Robotic pouring with self–supervised learning from video demonstration

Robotergesteuertes Gießen mit selbstüberwachtem Lernen aus Videodemonstrationen
Master thesis by Yasemin Göksu
Date of submission: December 30, 2024

1. Review: Alap Kshirsagar
2. Review: Oliver Hahn
Darmstadt

Erklärung zur Abschlussarbeit gemäß § 22 Abs. 7 APB TU Darmstadt

Hiermit erkläre ich, Yasemin Göksu, dass ich die vorliegende Arbeit gemäß § 22 Abs. 7 APB der TU Darmstadt selbtständig, ohne Hilfe Dritter und nur mit den angegebenen Quellen und Hilfsmitteln angefertigt habe Ich habe mit Ausnahme der zitierten Literatur und anderer in der Arbeit genannter Quellen keine fremden Hilfsmittel benutzt. Die von mir bei der Anfertigung dieser wissenschaftlichen Arbeit wörtlich oder inhaltlich benutzte Literatur und alle anderen Quellen habe ich im Text deutlich gekennzeichnet und gesondert aufgeführt. Dies gilt auch für Quellen oder Hilfsmittel aus dem Internet.

Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Mir ist bekannt, dass im Falle eines Plagiats (§ 38 Abs. 2 APB) ein Täuschungsversuch vorliegt, der dazu führt, dass die Arbeit mit 5,0 bewertet und damit ein Prüfungsversuch verbraucht wird. Abschlussarbeiten dürfen nur einmal wiederholt werden.

Bei einer Thesis des Fachbereichs Architektur entspricht die eingereichte elektronische Fassung dem vorgestellten Modell und den vorgelegten Plänen.

Darmstadt, 30. Dezember 2024

[illegible]

Yasemin Göksu

Abstract

This thesis explores the development of a self–supervised robotic framework for replicating human pouring tasks. The framework was implemented using CLIP–DINOiser [82] for object segmentation and minimal enclosing cylinders for geometric abstraction. The framework combines these methods with insights from a comprehensive literature review to enable a robot to learn and execute pouring motions with precision.

The improved framework was designed to capture and replicate demonstration trajectories, employing connected components analysis and advanced alignment techniques to ensure accurate motion replication. A human demonstration was used to validate the system's ability to pour materials, successfully showcasing the transfer of objects between containers.

This work contributes to the field of robotic learning by demonstrating a novel integration of vision–based and geometric techniques for dynamic task execution.

Zusammenfassung

Diese Arbeit untersucht die Entwicklung eines selbstüberwachten robotischen Frameworks zur Nachbildung menschlicher Eingießaufgaben. Das Framework wurde unter Verwendung von CLIP-DINOiser [82] für die Objekterkennung und minimalen einschließenden Zylindern für die geometrische Abstraktion implementiert. Es kombiniert diese Methoden mit Erkenntnissen aus einer umfassenden Literaturrecherche, um einem Roboter präzises Lernen und die Ausführung von Eingießbewegungen zu ermöglichen.

Das verbesserte Framework wurde entwickelt, um Demonstrationsbewegungen zu erfassen und nachzubilden, wobei eine Analyse zusammenhängender Komponenten und fortgeschrittene Ausrichtungstechniken angewandt wurden, um eine exakte Bewegungswiedergabe sicherzustellen. Eine menschliche Demonstration wurde genutzt, um die Fähigkeit des Systems zur Materialübertragung zu validieren, und zeigte dabei erfolgreich das Einschütten von Objekten zwischen Behältern.

Diese Arbeit leistet einen Beitrag zum Bereich des robotischen Lernens, indem sie eine neuartige Integration von visuellen und geometrischen Techniken für die Ausführung dynamischer Aufgaben demonstriert.

Contents

1 Introduction

Pouring is a fundamental but challenging task with applications in cooking, industrial automation, and healthcare. The goal of this thesis is to better understand what the current state–of–the–art algorithms for robot pouring do and to further explore self-supervised learning from video demonstrations as a novel approach to teach robots how to pour autonomously.
The aim is to enable robots not only to replicate pouring motions but also to understand the underlying physics, allowing them to adapt to varying conditions such as container shapes, liquid viscosities, and environmental factors.

2 Motivation

The field of robotics is rapidly evolving, with a growing emphasis on developing systems that can operate autonomously in diverse and dynamic environments. One critical challenge lies in enabling robots to perform complex tasks with minimal human intervention or explicit programming. Traditional approaches often rely heavily on detailed models of the task environment, which are not only time–consuming and resource–intensive to develop but also limit the robot's ability to adapt to new situations.

Self–supervised learning presents a compelling alternative. By allowing robots to learn directly from their interactions with the environment, this paradigm leverages inherent feedback mechanisms to build abstract representations of tasks. These representations enable robots to achieve a higher degree of generalization, making them capable of performing a wide range of activities beyond the initial scope of their training. For instance, in the context of robotic pouring, a self–supervised system can autonomously learn the nuances of liquid dynamics, container alignment, and flow control without requiring explicit models or predefined labels.

The ability to generalize self–supervised learning techniques to other tasks is equally transformative. A system trained for pouring can adapt its learned abstractions to related tasks, such as transferring, mixing, or distributing materials, showcasing the versatility of the approach. Moreover, eliminating the need for precise task modeling not only reduces development complexity but also makes robotic solutions more scalable and accessible across various domains.

By focusing on self–supervised robotic pouring, this research aims to demonstrate the power of self–supervised tasks in fostering abstraction, adaptability, and scalability in robotic systems. This approach paves the way for a new generation of robots capable of performing intricate tasks with greater autonomy and efficiency, contributing significantly to the advancement of intelligent robotics.

3 Literature Review

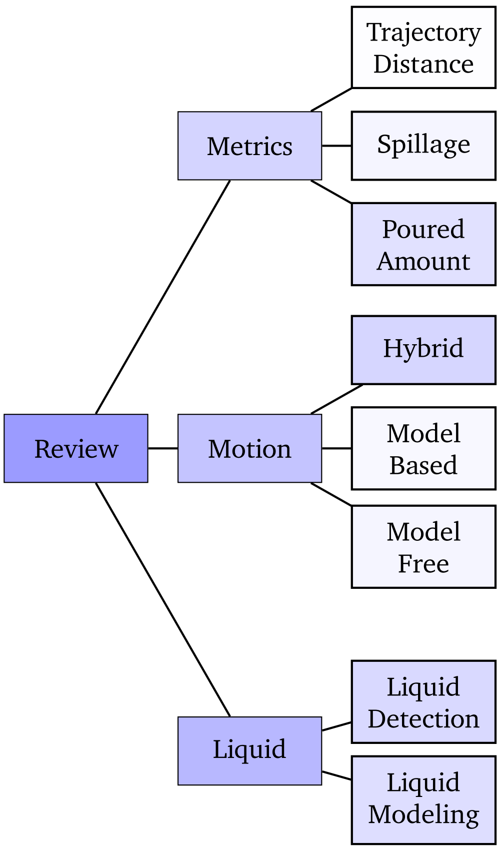


Figure 3.1: A higher intensity of the text panels provides an indication of a higher number of reviewed papers addressing the given topics.

This literature review explores the advancements and challenges in robotic pouring systems, synthesizing the findings of 39 scholarly articles published between 2003 and 2024. The reviewed papers were sourced from reputable databases including Google Scholar, arXiv, and IEEE Xplore, using the search phrase "robot pouring". The body of work spans over two decades, reflecting the evolution of robotic pouring techniques, algorithms, and applications in various domains.

In this literature review, only papers published in prominent robotics and automation conferences, namely TRO, RAL, ICRA, IROS, Humanoids, HRI, and CDC, were considered to ensure a comprehensive and high-quality analysis of relevant research.

The reviewed papers can be broadly categorized into three main groups: **liquids**, focusing on the detection and analysis of liquids in robotic pouring tasks; **motion**, addressing the generation of motion for pouring; and **metrics**, examining the evaluation criteria and metrics utilized in the reviewed studies.

3.1 Liquid

Liquid modeling and liquid detection, while both centered on the analysis and understanding of liquids, serve distinct purposes: liquid modeling involves the theoretical or computational representation of liquid properties and behaviors, whereas liquid detection focuses on identifying the presence of a liquid. The reviewed papers were categorized into these distinct subcategories to better understand the scope and focus of the existing research.

3.1.1 Liquid/Fluid/Partical Modeling

In the article from Tuomainen, Blanco-Mulero, and Kyrki [76], they suggest modeling the dynamics of the material's interaction with rigid bodies that manipulate it using a graph–based representation. They represent the particle interactions through message–passing using a graph neural network (GNN). They suggest minimizing the Wasserstein distance between a predicted granular particle distribution and their intended configuration to plan manipulation routes. They show that in both simulated and real–world situations, the suggested technique can flow granular materials into the required configuration.
The work of Matl et al. [49] describes a framework that uses real–world depth images of granular formations to deduce material properties, automatically calibrating a fast physics simulator to properly model granular materials. Using likelihood–free Bayesian inference, coefficients of sliding friction, rolling friction, and restitution of grains are computed using grain formation summary data.
In the work from Pan and Manocha [60], they introduce a new motion planning method for pouring a liquid body from a source to a target container. They employ a neural network to infer a set of important liquid–related characteristics from the observation of the current liquid configuration in order to handle liquid dynamics without the need for fluid simulations. They use stochastic optimization in a problem–specific search space to create a dataset of successful pouring samples in order to train the neural network.
In the letter from Wu and Chirikjian [80], they suggest a new way for robots to use physical simulations to "imagine" the open containability affordance of an item that hasn't been seen before. The robot uses an RGB–D camera to scan the object on its own. In order to quantify the open containability affordance, the scanned 3D model is utilized to physically simulate the dropping of particles onto an item and count the number of particles that are held in it.
Recurrent neural networks (RNNs) are used in the study of Chen, Huang, and Sun [11]

to predict water dynamics, allowing MPC to be used for pouring control.
The study by Dong et al. [20] proposes two approaches for controlling the motion of a service robot in pouring liquid from an unknown container into another unknown container without external tools. One of the approaches calculates the volume using the relation between the container's angle and volume.
PourNet from Babaians et al. [3] uses an Position Based Dynamics framework, to simulate the fluid [47].
The paper written by Matl, Matthew, and Bajcsy [48] proposes a method for robotic liquid pouring using haptic sensing. The robot moves a container through tilting motions, observing the shifting center of mass. This data is then analyzed using a physics–based model to estimate the liquid's mass and volume.
Do, Gordillo, and Burgard [18] use PreonLab to simulate fluids, a state–of–the–art fluid simulator developed by FIFTY2 [1].
The paper by Lopez-Guevara et al. [46] models liquid behavior by capturing real–world liquid flow data and training deep neural networks, specifically using Long Short–Term Memory (LSTM) units, to predict the dynamics of liquid drops on solid surfaces.
In the paper of Yamaguchi and Atkeson [86], 3D liquid flow is reconstructed from a stereo camera to learn dynamical models of liquid flow.

3.1.2 Liquid Detection

The authors Hanson et al. [27] describe a sensing method in their article that allows robots to identify the type of liquid present in an unfamiliar container. To accomplish this, they include Visible to Near Infrared (VNIR) reflectance spectroscopy into the end effector of a robot. They present a hierarchical model that uses spectral observations from two integrated spectrometers to infer the material classes of both containers and inside contents.
Liang et al. [43] address the difficult perception issue in robotic pouring. In order to forecast the liquid height from the audio fragment during the robotic pouring task, they suggest using audio vibration detecting and creating a deep neural network called PouringNet.
In the paper written by Kennedy et al. [37], a technique is being described to automatically dispense a precise volume of fluid without the use of precision pouring tools by utilizing solely visual feedback. The authors demonstrate a methodical technique with a hybrid control scheme that can withstand irregular flow and the starting volume of fluid in the pouring container.

