# CS225A Final Project Report: CookieBots


Aditya Dutt
*Mechanical Engineeering*
*Stanford University*
asdutt2@stanford.edu

Mallika Parulekar
*Computer Science*
*Stanford University*
mallika.parulekar@stanford.edu

Aarya Sumuk
*Mechanical Engineering*
*Stanford University*
asumuk@stanford.edu

Harshvardhan Agarwal
*Computer Science*
*Stanford University*
hvag976@stanford.edu



***Abstract*—Automating tasks in unstructured environments like cooking inside a kitchen is a challenging problem to solve using robots. In this project, we defined a specific task: making cookies, and attempted to solve the various challenges that come with it. We break down the task into 3 specific sub-tasks: (i) Cookie cutting, (ii) Placing onto a Tray, (iii) Cookie flattening. To solve the task, we present three innovative end-effectors that are multi-purpose and are designed specifically to solve the above three subtasks. We also introduce a vision pipeline in order to reliably detect cookie dough and use it in order to guide robot movement and cookie cutting and flattening. We validate our approach through real-world deployment on two Stretch3 robots.**



***Index Terms*—deformable, end-effector, segmentation, cutting, placing, flattening**


$$\begin{pmatrix} x \\ y \\ z \\ \theta \end{pmatrix} \tag{1}$$

## I. Robot Description

### A. Stretch 3 Robot

The Stretch 3 robot platform, produced by Hello Robot [1], is a mobile manipulation system equipped with a 7-degree-of-freedom (DOF) robotic arm mounted on a non-holonomic mobile base. It is designed for a wide range of domestic and assistive robotics tasks, such as grasping objects or opening doors. The robot is equipped with two cameras: a Gripper RGBD Camera and a Pan-tilt Head RGBD Camera. Since the pan-tilt head camera suffered from occlusions from the arm and given that we want precise, local vision, we opted to use the wrist camera. We also designed tools to be gripped so that the camera could clearly see the workspace.

While the Stretch platform is highly versatile and well-suited for mobile manipulation in constrained environments, it does have limitations. One of the primary weaknesses is the gripper, which has limited gripping force and cannot apply significant torque. This restricts its ability to manipulate heavy or tightly fastened objects and makes it less effective in tasks requiring firm or precision grasps. As a result, careful task design, object selection and end-effector tool designs are necessary to accommodate the hardware constraints.

### B. Tools

To compensate for the Stretch3's limited grip strength and torque, we developed three interchangeable, multi-functional tools tailored to our cookie-making pipeline. All tools were designed in SolidWorks and 3D printed using the printers at the Stanford Robotics Center.

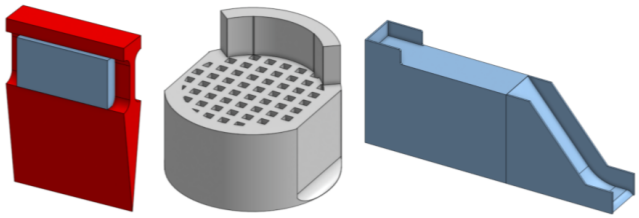

Fig. 1. Custom tool designs: (a) **Wedge**: Tapered cutting implement with ergonomic indents and padded grips to ensure slip-resistant handling. (b) **ScoopPress**: Versatile pick-and-place and flattening tool featuring a raised guard rail to secure slices during transfer and a press surface for uniform thickness. (c) **Slide**: Fixed fixture that stabilizes the dough during cutting and provides a guided incline for smooth transfer between tools, minimizing robot collisions.

1) **Wedge:** A precision cutting implement with a tapered blade edge for clean cookie outlines. Ergonomic indentations and sponges for padding on the top surface provide a secure grasp and prevent slips. The vertical mounting orientation guarantees an unobstructed, top-down view from the wrist camera, enabling accurate alignment and consistent cuts.
2) **ScoopPress:** A combined pick-up and flattening tool designed to receive freshly cut dough slices. A rear guard rail retains each piece during transport, while the flat underside serves as an adjustable press plate for uniform thickness. Rotating the end-effector along its negative pitch axis facilitates seamless placement onto the baking tray.
3) **Slide:** A stationary guide fixture that holds the uncut dough in place and channels sliced pieces along a sloped ramp. Integrated guard rails ensure collision-free transfers from the Wedge to the ScoopPress, reducing the need for complex handoff maneuvers.

