW Group, Munich, Germany.
[2]Institut FEMTO-ST, CNRS, Université de Franche-Comté, Besançon, France.

*Corresponding author(s). E-mail(s): chafic.ac.abou-akar@bmw.de;
†These authors contributed equally to this work.

Abstract

Smart robots play a crucial role in assisting human workers within manufacturing units (like Industry 4.0) by perceiving and analyzing their surroundings using Deep Learning (DL) models for Computer Vision (CV) applications. On the one hand, training DL models requires extensive annotated data. On the other hand, the scarcity and specificity of publicly available industrial datasets as well as the ethical, privacy, technical, and security challenges for capturing and annotating real images in industrial setups raise the problem of finding an alternative to train DL models for CV applications. In previous work, we proposed a simulation-based synthetic data generation (SDG) pipeline to render 200,000 images of eight industrial assets using NVIDIA Omniverse. In this study, we leverage the SDG pipeline to build and maintain dynamic and modular scenes, resulting in large-scale complex industrial simulation scenes. Furthermore, they feature Domain Randomization (DR) to increase content variability, and hence to bridge the reality gap. Inspired by real assembly lines, production areas, storage rooms, warehouses, offices set up, etc., we extensively render photorealistic images, rich in variations, capable of generalizing DL models to new unseen environments. Consequently, we introduce SORDI.ai, a comprehensive synthetic industrial image dataset for object detection applications. It comprises over a million images covering more than one hundred object classes belonging to logistics, transportation, signage, tools, and office assets.

For evaluation purposes, we trained object detection DL models with our synthetic dataset, and inferred over a target dataset containing real/synthetic

images. We gradually tested different levels of DR to demonstrated how does the reality gap bridge. Afterward, we showed the importance of mixing multi-domain training dataset to achieve better generalization, and the efficiency of our SDG pipeline to increase prediction accuracies in low real data regimes.

Keywords: Industry 4.0, Object Detection, Reality Gap, Simulation, Synthetic Data

1 Introduction

Manufacturing units around the world employ smart robots to carry out clearly defined tasks and assist human workers in their daily warehouse, production, assembly, and logistic tasks resulting in industrial process time, cost optimization, and quality enhancement [1–4]. Computer Vision (CV) tasks (e.g., image classification, object detection, etc. [3–6]) play a crucial role in enabling smart robots to perceive and understand their surroundings, allowing them to accurately locate and identify specific objects within a given scene [2]. As demonstrated throughout the years, Deep Learning (DL) outperformed traditional learning-based approaches in a wide range of CV applications [7–9]. Nevertheless, training DL models requires capturing, storing, and annotating large image datasets [10]. Furthermore, these training datasets should contain high quality images that are diverse, balanced, informative [11], and adheres to ethical guidelines, including principles of privacy, consent, etc. [12]. In an industrial context (e.g. manufacturing units, assembly lines), acquiring and annotating images is challenging because it is time-consuming, prone to human error [13, 14], and limited by ethical [12], privacy and security regulations [15, 16]. On the one hand, several studies throughout the years proposed real images datasets for industrial CV applications [3]. On the other hand, the majority are either not publicly accessible or, if available, are limited in number of assets/objects[1] due to their task-specific design [3]. As stated in [13, 17, 18], synthetic image data generation (SDG) [19–21] can address the above challenges, arising in an industrial setting, while generating a large image dataset with the desirable properties. Several SDG approaches were proposed in the literature [22–24]. In this study we consider simulation-based approaches which consist of an automated process to render and accurately annotate numerous synthetic images out of virtual scenes that include the main object of interest surrounded by randomized assets and distractors [18–21, 25–28]. Synthetic images, used mainly to train DL models, are considered the source domain. Whereas, real images captured in real industrial environments, used mainly for inference, are considered the target domain, equivalent to an evaluation or deployment domain [18]. The difference between both domains is known as the reality gap, sim2real gap, or in broader terms, domain gap [13, 29–31]. Many studies mention that the SDG depends on two fundamental approaches: Image Realism (IR) and Domain Randomization (DR) [18, 24, 32]. IR bridges the reality gab by enhancing the photorealism aspect of the synthetic image [18, 33, 34]. Whereas, DR randomizes the simulation components and assets while considering the real world as just a random instance of that simulation [19].

[1] For the rest of this paper, we will use the terms object and asset interchangeably

To bridge the reality gap, IR and DR must be combined together, especially that relying solely on image photorealism by increasing the rendering quality is costly [35, 36], and contrary to human perception, realism is not always beneficial in CV [3]. However, DR must be cautiously implemented in a way it does not decrease the visual fidelity components such as the asset's possible realistic textures and surfaces, which results in a higher reality gap [18, 37–39]. In accord, researchers argue that including domain knowledge leads to structured DR (SDR), generates better data, and as a result, it increases the models' performance and accuracy by learning relationships between assets [17, 18, 37, 40–48], especially in large scale scenes where applying random DR could turn the simulation scene into a chaotic and unrealistic environment. To the best of our knowledge, most available SDG tools [18, 25–28, 49–52] are not suited for large-scale scene creation such as constructing real-alike and equipped industrial areas, rooms, and buildings. Nevertheless, rendering images from large-scale scenes increases image diversity and background realism resulting in a larger dataset with rich multi-class annotations. Furthermore, it allows object detection in complex and crowded environment for object transportation, smart navigation, warehouse and inventory management, supply-chain and planning optimization, etc. However, existing industrial datasets [3, 18, 24, 49, 53–62] are designed for specific tasks, hence they do not cover a wide range of industrial assets. In a previous study [13], we proposed an SDG pipeline and rendered 200,000 images covering 8 annotated industrial assets. In this paper, we enhance and extend the proposed SDG pipeline for large-scale generation by (1) implementing the Universal Scene Description (USD) pipeline's interoperability and modularity features [63, 64] to maintain realism starting from the smallest 3D models into the larger scene compositions that were inspired by real-world environments, and (2) employing domain knowledge in the industrial field with DR to create an extensive library of SDR-based modular components that are used in all our simulations. Thus, we introduce SORDI.ai, a dataset for industrial object detection use cases with more than a million photorealistic path-trace rendered images of 111 industrial assets annotated with bounding boxes (bbox). SORDI.ai covers 20 scenes with various industrial environments like warehouses, storage rooms, offices, production/assembly lines, etc., and real-world industrial plant replicas. Furthermore, in a first experiment, we examine how DR bridges the domain gap, and the efficiency of background randomization for object detection tasks in crowded and real world environments such as industrial areas. Moreover, in a second experiment, we analyze the effect of camera viewport randomization, and the importance of mixing multi-domain datasets to increase a DL model's generalization ability. Additionally, we emphasize the boosting effect of synthetic data to increase the detection accuracy in low data regimes.

The remainder of this paper is organized as follows: First, we present in Section 2 major SDG concepts in the state of the art, related to IR and DR. Then, in Section 3, we present our 4-steps SDG pipeline. Furthermore, we present our dataset SORDI.ai in Section 4. In Section 5, we demonstrate how the DR approaches that we employed in SORDI.ai affect the performance of the DL object detection models and how they gradually bridge the gap between the source and the target domains. Finally, we discuss some limitations/future work in Section 7 and conclude in Section 8.

2 Related Work

In this section, we review the latest work related to SDG, DR, and the available industrial CV datasets.

2.1 Domain Randomization & SDG Toolkits

Morrical *et al.* classified 3 major DR types [50]: DOME [19, 20, 65], MESH [62, 66], and FAT [21]: These methods include full DR for the assets within its surrounding by either placing or falling the object of interest on randomized texture surfaces or in front of a dense background full of realistic assets. As a result, DR improved the neural network (NN) to learn important features of the object of interest. However, previous DR approaches randomize the assets in an unrealistic way which is unsuitable for an organized and structured industrial environment [2]. The industrial environment is known a priori [18], and it is less versatile compared to real-world environments used in large-scale datasets, e.g., MS COCO [67], and therefore it is easier and better defined to reproduce in a simulation [18]. For instance, Rutinowski *et al.* illustrated in [17] their real dataset creation framework, different possible scenarios for placing different types of pallets and small load carrier (KLT) boxes, stillages (mesh boxes), barrels, and forklifts within a warehouse. Additionally, they described distinct behaviors for the pallets (empty & fully loaded) and did not manifest any color or hue light variations in the scenes. Similarly, we twisted our SDG's DR with the domain knowledge of the industrial field to impose structure and context to our scene composition and to bridge the reality gap. In that way, NN takes into consideration the surroundings for each asset and learns asset relationships resulting in better accuracies, as shown in [18, 43, 45–47, 49].
In addition, DR approaches improved over time [65], thus many researchers built on top of them and extended several automated SDG tools such as CAD2Render [49], Kubric [25], BlenderProc [26], NDDS [27], NViSII [50], Unity Perception [28], Omniverse Replicator [51, 52], BlenderGen [18], etc. However, existing tools and approaches cannot provide large-scale scene constructions and randomizations in opposite to our SORDI.ai scalable and modular approach in which we ensure expert collaboration to contribute to the same "calibrated" simulation scene taking into consideration all tiny details of every 3D asset and all realism aspects that are previously mentioned [40]. Afterward, the calibrated sim is expanded with extensive SDR so it is ready to render high-quality and photorealistic industrial images.

2.2 CV for Industry

In a recent review [3], Naumann *et al.* listed logistic and warehouse-related applications based on CV. However, we notice that each industrial application focuses on specific assets such as pallets, small load carrier box, container, or forklift. Additionally, less than 10% of the datasets are publicly available. Hence, we conclude that the industrial field is scarce in data, and existing datasets cover single or small category of assets, contrary to CV benchmark and rich datasets MS COCO [67], Cityscapes [68], ImageNet [69], etc., or our proposed SORDI.ai dataset.

Specifically, researchers rendered a single industrial part or asset per image by randomizing it in front of a random background or in full DR environment. For instance, Zhu *et al.* published SIP-17, a dataset of 33,000 single asset images covering 17 industrial objects such as airgun, hammer, hook, wheel, etc. for 6 industrial use cases [53, 54]. The Synthetic Corrosion Dataset contains 270 training images to detect corrosion in industrial units and on products [55]. The dataset of Industrial Metal Objects (DIMO) includes 553,800 images of 6 metallic objects, e.g., cylinders, blocks, shafts, etc. with different shapes and materials and applied 71 combinations of texture and light randomizations to 600 diverse scenes with random set up of object shapes, materials, carriers, compositions, and lighting [56, 57]. Moonen *et al.* introduced their CAD2Render photorealistic SDG toolkit [49]: They import CAD models and polish them with random textures, then apply rust or scratches to the surface, before placing them in a random environment and path-trace rendering images. The authors rendered 20,000 images of small metal pieces to pick, and 80,000 images for 4 industrial tools (screwdriver, hammer, wrench, and combination wrench). Petsiuk *et al.* extended their SDG pipeline with CycleGAN [58] and translated real (test) images of 3D printed parts like turbine, cogwheel, chassis, and holder, into the synthetic domain to improve the performance of segmentation models solely trained on synthetic data. However, the proposed approach still requires the collection of real data to compose the real domain for training [59]. Dirr *et al.* proposed a pipeline to render physically-accurate electric wires images for segmentation. The authors produced 96,000 segmentation images for up to 6 cables with versatile deformations inside a container for a provisioning use case, and other 5,000 images for a electric cable benchmark [24]. PalLoc6D provides 200,000 images of a photorealistic DIN EN 13698-1 EPAL Euro pallet (EPAL 1) model with Physically Based Rendering (PBR) realistic textures using 3 levels of DR [60, 61]. Mayershofer *et al.* rendered 8,200 images of 5 types of KLT boxes with object-alike distractors [62]. Eversberg *et al.* studied in [18] the effect of DR over realism by rendering 5,000 images of turbine blades on which they applied different texture variations from MS COCO, random, realistic, and real material images, in front of background images that are chosen randomly from MS COCO dataset or the application domain. Then, they distracted the turbines with YCB tools [70].
In short, existing industrial datasets do not cover images for industrial environments inside manufacturing units for scene analysis use cases. They are focused on capturing a single or small group of assets placed on different surfaces and under varying light conditions that are, for instance, beneficial for a single asset or specific asset object recognition and 6D pose estimation. In contrast, our proposed dataset includes annotated industrial environment images, rendered from 20 scenes that contain logistics, transportation, tools, office, and signs assets placed in way similar to real-world scenarios.

3 Synthetic Data Generation Pipeline

In this section, we present our synthetic image data generation SDG pipeline. As illustrated in Fig. 1, our proposed pipeline is composed of four main steps: (1) **asset**

Table 1 Summary of the synthetic industrial datasets mentioned in Section 2.2

Dataset	Classes	Annotations	Use Cases
SIP-17 [53]	17 isolated & assembled parts	Classification	Cabin assembly, logistic picking, wheel assembly, engine assembly
Corrosion [55]	Corrosion	Classification	corrosion detection
DIMO [56]	6 metallic shapes	6D Pose	Reflective object pose estimation
Moonen *et al.* [49]	Small metal piece	6D Pose	Bin picking
Moonen *et al.* [49]	4 hand tools	2D Keypoint	2D keypoint detection
Petsiuk *et al.* [59]	3D printing CAD objects	Segmentation	Visual analysis & error detection in additive manufacturing
Electric Wires [24]	Wire	Segmentation	Provision of deformable linear objects
PalLoc6D [60]	Specific pallet	6D Pose	Material handling and tracking
Mayershofer *et al.* [62]	5 types of RL-KLT boxes	Bbox, Depth, Segmentation	KLT box detection
Eversberg *et al.* [18]	Turbine blades	Bbox	Object detection
(Ours)	111 objects of logistic, transportation, office, tools, signage	2D BBox	Industrial use cases based on object detection in complex environments

preparation to build the 3D asset library that is used in (2) **scene construction**. These scenes are inspired by real world setup for (3) **data capture** to render realistic and annotated images. Afterward, the (4) **quality assessment** step enhances the quality of our rendered dataset. Each step is detailed in further subsections.

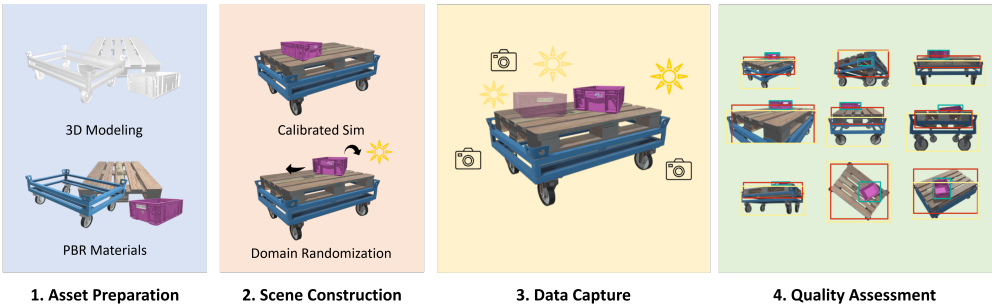


Fig. 1 Our SORDI.ai 4-step proposed SDG pipeline

3.1 Asset Preparation

In this section, we present the creation of a library of industrial simulation resources. It contains 3D models, textures, materials, and layouts that were imported into the USD pipeline to build our 3D scenes including:

1. **Hand-model industrial specific assets**: the USD pipeline allows the decentralization of team members allowing experts to collaborate on the same reference model while staying real-time updated with all changes - independently from the digital content creation (DCC) tools [63]. We used Blender, Unreal Engine, and Adobe Substances to produce industrial 3D PBR models with fine details and realistic surfaces. As mentioned in Appendix A.1, the high-definition and realistic PBR surfaces of the asset's 3D models reduce the reality gap. These assets are compliant with international industrial standards, e.g., the German Association of the Automotive Industry (VDA).
2. **Converted assets**: we converted existing 3D mesh models into USD formats and included them in the same project since the USD's interoperability allows the integration of the most recent DCC software into the workflow.
3. **Publicly available open-source assets**: we used certain free assets from open-source communities.

