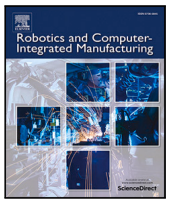
Robotics and Computer–Integrated Manufacturing 100 (2026) 103244

Contents lists available at ScienceDirect

Robotics and Computer-Integrated Manufacturing

journal homepage: www.elsevier.com/locate/rcim



Full length article

VL-GRiP3: A hierarchical pipeline leveraging vision-language models for autonomous robotic 3D grasping☆

Mirco Polonara [a,b,1], Xingyu Yang [b,1], Luca Carbonari [a,*], Xuping Zhang [b,*]

[a] DIISM, Polytechnic University of Marche, Via Brecce Bianche 12, Ancona, 62100, Italy
[b] Department of Mechanical and Production Engineering, Aarhus University, Katrinebjergvej 89F, Aarhus, 8200, Denmark

ARTICLE INFO

Dataset link: https://github.com/AU-DK-Robotics/VL-GRiP3

Keywords:
Autonomous grasping
Vision-language model
Point cloud augmentation
3D grasping pose
Robotic action

ABSTRACT

Autonomous grasping has long been a central topic in robotics, yet deployment in small and medium-sized enterprises (SMEs) is still hindered by low-level robot programming and the lack of natural language interaction. Recent Vision-Language-Action models (VLAs) allow robots to interpret natural language commands for intuitive interaction and control, but they still exhibit output uncertainty and are not yet well suited to directly generating reliable, precise actions in safety-critical industrial contexts. To address this gap, we present VL-GRiP3, a hierarchical Vision-Language model (VLM)-enabled pipeline for autonomous 3D robotic grasping that bridges natural language interaction and accurate, reliable manipulation in SME settings. The framework decomposes language understanding, perception, and action planning in a transparent modular architecture, improving flexibility and interpretability. Within this architecture, a single VLM backbone handles natural language interpretation, target perception, and high-level action planning. CAD-augmented point cloud registration then mitigates occlusions in single RGB-D views while keeping hardware cost low, and an M2T2-based grasp planner predicts accurate 3D grasp poses that explicitly account for complex object geometry from the augmented point cloud, enabling reliable manipulation of irregular industrial parts. Experiments show that our fine-tuned VLM modules achieve segmentation performance comparable to YOLOv8n, and VL-GRiP3 attains a 94.67% success rate over 150 randomized grasping trials. A comparative evaluation against state-of-the-art end-to-end VLAs further indicates that our modular, CAD-augmented design with explicit 3D grasp pose prediction yields more reliable and controllable behavior for SME manufacturing applications.

1. Introduction

Robotic grasping is fundamental to industrial automation, yet robust deployment in high-mix, semi-structured settings remains challenging. Random bin picking, where objects are heaped in arbitrary orientations, is still difficult because the robot must detect, localize, and grasp partially occluded, often unknown items [1–3]. Vision-guided manipulation has progressed from 2D cameras to stereo and depth sensors with modular pipelines for detection, pose estimation, and grasp planning [4], but these pipelines are brittle in heavy clutter, under variable lighting, or with reflective parts, and typically assume prior object knowledge [5,6]. Further learning-based vision methods [7,8] alleviate some limitations; CNN grasp predictors [9,10] and sim-to-real reinforcement learning [11] broaden object coverage but still demand large datasets and careful tuning. Contemporary grasping pipelines also require expert calibration of cameras and robots [12], per-object retraining, and manual tuning of motion primitives [9,13]; most terminate after predicting a grasp pose, leaving the synthesis of executable action scripts to human engineers and inflating cost and changeover time in high-mix or small-batch manufacturing [14]. Even data-driven approaches often depend on task-specific datasets that are costly to acquire [13,15]. Most importantly, a direct bridge between humans and robots is still missing: people express goals in natural language, whereas robots require structured command sequences, forcing experts to translate every new instruction into low-level code. These limitations, especially the language-action gap, highlight the need for a grasping pipeline that robustly handles novel, partially occluded parts, can be re-tasked in high-mix settings without expert re-engineering, and maps natural language instructions directly to executable actions.

With the rise of vision–language models (VLMs), which accept both natural language intent and visual context as input [16,17] and reason about responses through a unified, prompt-driven interface [18,19],

☆ This article is part of a Special issue entitled: 'AGI4RoboticsManufacturing' published in Robotics and Computer-Integrated Manufacturing.

* Corresponding authors.
E-mail addresses: l.carbonari@staff.univpm.it (L. Carbonari), xuzh@mpe.au.dk (X. Zhang).

[1] These authors contributed equally to this work.

https://doi.org/10.1016/j.rcim.2026.103244
Received 30 June 2025; Received in revised form 3 December 2025; Accepted 18 January 2026
Available online 27 January 2026

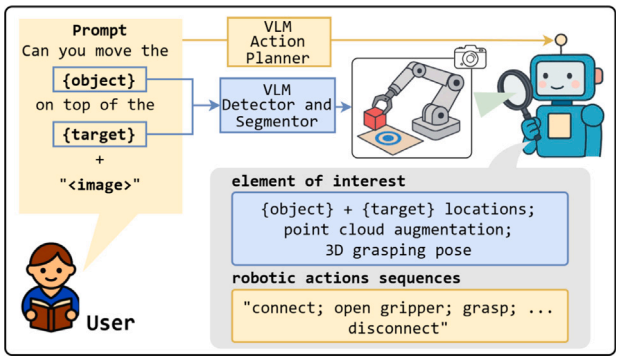


Fig. 1. Overview of VL-GRiP3: the user provides a natural language prompt and an image; VLM modules hierarchically extract relevant object and target regions and plan high-level actions, while point cloud augmentation and 3D grasp pose prediction enable reliable manipulation.

these models provide an attractive approach for enabling robots to understand human natural language requests while jointly processing visual information. Building on this, recent work has begun to extend VLMs to vision–language–action models (VLAs) [20], which close the loop from language and perception to robot actions. State-of-the-art VLAs typically follow two major architectural paradigms: end-to-end models that directly integrate an action layer into the VLM, and framework-based approaches that embed the VLM into a modular pipeline, both offering a new paradigm for language-conditioned control in autonomous robotic manipulation [20,21]. However, end-to-end VLAs are generative, black-box models that inherently introduce uncertainty [22], are difficult to debug, and are mainly trained by imitating demonstration data [23], which requires large quantities of high-quality, unbiased expert demonstrations; consequently, they are difficult to deploy in manufacturing environments due to safety concerns arising from physical interaction [24]. In contrast, framework-based approaches are more applicable in SME environments, as they decouple language understanding, perception, and action planning into modular steps, enhancing system flexibility and transparency while reducing data requirements. Yet most of these frameworks still reduce grasping to 2D picking points or simple grasping strategies [25,26], even though industrial parts are typically irregular and require geometry-aware 3D grasping [21] in SME production. Furthermore, many such systems use different models for perception and planning and rely on multi-camera or multi-view setups for object reconstruction, which increases system complexity and integration effort.

Building on this momentum, we propose VL-GRiP3, a hierarchical VLM-enabled pipeline with geometry-aware 3D grasping that closes the loop from natural language intent to reliable execution in semi-structured environments. An overview of the workflow is shown in Fig. 1, where the perception-to-action core is implemented with a single VLM backbone serving three roles: a VLM segmentor extracts the object identity, a VLM detector determines the target location, and a VLM action planner generates a feasible action sequence. Moreover, point cloud registration with CAD models of the objects is leveraged to resolve occlusions from a single RGB-D view, enabling fast, low-overhead deployment without multi-camera/view setups while maintaining perception accuracy, which aligns with SME deployment scenarios where hardware and integration resources are limited but engineered parts are typically accompanied by CAD models. In addition, a 3D grasp pose planner is integrated into this modular pipeline, enabling complex grasping with explicit consideration of object geometry. The pipeline then merges all outputs to translate the user command into executable robot scripts for task completion. Our main contributions are:

(1) **Hierarchical VLM-enabled pipeline.** We introduce VL-GRiP3, a hierarchical VLM-enabled pipeline that closes the loop between language understanding, extended perception, and action planning, tightly linking natural language intent to feasible, geometry-aware robotic grasping in semi-structured environments.
(2) **Point cloud augmentation with 3D grasping pose planning.** The segmented partial point cloud is aligned with a CAD model via 3D registration, recovering missing object features. Combined with an M2T2-based grasp pose predictor, the completed object representation enables accurate grasp planning in dense, unstructured environments while reducing the need for multi-camera or multi-view setups.
(3) **SME-oriented deployment.** VL-GRiP3 is deployed on a UR3e robotic cell equipped with a single RGB-D camera and tested on irregular industrial objects from SMEs. Experiments demonstrate 94.67% grasp-and-place success over 150 randomized trials, and comparisons with end-to-end baselines further show that the proposed CAD-augmented, geometry-aware grasping improves grasp robustness to uncertainty while reducing integration overhead, supporting its suitability for SME deployment.