## II. Environment Description

We did not work in a simulation environment, but rather worked directly with a real environment. Upon the table, we place our slide, with the cookie dough in the flat section at the top. Behind the slide, we place a baking tray, where the cookies will be placed and flattened. We position one Stretch Robot (hereby known as the Cutter) parallel to the slide and

the other Stretch Robot (hereby known as the Placer) parallel to the tray. These robots are perpendicular to one another. This setup is shown in Figure 2.

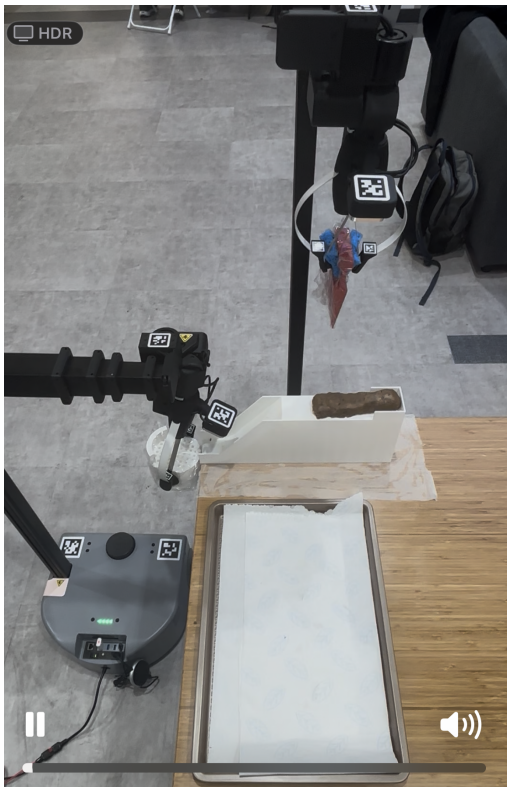


Fig. 2. CookieBot Environment Setup

## III. State Machine and Controllers

### A. State Machine

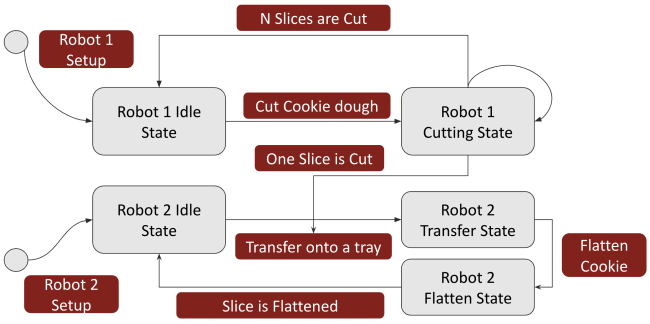


Fig. 3. State machine diagram illustrating the coordination between Cutter and Robot 2 for automated cookie preparation. Cutter is responsible for cutting dough slices, while Robot 2 transfers and flattens them. The transitions are triggered by discrete events such as setup completions, cutting actions, returning to idle state status, and flattening confirmations.

Figure 3 illustrates a state machine involving two robots that collaboratively prepare cookies by cutting, placing and flattening dough slices. The process begins with the setup of Cutter, transitioning into the *Idle State*. Once initialized, Cutter enters the *Cutting State* upon receiving a signal to cut cookie dough. The robot continues cutting slices in a loop until a predefined number $N$ of slices are prepared. Each time a slice is cut, Cutter remains in the cutting state, returning to the *Idle State* only when all $N$ slices are completed.

Each individual slice cut by Cutter is passed to Placer for further processing. Placer starts in its own *Idle State* post-setup. Upon receiving a slice, it transitions to the *Transfer State* where the slice is moved onto a tray. Subsequently, the system triggers a transition to the *Flatten State* where Placer flattens the cookie slice. Once the slice is flattened, Placer returns to the *Idle State* to await the next slice. This orchestrated handoff between robots ensures a continuous workflow from dough cutting to cookie placing and then shaping.