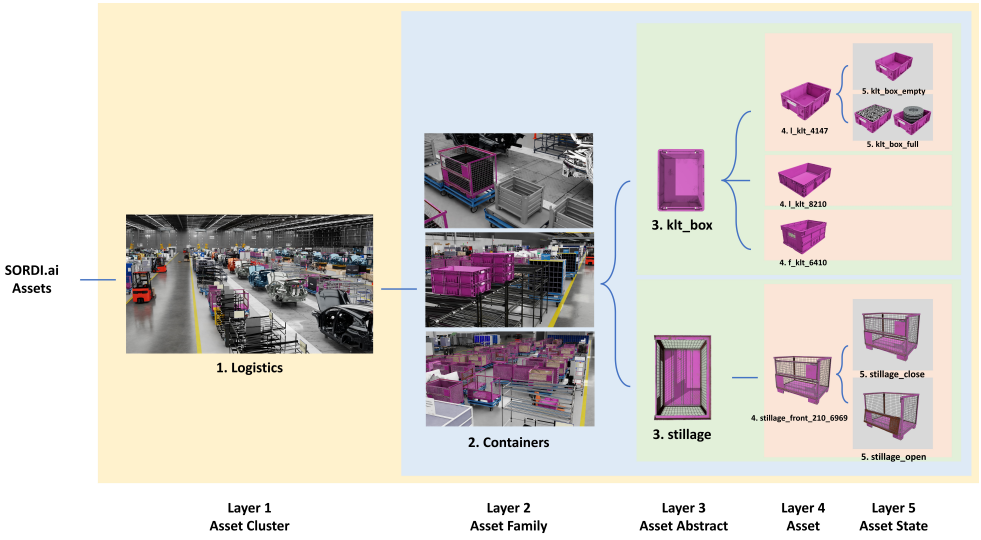


Fig. 2 Subpart of SORDI.ai's logistic containers branch

Asset classes: We defined a 5-layer taxonomy for our assets as seen in Fig. 2. In the first layer (asset cluster), we group the assets into 5 clusters: logistics, transportation, tools, signage, and office assets. In the second layer (asset family), we identify the assets' general function or type, such as storage, containers, mechanical tools, electrical tools, etc. The third layer (asset abstract) represents a class or category of objects without specifying any particular attributes, or details. It encapsulates a group of objects sharing common traits or belonging to the same category/asset family. In the fourth layer (asset), we define a more comprehensive and specific representation, version, and variation of the objects compared to the generic asset in layer 3, for e.g.,

the different shape standards of a KLT box: L-KLT 4147, L-KLT 6147, L-KLT 3147, etc. Last but not least, the fifth layer (asset state) represents the object's behavior or state, such as whether it is full, empty, open, or closed. The full tree taxonomy is available online: https://www.sordi.ai/tree.

3.2 Scene Construction

A large-scale and versatile dataset requires a large variety of virtual scenes. To easily maintain and upgrade the scenes in SORDI.ai, we present in this section our USD adaptation for scene modularity and composition - It enables the creation of a 3D scene by combining many "modular" smaller scenes into larger and more complex aggregated scenes. [63, 64] (check Fig. 4).

Scene randomization: In order to scale our image generation pipeline to meet the need of various industrial use cases, we must generate a huge number of synthetic images with a variety of scenarios and parameters. As mentioned in Appendix A, DR leverages a static scene into a dynamic scene with various combinations. In our dataset, we consider three levels of randomization: DR-1, DR-2, and DR-3.

Table 2 SORDI.ai randomization levels

	Asset					Room	
	Shape	Surface	Translation	Rotation	Behavior	Components	Light
No-DR	✗	✗	✗	✗	✗	✗	✗
DR-1	✓	✓	x, y	yaw	✓	✗	✗
DR-2	✓	✓	x, y	yaw	✓	Textures & Materials	✗
DR-3	✓	✓	x, y	yaw	✓	Textures & Materials	✓

For **DR-1**, we created a library of randomized assets, and asset groups (multiple assets related semantically to one another [48], e.g., KLT box, pallet, and dolly as in Fig. 1) based on visibility randomization components in an attempt to generate all possible logical combinations such as stacked KLT boxes as in Fig. A1, and possible asset behaviors and appearances. However, a single asset variation is limited/constrained and aligned by the official VDA standards for the assets' dimensions and shapes to maintain our scenes' realism. Then, we placed all composed groups in the scene and then applied for x and y positions in addition to yaw rotation randomizations. The randomization range is set to predefined regions to avoid asset collisions. In addition, we ignored the z-axis randomization to consider the physical gravity effect and avoid floating assets. However, some randomized asset groups can be spawned with non-zero values of z, roll, or pitch, e.g., leaning or stacked objects, etc. We elaborated on these cases in the next paragraphs. Thereupon, and in short, we maintained the scene content realism.

As for **DR-2**, we additionally randomized the walls, the ground, and the ceiling's materials and parametric textures to simulate the background variability. We used 320 Material Definition Language (MDL) materials selected from NVIDIA's vMaterials collection [71]. The collection features paint, stone, plaster, fabric, metal, concrete,

wood, textile, and so many other materials with various effects of reflectivity, emissive, opacity, etc. [72] as shown in Fig. 3. Last but not least, we include in **DR-3** the light source's color, intensity, and rotation randomizations as shown in Fig. 3. In Table 2, we compared between different DR levels.

Fig. 3 Mix of DR components from the same camera viewpoint

Scene composition: Based on the USD's feature for scene modularity [64], we built a 9-layer architecture to ensure the composition of large complex scenes without starting from scratch every time (check Fig. 4), featuring the advantage of assets reusability. Moreover, using the same components from other scenes guarantees our scene's industrial environment consistency:

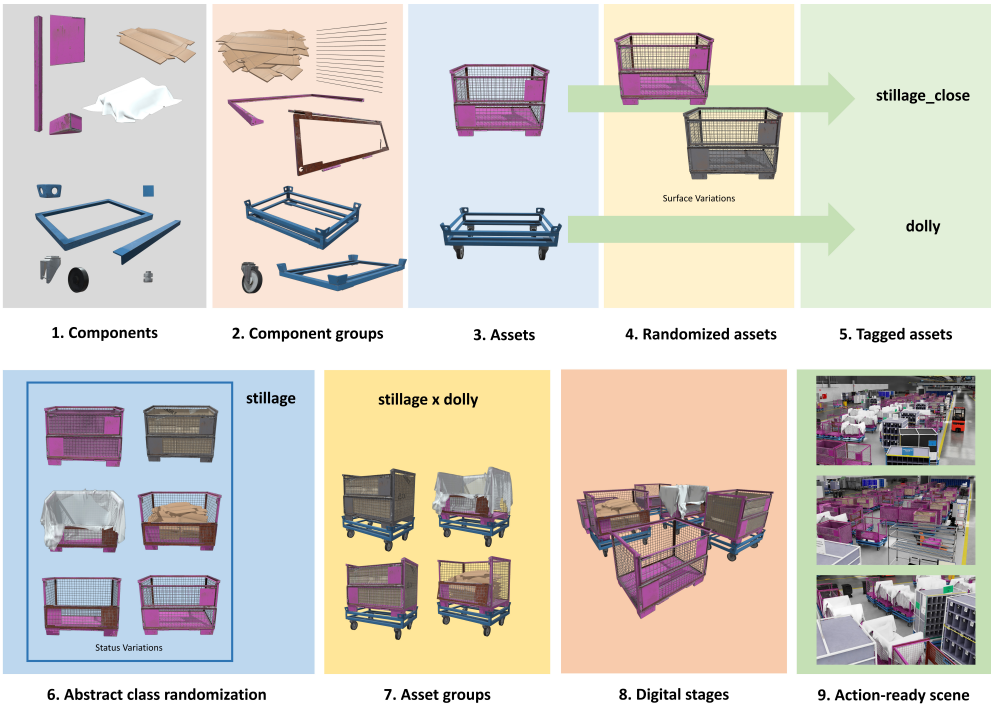


Fig. 4 SORDI.ai's 9-layer large-scale scene composition

1. **Component**: It is the smallest 3D objects defined as "children" assets used to construct parent assets, such as screws, axis, wheels, washers, pipes, etc. Each component surface is "polished" with its appropriate PBR material and texture.
2. **Component groups**: Combined/Collection of components used to create re-occurring components in larger "SORDI asset".
3. **SORDI asset**: A refined PBR asset assembled using component groups and components - This results in fine-detailed and complex industrial assets with multiple surfaces which were previously challenging to compose and to model [49]. Additionally, this asset is also textured with ambient occlusion and appropriate textile density maps to ensure texture details are rendered realistically. At this stage, the asset gains its behavioral status and functional role.
4. **Randomized asset**: The SORDI asset demonstrates appearance randomization by pulling in pre-rendered texture and shape variations.
5. **Tagged asset**: At this stage, the asset is labeled with its proper asset (object class) ID and name as object detection annotations. As an output, all the appearance variations of the same randomized SORDI asset (check Layer 4) are named and associated with the same tag.
6. **Abstract asset randomization**: It consists of multiple layers for content randomization, equivalent to layers 4 and 5 of the asset taxonomy. Therefore, we randomize different versions and behaviors of the same generic abstract asset. However, each behavior or status has its own tag from the previous layer, e.g., stillage_open, stillage_close, stillage_full, etc. Then, we randomize between these types of stillages. As a result, and when spawning it into the virtual scene, the asset will alternate between different behaviors, versions, and surfaces of the stillages.
7. **Asset groups**: A combination of different context-specific assets that are linked by physical, semantic, and functional relationships. e.g., KLT boxes on rack shelving, or a stack of KLT boxes, etc. For instance, spawning an asset group of "KLT stack on a pallet" would result in distinct variations of KLT stack compositions as illustrated in Figure A1. Sequentially, transform randomization is additionally applied on an upper/parent layer/object.
8. **Digital stages**: Building on top of the USD modularity, asset groups are spawned alongside other groups to form a realistic environment, and so on. This features scalable and growing scene groups for large-scale scene composition. Moreover, it is worth mentioning that all assets are spawned by reference, hence any change in the lower layers is propagated to all upper layers updating in real-time all asset groups and digital stage scenes.
9. **Action-ready scene**: It is the final camera-ready scene composed of multiple sub-scenes, and which is used to render our synthetic images.

Additional note: We mentioned previously that the transform randomization affects the x, y, and yaw axis only to maintain the scene's SDR, physics, and realism. For instance, it is possible to spawn leaning asset groups against the wall, so by running the asset groups' translation randomization, it will lean against many positions of the wall. Therefore, when checking its world coordinates, the roll and pitch values may not be equal to zero. Or in another scenario, we could spawn the asset group on a shelf or rack, so its world's z-axis value could be higher than 0.

Additionally, it is worth mentioning that the randomization effect of spawned assets is asynchronous. In other terms, if a randomized asset has n variations, spawning the randomized asset twice or m times would result in $2 \times n$ or $m \times n$ variations respectively, resulting in a larger variety of combinations. In contrast, a synchronous randomization would maintain the same variation of all the same spawned assets at the same time, resulting in n variations only for m spawned assets.

3.3 Camera Randomization, Data Capture & Annotation

The camera is also an essential asset within the scene to display the virtual environment and render photorealistic images. In our dataset, we used a path-trace rendering algorithm with NVIDIA RTX GPUs. However, as a normal 3D object, the camera has also 3D characteristics, e.g. visibility, position, and rotation. Therefore, the camera movement and behavior in the scene affect the data capture. As shown in Fig. 5, we define four main types of data capture:

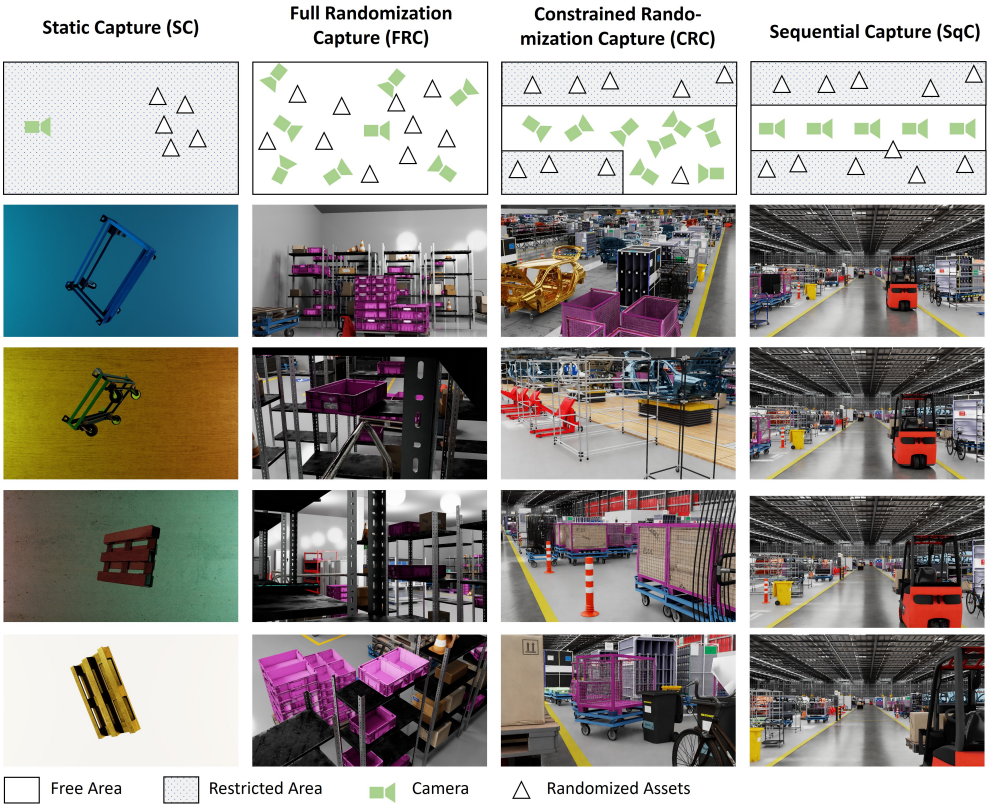


Fig. 5 Samples of the four main types of data capture SC, FRC, CRC, and SqC

1. **Static capture (SC)**: It consists of fixing the camera at a static position and rotation. In this case, DR is a must, otherwise, the same image is rendered at every frame.
2. **Full randomization capture (FRC)**: It consists of randomizing both camera's source and target points at any point in the scene's room. This diversifies the camera viewpoints, e.g. high and low angles, tilted and point of view, long and close shots, aerial shots, etc. Otherwise, randomizing the camera target only will be like a person standing in the same position looking around, and randomizing the camera source, only, results in blind areas. In short, the camera is placed all over the 3D scene, and it is pointing toward random positions.
3. **Constrained randomization capture (CRC)**: It is similar to the FRC, but the camera's source or target points are randomized within specific regions, for instance, we omit captures inside highly dense areas or inside closed assets that are not accessible by humans or robots. Images at these viewpoints are irrelevant for training. Moreover, we configure the camera's target point to ignore capturing empty regions of a virtual scene such as the ceiling.
4. **Sequential capture (SqC)**: Images are rendered with the same distance and angle (viewpoint) at which a camera - fixed on a robot - observes and captures in real scenarios. For instance, in the case of Autonomous Mobile Robots (AMRs), the virtual camera is placed at a fixed distance from the ground, and is moving through a well-defined path as it was fixed on a transport AMR navigating the scene. (check Section 5).

Alongside rendering images, Isaac Sim generates accurate annotations as bbox 2D Tight [73].

3.4 Quality Assessment

As a final step, quality assurance affects both images and their annotations. Therefore, we implemented multiple algorithms to remove:

1. **Duplicate or highly similar images**: we measure image similarity using Manhattan distance for image hashes as implemented in [13]. The smaller the distance is between two images, the more similar/identical the two images are.
2. **Close shot images of solid assets**: a solid asset does not have any gaps or perforations, and hence, it does not allow visibility through its mesh as shown in Fig. 6 a. Therefore, we discard the image where a solid asset's bbox occupies more than a predefined threshold of that image:

$$Occupancy = \frac{S_{\text{bbox}}}{S_{\text{image}}} = \frac{W_{\text{bbox}} \times H_{\text{bbox}}}{W_{\text{image}} \times H_{\text{image}}} \tag{1}$$

   where S is the surface area of a bbox or a rendered image, W and H are the corresponding width and height. In addition, the maximum $Occupancy$ is equal to 1 and it corresponds to a solid asset which totally covers the image.
3. **Non-informative images**: we consider a region without any annotated asset (without any bbox) as a non-informative area like room's components (wall, ceiling,

floor), high occlusion by an unlabeled object, etc. Hence, we discard an image if the non-informative area surpasses a predefined threshold as shown in Fig. 6 b.

4. **Bboxes with an area of zero**: a zero-area bbox results when the top and bottom coordinates or the right and left coordinates of a bbox are equal. In such a case, we omitted the corresponding bbox from the annotation files.