The rest of the paper is organized as follows: Section 2 presents the state-of-the-art related to this work; Section 3 illustrates the pipeline we proposed in detail; Section 4 shows the setup of the experiments and collection of data for training; Section 5 presents the evaluations of key function modules and also overall performance of the pipeline; and Section 6 summarizes the work with an outlook for future.

2. Relevant works

2.1. Autonomous robotic grasping

Traditional autonomous grasping has evolved through several research phases but remains highly dependent on specific object types and demands considerable engineering effort. It typically follows a pipeline of sensing, localization, approach, and grasp execution. In semi-structured or cluttered environments, camera systems play a crucial role in perceiving the robot's workspace, and both intrinsic and extrinsic sensor calibrations are essential to ensure accurate object localization [12,27]. Early analytic planners searched for force or wrench-closure contacts on precise CAD meshes, guaranteeing stability but assuming idealized models and precise calibration [28]. The advent of stereo vision and depth cameras shifted the field toward modular perception pipelines that chain object detection, 6-DoF pose estimation, grasp sampling, and grasp scoring, each stage requiring retuning whenever the sensor, gripper, or target object changes [6,29]. Supervised learning reduced hand-designed features but not the data burden: depth-image classifiers such as GQ-CNN [9] and pixel-wise GG-CNN [30], along with full 6-DoF detectors like GraspNet [31, 32], predict grasp success directly from sensor data yet still require large synthetic or real datasets and lose accuracy on novel shapes. Reinforcement-learning policies such as QT-Opt, trained on over 580 k real-world trials, discover closed-loop corrective behaviors but at the cost of massive data collection and infrastructure, making them difficult to replicate [11]. In summary, conventional grasping pipelines remain tightly coupled to part geometries, sensor calibrations, and workspace layouts; even minor changes to the object catalog or task configuration typically require extensive expert re-engineering. Even though modern robotic grasping systems are becoming increasingly sophisticated and capable, they still rely primarily on machine-level instructions. As a result, human commands must be translated into executable programs by experts [7]. A new paradigm, where the grasping pipeline can understand natural language, offers significant potential for improving human–robot collaboration and simplifying deployment.

2.2. Visual-Language Model

Visual Language Models (VLMs) are multimodal deep learning architectures designed to integrate visual and textual data, enabling tasks that involve the simultaneous interpretation of images or videos and natural language descriptions. These models often combine a vision backbone for feature extraction with a language model based on the Transformer architecture [33], which has been widely adopted in natural language processing through models such as BERT [34]. By pretraining on large-scale image–text pairs, VLMs learn aligned representations that capture semantic correspondences between visual content and linguistic expressions [35]. This shared vision-language understanding underlies various applications: image captioning, which involves generating coherent textual descriptions of an image [36]; visual question answering (VQA), where a system provides answers grounded in image content [37]; and cross-modal retrieval, which retrieves relevant images based on textual queries or vice versa [38]. Some VLMs also demonstrate few-shot or zero-shot learning capabilities, adapting efficiently to new tasks or domains with minimal labeled data [35]. Although challenges persist such as data scarcity, domain adaptation, and interpretability. Ongoing innovations in model design and training strategies continue to broaden the capabilities of VLMs in diverse application settings.

2.3. Vision-language model for object perception and robotic manipulation

One fundamental task for VLMs is visual question answering. Building on this, they have been extended to applications such as object detection and segmentation, where bounding boxes and masks are returned as textual output based on a user prompt and visual input [19,39]. In perception, they are used to localize objects and answer attribute-level queries (e.g., material, fragility) directly from single RGB views, giving planners access to high-level semantics alongside spatial coordinates [40]. VLM outputs can then be refined through segmentation and tracking modules to obtain accurate 3-D object representations that anchor subsequent reasoning in metric space [41]. Because VLMs provide visual reasoning through natural language, they have become central to embodied-AI pipelines, where they are integrated into physical robots to enhance intuitive human–robot collaboration and increase autonomy. These perceptual and reasoning capabilities are increasingly fused with policy generation to form vision-language-action models (VLAs). In such models, a language component consumes detections or attribute descriptions from a VLM and emits either executable robot code or motion objectives, allowing the system to assemble manipulation behaviors such as grasp-and-place sequences or trajectory plans [42,43]. Broadly, current approaches for connecting VLMs to robotic actions fall into two paradigms: end-to-end VLAs and framework-based architectures [20]. End-to-end VLAs integrate a low-level action layer, typically a diffusion-based policy as demonstrated by CogACT [44] and π_0 [45], which directly converts VLM-conditioned outputs into robot motions in action space. In contrast, framework-based approaches decouple high-level perception and task planning from low-level robotic control: the VLM primarily acts as an action planner, as in Instruct2Act [25] and Dexbotic [46], while conventional controllers execute the resulting commands, thereby closing the perception-action loop within a unified, language-conditioned framework.

However, bottlenecks still exist. For end-to-end VLAs, although they provide a unified paradigm for robotic control, they essentially function as black boxes and are primarily trained by imitation learning from demonstration data [23]. As a result, their outputs are often hard to trust, since they do not explain the rationale behind actions and offer limited mechanisms for failure detection or recovery [47], which is unacceptable in industrial settings. Current research hot spots, such as slow–fast multi-timescale reasoning and safety-aware self-correction [47,48], partially address these issues but significantly increase system complexity. Moreover, their generalization to unseen scenarios depends on large amounts of demonstration data; for instance, the RT series [49–51] are fine-tuned with nearly 130k demonstrations, which creates substantial barriers in terms of data collection, data screening, multi-camera setup, and computational cost that are unaffordable for SMEs.

In contrast, framework-based VLAs are more interpretable and controllable because they decompose language understanding, object perception, and action planning into modular steps, allowing individual parts of the pipeline to be debugged and adapted independently [20,46]. In manufacturing environments with a limited set of objects, pretrained VLMs in such frameworks can be further fine-tuned for perception and task planning, making it easier to retarget the system to new parts and specialized behaviors. However, most existing VLAs leave grasp execution to simple grasping strategies [25,26], whereas realistic manufacturing scenarios often involve complex object geometries that require geometry-aware 3D grasping [21], which typically relies on multi-camera or multi-view setups for accurate reconstruction and demands substantial engineering and hardware overhead. Consequently, developing and deploying framework-based VLAs that support precise, geometry-aware 3D grasping in specific manufacturing environments still requires considerable engineering effort.

3. VL-GRiP3: a VLM-based 3D grasping pipeline

We therefore present VL-GRiP3, a modular pipeline that enables a robot to follow free-form language instructions such as “move the red cup into the blue target.” It combines a compact vision-language model, PaliGemma [39], with a point cloud registration module, Predator [52] and grasp planner M2T2 [53]. A single compact VLM serves as perception-to-action core, unifying scene understanding, pixel-accurate segmentation, and the generation of a high-level action script, which is executed through a basic library of low-level robot actions.

3.1. Architecture of the pipeline

Fig. 2 illustrates the data flow through the main modules. User commands arrive either as keyboard input or as speech, which an integrated automatic-speech-recognition (ASR) module based on OpenAI Whisper [54] converts to text. The text command then moves through three stages: (i) identifying the target region, (ii) segmenting the object to grasp and manipulate, and (iii) generating the action plan. To maximize accuracy, we employ a VLM architecture operating in three specialized roles: a detector, fine-tuned for target detection; a segmentor, fine-tuned for object segmentation; and an action planner, fine-tuned for generating action plans from prompts. During inference, the same backbone is invoked first in its detector and segmentor roles for perception, and then in its action role to produce the structured task sequence.

