RoboBarista - Autonomy of a Franka Arm for Coffee-Making

Swastik Mahapatra [1], Ankit Aggarwal [2], Aayush Fadia [3] and Shreya Shri Ragi [4]

***Abstract*— This paper presents the design and implementation of an autonomous coffee-making system using a Franka robotic manipulator arm, aimed at advancing automation in the food and beverage industry. The system is capable of preparing various coffee beverages by autonomously detecting, grasping, and pouring ingredients into a user's mug according to specified recipes. Key challenges addressed include careful manipulation of ingredient cups to avoid spillage, ensuring accurate pouring to meet quantitative requirements for each recipe, and localizing all obstacles and the output cup to operate in a mobility-constrained environment. To simulate liquid handling in a controlled setting, colored beads were used as a proxy for liquid ingredients. The methodology integrates computer vision for output cup localization, precise grasping strategies, and constrained trajectory planning using MoveIt, with additional real-time feedback from a weighing scale and force-torque sensors to ensure accurate ingredient dispensing. A custom pouring controller was developed to map cup weight to tilt angle and pour-stop thresholds, achieving pour accuracy within 5 grams of the target amount and a pour success rate exceeding 88% . The system demonstrated robust perception and motion planning, with failures occurring only rarely. This work provides a framework adaptable to other beverage automation tasks, promoting use of robotics in the food and beverage industry. Future improvements include supporting a broader ingredient set, developing an interactive user interface, and refining motion planning to further increase reliability and versatility.**

I. INTRODUCTION

Robotic automation is increasingly transforming the food and beverage industry by improving efficiency, consistency, and personalization. Among its many applications, robotic coffee-making systems are gaining traction due to their ability to deliver high-precision, high-volume service with minimal human intervention. This work focuses on developing a robotic manipulator system that can prepare various types of coffee, such as lattes, cappuccinos, and Americanos, based on user preferences using a Franka Emika Panda arm.

To simulate liquid ingredients in a controlled and reusable setting, we use small colored beads (BB pellets), allowing us to focus on the mechanical and algorithmic performance of the system without the added complexities of liquid handling. The task introduces several challenges: the end-effector must maintain an upright orientation to avoid spillage, the operating space is confined to a compact table set-up, and the robot operates close to its base, where joint mobility is reduced. These constraints limit the effective configuration space and make motion planning using standard tools like MoveIt highly unreliable in this context.

In response, we designed a lookup table-based pouring controller that maps the input cup weight to the tilt angle and a pour-stop output weight. Through real-world trials, we analyzed the pour accuracy and the behavior of the system under different input conditions. We also examine the relationship between workspace layout, constrained motion, and planning success.

There are multiple potential benefits of this system, It can enhance efficiency in high-demand environments by automating coffee-making, reducing labor costs, and ensuring consistent quality. In addition, the system's ability to accurately mix ingredients based on user preferences can help increase customer satisfaction by providing personalized coffee experiences.

Beyond coffee preparation, this robotic manipulator system can further help the beverage industry. The principles developed here can be applied to other complex beverage tasks such as cocktail making, further expanding automation in restaurants and bars.

II. RELATED WORK

Previous works mainly focus on the handling and pouring of liquid using a robotic arm. [1] proposes a multimodal neural network, MP-Net, for robotic pouring by integrating audio and haptic inputs. It uses a weighing scale to measure the weight of the target container, providing ground-truth data to train the network. The scale readings are used to obtain the height of the air column in the container, facilitating an accurate estimation of the liquid height during pouring tasks. This method uses various sensors to develop a robust methodology for pouring. In contrast, [2] proposes a hybrid-pouring robot featuring a four-wheel drive omnidirectional platform and a parallel work arm. It uses a real-time control system combined with visual feedback to pour molten metal into molds. The design offers flexibility and reliability in multi-point pouring operations. This method aims to reduce the complexity of the task by creating a specialized hardware platform, thereby simplifying motion planning. In addition, [3] aims to solve a more general pouring problem by assessing the flow properties of the liquid. It proposes a system for making pancakes, including both the creation of batter and the pouring into a pan. It combines the use of haptic sensing