By developing a new probabilistic blood flow detection algorithm and a trajectory generating technique that directs autonomous suction instruments towards blood pooling, the authors Richter et al. [68] provide the first automated hemostasis solution.
Weight and eyesight are combined in the work of Kennedy et al. [36] to detect the fluid in the clear receiving container.
Takahashi and Yonekura [74] present a technique for annotating segmentation masks with invisible markers for object manipulation is presented.
In their study, the authors Do, Schubert, and Burgard [19] provide a new probabilistic method for utilizing an RGB–D camera to estimate the fill–level of a liquid in a cup.
In [9], the authors provide a multi–sensory pouring dataset that includes two multi–sensory networks that estimate pouring rate and pouring average height, as well as human pouring demonstrations of different granular material.
A new segmentation pipeline that can separate transparent liquids, like water, from an RGB image is proposed by the authors Narasimhan et al. [53]. They employ a generative model that, when trained solely on an unpaired dataset of colored and transparent liquid images, can convert photos of colored liquids into artificially created transparent liquid images.
The study by Wilson, Sterling, and Lin [78] introduces audio–based and audio–augmented techniques using multimodal convolutional neural networks (CNNs) to estimate poured weight, detect overflow, and classify liquid and target containers. The audio–based network uses raw audio from a pouring sequence, while the audio–augmented network uses video images. This is the first use of audio–visual neural networks.
In the study of Liang et al. [44] a multimodal pouring network (MP–Net) is proposed for accurate liquid height estimation in pouring tasks for service robots. The network uses audition and haptics input and is trained on a self–collected dataset. It is robust against noise and changes to the task and environment, and can estimate the shape of the target container by combining predicted height and force data.
The study of Dong et al. [20] proposes two approaches for controlling the motion of a service robot in pouring liquid from an unknown container into another unknown container without external tools. One of those methods calculates the poured volume using a model of the target container and the height of liquid.
The study of Piacenza, Lee, and Isler [63] focuses on the use of service robots for pouring water, focusing on their ability to perform this task without environmental instrumentation. The robot uses a simple PID controller, joint torque sensors, and tactile sensors at the fingertips. The robot can accurately pour within 10 ml of the target on average, while being robust enough to pour at different locations and with different grasps.
Schenck and Fox [70] propose both a model based and a model free method utilizing deep learning for estimating the volume of liquid in a container.

PourIt [45] trains a binary classification model to distinguish the presence of liquid using image–level labels.

3.2 Motion Planning

This section of the literature review focuses on motion planning in robotic pouring, with the reviewed papers categorized into three approaches: model free, model based, and hybrid controllers. Notably, the majority of controllers in this domain appear to adopt a hybrid approach, combining elements of both model free and model based methods to achieve effective pouring strategies.

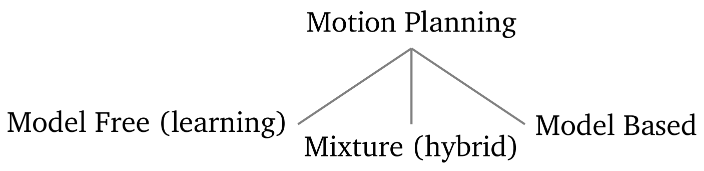


Figure 3.2: The reviewed papers were categorized into model free, model based and hybrid methods.

3.2.1 Model Based

The drop of particles in the target container is physically modeled in the publication of Wu and Chirikjian [80]. To achieve the ideal pouring position and orientation in the actual world, the robot uses physical simulations to picture pouring into the object.
In addition to calculating the volumetric flow rate, Kennedy et al. [36] estimate the maximum containable volume profile for a given angle and the fluid's time delay out of the container in their work. The intended volumetric flow rate is prescribed by a trapezoidal trajectory generation algorithm as a function of the estimation accuracy. Next, volumetric error decreases using a hybrid control technique, in which the container's angular velocity serves as the control output.
Dong et al. [20] offer two analytical methods. The target container is the main focus of Method 1. The action control makes use of a proportional derivative (PD) controller, which employs the poured volume as a control variable and the pouring container's angular speed as a process variable. The pouring container is the focus of Method 2. With just a proportional controller (P) that uses the rotation angle of the goal volume as a control variable and the angular speed of the pouring container as a process variable, Method 2's controller is straightforward.

3.2.2 Model Free

A method for creating pouring trajectories by learning from human subjects pouring motions is proposed by Huang and Sun [33]. The method determines the future pouring velocity by using force feedback from the cup.
A self–supervised method is used by Sermanet et al. [72] to learn robotic behaviors and representations solely from unlabeled videos taken from various angles. They use a metric learning loss to train their representations, which attracts numerous contemporaneous viewpoints of the same observation in the embedding space while rejecting temporal neighbors that are frequently functionally different but visually identical.
A neural network architecture is proposed by Urain, Tateo, and Peters [77] to train SVF on Lie Groups.
Time–Contrastive Networks (TCN) that learn from visual observations are extended by Dwibedi et al. [22] by jointly embedding multiple frames in the embedding space rather than a single frame. With just the learned embeddings as input, this self–supervised reinforcement learning technique uses algorithms such as Proximal Policy Optimization (PPO) to build continuous control policies.

3.2.3 Hybrid

SQUIRL, a meta–IRL technique for quick and reliable learning of long–horizon tasks, is introduced in the paper from Wu et al. [79]. SQUIRL picks up knowledge via related tasks, off–policy robot encounters, and video examples. A useful Q–functioned IRL formulation is used in the algorithm. It does not require trials during testing; instead, it uses off–policy robot encounters for training.
A method for directly incorporating such knowledge into measured features is presented by Nematollahi et al. [54]. They learn a motion encoded by a dynamic system in a Gaussian Mixture Model (GMM) representation by combining action demonstrations with external features.
By mapping points from known items to the novel ones, Brandi, Kroemer, and Peters [8] suggest a way to calculate the geometric parameters of novel things. Using probabilistic motor primitives (ProMPs) in warped environments, the resulting warped parameters are then used to learn pouring actions and task restrictions.
By monitoring the fluid height in the receiving container, Kennedy et al. [37] regulate the flow. They demonstrate a pouring model and a model based algorithm for controlling a robot arm that regulates the pouring rate using visual feedback.
The aim of Schenck and Fox [70] is visual closed–loop control for liquid pouring. They

present a real–time pouring control system that combines a PID controller with a model free approach. A deep neural network for pixel–level liquid detection, a model based approach to volume determination, a neural network for volume regression, and a ground truth for liquid detection based on thermal images are some of the main contributions.
Utilizing video demonstrations, Ramachandruni et al. [66] suggest an imitation learning framework for skill transfer from expert to robot. They employ a multi–level spatial attention module in conjunction with a visual feature representation network. They created a reinforcement learning problem for task imitation and used metric learning loss to train the network.
The manual acquisition of skills by human demonstration, automatic skill selection, and continuous parameter optimization for these skills are all examined in the work of Yamaguchi, Atkeson, and Ogasawara [83]. They used finite state machines to model skills.
Piacenza, Lee, and Isler [63] use a three–state finite state machine. By examining the first derivative of historical force data, they employ the BioTacs force sensor to identify the start of pouring during the first state. Next, a PID controller's smooth trapezoidal trajectory is produced.
In order to improve policy learning with unknown dynamics, Yamaguchi and Atkeson [84] take into consideration a temporal decomposition of dynamics. They use differential dynamic programming (DDP) to get a policy. Their DDP algorithm uses a stochastic evaluation function and is a first–order gradient descent algorithm.
A method for learning a pouring policy using Deep Deterministic Policy Gradients (DDPG) is presented by Do, Gordillo, and Burgard [18]. They employ a cutting–edge liquid simulator that makes it possible to learn about the dynamics of the liquid.
The receding horizon technique is used in [60] to present a feedback motion planning algorithm that resolves a spacetime optimization problem. A neural network provides guidance for the objective function. Numerous offline, autonomously produced, successful pouring trajectories with random configurations are used to train the neural network.
By employing RNN to estimate the water flow mechanism, Chen, Huang, and Sun [11] make it possible to use Model Predictive Control (MPC) for the task of precise pouring.
For a manipulator used to move liquid from a source to a target container, Pan and Manocha [61] construct a smooth, collision–free trajectory using an optimization–based motion planning technique. During the trajectory computation, they consider estimated (simplified) fluid dynamics constraints.
PourNet is introduced by Babaians et al. [3] as a hybrid planner. They employ deep reinforcement learning in conjunction with Nonlinear Model Predictive Control for joint planning and Proximal Policy Optimization (PPO) for end–effector planning. PourNet randomizes liquid parameters using a curriculum–based pouring scenario.
Reinforcement learning with learned dynamics and explicit planning is used by Yamaguchi

and Atkeson [85]. They employ stochastic differential dynamic programming (DDP) in neural networks.
In the work of Chen et al. [12], Visumotor policies are trained using RL frameworks. Then, given only still images of the task in the target domain, they use transfer learning to generalize to new task domains.

3.3 Metrics

The literature review revealed a diverse range of metrics employed for evaluating robotic pouring tasks. This thesis highlights a subset of these metrics.

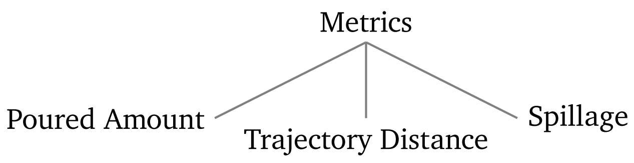


Figure 3.3: The metrics used in the reviewed papers can be categorized into the three categories: Spillage, Poured Amount and Trajectory Distance.

3.3.1 Spillage

The spillage is the amount of material that does not fall into the container while the pouring is being executed. In [63] and [18] they measured the spillage in ml, whereas in [60] they state the spillage as the fraction of material that did not land in the target container.

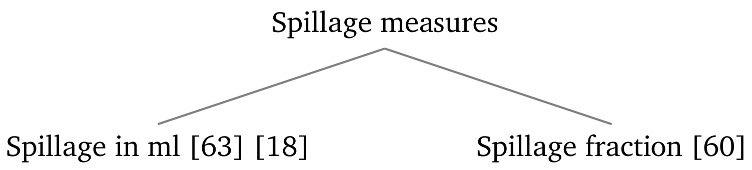


Figure 3.4: The graph further shows how the spillage measures were categorized.

3.3.2 Poured Amount

The poured amount itself can be categorized into the four categories: Rate of Successful Pours, Pouring Deviation, Pouring Curves and Target vs. Reached Amount.

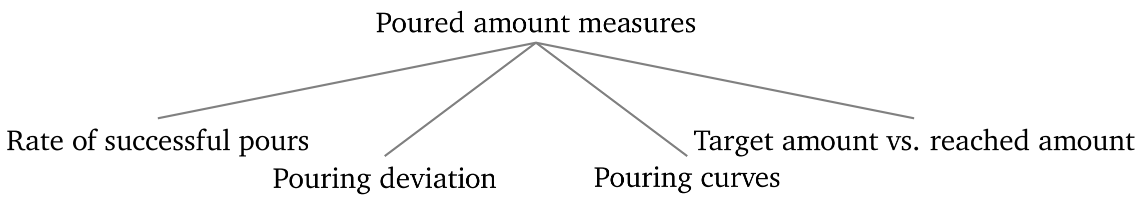


Figure 3.5: The graph shows which measures were found in the literature for poured amount measures.

Target Amount vs. Reached amount The target amount vs. reached amount compares how much the controller was supposed to fill up a container compared to how much the controller actually filled up the container.
In [18], [61] and [54], this was measured by comparing the reached vs. target percentage fillage.
In [3] they measured the reached vs. target fillage in grams, whereas in [36] and [70] they compared the reached vs. target fillage in ml.