The selected model returns location tokens, segmentation masks, and action sequences in text form. From these tokens we recover 2D object masks, extract depth to estimate 3D poses, and construct a partial point cloud. To overcome occlusions, the point cloud is aligned with a CAD model. This completed shape is passed to M2T2, which predicts suitable grasp and placement poses. Finally, the chosen action script and poses are combined and executed.

3.2. VLM for object segmentation and action planning

PaliGemma processes an RGB image and a textual prompt as a single autoregressive token sequence [19]. A SigLIP-So400 M vision encoder yields 256 visual tokens that are linearly projected into the embedding space of a 2b-parameter Gemma [55] decoder-only Transformer. Under prefix-LM masking, the decoder attends bidirectionally to the prompt and visual tokens and then auto-regressively produces the response. Its output include:

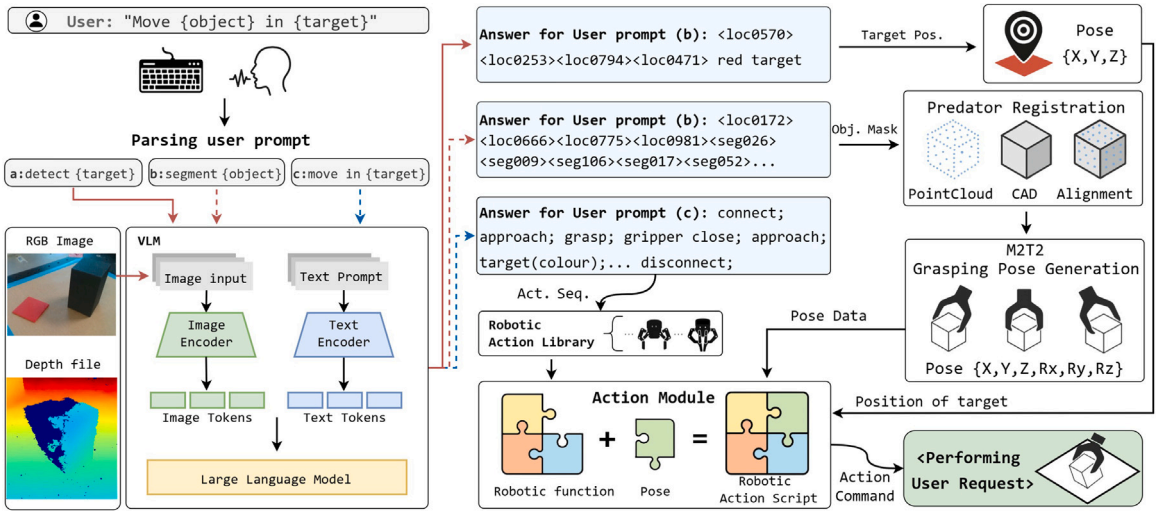


Fig. 2. VL-GRiP3 architecture: natural language command is typed or transcribed and parsed into three prompts for the VLM. The segmented object mask is used to extract the partial point cloud, which is then augmented with the CAD model via registration. 3D grasp poses are predicted from the augmented point cloud, and the target location and grasp pose are finally assembled into planned robot actions for precise object manipulation.

- Location tokens: ($\langle loc0000\rangle \ldots \langle loc1023\rangle$), four consecutive tokens encode the normalized bounding-box coordinates (x_{min},y_{min},x_{max},y_{max}).
- Segmentation tokens:($\langle seg000\rangle \ldots \langle seg127\rangle$), a 128 entry vector-quantized code book representing binary masks; typical objects are described by 20–40 tokens at 224px.
- Robotic action sequences: an ordered list of manipulation primitives (e.g., connect; approach; grasp) that the decoder generates because the model has been explicitly trained to output executable robot action sequences based on the format prompt.

The four location tokens are converted to pixel coordinates by linearly scaling their integer values (0-1023) to the image width or height. The segmentation tokens are decoded by a lightweight VQ-VAE to reconstruct a full resolution binary mask. Combining this mask with the depth image yields a partial three dimensional point cloud of the object; whereas the centroid of the destination bounding box, augmented with its median depth, defines the target pose in the camera frame. After converting both location and segmentation tokens to their corresponding pixel representations, the bounding-box outline and the binary mask in green are overlaid on the RGB image to verify correct target localization and object selection as illustrated in Fig. 3a.

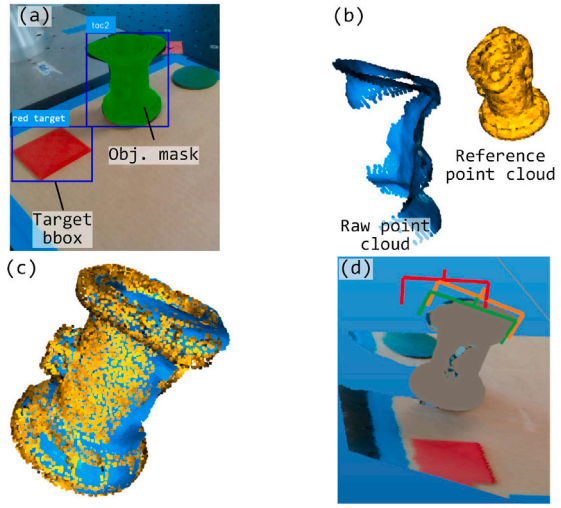


Fig. 3. Perception, augmentation, and grasping pose prediction: (a) Detected target bounding box and segmented object mask, (b) Raw and reference point cloud, (c) Augmented object point cloud, and (d) Predicted grasping poses with augmented point cloud.

3.3. Point cloud augmentation and grasping pose generation

A single RGB-D view invariably suffers from self-occlusion, leaving the rear surfaces of the object unobserved. To obtain a complete representation, the live point cloud is fused with the corresponding CAD model by means of OverlapPredator [52], which delivers both the putative correspondences and a closed-form estimate of the rigid pose in a single pass. This assumes CAD availability for target parts, which is common in SMEs producing engineered components, and allows us to avoid multi-camera calibration or heavy lighting control while still recovering complete geometry from a single view.

After the forward pass each point x_i is endowed with a 32-D descriptor f_i and two confidence scores: an *overlap probability* $o_i \in [0,1]$ and a *matchability probability* $m_i \in [0,1]$. Their product (1)

$$w_i = o_i\, m_i \tag{1}$$

quantifies how likely the point belongs to the truly overlapping region and can be matched reliably. The K points with the highest w_i in each cloud are retained, mutual nearest-neighbor matches are established in descriptor space, and every correspondence (p_k, q_k) is assigned the weight $w_k = w_{p_k} w_{q_k}$. The rigid pose is recovered in one shot by solving the weighted Umeyama/Kabsch problem (2)

$$\min_{R\in SO(3),\, t\in\mathbb{R}^3} \sum_{k=1}^{K} w_k \left\| R\, p_k + t - q_k \right\|^2, \tag{2}$$

whose closed-form solution is obtained with a single weighted SVD of the covariance matrix. Finally, the optimized transform (R,t) is applied to the CAD cloud and the result is merged with the live scene points, producing a dense point set that reveals the object's hidden surfaces (Fig. 3c).

A PointNet++ encoder produces multi-scale features that a masked-transformer *contact decoder* turns into per-point contact masks; a subsequent *action-decoder* MLP then regresses the gripper parameters (approach direction, contact direction, finger width) required to form 6-DoF grasp or placement poses. For our experiments, the public M2T2 implementation was adapted to ingest the unified, object-specific point clouds produced by the segmentation and registration stage, thereby

bypassing any additional scene reconstruction inside the model. Each predicted grasp token q_i yields (3) an objectness probability:

$$o_i = \sigma\big(\mathrm{MLP}_{\mathrm{obj}}(q_i)\big), \tag{3}$$

and (4) a per-point contact probability:

$$M_{\mathrm{pred},i}(p) = \sigma\big(q_i \cdot h(p)\big). \tag{4}$$

where $h(p)$ is the PointNet++ feature at point p. The confidence of the a grasp is defined as the maximum contact probability expressed as in (5):

$$s_i = \max_p M_{\mathrm{pred},i}(p), \tag{5}$$

so that $s_i \in [0, 1]$ expresses how strongly the network believes a valid contact lies on the object surface. At inference time we retain only grasp candidates that satisfy (6):

$$o_i > 0.60 \quad \text{and} \quad s_i > 0.60. \tag{6}$$

The top three poses that pass this test are ranked by s_i, exported as a JSON file, and visualized with gripper meshes in Fig. 3d, color-coded from green (highest) to red (lowest within threshold). The highest-ranked pose among them is serialized and passed to the Action Module, which parameterizes the low-level approach-grasp skill and dispatches it to the robot controller.