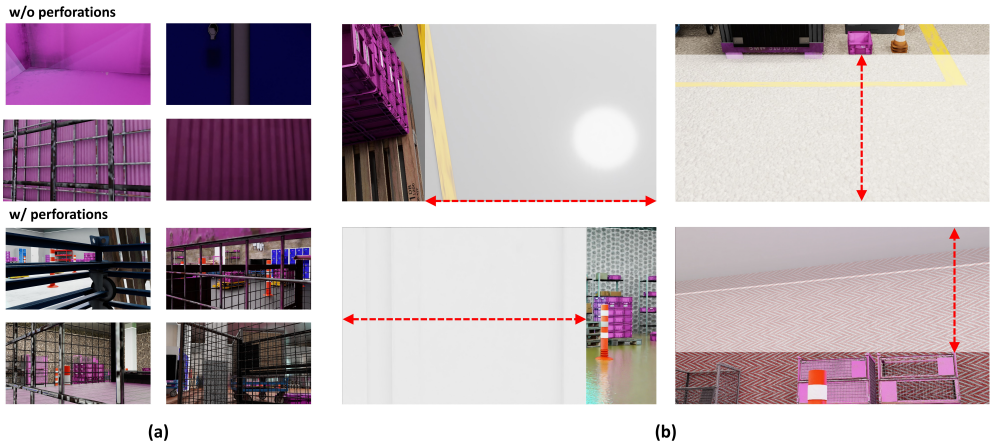


Fig. 6 Samples of (a) close shot images (b) non-informative images

4 SORDI.ai Dataset

In this section, we describe our proposed SORDI.ai dataset.

Material setup: In this release of SORDI.ai, we used a path-trace rendering algorithm with NVIDIA RTX A6000 48 GB, and RTX 3090 48 GB GPUs. We rendered images using NVIDIA Omniverse's Isaac Sim v2021.2.1 and converted the bbox annotations into BMW JSON format [74]. In addition to the annotated folders, we provide a general config.json file details of the dataset folders like: description, image dimension and type, dataset application, annotation types, supported objects and their instance numbers, scene environment, generation settings, etc.

4.1 Datasets Detail

Our proposed SORDI.ai dataset includes photorealistic renders from scenes that were inspired or replicated from real-world industrial areas, such as: BMW's Regensburg and Spartanburg plants, in addition to offices, warehouses, storage rooms, and tool shops. All images are in 720p and are associated with bbox 2D Tight annotations [73]. This release of SORDI.ai includes 36 datasets with a total of 1,191,893 annotated images for 111 annotated object classes. Additional information are found in

Appendix B.
Object distractors: We did not include any distractors, considering that our scenes are rich and dense in assets since they are inspired by real-world compositions and structures. Thus, by randomly moving the capture camera within the scene (FRC, CRC, and SqC), a natural occlusion behavior manifests by the closest industrial 3D models.

4.2 SORDI.ai Assets

We classified our annotated assets into five main clusters: logistic, transportation, tools, signage, and office:

1. **Logistic** assets are used within industrial operations to, for e.g., carry, hold, and store items, such as containers, boxes, storage equipment, robots. In addition, this cluster includes safety and environment setup objects like barriers, safety cones, etc.
2. **Transportation** assets consists of wheeled objects, mainly used for object transportation like: jack, smart transport robot, dolly, forklift, etc. or person transportation like bicycle, scooter, etc.
3. **Tools** are commonly found in a tool shop like hammer, pliers, wrench, etc. They also include safety tools like gloves, goggles, fire extinguisher, etc.
4. **Signage** assets include floor markings, signs, and logos.
5. **Office** assets include office stationary equipment, electronic devices and office furniture including kitchen items.

Furthermore, we assigned different annotations for the same object with different shapes or states, for e.g., different KLT box types (dimensions) or stillage states (open, and closed) as previously explained in Sect. 3.1.

4.3 Asset-Scene Correlation

In Figure 7, we visualized the asset distribution over the 5 asset clusters (logistic, transportation, tools, signage, and office) for each SORDI.ai dataset. Such distribution reflects the content/focus of the scene that we used to render images. For e.g., Office, Industrial Scene_7, and Industrial Scene_9 show significant usage of office assets (17, 22, and 13 resp.), showing that the scene contains office-inspired areas. Moreover, Rengensburg and Spartanburg scenes majorly contain logistic and transportation assets associated with assembly lines and production areas. In addition, the toolshop dataset presents clearly the usage of tool assets (15) compared to other datasets. For more details, we presented some dataset samples in Appendix D.

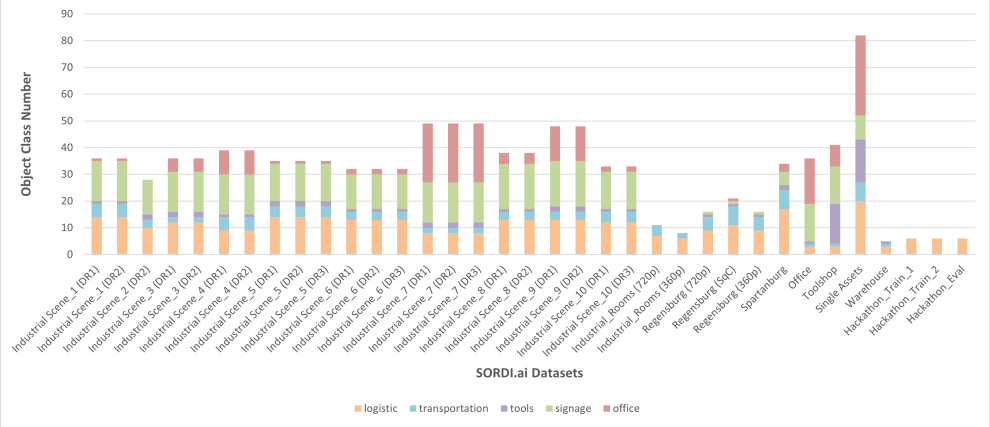


Fig. 7 SORDI.ai datasets asset distribution

5 Dataset Performance & Ablation Experiments

In this section, we conducted two sets of experiments to demonstrate how the DR techniques applied to SORDI.ai gradually bridge the reality gap. For evaluation purposes, we used the mean Average Precision (mAP) at an Intersection over Union (IoU) threshold equal to 0.5 (mAP@0.5). We conducted our experiments on an NVIDIA DGX server with 8 A100 40GB GPUs using the following training and evaluation tools [75, 76].

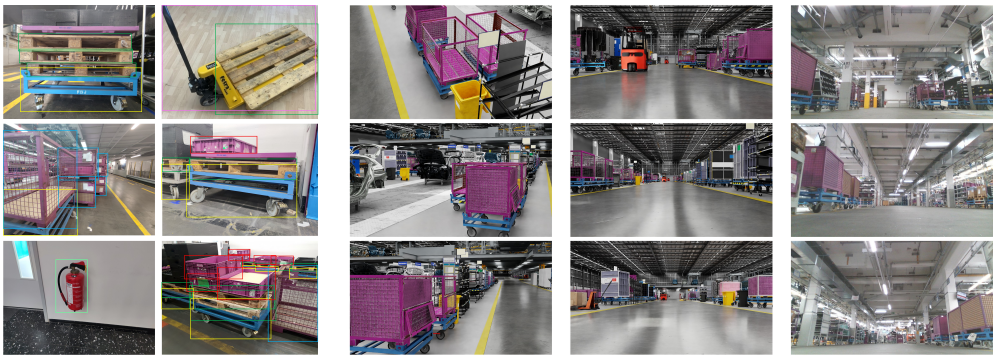


Fig. 8 Samples of the datasets used in (a) testing Experiment 1 with bbox: green: pallet, yellow: dolly, pink: jack, red: KLT box, blue: stillage, and aqua: fire extinguisher - and (b) training/testing Experiment 2: b1. CRC synthetic - b2. SqC synthetic - b3. SqC Real

5.1 Experiment 1: Bridging domain gaps

In this experiment, we gradually showcase how DR bridges the gap between domains, and we indicate the efficient DR base level for industrial object detection use cases.
DL architectures: We selected the MS COCO pre-trained Faster-RCNN (FRCNN) Resnet-101 [77], and EfficientDet D1 [78] models from TensorFlow-2 Model Zoo [79], fine-tuned it with SORDI.ai, and inferred over real/synthetic images captured in industrial setups. We used an 80-20% training-validation split ratio, 0.001 learning rate, and a batch size of 4 for 20 epochs. Furthermore, we employed YOLOv7 [80] with a training-validation split ratio of 80-20%, a learning rate equal to 0.01, a batch size of 8, and 10 epochs.
Data setup: We assess the performance of 4 SORDI.ai datasets characterized respectively with 4 levels of randomizations: $\text{Train}_{\text{No-DR}}$ [13], $\text{Train}_{\text{DR-1}}$, $\text{Train}_{\text{DR-2}}$, and $\text{Train}_{\text{DR-3}}$ (check Table 2). Furthermore, we rendered DR-1, DR-2, and DR-3 datasets using the same calibrated simulation scenes for the sake of comparison. However, the scenes used in the training set are not the same in the evaluation set: Each training dataset consists of 30,000 images. As for the evaluation datasets, we used 7 different datasets with 200 images each: $\text{Eval}_{\text{No-DR}}$, $\text{Eval}_{\text{DR-1}}$, $\text{Eval}_{\text{DR-2}}$, $\text{Eval}_{\text{DR-3}}$, $\text{Eval}_{\text{Regens.}}$, $\text{Eval}_{\text{Spartan.}}$, $\text{Eval}_{\text{Real}}$. We presented the dataset details in Table 3 below. Last but not least, we consider 6 industrial assets: KLT box (abstract class), dolly, pallet, stillage, fire extinguisher, and jack.

Table 3 Dataset details for Experiment 1

	Dataset	Size	Source	Description
Training	$\text{Train}_{\text{No-DR}}$	30K	[13]	Static scenes without randomization
	$\text{Train}_{\text{DR-1}}$	30K	Scenes: 5, 6, and 7	DR-1 randomization (asset, and position)
	$\text{Train}_{\text{DR-2}}$	30K	Scenes: 5, 6, and 7	DR-2 randomization (background)
	$\text{Train}_{\text{DR-3}}$	30K	Scenes: 5, 6, and 7	DR-3 randomization (light)
Evaluation	$\text{Eval}_{\text{No-DR}}$	200	[13]	Static scenes without randomization
	$\text{Eval}_{\text{DR-1}}$	200	Scenes: 1, 3, and 4	DR-1 randomizationn (asset, and position)
	$\text{Eval}_{\text{DR-2}}$	200	Scenes: 8, and 9	DR-2 randomization (background)
	$\text{Eval}_{\text{DR-3}}$	200	Scene: 10	DR-3 randomization (light)
	$\text{Eval}_{\text{Regens.}}$	200	Regens.	Dense scene
	$\text{Eval}_{\text{Spartan.}}$	200	Spartan.	Dense scene
	$\text{Eval}_{\text{Real}}$	200	*Captured (Fig. 8 a)*	Real hand annotated images from industrial areas

Additional notes: The calibrated simulation represents a first, single instance of the replicated environment. It is on a calibrated simulation where we apply DR to expand our scene variations and reduce the reality gap [81].
A dense scene is a crowded environment with extensive assets manifesting high level of occlusions. Since, labels are automatically and accurately generated to the pixel-level, far assets and areas up to 1px are annotated. Such annotations are unrecognizable by the model. Hence, for evaluation purposes only, we discarded them.
The real evaluation images were captured at random camera angles and in different locations - similarly to CRC (check Fig. 8 a).

Results: We presented this experiment's results in Table 4. Consequently, we conclude the following: (1) In $\text{Eval}_{\text{No-DR}}$, $\text{Eval}_{\text{DR-1}}$, $\text{Eval}_{\text{DR-2}}$, and $\text{Eval}_{\text{DR-3}}$ of our 3 used DL architectures, we notice that the maximum mAP is near the matrix's diagonal (where the training and evaluation datasets have the same DR level). This observation demonstrates that decreasing the domain gab yields better accuracies: In the upper triangular matrix (above the diagonal), we notice in each column the mAP's boost and gradual increase while adding additional DR levels until we reach a similar synthetic domain. (2) We notice that training on minimum DR-2 achieves the highest accuracies when inferring over $\text{Eval}_{\text{Regens.}}$, $\text{Eval}_{\text{Spartan.}}$, and $\text{Eval}_{\text{Real}}$. That is because factories are dense environments due to machinery, storage, and assembly line complex equipment, so the model has to be familiar with extreme background variations endorsing the efficiency of DR-2's wall texture randomization.

We presented in Table 7, and 6 additional details per object class and other evaluation metrics like Accuracy, Precision, F1-Score, and Recall results.

5.2 Experiment 2: Mixing domains

In this experiment, we study the effect of camera viewport randomization on bridging the reality gap, and the boosting effect of synthetic data to increase detection accuracies in low real data regimes.

DL architectures: We selected the MS COCO pre-trained Faster-RCNN (FRCNN) Resnet-101[77] model from TensorFlow-2 Model Zoo [79], fine-tuned it with SORDI.ai using an 80-20% training-validation split ratio, 0.001 learning rate, and a batch size of 4 for 20 epochs. Then, we inferred over 200 real hand-annotated images captured in industrial setups.

Data setup: We consider 2 SORDI.ai datasets for the same Regensburg dense scene, captured in CRC and SqC (check Fig. 8 b1, b2, and Table B1). Each dataset contains 8,800 synthetic images and consists of 2 industrial assets: dolly, and stillage. However, we gradually mixed both datasets with a step of 10%. For instance, a dataset with a mix ratio of 10% represents a dataset composed of 10% SqC (880 images) and 90% CRC (7,920 images), and so on. In total, we trained on 11 datasets with 8,800 images each. The models are evaluated on 200 hand-labeled real images captured from the viewpoint of an AMR inside industrial areas as shown in Fig. 8 b3. Moreover, we conducted a comparative study by gradually assessing detection models trained using mixtures of synthetic datasets ($\text{SqC}_{\text{Synth}}$, and the best $\text{CRC+SqC}_{\text{Synth}}$ combination) with real image dataset SqC_{Real} versus the real images dataset SqC_{Real} itself to examine the capability of our synthetic dataset in boosting the detection model accuracies in low (real) data regimes.

CRC vs $\text{SqC}_{\text{Synth}}$: In Table 5's first part ($\text{CRC+Synth}_{\text{Synth}}$), we gradually evaluated the dataset mixture, and we noticed that the mAP has significantly boosted from 24.63% (r = 0%) to 32.59% after integrating 30% of the SqC dataset. However, training on the SqC dataset, by itself (mAP = 29.90%, r = 100%), does not lead to the maximum accuracy (mAP_{max} = 36.98%, r_{max} = 90%). In other terms, including approx. 30% of a new domain dataset to our training is capable of boosting the model performance and generalizes better.

Sim2Real visual inspection: In Fig. 8.b2 and b3, we notice common features in terms of image content, as both sim and real environments have corridors with dollies, stillages, industrial racks, etc. on both sides. Additionally, the ceiling is similarly structured. Despite the consistency of camera motion, we notice that the real camera in b3 is more tilted up compared to the virtual camera in b2. Moreover, the real images are less saturated and present some motion blur and depth of field effects compared to the rendered images which are stabilized, sharper, and cleaner (without any real grain).
Synthetic vs Real: In the second part of Table 5, we gradually evaluated the mixture of synthetic data with real data (8,800 images) versus real data with the same size only (max. 4,400 images), e.g., at r = 10%, we compared models trained on 7,920 synthetic and 880 real images versus a model trained on 880 real images only. We conducted this experiment up to r = 50% due to the limitation of acquiring real data. We noticed that (1) our synthetic data significantly boosted the mAP for ratios up to 10% which is equivalent to 880 real images, and (2) adding real images to the training set yielded higher accuracies than training on synthetic data solely (r = 0%). Similarly to the previous analysis, we remarked that considering CRC data resulted in higher accuracies than using $\mathrm{SqC}_{\mathrm{Synth}}$ data only with real images. Last but not least, in all columns, we observed an mAP convergence [82] after r = 30%.