[1] Swastik Mahapatra is a Master's Student in Robotic Systems Development at Robotics Institute, SCS, Carnegie Mellon University, Pittsburgh, PA 15213, USA

[2] Ankit Agggarwal is a Master's Student in Robotic Systems Development at Robotics Institute, SCS, Carnegie Mellon University, Pittsburgh, PA 15213, USA

[3] Aayush Fadia is a Master's Student in Robotic Systems Development at Robotics Institute, SCS, Carnegie Mellon University, Pittsburgh, PA 15213, USA

[4] Shreya Shri Ragi is a Master's Student in Robotic Systems Development at Robotics Institute, SCS, Carnegie Mellon University, Pittsburgh, PA 15213, USA

and robust control methods to control the pouring process, allowing it to create pancakes of any shape. Furthermore, [4] introduces a vision-based system for estimating liquid volume in transparent containers. Using a two-step CNN architecture, it accurately estimates the liquid height with limited training data, complemented by simulation-driven pouring techniques for small openings. [5] proposes a framework for liquid perception that reduces reliance on pixel-wise annotations by using image-level labels and Class Activation Maps (CAM) to localize liquid regions. It leverages container pose estimation and 3D point cloud recovery for closed-loop robot control, validated on a novel dataset and a Franka robot. Sun et al. provided further exploration of learning-based pouring techniques, who developed a self-supervised method to model pouring dynamics and generalize to new containers, achieving high precision in pouring tasks [6].

III. METHODOLOGY

In this section, we discuss our methodology for our coffee-making simulation. The steps are outlined in Algorithm 1 and are elaborated on in the following. The initial arrangement can be found in Figure 1.

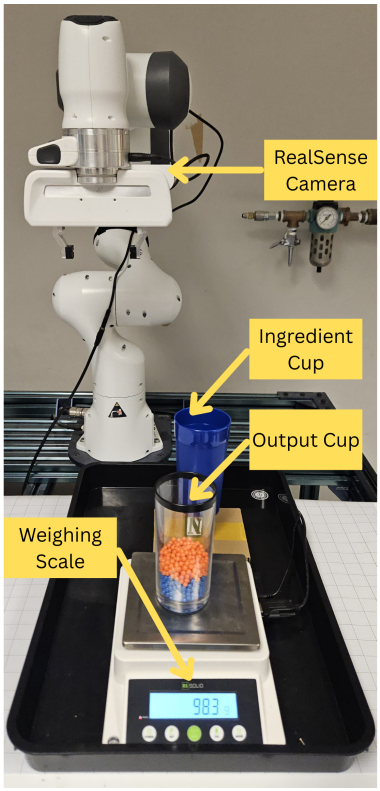

Fig. 1. Hardware Setup

A. Detection and Dimension Estimation of Output Cup

We use the RealSense camera mounted on the robotic arm to locate the coffee mug where the beverage will be prepared. This process involves positioning the arm to ensure that the camera captures the top view of the output platform(weighing scale). From this vantage point, edge

Algorithm 1 Automated Coffee Making Simulation

```
Determine position of user coffee mug
while Ingredients Remaining do
    Determine location of ingredient
    Plan non-colliding path to ingredient cup
    Move to ingredient cup and grasp
    Plan constrained path to above destination cup
    Move arm above destination cup
    while Required weight not reached do
        Tilt ingredient cup to fill coffee mug
    end while
end while
```

detection computer vision algorithms are applied to the RGB image to identify the rim of the mug as shown in Figure 2.

Next, we use the corresponding depth data from the localized pixels to determine physical attributes such as the height of the cup, its distance from the vantage point, etc, to generate a 3D cuboid volume that closely matches the shape of the mug. Finally, using calibration parameters, we translate this data into the global coordinate frame.

Key Applications of This Process:

1) **Precise Pouring:** Localizing the lid area allows accurate pouring of ingredients into the target mug.

2) **Collision Avoidance:** Registering the 3D cylindrical volume of the mug in the global collision check frame ensures that the MoveIt planner generates safe arm trajectories.

3) **Quality Control:** Implementing checks for minimum lid opening area and required mug volume prevents operational errors.

