Tampere University

Atte Nyyssönen

VISION-LANGUAGE MODELS IN INDUSTRIAL ROBOTICS

Bachelor's Thesis
Faculty of Engineering and Natural Sciences
Examiner: Roel Pieters
April 2024

ABSTRACT

Atte Nyyssönen: Vision-Language Models in Industrial Robotics
Bachelor's Thesis
Tampere University
Bachelor's Programme in Engineering Sciences
April 2024

As the use of industrial robots grows due to the demand for high productivity and quality in nearly all modern industrial sectors, so does the need for robotic systems to have rapid adaptability and guaranteed safety. While current robotic systems are limited to pre-known environments and fixed tasks, combining industrial robotics with Vision-Language Models (VLM) could endow robots with the ability to complete tasks based on language commands without explicit instructions in varied environments. This thesis examines current state-of-the-art robotic manipulation VLMs and the potential of integrating them into industrial systems by utilizing the VLMs requirements and benchmarking. Benchmarking applications play a pivotal role in generating information on the capabilities and weaknesses of VLMs tailored for robotic manipulation without requiring expensive real-world experiments.

The thesis consists of three parts: literature review, experimentation with results, and implementation requirements. The literature review explores the most common types of industrial robots and ways of conducting robotic manipulation with VLMs, as well as elaborates on state-of-the-art robotic manipulation VLMs and benchmark applications. The experimentation part includes running a robotic manipulation VLM with benchmarking software and elaborating on the model's results as well as the performance of other models. The implementation part examines the possibilities for integrating VLMs into industrial robotic systems by leveraging running requirements, performance results, and the present state of research.

The results of this thesis show that while robotic manipulation for task completion is achievable solely through language instructions by VLMs, the execution accuracy is not up to par with the requirements set for industrial robotics. The results also highlight other obstacles, such as research focusing on collaborative robots instead of industrial robots and the lack of computational power to run VLMs in industrial robotic systems. As robotic manipulation VLMs progress, the need for reliable benchmarking software that provides more satisfying information increases.

Keywords: industrial robotics, Vision-Language Model, benchmarking, language-conditioned robotic manipulation, collaborative robotics

TIIVISTELMÄ

Atte Nyyssönen: Näkömallien hyödyntäminen teollisuuden robotiikassa
Kandidaatintyö
Tampereen yliopisto
Teknisten tieteiden kandidaattiohjelma
Huhtikuu 2024

Teollisuusrobottien käyttäminen on yleistynyt melkein kaikilla nykypäivän teollisuuden sektoreilla, jotta vaatimukset korkeasta tuottavuudesta ja laadusta saadaan täytettyä. Samalla kasvaa tarve robottijärjestelmille, jotka kykenevät sekä varmistamaan turvallisuuden että mukautumaan erilaisiin tehtäviin. Nykyisten robottijärjestelmien toimintaa rajoittavat vaatimukset ennalta tunnetusta ympäristöstä sekä vakituiset tehtävät. Näistä rajoituksista voitaisiin päästä eroon yhdistämällä teollisuusrobotit ja näkömallit. Näkömallia hyödyntämällä robotti kykenisi käsittelemään ja toteuttamaan tietyllä kielellä annettuja käskyjä vaatimatta tarkempia ohjeita, vaikka ympäröivä ympäristö vaihtelisi. Tässä työssä tarkastellaan uusinta tekniikkaa edustavia robottien manipulointia varten tehtyjä näkömalleja ja niiden käyttöönoton mahdollisuuksia hyödyntämällä sekä näkömallien vaatimuksia että suorituskykyä mittaavaa sovellusta. Suorituskykyä mittaavat sovellukset ovat erittäin tärkeitä, sillä ne tuottavat tietoa robottien manipulointia varten tehtyjen näkömallien kyvyistä ja heikkouksista ilman kalliita kokeita.

Työ on jaettu kolmeen osaan seuraavasti: kirjallisuuskatsaus, koe tuloksineen ja käyttöönoton vaatimukset. Kirjallisuuskatsauksessa tarkastellaan yleisiä teollisuusrobottien tyyppejä ja tapoja toteuttaa robottien manipulointia näkömallien avulla. Kirjallisuuskatsauksessa käsitellään myös viimeisimpiä robottien manipulointia varten tehtyjä näkömalleja sekä niiden suorituskykyä mittaavia sovelluksia. Työn kokeessa mitattiin robottien manipulaatiota varten tehdyn näkömallin suorituskykyä siihen soveltuvalla ohjelmalla. Tämän jälkeen käsiteltiin mallilla saatuja tuloksia sekä vertailtiin niitä muiden samankaltaisten mallien tuloksiin. Käyttöönottoa käsittelevässä osassa tarkastellaan näkömallien ja teollisten robottijärjestelmien yhdistämisen mahdollisuutta soveltamalla näkömallien vaatimuksia, suorituskyvystä saatuja tuloksia ja tutkimustyön nykytilaa.

Työn tuloksista nähdään, että vaikka robottien manipulointia varten tehty näkömalli kykenee kielellisten ohjeiden perusteella suorittamaan annettuja tehtäviä robottia liikuttamalla, se ei kuitenkaan kykene suorittamaan tehtäviä teollisuuden roboteilta vaaditulla tarkkuudella. Tuloksissa havaitaan myös muita haasteita kuten aiemman tutkimuksen keskittyminen yhteistoiminnalliseen robotiikkaan sekä näkömallien suorittamiseen tarvittavan laskennallisen tehon puute nykyisissä teollisuuden robottijärjestelmissä. Kun robottien manipulointia varten tehdyt näkömallit paranevat, tulee myös suorituskykyä mittaavien ohjelmien kehittyä, jotta malleista saadaan tuotettua kattavaa tietoa mahdollisimman varmasti.

Avainsanat: teollisuuden robotiikka, näkömalli, suorituskyvyn mittaaminen, kielellä ehdollistettu robotin manipulointi, yhteistoiminnallinen robotiikka

CONTENTS

LIST OF SYMBOLS AND ABBREVIATIONS

AI	Artificial Intelligence
AMR	Autonomous Mobile Robot
AMSolver	Automatic Manipulation Solver
CALVIN	Composing Actions from Language and Vision
CLIP	Contrastive Language–Image Pre-training
CNN	Convolutional Neural Network
CPU	Central Processing Unit
DOF	Degree(s) of Freedom
GB	Gigabytes
GPU	Graphics Processing Unit
GUI	Graphical User Interface
HRC	Human-Robot Collaboration
HULC	Hierarchical Universal Language Conditioned Policies
HULC++	Hierarchical Universal Language Conditioned Policies 2.0
LCD	Language Control Diffusion
LH-MTLC	Long-Horizon Multi-Task Language Control
LLM	Large Language Model
MCIL	Multicontext Imitation Learning
MTLC	Multi-Task Language Control
OS	Operating System
PC	Personal Computer
PerAct	Perceiver-Actor
PLC	Programmable Logic Controller
PLM	Pre-trained Language Model
RGB	Red-Green-Blue
RGB-D	Red-Green-Blue-Depth
RT-2	Robot Transformer 2
SCARA	Selective Compliance Assembly/Articulated Robot Arm
SPIL	Skill Prior based Imitation Learning
TB	Terabyte
VAPO	Visual Affordance-guided Policy Optimization
VIMA-Bench	VisuoMotor Attention Benchmark
ViT	Vision Transformer
VLM	Vision-Language Model
VLMbench	Vision-and-Language Manipulation benchmark
WSL 2	Windows Subsystem for Linux version 2

1. INTRODUCTION

Since the 1980's the prominence of robots designed for industrial usage has been increasing especially in industries that capitalize on efficient manufacturing lines where tasks are repetitive and continuous. The prominence of robots is also increasing in other industrial sectors e.g. logistics due to their precision, consistency, and ability to complete tasks deemed too rigorous for humans. [1] As of the year 2022, estimates by the International Federation of Robotics reveal that an estimated 3,903,633 industrial robots were in operation worldwide with electrical and automotive industries having the largest growths in installed robots [2]. According to safety requirements outlined in the ISO standard 10218-1 [3] and 10218-2 [4] industrial robots used in manufacturing must work separately from humans in a designated zone due to the multiple potential hazards posed by the robot's lack of contextual awareness. This deficiency in awareness stems from the robot's inherent programming paradigm, wherein the robot's deterministic behaviour is caused by the strict adherence to program instructions to carry out a specific task. The robot will "blindly" follow the programmed instructions if abnormalities aren't detected, thus possibly causing damage to anything in the way. [3][4] Potential hazards like collisions especially with high-speed and heavy payload robots can be deadly to humans but also cause serious damage to other objects in the workspace.