Table 4 mAP@0.5 evaluation for models trained using different DR levels

	mAP@0.5	$\text{Eval}_{\text{No-DR}}$	$\text{Eval}_{\text{DR-1}}$	$\text{Eval}_{\text{DR-2}}$	$\text{Eval}_{\text{DR-3}}$	$\text{Eval}_{\text{Regens.}}$	$\text{Eval}_{\text{Spartan.}}$	$\text{Eval}_{\text{Real}}$
FRCNN Resnet-101	$\text{Train}_{\text{No-DR}}$	**87.61**	37.07	22.57	26.01	33.57	30.90	33.12
	$\text{Train}_{\text{DR-1}}$	61.09	56.64	47.14	36.42	44.15	26.32	42.86
	$\text{Train}_{\text{DR-2}}$	59.40	**58.67**	**51.44**	40.29	**48.03**	**39.11**	**47.34**
	$\text{Train}_{\text{DR-3}}$	61.53	57.98	51.16	**42.26**	47.37	38.56	43.96
YOLOv7	$\text{Train}_{\text{No-DR}}$	**85.48**	32.65	25.27	16.82	44.26	32.07	21.18
	$\text{Train}_{\text{DR-1}}$	59.74	60.08	48.70	30.97	52.59	34.55	24.22
	$\text{Train}_{\text{DR-2}}$	61.26	**63.13**	**58.53**	39.41	54.44	41.26	33.04
	$\text{Train}_{\text{DR-3}}$	61.73	61.65	57.16	**44.84**	**54.54**	**41.35**	**40.70**
Efficient-Det D1	$\text{Train}_{\text{No-DR}}$	**78.07**	26.47	13.80	15.48	26.78	15.39	28.49
	$\text{Train}_{\text{DR-1}}$	52.49	**48.64**	36.80	26.71	34.08	26.34	35.09
	$\text{Train}_{\text{DR-2}}$	55.09	47.57	**42.36**	32.43	35.30	33.91	38.69
	$\text{Train}_{\text{DR-3}}$	53.68	47.20	40.63	**34.65**	**39.94**	**39.67**	**42.00**

Table 5 AP@0.5 and mAP@0.5 evaluation for FRCNN Resnet-101 models trained using different combinations of camera viewpoint randomization (CRC, $\text{SqC}_{\text{Synth}}$), and real data ($\text{SqC}_{\text{Real}}$)

	$\text{CRC+SqC}_{\text{Synth}}$				$\text{SqC}_{\text{Synth}}\text{+SqC}_{\text{Real}}$			SqC_{Real}			$(\text{CRC+SqC}_{\text{Synth}})_{\text{r=90}}\text{+SqC}_{\text{Real}}$		
r	AP_{Dolly}	$\text{AP}_{\text{Stillage}}$	mAP	r	AP_{Dolly}	$\text{AP}_{\text{Stillage}}$	mAP	AP_{Dolly}	$\text{AP}_{\text{Stillage}}$	mAP	AP_{Dolly}	$\text{AP}_{\text{Stillage}}$	mAP
0	21.85	27.41	24.63	0	26.37	33.43	29.90	-	-	-	**33.22**	**40.74**	**36.98**
10	24.24	29.43	26.83	1	**38.17**	43.47	40.82	17.03	0.03	8.53	36.78	**47.57**	**42.17**
20	23.96	33.88	28.92	2	37.02	51.29	44.15	31.79	25.23	28.51	**38.36**	**52.05**	**45.20**
30	27.78	37.26	32.59	4	35.83	52.46	44.14	38.70	41.14	39.92	**39.11**	**53.72**	**46.42**
40	30.22	40.42	35.32	6	**41.22**	**52.28**	**46.75**	43.38	40.67	42.02	41.09	49.18	45.13
50	25.80	42.81	34.30	8	**45.82**	52.94	49.38	37.40	50.58	43.99	45.52	**56.51**	**51.01**
60	30.01	**42.82**	36.42	10	39.30	**54.95**	49.26	**49.51**	47.83	48.67	45.46	54.59	**50.02**
70	31.57	41.80	36.68	20	45.03	55.59	50.31	**47.11**	**56.31**	**51.71**	**47.11**	55.33	51.22
80	28.90	42.04	35.47	30	45.12	58.03	51.58	48.19	**61.36**	**54.78**	**48.59**	59.71	54.15
90	**33.22**	40.74	**36.98**	40	47.84	60.69	54.26	**49.09**	60.87	**54.98**	47.84	**61.56**	54.70
100	26.37	33.43	29.90	50	50.73	57.94	54.34	48.80	**61.42**	**55.11**	**51.61**	58.53	55.07

Table 6 Additional evaluation details for models trained using different DR levels (Accuracy@0.5, Precision@0.5, F1-Score@0.5, and Recall@0.5)

		FRCNN Resnet 101							**EfficientDet D1**						
		$Eval_{No\text{-}DR}$	$Eval_{DR\text{-}1}$	$Eval_{DR\text{-}2}$	$Eval_{DR\text{-}3}$	$Eval_{Regens.}$	$Eval_{Spartan}$	$Eval_{Real}$	$Eval_{No\text{-}DR}$	$Eval_{DR\text{-}1}$	$Eval_{DR\text{-}2}$	$Eval_{DR\text{-}3}$	$Eval_{Regens.}$	$Eval_{Spartan}$	$Eval_{Real}$
Accuracy	$Train_{No\text{-}DR}$	**78.21**	30.43	24.7	26.44	23.36	16.84	24.86	**70.06**	21.74	19.62	20.47	23.94	14.72	24.99
	$Train_{DR\text{-}1}$	59.34	49.93	44.07	37.12	26.23	15.12	30.71	54.01	34.32	35.56	30.01	29.49	23.55	31.75
	$Train_{DR\text{-}2}$	58.15	49.77	**47.62**	39.06	29.16	**21.86**	**35.58**	53.09	32.94	**39.01**	34.84	36.85	26.75	31.93
	$Train_{DR\text{-}3}$	59.98	**49.99**	47.50	**42.16**	**29.23**	18.44	34.85	52.82	**34.39**	38.84	**37.61**	**38.97**	**29.87**	**34.54**
Precision	$Train_{No\text{-}DR}$	**89.16**	62.62	51.25	60.18	34.40	24.04	38.97	**89.99**	76.67	68.09	70.37	44.69	29.46	45.37
	$Train_{DR\text{-}1}$	80.73	79.92	73.80	73.50	34.84	19.31	44.19	81.75	**88.38**	82.61	83.16	45.92	38.83	57.39
	$Train_{DR\text{-}2}$	82	**82.13**	**78.92**	78.05	**38.82**	**29.68**	**56.94**	81.85	87.54	84.60	87.38	58.73	46.35	59.49
	$Train_{DR\text{-}3}$	81.83	80.58	77.59	**79.19**	38.67	22.91	56.43	79.74	88.17	**86.95**	**89.62**	**61.31**	**49.52**	**62.75**
F1-Score	$Train_{No\text{-}DR}$	**87.77**	46.66	39.61	41.82	37.87	28.83	39.82	**82.4**	35.72	32.80	33.98	38.64	25.66	39.98
	$Train_{DR\text{-}1}$	74.48	66.60	61.18	54.14	41.56	26.26	46.99	70.14	51.10	52.46	46.17	45.55	38.12	48.20
	$Train_{DR\text{-}2}$	73.54	66.47	**64.52**	56.17	45.15	**35.87**	**52.48**	70.04	49.56	**56.12**	51.68	53.85	42.21	48.40
	$Train_{DR\text{-}3}$	74.98	**66.66**	64.40	**59.31**	**45.24**	31.13	51.69	69.13	**51.18**	55.95	**54.66**	**56.09**	**46.00**	**51.35**
Recall	$Train_{No\text{-}DR}$	**86.43**	37.18	32.28	32.05	42.12	36.00	40.71	**75.98**	23.28	21.61	22.40	34.03	22.73	35.74
	$Train_{DR\text{-}1}$	69.13	**57.09**	52.24	42.86	51.50	41.03	50.17	61.41	35.94	38.44	31.96	45.19	37.44	41.54
	$Train_{DR\text{-}2}$	66.67	55.82	54.56	43.88	53.95	45.33	**48.67**	61.21	34.56	**41.99**	36.69	49.72	38.76	40.80
	$Train_{DR\text{-}3}$	69.19	56.83	**55.05**	**47.41**	**54.51**	**48.56**	47.68	61.01	**36.05**	41.24	**39.32**	**51.69**	**42.94**	**43.45**

		YOLOv7						
		$Eval_{No\text{-}DR}$	$Eval_{DR\text{-}1}$	$Eval_{DR\text{-}2}$	$Eval_{DR\text{-}3}$	$Eval_{Regens.}$	$Eval_{Spartan}$	$Eval_{Real}$
Accuracy	$Train_{No\text{-}DR}$	**81.96**	40.93	35.84	23.73	26.61	18.95	23.50
	$Train_{DR\text{-}1}$	62.70	**65.68**	58.86	37.82	29.99	18.58	31.88
	$Train_{DR\text{-}2}$	63.13	65.52	**64.26**	44.69	**39.92**	**25.02**	35.06
	$Train_{DR\text{-}3}$	63.8	**65.68**	63.39	**50**	37.04	19.83	**40.98**
Precision	$Train_{No\text{-}DR}$	**96.29**	81.66	72.15	78.30	39.02	28.08	51.49
	$Train_{DR\text{-}1}$	87.63	94.20	90.87	87.92	39.30	24.71	70.08
	$Train_{DR\text{-}2}$	88.45	**95.29**	**95.24**	94.40	**55.26**	**36.94**	**76.79**
	$Train_{DR\text{-}3}$	89.12	94.48	94.24	**95.58**	49.24	25.30	75.94
F1-Score	$Train_{No\text{-}DR}$	**90.08**	58.09	52.77	38.35	42.04	31.87	38.06
	$Train_{DR\text{-}1}$	77.08	79.28	74.10	54.88	46.14	31.33	48.34
	$Train_{DR\text{-}2}$	77.40	79.17	**78.24**	61.78	**57.06**	**40.02**	51.92
	$Train_{DR\text{-}3}$	77.90	**79.29**	77.59	**66.67**	54.05	33.10	**58.14**
Recall	$Train_{No\text{-}DR}$	**84.63**	45.08	41.6	25.4	45.55	36.84	30.18
	$Train_{DR\text{-}1}$	68.80	**68.45**	62.56	39.89	55.86	42.82	36.9
	$Train_{DR\text{-}2}$	68.80	67.72	**66.40**	45.91	58.98	43.66	39.22
	$Train_{DR\text{-}3}$	69.19	68.31	65.95	**51.18**	**59.90**	**47.85**	**47.10**

Table 7 Experiment 1: AP@0.5 of FRCNN Resnet-101 based models tested on $\text{Eval}_{\text{Real}}$ per object class

		$\text{Eval}_{\text{Pallet}}$	$\text{Eval}_{\text{Dolly}}$	$\text{Eval}_{\text{KLT Box}}$	$\text{Eval}_{\text{Stillage}}$	$\text{Eval}_{\text{Jack}}$	$\text{Eval}_{\text{Fire Ext.}}$
FRCNN Resnet-101	$\text{Train}_{\text{No-DR}}$	25.57	39.78	11.01	58.79	13.64	**49.89**
	$\text{Train}_{\text{DR-1}}$	31.99	50.18	**27.22**	58.21	71.87	17.68
	$\text{Train}_{\text{DR-2}}$	33.94	**54.28**	24.84	**62.98**	**86.27**	21.72
	$\text{Train}_{\text{DR-3}}$	**36.42**	50.18	25.91	56.98	82.07	12.21
YOLOv7	$\text{Train}_{\text{No-DR}}$	5.55	30.22	26.88	22.76	19.44	22.22
	$\text{Train}_{\text{DR-1}}$	10.66	44.03	44.65	32.47	0.00	13.51
	$\text{Train}_{\text{DR-2}}$	12.87	**48.92**	41.83	49.80	9.30	**35.48**
	$\text{Train}_{\text{DR-3}}$	**22.34**	46.9	**49.05**	**55.66**	**42.11**	28.13
EfficientDet D1	$\text{Train}_{\text{No-DR}}$	10.65	23.88	29.76	47.83	21.33	**37.50**
	$\text{Train}_{\text{DR-1}}$	**16.48**	46.01	**32.92**	50.87	45.97	18.30
	$\text{Train}_{\text{DR-2}}$	7.76	53.43	32.07	55.67	67.65	15.58
	$\text{Train}_{\text{DR-3}}$	14.77	**54.85**	32.27	**56.91**	**82.16**	11.05

6 Additional Details: Object Representation

In this section, we will provide more details related to Experiment 1 in Section 5.1.

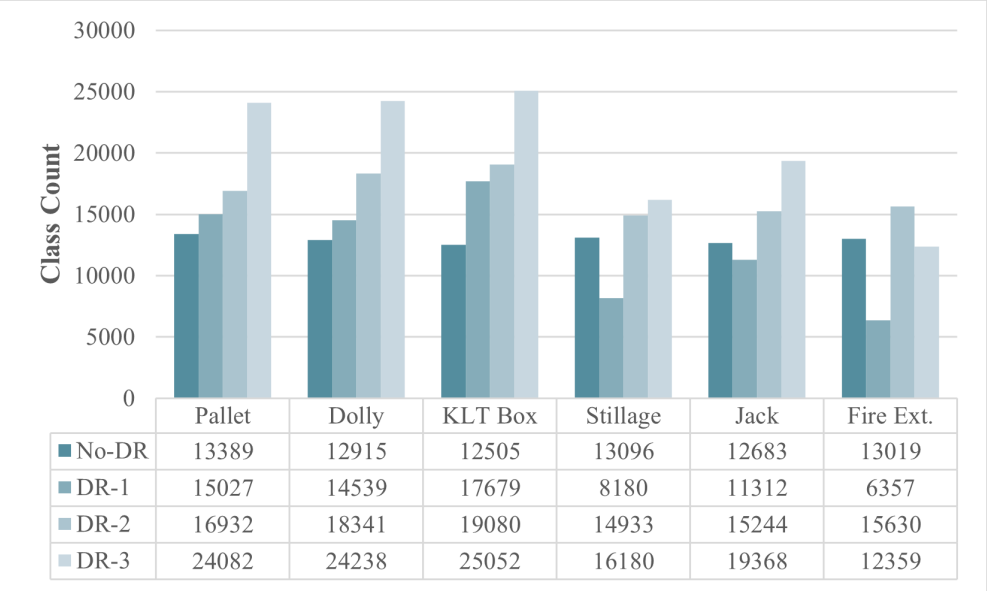
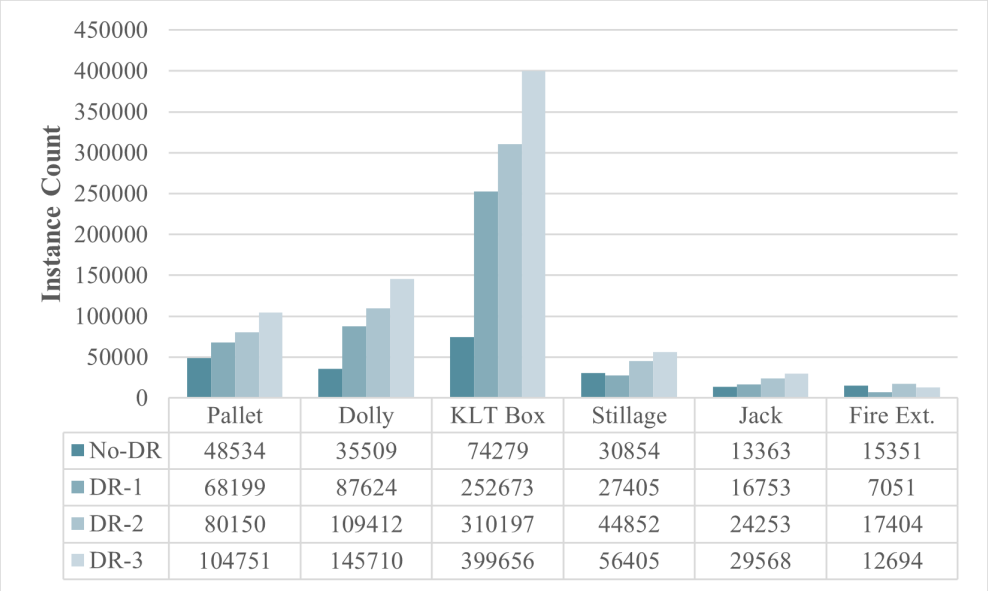


Fig. 9 Class count of the training datasets



	Pallet	Dolly	KLT Box	Stillage	Jack	Fire Ext.
No-DR	48534	35509	74279	30854	13363	15351
DR-1	68199	87624	252673	27405	16753	7051
DR-2	80150	109412	310197	44852	24253	17404
DR-3	104751	145710	399656	56405	29568	12694

Fig. 10 Instance count of the training datasets

Asset distribution: In Figure 6 (b), we notice that the asset instance occurrence in the 4 training datasets, is not balanced, this is mostly due to the reason that not all assets are equally available in an industrial area based on their functional role [13]. For instance, in a single industrial room, one or two jacks could cover the area to transport pallets. However, KLT boxes are significantly and extensively used since they carry various smaller manufacturing parts that are needed in multiple production steps. Thus, KLT boxes can be flexibly and easily placed in many places: on industrial shelves or racks, stacked, next to the assembly lines, etc. Moreover, we remark that the pallet and dolly distributions are quasi-similar due to the pallet-dolly relationship defined by a dolly transporting a pallet.

