Don Joseph Rajindu Goonewardena

Leveraging RGB-D Data and SAM Segmentation for Object Segmentation in Industrial Bin Picking

Master's thesis

School of Technology and Innovations
Sustainable and Autonomous
Systems
Master of Science in Technology

Vaasa 2025

UNIVERSITY OF VAASA
School of Technology and Innovations
Author: Don Joseph Rajindu Goonewardena
Title of the thesis: Leveraging RGB-D Data and SAM Segmentation for Object Segmentation in Industrial Bin Picking.
Degree: Master of Sciences
Degree Programme: Computing Science
Supervisor: Jani Boutellier
Year: 2025 **Pages:** 79

ABSTRACT:

Robotic bin picking requires a robotic arm to identify and extract objects from a container for placement on a production line. While this task can be broken down to two main aspects, the object selection and path planning and picking of the object. This thesis will focus on the latter. The task would include the selection of an object that can reliably be identified and picked in the least amount of time. The challenges faced here will include cluttered bins, occlusions and an ever-changing selection of objects that need to be identified by the segmentation model with the least amount of training time to identify novel objects. Traditional segmentation models struggle in these environments especially when they have not been trained on the specific objects being picked. This thesis investigates how the Segment Anything Model (SAM), a modern zero-shot segmentation model can be adapted in the simplest and effective way possible to industrial bin picking. The goal is to refine SAM's mask selection process so that a suitable candidate object can be selected from a bin with the least amount of training or tweaking of the segmentation pipeline. A modular scoring pipeline was designed and integrated into SAM. The experiments to test out the pipeline were conducted on both a synthetic dataset and real RGB-D images captured by a depth camera. Initial tests showed that a simple least depth-based mask performed well but suffered in the selection of masks that covered the whole object and in selecting objects that were away from the border of the image. By including all the aspects of the segmentation pipeline the quality of the object picked increased dramatically by avoiding issues such as partial masks, selection of objects too close to the bin walls and selection of the bin wall as an object. The findings highlight both the strengths and weaknesses of SAM in industrial context. More importantly, it shows that a carefully designed scoring system can negate those weaknesses and be used as an adaptable and effective segmentation solution for robotic bin picking offering a practical solution without retraining or complex hardware.

KEYWORDS: RGB-D segmentation; Segment Anything Model (SAM); bin picking; depth sensing; industrial robotics; mask scoring.

Contents

1 Introduction 9

2 Background and Literature Review 12

2.1 Evolution of Automation in Manufacturing 12

2.2 Role of Robotics in Modern Industry 12

2.3 Bin Picking in Industrial Contexts 13

2.3.1 Definition and Scope of Bin Picking 13

2.4 Depth Data in Machine Vision 14

2.4.1 Types of Depth Sensors 14

2.4.2 Advantages of Depth Information in Object Recognition 15

2.5 Segmentation in Industrial Robotics 15

2.5.1 Principles of Semantic Segmentation 15

2.5.2 Segmentation in Cluttered Scenes 17

2.6 Segment Anything Model (SAM) 18

2.6.1 Overview of SAM Architecture 18

2.6.2 Integration of SAM with Depth Information 19

2.6.3 Comparative Analysis of Segmentation Approaches 20

2.6.4 Depth Data Utilization in Bin Picking 20

2.6.5 Fusion of RGB and Depth Data 21

2.6.6 Object Detection and Pose Estimation 22

2.6.7 Handling Occlusions in Cluttered Bins 23

2.6.8 Integration with Depth-based Bin Picking 24

2.6.9 Workflow for Object Segmentation and Grasp Planning 26

2.7 Classic computer vision for segmentation and proposal 27

2.7.1 Bottom-Up Segmentation Approaches 27

2.7.2 Region Proposal Generation: Selective Search 27

2.7.3 Sliding Window Detection 28

2.7.4 Voxel-Space Processing 28

2.8 Supervised Learning Methods 29

2.8.1 Overview 29
2.8.2 Model Training and Validation 31
2.9 Reinforcement Learning 31
3 Methodology 33
3.1 System Setup 33
3.2 Image acquisition and loading of images 35
3.2.1 RealSense camera image acquisition and dataset creation. 35
3.2.2 Siléane Loader 36
3.3 Segmentation Pipeline 37
3.3.1 Evaluation Protocol 42
4 Experiments 44
4.1 Siléane Dataset Tests 44
4.1.1 Results with standalone SAM. 44
4.1.2 Results with SAM plus the custom scoring function and top most mask. 45
4.2 Ablation Study 49
4.2.1 Results of the performance considering only the depth results. 49
4.2.2 Results of the performance considering the depth and depth variance results 51
4.2.3 Results of the performance considering the depth, depth variance and area results. 52
4.2.4 Results of the performance considering the depth, depth variance, area and shape results. 54
4.2.5 Results of the performance considering the depth, depth variance, area, shape and border closeness results. 54
4.2.6 Final scoring system and an Intuitive look in terms of bin picking. 57
4.3 Real-World Demo 60
5 Discussion 63
5.1 Summary of findings. 63
5.2 Interpretation of the Results 64

6 Conclusion 67

7 Future work 68

Depth completion and sensor robustness 68

Adaptive weighing of scoring terms 68

Object separation factor 68

Efficiency and real time deployment 68

References 70

Figures

Figure 1 Typical industrial bin picking scenario (Imagen 2., 2025) 9
Figure 2. Real dataset capture setup including a depth and RGB intel RealSense D435i camera and a laptop with an intel i5-7300HQ at 2.5 GHz, 16 GB of RAM and a Nvidia 1050Ti GPU. 34
Figure 3 RealSense pipeline setup for RGB and depth streams which are combined using the object rs.align . 35
Figure 4 Loading of the Siléane dataset with the created dictionary. 36
Figure 5 Depth variance normalization. 38
Figure 6 Penalty score against the deviation from an ideal range between 5 and 2%. 39
Figure 7 Shape/compactness score using the Polsby-Popper compactness score. 40
Figure 8 Border penalty calculation. 41
Figure 9 Structure of the scoring equation. 42
Figure 10 Top mask vs the best mask where the green brick would be the ground truth mask, and the bright red highlight would be the generated mask that was selected after SAMs mask generation. 47
Figure 11 Filtering of the generated masks from the use of SAMs confidence and stability scores. 48
Figure 12 Ablation configurations for the Siléane bricks dataset. 49
Figure 13 Depth only best mask vs Topmost mask. 50
Figure 14 Depth + variance best masks selection ranked for one image. 52
Figure 15 Depth + variance + area best masks selection ranked for one image. 53
Figure 16 Segmentation without (left) and with (right) border penalty, giving the mask which is away from the border for easier grasping from a robotic arm. 55
Figure 17 Final weights for the scoring pipeline 57
Figure 18 Brick 32/36 best vs topmost mask selection comparison. 57
Figure 19 Brick 40 and 42 best mask segmentation. 58

Figure 20 Brick 43 and 52 best vs topmost mask showing that the best mask consistently tries to strike a balance between lowest depth and masks away from the border while the top most mask tend to select any mask with the least depth. 59

Figure 21 Best and topmost mask overlays, the object masks in blue would be the selected segmentation mask by the scoring system and masks in green are the top masks. 60

Figure 22 Improper segmentation for the top mask (in green) 61

Figure 23 Missing depth value filling solution using the Navier stokes algorithm provided by the Open CV library. 62

Figure 24 Gray scale depth images of image 18 and 19 of the real-world dataset collected using the RealSense RGB-D camera, where darker colors mean closer it is to the camera except the pure black areas meaning 0 mm depth or no valid depth reading. 62

Tables

Table 1 SAM segmentation across the Siléane bricks dataset 44

Table 2 Segmentation across image brick_0_008 using SAM for all objects in the image. 45

Table 3 Initial scoring pipeline results with SamPredictor across 30 images of the Siléane bricksdataset. 46

Table 4 Subset of the topmost mask's performances for 6 separate images in the dataset. 46

Table 5 Performance of the masks after the use of SAMs confidence scores and NMS on the 30 Siléane bricks dataset images. 48

Table 6 Best mask performance with depth alone. 49

Table 7 Best mask performance with depth and depth variance. 51

Table 8 Best mask performance with depth, variance and area. 52

Table 9 Best mask performance with depth, variance, area and shape. 54

Table 10 Performance with the full best mask pipeline 54

Table 11 Segmentation performance across all ablation settings in the best mask scoring pipeline and topmost masks 56

1 Introduction

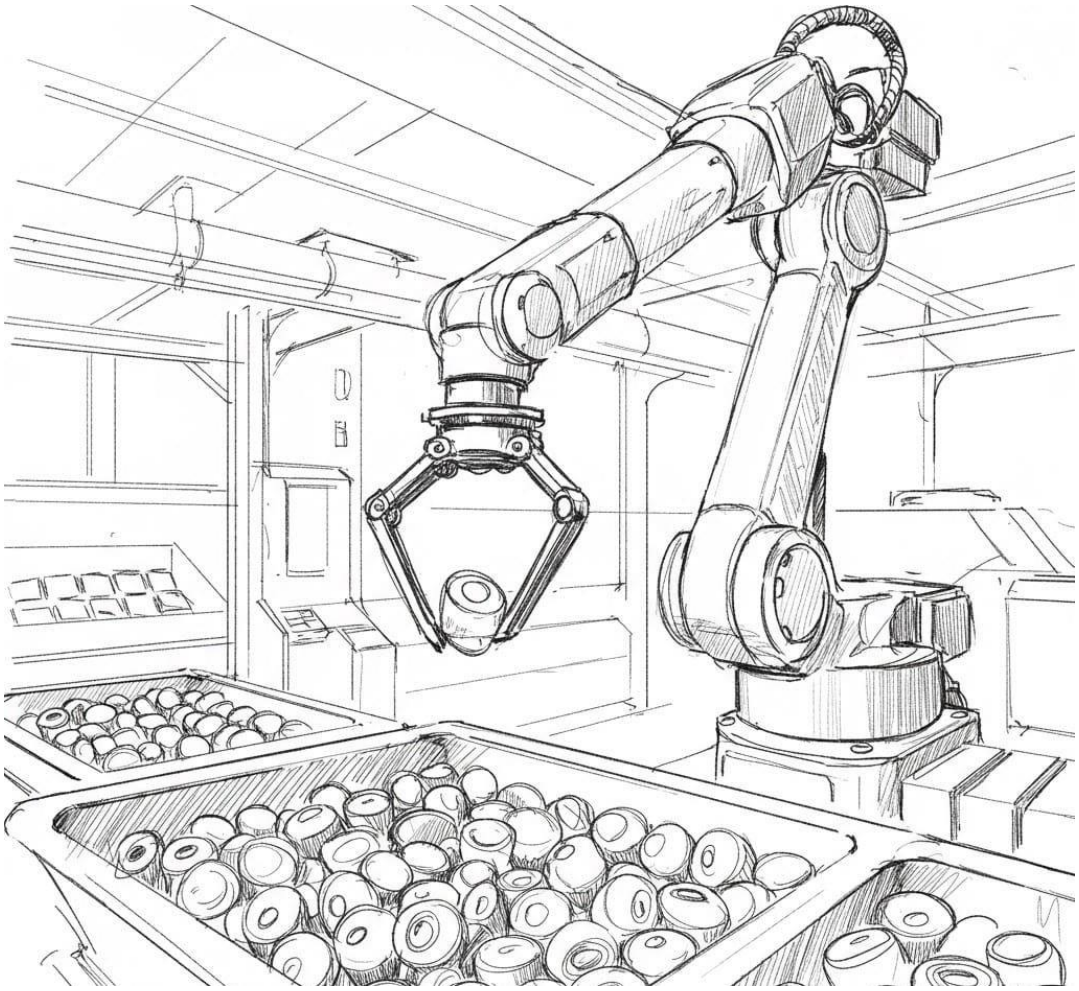

Figure 1 Typical industrial bin picking scenario (Imagen 2., 2025)

In industrial bin picking one rarely gets an organized bin to pick from. The parts overlap, reflect light and can be tricky for texture-based segmentation methods alone to identify the most suitable object to pick and it also can have changes in the dimensions and shape of the objects that needs to be picked from time to time with minimum time to retrain on the new objects put on to the production line. The Segment Anything Model (SAM) is almost the ideal model to be used for the latter challenge as it produces masks on novel objects without the need for retraining using prompts such as points, boxes or

masks (Kirillov et al., 2023) or even without any prompts. Using these prompts combining the image encoder, prompt encoder and mask decoder SAM produces accurate masks with low latency even when the prompts given are not ideal (Zhao et al., 2023). Even with the advantages it has as a zero-shot segmentation model, it still can struggle with accurate segmentation of occluded objects, metallic or glossy parts and textureless objects. In selection of the best option to pick from the bin various aspects of the object that need to be picked need to be considered. While depth data can be brought in to virtually give SAM a different perspective and almost creating a 3D perspective of the space to uncover some occluded objects (Yu et al., 2024). It might not be the most reliable as depth cannot determine objects, as smaller masks can be generated for objects in a bin but only represent a portion of the surface. Depth in this instance cannot mitigate this issue. While the fact that boundaries in depth often align with physical object edges and helps in distinguishing objects that look similar in the RGB domain is true, depth alone cannot determine if the mask generated partially due to poor lighting or object surface fully segments the object. While depth is an important part of the segmentation in bin picking shown by numerous studies there is a lack of studies where the evaluation framework combines geometry of the object, depth with zero shot segmentation without the need for retraining but by only calibrating the geometric and depth thresholds. This combination works well in not only bin picking datasets but also now it is much more accessible to get prebuilt sensors that combine the depth and RGB data without the need for complex calibration methods and multiple cameras and hardware. The contribution of this thesis would be based on a scoring-based mask selection framework that integrates multiple thresholds into a unified evaluation pipeline. It includes a depth aware scoring, where higher depth values and depth variance are penalized and preferring masks within operational ranges, area normalization where the mask size penalties have been tied to the ground truth masks obtained, to avoid being too large or small of a mask. Shape compactness calculated through the Polsby-Popper compactness score was used to ensure irregular masks were also penalized and border penalties to avoid objects that are too close to the image border in being picked up in the initial runs

of the bin picking. This study also includes detailed ablation experiments, CSV logging and overlay visualization for the transparent visualization of the methods developed.

The scope of the thesis consists of designing and implementing a multi-term scoring function for evaluations of masks in RGB-D datasets obtained from zero shot segmentation pipelines such as SAM in further developing on the accuracy of the segmentation for bin picking using depth data and keeping the pipeline and hardware as accessible as possible, conducting of ablation studies to quantify the impact of the scoring system for each component in the scoring system, benchmarking against the ground truth segmentation on the Siléane bricks dataset with metrics such as IoU, precision, recall and F1. Qualitative analysis of real-world data obtained through our own commercially available and accessible hardware was also included. Finally visual overlays and CSV summaries of studies conducted have been attached. While the scope here is limited to static dataset evaluation it lays the foundation for development of robust models to be used in real time segmentation in robotic bin picking pipelines. To summarize this thesis proposes an adjustable scoring-based selection process to improve the mask selection process in industrial bin picking for SAM based segmentation without the need for complex hardware setups and retraining.

To fully understand the challenges faced and addressed in this thesis it is important to grasp the whole process of industrial bin picking which includes perception, which is the area this thesis is based on, planning and manipulation which is based on the robotic systems used to carry on the picking process. Therefore, the following literature review will give an overview of industrial bin picking as a whole and then narrow down to the perception and segmentation methods most relevant to this study.

2 Background and Literature Review

2.1 Evolution of Automation in Manufacturing

Industrial automation has shifted from rigid, specialized setups to flexible, software-driven systems. Tools like ROS-Industrial make it easier to connect hardware from different vendors(Janis Arents et al., 2018), allowing robot models and controllers to be swapped with minimal effort. This flexibility, combined with GPU-powered deep learning, has enabled real-time perception to be part of the control loop (Janis Arents et al., 2018; Ribeiro et al., 2021).

Today, multi-modal sensing is common, and RGB-D pipelines with learned segmentation are standard for tasks like random bin-picking (Franaszek et al., 2024). But more sensors don't always help—larger setups can be harder to deploy (Xie et al., 2023a). New sensing methods, like vision-based tactile devices, can support RGB-D when visual cues aren't enough to judge grasp stability. At the same time, manipulation strategies are advancing. Robots no longer rely only on open-loop picks; reinforcement learning and hybrid methods can plan actions like pre-grasp pushes to improve success (Zeng et al., 2018). Overall, perception and control strategies have evolved together, making SAM's flexible, depth-aware segmentation a strong fit for these systems.