3.4. Robotic action library

The Action Module constitutes the final translation layer between the symbolic description produced by PaliGemma and the concrete joint level commands required by the robot controller. It receives two inputs: first, the action token sequence emitted by the vision language model for example *connect; approach; grasp; gripper close; target(); gripper open; disconnect*, and second, the geometric parameters computed upstream, namely the six degree of freedom grasp pose and the target position.

Internally, each action is matched to a low level primitive in the robot application interface. Connection and disconnection management, incremental tool motions, cartesian trajectories, and I/O operations are therefore invoked by simple string matching.

Upon reading the token <*connect*> opens a TCP channel to the robot via RTDE. The token <*approach*> then moves the end effector to a pre-grasp waypoint along the approach vector predicted by M2T2, after which <*grasp*> directs a cartesian motion to the exact 6-DoF pose provided in the JSON output. The command <*gripper close*> actuates the jaws, <*target*()> use the placement pose extracted from the vision language output and drives a straight-line motion to that location, <*gripper open*> releases the object, while <*disconnect*> finally closes the RTDE socket and ends the action sequence.

Each command is streamed to the controller only after the previous one has been acknowledged, and any exception such as loss of connection, joint limit violation, or force overload triggers an immediate disconnect to close the communication with the robot. Because the interpreter is table driven, extending the vocabulary to new action token or alternative robot back ends requires no changes to the sequencing logic. In this way the Action Module completes the perception to action loop: a single natural language instruction is converted, through PaliGemma and M2T2, into a sequence of verified joint motions that grasp and manipulate real objects in the workspace.

4. Dataset construction

This section details the data collection and annotation pipeline that supports the two learning tasks in our framework. We begin by describing the robotic setup used to acquire the RGB-D scenes, then present the object-centric dataset for detection and segmentation, and finally introduce the command-centric dataset for training the PaliGemma model that generates robotic action sequences token.

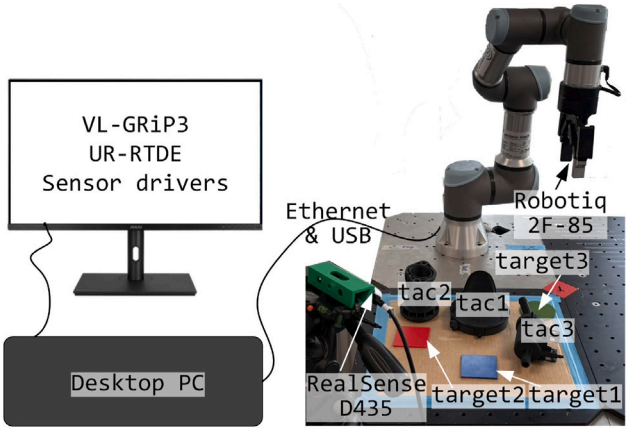


Fig. 4. Experimental setup for data collection and demonstration.

4.1. Robotic system setup

The experimental setup is shown in Fig. 4, consists of a 6-DoF UR3 robotic arm and a two-finger Robotiq parallel gripper used for grasping and manipulation tasks. An Intel RealSense D435 camera, rigidly mounted on a tripod beside the workspace, captures synchronized RGB-D frames of the scene. A desktop PC workstation equipped with an NVIDIA RTX 4090 GPU (24 GB VRAM) controls all hardware components and executes the complete VL-GRiP3 pipeline. The RealSense camera connects directly to the PC via USB, while the UR3 controller communicates with the host over TCP using the ur-rtde interface; the gripper closing force is fixed at 20N to be able to handle also fragile object. All experimental scripts are written in Python. VL-GRiP3 is implemented using PyTorch and the Transformers library. The vision-language model uses the *paligemma-3b-pt-448* checkpoint, which is fine-tuned on the same workstation to optimize performance for object detection, segmentation, and action-sequence generation.

4.2. Dataset for object detection and segmentation

The dataset consists in RGB images at 640 × 480 resolution acquired using an Intel RealSense camera. These images depict three distinct object *tac1*, *tac2*, and *tac3* corresponding to real industrial components supplied by a local small and medium-sized enterprise (SME). The parts were selected to span different geometric characteristics: the symmetrical piece stresses the CAD-based registration stage, while the asymmetrical ones challenge grasp planning and execution. Together, they provide a representative evaluation for SME manufactured components.

Each image is manually annotated with a pixel-level segmentation mask and an object class bounding box. Annotations are provided in the PaliGemma format [19], which encodes discrete location and segmentation tokens followed by the object label. A typical annotation sequence is:

```
'<loc0172><loc0666><loc0775><loc0981>
<seg026><seg009><seg106><seg017><seg052>. . . tac1>'
```

To minimize overfitting and ensure robust model generalization, the selected objects exhibit minimal rotational symmetry, and the images were captured from a wide range of viewpoints. The dataset is split into a validation set of 35 images and a training set comprising 248 images. The training set was generated by applying common data augmentation techniques such as horizontal flipping and in plane rotations to the remaining original samples. This annotated and augmented dataset underpins the training and validation of the PaliGemma perception model, which is dedicated to target-region detection and object segmentation.

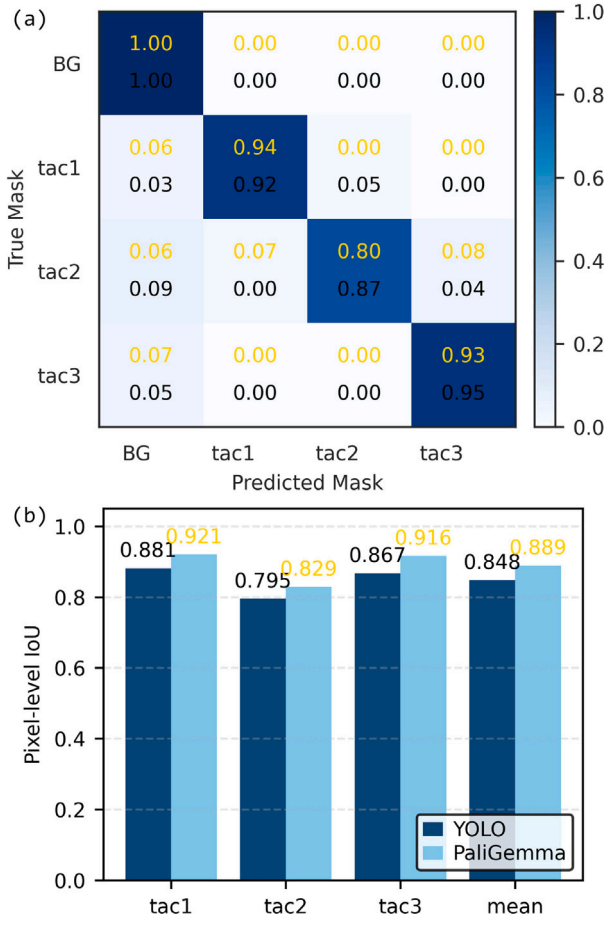


Fig. 5. Comparison of object segmentation performance: (a) Confusion matrix and (b) intersection over union (PaliGemma in yellow numbers, YOLOv8 in black numbers).