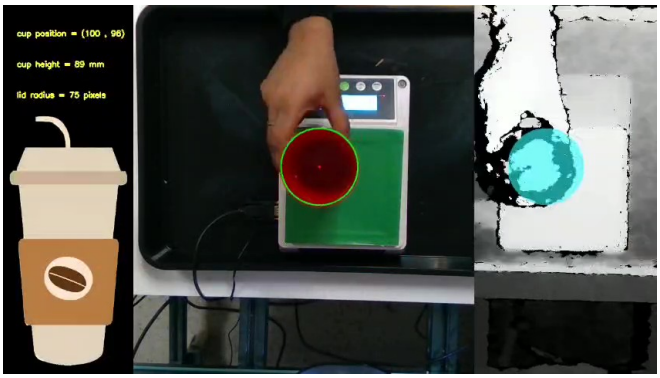

Fig. 2. Localizing Output cup

B. Grasping

The ingredient cups must be picked up with a side grip as shown in Figure 3. This means that the ”elbow” of the arm must protrude significantly to the right side of the base, as shown. This removed the left half of the table from consideration as the elbow would hit the virtual walls (either in this configuration or on the way), thus making grasping impossible.

After requesting and receiving an expansion of the viable robot workspace, we were only able to determine one viable position where the ingredient cups can be grasped without violating joint/collision constraints. To circumvent this, we added a shelf to the workspace. The height of the shelf allowed us to determine 3 viable input cup locations. We placed ingredient cups at these locations and recorded joint angles and poses for pre-grasp poses with the end-effector slightly offset to the cup. Since these cups would be placed in the same position every time, we decided to try simple and repeatable approaches. We used FrankaPy's `goto_joint` function to check whether we could successfully plan a trajectory to our desired pre-grasp positions. This naively planned a trajectory while ignoring constraints and attempted to execute it. However, in the middle of the test, the arm would detect a collision with a virtual wall and freeze. Since the simplest approach did not work, we decided to use MoveIt and its library of sophisticated planners to plan a path that would not result in collisions with virtual walls and other obstacles. This approach is elaborated on in the next section.

Once the pre-grasp position has been reached, we move the arm toward the cup and close the grippers. This results in the cup being gripped firmly in the "fingers" of the end-effector.

Fig. 3. Side-Grip

C. Trajectory Planning

Once the ingredient cup is gripped, it is brought over the coffee mug so that the ingredient can be poured into the mug below. To achieve this, a constrained path that does not allow the cup to tilt is planned to avoid spilling the ingredient along the way. This path also avoids collisions in the same way as above.

This is implemented using FrankaPy and MoveIt Commander. All ingredient cups and the output setup are added to the scene as obstacles using the `add_box` function of the `PlanningSceneInterface` class. This has been visualized using Rviz as seen in Figure 4. A trajectory is planned and executed from the current pose (ingredient cup location) to the goal pose (near the coffee mug rim) using the `plan` and `execute` functions of the `MoveGroupCommander` class.

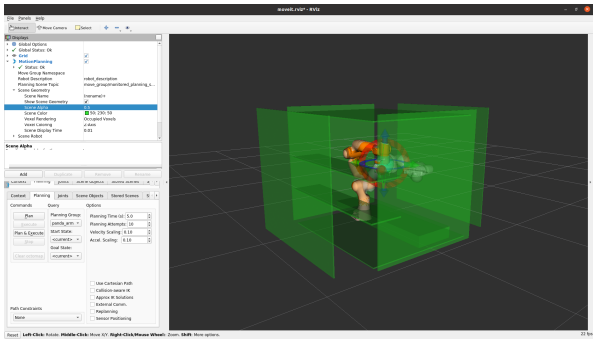

Fig. 4. Collision Objects

Constraints are added to the end-effector pose during trajectory execution to ensure that the cup remains upright throughout the motion, preventing spills. The `set_path_constraints` function and the `OrientationConstraint` `moveit_msgs` type are used to define and enforce this constraint during motion planning. The virtual walls of the workstation are also added to ensure that MoveIt does not plan trajectories outside of FrankaPy's constraints.

Additionally, execution monitoring is performed in real-time using feedback from the robot through MoveIt functions, allowing for potential error recovery and trajectory re-planning if deviations occur.