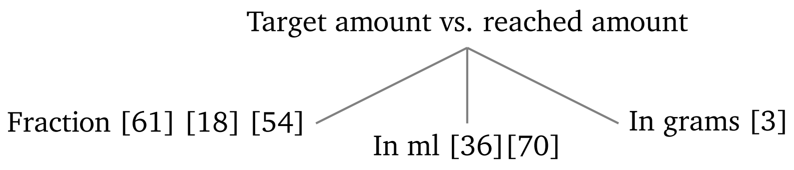


Figure 3.6: The target amount vs. reached amount were categorized into the fraction, ml and grams.

Pouring Deviation The pouring deviation or volume error measures the difference between the expected amount of material in the target container compared to how much material landed in the container.
In [3], [20], [44], [70] and [18] this was measured in ml, whereas in [3] this was given in grams.

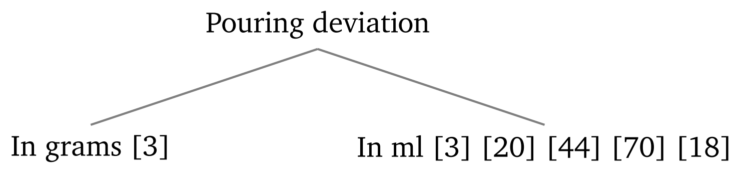


Figure 3.7: The pouring deviation was further divided into the units gram and ml.

Pouring Curves The pouring curves describe how much liquid is dispensed over time by the controller. In [63] they measured the poured weight over time. In [83] they measured the poured ratio over time.

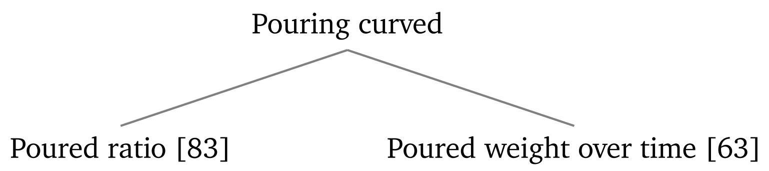


Figure 3.8: The pouring curves were further split up into the poured ratio and the poured weight over time.

Rate of Successful pours The rate of successful pours measures the rate of pours without spilling any material for a controller. This was measured in [80].

3.3.3 Function Distances/Trajectory Distances

Trajectory distances measure the distance between a test/validation trajectory and the outputted trajectory from the controller. In [33] they use dynamic time warping, which calculates the minimum normalized distance between two trajectories.

4 Theoretical Foundations

In this section, the key theoretical concepts underpinning this thesis will be explored. These foundations form the basis of the methodologies and approaches used throughout the research.

4.1 Connected Components Labeling

Connected component labeling is a crucial process for machine vision systems and various image processing applications. Its primary purpose is to identify individual connected components in a binary image and assign unique labels to them as separate objects [2][7]. The algorithm operates as follows [2]:

1. **Input:** A binary image with zero borders.
2. **Index Variable Definition:** Define an index variable (x, y) representing the image coordinates and the inner indexer.
3. **Scanning Process:**
    a) Perform a scan from left to right and then from top to bottom until an unlabeled foreground pixel p is encountered.

    b) Check the 8–connected neighbors of the pixel p:

    - **Case 1:** If all neighbors are zero pixels, mark p as an object and assign it a unique label.
    - **Case 2:** If p has only one non–zero neighboring pixel, label p with the same label as its neighbor. Repeat the search iteratively.
    - **Case 3:** If p has multiple non–zero neighbors, classify it as a branching pixel. Push its coordinates (x, y) onto a stack. Modify (x, y) to point to the first non–zero neighbor in a clockwise direction. Repeat the search iteratively.

    c) If p has no non–zero neighbors:

    i. If the stack is empty, the object is fully labeled. Return to the main scanning process in step 3(a).

    ii. If the stack is not empty, pop the last branching pixel's position from the stack and adjust (x, y) accordingly. Resume the steps for cases 1, 2, and 3 as described above.

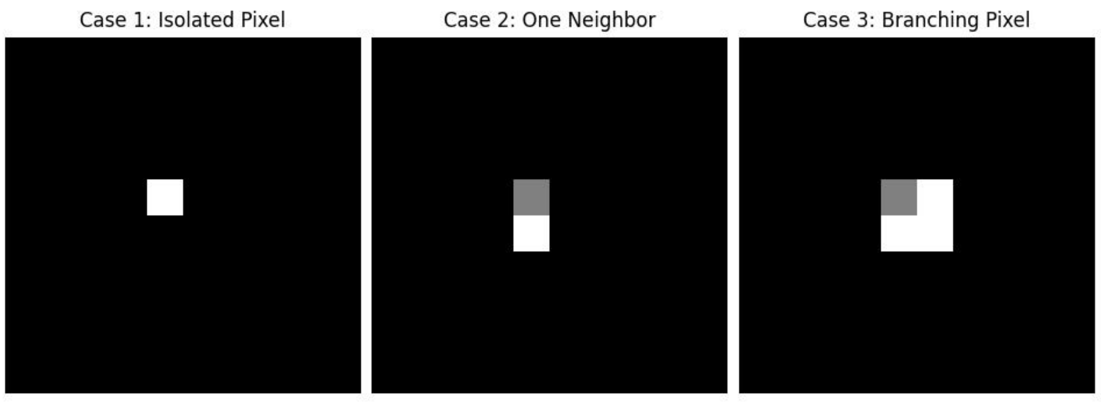


Figure 4.1: The three described cases in the algorithm are further visualized here.

For this thesis we want to be able to separate the connected components which were

identified as "pouring containers" into a greyscale image. This means that each pixel only contains one intesity value between 0 (black) and 255 (white).
After that apply binary thresholding to the greyscale image to segment it into fore– and background regions. The function `cv2.threshold` from the python library [55] assigns new values to pixels following this rule:

$$\text{binary\_image}(x, y) = \begin{cases} 255 & \text{if } I(x, y) < 127 \\ 0 & \text{if } I(x, y) \geq 127 \end{cases}$$

The next step is to use connected component labeling in order to identify and label contiguous regions in a binary image. The function `cv2.connectedComponents` from the python library [55] outputs:

- `num_labels`: The total number of connected components, including the background.
- `labels`: An array with the same dimensions as the input image, where each pixel is assigned an integer label corresponding to its connected component.

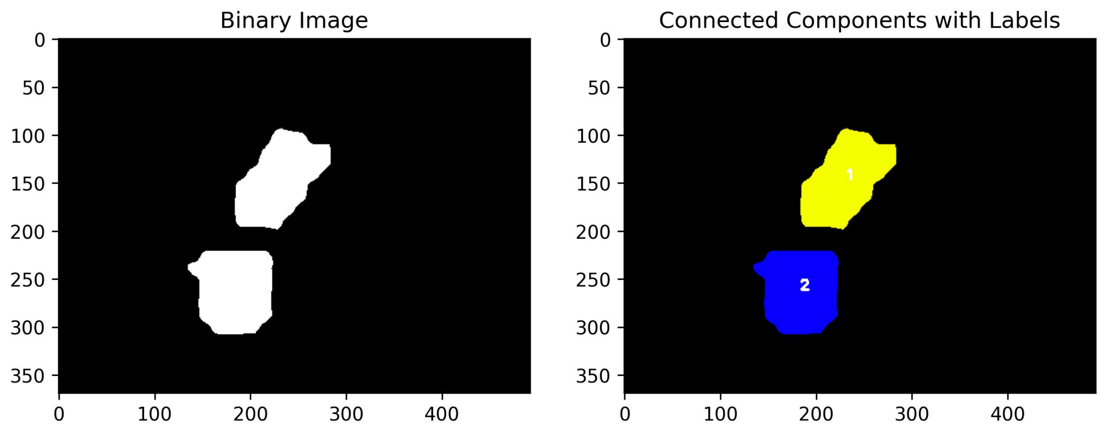


Figure 4.2: Connected components of identified pouring containers using `cv2.threshold` [55].

4.2 Machine Learning Methods

This section provides a concise overview of foundational machine learning algorithm types, including supervised, unsupervised, and reinforcement learning.

Supervised Learning In supervised learning, the model is trained using a dataset of labeled examples and is tasked with making predictions for unseen data points [51].

Semi–supervised Learning Semi–supervised learning utilizes a combination of labeled and unlabeled data. Initially, a portion of the model is trained with manually labeled examples. The trained model is then employed to generate predictions for the unlabeled data, creating pseudo–labels. Finally, the complete dataset, comprising both the original labeled data and the pseudo–labeled data, is used to train the entire system [51].

Unsupervised Learning In unsupervised learning, the model is provided only with unlabeled training data and is tasked with making predictions for unseen data points. Without labeled examples, assessing the model's performance quantitatively can be challenging. Common applications of unsupervised learning include problems such as clustering and dimensionality reduction [51].

Self–Supervised Learning (SSL) Self–supervised learning (SSL) represents a novel approach that does not require manual labeling. SSL tasks are typically categorized into pretext tasks and downstream tasks. Pretext tasks involve using supervised learning to extract representations, where labels are automatically derived from the data. Once this phase is completed, the model utilizes the learned representations to tackle downstream tasks [67].

4.3 K–Means

K–means is an unsupervised clustering algorithm [81].
Each cluster is represented by their centroid, which is typically the mean of the points in the cluster.
The general process of K–means clustering is [81]:

1. Initialize Random Centroids
2. Assign every point in the data to the closest centroid
3. Update the centroid based on the points in the cluster

Repeat the process until no point changes cluster.

4.4 PCA

The goal of a Principal Component Analysis (PCA) is to extract the most meaningful basis in order to represent a given data set. The new basis will additionally filter out noise. PCA's can be used in various applications such as dimensionality reduction and feature extraction [39].

PCA's compute principal components, which are a set of new orthogonal variables. These principal components are retrieved as linear combinations from the original data. PCA can be also formulated as the linear projection that minimizes the average projection cost defined as the mean squared distance between the data points and their projections [39].

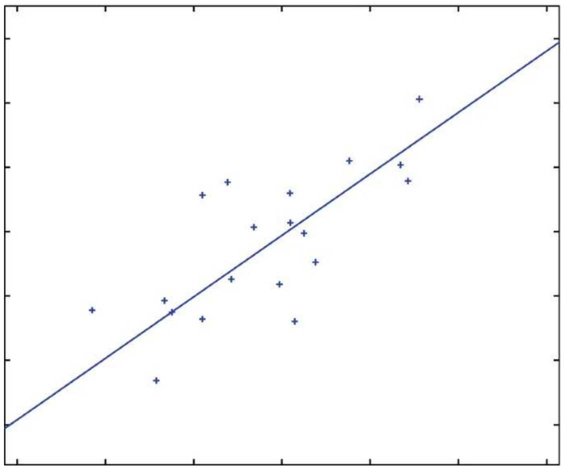


Figure 4.3: Example of PCA of 19 two dimensional samples [39].

4.5 Projection and Back–Projection in Camera Models

In this section we further elaborte the projection and back–projection in camera models. As foundation for this, we will use Hartley and Zisserman [28].

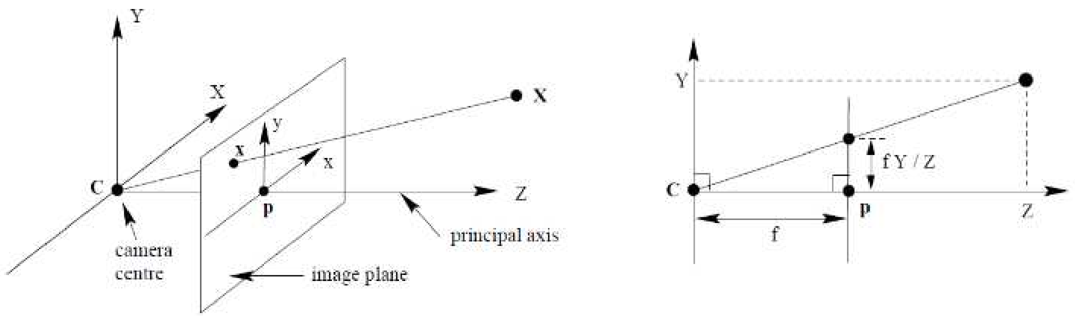


Figure 4.4: Visualizations of the camera model [28].