2.2 Role of Robotics in Modern Industry

In modern industrial robotic manipulation, robots need to overcome many variables such as unknown object types, deformable materials and unpredictable poses, therefore systems will fare much better if they infer from RGB-D data collected rather than relying on fixed CAD models. This is important in production environments where the product changes rapidly and the models used can be outdated quickly (Franaszek et al., 2024;

Mahler et al., 2017; Zeng et al., 2018). In these instances, SAM's zero-shot instance segmentation by prompting is particularly attractive.

Location of an object in a group of multiple objects depends on RGB-D data and geometric information, including in methods such as PPF where both RGB data, geometric and depth information were used to gain better results(Liu et al., 2021; Zhuang et al., 2021). Mixing of both RGB and depth data allows these industrial systems to tackle common issues such as shiny surfaces and surfaces with low texture. These setups not only help in bin picking but in many industrial processes involving computer vision (Chang et al., 2023; Franaszek et al., 2024; Nishi et al., 2023; Zou et al., 2020).

2.3 Bin Picking in Industrial Contexts

2.3.1 Definition and Scope of Bin Picking

Bin picking is the process of using automated robotics to recognize and localize objects even under occlusion, and to detect their pose despite variations in material and surface characteristics. Deep learning-based approaches have shown strong performance in these scenarios (Le & Lin, 2019). The issues are not only about visually detecting objects, but it is also about the surrounding environment and how much of space is available around the object (Schwarz et al., 2018). Therefore, the gripper designs, reachability and other aspects of the overall selection pipeline need to be thought of from the beginning. There are usually two main methods in object selection in bin picking, one would be 3D model matching and the other is segmentation and object selection through RGB-D data (Grard, n.d ;Cordeiro et al., 2023).Most pipelines start by capturing RGB-D data, mapping pixels to real-world coordinates, segmenting objects, and planning grasps that avoid collisions and respect robot limits (Le & Lin, 2019) ; (Iriondo et al., 2021). But industrially the goal would be to minimize the time between the start of perception to the picking as much as possible.

2.4 Depth Data in Machine Vision

2.4.1 Types of Depth Sensors

There are various types of depth sensors. Time-of-Flight (ToF) sensors use pulsed signals or continuous wave modulation and measure the time taken for the waves to travel back and forth to determine distance(Zollhöfer, 2019); (Shao et al., 2023).Pulsed ToF sensors requires mechanical scanning such as moving mirrors to collect point by point data, while matrix-based ToF captures full depth image data for all points in one go, which is fast and compact, ideal for industrial bin picking. Continuous Wave ToF measures phase shifts in the sent modulated light and can perform reliably at short to medium ranges, even under changing factory lighting (O'sullivan & Le Dortz, 2021; Zollhöfer, 2019). Structured light sensors project known patterns and calculated depth based on how those patterns deform when hitting surfaces (Shaikh & Chai, 2021). Devices like Intel's RealSense D-series can achieve millimeter-level accuracy at close range (Intel Corporation, 2019). However, structured light can struggle in multi-robot environments due to pattern interference, and its performance drops at longer distances because of weaker illumination(Zollhöfer, 2019). LiDAR excels at scanning large areas but is less suited for tight bin-picking spaces. However, newer solid-state versions are starting to change this by offering compact and cost-effective solutions for industrial robotics (Shaikh & Chai, 2021). Passive stereo systems, such as ZED, work well on textured surfaces but can be thrown off by smooth, reflective materials. Combining stereo with active sensing or fusion techniques helps overcome these limitations (Keselman et al., 2017). Each sensor type has its own noise characteristics. Structured light may lose detail on fine features and suffer from crosstalk, while ToF can be affected by multi-path reflections and ambient light (Zollhöfer, 2019). Encouragingly, newer sensors offer better factory calibration and more user-friendly SDKs, which help reduce setup and integration time. Even budget-friendly Continuous Wave ToF sensors can be effective in close-range, controlled environments—though they're less suitable for long-range tasks like forestry (McGlade et al., 2022). The

pipelines developed to be used with such sensors therefore need to take into consideration the nuances in each sensor type.

2.4.2 Advantages of Depth Information in Object Recognition

Depth's main advantage is that it captures geometry directly. Instead of depending on color or texture, it separates objects by their physical boundaries, which is vital when most of the detected items are similar(Schwarz et al., 2018). It is also less prone to error due to lighting changes (Le & Lin, 2019). Using depth reduces the risk of masks drifting onto other separated surfaces (Cordeiro et al., 2023).

Handling occlusion is another area where depth data is useful. Even when only part of an object is occluded, depth information can help connect the dots by piecing them together into a recognizable shape by connecting the detected segmentations into a probable depth related surface (Schwarz et al., 2018). When combined with SAM, these geometric guidelines can guide masks to follow actual edges more closely. While more flat or thin objects may not show much depth variation, but yet depth can be used to capture small tilts and bend on the object and to plan more reliable grasp approaches (Le & Lin, 2019). Together, these properties make the segmentation pipeline less prone to error, less likely to be fooled by glossy finishes or uniform textures, and more likely to produce masks that align with the detected object.

2.5 Segmentation in Industrial Robotics

2.5.1 Principles of Semantic Segmentation

Semantic segmentation gives every pixel in an image a label to identify as to what segment the specific pixel belongs to. The issue is how the algorithm is going to balance the sharp boundaries detected with segments that make sense in the real world. Traditional encoder–decoder CNNs compress and then rebuild features, but they often struggle

with thin shapes or tiny parts (Shvets et al., 2024). Vision transformers improved this by using attention to relationships between parts of an object which might not seem connected but are connected, which is especially useful in cluttered industrial scenes where one part of an object can appear in one of the images and be cut off by another object and the rest of the object might appear at another end.

SAM works by taking in prompts instead of fixed class labels. A strong encoder first creates general features from the image, while a prompt encoder turns user inputs (like points or boxes) into signals. A decoder then combines both the general features and the user prompt signals to produce the final mask (Kirillov et al., 2023). Because it relies on prompts, SAM can handle new object categories it hasn't seen before (Kirillov et al., 2023). Training on large, diverse datasets also helps it avoid overfitting, which is a common problem when models are trained only on a small set of industrial data (Kirillov et al., 2023)

In multi-task setups, for example in joint semantic and instance segmentation, loss needs to be balanced to ensure that the segmentation does not bias towards either the semantic side or the instance segmentation. Proper weighting will result in both segmentation methods balancing out meaningless outputs (Kirillov et al., 2023). Fusion of the data used also plays a major role in increasing the accuracy and practicality of segmentation. RGB and depth when combined at early or mid-layers result in counteracting weaknesses that might be unique to each method. Thus, producing masks that involves the appearance and geometry of the objects (Cordeiro et al., 2023; Zhuang et al., 2021). By creating artificial data which is also augmented in unlikely ways makes the model more robust to unexpected scenarios in the real world by focusing more on the essential structures of objects during the training phase (Eghbal-zadeh et al., 2024; Hernández-García & König, 2020; Rebuffi et al., 2021). Post-processing, although not as important as the model design, still matters, especially for edge cleanup, where there can be jagged or noisy boundaries and small objects where it can be merged into the background if not looked into.

2.5.2 Segmentation in Cluttered Scenes

Clutter makes segmentation harder by adding occlusions, unclear edges, sensor noise, and objects that look alike. Clustering based on features of the pixels tends to fail in cluttered environments, especially in environments with identical objects. When objects are occluded the objects that are overlapping might be segmented in a way that they are merged due to the similar features of their pixels. Pipelines that are not heavily based on features, such as geometric cues and boundary aware models usually do better in these scenarios(Xu et al., 2022). While merging of EGB and depth data is helpful here, it is crucial to align the two data streams accurately to avoid the borders shifting on to other objects (Liu et al., 2021).

Another way to reduce the occlusions or bring out more of the hidden surfaces in an occluded environment is to slightly move the pile so that a second scan of the scene has more information of the occluded areas resulting in better segmentation (Zeng et al., 2018). Starting with segmentation before checking gripper constraints also helps and is generally recommended, since it avoids false positives from background geometry (Cordeiro et al., 2023; Zeng et al., 2018). Another obvious strategy is to have multiple cameras from different angles so as to get more information of hidden areas of the scene (Zeng et al., 2018).

In bin picking scenarios the way training data is labeled matters too. Marking graspable regions instead of whole objects directs the model towards useful pixels, reducing errors in cluttered scenes (Nishi et al., 2023). Speed is one of the most important aspects in deciding a model to be used in an industrial bin picking process and segmentation is often the slowest step and can eat into the cycle time (Le & Lin, 2019). Efficient prompts, parallel GPU use, and depth-based filtering help keep processing fast, so that the robots spend more time in the manipulation process than computing

2.6 Segment Anything Model (SAM)

2.6.1 Overview of SAM Architecture

SAM's design is clean and practical. A transformer-based encoder first creates a detailed feature map of the image. A prompt encoder then turns inputs like points, boxes, or masks into guidance, and a lightweight decoder combines both the prompt guidance and the feature map to generate the final mask in a matter of few milliseconds once the features are ready (Kirillov et al., 2023; Shvets et al., 2024). SAM is able to reduce computationally heavy tasks such as the feature creation by reusing the features that were created for many prompts, which is especially helpful when a bin is full of similar objects. SAM also gives a confidence score for each mask, similar to IoU, which systems down the ladder can use to filter results (Kirillov et al., 2023).

The training regime uses a mix of dice and focal losses, making sure not only that the predicted masks overlap well but also considering the hard to categorize pixels and includes simulated multi-round prompting, improving the model's responsiveness to iterative changes between perception cycles where the scene usually changes between each cycle (Kirillov et al., 2023). Its SA-1B pretraining dataset is massive with millions of images and over a billion masks, supporting cross-domain generalization (Kirillov et al., 2023). While SAM trades specialization for variety, it can be paired with domain specific modules to specialize in certain tasks. Focused more on low latency there is also a faster version of SAM called FastSAM, which has a twostep process, detect all objects and use the prompts to filter the ones that are needed (Zhao et al., 2023). In practice, SAM performs best when combined well with other models. For example, where SAM produces the general masks and the other models refine them further.

2.6.2 Integration of SAM with Depth Information

Depth can be added to a SAM pipeline in several ways. At a basic level, depth-based prompts can be used to nudge SAM towards areas that correspond to object surfaces and not just the textures in the RGB image. Examples of these depth-based points can be some sparse points picked from a 3D point cluster in a depth map or dense masks, regions created by thresholding depth discontinuities (Danielczuk et al., 2019). An in-depth approach would be to fuse RGB and depth inside the encoder. Either early fusion, where RGB and depth are combined and processed, where the model can learn very fine alignment between colour edges and depth edges thereby reinforcing the boundary or late fusion, where each is encoded separately and then combined. Where each type of data would have their own encoder that deals with noise specific to its modality, for example glare in RGB and gaps in the depth map (Cordeiro et al., 2023; Zhuang et al., 2021).

More advanced methods bring depth into the attention layers, so that similar objects can be decided by colour, texture and depth. This helps the model separate objects that look alike in color but are at different depths (Yi et al., 2019). Training with a mix of synthetic and real datasets such as WISDOM (Danielczuk et al., 2019) plus real-world scenes further improves stability by training the model with almost unlimited amounts of variation through the synthetic datasets and on tackling the sensor imperfections of real-world sensors (Danielczuk et al., 2019). Whether through prompts, fusion, or attention, the aim is the same. To make sure masks follow the true 3D shape of objects, not just their RGB appearance.

2.6.3 Comparative Analysis of Segmentation Approaches

Mask R-CNN (He et al., 2017) remains a classic model for instance segmentation baseline, particularly when large, labeled datasets are available for the target classes (Xu et al., 2020). But pixel level annotation can be costly, and the model is not robust enough to segment objects it was not trained on. SAM, by contrast, segments on prompts without training for every novel object, making it more adaptable to industrial bin picking, especially with smart automated prompting. RGB-only prompts can blend overlapping parts in clutter, so aligning SAM with depth data whether through prompt generation or fused encoders improves object segmentation (Shvets et al., 2024; Zhuang et al., 2021).

Semantic PPF (Point Pair Features) pipelines are geometric methods that use pairs of 3D points to predict object pose. When combines with semantic labels these pipelines can estimate the pose of objects that are partially visible or occluded (Drost et al., 2010; Liu et al., 2018; Zhao et al., 2024). Similar to Mask R-CNN, this pipeline excels when the object shape is known but struggles when new objects where the geometry is unknown are introduced. Meanwhile, sim-to-real strategies can customize models to specific parts using synthetic pretraining plus self-supervised adaptation on real data (J. Chen et al., 2025). In general, it can be stated that in simple clusters of similar objects class, specific models will perform better accuracy wise than general purpose models.

2.6.4 Depth Data Utilization in Bin Picking

The quality of the depth data obtained from the sensors initially starts from the placement of the sensor. The camera needs to be positioned in a way that avoids steep angles that would normally distort structured light or weakens Time of Flight signals (RF Wireless World (no individual author listed), n.d.; Zanuttigh et al., 2016). Having multiple viewing angles by having sensors placed on opposite corners reduces levels of occlusion but also requires more complex synchronization and registration of the cameras or

sensors used. To ensure accurate synchronization of hardware triggers and accurate timestamps are some of the solutions (Liu et al., 2021; McGlade et al., 2022).

Factories tend to be very dynamic environments and the possibility of the position of the cameras being moved if not secured properly is high. Therefore, mounts must be rigid and fixed to stable surfaces to avoid drift. If portable or handheld sensors are used, they will need frequent recalibration (McGlade et al., 2022; Zanuttigh et al., 2016). Calibration has two parts. Intrinsics which corrects for the lens distortions, focal length and represent the geometry of the external scene accurately without distortions. Extrinsic, which defines the position and the orientation of the sensors relative to each other, and the robots coordinate frame. (Khoshelham & Elberink, 2012; Z. Zhang, 2000). Proper calibration is important because even small extrinsic errors can shift targets by millimeters which can hinder the picking process.

Finally, the optimal working ranges of the type of sensor used need to be taken into account. For example, structured light works best at close distances while ToF reaches farther but is not the most accurate at short distances. But in bin picking shot range accuracy is key. Making in this case structured light the better choice (Schellhase, 2023; Siltala & Latokartano, 2023; Thanh Ly et al., 2022). In practice, solid calibration usually matters more than clever algorithms.

2.6.5 Fusion of RGB and Depth Data

Normally the RGB pixel values and the depth maps come from different sensors or different parts of the same sensor, which usually have different values for the resolution and field of view. Therefore, to have good fusion between the two, they need to be normalized and combined. Ensuring each depth value is matched up with the proper corresponding pixel values (Donné & Geiger, n.d.; Wang et al., n.d.). Early fusion feeds raw RGB–depth edges directly into the feature extractor like in a CNN or transformer encoder, while mid or late fusion allows each data stream to be cleaned of its specific noise. For example, smoothing depth with bilateral filtering to keep sharp edges(Li et al., 2024; Tomasi & Manduchi, 1998; Zhou, 2024)

While RGB and depth sensors housed in a single unit do have their own intrinsics, their relative position to each other is fixed and factory calibrated, which makes the total calibration process easier (G. Chen et al., 2018; Villena-Martínez et al., 2017; C. Zhang et al., 2019). If there are multiple sensors used which are housed separately, like a RGB sensor, lidar sensor and a stereo camera. Extrinsics need to be calibrated in these setups, which is complicated. After calibration they also need to be recalibrated over time due to gradual shifting of the devices. Once aligned, fused data improves segmentation by combining shading from the RGB data and geometry from the depth, giving models such as SAM better prompts and cleaner masks.