D. Controlled Pouring

Once the ingredient cup is above the coffee mug, the ingredient cup must be tilted to pour its contents into the mug below. We have discretized the pouring process into pour and no-pour states. In the pour state, the last joint of the arm is actuated to tilt the cup to the desired pour angle. In the no-pour state, the cup will be held upright.

To build feedback into the system, we use 2 sensors - a weighing scale to weigh ingredients in the output cup and Franka's inbuilt force-torque sensor to weigh the input cup. These inputs are used to build the logic of the pour angle and time to pour. We created a lookup table, shown in Table I, based on empirical data that maps the weight of the input cup to a pour angle and a stopping offset. The stopping offset is a subtraction from the target weight to account for the beads being poured while the arm transitions back to the no-pour state. The arm will begin pouring using the looked-up pouring angle and pour until the offset target weight is reached on the weighing scale under the output cup. We

keep the offset values high in an attempt to bias our system to undershoot rather than overshoot.

TABLE I
LOOKUP TABLE FOR POURING ANGLE AND STOPPING OFFSET

Input Cup Weight (g)	Stopping Offset (g)	Pouring Angle(rad)
0 - 30	0	-1.0
30 - 50	6	-0.88
50 - 70	8	-0.88
70 - 110	8	-0.89
110 - 150	10	-0.89

After pouring the required amount of ingredients, we place the ingredient cup in its original place and move on to the next ingredient, if there is one, or else, we notify the user that their coffee is prepared and return the arm to its reset state, awaiting the following order.

E. Customer Order - User Input

We made a .yaml file with standard orders and ratios:

1) **Americano** - 45mL Espresso + 90mL Water
2) **Latte** - 50mL Espresso + 100mL Steamed Milk + 50mL Foam
3) **Cappuccino** - 60mL Espresso + 60mL Steamed Milk + 60mL Foam

These are parsed as an argument to the working script, which accesses the .yaml file to obtain target pour weights corresponding to each input cup. Example usage -

```
python3 CoffeePour.py Americano
```

IV. EVALUATION

We evaluate the performance of our end-to-end system with respect to two main metrics, namely pour amount accuracy and pour success rate. We also evaluate our subsystems individually - we measure success rates for the perception and motion planning subsystems.

A. Pour Amount Accuracy

We observe the amount of ingredient poured into the cup versus the target amount. The results can be seen (trial-wise) in Figure 5, and summarized in Figure 6.

While the exact target amount was not always poured, the actual amount poured was within 5g (above or below) the target amount.

B. Spillage/Pour Success Rate

Rare cases of spillage were encountered while pouring due to beads bouncing off the bottom of the destination mug, clearing the rim, and landing outside. This was observed in three out of twenty-five trials. Overall, bead spillage occurred 12% of the time. This indicates that our system has a success rate of 88%.