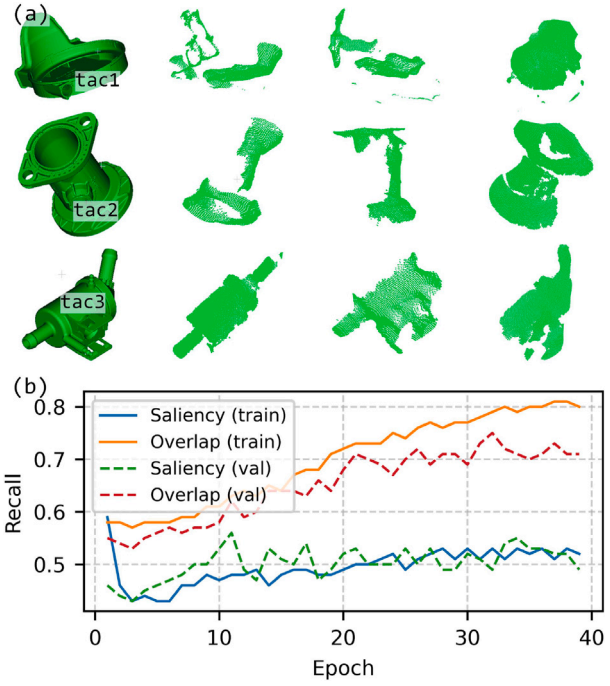


Fig. 6. Training and validation performance: (a) example dataset used for fine-tuning Predator, and (b) Recall of the training and validation.

4.3. Dataset for robotic actions

It comprises 306 annotated samples, partitioned into 231 instances for training and 75 for validation. This dataset is used to train PaliGemma for generating robotic action sequence tokens in response to natural-language commands. Since PaliGemma is a vision-language model, each annotation must reference an image. In our case, each command is paired with the RGB frame captured at the beginning of the task to provide the necessary visual context and ensure format compatibility

Each sample is stored as a JSON record within a single file, specifying (i) the input image, (ii) a natural-language prompt (prefix), and (iii) an ordered list of low-level skills (suffix). A typical entry is:

```
"image": "scene.jpg", "prefix": "move to the red target",
    "suffix": "connect; approach; grasp; gripper
        close; ...target(red); disconnect"'
```

The prefix expresses the high-level intention, while the suffix enumerates the primitive actions token required to execute it.

5. Experimental evaluation of VL-GRiP3

5.1. Object segmentation evaluation

To assess our perception module, we compared PaliGemma with YOLOv8n [56] on a held-out validation set. Accurate segmentation is essential for grasp planning, as it preserves the object's geometry for registration. Both models were fine-tuned on the same dataset, with results shown in Fig. 5. YOLOv8n achieved a recall of 91.3% across the three tissue classes, slightly higher than PaliGemma's 89.0%. However, PaliGemma reached a mean IoU of 88.9% versus 84.8% for YOLOv8n, a gain of 4.1 percentage points. This higher IoU yields cleaner segmentation masks, reducing irrelevant regions and producing denser, more precise point clouds, which directly improve CAD-based registration and subsequent grasp planning.

5.2. Evaluation of point cloud registration with predator

To fine-tune the registration module, we built a compact dataset using a single static RGB-D camera. Each of the three industrial parts (tac1–tac3) was rotated during acquisition, producing depth frames under different occlusions and lighting. This setup introduces variability in viewpoint, shadowing, and density, enabling the network to learn correspondences that remain reliable under realistic factory conditions. Fig. 6a shows the CAD models (tac1-3) alongside representative partial captures provided to Predator during fine-tuning.

Performance is measured with two recall metrics. Saliency recall evaluates how well the model identifies keypoints likely to form useful correspondences between partial and CAD clouds. Overlap recall quantifies the ability to predict regions that are simultaneously visible in both clouds, which is crucial for robust registration. Fig. 6b reports both metrics over 40 epochs: overlap recall increases from 0.55 to 0.80 on training and 0.72 on validation, plateauing after epoch 30; saliency recall rises more moderately from 0.46 to 0.54. The near coincident train/val curves confirm that the model captures general correspondences without over-fitting to these specific parts. As a result, Predator supports reliable CAD registration from a single RGB-D view, filling in occluded geometry and improving object surfaces for downstream grasp planning.

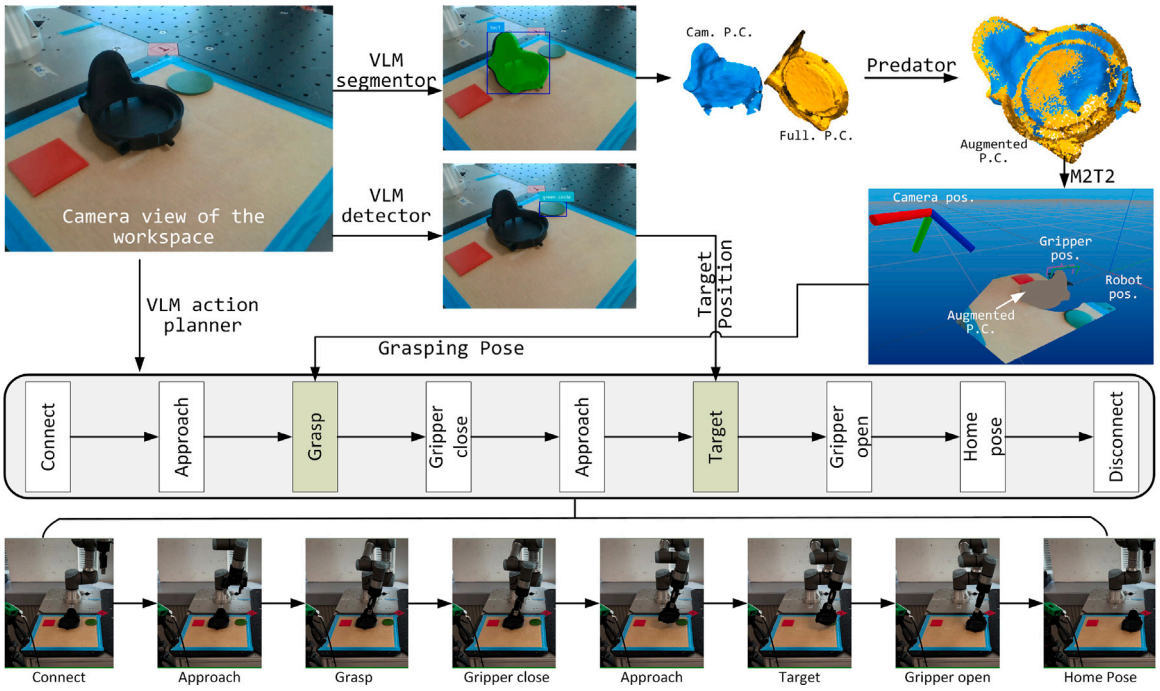


Fig. 7. The demonstration of grasping task for object from manufacturing, with command of "move tac1 to green circle".

Table 1
Comparison of Exact-Match Accuracy (EM%)

Model	EM%
PaliGemma-3B-mix-448	51.43
PaliGemma2-3B-mix-448	86.13

5.3. Robotic actions evaluation

To enable natural-language task planning, we fine-tune PaliGemma to convert user prompts into executable robotic action sequences, allowing the model to generate structured command lists without relying on pre-defined templates or hand-written scripts. We fine-tuned PaliGemma-3B-mix-448 on a dataset of robotic action sequences and evaluated it on a 70-prompt validation set whose instructions were unseen during training. Performance is reported as Exact-Match Accuracy (EM): the proportion of prompts for which the predicted action sequence exactly matches the Ref. [39]. Under this protocol, the fine-tuned PaliGemma-3B attains 51.43% EM. When we fine-tune and evaluate the newer PaliGemma 2-3B-mix-448 on the same data, EM rises to 86.13%, a gain of +34.7 percentage points as shown in Table 1. This substantial improvement demonstrates that the updated architecture transfers far more effectively to our manipulation domain, even with identical fine-tuning data and training procedure. The resulting high EM score indicates that the model consistently generates valid and accurate robot programs aligned with the user's intent. This enables task execution directly from natural language, eliminating the need for hard-coded routines.

5.4. VL-GRiP3 for manipulating real objects from manufacturing

The module-level evaluations of VL-GRiP3 confirm that each component of the pipeline performs reliably. To assess overall performance, we implement the full framework on the setup shown in Fig. 4. Three irregularly shaped automotive components (tac1, tac2, and tac3) are selected for grasping trials to test whether VL-GRiP3 can (i) correctly ground diverse natural-language commands to visual targets, (ii) automatically synthesize executable robot action scripts, and (iii) repeatedly execute geometry-aware 3D grasps, such as moving objects from random positions and orientations to designated locations or relocating them by a specified distance along different directions.