2.6.6 Object Detection and Pose Estimation

Depth maps can be turned into point clouds to be used in pose estimation. After the point cloud has been segmented it is aligned with the CAD model of the object by using algorithms such as ICP (iterative closest point). ICP tries to adjust the CAD model position and orientation until it matches as close as possible to the observed point. This results in a 6 Degrees of Freedom (DoF) pose. The segmentation here needs to be accurate as well. If not the ICP model might try to align the CAD to a part that's out of the object. Some objects can be symmetric or occluded, in such situations geometry alone cannot give the orientation of the object. In such situations adding semantic cues such as "top" through labels or context helps understand the alignment (Rusinkiewicz & Levoy, 2001; Wang et al., 2019; Xiang et al., 2018)

Fusion again makes this easier. While adding another set of data such as RGB data can increase the search time in the point cloud but color bounding boxes can narrow the search area in the point cloud. While depth denoising and surface normal estimation give better features for registration. If CAD models aren't available approximate geometry of the object or prior patterns the robot has learnt about grasping objects can be used to plan a stable grasp. For example, if the CAD model isn't available the covariance matrix of the segmented point cloud would be calculated which gives the main direction of the spread of points of the object. This generally aligns with the objects' natural shape

of the objects. Which can be used to infer a strategic approach of the gripper, especially parallel jaw grippers and suction grippers (Mahler et al., 2017; Miller & Allen, 2004; Schwarz et al., 2018). Training depth-only pose predictors on synthetic data also works well in controlled scenarios.

2.6.7 Handling Occlusions in Cluttered Bins

In industrial bin picking occlusions are unavoidable. To tackle this issue depth discontinuities are used to find object boundaries from sudden jumps in distances, even if only part of the RGB domain of that object is visible (Siltala & Latokartano, 2023; Wang et al., 2019; Yang et al., 2007). By creating a point cloud from the depth map and obtaining even fragments of the relevant object and grouping them using Euclidean distance and surface normal clusters that belong to the same object can be identified (Liu et al., 2021; Rabbani et al., 2006). Once clustered, these fragments can be matched to CAD models or reconstructed using the fragments using different points of view (Aldoma Buchaca et al., n.d.; Drost et al., 2010; Kazhdan et al., 2006).

SAM can rely on prompts to guide its segmentation and if these prompts are generated from depth data it will be able to better differentiate the foreground and background while generating masks that stay true to its geometric shape (Donne & Geiger, 2019; Kirillov et al., 2023; Li et al., 2024; Wang et al., 2019). In dealing with flat or symmetric objects semantic PPF can help by aligning visible fragments obtained by SAMs masks onto a full pose using geometry and semantic labels (Aldoma Buchaca et al., n.d.; Drost et al., 2010; Zhuang et al., 2021). The two physical changes that can be used to tackle occlusions is to use a multi view set up to obtain better viewing angles which need accurate extrinsics and as a last resort the pile can be shifted slightly to expose items further.

2.6.8 Integration with Depth-based Bin Picking

Hardware Components

When it comes to hardware to be used in a bin picking scenario, it needs to be selected to suit the required performance of the system. To obtain a view of objects here are some of the technologies that can be used. Modulated ToF (Time of Flight) such as a Kinect v2 works by emitting modulated infrared light and measuring the time it takes to return to the sensor. While it works in short to medium ranges and delivers decent frame rates which is suitable for bin picking it can struggle with reflective or transparent surfaces (Berger et al., 2013; Khoshelham & Elberink, 2012; Lachat et al., 2015). Passive stereo units like ZED can also be considered, where two RGB cameras are used to infer depth by comparing the differences in pixels. While they work well in outdoor scenarios due to not needing to project light. But it also struggles with shiny, reflective or texture less surfaces. For close range scenarios that need fine details captured structured light sensors such as the intel RealSense D series can be used. It emits a known pattern of infrared light and measures how it deforms. Here it can be caused with multiple structured light sensors due to the patterns interfering with each other and it is also sensitive to strong ambient light (Intel Corporation, 2019; Zollhöfer, 2019). When considering the manipulation hardware, a 6 DoF robot arm has been standard in the industry due to the flexibility it provides in reaching in cluttered bins and precision gripping. The end effector needs to be selected depending on the shape and texture of the object. While Vacuum grippers are great for flat, smooth surfaces, parallel-jaw grippers are good for rigid parts with graspable edges and custom grippers good for unusual shapes or delicate items.
The robot can be controlled by ROS drivers (Robot Operating System) which handles communication and control and MoveIt! which handles motion planning, collision handling and trajectory generation. The processing of the perception usually relies on the GPU and control is handled by a real time controller which handles tasks such as motion planning. The networking speed between the processes is critical as to stay synchronized

with low latency (Coleman & Robotics, n.d.; Create Realistic Robotics Simulations with ROS 2 MoveIt and NVIDIA Isaac Sim | NVIDIA Technical Blog, n.d.; Janis Arents et al., 2018). Specialized embedded boards like Jetson can run transformer models if optimized carefully to ensure the lowest latency.

Since both RGB and depth sensors are sensitive to lighting. It is also critical. Diffuse, synchronized light spreads evenly and avoids harsh shadows and bright spots which stabilizes RGB and depth data (Cippitelli et al., 2015; Zollhöfer, 2019). The mounts used to connect the lights should be rigid to avoid drift but flexible enough to adapt to new bin layouts. Finally, communication protocols used such as fieldbuses or low-latency Ethernet keep sensing and actuation tightly synchronized to deliver stable and reliable input for segmentation and depth estimation.

Software Stack and Data Flow

The software stack begins with acquisition: The ROS drivers or vendor SDKs stream RGB-D frames with timestamps and IDs (Janis Arents et al., 2018). While multi-camera setups add synchronization and triggers so that the depth and RGB frames that are captured align in time (Cippitelli et al., 2015; Lachat et al., 2015; Siltala & Latokartano, 2023). Next comes preprocessing. Pre-processing can include undistortion, which corrects lens distortion, depth correction where depth values need to be mapped to the proper pixels, and noise removal where random errors need to be removed from the depth map. Next extrinsic transforms are stored in tf trees in ROS, which allows the system to convert between different spaces. From pixel coordinated to 3D camera coordinated to robot coordinated. (Quigley et al., n.d.; Shaikh & Chai, 2021).

When depth input is given to the SAM model it can be done with shared encoders in the beginning of the fusion process, while late fusion is done with separate encoders and merged before decoding (Boulahia et al., 2021; Kirillov et al., 2023; Zollhöfer et al., 2018). SAM needs prompts to guide segmentation. These prompts are generated from depth edges and surface clusters. Once SAM outputs segment masks the masks are filtered out so that the generated masks align with geometric shapes. Next the 3D segments are

passed to pose estimators or grasp planners. While keeping latency low, batching and reusing of the SAM embeddings across multiple frames are used.

2.6.9 Workflow for Object Segmentation and Grasp Planning

A usual workflow for object segmentation starts with synchronized RGB-D frames and proper calibration. Next prompts are generated from the fused point cloud using two common methods, depth thresholding and surface normal clustering which produces regions which are then projected back on to the RGB frame. These prompts are used to guide SAM (Rabbani et al., n.d.; Yang et al., 2007; Zollhöfer et al., 2018). Afterwards SAM encodes the image once and applies the created prompts to generate segmentation masks. The masks can be cross checked against the main depth distributions of that specific surface to remove invalid masks (Kirillov et al., 2023; Rabbani et al., n.d.; Zollhöfer et al., 2018). The confidence scores obtained for the masks are combined with a geometry check when available and the most reliable masks are passed through for pose estimation or grasp planning.

The filtered maps are then projected on to a 3D space. If CAD models are available ICP can be used to align the masks to the model but with occlusions a method such as semantic PPF would be used to assist with the alignment (Drost et al., 2010; Zhuang et al., 2021). When CAD models are not available geometric aspects such as principal axes, curvature and machine learning models are trained to predict where a gripper can hold an object can be used (Kumra et al., 2020; Shao et al., 2023). To improve the accuracy of the whole process Multiview merging of segments can also be used to avoid some of the guess work in occluded object (Aldoma Buchaca et al., n.d.; Drost et al., 2010)

Finally, when it comes to grasp planning, grasp planning such as MoveIt! select candidates using kinematics and collision checks. However, when the visibility is low the grasp planner will plan a nudge of the scene to change the occlusions and then run segmentation again (Dogar & Srinivasa, 2010; Zeng et al., 2020). If the grasp fails, the force, torque or vision are used to obtain feedback to rerun the segmentation around the remaining

objects. To optimize time per pick, the segmentation and grasp planning parallelly. This makes the whole picking process smoother, minimizing delays.

2.7 Classic computer vision for segmentation and proposal

2.7.1 Bottom-Up Segmentation Approaches

In the beginning segmentation and object detection started by searching for lines or contours in images to identify the boundaries of the objects in the image. If many objects which are smaller are found, they would be in consideration of being grouped into a bigger region if the color and texture of the areas were similar (Nielsen & Nock, 2013). Regions were then given an objectness score to decide if the object was of the foreground or background. The main idea here was to generate as many candidate masks as possible to not miss any subjects in the image. Afterwards classifiers filter out lower probability masks (Girshick et al., 2013; Uijlings et al., 2013).

2.7.2 Region Proposal Generation: Selective Search

In industrial bin picking, however classical methods provided mixed results, the areas where these methods were effective are when lighting conditions were good and objects were clear with clear surfaces with distinct edges and boundaries. But in bin picking scenarios that was rarely the case. The bins are messy and cluttered and with low contrast, the performance of these systems dropped dramatically. A common scenario is when identical objects are packed tightly together, edge detection systems would not identify them separately most of the time. The classical methods did not understand the scene and even consider shadows or surface mask as object boundaries (Danielczuk et al., 2019; Shi & Malik, 2000)

Methods such as selective search improved the traditional segmentation by including multi scale segmentation. Objects of many sizes were analyzed on various cues such as color, texture and shape. This helps in identifying parts of many sizes in a bin. The issue that arose with this method is that it produced too many overlapping boxes requiring extra filtering afterwards.

2.7.3 Sliding Window Detection

In these methods the image was scanned part by part of features like HOG (edge patterns or SIFT (key points). Each scanned patch of the image would be described in these features, next a classifier such as a SVM would decide if the patch contains an object. This method was able to detect objects even with weak or noisy edges but was computationally heavy and struggled to detect unusual shapes compared to modern deep learning methods (Dalal & Triggs, 2005; Girshick et al., 2013; Lowe, 2004).

2.7.4 Voxel-Space Processing

The classic methods also extended to 3D recognition, RGB-D data was converted to point cloud by running box detectors through them and creating voxels or 3D pixels. This method was effective when the objects being detected had uniform textures or colors like in bis filled with identical parts. Although voxel space segmentation is simpler and less effective than today's state of the art it worked reasonably well (Maturana & Scherer, 2015).

While effective at the time, there were some challenges that needed to be addressed in an industrial setting. Since most of these methods relied on handcrafted feature mapping, they had to be returned all the time when the product was changed, or the parts changed (Zeng et al., 2018).

Another major challenge was occlusion, hierarchical segmentation failed in this aspect due to merging two objects into one or splitting one object into many segments (Le & Lin, 2019).

But even though these flawed methods provided the foundation for modern approaches, the modularity of classic methods in contour detection, region merging and finally proposal scoring continues in modern perception pipelines powered by deep learning. The visual cues from classical methods are also useful when combined with depth or pose estimation. They are also useful as baseline or sanity checks in domain adaptation (Danielczuk et al., 2019).

In the next section we will investigate the current deep learning methods used in bin picking segmentation scenarios.

2.8 Supervised Learning Methods

2.8.1 Overview

Deep learning pipelines have changed how robotic systems can navigate problem solving in cluttered bin picking scenarios, it offers a level of adaptability that older hand-crafted feature matching pipelines could not. Using deep learning models such as convolutional neural networks (CNN) robots can look at images semantically and create object masks. A well-known example is the Mask R-CNN, where it can integrate detection and segmentation into the same pipeline giving it the ability to segment single objects in a cluttered scene (He et al., 2018). Where classical methods used low level features to detect objects, deep learning methods can capture low level features such as edges and textures in early layers of the network and high-level features such as shapes in deeper layers at various size scales (Geirhos et al., 2018; Jogin et al., 2018). This allows these methods to infer objects in situations of high occlusion or low lighting.

In bin picking scenarios CNN based segmentation alone is not enough. It needs further information for the robot to plan on how to pick the object, therefore the pose is necessary. To obtain this some pipelines feed the feed the segmented masks to pose estimation modules to do so. Point Pair feature matching is one of the methods, however these methods can fail if the segmented object fed is occluded which can lead to increasing grasping errors (Aldoma Buchaca et al., n.d.; Danielczuk et al., 2019; Drost et al., 2010; D. Liu et al., 2021; Zhuang et al., 2021). To deal with these issues researchers had come up with joint methods that combine segmentation and pose estimation such as PPR-Net, which obtains pose information directly from a point cloud and estimated pose while having awareness of masked segments to average the poses over these regions to get stable results. These methods have shown improvements of 15-41% over previous (Dong et al., 2019). From an engineering point of view, it is inefficient to only focus on models seeing and identifying the objects, it needs to have an idea on if the objects detected can be grasped. By embedding affordance logic into deep segmentation models, the graspability of the objects can be determined. Which is what the GQ-CNN model does by using depth data to determine the success of a parallel jaw gripper success for candidate points (Mahler et al., 2017; Xie et al., 2023b). Thereby reducing the number of failed attempts on points that are visually clear by low in graspability.

There is another issue that deep learning models fail at. It is when they are trained on one specific type of object. But if this network is used on a different object, its accuracy can drop significantly (Danielczuk et al., 2019). To solve this foundation model such as Segment Anything has been used. These models have been designed to detect general object boundaries of various kinds and filters can be implemented depending on the industry. While adaptability is important, speed is also important. Which is why in light weight models such as FastSAM optimized CNN backbones have been used instead of computationally heavy encoders, allowing them speed in detection as well as adaptability (Zhao et al., 2023).

2.8.2 Model Training and Validation

As discussed previously with regards to segmentation strategies in the training stage the RGB and depth images need to be aligned perfectly for the model to learn anything relevant from the data. (Cordeiro et al., 2023; Zhuang et al., 2021) While industry-based data sets are small and specific, it can lead to overfitting. Therefore, pretraining models on large generic datasets helps the model to identify a broad variety of features which then can be fined tuned with the industry dataset (Kirillov et al., 2023). The loss functions are also optimized so that it considers Dice loss and focal loss together. Again, the batch composition such as amounts of clutter and clutter need to be balanced to prevent bias. When it comes to validation in segmentation models in bin picking, the model should not only be validated on clean benchmark data but also on real world use case scenario outcomes like pose accuracy and grasp success. This also needs to be tested across various sensors and lighting conditions to validate robustness (Cordeiro et al., 2023; Zhuang et al., 2021). Finally, calibration needs to be considered as it can insert its errors as model errors if not dealt with properly.

2.9 Reinforcement Learning

Reinforcement learning is different from supervised learning as it is not learned from labels but by trial and error. In bin picking a robot cannot always succeed with a one-shot grasp, in reinforcement learning it can learn extra strategies like pushing or reorienting objects to grasp the target object (Laskey et al., n.d.; Zeng et al., 2020). If the robot only sees RGB it can miss important 3D information. By adding depth data, it can update its policy accordingly (Cordeiro et al., 2023; Zhuang et al., 2021; Zollhöfer et al., 2018). Reinforcement learning relies heavily on reward, and if the only reward was the success of picking up the object, learning would be too slow. Therefore, the reward also needs to include reducing occlusions and aligning the gripper with the best grasp angles to

speed up learning (Kumra et al., 2020). The use of optimum force and torque can also be a part of the reward policy.

The various RL algorithms balances stability and efficiency at different proportions. For example, DQN variants work for discrete actions (e.g., suction, pinch, side grasp) It works well for small action sets and is efficient but struggles with noisy and very large actions, so Double DQN is used to reduce bias (Iriondo et al., 2021; Joshi et al., 2020). Continuous-action methods like PPO and SAC generate smooth motions, making them better for fine manipulation but it also needs more samples to learn from than value-based methods like DQN.

Having established a broader understanding of the requirements for accuracy and efficiency in bin picking, and how these factors can optimize industrial environments, the next step is to outline the methodology. Specifically, we focus on designing an accessible, simple, and modular segmentation pipeline built on SAM, aimed at addressing the limitations identified in prior work and enabling practical deployment in bin picking scenarios.

3 Methodology

3.1 System Setup

The experimental setup shown in figure 2 used for these tests are simple and easy to implement. Hardware wise a depth and RGB intel RealSense D435i was used to capture RGB-D data. It provides a resolution of 1280x720 pixels up to 30Hz, with an operational range from 0.3 m to 3m. As for processing, a laptop with an intel i5-7300HQ at 2.5 GHz, 16 GB of RAM and a Nvidia 1050Ti GPU that supports Cuda were used.