### B. Perception

*1) Grounded SAM (GSAM):* The backbone of our perception system for both our cutting and placing robots is our Grounding Dino and SAM pipeline. Grounding DINO [2] is a vision-language object detection model that detects objects in images based on free-form natural language queries. It combines a powerful visual backbone with a language-guided object detection head to output bounding boxes for described objects, with an open vocabulary. SAM (Segment Anything Model) [3], is a segmentation model that generates high-quality masks for objects in images. It takes spatial inputs such as a point or bounding box and outputs a precise segmentation mask for the corresponding object.

In our project, we prompt Grounding DINO using ”cookie dough” (for the cutting task) or ”cookie chunk” (for the flattening task), which provides us with bounding box predictions. We then feed those bounding boxes into SAM, which predicts the relevant masks. This process is highlighted in Figure 4.


Fig. 4. Grounding DINO and SAM Pipeline

*2) Cutting Task:* We can split the perception stack of the cutting task into two steps:

- First, we use Grounded DINO+SAM to obtain the segmentation mask of our cookie dough using the images received from the wrist came at a rate of 10Hz. As seen in the first image of Figure 4, the segmentation is robust to occlusions such as Cutter’s arm. This allows us to rely on the segmask even in the presence of partially-occluding clutter in the scene.
- Second, we generate a sequence of candidate cutting points (middle image in Figure 4) on the cookie dough mask, out of which the most feasible cut point is selected. In our case, we choose the second cut point from the right

side in the image frame, in order to prevent cutting the dough too close the edge.
- Finally, the selected point is reprojected to 3D and transformed to Cutter's base frame before being passed into the controller.


Fig. 5. Breakdown of the Perception stack of the Cutting task. First, the segmentation mask of the cookie dough is obtained from the wrist-camera (left). Next, a sequence of N evenly spaced cut points are obtained (middle). Finally, the second point from the right is selected and passed to Cutter for control (right).

*3) Placing Task:* We use Placer (Robot 2) to place the cookies onto set positions on the tray in an open-loop fashion. Given that we know the environment setup, we use pre-programmed primitives to determine how far to move the base and arm in each direction in order to successfully place each cookie. Each subsequent cookie's position is determined as a fixed distance to the right of the previous cookie.

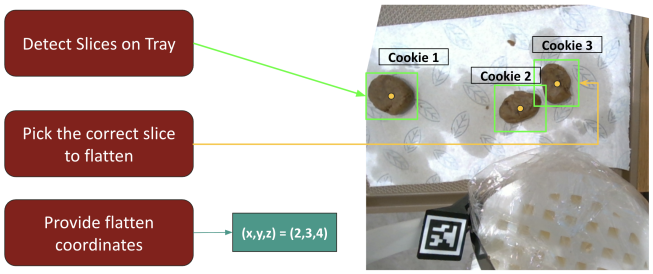


Fig. 6. Instances of Cookie chunks are detected on the tray using Grounded SAM. The rightmost cookie in the frame is chosen to be flattened. The desired position for the tool, i.e the center of the cookie, is sent to Cutter.

*4) Flattening Task:* For the flattening task, we use our Grounded SAM Model to receive detections of cookie chunk instances on the tray. In order to isolate the cookie we want to flatten, we decided to use a fixed method by selecting the cookie whose center is farthest to the right in the image frame. A more robust approach to this is mentioned in the Future Works section. Finally, we use the center point of the selected cookie as the desired position of our end-effector to move to for flattening the cookie.

### C. Trajectory Planning and Control

Due to the unique design of the Stretch Robot, we resort to simple Cartesian Control as each joint is prismatic while the base has only has 2 degrees of freedom. We implement quintic polynomial trajectory optimization ensuring smooth, damage-free manipulation. The fifth-degree polynomials provide six degrees of freedom enabling specification of position, velocity, and acceleration at trajectory endpoints:

$$\theta(t) = a_0 + a_1 t + a_2 t^2 + a_3 t^3 + a_4 t^4 + a_5 t^5$$

Zero boundary conditions ($\dot{\theta}_0 = \dot{\theta}_f = \ddot{\theta}_0 = \ddot{\theta}_f = 0$) ensure smooth starts and stops, minimizing jerk and preventing sudden accelerations that could damage delicate cookie dough during the placement onto the tray and while pushing the sliced cookie dough onto the Slide.