The projective transformation is a function from a point P in a 3D space to a projected 2D point P' in the image plane Pi'. For this we use the camera matrix mode, which includes important parameters, that affect how the world point P is mapped to the image coordinates P'. The translation of the image plane to the digital image is given by c_x and c_y. The distance between the image plane and the center of the camera is the focal length f. The parameters k and l correspond to the change of units in the two axes of the image plane. The matrix K is often referred to as the camera matrix.

Now the mapping is:

$$P' = MP = \begin{bmatrix} \alpha & 0 & c_x \\ 0 & \beta & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} I & 0 \end{bmatrix} P = \begin{bmatrix} fk\frac{x}{z} & 0 & c_x \\ 0 & fl\frac{y}{z} & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} I & 0 \end{bmatrix} P = K \begin{bmatrix} I & 0 \end{bmatrix} P \quad (4.1)$$

In case the cameras coordinate system axes are slightly larger or smaller than 90 degrees, we can also introduce the skewness and distortion. Since most cameras have zero–skew, this is not further elaborated.

In order to retrieve the real world coordinates, transform the given equation system to retrieve P. It hence yields that

$$x = \frac{x' - c_x z}{\alpha}$$

$$y = \frac{y' - c_y z}{\beta}$$

4.6 DINO

DINO (self–distillation with no labels) is a self–supervised method, which is a form of self–distillation without labels [10].
It uses the following key concepts:

Knowledge Distillation A common method to enhance machine learning performance involves training multiple models on the same dataset and averaging their predictions, but this approach can be complex and resource–intensive. Knowledge Distillation addresses this by condensing the knowledge of multiple models into a single, deployable model [31].

Vision Transformer (ViTs) Vision Transformers (ViTs) are transformer–based models for image classification that bypass the need for convolutional networks by directly processing sequences of image patches, achieving state–of–the–art results on benchmarks with greater efficiency when pre–trained on large datasets. [21]

Teacher Network A Teacher Network is dynamically constructed during training using the student's past states. The teacher is updated through an exponential moving average (EMA) of the student's weights, following a cosine schedule to gradually adjust the influence of past updates [10].

4.7 DINOBot

DINOBot is an advanced imitation learning framework for robotic manipulation that utilizes features from Vision Transformers trained with DINO to enhance learning and generalization. By leveraging image–level and pixel–level similarities to align actions with novel objects, it achieves effective interaction and improved efficiency in real–world tasks, such as pouring [59].

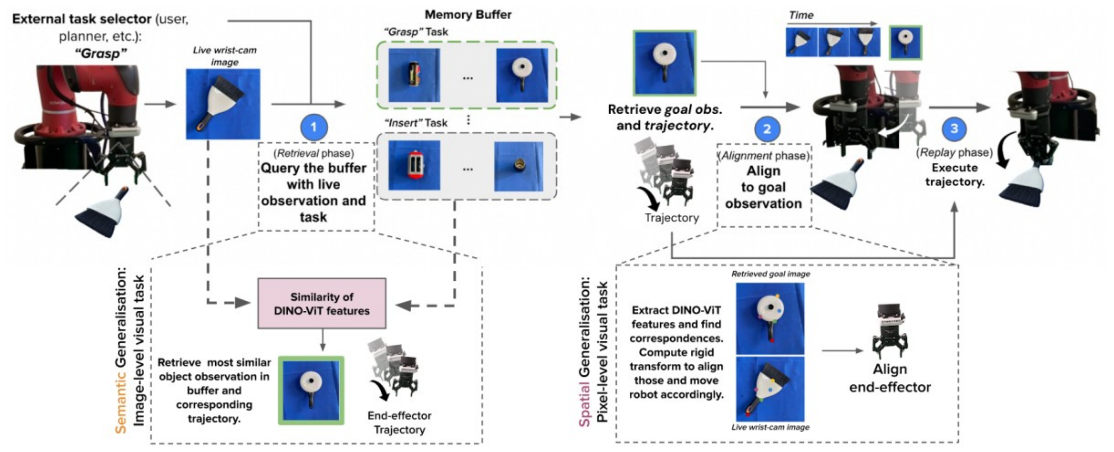

External task selector (user, planner, etc.): "Grasp"
Live wrist-cam image
Memory Buffer
"Grasp" Task
"Insert" Task
1
(Retrieval phase) Query the buffer with live observation and task
Retrieve goal obs. and trajectory.
Trajectory
2
(Alignment phase) Align to goal observation
Time
3
(Replay phase) Execute trajectory.
Semantic Generalisation: Image-level visual task
Similarity of DINO-ViT features
Retrieve most similar object observation in buffer and corresponding trajectory.
End-effector Trajectory
Spatial Generalisation: Pixel-level visual task
Retrieved goal image
Extract DINO-ViT features and find correspondences. Compute rigid transform to align those and move robot accordingly.
Live wrist-cam image
Align end-effector

Figure 4.5: Setup uf the DINOBot framework. When encountering a new object, the robot identifies the most similar demonstrated object by leveraging image–level (semantic) reasoning and retrieves its image along with the corresponding trajectory. It then uses pixel–level (spatial) reasoning to align its end–effector with the retrieved image before executing the trajectory, relying on features extracted and matched using DINO–ViT [59].

4.8 CLIP

This section discusses the neural network CLIP (Contrastive Language–Image Pre–Training), introduced by OpenAI in 2021 [64]. CLIPs original pre–training task is to predict captions for images [65].

Pre–training The limitation of other Deep Learning approaches like ResNets [29] or Vision Transformers [21] in Computer Vision is explained by the fact that datasets require a lot of annotations and only contain a finite set of classes to predict [64]. Pre–training describes all of those models, which are (pre–)trained on one dataset. Because a lot of tasks in Computer Vision are similar, those models are then supposed to perform well on other datasets [90]. As stated before, CLIP's pre–training task is to predict which caption is paired with which image. Additionally, CLIP collects information about how to represent the knowledge we gathered from the data used for training. Then CLIP is able to also adapt to new tasks, such as geo–localization, optical character recognition and others [65]. CLIP is pre–trained on the dataset WIT (WebImageText). This dataset contains 400 million publicly available (image, text) pairs. In addition, WIT has a similar total word count to the dataset used to train GPT–2 [65]. Sadly, this dataset is not openly available.

Zero–shot learning Zero–shot learning deals with problems for which the available training data does not provide enough examples of desired output classes for the tasks we want to solve. This is relevant, since it is not possible to collect every (image, text) pair available. The idea behind zero–shot learning is that unseen objects are learnable, because of acquired general knowledge on how to represent the seen training classes. An example can be found in Figure 4.6. The training samples only contain images which can be classified in three classes, where one class is one number. On the other hand, the test samples display unseen characters as output classes. Because the zero–shot classification produced a representation of each number class in the training set, new classes can be described using the gathered general knowledge [40].

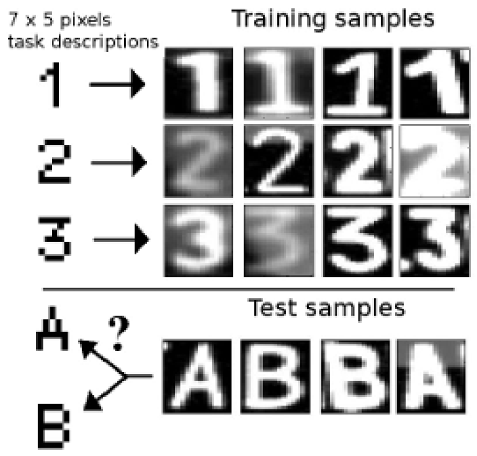


Figure 4.6: Example zero–shot classification for a character recognition problem [40]

In this example, CLIP is supposed to learn visual concepts from the supervision which is contained in natural language [64].

Contrastive learning Recent work in contrastive representation learning for images have found that contrastive objectives can learn better representations than their equivalent predictive objective [75]. So instead of using the predictive learning objective, which learns latent representations that predict one view from another, with loss measured in the output space [75], CLIP uses contrastive learning. The idea behind contrastive learning is based on recognizing the similarities and differences between objects by grouping feature embeddings of similar samples closer together and separate feature embeddings of dissimilar samples [30]. The contrastive objective representations are learned by contrasting similar and dissimilar views, with loss measured in representation space [75]. A visualization of this concept can be foundin Figure 4.7.

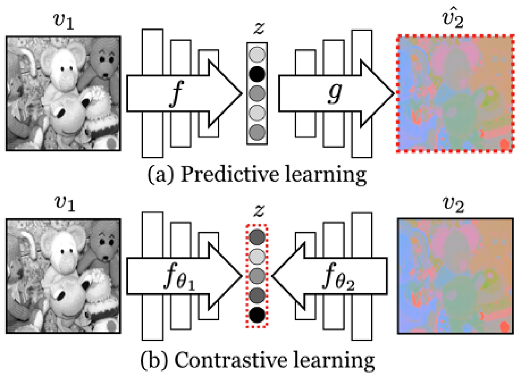
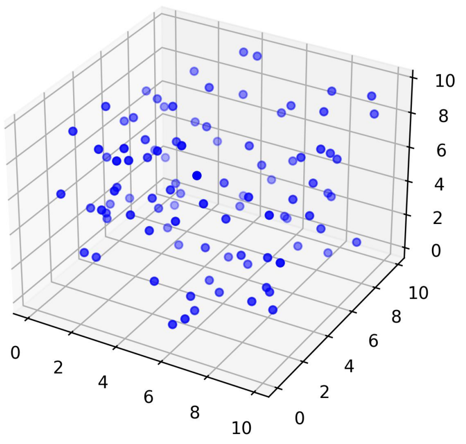


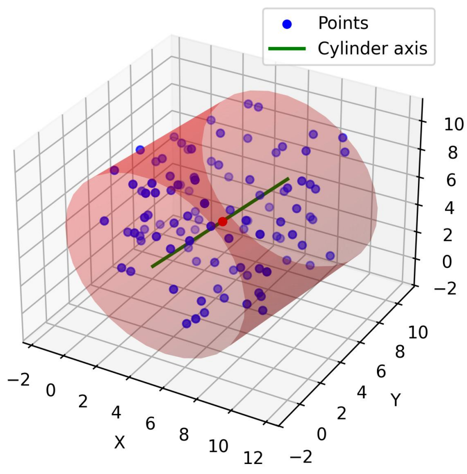
Figure 4.7: Predictive Learning vs Contrastive Learning. The red dotted outlines show where the loss function is applied [75].

4.9 Minimal Enclosing Cylinder

The approach to determine the minimal enclosing cylinder is derived from the method mentioned in Petitjean [62].
Let:



(a) Point cloud



(b) Minimal enclosing cylinder

Figure 4.8: Side–by–side comparison of point cloud and corresponding minimal enclosing cylinder.

- $P = \{p_1, p_2, \ldots, p_n\}$ be the set a points in $\mathbb{R}^3$.
- $\vec{a}$ be the unit vector representing the axis direction of the cylinder.
- $\vec{c}$ be a point on this axis.

The goal is to find:

1. The optimal cylinder axis direction $\vec{a}$.
2. A point $\vec{c}$ on this axis.
3. The minimal radius r such that all points in P lie within or on the surface of the cylinder.

The first step to find the minimal enclosing cylinder is to make an initial guess for the axis direction $\vec{a}_0$, which is supposed to give an estimation for the orientation of the points. One method that can be used for this use case is the Principal Component Analysis (PCA).