The testing procedure of VL-GRiP3 for one grasping is illustrated in Fig. 7. The whole process begin when a user issues a command, either typed or spoken, to VL-GRiP3. Spoken commands are first transcribed into text by an integrated Whisper module. For example, the user might say "move tac1 to green circle". Next, the global view camera captures the RGB and depth images of the workspace. VL-GRiP3 then interprets the text command as three dedicated prompts for the VLM: segment the object to be grasped, detect the target location, and plan the robotic actions required to fulfill the command. We refer to these components as the VLM segmentor, detector, and action planner, respectively. The VLM segmentor outputs an object mask from the RGB image, from which we derive the partial point cloud via the depth image. This partial cloud is aligned and merged with the full CAD-derived point cloud using a fine-tuned Predator model to reconstruct occluded surfaces from a single view. In semi-structured manufacturing environments, where part geometries are typically known, this yields a complete object model. If the full point cloud of the object is unknown, an additional camera can be mounted on the robotic arm for active perception, however, this is out of scope for this work and will be explored in the future. Once the object's complete point cloud is registered in the workspace, the M2T2 module plans candidate grasp poses based on the full geometry, and the one with the highest confidence is selected as the grasping pose to be executed.

Simultaneously, the VLM detector locates the target specified in the command and records its pose in the robot's base frame for use by the action script. The VLM action planner then generates a sequence of fundamental robotic actions such as connect/disconnect, open/close gripper, move, approach, etc., in text form. These actions feed into our Robotic Action Library to assemble a fully executable script, into which the grasp pose and target pose are also injected. The assembled script drives the robot to complete the requested task. Although the action planner could in principle generate low-level scripts directly, doing so reliably would require extensive VLM pre-training and still introduces syntactic uncertainties; we therefore defer low-level script generation to future work.

Table 2
Success rate for repetitive experiments with VL-GRiP3.

Commands	Tac1	Tac2	Tac3
Move < *tac* > to green circle	10/10	9/10	9/10
Place < *tac* > in red square	10/10	9/10	8/10
Locate < *tac* > on blue square	10/10	10/10	9/10
Shift < *tac* > in positive x by 0.08	10/10	10/10	9/10
Shift < *tac* > in negative y by 0.05	10/10	10/10	9/10

Although the evaluation tasks are formulated as grasp-and-place operations, they are far from trivial. Each trial involves objects with complex and asymmetric geometries, placed in random initial poses, which require accurate 6D grasp pose prediction to ensure success. Moreover, the system must jointly perform semantic understanding of the natural language prompt, precise localization of both the object and the target, and robust integration of perception and action generation. We selected this setup because it directly demonstrates the strengths of VL-GRiP3 in handling the full pipeline; from language comprehension to reliable execution, under realistic manufacturing conditions where object shapes and placements vary.

To evaluate the performance of VL-GRiP3 more rigorously, we conducted repetitive experiments on all three objects with different user commands via real grasping trials, and the success rates are summarized in Table 2. For each command, we conducted 10 grasping trials, placing the object in a random pose and varying the target location across trials to test whether VL-GRiP3 could segment the object, generate the grasping pose, and find the target correctly. From the results, we found that all 50 grasping trials for tac1 were successful, particularly due to its simple shape, which allows the full point cloud to be easily aligned with the partial point cloud captured by the camera. However, for grasping trials with tac2 and tac3, we did experience some failures during grasping. The reason is that the generated grasping pose was not good enough to ensure a firm grip, and the objects fell from the gripper. The key to a better grasping pose is a more accurate object point cloud. Since we only took one image of the workspace from a fixed perspective, and the camera was also affected by lighting conditions, it became more difficult to obtain high-quality point clouds for more complex objects. Even so, we found only 2 and 6 failures for tac2 and tac3 over 50 grasping trials, respectively. The success rates were 96% and 88%, respectively, and the overall success rate for all 150 grasping trials was 94.67%, which demonstrates the reliability and robustness of VL-GRiP3.

5.5. Comparative evaluation with Pi0 and Pi05

To address the need for stronger baselines, we conducted additional experiments comparing our VL-GRiP3 modular pipeline with two state of the art vision-language-action (VLA) foundation models from the Pi family: Pi0 and Pi05. Both models are open-source, pretrained on large-scale multi-robot demonstration datasets and designed to be fine-tuned on new robots and tasks using paired demonstrations [57]. Pi0 represents the lightweight baseline of the Pi family, while Pi05 is a larger variant with extended multimodal training for improved generalization. Architecturally, both operate in an end-to-end manner: given an RGB input and a natural language prompt, they jointly encode vision and language into a shared latent space and autoregressively decode low-level robot actions in the form of joint velocities and gripper commands. At each timestep, the predicted action is executed, the new state is observed, and the loop continues until task completion. This design allows them to generalize across robots and tasks but provides no modular separation between perception, grasp detection, and action generation. In practice, the internal decision-making process is not observable or adjustable, which makes adaptation and debugging challenging.

Table 3
Comparison of grasping and placing performance across orientations. Values are reported as failure–success (F–S) counts over 5 trials.

Orientation	Pi0		Pi05		VL-GRiP3	
	Grasp	Place	Grasp	Place	Grasp	Place
0 °	1–4	4–0	0–5	0–5	0–5	0–5
45 °	1–4	2–2	1–4	0–4	0–5	0–5
90 °	1–4	1–3	3–2	0–2	1–4	0–4

By contrast, VL-GRiP3 follows a modular architecture. Visual perception is handled by a vision-language transformer and grasping is explicitly solved by the M2T2 grasp prediction module. This separation of concerns makes the system more interpretable, as each stage can be individually monitored and fine-tuned. We designed a controlled evaluation to compare our VL-GRiP3 modular pipeline with the Pi architecture (Pi0 and Pi05 variants) on the same manipulation task. The prompt was "place tac2 in the red square", where tac2 is an industrial component with asymmetrical geometry. To test robustness to object pose, tac2 was presented in three orientations: 0°, 45°, and 90°, while the target square was randomly positioned within the workspace. For each orientation, we conducted five randomized trials per model, yielding a total of 45 test episodes across Pi0, Pi05 and VL-GRiP3. For Pi0 and Pi05, we fine-tuned the models on 110 real demonstrations collected in our environment. Each demonstration consisted of paired RGB observations from both the external and wrist cameras, the corresponding natural language instruction, and synchronized robot control signals (joint velocities and gripper state). This format follows the requirements of the Pi architecture, which learns end-to-end mappings from multimodal inputs to low-level robot actions.

Table 3 summarizes the comparative evaluation of Pi0, Pi05, and our VL-GRiP3 system on the industrial pick-and-place task. As expected, Pi05 consistently outperformed Pi0, confirming the improved robustness of the $\pi_{0.5}$ variant, which benefits from pretraining on the DROID dataset [58] and employs knowledge insulation for stronger language grounding. Nevertheless, both Pi0 and Pi05 showed notable limitations in grasping performance when the orientation of the tac2 object was varied. In particular, their action policies often mispredicted grasp points at 45° and 90°, leading to failed executions despite correct placement of targets. VL-GRiP3 achieved almost flawless grasping and placing across all orientations. This improvement is largely due to the use of the M2T2 grasp prediction module, which provided accurate grasp points without requiring additional fine-tuning. The discrepancy highlights a key architectural difference: while Pi0 and Pi05 are end-to-end black-box models, mapping directly from multimodal input to actions, VL-GRiP3 is modular, separating perception, grasp detection, and action execution. This separation allows individual modules to be fine-tuned and debugged in isolation, making the system more transparent and adaptable. Another critical factor is data efficiency. Although we fine-tuned Pi0 and Pi05 on 110 demonstrations collected in our environment, this amount of data proved insufficient to achieve reliable generalization across object orientations. Collecting such demonstrations is also costly and time consuming, which limits the practical usability of end-to-end approaches in SMEs where production tasks change frequently. In comparison, VL-GRiP3 achieved superior performance with a much smaller dataset, as only the perception module required adaptation. This modularity is therefore not only advantageous for interpretability but also for reducing the data and time burden associated with training, making it more suitable for deployment in dynamic industrial settings.