As for the software the IDE used was PyCharm, a segmentation model implemented in PyTorch called Segment Anything Model (SAM) was used with its ViT-B encoder. The other libraries used we OpenCV for image processing and NumPy for mathematical calculations. A custom scoring system was created to determine the best mask, taking into account factors such as median depth, mask areas and border penalties.

Figure 2. Real dataset capture setup including a depth and RGB intel RealSense D435i camera and a laptop with an intel i5-7300HQ at 2.5 GHz, 16 GB of RAM and a Nvidia 1050Ti GPU.

3.2 Image acquisition and loading of images

As discussed further down this study, a subset of the Siléane dataset was used to run a quantitative analysis of the discussed pipeline and a RealSense RGB+D camera was used to collect a set of data to run a qualitative analysis of the pipeline. As expected, there are two methods that needed to be used to collect and feed the image data in those instances.

3.2.1 RealSense camera image acquisition and dataset creation.

```
pipeline = rs.pipeline()
config = rs.config()
config.enable_stream(rs.stream.color,   640,   480,   rs.for-
mat.bgr8, 30)
config.enable_stream(rs.stream.depth,   640,   480,   rs.for-
mat.z16, 30)
pipeline.start(config)
align = rs.align(rs.stream.color)
for _ in range(30):  # allow auto-exposure to settle
    frames = pipeline.wait_for_frames()

frames = pipeline.wait_for_frames()
aligned_frames = align.process(frames)
color_frame = aligned_frames.get_color_frame()
depth_frame = aligned_frames.get_depth_frame()
```

Figure 3 RealSense pipeline setup for RGB and depth streams which are combined using the object rs.align .

A pipeline is created to start streaming from RealSense camera, it creates two streams, a standard RGB stream and a stream of 16bit unsigned integers representing depth. Both streams are synchronized with time but importantly with different perspectives since the camera has an offset between the two sensors. This needs to be realigned with the object rs.align. A lower resolution was selected here to reduce the processing time for the encoding process. Next the image and depth data are collected one at a time as the

position of the objects in the image needs to be changed for each image. A warmup time is given to the camera so that the auto exposure settles. Finally, both data streams are converted to NumPy arrays, and the depth array is converted to mm before being saved as png files.

This cycle was repeated for all images collected.

The images collected here were then manually annotated to mark some probable ground truths. This was done to see how the selection pipeline behaved compared to an intuitive selection.

3.2.2 Siléane Loader

The subset used from the Siléane data was the Siléane bricks dataset. The SiléaneBricks-Dataset loader is a custom dataset class to load the BGR, depth and segmentation files from the specific local folder. It then maps the corresponding image, depth and segmentation files to a specific naming order and creates naming order. Example: "frame_001" → "rgb/frame_001.png". Next the camera intrinsics are loaded from text files provided with the Siléane dataset. It is needed to align the depth and RGB data in a 3D space. After converting the bgr data to RGB data and normalizing depth to convert the raw depth values to meters. Finally, a dictionary is created with the keys below which can be loaded into the segmentation pipeline with SAM and the scoring system.

```
self.intrinsics = self._load_intrinsics(os.path.join(root,
"camera_params.txt"))
depth_raw = cv2.imread(self.depth_map[frame_id], cv2.IM-
READ_UNCHANGED)
depth_meters = self._normalize_depth(depth_raw)
    "rgb": rgb,
    "depth": depth_meters,
    "segmentation": seg_mask,
    "intrinsics": self.intrinsics,
    "frame_id": frame_id
```

Figure 4 Loading of the Siléane dataset with the created dictionary.

3.3 Segmentation Pipeline

Step 1: Depth-based candidate box extraction

While in the initial iteration of the scoring pipeline SamPredictor was used as an option in the segmentation process to reduce the computational load by only segmenting relevant parts of the image, it was found to be less accurate in obtaining proper segmentation mask. The issuing of bounding boxes also seemed irrelevant in most cases as the whole box area needs to be scanned for objects. Which meant that the computation time saved by using SamPredictor was not worth the tradeoff in time gained for the accuracy of the masks obtained by the SamAutomaticMaskGenerator.

Step 2: SAM automatic segmentation

Therefore, the whole image is fed into SAM and its SamAutomaticMaskGenerator utility is used to generate masks for the whole image. This is done by the image encoder, a vision transformer. Then the lightweight mask decoder gives the output (Kirillov et al., 2023)

Step 3: Scoring function

All the segments obtained through the models are then run through a scoring function (penalty). The point of this scoring function is to select objects that are most suitable for bin picking depending on the object's depth, depth variance, area, shape, proximity to the image border and SAM confidence levels. The weights of these specific elements will be left to the user to be tuned intuitively. Although practically the only aspect that would need tuning would be the area dependent weights. It is initially filtered by the median depth of all the pixels in a mask to ensure that the objects detected are in a proper working range and also so that outliers due to random error depths do not affect the selection of suitable masks. Next the standard deviation of depth values of the masks are used to select masks that have consistent depth values. In taking the images of objects in a box it was noticed that the walls of the box in the image tends to come up as good segments

due to it large area and it being close to the camera, therefore it was necessary to normalize the area of the masks to the total image area and penalize out masks that were too large or small (covering >10% or <5% of the image). Finally, it was also necessary to consider that in bin picking, picking objects at the edge of the image would be slightly harder than the ones in the middle, therefore a border penalty was also created. All of the factors that are used to compute a mask score will be shown below including snippets from the code that computes those features.

1. **Median Depth**

$$\tilde{d} = \text{median}(D_{\text{mask}}) \qquad (1)$$

Definition: The median depth of all valid pixels inside the mask.
The median depth was then used to calculate the depth score for each mask. No further equations were needed here. It keeps the calculation simple and universal for all scenarios. By having the depth score as the median depth of a mask, the score for specific objects can defer between various hardware setups but it won't affect the selection of the objects to be picked as it is relative to all the other objects in the same hardware setup.

2. **Depth Variance**

$$\sigma_d = \text{std}(D_{\text{mask}}) \qquad (2)$$

Definition: The standard deviation of depth values inside the mask.
The standard deviation of the mask matters in making sure that the objects cover one single object. If the depths are too varied it might be due to a noisy mask or a mask that covers multiple objects.

```
dv = np.std(vals)
var_score = min(dv / 0.02, 1.0)    # normalize by 2 cm, cap
at 1.0
```

Figure 5 Depth variance normalization.

3. **Area Normalization**

$$A_{\text{norm}} = \frac{|mask|}{H \times W} \quad (3)$$

Definition: Ratio of mask area to total image area.

To avoid the use of masks that were too large or too small to be viable in practical use any mask that was greater than 5% or less than 2% of the image was given a higher penalty.

```
area_pixels = mask.sum()
    area_norm = area_pixels / (H * W)   # fraction of image
covered

    if area_norm < 0.02:  # too small
        area_score = (0.02 - area_norm) / 0.02   # penalty
grows as it gets smaller
    elif area_norm > 0.05:  # too large
        area_score = (area_norm - 0.05) / 0.05   # penalty
grows as it gets larger
    else:
        area_score = 0.0  # ideal range → no penalty
```

Figure 6 Penalty score against the deviation from an ideal range between 5 and 2%.

The area score for the real-world dataset is however one aspect that needs more of a nuanced approach compared to a fixed a fixed boundary score. Here the score is given by how far the segmented mask area deviates from the average ground truth area. This helps to have a better selection of masks generated without retraining on specific data. This, however, can be switched depending on user preference.

4. **Shape Compactness**

$$C = 1 - \frac{4\pi \cdot A}{P^2 + \epsilon} \quad (4)$$

- Where A= mask area, P= contour perimeter.

By using a compactness score the uniformity of the masks will be assessed and scored accordingly to avoid dealing with obscure shapes that might appear. This is done by using the Polsby-Popper compactness score (Belotti et al., n.d.). The implementation of which can be seen in figure 7.

```
contours, _ = cv2.findContours(mask.astype(np.uint8),
                                    cv2.RETR_EXTERNAL,
                                    cv2.CHAIN_APPROX_SIMPLE)

    if contours:
        # Use total perimeter across all contours
        perimeter = sum(cv2.arcLength(c, True) for c in con-
tours)

        # Area is already mask.sum()
        compactness = (4 * np.pi * area_pixels) / (perimeter
** 2 + 1e-6)

        # Normalize compactness into [0,1] range
        # Perfect circle → compactness ≈ 1 → shape_score ≈ 0
        # Jagged/elongated → compactness ≪ 1 → shape_score
closer to 1
        shape_score = 1.0 / (compactness + 1.0)
    else:
        shape_score = 1.0
```

Figure 7 Shape/compactness score using the Polsby-Popper compactness score.

Here the boundary of a shape of a mask is obtained and measured to get the perimeter, which is used to calculate the compactness score, which if ideally fits gives a score of 1. Which would then give a shape penalty score of 0.

5. **Border Penalty**

$$P_{\text{border}} = \frac{1}{\min(\Delta x, \Delta y)+1} \qquad (5)$$

Where $\Delta x, \Delta y$are distances from the mask to the nearest image border.

The masks are the image border was deemed a penalty due to the fact that masks found at the border tend to be incomplete due to being cut off by the camera frame and also

might be too close to the bin walls hindering the picking process. As SAM tend prefere flat objects with large surface area in segmentation, giving a border penalty reduces the possibility of a wall being detected as an object to be picked.

```
edge_dist = min(xs.min(), W - xs.max(), ys.min(), H -
ys.max())
border_penalty = 1.0 / (edge_dist + 1)

#xs.min() → leftmost pixel of the mask.
#W - xs.max() → distance from right edge.
#ys.min() → topmost pixel of the mask.
#H - ys.max() → distance from bottom edge.
```

Figure 8 Border penalty calculation.

As seen above in figure 8, first it finds the smallest distance between the mask and any of the four image borders: the left edge (xs.min()), the right edge (W - xs.max()), the top edge (ys.min()), and the bottom edge (H - ys.max()). That minimum value (edge_dist) represents how close the mask is to the nearest border. Then, the border_penalty is computed as 1.0 / (edge_dist + 1), which means the closer the mask is to the edge, the larger the penalty becomes. The more pixels it has between itself and any image border, lower the penalty score.

Step 4: Best mask and top mask selection

Finally, after running all the masks through the scoring function, *the mask with the lowest score is selected as the best mask*. This mask is compared with the top mask to evaluate the performance of the best mask*. The top mask in the mask with the lowest median depth*. In the evaluation phase where the masks selected needs to be selected with ground truths, the objects that have the highest IoUs with the specific masks are selected as the ground truth object that the rest of the evaluation metrics are carried up on.

For the best mask selection, the penalties were added up given their own weights for each category. SAMs own IoUs and ground masks were also considered in this score. Higher the total score is for a segment lower the rank in terms of being picked.

```
score = (
        weights["depth"] * depth_score +
        weights["var"] * var_score +
        weights["area"] * area_score +
        weights["shape"] * shape_score +
        weights["border"] * border_penalty +
        weights["sam_iou"] * sam_iou_score +
        weights["sam_stability"] * sam_stability
    )
```

Figure 9 Structure of the scoring equation.

3.3.1 Evaluation Protocol

- **Metrics**: IoU, precision, recall and F1.

The performance of the model was evaluated on a quantitative basis using standard segmentation metrics as mentioned above.

Intersection over Union measures the overlap between the predicted masks and the ground truth masks provided in the Siléane dataset itself.

Precision gives us how clean and accurate the predicted mask is, which should ideally include only the object that needs to be detected and not the background as well.

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives + False Positives}}$$

Recall shows how much of the intended object was successfully identified. More of the object if included in the segment, higher the recall.

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives + False Negatives}}$$

F1 score takes into consideration both precsion and recall. It is a balance between the two. A high F1 score would mean a high recall and precision.

$$F1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

In addition to the quantitative validation on the Siléane dataset, a qualitative analysis of the generated masks for both the dataset and a small sample of manually annotated real word images would be compared so that the performance of the model can be analyzed in an artificial environment and in the real world.

4 Experiments

4.1 Siléane Dataset Tests

The evaluation was carried out using the Siléane bricks dataset, which contains both synthetic and real bin-picking scenes designed to replicate cluttered industrial environments. The segmentation pipeline was executed across 30 frames with various object orientations, number of objects and positions. This provided sufficient data to benchmark the performance of the depth aware scoring factors and to compare with different configurations while keeping the computational load manageable.

4.1.1 Results with standalone SAM.

The results of the performance of SAM on its own can be seen next. This was SAM as a standalone segmentation model used to segment the Bricks subset of simulated RGB and depth data. SAM was run on the first 10 of the images. The SAM model was able to run through and generate extremely good results on the objects using only the simulated RGB data.

Table 1 SAM segmentation across the Siléane bricks dataset

	IoU	**Precision**	**Recall**	**F1**
Average	0.94	0.965	0.961	0.963

Table 2 Segmentation across image brick_0_008 using SAM for all objects in the image.

frame_id	object_id	IoU	Precision	Recall	F1
brick_0_008	8192	0.975612	0.9992301	0.976346	0.987655
brick_0_008	16384	0.955426	0.986987	0.967615	0.977205
brick_0_008	24576	0.955789	0.9752025	0.979597	0.977395
brick_0_008	32768	0.970503	0.9781013	0.992059	0.98503
brick_0_008	40959	0.969382	1	0.969382	0.984453
brick_0_008	49151	0.800253	0.8043202	0.993721	0.889045
brick_0_008	57343	0.955275	1	0.955275	0.977126
brick_0_008	65535	0.971714	0.9948893	0.976589	0.985654

As observed in table 2 above, the results are extremely accurate and precise. Therefore, it was clear that SAM was extremely accurate in semantic segmentation. These results lead to considering the assumption that it would also be able to handle the use of depth obtain accurate results in a bin picking scenario as well. But it should also be noted that to come up with these segment masks that fit almost perfectly, SAM generated 300 to 400 masks per scene, while only having less than 10 bricks to detect in any given scene. This high number of masks can also be due to the simulated nature of the dataset fed into SAM.

4.1.2 Results with SAM plus the custom scoring function and top most mask.

The initial iteration of the scoring function used SAM predictor over the whole image to obtain segment masks to filter using the scoring pipeline. After the use of depth and custom scoring function, these were the results obtained from the model across 30 images of the Siléane bricks dataset.

Table 3 Initial scoring pipeline results with SamPredictor across 30 images of the Siléane bricks-dataset.

Average	**IoU**	**Precision**	**Recall**	**F1**
Best mask	0.22396	0.22583	0.90604	0.35505
Topmost mask	0.56760	0.8969	0.61979	0.66475

SAM initially performs poorly in providing proper masks that could be filtered out for the objects being selected as the most suitable for bin picking. Using the metrics collected for the segmentation masks selected using the scoring function, there are two notable aspects. They are the low average precision and the highest average recall of the masks selected. While the Topmost masks tend to do much better overall. With average recall values and high precision, leading to notable average F1 score. Below in table 4 are some of the scores for 6 separate images in the dataset.

Table 4 Subset of the topmost mask's performances for 6 separate images in the dataset.

TopMask IoU	**TopMask Precision**	**TopMask Recall**	**TopMask F1**
0.768019727	0.794388856	0.958570076	0.868790902
0.845803133	0.981549315	0.859467807	0.916460827
0.025351282	0.977272727	0.025366237	0.04944897
0.666335291	0.990063113	0.670821581	0.799761362
0.553863999	0.640509725	0.803702924	0.712886069
0.060431347	0.371879106	0.067300832	0.11397503

It was also noteworthy that the top mask was also perceptible to irregularities in its precision and recall scores, scoring a 0.06 for recall in one instance. While in figure 10 it is visible neither of the methods perform well in selecting the topmost or best mask to pick accurately. The green brick would be the ground truth mask, and the bright red highlight would be the generated mask that was selected after SAMs mask generation.