The next step is to estimate an initial point $\vec{c}_0$ on the axis as the centroid of P:

$$\vec{c}_0 = \frac{1}{n}\sum_{i=1}^{n} p_i.$$

Next define the initial parameters θ_0 for the optimization as:

$$\theta_0 = (\vec{a}_0, \vec{c}_0)\,.$$

The objective function we minimize is:

$$r = \min_{\vec{a},\vec{c}} \max_{i} \left(\|(\vec{p}_i - \vec{c}) - ((\vec{p}_i - \vec{c}) \cdot \vec{a})\, \vec{a}\| \right).$$

It consists of the the maximum of the radial distance from each point in P to the axis defined by $(\vec{a}, \vec{c})$. The aim is to minimize this function, as it represents the radius of the cylinder.
Use the BFGS optimization method [73] to find the optimal radius r, with respect to $\vec{a}$ and $\vec{c}$, with the initial guess θ_0. The optimization then provides $\vec{a}_{\text{opt}}$: the optimal vector for the cylinder's axis, $\vec{c}_{\text{opt}}$: the optimal point on the axis, and r_{opt}: the minimal radius of the enclosing cylinder. Lastly normalize $\vec{a}_{\text{opt}}$ to ensure it is a unit vector:

$$\vec{a}_{\text{opt}} = \frac{\vec{a}_{\text{opt}}}{\|\vec{a}_{\text{opt}}\|}.$$

In summary, the minimal enclosing cylinder is defined by the axis $\vec{a}_{\text{opt}}$, point $\vec{c}_{\text{opt}}$, and radius r_{opt}.

5 Baseline Implementation: Using DINOBot

DINOBot [58] introduces a one–shot imitation learning framework for robot manipulation tasks that uses Vision Transformers (DINO–ViTs) in order to be adaptable to unseen objects. The framework is structured into three different phases: semantic retrieval, spatial alignment, and trajectory replay.

5.1 Semantic Retrieval

In this phase, the robot utilizes its wrist–mounted RGB–D camera to capture observations before the task execution. These observations are used to query a memory buffer containing previously stored demonstrations. Each demonstration in the memory includes:

1. Wrist–camera images associated with the task.
2. The corresponding end–effector trajectory.

The framework retrieves the most visually similar demonstration based on the DINO–ViT model's semantic reasoning capabilities.

5.2 Spatial Alignment

Once the demonstration is retrieved, the robot performs spatial alignment with the target object. Using the DINO–ViT model's pixel–level feature extraction, the framework aligns the robot's end–effector to the retrieved demonstration image.

5.3 Trajectory Replay

Following spatial alignment, the robot executes the trajectory associated with the retrieved demonstration.

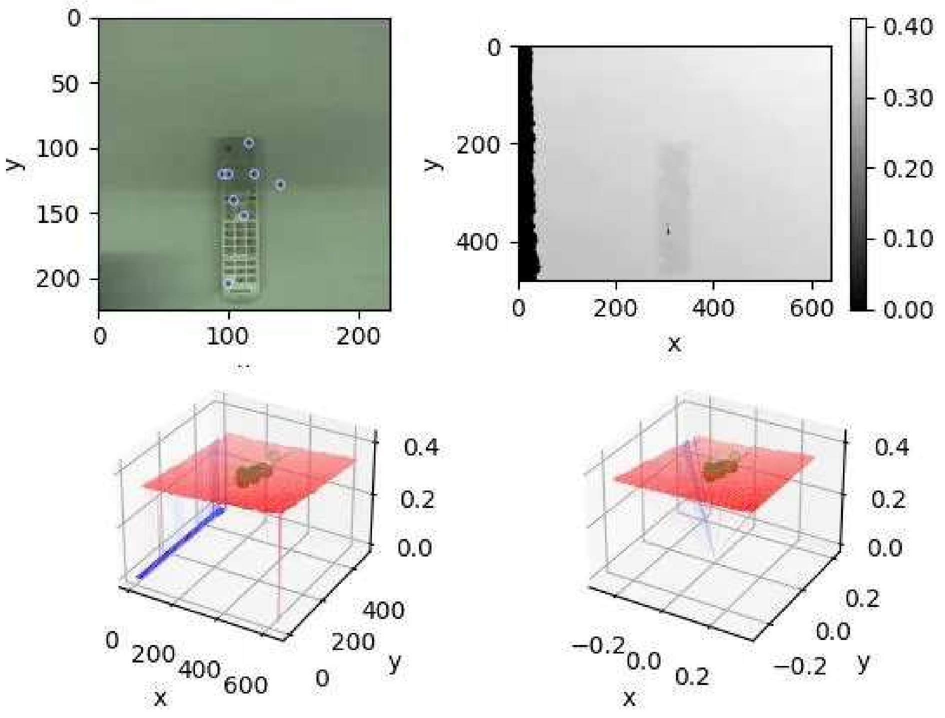


Figure 5.1: DINOBot found keypoints for a remote and maps those into world coordinates.

6 Improved Robotic Framework: Using CLIP–DINOiser and Geometric Abstraction

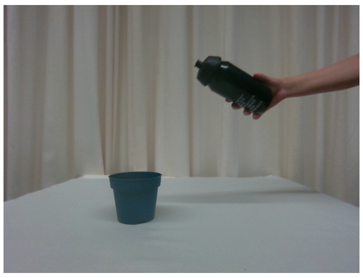
In the next sections we explain how the implementation of the self–supervised pouring from human demonstration was realized. The improvement compared to the baseline algorithm is that we can now use a third persons perspective, in order to learn pouring. The method itself is able to be generalized to other tasks, as well as to other container types as displayed in the human demonstration.

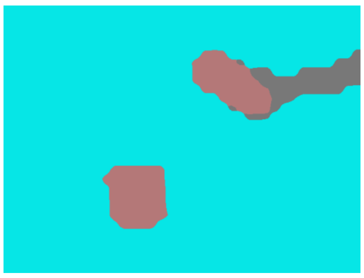
6.1 Extraction

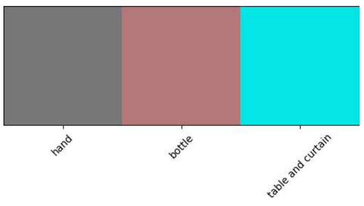
To analyze human pouring actions in a demonstration video, a systematic approach was used to extract velocities and rotational changes. The following steps showcase how this was achieved:

Segmentation of Pouring Containers Each frame of the demo video was processed to identify the containers involved in the pouring action. This was achieved using CLIP–DINOiser [82], a segmentation framework capable of leveraging pre–trained visual and textual embeddings to accurately extract segmentation masks. An example for this can be found in the Figure 6.1.

Identification of Separate Pouring Containers The pouring containers in the segmentation mask are separated using connected components labeling. An example for this can be found in the Figure 6.2.


(a) Picture of human pouring.


(b) Segmentation mask of pouring picture.



(c) Labels of segmentation mask.

Figure 6.1: The segmentation mask was generated using CLIP–DINOiser [82].

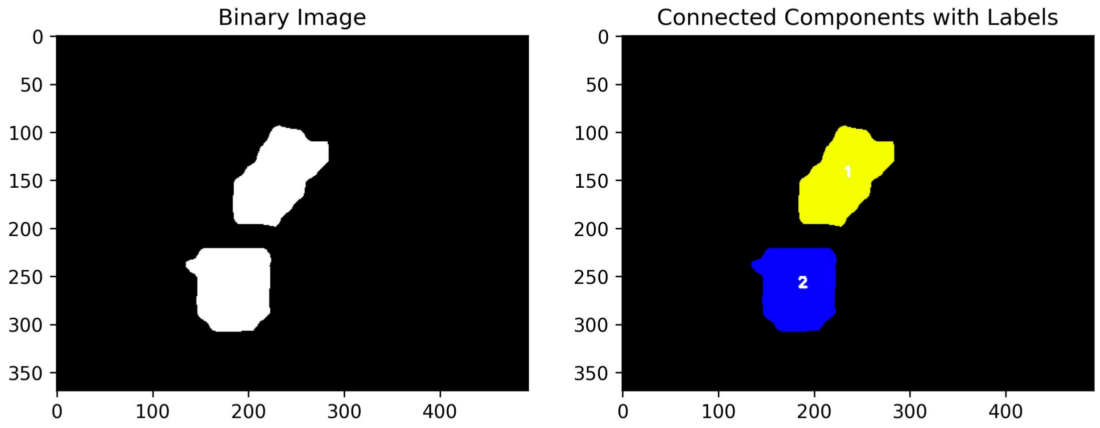


Figure 6.2: Plot of connected Components from identified pouring containers.

Depth Extraction in World Coordinates Depth information of the identified containers was obtained from depth maps captured by the demonstration setup. Using camera intrinsics, the depth data was transformed from pixel space into the world coordinate system. This transformation was crucial for accurately situating the containers in 3D space, enabling precise computation of their movement.

Geometric Representation Using Minimal Enclosing Cylinders To simplify the spatial representation of the containers, the identified 3D points were enclosed within minimal enclosing cylinders. This can be seen in the Figure 6.3.

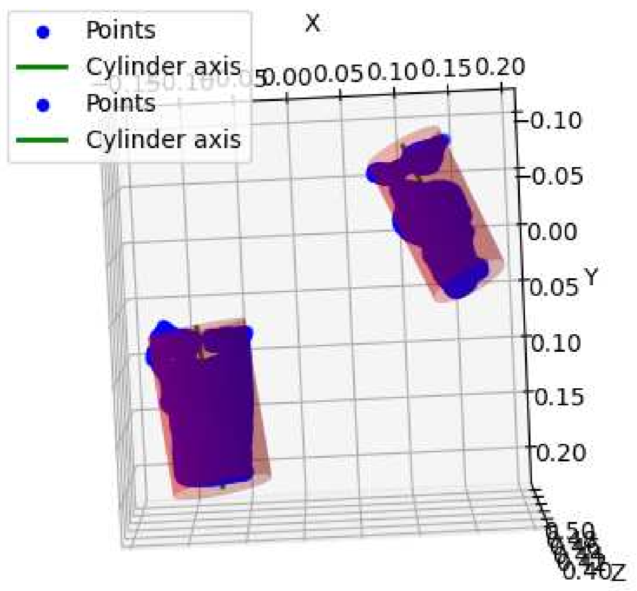


Figure 6.3: The minimal enclosing cylinders from the pouring containers in world coordinates.

Velocity and Rotational Change Calculation Linear velocities were derived from the temporal differences in the cylinder's centroid position, normalized by the frame duration. Similarly, rotational changes were calculated by tracking the orientation vectors of the cylinder's principal axis and computing the angular displacement between consecutive frames.

6.2 Alignment

To replicate the pouring motion demonstrated by a human, an alignment phase was implemented to initialize the robot's configuration relative to the target container.

Capturing the Robot's Initial Configuration A picture of the robot in its current live configuration was taken, capturing the source and target container within the workspace. This image serves as the input for analyzing the spatial arrangement of the robot's end effector and its alignment with the target container.

Extracting Minimal Enclosing Cylinders The minimal enclosing cylinder abstraction was employed, as previously described, to represent both the source and the target container. Using the segmentation methods CLIP–DINOiser [82], segmentation masks were generated to isolate these objects. The extracted 3D points from the segmented regions were used to compute the minimal enclosing cylinders.

Aligning the Source and Target Cylinders The minimal enclosing cylinder representing the source pouring container's cylinder was positioned relative to the target container's cylinder using the spatial relationship observed in the demonstration's first frame. This alignment ensures that the source cylinder's relative position and orientation to the target cylinder mimic the setup at the beginning of the pouring demonstration.

6.3 Execution

After the alignment phase, the robot executes the pouring motion by applying the learned velocities and rotational changes extracted from the demonstration. This phase translates the human pouring dynamics into the robot's operational parameters.

