Vikki: Judges, we developed our robot based on three design principles: Reliability, repeatability, and stability, by focusing on the error handling strategies.

We use both hardware and software solutions to achieve the goals.

Hardware:

- Core: **Chloe D.** We went through multiple design iterations and came to this final compact design. We added extra pins and connectors to improve the robot's structural integrity.

- Forklift dumper: **Chloe W.** We developed a unique, and innovative attachment inspired by the forklift and dumper idea. It is driven by a large motor through the parallelogram linkage mechanism. Our innovative solution is to connect the Wind Turbine and the Toy Factory missions in the same run. As you can see, the robot collects the energy cells from the wind turbine when the arm is low. And the cells can be released to the Toy factory when the dumper tilt around the pivot point when the motor lifts it up to the desired height. This makes our attachment both a passive and an active attachment. We also used this attachment to multitask many other missions.

- Software:
  - PID: **Gabby** One of the coding strategies we used extensively in our missions is the Gyro-PID. We use gyro-PID strategy to ensure the robot running in straight lines. We also combined PID with speed change, acceleration, and deceleration, to prevent drifting and ensure the stability of the robot.

    - Color sensors: **Melody** There are two color sensors in the front area of our robot. Through our testing, we learned that the motor rotation error will build up proportionally to the traveled distance. So, we use the centre color sensor to detect a line and then obtain new references to reduce the proportional errors.

    - **Renee** During our testing, we found that even with the PID and line detection strategies, our robot may not position itself to the landmark repeatably, especially for our Run 1, where our robot runs for about 70 seconds to clear 8 missions. So, we made an **advanced** line squaring strategy, by using two color sensors in the offset position to further improve the robot position accuracy.

  - **Jacqueline** In addition to improving the accuracy of robot positioning, we also use the Stall detection strategy to recalibrate the starting position of the attachment. For example, this main arm is very critical to many missions, so it has to position to the desired height repetitively to improve the reliability.

Mission Planning

- **Ariel** For the robot missions, we have 4 runs in total. The first run is the most critical one since it takes about 1 minute and 10 seconds to finish and needs to clear 8 missions in the North and Central areas (Show a path map). As a matter of fact, we spent most of our time testing this run to ensure a good successful rate. This kind of feedback from our testing helps us improve the robot design in both hardware and software.

- **Vikki** For example, we developed the Advanced Line Squaring strategy based on this feedback-testing process. The advanced line squaring strategy is a good way to pinpoint a specific spot on the map, as both colour sensors look for different lines, pinpointing its location repetitively.

Conclusion

Vikki Judges, this is the conclusion of our robot design presentation, and we are delighted to answer your questions now.

You said you used PID a lot; what is it?

- It is an error feedback algorithm that helps the robot go straight. P is proportional, is the current error. It is integral, is the sum of the historical errors. D is derivative, is the future errors we might make. We combine all of these errors to help our robot move straight reliably.

So what's the coefficient of the PID?

- Our coefficient for Proportional is 0.6. We found this by isolating the P because the P is the most important and relevant error. We did trial and error and observed the performance of our robot. We started with one and slowly changed the number higher or lower to find out the best coefficient. In the end, we found that 0.6 works well as it oscillates nicely. Then we moved onto our integral coefficient. The integral coefficient is 0.0014. Because integral adds all the historical errors together, it will be too large of an error to be represented in the code. Because of that, we had to scale it down significantly so that it could be represented in our code. Our derivative is set at 0 because it is hard to predict and sometimes causes even more error than how it started out. However, we acknowledge that no matter how good our PID is it will still have errors. Therefore we made the funnel.

You talked about your advanced line squaring method. What is that?

- Line squaring: Unlike the traditional line squaring method, which uses two color sensors that are along the same line each other, we used two color sensors that are offset for our advanced line squaring. For example, our robot uses the gyro assisted PID when running. There can be errors in that, so we also use a line follower done by the color sensor to make our robot more accurate. We use the center color sensor to follow the edge of the line. We move until the other color sensor detects the edge of the T. This way we can more accurately position our robot than the traditional line squaring. The problem with traditional line squaring is that they can stop at any point along a line and still be perpendicular to it. With ours, it will be perpendicular and still be at the same point every time, making our robot more repeatable.