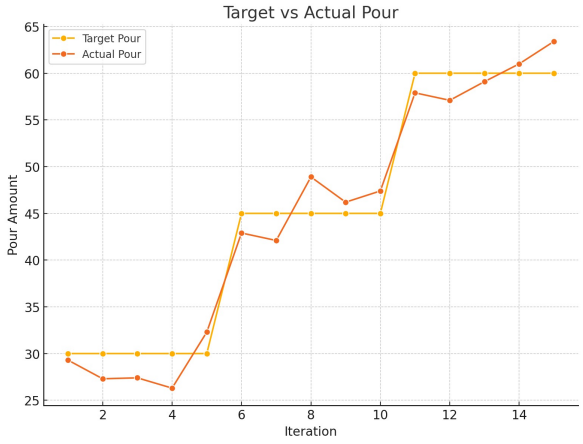

Fig. 5. Trial-wise Target Pour Amount v/s Actual Amount Poured

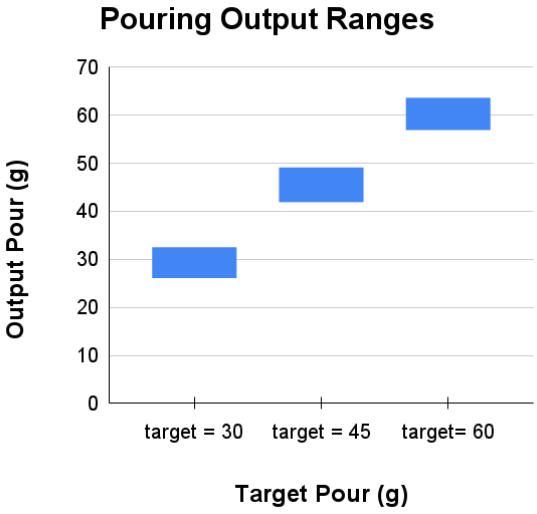

Fig. 6. Range of Actual Poured Amount v/s Target Amount to pour

C. Pouring with Multiple Inputs

Taking an example of the Latte customer order, the system was executed 5 times and was successful 4 times. The failure case is attributed to MoveIt planning errors and hardware faults, as rebooting the system fixed these errors. Fig 7 shows the output cup after a successful run of the system.

V. CHALLENGES

The system requires the end-effector to maintain an upright orientation throughout the motion to avoid spillage. This constraint effectively removes 2 to 3 degrees of freedom (DoF) from the robot's control, leaving only 4 to 5 usable DoF. Since a general 6-DoF task (involving both position and orientation) typically requires at least 6 unconstrained DoF, this severely limits the robot's kinematic redundancy.

The operating environment further restricts the motion. The robot works close to its base, at a low height on a compact table. At these heights, some joints, particularly those acting as shoulders or elbows, reach mechanical limits or

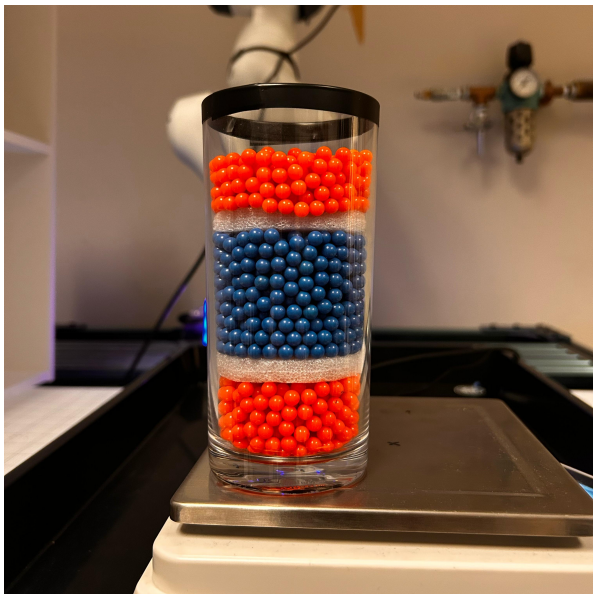

Fig. 7. Output Order - Latte

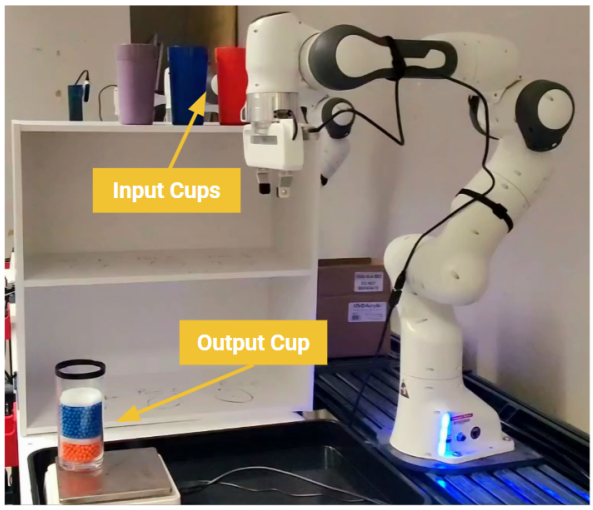

Fig. 8. New positions for input cups

lose the ability to produce useful rotations. This reduces the set of reachable poses and reduces the effective configuration space of the robot.

In addition, the workspace is cluttered with virtual walls, ingredient cups, and a weighing scale, increasing collision likelihood and reducing the valid sampling space for motion planning. As a result, sampling-based planners, such as RRT* and OMPL, often fail to find valid trajectories, especially for constrained motions.