6. Discussion and conclusion

In this paper, we presented VL-GRiP3, a hierarchical VLM-enabled pipeline that closes the loop between language understanding, extended

perception, and action planning, tightly linking natural language intent to feasible, geometry-aware 3D grasping in semi-structured environments. A single VLM backbone performs three roles: target detector, object segmentor, and robotic action planner, while segmented partial point clouds are aligned with CAD models via 3D registration and passed to an M2T2-based 3D grasp pose planner. This CAD-augmented representation recovers missing object geometry from a single RGB-D view, enabling accurate 3D grasping without multi-camera or multi-view setups.

Module-level experiments support these design choices. After fine-tuning on the image dataset, the perception VLM attains 88.9% mean IoU, surpassing a comparably trained YOLOv8 by 4.1%. The Predator-based registration module restores hidden geometry with 0.72 overlap recall on unseen objects, enabling precise 6-DoF pose recovery. On the command dataset, the upgraded PaliGemma2 reaches 86.13% exact-match accuracy, demonstrating robust performance in converting natural-language prompts into executable skill sequences, even for unseen instructions. Although unseen parts may require limited retraining of a specific module, only the affected component is updated, avoiding full-pipeline retraining and preserving flexibility in high-mix manufacturing settings.

When deployed on a UR3e robotic cell with a single RGB-D camera and heterogeneous industrial components, VL-GRiP3 achieves a 94.67% grasp-and-place success rate over 150 trials, without task-specific coding or manual motion tuning. The pipeline transforms natural-language instructions into deployable robot programs, completes partially occluded views through CAD-guided registration, and confines any adaptation to the modules affected by a new part or task. In comparative evaluations against state-of-the-art end-to-end VLA approaches, which pursue broad generalization but require large-scale datasets, substantial computational resources, and often exhibit higher variability in action generation, VL-GRiP3's modular framework with 3D grasp prediction delivers more reliable and deterministic behavior, and is easier to inspect and debug, making it better aligned with SME requirements for repeatability, traceability, and transparency.

Future work will build on VL-GRiP3's current ability to grasp and manipulate arbitrary components. The next step is to move beyond generic grasping and toward object-specific manipulation, where grasp poses and strategies are selected based on the geometry and functional role of the object. This direction would enable more semantically informed and task-aware manipulation, bridging the gap between perception, reasoning, and purposeful interaction in manufacturing settings. In parallel, we envision leveraging VL-GRiP3's modular pipeline to automatically curate paired datasets of visual observations and corresponding robot programs. These image-script pairs could support training a vision language action model for direct robot program generation, offering a scalable path toward more general and autonomous robotic skill acquisition.

CRediT authorship contribution statement

Mirco Polonara: Writing – original draft, Investigation, Visualization, Validation, Software, Methodology, Formal analysis, Data curation, Conceptualization. **Xingyu Yang:** Writing – original draft, Investigation, Visualization, Validation, Methodology, Data curation, Conceptualization. **Luca Carbonari:** Writing – review & editing, Writing – original draft, Data curation. **Xuping Zhang:** Writing – review & editing, Supervision, Methodology.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

The authors would like to thank José Pablo Vásquez for his help in implementing the Predator, and also thank the company Techpol-Srl for providing the object data used in the experiments.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.rcim.2026.103244.

Data availability

Code and data for this work are available at GitHub: https://github.com/AU-DK-Robotics/VL-GRiP3.

References

[1] T.-T. Le, C.-Y. Lin, Bin-picking for planar objects based on a deep learning network: a case study of USB packs, Sensors 19 (16) (2019) 3602.
[2] C. Zhuang, H. Wang, H. Ding, AttentionVote: A coarse-to-fine voting network of anchor-free 6D pose estimation on point cloud for robotic bin-picking application, Robot. Comput.-Integr. Manuf. 86 (2024) 102671.
[3] A. Herzog, P. Pastor, M. Kalakrishnan, L. Righetti, J. Bohg, T. Asfour, S. Schaal, Learning of grasp selection based on shape-templates, Auton. Robots 36 (2014) 51–65.
[4] Y. Domae, H. Okuda, Y. Taguchi, K. Sumi, T. Hirai, Fast graspability evaluation on single depth maps for bin picking with general grippers, in: 2014 IEEE International Conference on Robotics and Automation, ICRA, IEEE, 2014, pp. 1997–2004.
[5] K. Kleeberger, R. Bormann, W. Kraus, M.F. Huber, A survey on learning-based robotic grasping, Curr. Robot. Rep. 1 (2020) 239–249.
[6] A. Sahbani, S. El-Khoury, P. Bidaud, An overview of 3D object grasp synthesis algorithms, Robot. Auton. Syst. 60 (3) (2012) 326–336.
[7] Z. Zhou, L. Li, A. Fürsterling, H.J. Durocher, J. Mouridsen, X. Zhang, Learning-based object detection and localization for a mobile robot manipulator in SME production, Robot. Comput.-Integr. Manuf. 73 (2022) 102229.
[8] X. Yang, Z. Zhou, J.H. Sørensen, C.B. Christensen, M. Ünalan, X. Zhang, Automation of SME production with a Cobot system powered by learning-based vision, Robot. Comput.-Integr. Manuf. 83 (2023) 102564.
[9] J. Mahler, J. Liang, S. Niyaz, M. Laskey, R. Doan, X. Liu, J.A. Ojea, K. Goldberg, Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics, 2017, arXiv preprint arXiv:1703.09312.
[10] J. Redmon, A. Angelova, Real-time grasp detection using convolutional neural networks, in: 2015 IEEE International Conference on Robotics and Automation, ICRA, IEEE, 2015, pp. 1316–1322.
[11] D. Kalashnikov, A. Irpan, P. Pastor, J. Ibarz, A. Herzog, E. Jang, D. Quillen, E. Holly, M. Kalakrishnan, V. Vanhoucke, et al., Scalable deep reinforcement learning for vision-based robotic manipulation, in: Conference on Robot Learning, PMLR, 2018, pp. 651–673.
[12] L. Li, X. Yang, R. Wang, X. Zhang, Automatic robot hand-eye calibration enabled by learning-based 3D vision, J. Intell. Robot. Syst. 110 (3) (2024) 130.
[13] J. Xu, S. Jin, Y. Lei, Y. Zhang, L. Zhang, RT-grasp: Reasoning tuning robotic grasping via multi-modal large language model, in: 2024 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, IEEE, 2024, pp. 7323–7330.
[14] M. Polonara, A. Romagnoli, G. Biancini, L. Carbonari, Introduction of collaborative robotics in the production of automotive parts: A case study, Machines 12 (3) (2024) 196.
[15] Z. Xie, X. Liang, C. Roberto, Learning-based robotic grasping: A review, Front. Robot. AI 10 (2023) 1038658.
[16] J. Li, D. Li, S. Savarese, S. Hoi, Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models, in: International Conference on Machine Learning, PMLR, 2023, pp. 19730–19742.
[17] Z. Peng, W. Wang, L. Dong, Y. Hao, S. Huang, S. Ma, F. Wei, Kosmos-2: Grounding multimodal large language models to the world, 2023, arXiv preprint arXiv:2306.14824.
[18] T. Ren, S. Liu, A. Zeng, J. Lin, K. Li, H. Cao, J. Chen, X. Huang, Y. Chen, F. Yan, et al., Grounded sam: Assembling open-world models for diverse visual tasks, 2024, arXiv preprint arXiv:2401.14159.
[19] L. Beyer, A. Steiner, A.S. Pinto, A. Kolesnikov, X. Wang, D. Salz, M. Neumann, I. Alabdulmohsin, M. Tschannen, E. Bugliarello, et al., Paligemma: A versatile 3b vlm for transfer, 2024, arXiv preprint arXiv:2407.07726.
[20] R. Sapkota, Y. Cao, K.I. Roumeliotis, M. Karkee, Vision-language-action models: Concepts, progress, applications and challenges, 2025, arXiv preprint arXiv:2505.04769.