Stackable assets: We found that the pallets and KLT boxes have the lowest AP compared to other assets, because the pallet and KLT box are horizontally and vertically stackable assets. Therefore, it is possible that multiple stacked instances are detected as a single object instance. Especially in our KLT box detection case, we considered the superclass KLT box, which includes different size KLT box children assets, so 2 or more smaller KLT boxes can be confused with a larger one as visualized in Figure A1's KLT box randomization. However, we noticed that applying DR increases the average precision (AP) of stackable assets.

Large assets: We noticed that large assets, like dolly, stillage, and jack, achieved higher AP since they are easier to detect and recognize, and there are no complex combinations or placements for them, mainly due to the fact that they are big in size

and heavy so they are not classified as flexible assets that can be easily manipulated.

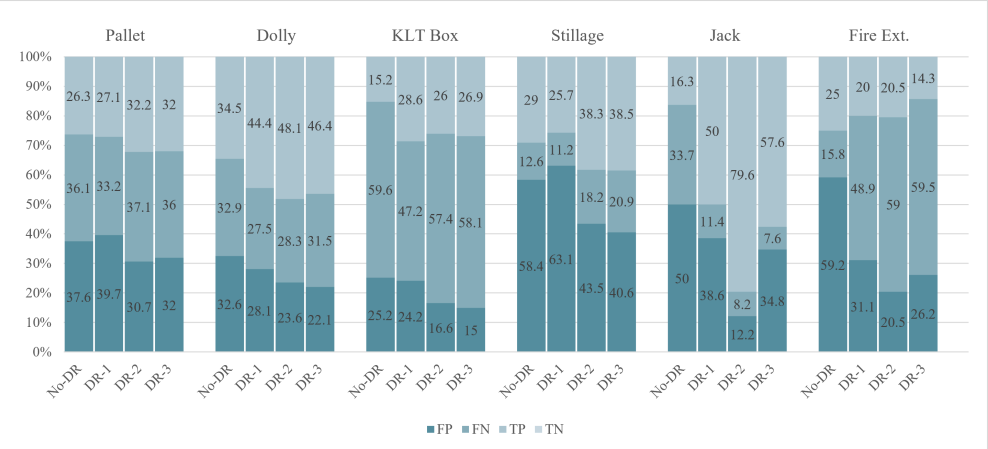


Fig. 11 False positive (FP), false negative (FN), true positive (TP), and true negative (TN) metrics of the FRCNN Resnet-101 models tested on $\text{Eval}_{\text{Real}}$ per class with an IoU threshold = 0.5

Natural occlusions and data imbalance: Due to the real-world inspiration to build our dense scenes, a multi-layer of assets exists. For this reason, annotated high-occluded, far (small) and background objects occur in the rendered images leading to false predictions as shown in Figure 11. For example, fire extinguishers are not randomly distributed, and they are mostly placed in specific places, such as hanging on the walls which forms a background of our rendered viewports. As a result, highly occluded fire extinguisher objects are present in the training set of DR-1, DR-2, and DR-3 as shown in Figure 12, in comparison to the No-DR scene which is less complex as shown in Figure D6. In addition, since we normally randomized the capture camera source and "look at" positions, the fire extinguisher instance does not equally appear as other frequent logistic assets that can be flexibly placed within the virtual scenes.

7 Limitations and Future Work

In this section, we discuss limitations of our proposed contribution, and therefore, possible future tasks to address them.

Scene updates: The first limitation is related to the manual work that is still necessary at the beginning of each simulation-based SDG to collect the 3D models, construct the calibrated simulation scenes, and define SDR setting. Furthermore, industrial and technological updates (new or modified) should be also integrated in the SDG pipeline, which is easier thanks to the USD interoperability and modularity. Nevertheless, it is worth looking into 3D scanning and digital twin methods to automatically build large-scale, structured, industrial calibrated simulations and to maintain their continuous relevance.

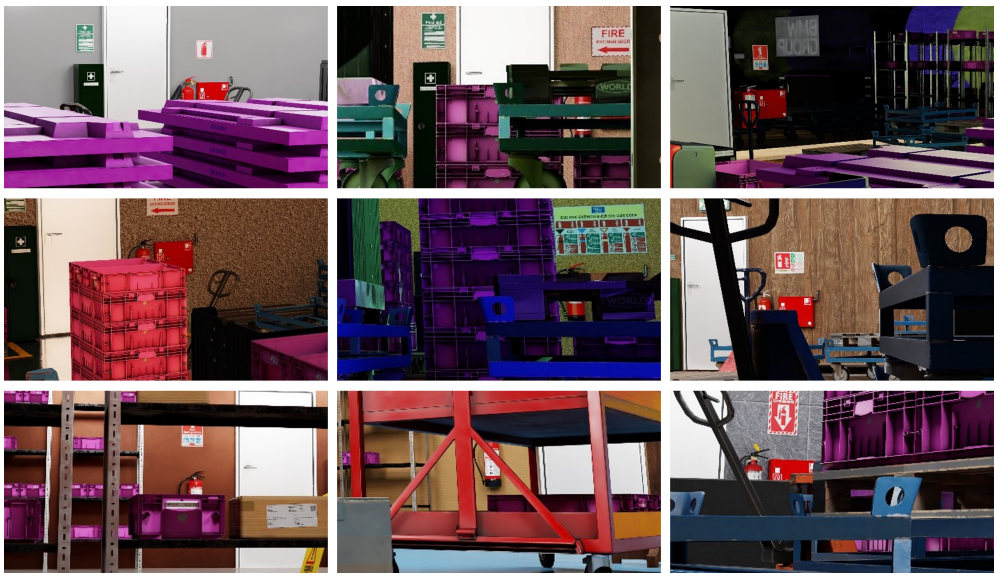


Fig. 12 Occluded background asset: fire extinguisher (zoomed in)

Domain randomization: As previously mentioned, an industrial area is a well-defined and structured area which is limited in variability. Hence, many areas share common patterns [48]. On the other hand, it is significant to configure adequate randomization parameters depending on the industrial use case. While some cases are characterized with minimum variability, some other cases may contain higher variations. Therefore, as future task we propose defining DR strength (low, moderate, high) for each of the DR levels and to automate the process of defining structured randomizations featuring physics simulation powered by NVIDIA PhysX SDK [83].
Scene variability: Furthermore, to increase our dataset intra and inter variabilities, future implementations may cover (1) new DR configurations like optical DR related to the camera settings, realistic images as textures in DR-2 and DR-3, and (2) new components like interactive human workers, new industrial objects and updated textures.
Larger-scale scenes: Subsequently, our USD-based scalable and modular (layered) SDG pipeline provides an easy upgrade and update of the calibrated simulation scenes by re-using/spawning pre-configured randomized assets and digital stages (check Fig. 4). Thus, it is possible to consider new environments, scenarios and content, resulting in even larger industrial and complex scenes. Nevertheless, rendering photo-realistic images with NVIDIA Omniverse requires expensive hardware and GPUs like RTX 3090 and RTX A6000. Hence, the necessity of looking into optimization methods to load large scale scenes, and to render complex images. Additionally, it is worth checking Generative AI methods to enhancing synthetic data realism and quality [84–89].
Synthetic annotations: A 2D Tight bbox covers the visible part of a labeled object

only. However, in case of occlusions, one or more pixels can still possibly exist near the edges (visually, hard to perceive), or through a perforated front object. These "leaking" pixels result in an "expanded" bbox. Additionally, we did not filter any small bounding boxes associated with far background assets (except zero-area bboxes), because the cleaning thresholds of a depth-based or area-based methods may be customized per object class. On another hand, calculating an occlusion ratio based on the percentage of bounding box overlap may be inefficient when the object's actual segmentation area is significantly less than the bbox area. For instance, if the front object is highly perforated (e.g., stillage, jack, or dolly), the back object remains clearly visible. Therefore, future experiments may address (1) refining "expanded" bboxes, and (2) bbox filtering strategies.

Multi-modal data: We are working on considering multi-modal annotations as shown in Figure 13: E.g., semantic/instance segmentation for image segmentation, or depth images for obstacle avoidance and depth estimation tasks [73]. Providing multi-modal data extends the industrial coverage of SORDI.ai to solve more complex industrial use cases.

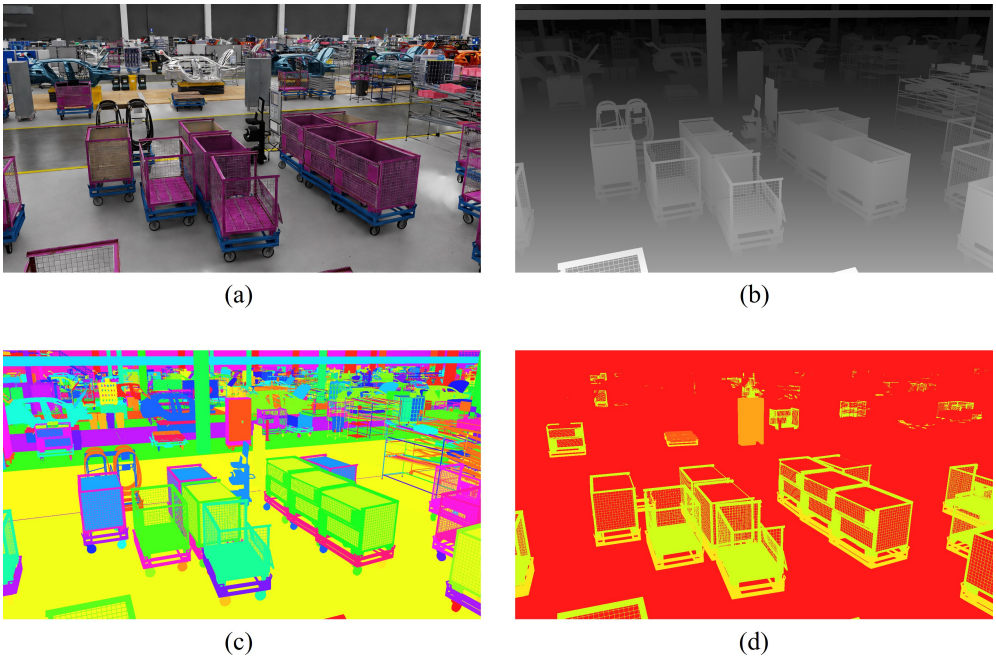


Fig. 13 (a) Plain color path-trace rendered, (b) depth, (c) instance segmentation, and (d) semantic segmentation image

Model sensitivity and evaluation: Last but not least, we noticed that YOLOv7 and EfficientDet-D1 models achieved their best accuracies on $\text{Eval}_{\text{Real}}$ with $\text{Train}_{\text{DR-3}}$ in comparison to FRCNN's models that got the maximum accuracies with $\text{Train}_{\text{DR-2}}$

which does not include any light randomization. Hence, further experimentations related to model sensitivity on domain randomization are worth to investigate, in addition to exploring new evaluation metrics and methods to assess the domain gap.

8 Conclusion

In this paper, we leveraged our previous work in industrial SDG [13] by implementing USD's features of flexibility, interoperability and modularity: We built an extensive PBR 3D models library of industrial assets that are compliant with the VDA standards. Additionally, twisted with SDR, we bridged the reality gap. Moreover, we proposed a 5-layer asset taxonomy, a modular 9-layer scene composition, and a 4-step SDG pipeline for large-scale scene construction. As a result, we presented SORDI.ai, a synthetic industrial dataset with over a million photorealistic images, path-trace rendered from 20 various manufacturing scenes, covering more than 100 annotated assets. Last but not least, we investigated how DR gradually bridges the reality gap and the efficiency of mixing DR levels in the training dataset to increase the detection mAP. We hope that this dataset will enable researchers in industrial AI/CV and robotics to reach their goals in training more generalized models that adapt to new industrial areas.

Declarations

Conflict of interest: The authors declare that they have no conflict of interest.

Data availabilty: Kindly check the official project website https://sordi.ai for synthetic dataset availability conditions.

References

[1] Kaveh Azadeh, René De Koster, and Debjit Roy. Robotized and automated warehouse systems: Review and recent developments. *Transportation Science*, 53 (4):917–945, 2019.

[2] Jérôme Rutinowski, Hazem Youssef, Anas Gouda, Christopher Reining, and Moritz Roidl. The potential of deep learning based computer vision in warehousing logistics. *Logistics Journal: Proceedings*, 2022(18), 2022.

[3] Alexander Naumann, Felix Hertlein, Laura Doerr, Steffen Thoma, and Kai Furmans. Literature review: Computer vision applications in transportation logistics and warehousing. *arXiv preprint arXiv:2304.06009*, 2023.

[4] Janis Arents and Modris Greitans. Smart industrial robot control trends, challenges and opportunities within manufacturing. *Applied Sciences*, 12(2):937, 2022.

[5] Longfei Zhou, Lin Zhang, and Nicholas Konz. Computer vision techniques in manufacturing. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2022.

[6] Stefan-Octavian Bezrucav, Nils Mandischer, and Burkhard Corves. Artificial intelligence task planning of cooperating low-cost mobile manipulators: A case study on a fully autonomous manufacturing application. *Procedia Computer Science*, 217:306–315, 2023.

[7] Sparsh Mittal and Shraiysh Vaishay. A survey of techniques for optimizing deep learning on gpus. *Journal of Systems Architecture*, 99:101635, 2019.

[8] Iqbal H Sarker. Deep learning: a comprehensive overview on techniques, taxonomy, applications and research directions. *SN Computer Science*, 2(6):420, 2021.

[9] Christian Janiesch, Patrick Zschech, and Kai Heinrich. Machine learning and deep learning. *Electronic Markets*, 31(3):685–695, 2021.

[10] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *Proceedings of the IEEE international conference on computer vision*, pages 843–852, 2017.

[11] Aria Salari, Abtin Djavadifar, Xiangrui Liu, and Homayoun Najjaran. Object recognition datasets and challenges: A review. *Neurocomputing*, 495:129–152, 2022.

[12] Jerone Andrews, Dora Zhao, William Thong, Apostolos Modas, Orestis Papakyriakopoulos, and Alice Xiang. Ethical considerations for responsible data curation. *Advances in Neural Information Processing Systems*, 36, 2024.

[13] Chafic Abou Akar, Jimmy Tekli, Daniel Jess, Mario Khoury, Marc Kamradt, and Michael Guthe. Synthetic object recognition dataset for industries. In *2022*

35th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI), volume 1, pages 150–155. IEEE, 2022.

[14] Morgane Ayle, Jimmy Tekli, Julia El-Zini, Boulos El-Asmar, and Mariette Awad. Bar—a reinforcement learning agent for bounding-box automated refinement. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 2561–2568, 2020.

[15] Jimmy Tekli, Bechara Al Bouna, Gilbert Tekli, and Raphaël Couturier. A framework for evaluating image obfuscation under deep learning-assisted privacy attacks. *Multimedia Tools and Applications*, 82(27):42173–42205, 2023.

[16] Jimmy Tekli, Bechara Al Bouna, Raphaël Couturier, Gilbert Tekli, Zeinab Al Zein, and Marc Kamradt. A framework for evaluating image obfuscation under deep learning-assisted privacy attacks. In *2019 17th International Conference on Privacy, Security and Trust (PST)*, pages 1–10. IEEE, 2019.

[17] Jérôme Rutinowski, Hazem Youssef, Sven Franke, Irfan Fachrudin Priyanta, Frederik Polachowski, Moritz Roidl, and Christopher Reining. Semi-automated computer vision-based tracking of multiple industrial entities: a framework and dataset creation approach. *EURASIP Journal on Image and Video Processing*, 2024(1):8, 2024.

[18] Leon Eversberg and Jens Lambrecht. Generating images with physics-based rendering for an industrial object detection task: Realism versus domain randomization. *Sensors*, 21(23):7901, 2021.

[19] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 23–30. IEEE, 2017.

[20] Jonathan Tremblay, Aayush Prakash, David Acuna, Mark Brophy, Varun Jampani, Cem Anil, Thang To, Eric Cameracci, Shaad Boochoon, and Stan Birchfield. 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*, pages 969–977, 2018.

[21] Jonathan Tremblay, Thang To, and Stan Birchfield. Falling things: A synthetic dataset for 3d object detection and pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 2038–2041, 2018.

[22] Lukas Block, Adrian Raiser, Lena Schön, Franziska Braun, and Oliver Riedel. Image-bot: Generating synthetic object detection datasets for small and medium-sized manufacturing companies. *Procedia CIRP*, 107:434–439, 2022.