### D. Motion Primitives

To modularize the system and promote reusability, we define two classes: `CookieCutter` and `CookiePlacer` (which includes the flattening functionality). Each class encapsulates a set of motion primitives implemented on the Stretch robot. These primitives are executed using smooth minimum-jerk trajectories described above to ensure precision and to prevent damage to the soft cookie dough.

*a) CookieCutter:* is responsible for performing the full cutting motion, including vertical and horizontal actuation of the cutting tool. The following primitives are defined:

- `min_jerk_trajectory(start, end)`: A shared utility function that generates a time-indexed trajectory for smooth transitions in joint space or Cartesian motion.
- `smooth_lift_motion(z_start, z_end)`: Moves the lift vertically from `z_start` to `z_end` using a quintic minimum-jerk trajectory to avoid sudden accelerations. This is used to cut the dough, and we use this by cutting from a height (setting `z_start` to max height) in order to increase force.
- `smooth_base_translate(dxy)`: Smoothly translates the robot base forward or backward along a single axis using a minimum jerk trajectory. This is used to push the dough down the slide and then return to the next cutting position.
- `cut_once()`: Executes one full cut by combining a lift descent, a base push through the dough, a lift ascent, and a return to the next cutting position. Each substep uses smooth motion to ensure a clean cut and maintain hardware stability.

*b) CookiePlacer:* handles picking up the cut dough, positioning it above the tray, and depositing it gently as well as flattening it. In addition to also having the `min_jerk_trajectory(start, end)`, `smooth_base_translate(dxy)` and `smooth_lift_motion(z_start, z_end)` functions, it includes the following primitives:

- `drop_cookie(pitch_angle)`: Rotates the wrist pitch downward to gently release the cookie onto the tray using a smooth interpolation of wrist joint angles.
- `undrop_cookie(pitch_angle)`: Returns the wrist pitch to a neutral post-drop configuration. This prepares the robot for subsequent movements.

- `return_home(place_distance)`: Retracts the arm and base, moving the robot to its initial configuration, ready to catch the next cookie.
- `place_cookie(place_distance)`: A function that executes the entire placing sequence: retracting the arm slightly, raising the lift, translating the base forward to the tray, lowering the lift to the deposit height, releasing the cookie via wrist pitch (drop), lifting back up (undrop), and moving the arm to clear the camera's view for the flattening task.
- `flatten()`: Lowers the cookie press to compress the dough, holds momentarily, and then lifts back up.

### E. Robot Communication

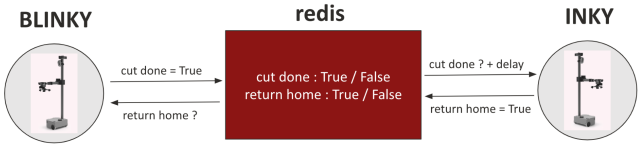


Fig. 7. Communication protocol between Cutter (Robot 1) and Placer (Robot 2) using Redis as a shared data store. The two robots use boolean flags (`cut done` and `return home`) to signal task completions and synchronization points in the workflow.

Figure 7 illustrates the communication protocol between the two robots, Cutter and Placer, mediated through a Redis server [4]. Cutter, after completing a dough cutting task, updates the Redis store with a flag indicating that the cut is done (`cut done = True`). Placer periodically polls Redis to check whether this flag has been set, potentially applying a delay to synchronize with the timing of operations. Upon confirming that a cut is completed, Placer proceeds with its task and subsequently writes back to Redis with a `return home = True` signal.

This return signal informs Cutter that it may now reset its state or return to its home position. Cutter reads this flag from Redis to confirm that the downstream process (handled by Placer) has completed successfully before proceeding. This architecture provides a lightweight but effective mechanism for coordinating multi-robot workflows by leveraging Redis as a central shared memory. The system design ensures that both robots remain decoupled in execution but synchronized through shared flags representing task completion and readiness.