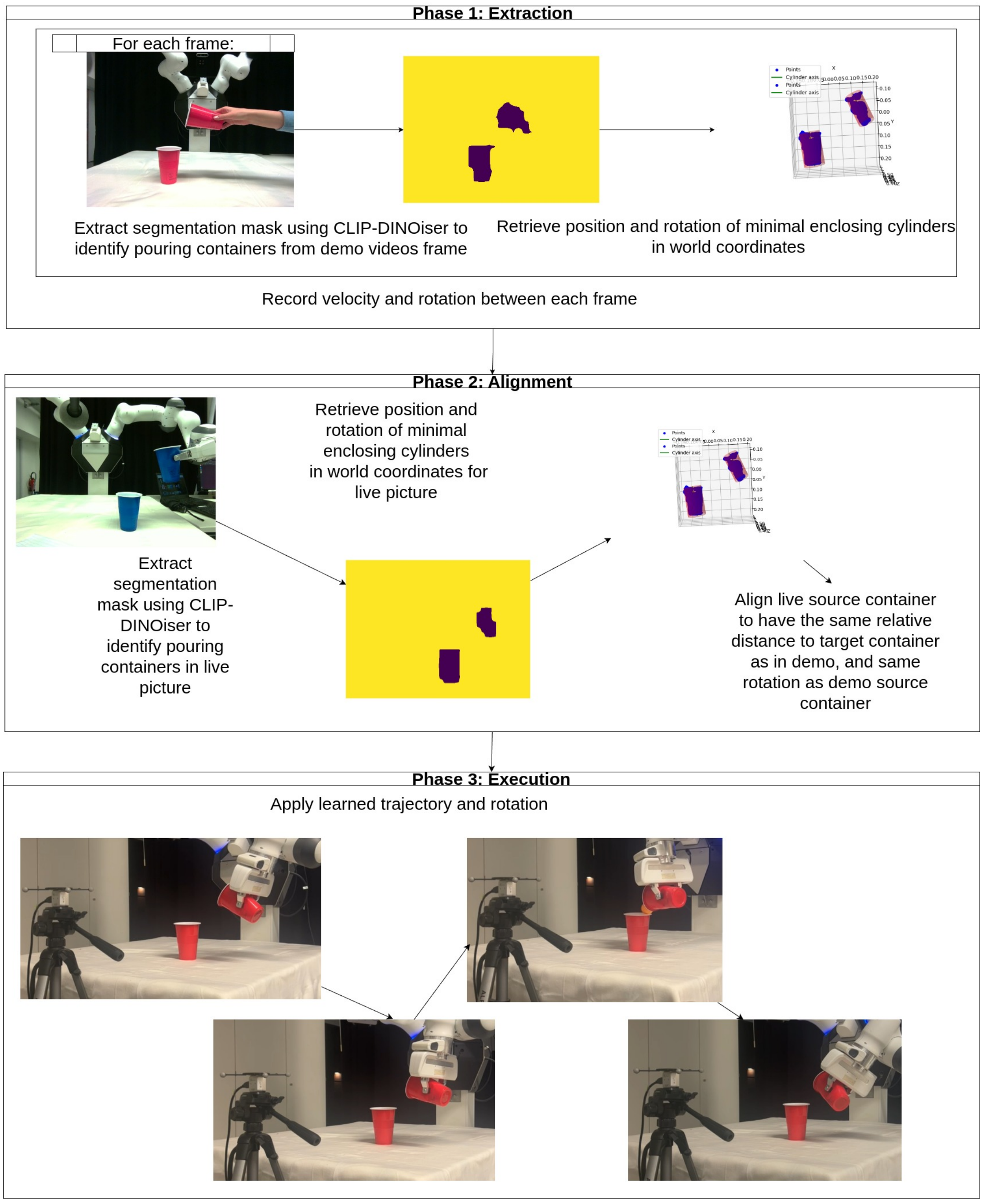

Phase 1: Extraction
For each frame:
Extract segmentation mask using CLIP-DINOiser to identify pouring containers from demo videos frame
Retrieve position and rotation of minimal enclosing cylinders in world coordinates
Record velocity and rotation between each frame
Phase 2: Alignment
Retrieve position and rotation of minimal enclosing cylinders in world coordinates for live picture
Extract segmentation mask using CLIP-DINOiser to identify pouring containers in live picture
Align live source container to have the same relative distance to target container as in demo, and same rotation as demo source container
Phase 3: Execution
Apply learned trajectory and rotation

Figure 6.4: Overall visualization of the CLIP–DINOiser implementation.

7 Experimental Results

In this experiment, we aimed to achieve successful material pours, defined as instances where the poured materials successfully entered the target containers. To validate this process, we designed a setup combining precise motion tracking and Computer Vision technologies to facilitate and evaluate the pouring tasks.

Test Setup The experimental setup involved utilizing OptiTrack [56], a motion capture system, to track the position of the camera throughout the experiment. The camera used for this purpose was the RealSense D405 [16], which provided high–resolution depth sensing critical for accurately capturing the spatial alignment of source and target containers. The camera gets tracked using Optitrack Markers [56]. The robot used in this setup is a Franka Kobo [24].

To provide visual context, we included photographs of the test objects used during the experiment, such as the source and target containers, as well as the materials poured during the trials.

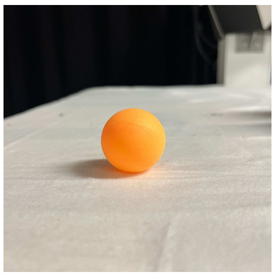


Figure 7.4: The pouring materials are ping pong balls

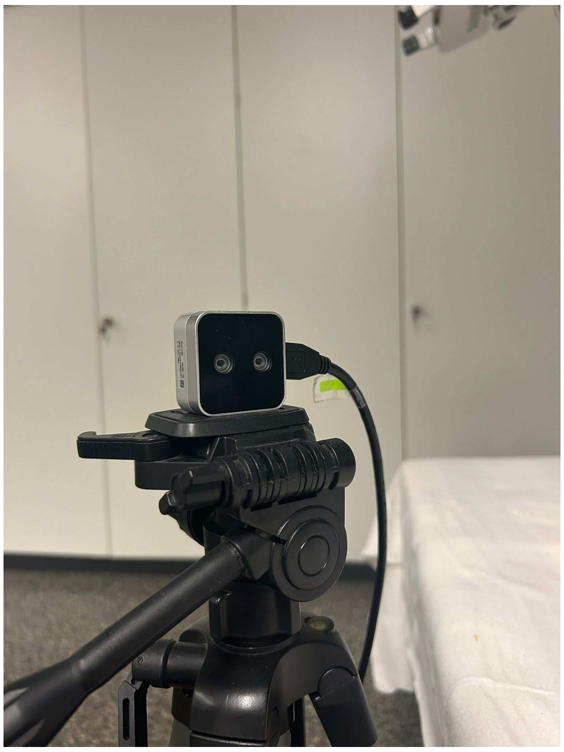


Figure 7.1: Camera with camera stand

Results Overview A single demonstration trajectory was recorded using the designated camera, utilizing the pouring container (a red cup) as both the source and target container to illustrate the ideal pouring action. During this demonstration, we successfully transferred ping pong balls from one cup to another, effectively showcasing the feasibility and precision of the pouring task executed by the improved robotics framework using CLIP–DINOiser and geometric abstraction.

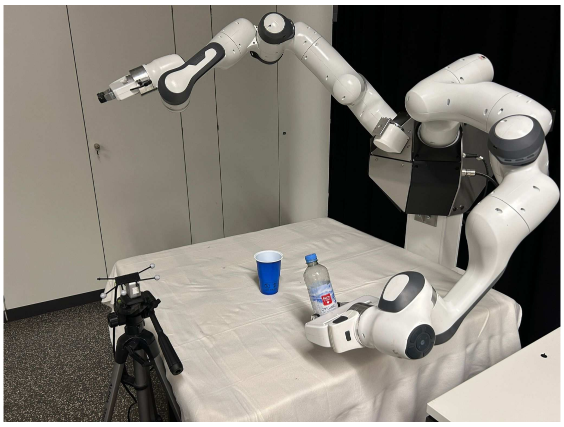


Figure 7.2: Test Setup with Robot

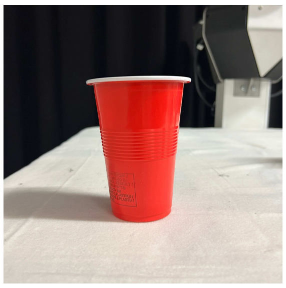


Figure 7.3: The pouring containers are plastic cups

8 Discussion

This thesis aimed to develop a robotic pouring framework leveraging advanced segmentation, geometric abstraction, and motion learning techniques. The research successfully achieved several milestones, including the implementation of a baseline system and an improved framework. The improved framework integrated state–of–the–art tools such as CLIP–DINOiser for object segmentation and minimal enclosing cylinders for geometric abstraction.

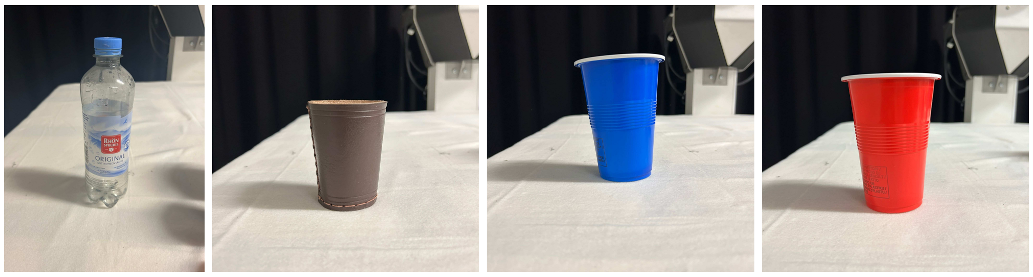
However, a significant limitation arose during the testing phase. A critical component of the experimental setup, the OptiTrack system responsible for tracking the camera position, became inoperable due to hardware failure. This issue prevented a thorough evaluation of the framework's performance. Without the ability to accurately monitor and validate the robot's movements in 3D space, the intended testing and benchmarking against the baseline could not be fully realized.

In my extensive literature review on robotic pouring, I observed that there is no universally accepted state–of–the–art metric for evaluating performance in this domain.
Additionally, most approaches identified in the review were hybrid methods, combining elements of both model based and model free strategies, rather than adhering strictly to one paradigm.

Despite this setback, the framework design and implementation represent a valuable contribution to the field, offering a structured approach to robotic pouring.

9 Outlook

To further validate the proposed robotic pouring framework, future tests will focus on evaluating the approach using a variety of pouring materials and containers, aiming to assess its robustness and adaptability in different scenarios. These planned experiments will provide deeper insights into the performance of the framework under diverse conditions, paving the way for broader applications.
The containers in the Figure 9.1 can be the test source and target containers. The considered pouring materials can be seen in the Figure 9.2. The tests to evaluate the proposed robotic pouring framework in this thesis can be found in the Tables 9.3, 9.1, and 9.2.
Future work will also focus on extending the application of the framework to other robotic tasks beyond pouring, with the goal of evaluating its ability to generalize across different scenarios.



(a) Bottle (b) Leather cup (c) Plastic cup (d) Plastic cup

Figure 9.1: Pictures of the planned pouring containers for more extensive testing.

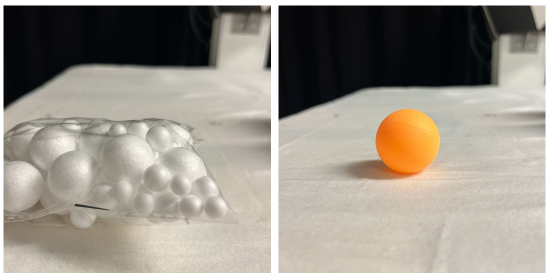

(a) Styrofoam balls (b) Ping pong ball

Figure 9.2: Planned pouring materials to use for more extensive testing.