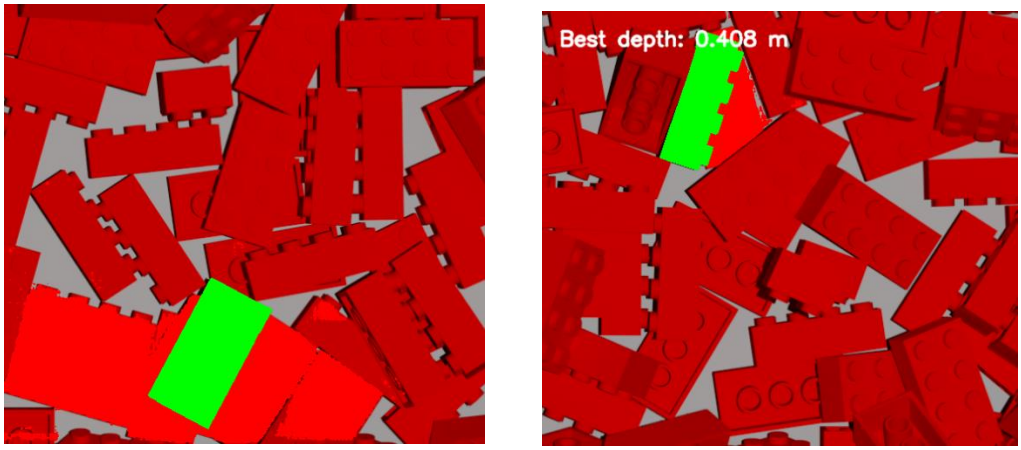

Figure 10 Top mask vs the best mask where the green brick would be the ground truth mask, and the bright red highlight would be the generated mask that was selected after SAMs mask generation.

The two masks above were one of the worst mask selections from the images that were obtained from the top mask and the best mask (masks obtained through the scoring pipeline). While these values are far from standards necessary for use in an industrial environment, let's break down the results as to what causes such results. While SAM was able to obtain extremely accurate segmentations of all the bricks in the Siléane dataset without using depth data to select the best object for bin picking. It also created substantial number of potential masks and used its own scoring system to select the best masks that would fit all the objects in the image dataset. Thereby obtaining masks that performed extremely well in all scoring metrics. But when obtaining the results of the masks using our scoring system and lowest overall depth and using them to select the best brick to pick, the masks that fit the description would not fit the brick perfectly as we hoped for. The most likely culprit for this specific issue is the overabundance of masks that SamPredictor creates initially. This makes the creation of the best object selection complicated and time consuming to calculate. We will investigate the refining the mask selection process next. The refinement process to have better mask selection was, to take into consideration the use of SAMs own confidence scores as filter after the mask generation through SamAutomaticMaskGenerator, which performs non maximal suppression (Kirillov et al., 2023).

```
masks = mask_generator.generate(rgb)
filtered_masks = [
    m for m in masks
    if m.get("predicted_iou", 0) > 0.8 and m.get("stabil-
ity_score", 0) > 0.9
]
if not filtered_masks:
    continue
```

Figure 11 Filtering of the generated masks from the use of SAMs confidence and stability scores.

By using these confidence scores, the masks fed into the scoring system would be of a higher quality and have a better chance of aligning with the objects in the image. After this initial filtration process. The evaluation metric reflected this change. It should also be noted that in SAMs generate() function, non maximal suppression (NMS) is included to remove almost identical masks (Kirillov et al., 2023).

Table 5 Performance of the masks after the use of SAMs confidence scores and NMS on the 30 Siléane bricks dataset images.

Average	**IoU**	**Precision**	**Recall**	**F1**
Best mask	0.783426	0.880086	0.876465	0.870563
Topmost mask	0.505799	0.947529	0.537752	0.580217

When comparing table 5 with the performance of the scoring system before the use of SAMs confidence levels as an initial filter. There are performance increases of more than 100% on almost all metrics, excluding BestMaskRecall. While the performance of the Topmost mask has decreased slightly, the best masks outperform in a more stable fashion than what the topmost mask was able to provide before the SAM filtering process.

Next the improved filtering system is run through an ablation study to find out what aspects of the scoring system are responsible for most of the performance gains.

4.2 Ablation Study

In the ablation study we will be starting off with only using the depth values in the custom scoring system and adding each aspect of the scoring system one at a time and monitoring how they perform with addition or removal of a metric. The weights used for each run in the ablation test are given in figure 12 below.

```
ABLATION_CONFIGS                    =                    {
"depth_only":    {"depth":30.0,    "var":0.0,    "area":0.0,
"shape":0.0,   "border":0.0,   "sam_iou":0.0,   "sam_stabil-
ity":0.0},
"depth_var": {"depth":30.0, "var":0.5, "area":0, "shape":0,
"border":0.0,        "sam_iou":1,        "sam_stability":1},
"depth_var_area":   {"depth":30.0,   "var":0.5,   "area":5.0,
"shape":0,   "border":0,   "sam_iou":1,   "sam_stability":1},
"depth_var_area_shape": {"depth":30.0, "var":0.5, "area":5.0,
"shape":1.5, "border":0.0, "sam_iou":1, "sam_stability":1},
"depth_var_area_shape_border":   {"depth":30.0,   "var":0.5,
"area":5.0,    "shape":1.5,    "border":6.0,    "sam_iou":1,
"sam_stability":1},
```

Figure 12 Ablation configurations for the Siléane bricks dataset.

4.2.1 Results of the performance considering only the depth results.

Table 6 Best mask performance with depth alone.

	IoU	**Precision**	**Recall**	**F1**
AVERAGE	0.505799	0.947529	0.537752	0.580217

The results show the performance of scoring systems performance over 30 scenes in the Siléane bricks dataset. But as it is clearly visible it is not handling the mask selection process well for this process. But it is identical to the performance of the top mask selection, which makes the most sense. Almost all of the segmentation by custom scoring with only depth and the top mask selection is identical. This is due to the fact that using only a depth penalty on the masks and selecting the mask that has the least depth would

give us the exact same result. It can again be observed that while depth can be an important aspect in the mask selection process, it cannot be the sole criteria as seen by the first two images, an object in the border of the image was selected as the object to be picked. This would improve in the coming steps.

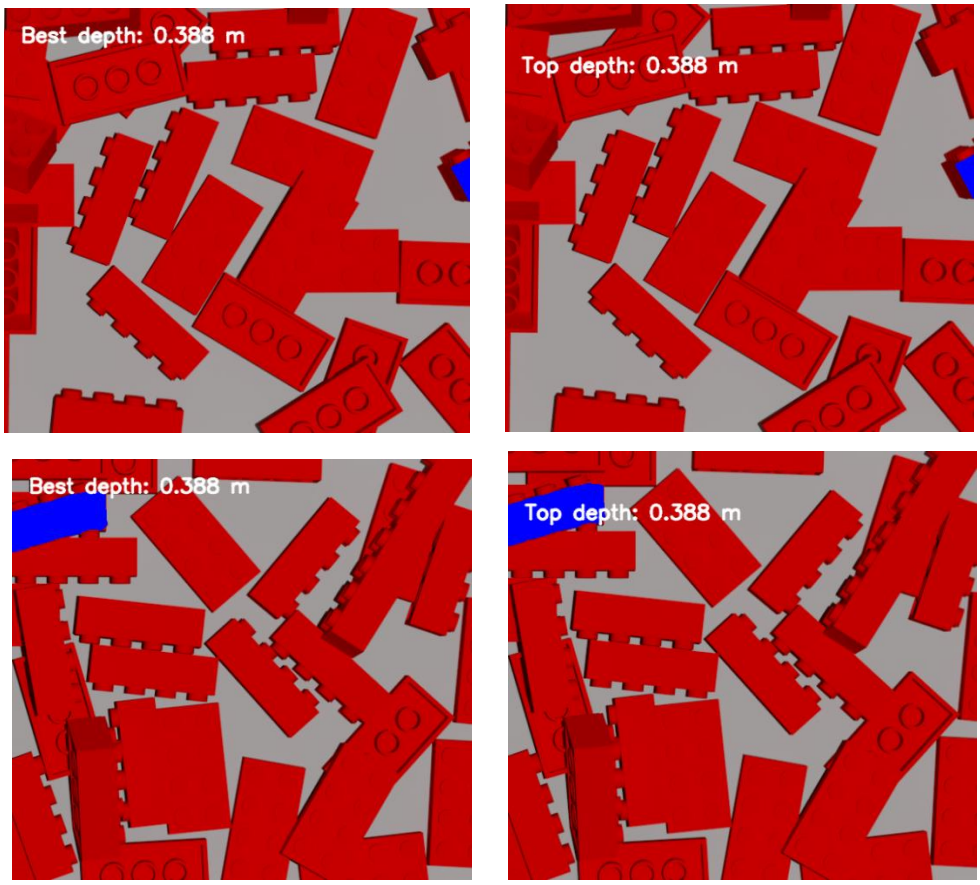

Figure 13 Depth only best mask vs Topmost mask.

4.2.2 Results of the performance considering the depth and depth variance results

Table 7 Best mask performance with depth and depth variance.

	IoU	**Precision**	**Recall**	**F1**
AVERAGE	0.471308	0.944517	0.505817	0.552507

By considering the variance in height the results seem to select smaller masks although they still are focused on the image with the least depth. It also does not seem to focus on the topmost image as much due to the addition of variance. This factor can be an influence on how the scoring system needs to be tuned for a real-world dataset. While some datasets might need to be penalized on depth variance, the bricks due to their specific shape might do better without a lot of influence from depth variance. Below in figure 14 are few of the top candidate masks for an image in the dataset. It should be noted that the masks of some cutouts (circles) of the bricks were selected in the latter images in figure 14, this is due to the area of the mask not being considered.

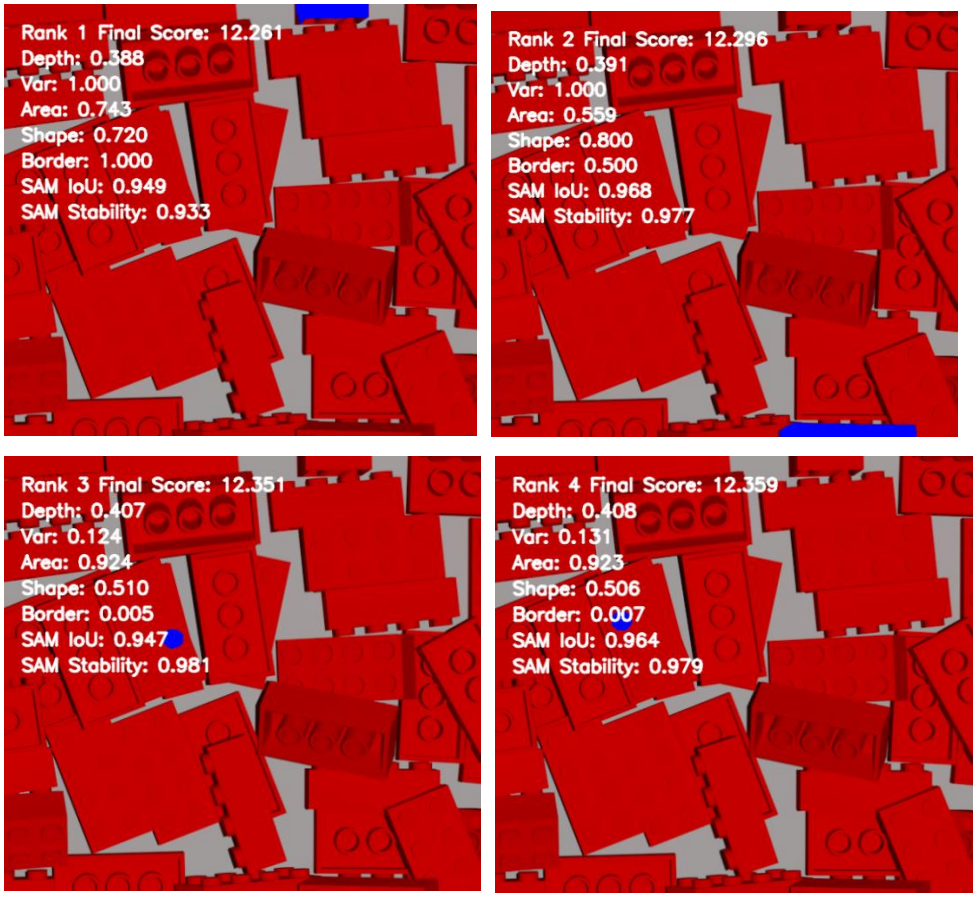

Figure 14 Depth + variance best masks selection ranked for one image.

4.2.3 Results of the performance considering the depth, depth variance and area results.

Table 8 Best mask performance with depth, variance and area.

	IoU	**Precision**	**Recall**	**F1**
AVERAGE	0.748533	0.832725	0.883677	0.847803

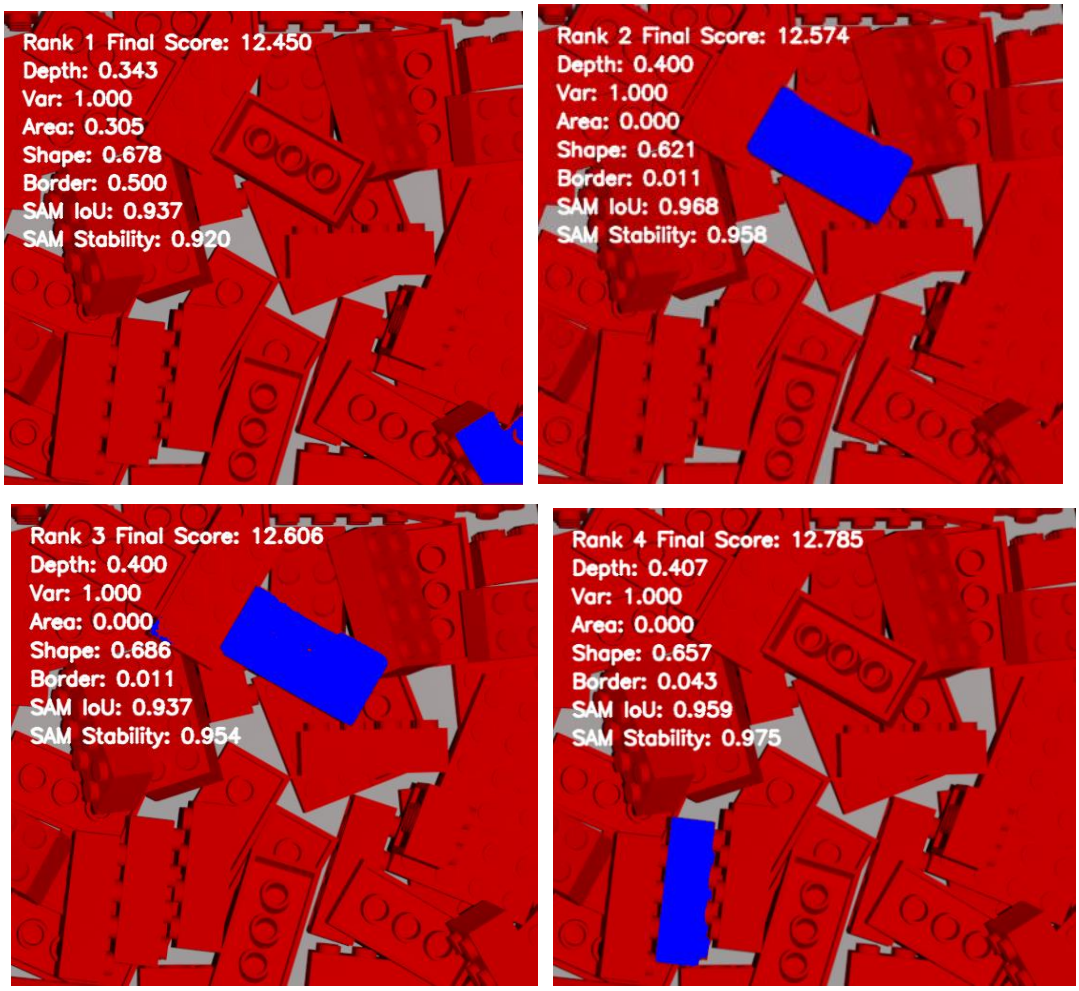

Figure 15 Depth + variance + area best masks selection ranked for one image.

While area was given less weight than the depth as can be observed, in the rank 1 image it has a higher area score, which is the worst out of the 4 but it has much more of a lower depth, making it the most suitable at this stage. It is also worth noting that the introduction of area penalty made way for the selection of masks that were similar to the objects that they belong to.

4.2.4 Results of the performance considering the depth, depth variance, area and shape results.

Table 9 Best mask performance with depth, variance, area and shape.