In recent years there have been developments towards collaborative robots, also called cobots, that are designed for direct human-robot collaboration (HRC). According to Wang *et al.* [5] HRC refers in a manufacturing context to an industrial environment where robots and humans work together near one another. This collaborative interaction enables the completion of tasks that require contextual awareness and problem-solving on a human level but also consistency and load handling on a robot level [5]. For the collaboration to be safe for humans, cobots have strict safety features that limit e.g. speed and force [6]. Restrictions like slower movements have a negative impact on the robot's production efficiency and limit their potential use cases. This has sparked a growing interest in developing innovative solutions to endow industrial robots with contextual awareness using the help of Artificial intelligence (AI) [7]. The goal is to enable robots to adapt to any state of the surrounding environment, while also improving safety without needing heavy restrictions that hinder their usability.

The meteoric rise of AI and the rapid expansion of research especially in robotics adjacent fields that include computer vision and natural language processing [8] has brought about revolutionary potential for technological progression in robotics. The union of computer vision and natural language processing has made the creation of Vision-Language models (VLM) possible. With VLMs, computers can support and manipulate human language as well as support the processing, understanding, and analysis of digital images. Concurrently the recent advances in multi-modal learning have created the conditions for efficient training of VLMs. Multi-modal learning uses specialized modelling algorithms and strategies that enable a machine to learn from multiple forms of data. [9] For example, OpenAI's VLM Contrastive Language–Image Pre-training (CLIP) used the prediction of image captioning, which image goes with which caption, to create a base for image representations from scratch. After the base was created, natural language was used to reference learned visual concepts. This multi-modal training made it possible for CLIP to universally understand images and text while outperforming or matching models that were trained on a specific dataset. [10]

Combining industrial robots with VLMs, and multi-modal learning could give robots the ability to process and understand different contexts and situations. This ability would give robots the necessary knowledge for adapting the programming for even the possible corner cases that the programmers of the robot could have missed. The greatest benefit of VLMs in industrial robots would be turning instructions which are communicated by human speech to direct manipulation without the need to specify every step in the process. This thesis assesses the viability of integrating robots with VLMs by their performance, which is determined by benchmarking.

2. THESIS SCOPE AND OBJECTIVES

Given the broad complexity of the chosen topic and the inherent work limit imposed by the nature of the undergraduate thesis, the scope is deliberately confined. The primary objective of the thesis is to experiment with an existing VLM by investigating hardware, software, and system necessities for successful execution with a benchmark. Subsequently analysing the results and observations to assess the viability of incorporating VLMs into industrial robotics.

The scope can be divided into separate objectives which include:

- Looking into contemporary research and terminology regarding industrial robotics, machine learning and benchmarking.
- Assessing the required hardware setup and system requirements for model execution with, including software components and essential datasets.
- Evaluating the chosen VLMs focus and limitations by benchmarking.
- Comparing the results of the chosen VLM to other models that have been benchmarked with the same solution by accentuating the differences in e.g. implementations.
- Determining the practical implications of model usage in industrial setting by considering the implementation challenges and contextual relevance of industrial robots.

3. LITERATURE REVIEW

3.1 Industrial robots

According to ISO standard 8373 industrial robots are defined as automatically controlled reprogrammable multipurpose manipulators, that are programmable on three or more axes. These axes are also called degree(s) of freedom (DOF). The DOF are independent variables that define a robot's configuration or state. In industrial robots, the last link or the endpoint is fitted with an end-effector. The end-effector is the part of the robot that interacts with the environment, e.g. a gripper or a tool is fitted into the last link of the robot to accomplish tasks. [11] The structure of the four typical industrial robots is shown in Figure 1 and elaborated on afterward.

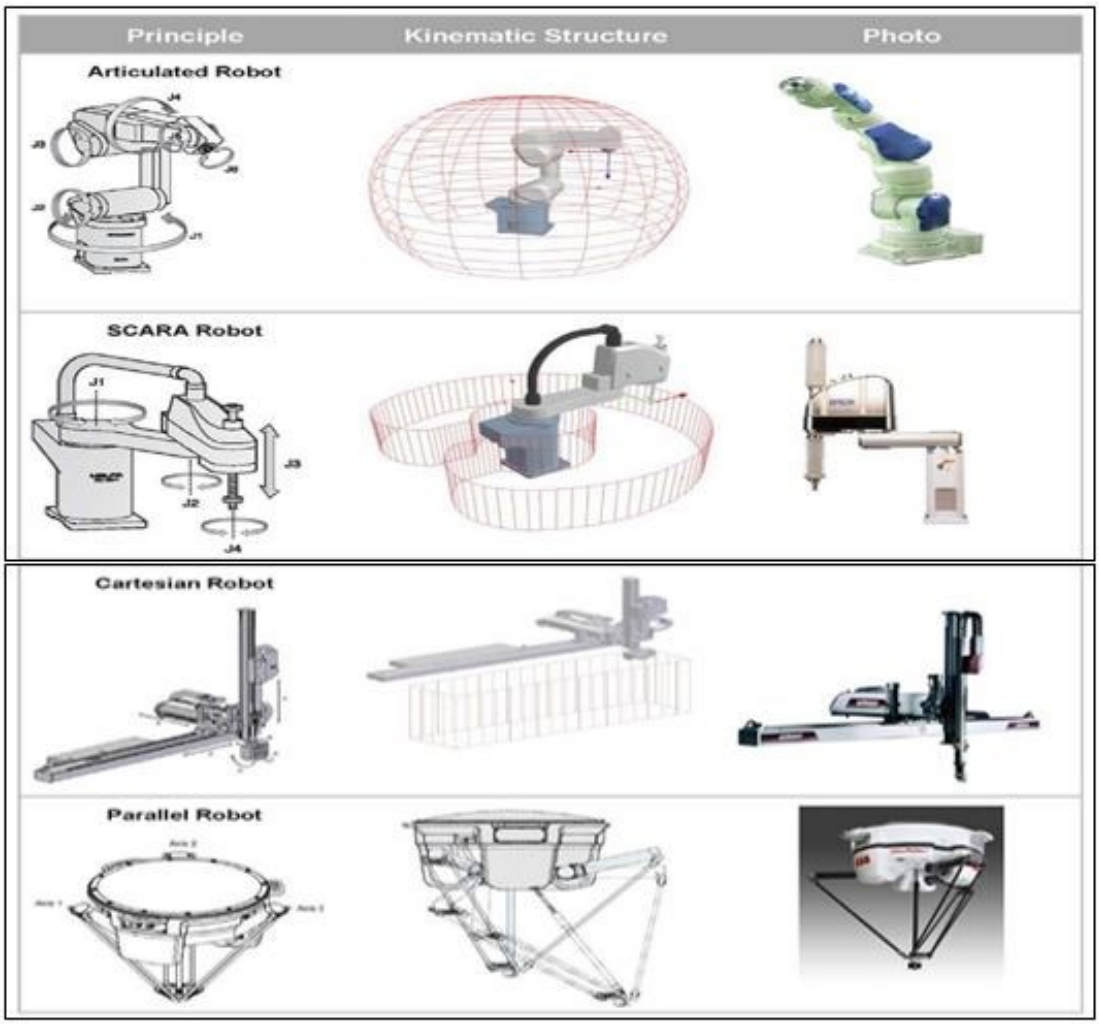

Figure 1: Four of the typical industrial robot structures. [12]

Articulating robots have at least three rotary joints which determines the robot's range of motion and DOF. The robot's motions are designed to mimic movements that are achievable with a worker's arm. [12] Thus, articulated robots can be designed for tasks that require similar motions. Depending on the design and application, articulated robots can handle light to heavy payloads [12]. The necessary data for calculations that enable the automated control of the robot's manipulation are derived from rotary sensors infused in the joints of the robot as well as application-dependent sensors of the end-effector. End-effector data for location and orientation mapping could be gathered for example with ultrasonic waves. [13] Articulated robots usually have 6 DOF [13] and are the most common type of industrial robot configuration [14 p. 22].