Table 9.1: Percentage of successful Pours per pouring container, as a metric for more extensive testing

	Target (a)	Target (b)	Target (c)	Target (d)
Source (a)	- %	- %	- %	- %
Source (b)	- %	- %	- %	- %
Source (c)	- %	- %	- %	- %
Source (d)	- %	- %	- %	- %

Table 9.2: Percentage of successful pours per pouring material, as a metric for more extensive testing

Live \Demo	Pouring Material (a)	Pouring Material (b)	
Pouring Material (a)	- %	- %	- %
Pouring Material (b)	- %	- %	- %

Table 9.3: Percentage of successful Pours per container category, as a metric for more extensive testing

	Target Cups	**Target Bottles**
Source Cups	- %	- %
Source Bottles	- %	- %

Bibliography

[1] FIFTY2 Technology GmbH. PreonLab, https://www.fifty2.eu/preonlab/.

[2] Ayman AbuBaker et al. “One Scan Connected Component Labeling Technique”. In: *2007 IEEE International Conference on Signal Processing and Communications*. 2007, pp. 1283–1286. DOI: 10.1109/ICSPC.2007.4728561.

[3] Edwin Babaians et al. “PourNet: Robust Robotic Pouring Through Curriculum and Curiosity-based Reinforcement Learning”. In: *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2022, pp. 9332–9339. DOI: 10.1109/IROS47612.2022.9981195. URL: https://ieeexplore.ieee.org/abstract/document/9981195.

[4] Michael Beetz et al. “Know Rob 2.0 — A 2nd Generation Knowledge Processing Framework for Cognition-Enabled Robotic Agents”. In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. 2018, pp. 512–519. DOI: 10.1109/ICRA.2018.8460964. URL: https://ieeexplore.ieee.org/abstract/document/8460964.

[5] H. Ben Amor et al. “Special issue on autonomous grasping and manipulation”. In: (2014).

[6] B. Bocsi, L. Csato, and J. Peters. “Indirect Robot Model Learning for Tracking Control”. In: (2014). URL: http://www.ias.tu-darmstadt.de/uploads/Publications/Bocsi_AR_2014.pdf.

[7] Federico Bolelli et al. “A State-of-the-Art Review with Code about Connected Components Labeling on GPUs”. In: *IEEE Transactions on Parallel and Distributed Systems* (2024), pp. 1–20. DOI: 10.1109/TPDS.2024.3434357.

[8] Sascha Brandi, Oliver Kroemer, and Jan Peters. “Generalizing pouring actions between objects using warped parameters”. In: *2014 IEEE-RAS International Conference on Humanoid Robots*. 2014, pp. 616–621. DOI: 10.1109/HUMANOIDS.2014.7041426. URL: https://ieeexplore.ieee.org/abstract/document/7041426.

[9] Alexis Burns et al. “Look and Listen: A Multi-Sensory Pouring Network and Dataset for Granular Media from Human Demonstrations”. In: *2022 International Conference on Robotics and Automation (ICRA)*. 2022, pp. 2519–2524. DOI: 10.1109/ICRA46639.2022.9812125. URL: https://ieeexplore.ieee.org/abstract/document/9812125.

[10] Mathilde Caron et al. “Emerging Properties in Self-Supervised Vision Transformers”. In: *CoRR* abs/2104.14294 (2021). arXiv: 2104.14294. URL: https://arxiv.org/abs/2104.14294.

[11] Tianze Chen, Yongqiang Huang, and Yu Sun. “Accurate Pouring using Model Predictive Control Enabled by Recurrent Neural Network”. In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2019, pp. 7688–7694. DOI: 10.1109/IROS40897.2019.8967802. URL: https://ieeexplore.ieee.org/document/8967802.

[12] Xi Chen et al. “Adversarial Feature Training for Generalizable Robotic Visuomotor Control”. In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. 2020, pp. 1142–1148. DOI: 10.1109/ICRA40945.2020.9197505. URL: https://ieeexplore.ieee.org/document/9197505.

[13] A.G. Kupcsik et al. “Model-based Contextual Policy Search for Data-Efficient Generalization of Robot Skills”. In: (conditionally accepted).

[14] K. Muelling et al. “Learning Strategies in Table Tennis using Inverse Reinforcement Learning”. In: (accepted).

[15] C. Dann, G. Neumann, and J. Peters. “Policy Evaluation with Temporal Differences: A Survey and Comparison”. In: March (2014), pp. 809–883. URL: http://jmlr.org/papers/volume15/dann14a/dann14a.pdf.

[16] Intel Corporation. *Intel RealSense Depth Camera D405*. Available at https://www.intelrealsense.com/depth-camera-d405/. 2024. URL: https://www.intelrealsense.com/depth-camera-d405/.

[17] Renaud Detry, Jeremie Papon, and Larry Matthies. “Task-oriented grasping with semantic and geometric scene understanding”. In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2017, pp. 3266–3273. DOI: 10.1109/IROS.2017.8206162. URL: https://ieeexplore.ieee.org/document/8206162.

[18] Chau Do, Camilo Gordillo, and Wolfram Burgard. “Learning to Pour using Deep Deterministic Policy Gradients”. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2018, pp. 3074–3079. DOI: 10.1109/IROS.2018.8593654. URL: https://ieeexplore.ieee.org/abstract/document/8593654/.

[19] Chau Do, Tobias Schubert, and Wolfram Burgard. “A probabilistic approach to liquid level detection in cups using an RGB-D camera”. In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2016, pp. 2075–2080. DOI: 10.1109/IROS.2016.7759326. URL: https://ieeexplore.ieee.org/document/7759326.

[20] Chenyu Dong et al. “Precision Pouring into Unknown Containers by Service Robots”. In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2019, pp. 5875–5882. DOI: 10.1109/IROS40897.2019.8967911. URL: https://ieeexplore.ieee.org/abstract/document/8967911.

[21] Alexey Dosovitskiy et al. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. 2021. arXiv: 2010.11929 [cs.CV]. URL: https://arxiv.org/abs/2010.11929.

[22] Debidatta Dwibedi et al. “Learning Actionable Representations from Visual Observations”. In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2018, pp. 1577–1584. DOI: 10.1109/IROS.2018.8593951. URL: https://ieeexplore.ieee.org/document/8593951.

[23] Christof Elbrechter et al. “Discriminating liquids using a robotic kitchen assistant”. In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2015, pp. 703–708. DOI: 10.1109/IROS.2015.7353449. URL: https://ieeexplore.ieee.org/document/7353449.

[24] Franka Emika GmbH. *Franka Emika Panda Robot*. Accessed: 2024-12-29. 2024. URL: https://www.franka.de.

[25] F. Gravot et al. “Cooking for humanoid robot, a task that needs symbolic and geometric reasonings”. In: *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.* 2006, pp. 462–467. DOI: 10.1109/ROBOT.2006.1641754. URL: https://ieeexplore.ieee.org/abstract/document/1641754.

[26] Andrei Haidu et al. "KnowRobSIM — Game Engine-Enabled Knowledge Processing Towards Cognition-Enabled Robot Control". In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2018, pp. 4491–4498. DOI: 10.1109/IROS.2018.8593935.

[27] Nathaniel Hanson et al. "SLURP! Spectroscopy of Liquids Using Robot Pre-Touch Sensing". In: *2023 IEEE International Conference on Robotics and Automation (ICRA)*. 2023, pp. 3786–3792. DOI: 10.1109/ICRA48891.2023.10161084. URL: https://ieeexplore.ieee.org/abstract/document/10161084.

[28] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. USA: Cambridge University Press, 2000. ISBN: 0521623049.

[29] Kaiming He et al. *Deep Residual Learning for Image Recognition*. 2015. arXiv: 1512.03385 [cs.CV]. URL: https://arxiv.org/abs/1512.03385.

[30] Kaiming He et al. *Momentum Contrast for Unsupervised Visual Representation Learning*. 2020. arXiv: 1911.05722.

[31] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. *Distilling the Knowledge in a Neural Network*. 2015. arXiv: 1503.02531 [stat.ML].

[32] Yongqiang Huang and Yu Sun. "A dataset of daily interactive manipulation". In: *The International Journal of Robotics Research* 38.8 (2019), p. 8790886. URL: https://rpal.cse.usf.edu/datasets_manipulation.html.

[33] Yongqiang Huang and Yu Sun. "Learning to pour". In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2017, pp. 7005–7010. DOI: 10.1109/IROS.2017.8206626. URL: https://ieeexplore.ieee.org/document/8206626.

[34] Chen Jiang, Masood Dehghan, and Martin Jagersand. "Understanding Contexts Inside Robot and Human Manipulation Tasks through Vision-Language Model and Ontology System in Video Streams". In: *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2020, pp. 8366–8372. DOI: 10.1109/IROS45743.2020.9340905. URL: https://ieeexplore.ieee.org/abstract/document/9340905.

[35] Gayane Kazhoyan and Michael Beetz. "Programming robotic agents with action descriptions". In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2017, pp. 103–108. DOI: 10.1109/IROS.2017.8202144. URL: https://ieeexplore.ieee.org/abstract/document/8202144.

[36] Monroe Kennedy et al. “Autonomous Precision Pouring From Unknown Containers”. In: *IEEE Robotics and Automation Letters* 4.3 (2019), pp. 2317–2324. DOI: 10.1109/LRA.2019.2902075. URL: https://ieeexplore.ieee.org/abstract/document/8653969.

[37] Monroe Kennedy et al. “Precise dispensing of liquids using visual feedback”. In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2017, pp. 1260–1266. DOI: 10.1109/IROS.2017.8202301. URL: https://ieeexplore.ieee.org/abstract/document/8202301.

[38] N. Koenig and A. Howard. “Design and use paradigms for Gazebo, an open-source multi-robot simulator”. In: *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*. Vol. 3. 2004, 2149–2154 vol.3. DOI: 10.1109/IROS.2004.1389727.

[39] Takio Kurita. “Principal Component Analysis (PCA)”. In: *Computer Vision: A Reference Guide*. Cham: Springer International Publishing, 2019, pp. 1–4. ISBN: 978-3-030-03243-2. DOI: 10.1007/978-3-030-03243-2_649-1. URL: https://doi.org/10.1007/978-3-030-03243-2_649-1.

[40] Hugo Larochelle, Dumitru Erhan, and Y. Bengio. *Zero-data Learning of New Tasks*. Jan. 2008.

[41] Hugo Larochelle, Dumitru Erhan, and Yoshua Bengio. “Zero-data Learning of New Tasks”. In: *Proceedings of the Twenty-Third AAAI Conference on Artificial Intelligence, AAAI 2008, Chicago, Illinois, USA, July 13-17, 2008*. Ed. by Dieter Fox and Carla P. Gomes. AAAI Press, 2008, pp. 646–651. URL: http://www.aaai.org/Library/AAAI/2008/aaai08-103.php.

[42] Kimin Lee et al. *Making Stochastic Neural Networks from Deterministic Ones*. 2017. URL: https://sites.google.com/site/brainrobotdata/home/pouring-dataset.

[43] Hongzhuo Liang et al. “Making Sense of Audio Vibration for Liquid Height Estimation in Robotic Pouring”. In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2019, pp. 5333–5339. DOI: 10.1109/IROS40897.2019.8968303. URL: https://ieeexplore.ieee.org/document/8968303.

[44] Hongzhuo Liang et al. “Robust Robotic Pouring using Audition and Haptics”. In: *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2020, pp. 10880–10887. DOI: 10.1109/IROS45743.2020.9340859. URL: https://ieeexplore.ieee.org/abstract/document/9340859.

[45] Haitao Lin, Yanwei Fu, and Xiangyang Xue. *PourIt!: Weakly-supervised Liquid Perception from a Single Image for Visual Closed-Loop Robotic Pouring*. 2023. arXiv: 2307.11299 [cs.RO].

[46] Tatiana Lopez-Guevara et al. "Stir to Pour: Efficient Calibration of Liquid Properties for Pouring Actions". In: *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2020, pp. 5351–5357. DOI: 10.1109/IROS45743.2020.9340852. URL: https://ieeexplore.ieee.org/abstract/document/9340852.