	IoU	Precision	Recall	F1
AVERAGE	0.762948	0.845712	0.888367	0.857336

While no major improvements were seen, all performance metrics in terms of IoU, Precision and F1 scores improved by approximately 0.02. The shape penalty would also implement more standard, well rounded shapes when it comes to mask selection. Therefore, it would be a safe assumption to say that these improvements would come through a more refined mask that stays within the borders of the specific ground truth masks, while in the terms of bin picking, no major changes were seen in the sense of the object selected.

4.2.5 Results of the performance considering the depth, depth variance, area, shape and border closeness results.

Table 10 Performance with the full best mask pipeline

	IoU	Precision	Recall	F1
AVERAGE	0.783426	0.880086	0.876465	0.870563

Figure 16 Shows the effects of adding the border penalty to the scoring function has on the segment selection for the same image.

Figure 16 Segmentation without (left) and with (right) border penalty, giving the mask which is away from the border for easier grasping from a robotic arm**.**

Again, with the addition of the border penalty all the segmentation metrics saw an increase of 0.02 approximately. This would be possibly due to the availability of cleaner masks away from the border but more importantly this gives the selected object a better chance of being picked more easily due to the ease of maneuvering a robotic arm away from the borders of a bin and having a full view of the entire object in frame. The takeaway here would be that the weights of the border penalty also need to be higher to make sure the objects selected from rest of the elements in the scoring system do not fall into the edge of the image boundary.

Finally let's consider all the results of the ablation study overall.

Table 11 Segmentation performance across all ablation settings in the best mask scoring pipeline and topmost masks

Scoring parameters	**IoU**	**Precision**	**Recall**	**F1**
depth	0.505799	0.947529	0.537752	0.580217
depth/var	0.471308	0.944517	0.505817	0.552507
depth/var/area	0.748533	0.832725	0.883677	0.847803
depth/var/area/shape	0.762948	0.845712	0.888367	0.857336
depth/var/area/shape/border	0.783426	0.880086	0.876465	0.870563
Topmost mask	0.505799	0.947529	0.537752	0.580217

While all the results shown in table 11 ablation study have been performed after being prefiltered with the SAM confidence scores, when starting off with only the depth the results were identical to obtaining the mask with least median depth which was expected. When the depth variance penalty was introduced the performance in segmentation dropped slightly. But what needs to be considered is that in a compact environment the probability for SAM to combine masks on various depth levels is higher, therefore depth variance filters are needed even though in this case which might mean that the variance levels need to be lowered or not used. Addition of the area penalty on top of the variance penalty brought the performance metrics of the scoring pipeline to respectable levels however since the area filter is added manually between a specific range it was also decided to add an option to select an average area that a ground truth mask of a specific object that would be selected for bin picking as a comparison to give an area score, therefore further a mask area goes from the average area of the ground truths of that dataset higher the penalty. The shape score does increase the performance of the scoring system helping it to select smoother masks. When it came to the border score the performance again did not change massively however it was found that the while the segmentation was at a good level, when it came to the object picking from a bin, selection of the masks from the central areas of the bin gives the robotic arm more room

to maneuver, therefore the weight on the border score was also increased for the final scoring pipeline. While the depth penalty weight was also increased. The following section will investigate how the object selection has improved with regards to bin picking.

4.2.6 Final scoring system and an Intuitive look in terms of bin picking.

```
{"depth":30.0, "var":0.0, "area":5.0, "shape":1.5, "bor-
der":10.0, "sam_iou":0.5, "sam_stability":0.5}
```

Figure 17 Final weights for the scoring pipeline

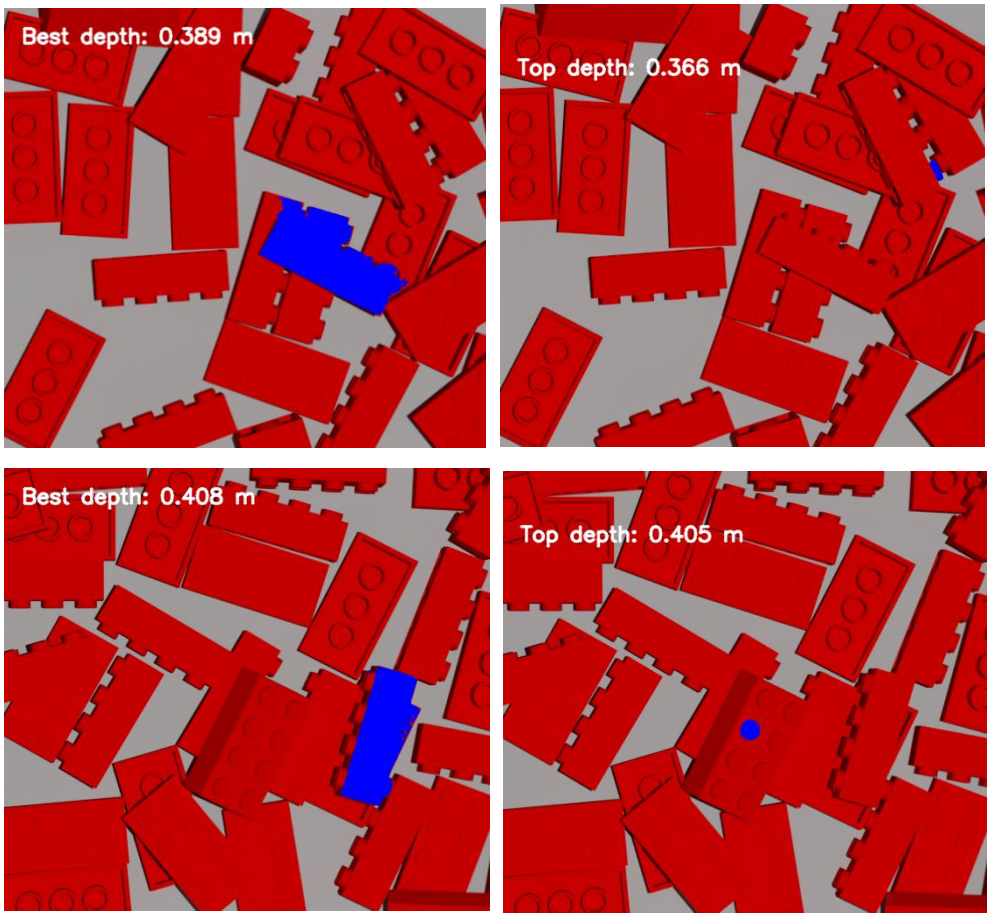

Figure 18 Brick 32/36 best vs topmost mask selection comparison.

The changes made to the scoring system in figure 18 have brought the selected segmentation masks for bin picking further away from the borders of the image. The biggest issue with the topmost mask selection is the masks selected more frequently do not

cover the whole object and lock onto a minor part of the object. This mostly does not occur with the best mask strategy due to the area restrictions. The issue that the best mask selection pipeline faces at times is the selection of masks that involve two or more objects as seen in figure 19. This could be an issue that needs to be looked into further in the future.

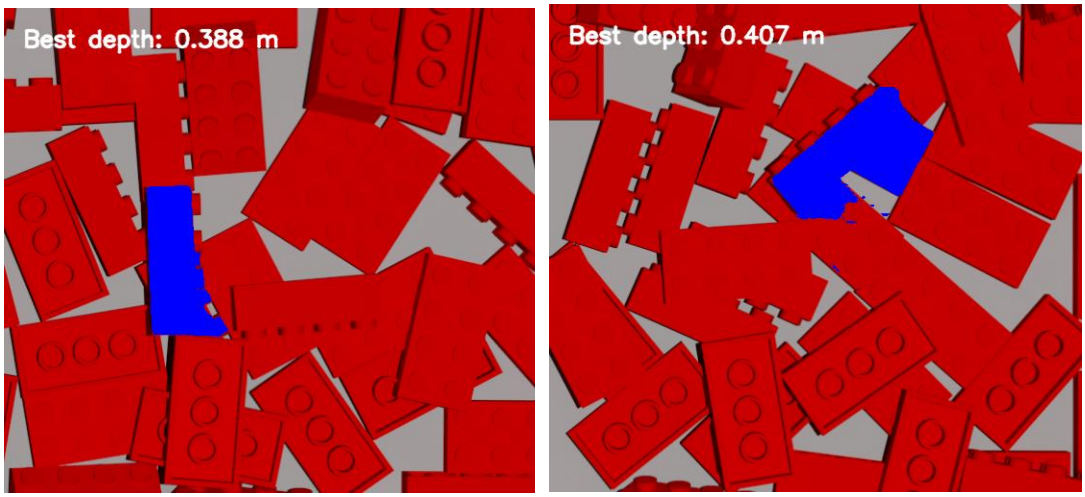

Figure 19 Brick 40 and 42 best mask segmentation.

However, the updated scoring also gets the segmentation into the center of the image in most situations which is important. This can be seen more clearly in figure 20 below when compared with the topmost mask.

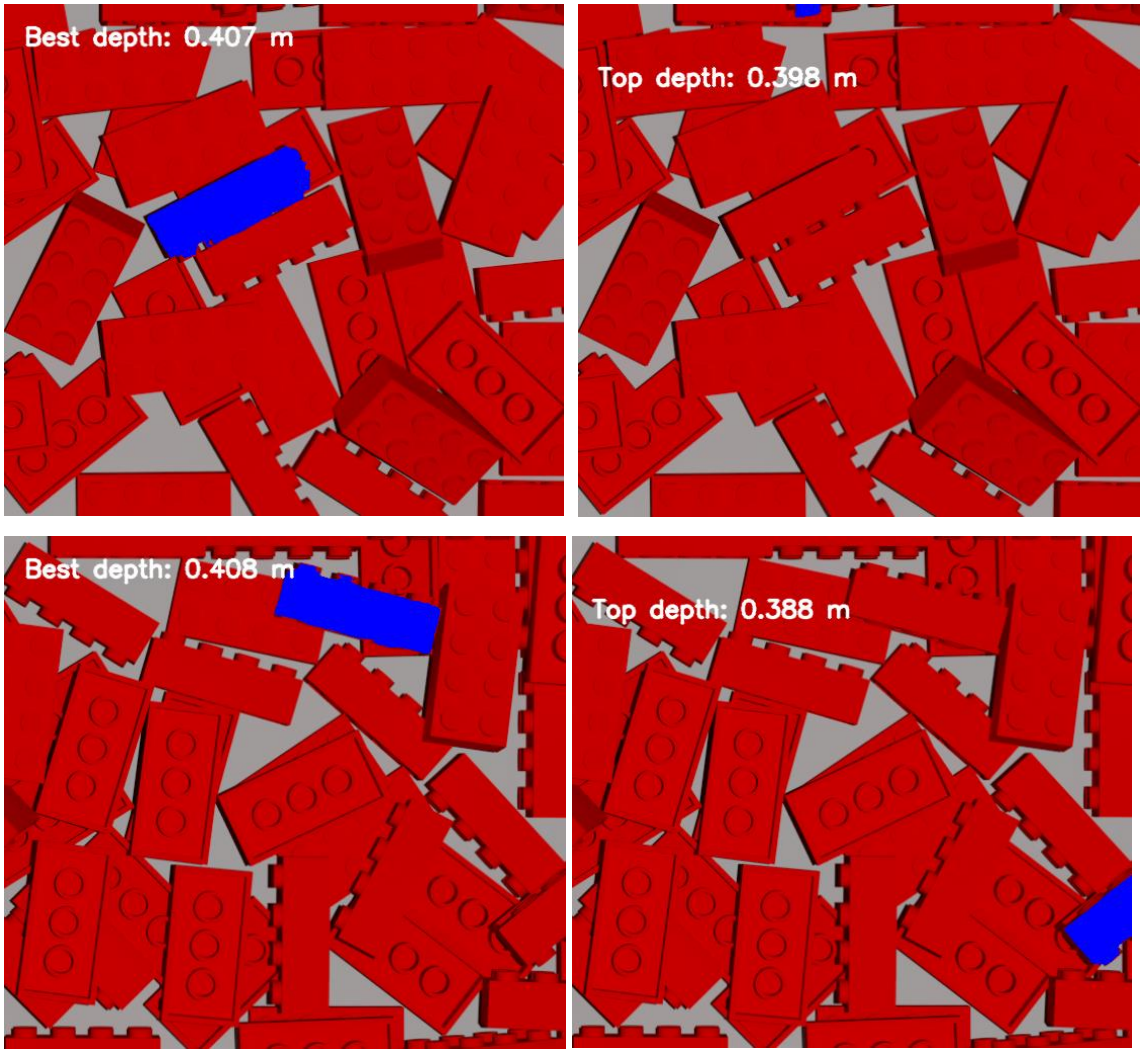

Figure 20 Brick 43 and 52 best vs topmost mask showing that the best mask consistently tries to strike a balance between lowest depth and masks away from the border while the top most mask tend to select any mask with the least depth.

Another aspect that is visible here is that even though the pipeline is not perfect in identifying objects it does perform well in most instances, as seen in the best depth images. All the original depth images of both the Siléane and the captured datasets will be uploaded into an online database to be accessed, and links are available in.

In the following section the study focuses on real world datasets.

4.3 Real-World Demo

To have a qualitative analysis of how SAM performs on real-world data obtained through an off-the-shelf RealSense RGB+D camera, 20 images with mixed object orientation of reflective metal objects that were identical in shape and size were taken. The best object to pick (intuitive) for a robot was selected manually and marked. Finally, the selected object masks were selected for both the topmost mask and the best mask and the overlap was visually compared and result documented. This would provide us with a clearer understanding of how SAM would behave is bin picking in scenarios and environments that are far from ideal. It should be noted that these images were taken in room lighting conditions and are less controlled and calibrated than a factory environment. The object masks in blue would be the selected segmentation mask by the scoring system and masks in green are the top masks. The dot structure that is visible in a pink hue is from the depth measurement IR dots that are visible to the camera alone.

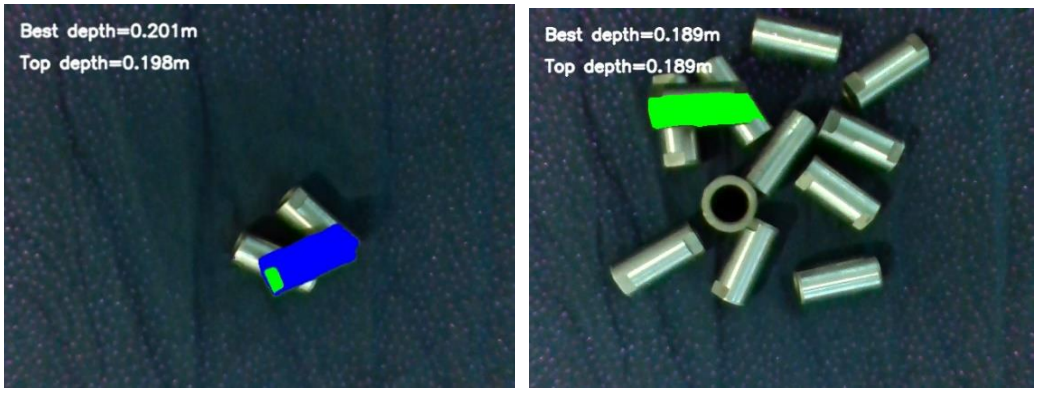

Figure 21 Best and topmost mask overlays, the object masks in blue would be the selected segmentation mask by the scoring system and masks in green are the top masks.

From the observations from the images (figure 21) taken the topmost mask can overlap fully with the best mask at times but the topmost mask cannot be relied upon to obtain the full mask for the object. Another positive observation for the best mask is that the segmentation on all 20 of the pictures taken did not involve the combination of two or many objects. As was evident in the Siléane bricks dataset, the topmost mask is at a

disadvantage when it comes to selecting the correct object mask especially when there are distinct shapes on the object itself as seen below in figure 22.