Due to these combined factors, only one specific spot on the table allows the robot to successfully complete the entire sequence: pick up the ingredient cup, move to the output cup, pour while maintaining orientation, and return the cup. This limitation could be mitigated in a larger or more open workspace.

Alternatively, we were able to use a vertical shelf to add find 3 input cup locations, which were much higher than the operating platform. This, however, added another obstacle to an already obstacle-dense environment. Fig 8 shows the new system configuration that was used for the stretch goals

VI. CONCLUSION

We successfully developed and implemented an autonomous coffee-making system using a Franka robotic manipulator arm. The system demonstrated capabilities in precise cup detection, accurate grasping, and controlled pouring, effectively addressing the significant challenges posed by a constrained workspace and stringent orientation requirements. By employing a combination of computer vision, constrained motion planning with MoveIt, and real-time sensory feedback, we achieved pour accuracy within ±5 grams of the target amount, maintaining a pour success rate of 88% despite the inherent complexities of the task.

This study emphasizes the viability and possible benefits of robotic automation in food and beverage contexts, focusing on chances for greater efficiency, consistency, and consumer satisfaction through tailored beverage preparation. Furthermore, the architecture and methodologies proposed herein are easily scalable to broader industrial applications, including more complicated activities like cocktail preparation.

VII. FUTURE WORK

In the future, more positions on the shelf could be potential input cup locations to create more complex coffee orders - frappuccino, mochas, etc. Syrups and flavorings could also be added to make a complete coffee order.

Further refinement of the pouring method and tuning the lookup table to be more robust will help to create a more reliable system. Other pouring methods, such as reinforcement learning or trajectory optimization, can also be explored.

To expand the project's use case, the created system can be adapted for several other beverage-based tasks, such as tea, milkshakes, and cocktail mixing.

ACKNOWLEDGMENT

We express our profound appreciation to the Robotics Institute at Carnegie Mellon University for providing the necessary resources and environment for this research. We are particularly grateful to Prof. Oliver Kromer, the instructor of the 16662 Robot Autonomy course, whose expert guidance was essential during this project's development. Additionally, we thank our teaching assistants, Shrudhi Ramesh Shanthi and Madhusha Goonesekera, for their dedicated support and valuable feedback, which were critical to our success. Their efforts were key in addressing the challenges of designing and implementing robotic systems.

REFERENCES

[1] Liang, Hongzhuo and Zhou, Chuangchuang and Li, Shuang and Ma, Xiaojian and Hendrich, N. and Gerkmann, Timo and Sun, Fuchun and Zhang, Jianwei. (2020). Robust Robotic Pouring using Audition and Haptics. 10.48550/arXiv.2003.00342.

[2] W. Chengjun, D. Hao, and L. Long, “Design, simulation, control of a hybrid pouring robot: enhancing automation level in the foundry industry,” Robotica, vol. 42, no. 4, pp. 1018–1038, 2024. doi:10.1017/S0263574723001881

[3] Luo, Xinyuan and Jin, Shengmiao and Huang, Hung-Jui and Yuan, Wenzhen. (2024). An Intelligent Robotic System for Perceptive Pancake Batter Stirring and Precise Pouring. 10.48550/arXiv.2407.01755.

[4] Zhang, Chenxi, Xu, Yuan, and Li, Shuang. “Learning-Based Robotic Pouring using Vision and Simulation,” arXiv:2404.16529v1, 2024. https://arxiv.org/html/2404.16529v1

[5] Haitao Lin, Yanwei Fu and Xiangyang Xue. ”PourIt!: Weakly-Supervised Liquid Perception from a Single Image for Visual Closed-Loop Robotic Pouring,” Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2023, pp. 241-251. https://doi.org/10.48550/arXiv.2307.11299

[6] Huang, Yongqiang and Wilches, Juan and Sun, Yu. ”Robot gaining accurate pouring skills through self-supervised learning and generalization,” Journal of Robotics and Automation, vol. 30, no. 4, pp. 312–321, 2020. doi:10.1016/j.jra.2020.10.001