Selective compliance assembly, or articulated, robot arm (SCARA) robots have two parallel joints that give the robot the ability to traverse a plane. This means SCARA robots have flexible movements on an X-Y axis while remaining rigid on the Z-axis. [12] Due to the rigidness of the Z-axis SCARA robot end-effectors must provide the ability to move on the Z-axis if the application needs it. According to the structure of He, *et al.* [15] automatic control of a SCARA robot is comparable to an articulated robot, but SCARA robots only have 4 DOF. Limitations that stem from the 4 DOF make SCARA robots cheaper and more efficient than articulated robots. The efficiency difference is the result of 4 DOF calculations being simpler than higher degrees. SCARA robots are utilized especially in manufacturing lines for picking and assembly. [15]

Cartesian robots also known as linear robots use joints and axes with a sturdy structure to operate in a Cartesian coordinate system. The movement of a cartesian robot can either be two- or three-dimensional. [12] Use cases of a cartesian robot range from 3D printers [12] to harvesting [16]. The end-effector can be anything from a gripper to an articulating robot arm [16]. The DOF for cartesian robots is either two or three depending on the dimensions in use. The range of motion is only constrained by the length of the linear rail structure constructed for the robot system. Automated control of the robot's manipulation is achieved by mapping location coordinates based on interests in the area surrounded by the robot's structure [16]. These coordinates for the interest points can be located with vision sensors e.g. a camera [16].

Parallel robots function by separately controlled chains, cables, or other strands connecting to a single endpoint. This endpoint can be e.g. an end-effector or a platform. [12] This enables parallel robots to accomplish a variety of tasks ranging from high-speed tracking photography to exoskeletons [17]. Due to the nature of parallel robots' definition, the DOF and automated control are dependent on the application. For example, a pick-

and-place parallel robot that utilizes cables may have 4 DOF, while a cable-driven 3D printer can have 6 DOF [17].

Outside the typical industrial robots shown in Figure 1, which are either stationary or have restricted movement, there has been a rise in adaptations of autonomous mobile robots (AMRs) boosted by the COVID-19 pandemic. AMRs utilize a multitude of different navigation techniques to calculate a conflict-free path for movement in a dynamic environment. The calculations that enable the AMR to avoid oncoming obstacles depend on visionary data gathered with a sensor system, which needs to be robust to enable real time feedback. [18] An example of a modern AMR is the TUG Autonomous Mobile Robot developed by Aethon [19]. TUG is a self-driving robot that autonomously performs delivery tasks while navigating the complex environments of a hospital [19].

3.2 Achieving robot manipulation with vision-language models

To establish robotic manipulation from certain inputs e.g. starting a task described with speech and following through by utilizing sensory inputs, the model is required to handle multimodal data efficiently [20]. Models must overcome crucial challenges to establish an understanding of handling robotic manipulation. These challenges include extracting semantic meaning from natural language, utilizing built-in sensors to form an understanding of the surrounding environment, grounding natural language to perceived objects in the environment, and forming precise motion controls [20]. Many model approaches split the implementation into three modules that handle the listed challenges when combined. The three modules include a language module, a perception module, and a control module. [20] A regular VLM such as CLIP [10] can understand images taken of the surrounding environment and describe them but without a control module, the model cannot convert its understanding into robotic manipulation.

A language module is used to extract and process the semantics of natural language [20]. The module can generate high-level semantic knowledge of a given task, which is then fed forward to enable the grounded creation of low-level policy [20]. Some early approaches to language-conditioned robotic manipulation e.g. Concept2Robot [21] used a pre-trained language model (PLM) BERT [22]. Research findings showed that scaling a PLM leads to better performances, thus newer adaptations utilize general-purpose large language models (LLM) [20]. For example, Rana *et al.* [23] leverage GPT-4 [24] LLM to develop a scalable large-scale robotic task planner.

Perception module's tasks include object detection, semantic segmentation, and 3D reconstruction [20]. All these tasks enable the module to produce information on the perceived environment state. To produce this information the module leverages either convolutional neural networks (CNN) or vision transformers (ViT) [20]. CNNs such as ResNet [25] can detect objects and perform image segmentation from computer vision. ViT has emerged as an alternative to CNNs due to its different strengths and weaknesses [26]. ViT accomplishes image segmentation and recognition by breaking down images into patches which are serialized into vectors. These vectors are then encoded by a transformer. [27]

The control module takes the data from language and perception modules combined as its input [20]. This multimodal information is combined with robotic data, and imitation or reinforcement learning to achieve policy formation for manipulation towards goals [20]. In other words, the control module generates low-level policies that complete the multimodal high-level planning. Approaches that use imitation learning usually involve behavioral cloning from robotic data [28], while applications utilizing reinforcement learning use reward functions to work towards goals [20]. An example of a control module implementation is Transporter [29], that parametrizes robot actions from visual inputs.

3.3 Benchmarking

In assessing any AI models, whether VLM or other, reliability and accuracy are especially important in robotics due to the applications having real-life impacts and potential safety hazards. A poorly functioning model for continuous control policies can cause serious injuries to workers or monetary losses due to dysfunctional task executions. Diverse benchmarks provide objective, reliable, and reproducible information about the potential strengths and shortcomings of a model's accuracy from multiple points of view. With the help of reliable benchmarks, the performance of a certain model can be used as a standard for comparison. This means that different models evaluated on the same benchmark can be compared against the standard to produce reliable information. As stated in Chapter 3.1 articulated robots are the most common industrial robots, thus state-of-the-art benchmarks for robot manipulation focus on articulated configurations [20]. Robot manipulation benchmarks utilize a multitude of simulators to enable thorough testing and hard-to-replicate exploration of complex scenarios [20]. Some state-of-the-art benchmarks for evaluating robotic manipulation VLMs are VisuoMotor Attention Benchmark (VIMA-Bench) [30], Vision-and-Language Manipulation benchmark (VLMbench) [31], and Composing Actions from Language and Vision (CALVIN) [32].

VIMA-Bench uses a simulator-based approach to evaluate models' ability to execute general robot manipulation tasks from multimodal inputs [30]. The multimodal inputs are composed of a combination of text and Red-Green-Blue (RGB) images. The simulated environment used within the evaluation consists of a tabletop setting where the robot arms' task is to manipulate objects [30]. VIMA-Bench is built on top of the Ravens physics simulator [29] that simulates the movements of a Universal Robot UR5 arm [33][30]. The strength of VIMA-Benches evaluation comes from a multilevel test structure that systemically checks a model's ability to generalize into tasks with 4 different difficulty levels. The 4 levels of difficulty are defined as random object placement with seen prompts, novel combinations of seen objects, introduction of new objects not seen during training, and completely new tasks and prompts. The possible objects and textures can be seen in the papers Appendix A, and examples of tasks with prompts in Appendix B. The tasks cover a model's ability *e.g.* imitation learning, reasoning, and goal-reaching beyond what it was trained with. VIMA-Bench has a few shortcomings that include limited realism and task complexity as well as limited action primitives. [30] This means that the simulator and tasks cannot be directly applied to an industrial setting. Limited action primitives mean that there is a limited number of orders for the robot to complete and the orders are not intricate e.g. wipe, pick-and-place.