[21] S. Li, Z. Yan, Z. Wang, Y. Gao, VLM-msgraph: Vision language model-enabled multi-hierarchical scene graph for robotic assembly, Robot. Comput.-Integr. Manuf. 94 (2025) 102978.
[22] R. Zhang, H. Zhang, Z. Zheng, VL-uncertainty: Detecting hallucination in large vision-language model via uncertainty estimation, 2024, arXiv preprint arXiv:2411.11919.
[23] S. Zhai, Q. Zhang, T. Zhang, F. Huang, H. Zhang, M. Zhou, S. Zhang, L. Liu, S. Lin, J. Pang, A vision-language-action-critic model for robotic real-world reinforcement learning, 2025, arXiv preprint arXiv:2509.15937.
[24] B. Zhang, Y. Zhang, J. Ji, Y. Lei, J. Dai, Y. Chen, Y. Yang, Safevla: Towards safety alignment of vision-language-action model via constrained learning, 2025, arXiv preprint arXiv:2503.03480.
[25] S. Huang, Z. Jiang, H. Dong, Y. Qiao, P. Gao, H. Li, Instruct2act: Mapping multi-modality instructions to robotic actions with large language model, 2023, arXiv preprint arXiv:2305.11176.
[26] Y. Li, P. Jiang, C. Chai, X. Zhang, C. Liu, A framework for robotic manipulation tasks based on multiple zero shot models, Sci. Rep. 15 (1) (2025) 31141.
[27] Z. Zhou, L. Li, R. Wang, X. Zhang, Experimental eye-in-hand calibration for industrial mobile manipulators, in: 2020 IEEE International Conference on Mechatronics and Automation, ICMA, IEEE, 2020, pp. 582–587.
[28] A. Bicchi, V. Kumar, Robotic grasping and contact: A review, in: Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065), 1, IEEE, 2000, pp. 348–353.
[29] J. Bohg, A. Morales, T. Asfour, D. Kragic, Data-driven grasp synthesis—a survey, IEEE Trans. Robot. 30 (2) (2013) 289–309.
[30] D. Morrison, P. Corke, J. Leitner, Closing the loop for robotic grasping: A real-time, generative grasp synthesis approach, 2018, arXiv preprint arXiv:1804.05172.
[31] H.-S. Fang, C. Wang, M. Gou, C. Lu, Graspnet-1billion: A large-scale benchmark for general object grasping, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 11444–11453.
[32] Z. Liu, Z. Chen, S. Xie, W.-S. Zheng, Transgrasp: A multi-scale hierarchical point transformer for 7-dof grasp detection, in: 2022 International Conference on Robotics and Automation, ICRA, IEEE, 2022, pp. 1533–1539.
[33] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Adv. Neural Inf. Process. Syst. 30 (2017).
[34] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, in: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), 2019, pp. 4171–4186.
[35] A. Radford, J.W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al., Learning transferable visual models from natural language supervision, in: International Conference on Machine Learning, PmLR, 2021, pp. 8748–8763.
[36] M.Z. Hossain, F. Sohel, M.F. Shiratuddin, H. Laga, A comprehensive survey of deep learning for image captioning, ACM Comput. Surv. 51 (6) (2019) 1–36.
[37] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. Lawrence Zitnick, D. Parikh, VQA: Visual question answering, in: Proceedings of the IEEE International Conference on Computer Vision, ICCV, 2015, pp. 2425–2433.
[38] Y.-C. Chen, L. Li, L. Yu, A. El Kholy, F. Ahmed, Z. Gan, Y. Cheng, J. Liu, Uniter: Learning universal image-text representations, 2019.
[39] A. Steiner, A.S. Pinto, M. Tschannen, D. Keysers, X. Wang, Y. Bitton, A. Gritsenko, M. Minderer, A. Sherbondy, S. Long, et al., Paligemma 2: A family of versatile vlms for transfer, 2024, arXiv preprint arXiv:2412.03555.
[40] J. Gao, B. Sarkar, F. Xia, T. Xiao, J. Wu, B. Ichter, A. Majumdar, D. Sadigh, Physically grounded vision-language models for robotic manipulation, in: 2024 IEEE International Conference on Robotics and Automation, ICRA, IEEE, 2024, pp. 12462–12469.
[41] W. Huang, C. Wang, R. Zhang, Y. Li, J. Wu, L. Fei-Fei, Voxposer: Composable 3d value maps for robotic manipulation with language models, 2023, arXiv preprint arXiv:2307.05973.
[42] J. Duan, W. Yuan, W. Pumacay, Y.R. Wang, K. Ehsani, D. Fox, R. Krishna, Manipulate-anything: Automating real-world robots using vision-language models, 2024, arXiv preprint arXiv:2406.18915.
[43] J. Liang, W. Huang, F. Xia, P. Xu, K. Hausman, B. Ichter, P. Florence, A. Zeng, Code as policies: Language model programs for embodied control, in: 2023 IEEE International Conference on Robotics and Automation, ICRA, IEEE, 2023, pp. 9493–9500.
[44] Q. Li, Y. Liang, Z. Wang, L. Luo, X. Chen, M. Liao, F. Wei, Y. Deng, S. Xu, Y. Zhang, et al., Cogact: A foundational vision-language-action model for synergizing cognition and action in robotic manipulation, 2024, arXiv preprint arXiv:2411.19650.
[45] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, et al., π_0: A vision-language-action flow model for general robot control, 2024, arXiv preprint arXiv:2410.24164.
[46] B. Xie, E. Zhou, F. Jia, H. Shi, H. Fan, H. Zhang, H. Li, J. Sun, J. Bin, J. Huang, et al., Dexbotic: Open-source vision-language-action toolbox, 2025, arXiv preprint arXiv:2510.23511.
[47] C. Li, J. Liu, G. Wang, X. Li, S. Chen, L. Heng, C. Xiong, J. Ge, R. Zhang, K. Zhou, et al., A self-correcting vision-language-action model for fast and slow system manipulation, 2024, arXiv preprint arXiv:2405.17418.
[48] J. Bjorck, F. Castañeda, N. Cherniadev, X. Da, R. Ding, L. Fan, Y. Fang, D. Fox, F. Hu, S. Huang, et al., Gr00t n1: An open foundation model for generalist humanoid robots, 2025, arXiv preprint arXiv:2503.14734.
[49] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, et al., Rt-1: Robotics transformer for real-world control at scale, 2022, arXiv preprint arXiv:2212.06817.
[50] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, X. Chen, K. Choromanski, T. Ding, D. Driess, A. Dubey, C. Finn, et al., Rt-2: Vision-language-action models transfer web knowledge to robotic control, 2023, arXiv preprint arXiv:2307.15818.
[51] Q. Vuong, S. Levine, H.R. Walke, K. Pertsch, A. Singh, R. Doshi, C. Xu, J. Luo, L. Tan, D. Shah, et al., Open x-embodiment: Robotic learning datasets and rt-x models, in: Towards Generalist Robots: Learning Paradigms for Scalable Skill Acquisition@ CoRL2023, 2023.
[52] S. Huang, Z. Gojcic, M. Usvyatsov, A. Wieser, K. Schindler, Predator: Registration of 3d point clouds with low overlap, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 4267–4276.
[53] W. Yuan, A. Murali, A. Mousavian, D. Fox, M2t2: Multi-task masked transformer for object-centric pick and place, 2023, arXiv preprint arXiv:2311.00926.
[54] A. Radford, J.W. Kim, T. Xu, G. Brockman, C. McLeavey, I. Sutskever, Robust speech recognition via large-scale weak supervision, in: International Conference on Machine Learning, PMLR, 2023, pp. 28492–28518.
[55] G. Team, T. Mesnard, C. Hardin, R. Dadashi, S. Bhupatiraju, S. Pathak, L. Sifre, M. Rivière, M.S. Kale, J. Love, et al., Gemma: Open models based on gemini research and technology, 2024, arXiv preprint arXiv:2403.08295.
[56] M. Sohan, T. Sai Ram, C.V. Rami Reddy, A review on yolov8 and its advancements, in: International Conference on Data Intelligence and Cognitive Informatics, Springer, 2024, pp. 529–545.
[57] P. Intelligence, K. Black, N. Brown, J. Darpinian, K. Dhabalia, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, et al., A vision-language-action model with open-world generalization, 2025, arXiv preprint arXiv:2504.16054.
[58] A. Khazatsky, K. Pertsch, S. Nair, A. Balakrishna, S. Dasari, S. Karamcheti, S. Nasiriany, M.K. Srirama, L.Y. Chen, K. Ellis, et al., Droid: A large-scale in-the-wild robot manipulation dataset, 2024, arXiv preprint arXiv:2403.12945.