[47] Miles Macklin and Matthias Müller. "Position Based Fluids". In: *ACM Transactions on Graphics* 32 (July 2013), 104:1–. DOI: 10.1145/2461912.2461984.

[48] Carolyn Matl, Robert Matthew, and Ruzena Bajcsy. "Haptic Perception of Liquids Enclosed in Containers". In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2019, pp. 7142–7149. DOI: 10.1109/IROS40897.2019.8968528. URL: https://ieeexplore.ieee.org/abstract/document/8968528.

[49] Carolyn Matl et al. "Inferring the Material Properties of Granular Media for Robotic Tasks". In: *IEEE International Conference on Robotics and Automation (ICRA)*. 2020, pp. 2770–2777. DOI: 10.1109/ICRA40945.2020.9197063. URL: https://ieeexplore.ieee.org/document/9197063/.

[50] T. Meyer et al. "Predicting Motor Learning Performance from Electroencephalographic Data". In: 1 (2014). URL: http://www.ias.tu-darmstadt.de/uploads/Publications/Meyer_JNER_2013.pdf.

[51] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of Machine Learning*. 2nd. The MIT Press, 2018. ISBN: 0262039400.

[52] J. Moldenhauer et al. "Analysis of Human Motion for Humanoid Robots". In: *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*. 2005, pp. 311–316. DOI: 10.1109/ROBOT.2005.1570137. URL: https://ieeexplore.ieee.org/document/1570137.

[53] Gautham Narasimhan et al. "Self-supervised Transparent Liquid Segmentation for Robotic Pouring". In: *2022 International Conference on Robotics and Automation (ICRA)*. 2022, pp. 4555–4561. DOI: 10.1109/ICRA46639.2022.9812000. URL: https://ieeexplore.ieee.org/abstract/document/9812000.

[54] Iman Nematollahi et al. “Augmenting Action Model Learning by Non-Geometric Features”. In: *2019 International Conference on Robotics and Automation (ICRA)*. 2019, pp. 7769–7775. DOI: 10.1109/ICRA.2019.8794153. URL: https://ieeexplore.ieee.org/document/8794153.

[55] OpenCV Team. *OpenCV Library*. Accessed: 2024-12-26. 2024. URL: https://opencv.org/.

[56] OptiTrack. *OptiTrack - Motion Capture Systems*. Accessed: 2024-12-29. 2024. URL: https://www.optitrack.com/.

[57] Joni Pajarinen and Ville Kyrki. “Decision making under uncertain segmentations”. In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. 2015, pp. 1303–1309. DOI: 10.1109/ICRA.2015.7139359. URL: https://ieeexplore.ieee.org/abstract/document/7139359.

[58] Norman Di Palo and Edward Johns. *DINOBot: Robot Manipulation via Retrieval and Alignment with Vision Foundation Models*. 2024. arXiv: 2402.13181 [cs.RO].

[59] Norman Di Palo and Edward Johns. *DINOBot: Robot Manipulation via Retrieval and Alignment with Vision Foundation Models*. 2024. arXiv: 2402.13181 [cs.RO]. URL: https://arxiv.org/abs/2402.13181.

[60] Zherong Pan and Dinesh Manocha. “Feedback motion planning for liquid pouring using supervised learning”. In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2017, pp. 1252–1259. DOI: 10.1109/IROS.2017.8202300. URL: https://ieeexplore.ieee.org/document/8202300.

[61] Zherong Pan and Dinesh Manocha. “Motion planning for fluid manipulation using simplified dynamics”. In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2016, pp. 4224–4231. DOI: 10.1109/IROS.2016.7759622. URL: https://ieeexplore.ieee.org/document/7759622.

[62] Michel Petitjean. “About the algebraic solutions of smallest enclosing cylinders problems”. In: *Applicable Algebra in Engineering, Communication and Computing* 23.3–4 (July 2012), pp. 151–164. ISSN: 1432-0622. DOI: 10.1007/s00200-012-0171-y. URL: http://dx.doi.org/10.1007/s00200-012-0171-y.

[63] Pedro Piacenza, Daewon Lee, and Volkan Isler. “Pouring by Feel: An Analysis of Tactile and Proprioceptive Sensing for Accurate Pouring”. In: *2022 International Conference on Robotics and Automation (ICRA)*. 2022, pp. 10248–10254. DOI: 10.1109/ICRA46639.2022.9811898. URL: https://ieeexplore.ieee.org/abstract/document/9811898.

[64] Alec Radford et al. *CLIP: Connecting text and images*. [Accessed 06-06-2024]. 2021. URL: https://openai.com/index/clip.

[65] Alec Radford et al. *Learning Transferable Visual Models From Natural Language Supervision*. 2021. arXiv: 2103.00020 [cs.CV].

[66] Kartik Ramachandruni et al. "Attentive Task-Net: Self Supervised Task-Attention Network for Imitation Learning using Video Demonstration". In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. 2020, pp. 4760–4766. DOI: 10.1109/ICRA40945.2020.9197544. URL: https://ieeexplore.ieee.org/document/9197544.

[67] Veenu Rani et al. "Self-supervised Learning: A Succinct Review". In: *Archives of Computational Methods in Engineering* 30.4 (May 2023), pp. 2761–2775. ISSN: 1886-1784. DOI: 10.1007/s11831-023-09884-2.

[68] Florian Richter et al. "Autonomous Robotic Suction to Clear the Surgical Field for Hemostasis Using Image-Based Blood Flow Detection". In: *IEEE Robotics and Automation Letters* 6.2 (2021), pp. 1383–1390. DOI: 10.1109/LRA.2021.3056057. URL: https://ieeexplore.ieee.org/abstract/document/9343724.

[69] H. Sakoe and S. Chiba. "Dynamic programming algorithm optimization for spoken word recognition". In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 26.1 (1978), pp. 43–49. DOI: 10.1109/TASSP.1978.1163055. URL: https://ieeexplore.ieee.org/document/1163055/authors#authors.

[70] Connor Schenck and Dieter Fox. "Visual closed-loop control for pouring liquids". In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. 2017, pp. 2629–2636. DOI: 10.1109/ICRA.2017.7989307. URL: https://ieeexplore.ieee.org/abstract/document/7989307.

[71] Pierre Sermanet et al. "Time-Contrastive Networks: Self-Supervised Learning from Video". In: *arXiv preprint arXiv:1704.06888* (2017).

[72] Pierre Sermanet et al. *Time-Contrastive Networks: Self-Supervised Learning from Video*. 2018. arXiv: 1704.06888 [cs.CV]. URL: https://arxiv.org/abs/1704.06888.

[73] D. F. Shanno. "Conditioning of quasi-Newton methods for function minimization". eng. In: *Mathematics of Computation* 24.111 (1970), pp. 647–656. ISSN: 0025-5718.

[74] Kuniyuki Takahashi and Kenta Yonekura. “Invisible Marker: Automatic Annotation of Segmentation Masks for Object Manipulation”. In: *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2020, pp. 8431–8438. DOI: 10.1109/IROS45743.2020.9341332. URL: https://ieeexplore.ieee.org/document/9341332/.

[75] Yonglong Tian, Dilip Krishnan, and Phillip Isola. *Contrastive Multiview Coding*. 2020. arXiv: 1906.05849.

[76] Neea Tuomainen, David Blanco-Mulero, and Ville Kyrki. “Manipulation of Granular Materials by Learning Particle Interactions”. In: *IEEE Robotics and Automation Letters* 7.2 (2022), pp. 5663–5670. DOI: 10.1109/LRA.2022.3158382. URL: https://ieeexplore.ieee.org/document/9732690.

[77] Julen Urain, Davide Tateo, and Jan Peters. “Learning Stable Vector Fields on Lie Groups”. In: *IEEE Robotics and Automation Letters* 7.4 (2022), pp. 12569–12576. DOI: 10.1109/LRA.2022.3219019. URL: https://ieeexplore.ieee.org/abstract/document/9935105.

[78] Justin Wilson, Auston Sterling, and Ming C. Lin. “Analyzing Liquid Pouring Sequences via Audio-Visual Neural Networks”. In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2019, pp. 7702–7709. DOI: 10.1109/IROS40897.2019.8968118. URL: https://ieeexplore.ieee.org/abstract/document/8968118.

[79] Bohan Wu et al. “SQUIRL: Robust and Efficient Learning from Video Demonstration of Long-Horizon Robotic Manipulation Tasks”. In: *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2020, pp. 9720–9727. DOI: 10.1109/IROS45743.2020.9340915. URL: https://ieeexplore.ieee.org/document/9340915.

[80] Hongtao Wu and Gregory S. Chirikjian. “Can I Pour Into It? Robot Imagining Open Containability Affordance of Previously Unseen Objects via Physical Simulations”. In: *IEEE Robotics and Automation Letters* 6.1 (2021), pp. 271–278. DOI: 10.1109/LRA.2020.3039943. URL: https://ieeexplore.ieee.org/document/9269438.

[81] Junjie Wu. “Cluster Analysis and K-means Clustering: An Introduction”. In: *Advances in K-means Clustering: A Data Mining Thinking*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 1–16. ISBN: 978-3-642-29807-3. DOI: 10.1007/978-3-642-29807-3_1. URL: https://doi.org/10.1007/978-3-642-29807-3_1.

[82] Monika Wysoczańska et al. *CLIP-DINOiser: Teaching CLIP a few DINO tricks for open-vocabulary semantic segmentation*. 2024. arXiv: 2312.12359 [cs.CV]. URL: https://arxiv.org/abs/2312.12359.

[83] Akihiko Yamaguchi, Christopher G Atkeson, and Tsukasa Ogasawara. "Pouring skills with planning and learning modeled from human demonstrations". In: *International Journal of Humanoid Robotics* 12.03 (2015), p. 1550030. URL: https://ieeexplore.ieee.org/document/7041472.

[84] Akihiko Yamaguchi and Christopher G. Atkeson. "Differential dynamic programming with temporally decomposed dynamics". In: *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*. 2015, pp. 696–703. DOI: 10.1109/HUMANOIDS.2015.7363430. URL: https://ieeexplore.ieee.org/document/7363430.

[85] Akihiko Yamaguchi and Christopher G. Atkeson. "Neural networks and differential dynamic programming for reinforcement learning problems". In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. 2016, pp. 5434–5441. DOI: 10.1109/ICRA.2016.7487755. URL: https://ieeexplore.ieee.org/document/7487755.

[86] Akihiko Yamaguchi and Christopher G. Atkeson. "Stereo vision of liquid and particle flow for robot pouring". In: *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*. 2016, pp. 1173–1180. DOI: 10.1109/HUMANOIDS.2016.7803419. URL: https://ieeexplore.ieee.org/abstract/document/7803419.

[87] Jiasheng Ye et al. *DINOISER: Diffused Conditional Sequence Learning by Manipulating Noises*. 2024. arXiv: 2302.10025 [cs.CL]. URL: https://arxiv.org/abs/2302.10025.

[88] Dandan Zhang et al. "Explainable Hierarchical Imitation Learning for Robotic Drink Pouring". In: *IEEE Transactions on Automation Science and Engineering* 19.4 (2022), pp. 3871–3887. DOI: 10.1109/TASE.2021.3138280. URL: https://ieeexplore.ieee.org/document/9667114.

[89] R.D. Zollner and R. Dillmann. "Using multiple probabilistic hypothesis for programming one and two hand manipulation by demonstration". In: *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No.03CH37453)*. Vol. 3. 2003, 2926–2931 vol.3. DOI: 10.1109/IROS.2003.1249315. URL: https://ieeexplore.ieee.org/document/1249315.

[90] Barret Zoph et al. “Rethinking Pre-training and Self-training”. In: *Advances in Neural Information Processing Systems*. Ed. by H. Larochelle et al. Vol. 33. Curran Associates, Inc., 2020, pp. 3833–3845. URL: `https://proceedings.neurips.cc/paper_files/paper/2020/file/27e9661e033a73a6ad8cefcde965c54d-Paper.pdf`.