## IV. Human Interface and Intervention

The system incorporates human oversight capabilities recognizing that full automation may encounter edge cases requiring intervention. Our interface provides both monitoring and direct intervention capabilities.

Human operators can observe system status through visual feedback showing robot states, task progress, and detected objects. The interface displays camera feeds with overlaid detection results, enabling quality assessment throughout the process. During our experiments we found that there are cases when the cookie doesn't roll off the slide into the catcher or it sometimes land very close to the edge of the tray, obstructing the cookie flattening process. So we designed the workflow in such a way that the system pauses automatically before the flattening step and only continues with the process after human confirmation.

The intervention design philosophy balances automation efficiency with human oversight, enabling the system to handle routine operations while gracefully managing exceptional cases through human collaboration.

## V. Challenges & Future Work

### A. Current Limitations

Several technical challenges emerged during development and deployment:

**Force Sensing Limitations:** The Stretch 3's limited force sensing ($\sim$ 5N resolution) constrains delicate manipulation feedback. Current sensing provides only crude force estimation, limiting adaptive responses to varying dough properties.

**Singularity Positions:** The robot's kinematic configuration occasionally encounters singular positions limiting manipulation workspace, particularly during extended reach operations.

**Vision Occlusions:** Overhead camera views suffer from arm occlusions during manipulation, requiring careful coordination of robot positioning and camera placement.

**Deformable Object Handling:** Cookie dough's varying consistency (based on time since refrigeration) and deformable nature poses a challenge across both perception accuracy and manipulation planning.

### B. Future Work

We have several ideas for how we can extend and make this system more robust:

**Vision-Guided Placement:** Use the wrist RGBD camera (or potentially the overhead camera) in a closed-loop visual servoing pipeline to precisely align and place dough pieces, rather than our current open-loop operation, to generalize for varying tray locations.

**Drop/Release Optimization:** Explore alternative end-effector tooling or novel motions (e.g. gentler tilt) to ensure release of dough without rolling, to create a uniform pattern of cookie dough placement in the tray. Alternatively, if this proves difficult, add a step in the state machine to rearrange the cookie dough after placement.

**Memory Bank of Flattened Cookies**: Implement a more robust pipeline for selecting which cookie dough placed on the tray needs to be flattened. The current approach will fail if the cookie chunk that was placed most recently is not the rightmost chunk on the tray (placed above or below the previous chunk). In such scenarios, one could use a Memory bank of previously flattened/placed cookies that can be compared against during the object instance detection step.

**Robustness to Dough Variability:** Extend the system to characterize dough stiffness online (e.g. via brief probing motions or computer vision). Then, automatically adjust grip

force and choose different actions for cutting, placing or flattening based on characterized stiffness.

**Alternative Mobile Manipulator Evaluation:** Benchmark a more powerful mobile manipulator (e.g. the TidyBot) to overcome the Stretch 3's force and torque limits, and compare performance on the same dough-handling tasks.

## VI. Link to Demo Videos

- One Cut and Place Demo: 1x Speed, 3x Speed
- Full Demo (3 Cuts, 3 Places): 1x Speed, 3x Speed

## References


[1] C. C. Kemp, A. Edsinger, H. M. Clever, and B. Matulevich, "The design of stretch: A compact, lightweight mobile manipulator for indoor human environments," 2022. [Online]. Available: https://arxiv.org/abs/2109.10892

[2] S. Liu, Z. Zeng, T. Ren, F. Li, H. Zhang, J. Yang, Q. Jiang, C. Li, J. Yang, H. Su, J. Zhu, and L. Zhang, "Grounding dino: Marrying dino with grounded pre-training for open-set object detection," 2024. [Online]. Available: https://arxiv.org/abs/2303.05499

[3] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick, "Segment anything," 2023. [Online]. Available: https://arxiv.org/abs/2304.02643

[4] D. Eddelbuettel, "A brief introduction to redis," 2022. [Online]. Available: https://arxiv.org/abs/2203.06559