VLMBench utilizes CoppeliaSim [34] simulated environment to render movements of a 7-DOF Franka Emika Panda robot arm [35][31]. The arm is standing on a wooden table and data is gathered with 5 RGB-Depth (RGB-D) cameras embedded into the environment. VLMBench tests a model's ability to generate an executable manipulation trajectory to achieve specific task goals with the help of visual observations. The tasks are given to the model as compositional language instructions such as "Open the door of the fridge" or "Open the door of a dishwasher". The strength of VLMBench compared to other benchmarks comes from its implementation of Automatic Manipulation Solver (AMSolver) that scales and builds complex multi-step tasks. With the help of AMSolver VLMBench can automatically generate trajectories, formulate tasks from constraints, automate 6-DOF grasping, and adapt into novel objects. The automation also enables variations inside the tests, in other words, while the task goal remains the same, objects in the environment can change. Other existing state-of-the-art benchmarks don't have all these test functionalities and instead have only a few. This means that while other benchmarks focus on high-level task descriptions, VLMBench also has low-level robot actions. All the task details can be found in the papers Appendix A. VLMBenches limitations include the absence of objects with soft materials: only rigid body objects are considered. The evaluation only supports direct tasks such as pick-and-place. As was the

case with VIMA-Bench, VLMBench is not easily adaptable to the real world because the training and validation data is generated inside a simulator. [31]

CALVIN benchmark evaluates a model's ability to learn long-horizon language-conditioned tasks from unstructured teleoperated "play" data collected with the help of a virtual reality headset [32]. Long horizon means in the context of robotics that a certain end goal is reached successfully only if the execution of multiple sub-tasks is successful. CALVIN offers four different but structurally related environments: A, B, C, and D [32]. This signifies that while every environment contains a 7-DOF Franka Emika Panda robot arm [35] with a parallel gripper, a desk with a sliding door and a functioning drawer, a button, a switch, and three different colored and shaped rectangular blocks, the placement of every object, except the desk and robot arm, varies between environments [32]. The teleoperated "play" data was collected in the 4 environments so that a dataset corresponds with an environment. CALVIN has two evaluation metrics: Multi-Task Language Control (MTLC) and Long-Horizon MTLC (LH-MTLC). MTLC gathers information on the model's performance during a single manipulation task. Each task is performed 10 times from different starting points to evaluate how well the model has learned multi-task language-conditioned policy generalization. In total, there are 34 different tasks, and the tasks are elaborated on in the papers Appendix A. LH-MTLC evaluates the model's ability to complete multiple language instructions in a row. The 34 tasks are treated as subgoals and valid sequences of five sequential tasks are computed for the model to solve. In total, 1000 evaluation sequences that can be achieved from a predefined initial environment state. The main difference between MTLC and LH-MTLC is that in MTLC the environment and robot are reset after every task, LH-MTLC the robot and environment are only reset after every sequence. The environment-based; trained model must compute continuous visuomotor control policies in the simulated environment with the help of onboard sensors. [32] The simulated sensor data includes RGB-D images from a static and gripper camera, proprioceptive information of the robot arm, and a vision-based sense of touch [36][32]. CALVIN's limitations are the exclusive evaluation of long-horizon language-conditioned continuous control policies and limited environment variation. Thereby the usage of the evaluation is only limited to a certain type of information gathering. [32]

3.4 Vision-language models for robotic manipulation

The latest research for language-conditioned applications in robotics has been divided into mobile robot navigation, human-robot interaction, and robot arm manipulation [20]. As elaborated in Chapter 3.1 most common industrial robot implementations use a robot

arm, and since state-of-the-art benchmarks covered in Chapter 3.3 also focus on robot arms, this chapter will focus on VLMs made specifically for robot arm manipulation. Current state-of-the-art robot manipulation VLMs, as clarified in Chapter 3.2, combine a language module and a perception module to handle sentences and perceive the surrounding environment respectively. Hence, the robotic manipulation VLMs discussed in this chapter are divided according to the different approaches of perception module implementations.

Some state-of-the-art VLMs for robot manipulation that leverage CNNs:

- Multicontext Imitation Learning (MCIL) presented by Lynch & Sermanet maps task description types to corresponding latent goal spaces [37]. These embeddings are then fed to the Multicontext Latent Motor Plan that utilizes imitation learning to train 3 networks: posterior mapping from full state-action demonstrations over recognized plans, prior learned mapping from the initial state to possible plans of reaching the goal, and a conditioned policy on goals and plans, that decodes the recognized plan and forces it to reconstruct actions that reached the goal. This module enables the MCIL model to connect language instructions with low-level action primitives as well as sequencing actions to solve long-horizon tasks. For more thorough explanations of the implementations of modules see the papers Appendices from A to F. [37]
- Hierarchical Universal Language Conditioned Policies (HULC) [38], and its successor Hierarchical Universal Language Conditioned Policies 2.0 (HULC++) [39]. HULC uses MCIL [37] as a base but improves key components [38]. Compared to MCIL, HULC adds decomposition of control into a hierarchical approach where global plans are generated with a static camera, and local policies are learned utilizing a gripper camera conditioned on the plan [38]. HULC also implements a multimodal transformer encoder which builds a contextualized representation of abstract behaviors as well as semantic alignments of video and language to address the relation of language instructions to the onboard perception and actions of robots [38]. HULC++ builds upon HULC [38] by employing Visual Affordance-guided Policy Optimization (VAPO) [40][39]. VAPO extracts visual affordances from unlabelled human teleoperated play [40]. These affordances guide model-based policies and reinforcement learning policies to efficiently learn tasks for robot manipulation [40]. With the learning policies of VAPO [40] HULC++ can sample-efficiently learn general-purpose, language-conditioned robot controls [39].

- Skill Prior based Imitation Learning (SPIL) demonstrated by Zhou *et al.* uses the same encoders as HULC for the perception module and the language module, but the action space is comprised of skill embeddings instead of cartesian action space embeddings [41][38]. In other words, where HULCs actions are a one-time step with a single 7-DOF movement, SPIL has action sequences over a horizon. This enables the model to complete tasks by choosing varying action sequences [41]. The action sequences are learned while training and the model learns to utilize multiple sequences based on observations to achieve a given goal [41].
- Language Control Diffusion (LCD) showcased by Zhang *et al.* differs from the other models by adapting a diffusion model to control high-level policies [42]. The high-level diffusion policy employs 11 billion parameter LLM T5-XXL [43] for textual encoding, and perception encoding is handled through a temporal U-Net [44], that is modified to resemble a latent diffusion model [42]. The goals generated by the high-level diffusion policy are then fed to HULCs architecture and encoders, that direct the model's low-level policy [38][42]. This is done since diffusion models tend to be inaccurate at control when high-dimensional image spaces are used [42].

A few state-of-the-art VLMs for robot manipulation that utilize ViT:

- Perceiver-Actor (PerAct) introduced by Shridhar *et al.* [45] learns to imitate 6-DOF manipulation tasks by using CLIP's language encoder [10] and 3D voxel observation patches flattened into sequences of voxel encodings. The combination of the encodings produces input sequences for the Perceiver Transformer. The perceiver transformer generates predictions for discretized actions based on the sequence's features. These predictions are then used for rotation and gripper control. [45]
- PaLM-E by Driess *et al.* [46] combines a 540 billion parameter PaLM LLM [47], and a 22 billion parameter ViT [48] to create a 562 billion parameter VLM that can solve e.g. zero-shot multimodal chain-of-though reasoning and few-shot prompting. When integrated into a control loop with multimodal sentences PaLM-E can understand the high-level policies that control and sequence the low-level policies [46].