[23] Georgios Georgakis, Arsalan Mousavian, Alexander C Berg, and Jana Kosecka. Synthesizing training data for object detection in indoor scenes. *Robotics: Science and Systems (RSS)*, 2017.

[24] Jonas Dirr, Daniel Gebauer, Jiajun Yao, and Rüdiger Daub. Automatic image generation pipeline for instance segmentation of deformable linear objects. *Sensors*, 23(6):3013, 2023.

[25] Klaus Greff, Francois Belletti, Lucas Beyer, Carl Doersch, Yilun Du, Daniel Duckworth, David J Fleet, Dan Gnanapragasam, Florian Golemo, Charles Herrmann,

et al. Kubric: A scalable dataset generator. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3749–3761, 2022.

[26] Maximilian Denninger, Martin Sundermeyer, Dominik Winkelbauer, Dmitry Olefir, Tomas Hodan, Youssef Zidan, Mohamad Elbadrawy, Markus Knauer, Harinandan Katam, and Ahsan Lodhi. Blenderproc: Reducing the reality gap with photorealistic rendering. In *International Conference on Robotics: Sciene and Systems, RSS 2020*, 2020.

[27] Thang To, Jonathan Tremblay, Duncan McKay, Yukie Yamaguchi, Kirby Leung, Adrian Balanon, Jia Cheng, William Hodge, and Stan Birchfield. Ndds: Nvidia deep learning dataset synthesizer. In *CVPR 2018 Workshop on Real World Challenges and New Benchmarks for Deep Learning in Robotic Vision, Salt Lake City, UT, June*, volume 22, 2018.

[28] Steve Borkman, Adam Crespi, Saurav Dhakad, Sujoy Ganguly, Jonathan Hogins, You-Cyuan Jhang, Mohsen Kamalzadeh, Bowen Li, Steven Leal, Pete Parisi, et al. Unity perception: Generate synthetic data for computer vision. *arXiv preprint arXiv:2107.04259*, 2021.

[29] Pascalis Trentsios, Mario Wolf, and Detlef Gerhard. Overcoming the sim-to-real gap in autonomous robots. *Procedia CIRP*, 109:287–292, 2022.

[30] Wenshuai Zhao, Jorge Peña Queralta, and Tomi Westerlund. Sim-to-real transfer in deep reinforcement learning for robotics: a survey. In *2020 IEEE symposium series on computational intelligence (SSCI)*, pages 737–744. IEEE, 2020.

[31] Adrien Gaidon, Qiao Wang, Yohann Cabon, and Eleonora Vig. Virtual worlds as proxy for multi-object tracking analysis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4340–4349, 2016.

[32] Ole Schmedemann, Melvin Baaß, Daniel Schoepflin, and Thorsten Schüppstuhl. Procedural synthetic training data generation for ai-based defect detection in industrial surface inspection. *Procedia CIRP*, 107:1101–1106, 2022.

[33] Shaojing Fan, Tian-Tsong Ng, Bryan Lee Koenig, Jonathan Samuel Herberg, Ming Jiang, Zhiqi Shen, and Qi Zhao. Image visual realism: From human perception to machine computation. *IEEE transactions on pattern analysis and machine intelligence*, 40(9):2180–2193, 2017.

[34] Shaojing Fan, Tian-Tsong Ng, Jonathan S Herberg, Bryan L Koenig, Cheston Y-C Tan, and Rangding Wang. An automated estimator of image visual realism based on human cognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4201–4208, 2014.

[35] Stefan Grushko, Aleš Vysockỳ, Jakub Chlebek, and Petr Prokop. Hadr: Applying domain randomization for generating synthetic multimodal dataset for hand instance segmentation in cluttered industrial environments. *arXiv preprint arXiv:2304.05826*, 2023.

[36] Apostolia Tsirikoglou, Gabriel Eilertsen, and Jonas Unger. A survey of image synthesis methods for visual machine learning. In *Computer Graphics Forum*, volume 39, pages 426–451. Wiley Online Library, 2020.

[37] Fabio Muratore, Fabio Ramos, Greg Turk, Wenhao Yu, Michael Gienger, and Jan Peters. Robot learning from randomized simulations: A review. *Frontiers in Robotics and AI*, page 31, 2022.

[38] Raghad Alghonaim and Edward Johns. Benchmarking domain randomisation for visual sim-to-real transfer. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 12802–12808. IEEE, 2021.

[39] Sergey Zakharov, Rareș Ambruș, Vitor Guizilini, Wadim Kehl, and Adrien Gaidon. Photo-realistic neural domain randomization. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXV*, pages 310–327. Springer, 2022.

[40] Ville Kyrki and Oliver Struckmeier. Guided domain randomization with meta reinforcement learning, 2022.

[41] Fabio Muratore, Theo Gruner, Florian Wiese, Boris Belousov, Michael Gienger, and Jan Peters. Neural posterior domain randomization. In *Conference on Robot Learning*, pages 1532–1542. PMLR, 2022.

[42] Sergey Zakharov, Wadim Kehl, and Slobodan Ilic. Deceptionnet: Network-driven domain randomization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 532–541, 2019.

[43] Bhairav Mehta, Manfred Diaz, Florian Golemo, Christopher J Pal, and Liam Paull. Active domain randomization. In *Conference on Robot Learning*, pages 1162–1176. PMLR, 2020.

[44] Michael Dennis, Natasha Jaques, Eugene Vinitsky, Alexandre Bayen, Stuart Russell, Andrew Critch, and Sergey Levine. Emergent complexity and zero-shot transfer via unsupervised environment design. *Advances in neural information processing systems*, 33:13049–13061, 2020.

[45] Aayush Prakash, Shaad Boochoon, Mark Brophy, David Acuna, Eric Cameracci, Gavriel State, Omer Shapira, and Stan Birchfield. Structured domain randomization: Bridging the reality gap by context-aware synthetic data. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 7249–7255. IEEE, 2019.

[46] Amlan Kar, Aayush Prakash, Ming-Yu Liu, Eric Cameracci, Justin Yuan, Matt Rusiniak, David Acuna, Antonio Torralba, and Sanja Fidler. Meta-sim: Learning to generate synthetic datasets. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4551–4560, 2019.

[47] Jeevan Devaranjan, Amlan Kar, and Sanja Fidler. Meta-sim2: Unsupervised learning of scene structure for synthetic data generation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVII 16*, pages 715–733. Springer, 2020.

[48] Chafic Abou Akar, Andre Luckow, Ahmad Obeid, Christian Beddawi, Marc Kamradt, and Abdallah Makhoul. Enhancing complex image synthesis with conditional generative models and rule extraction. In *2023 International Conference on Machine Learning and Applications (ICMLA)*, pages 136–143. IEEE, 2023.

[49] Steven Moonen, Bram Vanherle, Joris de Hoog, Taoufik Bourgana, Abdellatif Bey-Temsamani, and Nick Michiels. Cad2render: A modular toolkit for gpu-accelerated photorealistic synthetic data generation for the manufacturing industry. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 583–592, 2023.

[50] Nathan Morrical, Jonathan Tremblay, Yunzhi Lin, Stephen Tyree, Stan Birchfield, Valerio Pascucci, and Ingo Wald. NViSII: A scriptable tool for photorealistic image generation. In *ICLR Workshop on Synthetic Data Generation*, May 2021.

[51] Kshitiz Gupta. Closing the Sim2Real Gap with NVIDIA Isaac Sim and NVIDIA Isaac Replicator. https://developer.nvidia.com/blog/closing-the-sim2real-gap-with-nvidia-isaac-sim-and-nvidia-isaac-replicator/, 2022. Online; accessed 17 May 2023.

[52] Omniverse, NVIDIA. Replicator. https://docs.omniverse.nvidia.com/prod_extensions/prod_extensions/ext_replicator.html, 2023. Online; accessed 18 May 2023.

[53] Xiaomeng Zhu, Talha Bilal, Pär Mårtensson, Lars Hanson, Mårten Björkman, and Atsuto Maki. Towards sim-to-real industrial parts classification with synthetic dataset. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2023.

[54] Xiaomeng Zhu. Synthetic Industrial Parts dataset (SIP-17). https://www.kaggle.com/datasets/mandymm/synthetic-industrial-parts-dataset-sip-17, 2023. Online; accessed 2 June 2023.

[55] Synthetic Corrosion. Synthetic corrosion dataset dataset. https://universe.roboflow.com/synthetic-corrosion/synthetic-corrosion-dataset, aug 2022. URL https://universe.roboflow.com/synthetic-corrosion/synthetic-corrosion-dataset. visited on 2023-06-02.

[56] Peter De Roovere, Steven Moonen, Nick Michiels, and Francis wyffels. Sim-to-real dataset of industrial metal objects. *Machines*, 12(2):99, 2024.

[57] Peter De Roovere, Steven Moonen, Nick Michiels, and Francis Wyffels. Dataset of Industrial Metal Objects. https://pderoovere.github.io/dimo/, 2023. Online; accessed 2 June 2023.

[58] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2223–2232, 2017.

[59] Aliaksei Petsiuk, Harnoor Singh, Himanshu Dadhwal, and Joshua M Pearce. Synthetic-to-real composite semantic segmentation in additive manufacturing. *Journal of Manufacturing and Materials Processing*, 8(2):66, 2024.

[60] Markus Knitt, Jakob Schyga, Asan Adamanov, Johannes Hinckeldeyn, and Jochen Kreutzfeldt. Estimating the pose of a euro pallet with an rgb camera based on synthetic training data. *arXiv preprint arXiv:2210.06001*, 2022.

[61] Markus Knitt, Jakob Schyga, Asan Adamanov, Johannes Hinckeldeyn, and Jochen Kreutzfeldt. Palloc6d-estimating the pose of a euro pallet with an rgb camera based on synthetic training data, 2022.

[62] Christopher Mayershofer, Tao Ge, and Johannes Fottner. Towards fully-synthetic training for industrial applications. In *LISS 2020: Proceedings of the 10th International Conference on Logistics, Informatics and Service Sciences*, pages 765–782. Springer, 2021.

[63] Pixar Animation Studios. Introduction to USD. https://openusd.org/release/intro.html, 2021. Online; accessed 18 May 2023.

[64] Pixar Animation Studios. Usdz File Format Specification. https://openusd.org/release/spec_usdz.html, 2021. Online; accessed 5 June 2023.

[65] Goran Paulin and Marina Ivasic-Kos. Review and analysis of synthetic dataset generation methods and techniques for application in computer vision. *Artificial Intelligence Review*, pages 1–45, 2023.

[66] Stefan Hinterstoisser, Olivier Pauly, Hauke Heibel, Marek Martina, and Martin Bokeloh. An annotation saved is an annotation earned: Using fully synthetic training for object detection. In *Proceedings of the IEEE/CVF international conference on computer vision workshops*, pages 0–0, 2019.

[67] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014.

[68] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Scharwächter, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset. In *CVPR Workshop on the Future of Datasets in Vision*, volume 2. sn, 2015.

[69] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[70] Berk Calli, Arjun Singh, Aaron Walsman, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M Dollar. The ycb object and model set: Towards common benchmarks for manipulation research. In *2015 international conference on advanced robotics (ICAR)*, pages 510–517. IEEE, 2015.

[71] NVIDIA Developer. vMaterials. https://developer.nvidia.com/vmaterials, 2023. Online; accessed 5 June 2023.

[72] NVIDIA Omniverse. Omniverse MDL Materials. https://docs.omniverse.nvidia.com/prod_materials-and-rendering/prod_materials-and-rendering/materials.html/, 2023. Online; accessed 18 May 2023.

[73] NVIDIA Omniverse. Annotators Information. https://docs.omniverse.nvidia.com/prod_extensions/prod_extensions/ext_replicator/annotators_details.html, 2023. Online; accessed 4 June 2023.

[74] BMW TechOffice MUNICH. LabelTool lite. https://github.com/BMW-InnovationLab/BMW-Labeltool-Lite, 2023. Online; accessed 5 June 2023.

[75] BMW TechOffice MUNICH. Tensorflow 2 Object Detection Training GUI for Linux. https://github.com/BMW-InnovationLab/BMW-TensorFlow-Training-GUI, 2023. Online; accessed 5 June 2023.

[76] BMW TechOffice MUNICH. BMW AI Evaluation GUI. https://github.com/BMW-InnovationLab/SORDI-AI-Evaluation-GUI, 2023. Online; accessed 5 June 2023.

[77] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[78] Mingxing Tan, Ruoming Pang, and Quoc V Le. Efficientdet: Scalable and efficient object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10781–10790, 2020.

[79] Tensorflow. TensorFlow 2 Detection Model Zoo. https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/tf2_detection_zoo.md, 2021. Online; accessed 5 June 2023.

[80] Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7464–7475, 2023.

[81] Lilian Weng. Domain Randomization for Sim2Real Transfer. https://lilianweng.github.io/posts/2019-05-05-domain-randomization/, 2019. Online; accessed 18 May 2023.

[82] Chafic Abou Akar, Anthony Semaan, Youssef Haddad, Marc Kamradt, and Abdallah Makhoul. Mixing domains for smartly picking and using limited datasets in industrial object detection. In *International Conference on Computer Vision Systems*, pages 270–282. Springer, 2023.

[83] NVIDIA Omniverse. Physics Core. https://docs.omniverse.nvidia.com/extensions/latest/ext_physics.html, 2023. Online; accessed 11 August 2023.

[84] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[85] Florinel-Alin Croitoru, Vlad Hondru, Radu Tudor Ionescu, and Mubarak Shah. Diffusion models in vision: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

[86] Zhengwei Wang, Qi She, and Tomas E Ward. Generative adversarial networks in computer vision: A survey and taxonomy. *ACM Computing Surveys (CSUR)*, 54(2):1–38, 2021.

[87] Chafic Abou Akar, Rachelle Abdel Massih, Anthony Yaghi, Joe Khalil, Marc Kamradt, and Abdallah Makhoul. Generative adversarial network applications in industry 4.0: A review. *International Journal of Computer Vision*, pages 1–60, 2024.

[88] Pourya Shamsolmoali, Masoumeh Zareapoor, Eric Granger, Huiyu Zhou, Ruili Wang, M Emre Celebi, and Jie Yang. Image synthesis with adversarial networks: A comprehensive survey and case studies. *Information Fusion*, 72:126–146, 2021.

[89] Salman Khan, Muzammal Naseer, Munawar Hayat, Syed Waqas Zamir, Fahad Shahbaz Khan, and Mubarak Shah. Transformers in vision: A survey. *ACM computing surveys (CSUR)*, 54(10s):1–41, 2022.

[90] Théo Jaunet, Guillaume Bono, Romain Vuillemot, and Christian Wolf. Visualizing the sim2real gap in robot ego-pose estimation. In *eXplainable AI approaches for debugging and diagnosis.*, 2021. URL https://openreview.net/forum?id=SkvZsABQXE.

[91] Apostolia Tsirikoglou, Joel Kronander, Magnus Wrenninge, and Jonas Unger. Procedural modeling and physically based rendering for synthetic data generation

in automotive applications. *arXiv preprint arXiv:1710.06270*, 2017.

[92] Dominik Schraml. Physically based synthetic image generation for machine learning: a review of pertinent literature. *Photonics and Education in Measurement Science 2019*, 11144:108–120, 2019.

[93] Tomáš Hodaň, Vibhav Vineet, Ran Gal, Emanuel Shalev, Jon Hanzelka, Treb Connell, Pedro Urbina, Sudipta N Sinha, and Brian Guenter. Photorealistic image synthesis for object instance detection. In *2019 IEEE international conference on image processing (ICIP)*, pages 66–70. IEEE, 2019.

[94] Omniverse, NVIDIA. Randomizer Tool. https://docs.omniverse.nvidia.com/prod_extensions/prod_extensions/ext_randomizer-tool.html, 2023. Online; accessed 18 May 2023.

Appendix A Preliminaries

In this section, we review briefly the reality gap concept and how to reduce it via IR and DR.

A.1 Reality gap

The real world is a complex domain including enormous and various combinations of environmental and behavioral factors where some of them are considered rare events that are hard to reproduce. As mentioned earlier, the reality gap in CV is the difference in performance between DL models trained on synthesized images versus real captured images [19, 29–31, 45, 90]. Furthermore, the reality gap can be classified as [36, 51]:

1. **Visual gap**, or appearance/perceptual gap, refers to the differences between a synthetic image and a realistic image in terms of image quality, colors, realism (especially with respect to the quality of the rendering system compared to a real camera sensor), as well as assets' shapes, materials, and details.
2. **Content gap** refers to the differences in terms of diversity, distribution, placement, composition, and behavior of objects between the virtual environment and the real world.