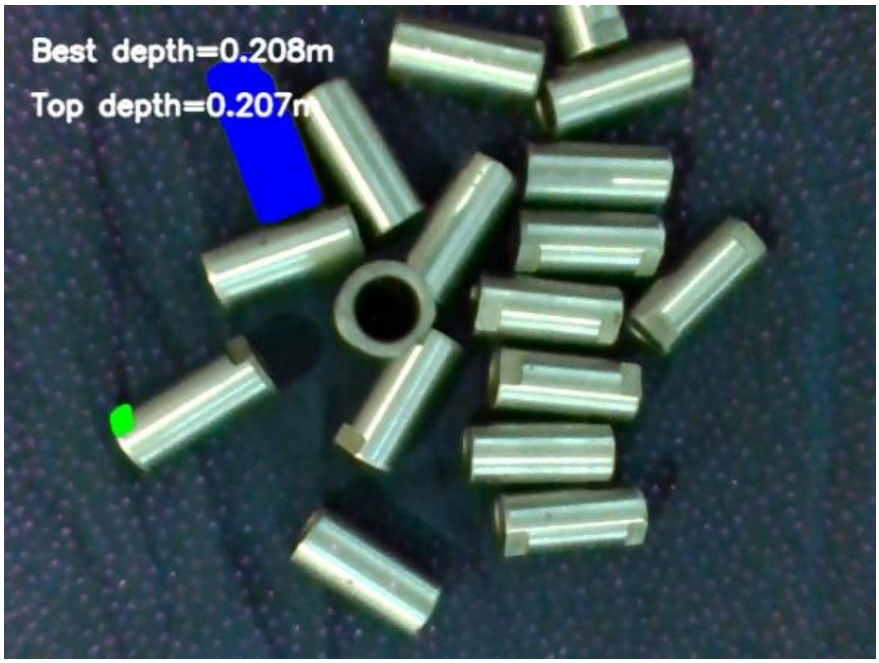

Figure 22 Improper segmentation for the top mask (in green)

It should also be noted that during the data collection process there were multiple masks that dept data could not be accounted for, which might be due to the material nature or device inaccuracy, to make sure most masks have a median depth values the Navier stokes algorithm provided by the Open CV library was used to fill in empty mask depth information with (Bertalmio et al., n.d.).

```
vals = depth_meters[mask]
vals = vals[vals > 0]

if len(vals) == 0:
    # Fallback: use nearest-neighbor depth for mask pixels
    depth_filled = cv2.inpaint(
        (depth_meters * 1000).astype(np.uint16),   # convert
to mm for inpainting
        (depth_meters == 0).astype(np.uint8),      # mask of
invalid pixels
        3,                                         # radius
        cv2.INPAINT_NS                             # Navier-
Stokes method
    ).astype(np.float32) / 1000.0                  # back to
meters

    vals = depth_filled[mask]
    vals = vals[vals > 0]
```

Figure 23 Missing depth value filling solution using the Navier stokes algorithm provided by the Open CV library**.**

In figure 24, showing the gray scale depth images, we could see the areas in black are areas with missing depth values that need to be filled in.

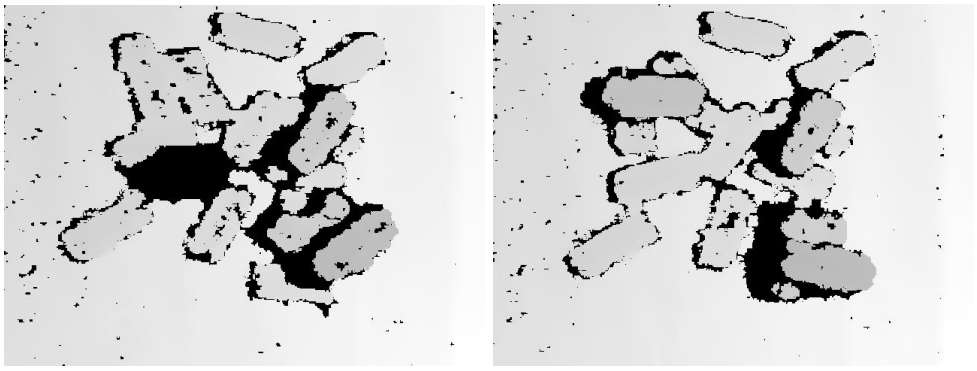

Figure 24 Gray scale depth images of image 18 and 19 of the real-world dataset collected using the RealSense RGB-D camera, where darker colors mean closer it is to the camera except the pure black areas meaning 0 mm depth or no valid depth reading.

5 Discussion

5.1 Summary of findings.

While SAM by itself does an excellent job in creating segmentation masks as seen previously for the bricks subset from the Siléane dataset. To consider what aspects or attributes would bring out the best segmentation mask when considering a bin picking scenario, it was decided to design a scoring pipeline and run it across all the masks that SAM created and compare it against what the performance would be when compared against a much simpler lowest median depth filter (top most mask). The results favored the simpler method of filtering until the introduction of the shape penalty. While the performance of the lowest median depth filter was not acceptable it was almost twice as good as the scoring pipeline in terms of quantitative output before the introduction of the shape penalty. This was mostly due to the overabundance of masks created by SAM model, which required a precise and more complicated scoring pipeline. Therefore, the same filters were decided to be run on the masks after they had been filtered out by using SAMs own confidence and stability scores and the addition of the shape penalty. This resulted in much improved results quantitatively in all aspects of IoU, precision, recall and F1 scores. While the scores showed that the scoring pipeline was doing better compared to the lowest median depth mask filter, when the overlay images were compared it could be noticed that even though the topmost mask were getting lower scores than the custom scoring system, it tends to select more logical candidates in segmenting for bin selection. With closer analysis it was also noticed that although the topmost mask gave outputs that were generally more suitable for bin picking it also gave a few masks that were on the edge of the image frame and a few that were partials of the object that was selected. To counteract this the depth, area and border penalties were increased in the scoring pipeline.

This was made clearer when the pipeline was tested on real world data collected by our RealSense RGB +D camera. While generally the scoring system would provide logical

masks selections for bin picking, the topmost mask tends to select any mask which can lead to partial mask of the topmost object.

5.2 Interpretation of the Results

While the objective of this study is to explore methods of using a zero-shot segmentation method such as SAM and have an uncomplicated path to using those segmentations in a bin picking use case, the start of process was to study how depth alone can be used for this process. While testing the in-house RGB-D camera to detect the topmost point there were a few outliers which led to the use of median depth in selecting the topmost mask. This increased the stability of our depth scores as it calculated the median depth for each mask instead of going for the mask with the least given single depth measurement. Using depth alone had another major flaw, this was that when SAM analyzed the photo and output possible segmentation masks it gives multiple masks which includes masks of parts of the full objects. In this situation there is a high likelihood that the least depth could be found on one of these incomplete masks which happened often as observed in fig 12, this issue did not fully rectify even with the use of SAMs own confidence scores to filter out the best masks possible. While carrying out more tests it was obvious that there needed to be more factors integrated into the selection process of an object that needs to be selected from a robotic arm in the most efficient way possible.

The first issue other than the depth that popped up during the testing phase was the selection of large flat areas by SAM which was as issue, where walls and floor areas were considered as objects by SAM. To negate this a fixed area penalty was introduced for masks going above a certain threshold of the percentage the total image area. Keeping this value around 15% would usually stop SAM from selecting too large of a mask. This strategy was effective with the testing done on the Siléane bricks dataset as the objects seemed to have an average area and the masks created were clean due to the synthetic nature of the RGB data. The masks created for the real-world datasets differ significantly even with the use of SAM confidence scores. Also, with the initial ablation test addition to the area penalty reduces the segmentation performance of the masks, this was due

to the score being designed in a way that smaller the mask was less of a penalty it gets. This was changed to the mask only receiving a penalty when it went above the set limit. Since the scoring system needs to be used with various object sizes in different poses, the average area of the ground truth of the real-world target object was selected as the benchmark of the area penalty. Therefore, this can be used as simple pretraining process in the scoring pipeline, but if the ground truth masks are not available to create an average target area, a fixed limit would be set. While the area was an important factor in the selection process of a mask that fits an object that needs to be picked, the distance from the object to the borders of the image were considered as at times it could be the segments selected at the borders of the image has the potential of being part of the wall of the box and also the chance of the detected object being close to the wall increases as it gets to the border of the image, this is important as the this might reduce the approachability of the object by a robotic arm. This prompted the addition of the border penalty. Next depth variance was included to make sure that the masks selected did not select multiple objects of varied heights, which tends to happen when objects are occluded and of similar texture. The amount of depth variance, however, can depend on object to object and the preferences of the picking process. The inclusion of the depth variance penalty can hinder the selection of objects that are inclined at an angle, if this hinders the picking process the variance can be adjusted accordingly.

When it comes to the segmentation performance and quality the most important factor in the scoring pipeline was the shape, this made sure that the shape obtained from the masks selected were not too irregular and next was the area that made the performance of the segmentation masks better, but the factors in the mask that point towards the selection of an object that suitable for picking from a bin would be the depth, border distance and depth variance in that order. These are the factors that increase the accessibility of the object to be able to be picked by a robotic arm with ease.

In (Ji et al., 2024) SAMs performance on industrial datasets were not very transferable. Many of the studies such as (Franaszek et al., 2024) have looked into fusing depth into SAM itself there haven't been any significant studies on SAM being combined with a scoring system including depth to be used in a flexible multi object bin picking scenario.

While the method is performing at an acceptable level there are some factors that need to be considered further in making this a viable approach to bin picking. Areas such as dealing with sensor noise specially in depth and missing depth values while partially can be handled by improved hardware, a solution withing the pipeline itself would benefit the performance greatly, an additional factor can also be added to the scoring system if the depth values are accurate enough and that would be a factor considering how separated an object is from rest of the surroundings. This would also help with the bin picking selection. Time would have to be a weakness of the SAM pipeline. For the complete processing of a set of 30 images in the Siléane bricks dataset the average time was 7 mins, which would be a major disadvantage of using SAM in bin picking taking 14 seconds per frame. But these results are from a low powered CPU and GPU nonetheless effort needs to be taken to reduce the processing times for the pipeline in general.

6 Conclusion

This study investigated the use of the Segment Anything Model as a zero-shot segmentation tool for robotic bin picking and developed a scoring pipeline to refine the mask selection in overall industrial bin picking scenarios. While SAM has the ability to produce segmentation masks across diverse datasets, its tendency to produce an overabundance of potential masks, selections of large flat areas as objects and partial and illogical candidate masks at times made it a challenge to use directly in bin picking.

While the scoring pipeline did perform better than directly using SAMs topmost segment to select the object to be picked while avoiding most situations of partial masks. Therefore, while SAMs raw output were not directly suitable for bin picking, the integration of depth and geometric penalties in the scoring system was able to demonstrate that SAM can be adapted into a modular, flexible and effective segmentation pipeline for industrial environments.

In summary, this work contributes to a clear methodology for adapting SAM to bin picking scenarios while depth is still a critical factor a combination of geometric, statistical and confidence-based penalties is necessary to adapt such zero shot pipelines into fast changing industrial environments without retraining. The finding highlights both the potential and limitations of this pipeline while paving the way for future research into efficient, adaptive and real time segmentation in bin picking.

7 Future work

While the scoring pipeline introduced in this thesis can significantly improve the practical use of SAM in bin picking it also leaves several directions open for further exploration and research.

Depth completion and sensor robustness

While the current implementation uses inpainting to deal with missing depth values, a more reliable method should be investigated in the future. Such methods could include methods such as multi view fusion.

Adaptive weighing of scoring terms

The scoring pipeline now mostly depends on fixed weights for each of its elements for each dataset, therefore introducing a reinforcement learning or a supervised training method from ground truth bin picking outcomes would allow the bin picking of varying objects much more robustly.

Object separation factor

Another factor that can improve the performance of the scoring pipeline greatly is the inclusion of a measure that could give how well a specific object is separated from the rest. This would further enhance the prioritization of masks that are more accessible for robotic grasping.

Efficiency and real time deployment

The whole process of SAM inference and scoring takes several seconds per frame on comparatively modest hardware, but efforts need to be made on GPU acceleration,

model distillation or integration with lightweight models such as FastSAM to achieve that real time performance that is necessary for industrial viability.

Integration with robotic grasping.

Finally, a robotic grasping pipeline could be implemented to have a more holistic measure of system performance in real world applications.

References

Aldoma Buchaca, A., Tombari, F., di Stefano, L., & Vincze, M. (n.d.). A Global Hypotheses Verification Method for 3D Object Recognition. *Proc. of the European Conference on Computer Vision 2012 (ECCV2012)*, 14. Retrieved December 2, 2025, from https://repositum.tuwien.at/handle/20.500.12708/72902

Belotti, P., Buchanan, A., & Ezazipour, S. (2025). Political Districting to Optimize the Polsby-Popper Compactness Score with Application to Voting Rights. *Operations Research*, *73*(5), 2330–2350. https://doi.org/10.1287/opre.2024.1078

Berger, K., Meister, S., Nair, R., & Kondermann, D. (2013). A State of the Art Report on Kinect Sensor Setups in Computer Vision. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *8200 LNCS*, 257–272. https://doi.org/10.1007/978-3-642-44964-2_12

Bertalmio, M., Bertozzi, A. L., & Sapiro, G. (n.d.). Navier-stokes, fluid dynamics, and image and video inpainting. *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, *1*. https://doi.org/10.1109/CVPR.2001.990497

Chang, Z., Wu, H., & Li, C. (2023). YOLOv4-tiny-based robust RGB-D SLAM approach with point and surface feature fusion in complex indoor environments. *Journal of Field Robotics*, *40*(3), 521–534. https://doi.org/10.1002/rob.22145

Chen, G., Cui, G., Jin, Z., Wu, F., & Chen, X. (2018). Accurate Intrinsic and Extrinsic Calibration of RGB-D Cameras with GP-based Depth Correction. *IEEE Sensors Journal*, *1*, 1558–1748. https://doi.org/10.1109/JSEN.2018.2889805

Chen, J., Zhou, Z., Li, X., Zheng, Y., Bao, T., & He, Z. (2025). ZeroBP: Learning Position-Aware Correspondence for Zero-Shot 6D Pose Estimation in Bin-Picking. *2025 IEEE International Conference on Robotics and Automation (ICRA)*, 12266–12272. https://doi.org/10.1109/ICRA55743.2025.11128347

Cippitelli, E., Gasparrini, S., Gambi, E., Spinsante, S., Wåhslény, J., Orhany, I., & Lindhy, T. (2015). Time synchronization and data fusion for RGB-Depth cameras and inertial sensors in AAL applications. *2015 IEEE International Conference on Communication Workshop, ICCW 2015*, 265–270. https://doi.org/10.1109/ICCW.2015.7247189

Cordeiro, A., Rocha, L. F., Costa, C., Costa, P., & Silva, M. F. (2022). Bin Picking Approaches Based on Deep Learning Techniques: A State-of-the-Art Survey. *2022 IEEE International Conference on Autonomous Robot Systems and Competitions, ICARSC 2022*, 110–117. https://doi.org/10.1109/ICARSC55462.2022.9784795

Cordeiro, A., Souza, J. P., Costa, C. M., Filipe, V., Rocha, L. F., & Silva, M. F. (2023). Bin Picking for Ship-Building Logistics Using Perception and Grasping Systems. *Robotics*, *12*(1). https://doi.org/10.3390/robotics12010015

Dalal, N., & Triggs, B. (2005). Histograms of Oriented Gradients for Human Detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)* (Vol. 1, pp. 886–893). https://doi.org/10.1109/CVPR.2005.177

Danielczuk, M., Matl, M., Gupta, S., Li, A., Lee, A., Mahler, J., & Goldberg, K. (2019). Segmenting Unknown 3D Objects from Real Depth Images using Mask R-CNN Trained on Synthetic Data. *2019 International Conference on Robotics and Automation (ICRA)*, 7283–7290. https://doi.org/10.1109/ICRA.2019.8793744

Dong, Z., Liu, S., Zhou, T., Cheng, H., Zeng, L., Yu, X., & Liu, H. (2019). PPR-Net:Point-wise Pose Regression Network for Instance Segmentation and 6D Pose Estimation in Bin-picking Scenarios. *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 1773–1780. https://doi.org/10.1109/IROS40897.2019.8967895

Donne, S., & Geiger, A. (2019). Learning Non-Volumetric Depth Fusion Using Successive Reprojections. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 7626–7635. https://doi.org/10.1109/CVPR.2019.00782

Drost, B., Ulrich, M., Navab, N., & Ilic, S. (2010). Model globally, match locally: Efficient and robust 3D object recognition. *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 998–1005. https://doi.org/10.1109/CVPR.2010.5540108

Eghbal-zadeh, H., Zellinger, W., Pintor, M., Grosse, K., Koutini, K., Moser, B. A., Biggio, B., & Widmer, G. (2024). Rethinking data augmentation for adversarial robustness. *Information Sciences*, *654*. https://doi.org/10.1016/j.ins.2023.119838

Franaszek, M., Rachakonda, P., Piliptchak, P., & Saidi, K. S. (2024). Merging Outcomes of SAM Applied to RGB and Depth Images in Bin Picking Applications. *6th International Conference on Advances in Signal Processing and Artificial Intelligence*, 17–19.

Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2013). Rich feature hierarchies for accurate object detection and semantic segmentation. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 580–587. https://doi.org/10.1109/CVPR.2014.81

Grard, M. (2019). *Generic instance segmentation for object-oriented bin-picking*. https://theses.hal.science/tel-03081227v1

He, K., Gkioxari, G., Dollar, P., & Girshick, R. (2017). Mask R-CNN. *2017 IEEE International Conference on Computer Vision (ICCV)*, 2980–2988. https://doi.org/10.1109/ICCV.2017.322

Hernández-García, A., & König, P. (2020). *Data augmentation instead of explicit regularization*. http://arxiv.org/abs/1806.03852

Imagen 2. (2025). Industrial bin picking image created for the prompt " An image an robotic industrial bin picking scenario, where a robotic arm is picking an object from a bin with similar objects." In *Google*.

Intel Corporation. (2019). *Intel ® RealSenseTM Depth Module D400 Series Custom Calibration*. http://www.intel.com/design/literature.htm.

Iriondo, A., Lazkano, E., & Ansuategi, A. (2021). Affordance-based grasping point detection using graph convolutional networks for industrial bin-picking

applications. *Sensors (Switzerland)*, *21*(3), 1–26. https://doi.org/10.3390/s21030816

Janis Arents, Cacurs, R., & Greitans, M. (2018). Integration of Computervision and Artificial Intelligence Subsystems with Robot Operating System Based Motion Planning for Industrial Robots. *Automatic Control and Computer Sciences*, *52*(5), 392–401. https://doi.org/10.3103/S0146411618050024

Ji, W., Li, J., Bi, Q., Liu, T., Li, W., & Cheng, L. (2024). Segment Anything Is Not Always Perfect: An Investigation of SAM on Different Real-world Applications. In *Machine Intelligence Research* (Vol. 21, Issue 4, pp. 617–630). Chinese Academy of Sciences. https://doi.org/10.1007/s11633-023-1385-0

Kazhdan, M., Bolitho, M., & Hoppe, H. (2006). Poisson surface reconstruction | Proceedings of the fourth Eurographics symposium on Geometry processing. *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*, 61–70. https://doi.org/10.5555/1281957.1281965

Keselman, L., Woodfill, J. I., Grunnet-Jepsen, A., & Bhowmik, A. (2017). *Intel RealSense Stereoscopic Depth Cameras*. http://arxiv.org/abs/1705.05548

Khoshelham, K., & Elberink, S. O. (2012). Accuracy and resolution of kinect depth data for indoor mapping applications. *Sensors*, *12*(2), 1437–1454. https://doi.org/10.3390/s120201437

Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A. C., Lo, W.-Y., Dollár, P., & Girshick, R. (2023). Segment Anything. *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, 3992–4003. https://doi.org/10.1109/ICCV51070.2023.00371

Kumra, S., Joshi, S., & Sahin, F. (2020). Antipodal Robotic Grasping using Generative Residual Convolutional Neural Network. *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 9626–9633. https://doi.org/10.1109/IROS45743.2020.9340777

Lachat, E., Macher, H., Landes, T., & Grussenmeyer, P. (2015). Assessment and Calibration of a RGB-D Camera (Kinect v2 Sensor) Towards a Potential Use for

Close-Range 3D Modeling. *Remote Sensing 2015, Vol. 7, Pages 13070-13097*, *7*(10), 13070–13097. https://doi.org/10.3390/RS71013070

Laskey, M., Lee, J., Chuck, C., Gealy, D., Hsieh, W., Pokorny, F. T., Dragan, A. D., & Goldberg, K. (2016). Robot grasping in clutter: Using a hierarchy of supervisors for learning from demonstrations. *2016 IEEE International Conference on Automation Science and Engineering (CASE)*, 827–834. https://doi.org/10.1109/COASE.2016.7743488

Le, T. T., & Lin, C. Y. (2019). Bin-picking for planar objects based on a deep learning network: A case study of usb packs. *Sensors (Switzerland)*, *19*(16). https://doi.org/10.3390/s19163602

Li, X., Qiao, Z., Wan, G., Zhu, S., Zhao, Z., Fan, X., Shi, P., & Wan, J. (2024). Depth-Guided Bilateral Grid Feature Fusion Network for Dehazing. *Sensors 2024, Vol. 24, Page 3589*, *24*(11), 3589. https://doi.org/10.3390/S24113589

Liu, D., Arai, S., Miao, J., Kinugawa, J., Wang, Z., & Kosuge, K. (2018). Point pair feature-based pose estimation with multiple edge appearance models (PPF-MEAM) for robotic bin picking. *Sensors (Switzerland)*, *18*(8). https://doi.org/10.3390/s18082719

Liu, D., Arai, S., Xu, Y., Tokuda, F., & Kosuge, K. (2021). 6D Pose Estimation of Occlusion-Free Objects for Robotic Bin-Picking Using PPF-MEAM with 2D Images (Occlusion-Free PPF-MEAM). *IEEE Access*, *9*, 50857–50871. https://doi.org/10.1109/ACCESS.2021.3068467

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, *60*(2), 91–110. https://doi.org/10.1023/B:VISI.0000029664.99615.94/METRICS

Mahler, J., Liang, J., Niyaz, S., Laskey, M., Doan, R., Liu, X., Ojea, J. A., & Goldberg, K. (2017). Dex-Net 2.0: Deep Learning to Plan Robust Grasps with Synthetic Point Clouds and Analytic Grasp Metrics. *ArXiv*. http://arxiv.org/abs/1703.09312

Maturana, D., & Scherer, S. (2015). VoxNet: A 3D Convolutional Neural Network for real-time object recognition. *2015 IEEE/RSJ International Conference on*

Intelligent Robots and Systems (IROS), 922–928. https://doi.org/10.1109/IROS.2015.7353481

McGlade, J., Wallace, L., Reinke, K., & Jones, S. (2022). The Potential of Low-Cost 3D Imaging Technologies for Forestry Applications: Setting a Research Agenda for Low-Cost Remote Sensing Inventory Tasks. *Forests*, *13*(2), 204. https://doi.org/10.3390/f13020204

Miller, A. T., & Allen, P. K. (2004). Graspit: A versatile simulator for robotic grasping. *IEEE Robotics and Automation Magazine*, *11*(4), 110–122. https://doi.org/10.1109/MRA.2004.1371616

Nielsen, F., & Nock, R. (2013). Consensus Region Merging for Image Segmentation. *2013 2nd IAPR Asian Conference on Pattern Recognition*, 325–329. https://doi.org/10.1109/ACPR.2013.142

Nishi, T., Kawasaki, S., Iewaki, K., Okura, F., Petit, D., Takano, Y., & Harada, K. (2023). M3R-CNN: on effective multi-modal fusion of RGB and depth cues for instance segmentation in bin-picking. *Advanced Robotics*, *37*(18), 1143–1157. https://doi.org/10.1080/01691864.2023.2257266

O'sullivan, P., & Le Dortz, N. (2021). Time of Flight System Design-Part 1: System Overview. In *VISIT ANALOG.COM* (Vol. 55, Issue 3).

Rabbani, T., Van Den Heuvel, F. A., & Vosselman, G. (2006). SEGMENTATION OF POINT CLOUDS USING SMOOTHNESS CONSTRAINT. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, *36*.

Rebuffi, S.-A., Gowal, S., Calian, D. A., Stimberg, F., Wiles, O., & Mann, T. (2021, November 9). Data Augmentation Can Improve Robustness. *35th Conference on Neural Information Processing Systems*. https://doi.org/10.48550/arXiv.2111.05328

RF Wireless World (no individual author listed). (n.d.). *Stereo Vision vs. Structured Light vs. Time of Flight (ToF) | RF Wireless World*. RF Wireless World. Retrieved December 2, 2025, from https://www.rfwireless-world.com/terminology/stereo-vision-vs-structured-light-vs-time-of-flight.html

Ribeiro, E. G., de Queiroz Mendes, R., & Grassi, V. (2021). Real-time deep learning approach to visual servo control and grasp detection for autonomous robotic manipulation. *Robotics and Autonomous Systems*, *139*, 103757. https://doi.org/10.1016/j.robot.2021.103757

Rusinkiewicz, S., & Levoy, M. (2001). Efficient variants of the ICP algorithm. *Proceedings of International Conference on 3-D Digital Imaging and Modeling, 3DIM*, 145–152. https://doi.org/10.1109/IM.2001.924423

Schellhase, F. (2023). *Achieving Accuracy in Bin Picking with TI DLP® Technology-powered Structured Light Systems*. www.ti.com

Schwarz, M., Milan, A., Periyasamy, A. S., & Behnke, S. (2018). RGB-D object detection and semantic segmentation for autonomous manipulation in clutter. *International Journal of Robotics Research*, *37*(4–5), 437–451. https://doi.org/10.1177/0278364917713117

Shaikh, M. B., & Chai, D. (2021). Rgb-d data-based action recognition: A review. In *Sensors* (Vol. 21, Issue 12). MDPI AG. https://doi.org/10.3390/s21124246

Shao, Q., Hu, J., Wang, W., Fang, Y., Liu, W., Qi, J., Ma, J., & Jiao, S. (2023). Suction Grasp Region Prediction using Self-supervised Learning for Object Picking in Dense Clutter. In *2019 IEEE 5th International Conference on Mechatronics System and Robots (ICMSR)* (pp. 7–12). IEEE. https://doi.org/10.1109/ICMSR.2019.8835468

Shvets, M., Zhao, D., Niethammer, M., Sengupta, R., & Berg, A. C. (2024). Joint Depth Prediction and Semantic Segmentation with Multi-View SAM. *2024 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 1317–1327. https://doi.org/10.1109/WACV57701.2024.00136

Siltala, N., & Latokartano, J. (2023). *Comparative Analysis of 3D-Depth Cameras in Industrial Bin Picking Solution*. Tampereen yliopisto.

Thanh Ly, P., Chi Nguyen, Q., Duy Hung Nguyen, N., Pham, P. T., & Hong, K. S. (2022). Structured-Light-Based 3D Scanning System for Industrial Manipulator in Bin Picking Application. *2022 Australian and New Zealand Control Conference, ANZCC 2022*, 34–39. https://doi.org/10.1109/ANZCC56036.2022.9966979

Tomasi, C., & Manduchi, R. (1998). Bilateral filtering for gray and color images. *Sixth International Conference on Computer Vision (IEEE Cat. No.98CH36271)*, 839–846. https://doi.org/10.1109/ICCV.1998.710815

Uijlings, J. R. R., Van De Sande, K. E. A., Gevers, T., & Smeulders, A. W. M. (2013). Selective Search for Object Recognition. *International Journal of Computer Vision 2013 104:2*, *104*(2), 154–171. https://doi.org/10.1007/S11263-013-0620-5

Villena-Martínez, V., Fuster-Guilló, A., Azorín-López, J., Saval-Calvo, M., Mora-Pascual, J., Garcia-Rodriguez, J., & Garcia-Garcia, A. (2017). A Quantitative Comparison of Calibration Methods for RGB-D Sensors Using Different Technologies. *Sensors 2017, Vol. 17, Page 243*, *17*(2), 243. https://doi.org/10.3390/S17020243

Wang, C., Xu, D., Zhu, Y., Martin-Martin, R., Lu, C., Fei-Fei, L., & Savarese, S. (2019). DenseFusion: 6D Object Pose Estimation by Iterative Dense Fusion. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, *2019-June*, 3338–3347. https://doi.org/10.1109/CVPR.2019.00346

Xiang, Y., Schmidt, T., Narayanan, V., & Fox, D. (2018, June 26). PoseCNN: A Convolutional Neural Network for 6D Object Pose Estimation in Cluttered Scenes. *Robotics: Science and Systems XIV*. https://doi.org/10.15607/RSS.2018.XIV.019

Xie, Z., Liang, X., & Roberto, C. (2023). Learning-based robotic grasping: A review. In *Frontiers in Robotics and AI* (Vol. 10). Frontiers Media S.A. https://doi.org/10.3389/frobt.2023.1038658

Xu, Y., Arai, S., Liu, D., Lin, F., & Kosuge, K. (2022). FPCC: Fast point cloud clustering-based instance segmentation for industrial bin-picking. *Neurocomputing*, *494*, 255–268. https://doi.org/10.1016/j.neucom.2022.04.023

Xu, Y., Arai, S., Tokuda, F., & Kosuge, K. (2020). A Convolutional Neural Network for Point Cloud Instance Segmentation in Cluttered Scene Trained by Synthetic

Data without Color. *IEEE Access*, *8*, 70262–70269. https://doi.org/10.1109/ACCESS.2020.2978506

Yang, Q., Yang, R., Davis, J., & Nistér, D. (2007). Spatial-depth super resolution for range images. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. https://doi.org/10.1109/CVPR.2007.383211

Yi, L., Zhao, W., Wang, H., Sung, M., & Guibas, L. J. (2019). GSPN: Generative Shape Proposal Network for 3D Instance Segmentation in Point Cloud. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 3942–3951. https://doi.org/10.1109/CVPR.2019.00407

Yu, Z., Zhang, X., Li, Z., Bin, Y., & Xiao, G. (2024). Exploring Deeper! Segment Anything Model with Depth Perception for Camouflaged Object Detection. In *CEUR Workshop Proceedings* (Vol. 2657). CEUR-WS. https://doi.org/10.1145/nnnnnnn.nnnnnnn

Zanuttigh, P., Mutto, C. D., Minto, L., Marin, G., Dominio, F., & Cortelazzo, G. M. (2016). Time-of-flight and structured light depth cameras: Technology and applications. *Time-of-Flight and Structured Light Depth Cameras: Technology and Applications*, 1–355. https://doi.org/10.1007/978-3-319-30973-6/COVER

Zeng, A., Song, S., Yu, K.-T., Donlon, E., Hogan, F. R., Bauza, M., Ma, D., Taylor, O., Liu, M., Romo, E., Fazeli, N., Alet, F., Dafle, N. C., Holladay, R., Morena, I., Qu Nair, P., Green, D., Taylor, I., Liu, W., … Rodriguez, A. (2018). Robotic Pick-and-Place of Novel Objects in Clutter with Multi-Affordance Grasping and Cross-Domain Image Matching. *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 3750–3757. https://doi.org/10.1109/ICRA.2018.8461044

Zhang, C., Huang, T., & Zhao, Q. (2019). A New Model of RGB-D Camera Calibration Based on 3D Control Field. *Sensors 2019, Vol. 19, Page 5082*, *19*(23), 5082. https://doi.org/10.3390/S19235082

Zhang, Z. (2000). A flexible new technique for camera calibration. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *22*(11), 1330–1334. https://doi.org/10.1109/34.888718

Zhao, X., Cui, X., & Fu, H. (2024). 6DOF Pose Estimation of a 3D Rigid Object Based on an Adaptive Model Curvature Point Pair Features. *2024 10th International Conference on Control, Automation and Robotics (ICCAR)*, 53–58. https://doi.org/10.1109/ICCAR61844.2024.10569898

Zhao, X., Ding, W., An, Y., Du, Y., Yu, T., Li, M., Tang, M., & Wang, J. (2023). Fast Segment Anything. *ArXiv*. http://arxiv.org/abs/2306.12156

Zhou, H. (2024). Dual-Stream Bilateral Feature Fusion Network for RGB-D Salient Object Detection. *2024 3rd International Conference on Cloud Computing, Big Data Application and Software Engineering, CBASE 2024*, 787–790. https://doi.org/10.1109/CBASE64041.2024.10824343

Zhuang, C., Wang, Z., Zhao, H., & Ding, H. (2021). Semantic part segmentation method based 3D object pose estimation with RGB-D images for bin-picking. *Robotics and Computer-Integrated Manufacturing*, *68*. https://doi.org/10.1016/j.rcim.2020.102086

Zollhöfer, M. (2019). Commodity RGB-D Sensors: Data Acquisition. In *RGB-D Image Analysis and Processing. Advances in Computer Vision and Pattern Recognition* (pp. 3–13). https://doi.org/10.1007/978-3-030-28603-3_1

Zou, Y., Eldemiry, A., Li, Y., & Chen, W. (2020). Robust RGB-D slam using point and line features for low textured scene. *Sensors (Switzerland)*, *20*(17), 1–20. https://doi.org/10.3390/s20174984