Couple state-of-art VLMs for robot manipulation with other types of approaches:

- Robot Transformer 2 (RT-2) showcased by Brohan *et al.* as Google DeepMind's latest research [49], uses either a 55 billion parameter PaLI-X [50] model or a 12

billion parameter PaLM-E [46] model for the language module as well as perception module. It combines the physical motions of robot data with interpreting images and text learned from web data to derive robotic policies [49]. This combination enables the model to generalize into novel objects and instructions that are semantically varied. The action space for the robotic policies consists of a robot end-effector with 6-DOF positional and rotational displacement as well as the robot grippers' level of extension. [49]

- CLIPort presented by Shridhar *et al.* [51] uses CLIP [10] for both the language module and the perception module. CLIP [10] provides the model with visual encoding and sentence encoding that are passed through a Transporter [29] to produce a semantic stream and a spatial stream [51]. The fusion of these streams is embedded with goal encoding to produce an end-to-end imitation-learning model that can generate either single-task or multi-task language-conditioned policies depending on the provided training data [51].

4. EXPERIMENT SETUP

4.1 CALVIN

CALVIN [32] was chosen as the benchmark for the experimentation due to it being an actively used, open-source project with easy-to-understand concepts and simple documentation [52]. Out of the 4 possible manipulation environments [32] only environment D was used because of memory constraints elaborated on in the upcoming subchapter 4.3. Instead of training a model from scratch with the downloaded dataset, a version of an MCIL [37] model was used. Mees, *et al.* developed their own MCIL model for the baseline measurements of CALVIN following the implementation of the previously mentioned MCIL model created by Lynch & Sermanet [32][37]. Performance evaluations executed on the MCIL model during the experiment included both available evaluations, MTLC and LH-MTLC.

4.2 Multicontext Imitation Learning model

The MCIL model by Mees, *et al.* was chosen as the vision-language model to experiment on because as stated earlier it is used as a baseline measurement for CALVIN benchmarking performances [32]. Another reason is that the developers of CALVIN also offer the MCIL model as a pre-trained version with the training executed using only static RGB images on the corresponding dataset to environment D [52]. According to the CALVIN documentation, the MCIL model is also the only model that can run the MTLC evaluation without modifications to the code [52]. The same combability is also assured for LH-MTLC evaluation.

4.3 System requirements and software requirements

Running the MCIL model on the CALVIN benchmark was conducted on a personal computer (PC) running the Windows 11 operating system (OS). The PC's hardware included an Nvidia GeForce RTX 3060 [53] graphics processing unit (GPU) and an AMD Ryzen 5 5600X 6-Core [54] central processing unit (CPU). CALVIN relies on a software called TACTO for high-resolution vision-based tactile sensor simulation [36][55], the developers of TACTO recommend running Ubuntu as the OS [55]. Following this recommendation, the possibility of running into more conflicts with Windows further down the line warranted a switch to Ubuntu-based OS.

Because the code running CALVIN, according to one of the main developers Hermann, was only tested with Nvidia GPUs [52] direct access to the PC's GPU was necessary. Thus, Windows Subsystem for Linux version 2 (WSL 2) [56] and Nvidia's CUDA platform [57] were necessary to work around both the issue of using Ubuntu OS inside Windows and to get accelerated GPU computing technology to work with WSL 2. The use of WSL 2 was chosen instead of dual booting with separate OSs due to limited knowledge of the implementation and the fact that if done wrong it could mess with the PC's core functionalities. This workaround made running the CALVIN benchmark evaluation and rendering the graphical user interface (GUI) simulation possible.

The PC must also have at least 400 gigabytes (GB) of storage available to successfully download and extract both the pre-trained MCIL model, and the dataset related to environment D [32][52]. Due to the PC's primary storage capacity being constrained, a Toshiba external hard drive with 1 terabyte (TB) of storage capacity was utilized for file transferring [58].

5. EXPERIMENT AND RESULTS

5.1 CALVIN output examples

Figure 2 presents the command-line interface output of a finished MTLC evaluation with the optional debug argument [52] to print out more information on the evaluation. The output shown in Figure 2 includes a red “F and a green “S” for failure and success respectively. There are ten indicators, one for each repetition of the task. The indicators are followed by the task to complete, and lastly, it shows the number of successful executions out of ten. At the bottom of Figure 2 is the total success rate when every repetition of different tasks is combined. In total the output includes 34 tasks, but Figure 2 is only a snippet of the output.

Figure 2: Snippet of a command-line interface output for a finished MTLC evaluation.

The output of a finished LH-MTLC evaluation is shown in Figure 3. The optional debug argument [52] was again used akin to Figure 2. Going from top to bottom first in Figure 3 is the information for two sequences evaluated and at which subtask the model failed. This sequence output is generated for every sequence run during the evaluation. In Figure 3 after the sequence prints is the average length of subtasks completed in a row, and after that, there is a more detailed breakdown for the percentage of success for any task in a row. The last output in Figure 3 is a piece of the list of each task that ran during the sequences, how many times the task was executed, and what was the number of successes out of those executions.

```
Evaluating sequence: move_slider_left -> turn_on_lightbulb -> open_drawer -> r
otate_blue_block_right -> lift_blue_block_table
Subtask: move_slider_left fail

Evaluating sequence: move_slider_right -> turn_on_lightbulb -> push_pink_block
_left -> open_drawer -> push_into_drawer
Subtask: move_slider_right fail Results for Epoch 0:
Average successful sequence length: 0.33
Success rates for i instructions in a row:
1: 27.3%
2: 5.2%
3: 0.5%
4: 0.0%
5: 0.0%
turn_off_led: 60 / 65 |  SR: 92.3%
open_drawer: 94 / 154 |  SR: 61.0%
push_pink_block_right: 14 / 38 |  SR: 36.8%
rotate_blue_block_left: 3 / 29 |  SR: 10.3%
```

Figure 3: Part of the command-line interface output for a finished LH-MTLC evaluation.

Figure 4 shows a freeze-frame of the simulated environment shown in the GUI. The environment is rendered concurrently with both evaluations so that the user can see how the model moves the robot arm when given a task to complete. The text at the bottom of Figure 4 describes the current evaluation task in progress, as well as the x and y coordinates, and RGB data of the texture under the user's cursor.

Figure 4: Freeze-frame of the simulated environment in use during the evaluation. This pop-up window is a part of the GUI of CALVIN.

The simulated environment in Figure 4 is only shown to the user if the optional debug argument [52] is in use while running the evaluations. The user can only see a less detailed command-line interface output if the evaluation is performed without the debug argument.

5.2 Performance of Multi-Context Imitation Learning model

This chapter details the performance results of the pre-trained Mees *et al.* MCIL [32] model when evaluated with the CALVIN benchmark. Table 1 presents the performance results of the MCIL model with MTLC evaluation. The MTLC evaluation was executed four times with the same command detailed in the documentation of CALVIN [52]. The repetitions were made to ascertain the consistency of the model's performance.

Table 1: results of the MTLC evaluation.

Run	1.	2.	3.	4.
Total success ratio (%)	41,3	41,3	41,3	41,3

The first row of Table 1 includes the enumeration for each execution of the MTLC evaluation. The second row of Table 1 has the total success ratio in percentage for successful executions out of the 340 executions. The number 340 comes from the fact that there are 34 tasks, and each one is repeated 10 times. In Table 1 the success ratio remains the same in every execution, thus the model's performance is deemed consistent.

Table 2 showcases the performance results of the pre-trained Mees *et al.* MCIL [32] model with LH-MTLC evaluation. The LH-MTLC evaluation was run four times by utilizing the same command given in the documentation of CALVIN [52]. As was the case with MTLC evaluation, the evaluation was repeated to confirm the consistency of the model's performance.

Table 2: results of the LH-MTLC evaluation.