On the one hand, to bridge the visual gap, other studies enhanced IR by utilizing 3D components with advanced rendering capabilities such as **physically-based rendered (PBR) materials** besides **realistic textures** applied on **3D models featuring high levels of detail** and **physically-based light controls** [18]. Afterward, virtual scenes are rendered using **advanced algorithms** like path tracing or iRay and **powerful GPU hardware within multi-GPU architecture** along with **accurate camera models**. Moreover, these visual components can be randomized to generalize the model and adapt to new light conditions, and object status, e.g., high or low usability signs such as scratches, dust, etc., or new object versions with new paint color or materials, etc.
On the other hand, the content gap is related to scene composition: how the virtual scene is similar to the real world in terms of the asset's physical positions, functionalities, and behaviors. More details in the following Section A.3.

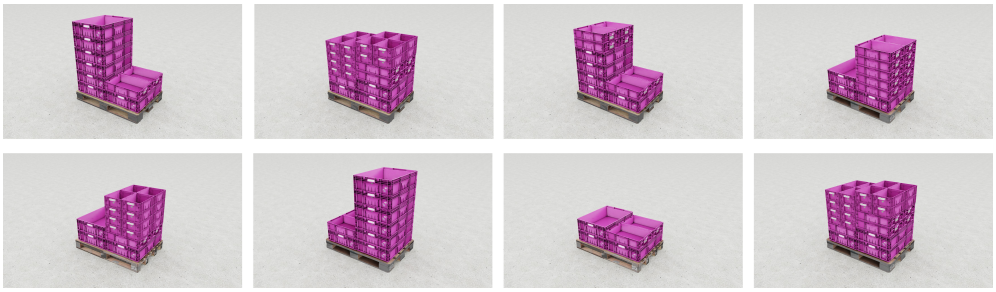


Fig. A1 SDR of three types of small load carrier stacked

A.2 IR Aspects

Several studies investigated and enhanced aspects of IR to bridge the visual gap. Tsirikoglou *et al.* identified five realism aspects: overall scene composition, geometric structure, illumination, material properties, and optical effects [91]. Many researchers focused on the importance of highly realistic 3D models that look very similar to real objects [13, 18, 60, 61] to reduce the visual gap. In addition, they endorse the physically-based rendered (PBR) approach to define the visual properties of these 3D models' surfaces such as color, roughness, metalness, etc. to define how any light interacts with the surfaces so it results in a feeling of various realistic surfaces such as wood, metal, glass, etc. [18, 32]. Therefore, rendering images with PBR yields better results especially when the light and scene conditions are more complex [92, 93]. As for the scene composition, we highlighted in a previous publication the importance of scene content creation based on real industrial scenarios and asset composition [13], and presented realistic models for eight industrial assets to reduce the reality gap.

A.3 DR Components

Rendering images from a single scene with static assets overfits the DL model to specific simulation instances. In turn, the DL model does not generalize well to real-world instances which is extensive in randomizations [37]. As a solution, DR is capable of leveraging all virtual scenes from a static into a **dynamic environment** where assets are **spawn or hidden** in respect to **physical, logical and semantic constraints** and at **different positions** of the 3D scene. Furthermore, knowing the functionality and behavior of each object, it is possible to avoid unrealistic scenarios if the randomization is set up via a knowledge-based approach. This lead to "Structured DR (SDR)" [45] as shown in Figure A1. In the rest of this paper and in our pipeline description, we refer to SDR as DR, since all assets in SORDI.ai are spawned in a structured way. In this section, we list the four DR components [94] that we applied in our SDG pipeline:

1. **Textures & materials**: The texture defines a 3D model's (mesh) appearance and details, e.g., scratches, patterns, colors, bumps, etc., whereas, a material determines the physical property of the surface, e.g., reflectivity, roughness, metallic, transmission, transparency, etc. Both textures & materials affect the visual (realistic) appearance of the 3D model. Hence, all mentioned parameters are subject to randomization in order to generalize the DL model's performance in detecting a larger variety of the same asset using different sensors/cameras and in different environmental conditions, especially when it comes to high-reflection surfaces.
2. **Light**: It consists of randomizing the physically-based light control parameters such as the light color, temperature, intensity, and directions. Additionally, the light and material components are interdependent and they can impact one another. The properties of the material can cause a surface to reflect or absorb certain wavelengths of light, while the lighting conditions can alter the asset's surface appearance of the material, resulting in various realistic and complex combinations.
3. **Visibility**: It is a randomization following a symmetric Bernoulli distribution for simply visualizing the asset or obscuring it in the 3D space of the virtual scene.

4. **Transform**: An asset's transform settings consist of position, rotation, and scale properties. It manipulates an object's position along the x, y, and z-axis, 3D rotates it around its pivot point, and changes its size in all dimensions respectively. Hence, randomizing the position and rotation properties leads to placing the object in a defined area at different orientations. However, it is essential to consider the physical properties when spawning or replacing assets to avoid asset collisions or floating assets.

In Figure 3, we combined all four randomization components in a single scene, and from a single camera viewpoint we rendered distinct and various images.

Appendix B SORDI.ai Dataset Details

SORDI.ai dataset is split into multiple folders to distinguish between the following major differences: Scene, DR level, annotated assets, data capture type, rendering algorithm and data cleaning. In Table B1, we present 35 new datasets in addition to our previous dataset in [13].

Table B1 SORDI.ai datasets brief

Dataset	Environment	Asset Number	Capture	DR Level	Resolution	Dataset Size
Industrial Scene_1	Industrial, Warehouse	36	FRC	DR-1	720p	9,325
Industrial Scene_1	Industrial, Warehouse	36	FRC	DR-2	720p	36,943
Industrial Room 2	Industrial	28	FRC	DR-2	720p	12,941
Industrial Scene_3	Storage Room	36	FRC	DR-1	720p	6,360
Industrial Scene_3	Storage Room	36	FRC	DR-2	720p	35,033
Industrial Scene_4	Industrial	39	FRC	DR-1	720p	8,742
Industrial Scene_4	Industrial	39	FRC	DR-2	720p	39,759
Industrial Scene_5	Logistic	35	FRC	DR-1	720p	9,178
Industrial Scene_5	Logistic	35	FRC	DR-2	720p	43,866
Industrial Scene_5	Logistic	35	FRC	DR-3	720p	46,064
Industrial Scene_6	Industrial	32	FRC	DR-1	720p	9,127
Industrial Scene_6	Industrial	32	FRC	DR-2	720p	20,632
Industrial Scene_6	Industrial	32	FRC	DR-3	720p	45,866
Industrial Scene_7	Industrial, Office	49	FRC	DR-1	720p	9,538
Industrial Scene_7	Industrial, Office	49	FRC	DR-2	720p	24,468
Industrial Scene_7	Industrial, Office	49	FRC	DR-3	720p	4,488
Industrial Scene_8	Industrial, Storage Room	38	FRC	DR-1	720p	8,889
Industrial Scene_8	Industrial, Storage Room	38	FRC	DR-2	720p	20,711
Industrial Scene_9	Industrial, Office	48	FRC	DR-1	720p	8,785
Industrial Scene_9	Industrial, Office	47	FRC	DR-2	720p	43,554
Industrial Scene_10	Logistic	33	FRC	DR-1	720p	26,921
Industrial Scene_10	Logistic	33	FRC	DR-3	720p	26,089
Industrial Rooms (Default)	Industrial, Warehouse	11	FRC	No-DR	720p	17,301
Industrial Rooms (Default)	Industrial, Warehouse	8	FRC	No-DR	360p	12,123
Regensburg	Industrial (Dense), Plant	16	CRC	No-DR	720p	43,964
Regensburg	Industrial (Dense), Plant	21	SqC	No-DR	720p	18,545
Regensburg	Industrial (Dense), Plant	16	CRC	No-DR	360p	58,289
Spartanburg	Industrial (Dense)	34	CRC	No-DR	720p	104,047
Office	Office	36	CRC	DR2	720p	23,029
Toolshop	Toolshop	41	FRC	DR2	720p	12,963
Single Asset	Random	82	SC	DR3	720p	172,148
Warehouse Boxes	Warehouse	5	CRC	DR1	720p	7,999
Hackathon_Train_1 (2022)	Industrial	6	FRC	DR3	720p	10,234
Hackathon_Train_2 (2022)	Industrial	6	CRC	DR1	720p	9,973
Hackathon_Eval (2022)	Industrial	6	SqC	No-DR	720p	3,999
[13]	Industrial	8	FRC	No-DR	720p	200,000
Total (SORDI.ai)		111				**1,191,893**

Appendix C List of Abbreviations

In Table C2, we list the most commonly used abbreviations in our paper.

Table C2 List of abbreviations

AI	Artificial Intelligence
AMR	Autonomous Mobile Robots
AP	Average Precision
CRC	Constrained Randomization Capture
CV	Computer Vision
DCC	Digital Content Creation
DL	Deep Learning
DR	Domain Randomization
DR-1	First layer of Domain Randomization
DR-2	Second layer of Domain Randomization
DR-3	Third layer of Domain Randomization
Eval	Evaluation Dataset
FRC	Full Randomization Capture
IR	Image Realism
KLT	Kleinladungsträger (Small Load Carrier)
MDL	Material Definition Language
NN	Neural Network
No-DR	No usage for any Domain Randomization
PBR	Physically Based Rendering
r	ratio
Regen.	Regensburg
SC	Static Capture
SDG	Synthetic Data Generation
SDK	Software Development Kit
sim	Simulation
SORDI	Synthetic Object Recognition Dataset for Industries
Spartan.	Spartanburg
SqC	Sequential Capture
Synth	Synthetic
USD	Universal Scene Description
VDA	German Association of the Automotive Industry

Appendix D SORDI.ai Samples

In the following images, we present rendered samples from different SORDI.ai dataset folders paired with their bounding box images.

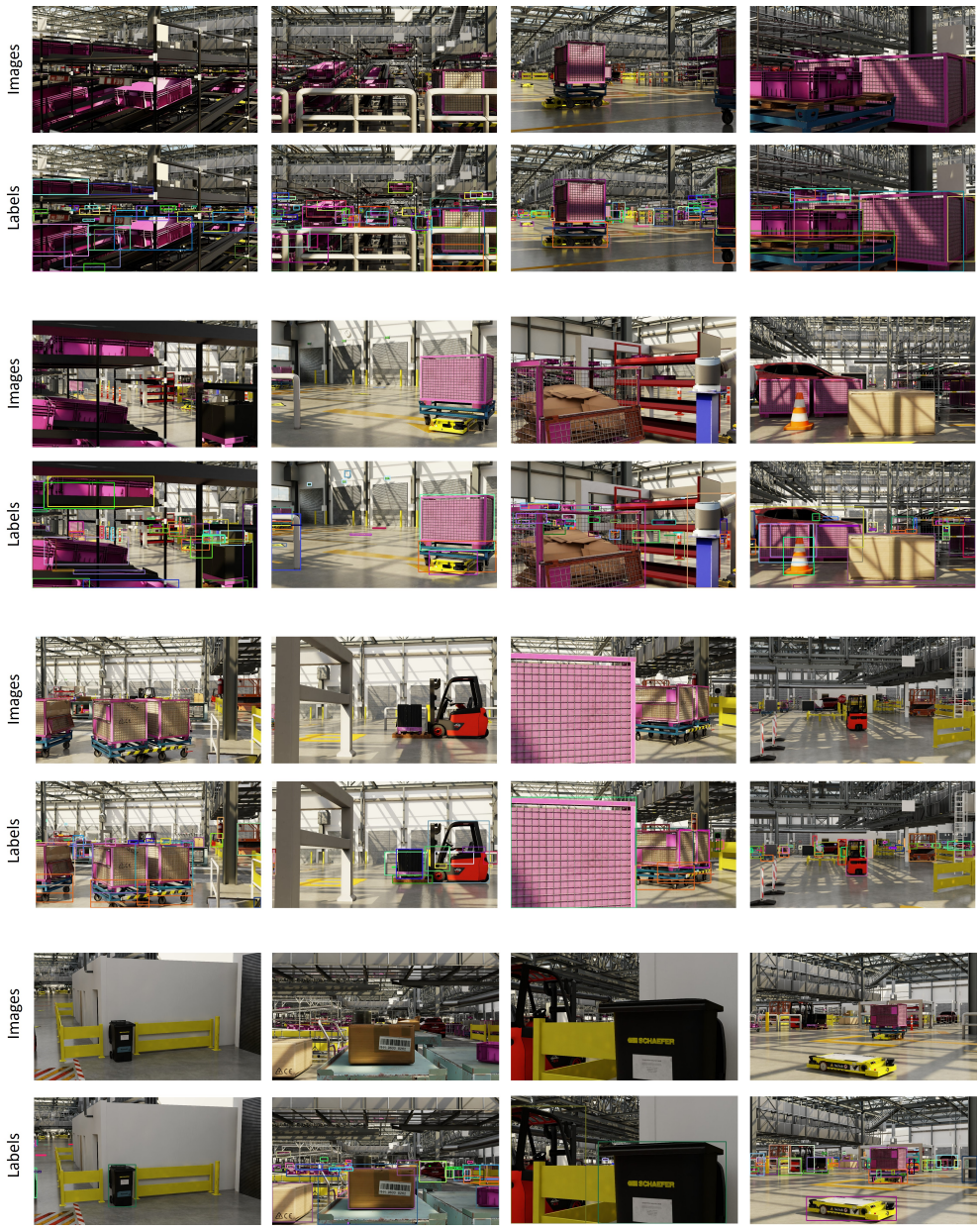


Fig. D2 Samples from Spartanburg dataset

Fig. D3 Samples from Regensburg (CRC) dataset

Fig. D4 Samples from Regensburg's (SqC) datasets

Fig. D5 Samples from the office and toolshop datasets

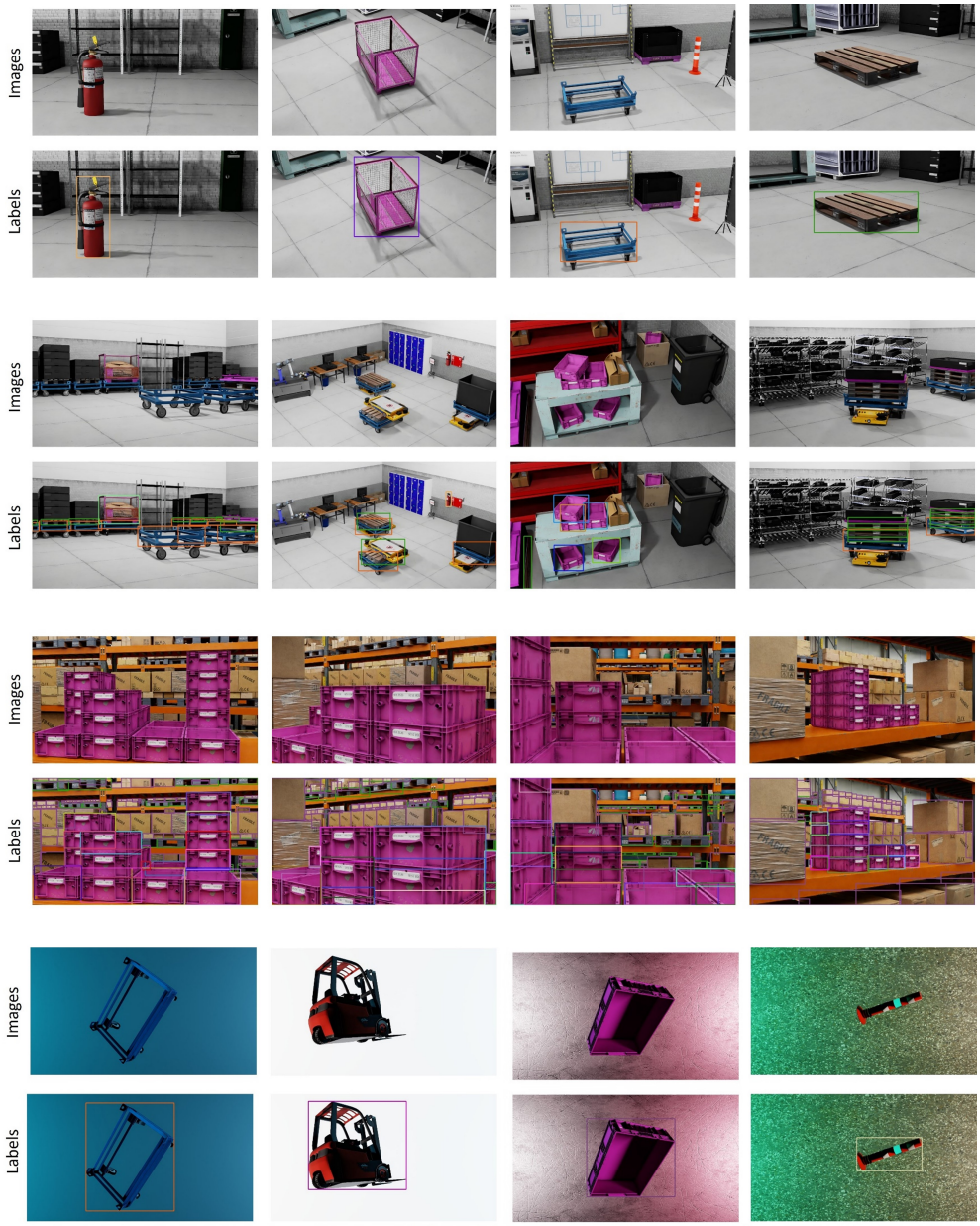


Fig. D6 Samples from the [13], Industrial Room Default, Warehouse, and single asset datasets