	*Success rates for n*umber *of instructions in a row (%)*					
Run	1:	2:	3:	4:	5:	Avg. sequence length
1.	27,3	5,2	0,5	0,0	0,0	0,33
2.	27,3	5,2	0,5	0,0	0,0	0,33
3.	27,3	5,2	0,5	0,0	0,0	0,33
4.	27,3	5,2	0,5	0,0	0,0	0,33

As can be seen from Table 2, the consistency of the model's performance did not vary during the evaluations. In Table 2, underneath the run is the indicator for each evaluation execution. After the run column, there are the success rates for each of the five possible sequential tasks described in percentage of success. The last column in Table 2 includes the average sequence length which is counted as successful subtask executions. The maximum amount achievable for the average sequence length is five.

5.3 Performance of other models

The creators of CALVIN maintain a website [59] where they host a leaderboard for open-source models that outperform the baselines set by MCIL utilizing the same static RGB imaging as well as gripper RGB-(D) images [32]. The performance data is taken directly from the models' research papers. At the time of referencing there are three leaderboards that are categorized by the properties of the training environment and the testing environment [59]. As stated in Chapter 4.1 the pre-trained MCIL model executed during the experiment was trained on environment D and evaluated on the same environment, thus the "Train D → Test D" leaderboard [59] will work as the basis for the information on other VLMs performance. As of the time of this citation the open-source VLMs that outperform the benchmark results gathered during the experimentation with MCIL [32], as shown in Table 1 and Table 2, according to the leaderboard [59], are HULC [38], its successor HULC++ [39], SPIL [41], and LCD [42].

The LH-MTLC performance metrics of HULC [38], SPIL [41], HULC++ [39], and LCD [42] are presented in Table 3 in a similar format as shown in Table 2. In Table 3, each model

is enumerated into rows. LCD [42] also has MTLC evaluations which are presented with an extra column depicting the success ratio.

Table 3: results of other models' evaluations with CALVIN.

	MTLC (%)	*Success rates for n*umber *of instructions in a row (%)*					
VLM		1:	2:	3:	4:	5:	Avg. sequence length
HULC [38]	-	82,7	64,9	50,4	38,5	28,3	2,64
SPIL [41]	-	84,6	65,1	50,8	38,0	28,6	2,67
HULC++ [39]	-	93	79	64	52	40	3,3
LCD [42]	74,01	88,7	69,9	54,5	42,7	32,2	2,88

The performance metrics of other VLMs mentioned in chapter 3.4 that utilize different benchmark software or have implemented their own benchmarks, cannot be directly compared to the VLM performance evaluations gathered with the help of CALVIN. These metrics are still important because they help with deriving the progress state of robotic control via VLMs in general. For example, Google's latest 12 billion and 55 billion parameter RT-2 models' performance is measured with their own quantitative evaluations that include seen tasks, unseen objects, unseen backgrounds, and unseen environments [49]. The 12 billion RT-2 model successfully executes seen tasks at 93 % and the average for tasks including unseen variables is 62 % [49]. CLIPort by Shridhar *et al*. also utilizes self-developed benchmarks and even real-robot experiments for performance evaluations [51]. These evaluations are elaborated on in the papers Table 1 and Table 2 [51]. Summarizing the results, CLIPort achieves approximately 70 % accuracy in simple block manipulation tasks using a real robot arm [51]. The simulated results vary drastically depending on the amount of used training demonstrations, but the worst accuracy of a task for seen variables is a sequential assembly of kits at 11,4 %. The worst-performing task when unseen elements are included is the sequential stacking of a block pyramid at 22,2 %. The best-performing task with seen elements is the putting of blocks into bowls at 100 %, while the best accuracy for a task with unseen variables is packing objects by a defined group at 80,3 %. [51]

5.4 Deriving differences from results

The benchmark results of MCIL [32] trained with only static RGB imagery seen in Table 1 and Table 2 show that imitation learning over unstructured data from restricted input does not generate an accurate policy for robot arm manipulation. HULC [38], which improves MCIL [37] by e.g. decomposing robot learning into a hierarchical approach, discrete latent planning, and a multimodal transfer encoder, enhances the subtask completion length to eight times greater. SPIL [41] has a minor upgrade to average length when compared to HULC [38]. This is probably because manipulation is trained over action sequences. The more advanced models such as RT-2 [49], which utilizes billion parameter VLMs for policy generation, perform better because the large-scale knowledge combines well with robotic data.

Even though all the performances shown in Chapter 5.3 cannot be directly compared, there is a trend of increasing accuracy when the architecture of the models advances. To summarize the architectural advancements of models such as optimization, refinery, vast parametrization of used AI models in modules, and the number of variables learned during training seem to result in a more precise robotic manipulation.

6. IMPLEMENTATION IN PRACTICE

Current real-world industrial robot control systems use controllers with low computational power that still enable the robot to perform robust movements with extreme precision, and continuous feedback. Most commonly these computation units are either industrial PCs or programmable logic controllers (PLC). [60] The VLMs explored in this thesis rely on high-performing GPUs to enable the massive calculations needed for decision-making [41][42][45]. Table 4 highlights the requirements of running or training a robotic control VLM. In Table 4, the rows contain info in the following order: the model, training or running, and the GPU requirement for the previous.

Table 4: robotic control VLMs, and the GPU(s) required for training or running.

LCD [42]	Running	Nvidia Titan RTX [61] or Nvidia A10 [62]
SPIL [41]	Running	2 Nvidia Tesla V100 16 GB [63]
PerAct [45]	Training	8 Nvidia Tesla P100 [64]

As of writing the current industrial robot system implementations utilizing industrial PCs or PLCs cannot fulfill the running requirements shown in Table 4 [60], thus integrating a VLM into an already existing system would require a total overhaul. Combining this cost with the fact that as of this writing, state-of-the-art VLMs do not provide nearly 100 % accurate execution performances, as seen in Chapter 5, is detrimental to the potential uses in industrial applications. For a cost example, as stated by Wei *et al.* [65] even small errors due to degradation in manufacturing machines can be incredibly costly to the manufacturer. Deriving from this result, a VLM with less than 100 % performance accuracy would cause even more costs due to faulty movements.

Another major issue with integrating VLMs with industrial robotics is that the majority of the AI-based robotic control research is done with cobots [66]. All the VLMs [38][39][41][42][45][49] with real-world experiments and benchmarks [30][31][32] elaborated on in this thesis use cobots such as Franka Emika Panda [35]. This means that the research progress achieved with VLM-controlled cobots might not be completely adaptable to industrial robots.

7. CONCLUSIONS

While current state-of-the-art VLMs are capable of controlling robotic manipulation when combined with a control module, there are still multiple limitations that hinder their usability in the real world. These limitations stem from the unreliable execution accuracy and the toughness of integration when it comes to real-world systems. Even though these limitations especially affect robots as well as cobots in industrial settings, the current capabilities of VLMs could be helpful in less demanding environments such as robots created for simple household tasks or mobile robot movements.

Future directions of research could focus on making VLM based, and other robot manipulation models more accurate. As stated earlier and as of writing the accuracy of VLM robotic manipulation is not up to par with the rigorous efficiency and reliability requirements of continuous tasks in industrial environments. Even though there has been significant progress in capabilities relating to generalization and object detection, these abilities without accuracy are more suited for applications outside the industrial setting. AI models for controlled robot manipulation also require a lot of computational power and robotic data. While data gathering should not be an issue, especially if robotic systems are already in place, the computational costs could bottleneck potential investments. Thus, future research could explore making the calculation algorithms required for robotic control more lightweight and efficient. More research is also needed to create standardized open-source benchmarks that produce reliable and reproducible information on models made for robotic control in industrial robots. The main driver of these benchmarks should be to cover as much ground as possible on real-world tasks and applications from a variety of viewpoints and requirements. There also must be a shift from cobot-focused research towards industrial robot system research, but first, the control hardware used in industrial robot manipulation must become capable enough.

REFERENCES

[1] Y. Domae, Recent Trends in the Research of Industrial Robots and Future Outlook, Journal of robotics and mechatronic, vol. 31, no. 1, 2019, pp. 57–62, Available at: https://doi.org/10.20965/jrm.2019.p0057.

[2] C. Müller, World Robotics 2023 – Industrial Robots, IFR Statistical Department, VDMA Services GmbH, 2023, ISBN 978-3-8163-0760-0, Available at: https://ifr.org/img/worldrobotics/Executive_Summary_WR_Industrial_Robots_2023.pdf

[3] Robots and robotic devices. Safety requirements for industrial robots. Part 1: Robots, International Organization for Standardization, SFS-EN ISO 10218-1, 2011.

[4] Robots and robotic devices. Safety requirements for industrial robots. Part 2: Robot systems and integration, International Organization for Standardization, SFS-EN ISO 10218-2, 2011.

[5] L. Wang, R. Gao, J. Váncza, *et al*. Symbiotic human-robot collaborative assembly. CIRP annals 68.2, 2019, pp. 701–726. ISSN: 0007-8506.

[6] Robots and robotic devices. Collaborative robots, International Organization for Standardization, ISO/TS 15066, 2016.

[7] Growth in AI and Robotics Research Accelerates, Nature (London), vol. 610, no. 7931, 2022, pp. S9–S9, Available at: https://doi.org/10.1038/d41586-022-03210-9.

[8] A. Mogdala, M. Kalimuthu, D. Klakow, Trends in Integration of Vision and Language Research: A Survey of Tasks, Datasets, and Methods, The Journal of Artificial Intelligence Research, vol. 71, 2021, pp. 1183–1317, Available at: https://doi.org/10.1613/JAIR.1.11688

[9] S. Uppal, S. Bhagat, D. Hazarika, *et al.* Multimodal Research in Vision and Language: A Review of Current and Emerging Trends, Information Fusion, vol. 77, 2022, pp. 149-171, Available at: https://doi.org/10.1016/j.inffus.2021.07.009.

[10] A. Radford, J. W. Kim, C. Hallacy, *et al.* Learning Transferable Visual Models From Natural Language Supervision, Proceedings of Machine Learning Research, vol. 139, 2021, pp. 8748–8763, Available at: https://doi.org/10.48550/arXiv.2103.00020

[11] Robotics. Vocabulary, International Organization for Standardization, ISO 8373, 2021.

[12] OSHA Technical Manual, Section IV: Chapter 4, Industrial Robot Systems and Industrial Robot System Safety, Occupational Safety and Health Administration, Last updated: 04.2022, Available at: https://www.osha.gov/otm/section-4-safety-hazards/chapter-4#structure

[13] A. Marwan, M. Simic, F. Imad, Calibration method for articulated industrial robots, Procedia Computer Science, vol. 112, 2017, pp. 1601–1610, Available at: https://doi.org/10.1016/j.procs.2017.08.246.

[14] M. Wilson, Implementation of robot systems: an introduction to robotics, automation, and successful systems integration in manufacturing, First edition, 2015, London, England: Elsevier, ISBN: 0-12-404749-1.

[15] S. He, C. Yan, Y. Deng, *et al.* A Tolerance Constrained G2 Continuous Path Smoothing and Interpolation Method for Industrial SCARA Robots, Robotics and Computer-Integrated Manufacturing, vol. 63, 2020, pp. 101907-, Available at: https://doi.org/10.1016/j.rcim.2019.101907.

[16] J. Barnett, M. Duke, C. K. Au, *et al.* Work Distribution of Multiple Cartesian Robot Arms for Kiwifruit Harvesting, Computers and Electronics in Agriculture, vol. 169, 2020, pp. 105202-, Available at: https://doi.org/10.1016/j.compag.2019.105202.

[17] S. Qian, B. Zi, WW. Shang, *et al.* A Review on Cable-driven Parallel Robots, Chinese Journal of Mechanical Engineering, vol. 31, 2018, Article 66, Available at: https://doi.org/10.1186/s10033-018-0267-9.

[18] A. Loganathan & N.S. Ahmad, A Systematic Review on Recent Advances in Autonomous Mobile Robot Navigation, Engineering Science and Technology, an International Journal, vol. 40, 2023, pp. 101343-, Available at: https://doi.org/10.1016/j.jestch.2023.101343.

[19] Aethon, Mobile Robots for Healthcare, 2018, URL: https://aethon.com/mobile-robots-for-healthcare/, Accessed 13.3.2024.

[20] H. Zhou, X. Yao, Y. Meng, *et al.* Language-Conditioned Learning for Robotic Manipulation: A Survey. arXiv.Org, 2024, Available at: https://doi.org/10.48550/arxiv.2312.10807.

[21] L. Shao, T. Migimatsu, Q. Zhang, *et al.* Concept2Robot: Learning Manipulation Concepts from Instructions and Human Demonstrations, The International Journal of Robotics Research, vol. 40, no. 12–14, 2021, pp. 1419–1434, Available at: https://doi.org/10.1177/02783649211046285.

[22] J. Devlin, M. W. Chang, K. Lee, *et al.* BERT: Pre-Training of Deep Bidirectional Transformers for Language Understanding, NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference, vol. 1, 2019, pp. 4171–4186.

[23] K. Rana, J. Haviland, S. Garg, *et al.* SayPlan: Grounding Large Language Models Using 3D Scene Graphs for Scalable Robot Task Planning, arXiv.Org, 2023, Available at: https://doi.org/10.48550/arxiv.2307.06135.

[24] J. Achiam, S. Adler, S. Agarwal, *et al.* GPT-4 Technical Report, arXiv.Org, 2024, Available at: https://doi.org/10.48550/arxiv.2303.08774.

[25] K. He, X. Zhang, S. Ren, *et al.* Deep Residual Learning for Image Recognition, 2016 IEEE Conference on Computer Vision and Pattern Recognition, vol. 2016-, IEEE, 2016, pp. 770–778, Available at: https://doi.org/10.1109/CVPR.2016.90.

[26] Y. Bai, J. Mei, A. Yuille, *et al.* Are Transformers More Robust Than CNNs?, Advances in Neural Information Processing Systems, vol. 32, 2021, pp. 26831–26843.

[27] A. Dosovitskiy, L. Beyer, A. Kolesnikov, *et al.* An Image Is Worth 16x16 Words: Transformers for Image Recognition at Scale, arXiv.Org, 2021, Available at: https://doi.org/10.48550/arxiv.2010.11929.

[28] Z. Tianhao, Z. McCarthy, O. Jow, *et al.* Deep Imitation Learning for Complex Manipulation Tasks from Virtual Reality Teleoperation, 2018 IEEE International Conference on Robotics and Automation, 2018, pp. 5628–5635, Available at: https://doi.org/10.1109/ICRA.2018.8461249.

[29] A. Zeng, P. Florence, J. Tompson, *et al.* Transporter Networks: Rearranging the Visual World for Robotic Manipulation, Proceedings of Machine Learning Research, vol. 155, 2020, pp. 726–747.

[30] Y. Jiang, A. Gupta, Z. Zhang, *et al.* VIMA: General Robot Manipulation with Multimodal Prompts, arXiv.Org, 2023, Available at: https://doi.org/10.48550/arxiv.2210.03094.

[31] K. Zheng, X. Chen, O. C. Jenkins, *et al.* VLMbench: A Compositional Benchmark for Vision-and-Language Manipulation, arXiv.Org, 2022, Available at: https://doi.org/10.48550/arxiv.2206.08522.

[32] O. Mees, L. Hermann, E. Rosete-Beas, *et al.* CALVIN: A Benchmark for Language-Conditioned Policy Learning for Long-Horizon Robot Manipulation Tasks, IEEE Robotics and Automation Letters, vol. 7, no. 3, 2022, pp. 7327–7334, Available at: https://doi.org/10.1109/LRA.2022.3180108.