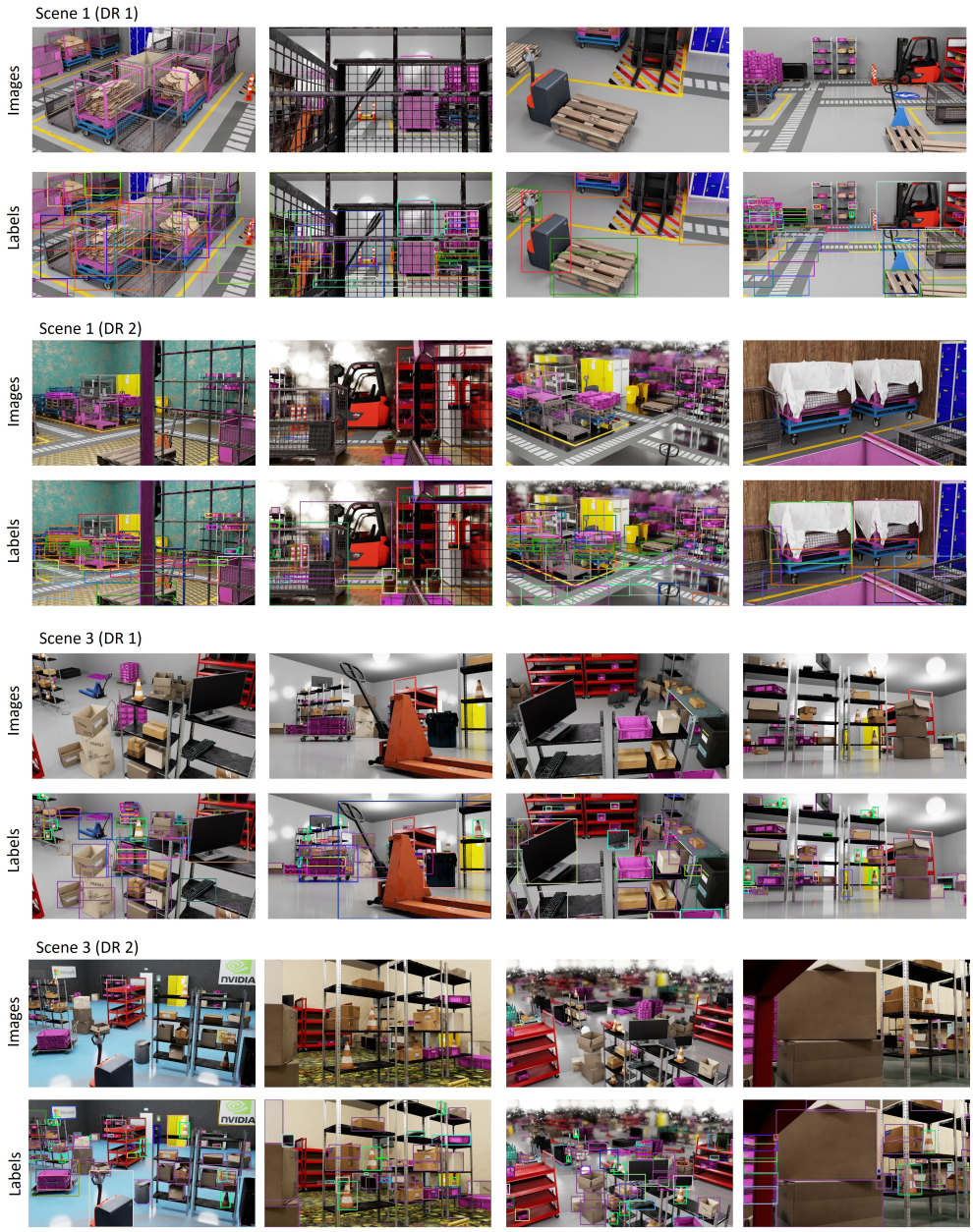


Fig. D7 Samples from Industrial Scenes 1 and 3 in DR1 and DR2

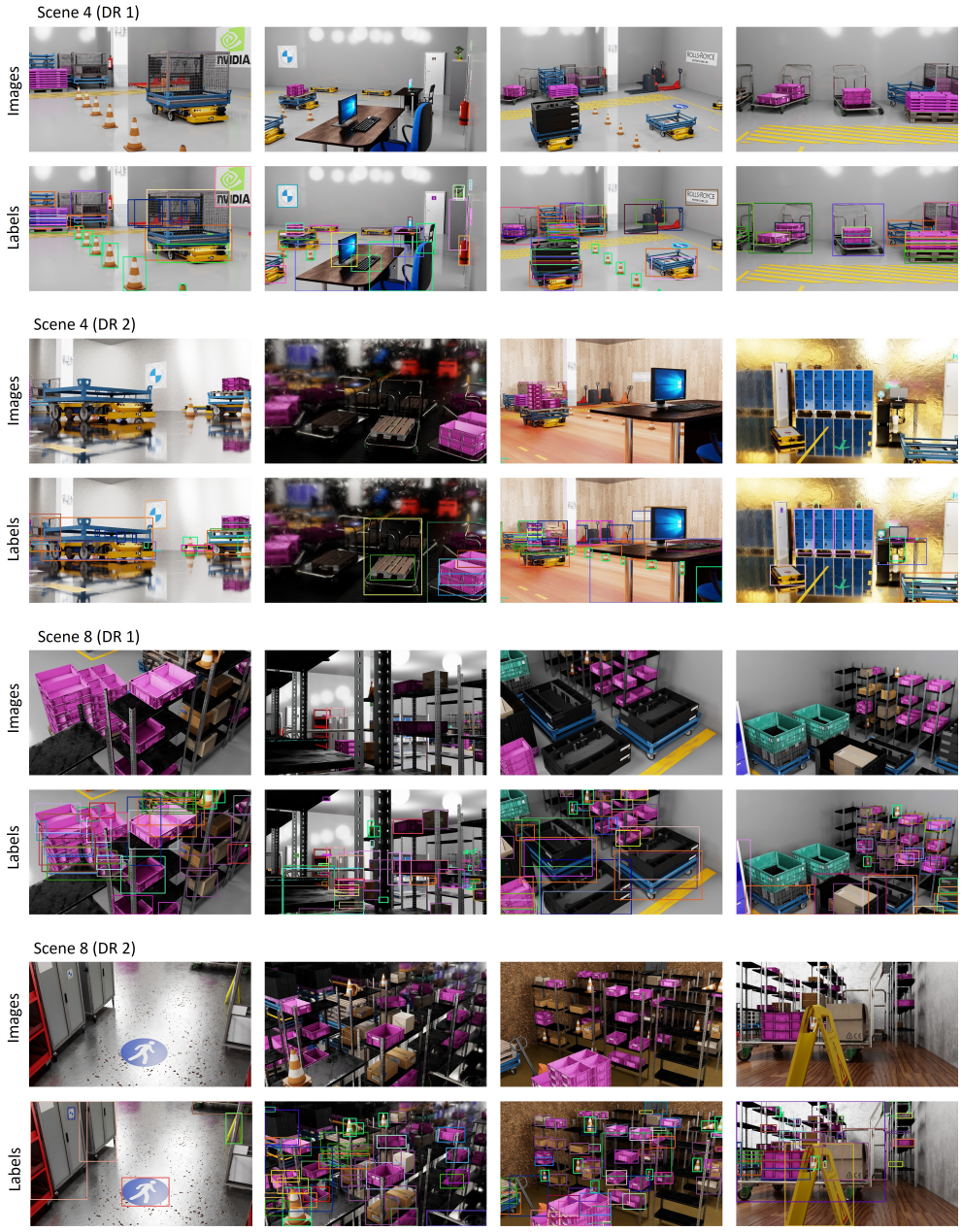


Fig. D8 Samples from Industrial Scenes 4 and 8 in DR1 and DR2

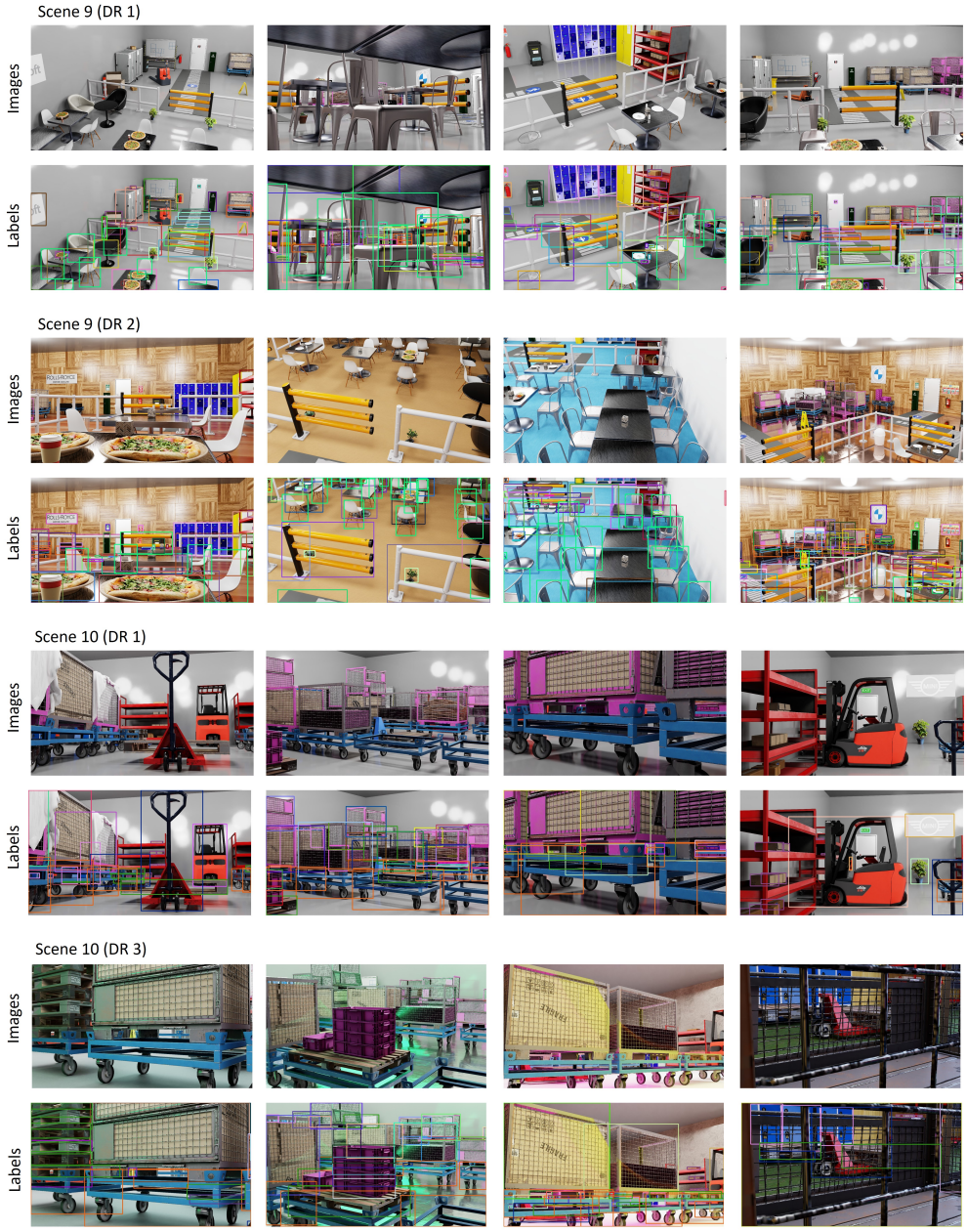


Fig. D9 Samples from Industrial Scenes 9 and 10 in DR1 and DR2, and DR1 and DR3 resp.

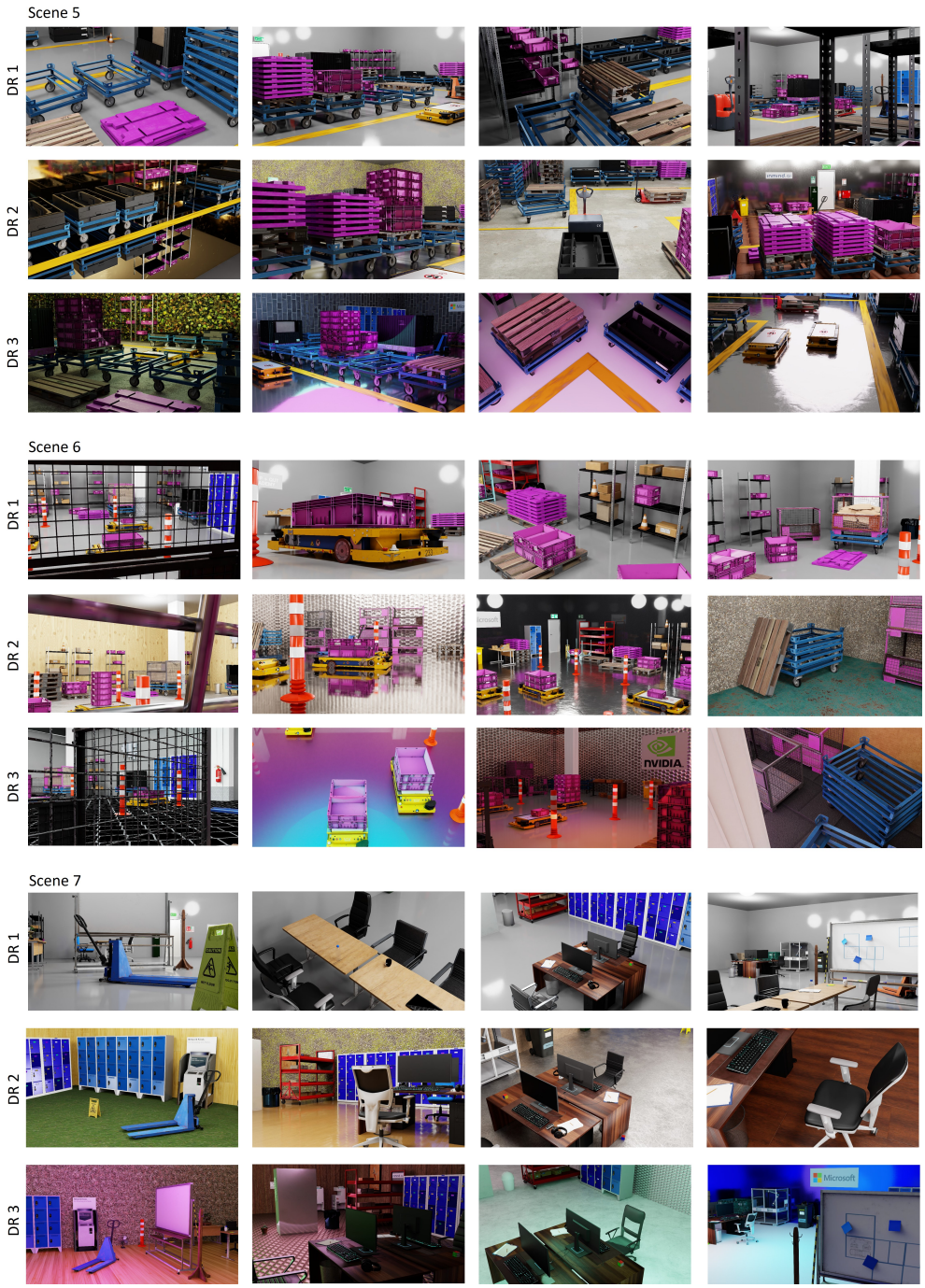


Fig. D10 DR1, DR2, and DR3 levels from Industrial Scenes 5, 6, and 7