[33] Universal Robots A/S, Products & Services – UR5e, 2024, URL: https://www.universal-robots.com/products/ur5-robot, Accessed: 15.3.2024.

[34] E. Rohmer, S. P. N. Singh, M. Freese, V-REP: A Versatile and Scalable Robot Simulation Framework, 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, 2013, pp. 1321–1326, Available at: https://doi.org/10.1109/IROS.2013.6696520.

[35] Franka Emika GmbH, Product Manual Franka Emika Robot [EN], PDF-document, 2021, Available at: https://download.franka.de/documents/100010_Product%20Manual%20Franka%20Emika%20Robot_10.21_EN.pdf.

[36] S. Wang, M. Lambeta, P.-W. Chou, *et al.* Tacto: A fast,flexible and open-source simulator for high-resolution vision-based tactile sensors, IEEE Robotics and Automation Letters, vol. 7, no. 2, 2022, pp. 3930-3937.

[37] C. Lynch & P. Sermanet, Language Conditioned Imitation Learning over Unstructured Data, arXiv.Org, 2021, Available at: https://doi.org/10.48550/arxiv.2005.07648.

[38] O. Mees, L. Hermann, W. Burgard, What Matters in Language Conditioned Robotic Imitation Learning Over Unstructured Data, IEEE Robotics and Automation Letters, vol. 7, no. 4, 2022, pp. 11205–11212, Available at: https://doi.org/10.1109/LRA.2022.3196123.

[39] O. Mees, J. Borja-Diaz, W. Burgard, Grounding Language with Visual Affordances over Unstructured Data, 2023 IEEE International Conference on Robotics and Automation, vol. 2023-, 2023, pp. 11576–11582, Available at: https://doi.org/10.1109/ICRA48891.2023.10160396.

[40] J. Borja-Diaz, O. Mees, G. Kalweit, *et al.* Affordance Learning from Play for Sample-Efficient Policy Learning, 2022 International Conference on Robotics and Automation, 2022, pp. 6372–6378, Available at: https://doi.org/10.1109/ICRA46639.2022.9811889.

[41] H. Zhou, B. Zhensan, X. Yao, *et al.* Language-Conditioned Imitation Learning with Base Skill Priors under Unstructured Data, arXiv.Org, 2024, Available at: https://doi.org/10.48550/arxiv.2305.19075.

[42] E. Zhang, Y. Lu, W. Wang, *et al.* Language Control Diffusion: Efficiently Scaling through Space, Time, and Tasks, arXiv.Org, 2024, Available at: https://doi.org/10.48550/arxiv.2210.15629.

[43] C. Raffel, N. Shazeer, A. Roberts, *et al.* Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer, Journal of Machine Learning Research, vol. 21, 2020, pp. 5485–5551.

[44] M. Janner, Y. Du, J. B. Tenenbaum, *et al*. Planning with Diffusion for Flexible Behavior Synthesis, arXiv.Org, 2022, Available at: https://doi.org/10.48550/arxiv.2205.09991.

[45] M. Shridhar, L. Manuelli, D. Fox, Perceiver-Actor: A Multi-Task Transformer for Robotic Manipulation, arXiv.Org, 2022, Available at: https://doi.org/10.48550/arxiv.2209.05451.

[46] D. Driess, F. Xia, M. S. M. Sajjadi, *et al.* PaLM-E: An Embodied Multimodal Language Model, arXiv.Org, 2023, Available at: https://doi.org/10.48550/arxiv.2303.03378.

[47] A. Chowdhery, S. Narang, J. Devlin, *et al.* PaLM: Scaling Language Modeling with Pathways, arXiv.Org, 2022, Available at: https://doi.org/10.48550/arxiv.2204.02311.

[48] M. Dehghani, J. Djolonga, B. Mustafa, *et al.* Scaling Vision Transformers to 22 Billion Parameters, arXiv.Org, 2023, Available at: https://doi.org/10.48550/arxiv.2302.05442.

[49] A. Brohan, N. Brown, J. Carbajal, *et al.* RT-2: Vision-Language-Action Models Transfer Web Knowledge to Robotic Control, arXiv.Org, 2023, Available at: https://doi.org/10.48550/arxiv.2307.15818.

[50] X. Chen, J. Djolonga, P. Padlewski, *et al.* PaLI-X: On Scaling up a Multilingual Vision and Language Model, arXiv.Org, 2023, Available at: https://doi.org/10.48550/arxiv.2305.18565.

[51] M. Shridhar, L. Manuelli, D. Fox, CLIPort: What and Where Pathways for Robotic Manipulation, arXiv.Org, 2021, Available at: https://doi.org/10.48550/arxiv.2109.12098.

[52] CALVIN GitHub page, 2024, URL: https://github.com/mees/calvin, Accessed: 1.4.2024.

[53] NVIDIA Corporation, Marketing page for GeForce RTX 3060 product series, URL: https://www.nvidia.com/fi-fi/geforce/graphics-cards/30-series/rtx-3060-3060ti/, Accessed: 2.4.2024.

[54] AMD, Marketing page for AMD Ryzen 5 5600X Desktop Processors, URL: https://www.amd.com/en/products/processors/desktops/ryzen/amd-ryzen-5-5600x.html, Accessed: 4.4.2024.

[55] TACTO GitHub page, 2024, URL: https://github.com/facebookresearch/tacto, Accessed: 9.4.2024.

[56] Microsoft, Windows Subsystem for Linux Documentation, Article, 2022, Last updated: 27.6.2022, URL: https://learn.microsoft.com/en-us/windows/wsl/, Accessed: 9.4.2024.

[57] NVIDIA Corporation, NVIDIA Developer: CUDA Toolkit webpage, 2024, URL: https://developer.nvidia.com/cuda-toolkit, Accessed: 9.4.2024.

[58] Toshiba, Marketing page for External Hard Disk Drives, URL: https://storage.toshiba.com/consumer-hdd/external, Accessed: 9.4.2024.

[59] Autonomous Intelligent Systems Lab; University of Freidburg, Calvin: A Benchmark for Language-conditioned Policy Learning for Long-horizon Robot Manipulation Tasks, 2024, URL: http://calvin.cs.uni-freiburg.de, Accessed: 10.4.2024.

[60] P. Bilancia, J. Schmidt, R. Raffaeli, *et al.* An Overview of Industrial Robots Control and Programming Approaches, Applied Sciences, vol. 13, no. 4, 2023, pp. 2582-, Available at: https://doi.org/10.3390/app13042582.

[61] NVIDIA Corporation, Marketing page for NVIDIA Titan RTX, URL: https://www.nvidia.com/en-eu/deep-learning-ai/products/titan-rtx/, Accessed: 12.4.2024.

[62] NVIDIA Corporation, Marketing page for NVIDIA A10 Tensor Core GPU, URL: https://www.nvidia.com/en-us/data-center/products/a10-gpu/, Accessed: 12.4.2024.

[63] NVIDIA Corporation, Marketing page for NVIDIA Tesla V100, URL: https://www.nvidia.com/en-gb/data-center/tesla-v100/, Accessed: 12.4.2024.

[64] NVIDIA Corporation, Marketing page for NVIDIA Tesla P100, URL: https://www.nvidia.com/en-us/data-center/tesla-p100/, Accessed: 12.4.2024.

[65] S. Wei, M. Nourelfath, N. Nahas, Analysis of a Production Line Subject to Degradation and Preventive Maintenance, Reliability Engineering & System Safety, vol. 230, 2023, pp. 108906-, Available at: https://doi.org/10.1016/j.ress.2022.108906.

[66] A. Borboni, K. V. V. Reddy, I. Elamvazuthi, *et al.* The Expanding Role of Artificial Intelligence in Collaborative Robots for Industrial Applications: A Systematic Review of Recent Works, Machines (Basel), vol. 11, no. 1, 2023, pp. 111-, Available at: https://doi.org/10.3390/machines11010111.