Hochschule Kaiserslautern
University of Applied Sciences

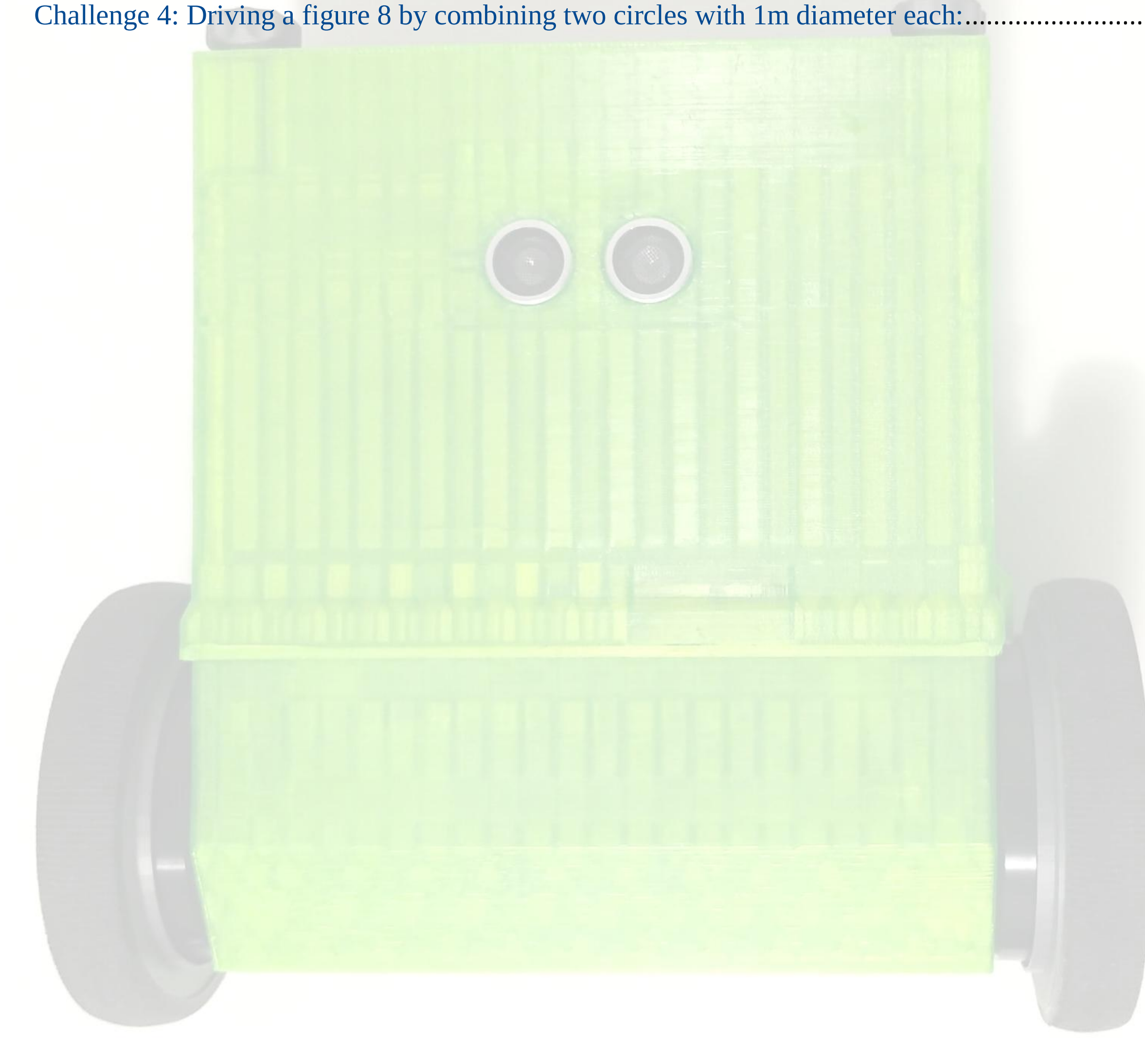
M.Sc. in Mechatronics Engineering

System level rapid development in mechatronics
WS 2023/24

(Milestone-3)

Prepared by
Ganesh Dola

Table of Contents



Section1: The challenges

Introduction:

The main goal of this report is to perform the four major tasks/challenges given to the master by using a two wheel robot with the help of Matlab/Simulink software. Those tasks are mentioned as below and explained in detail:

1. Driving as fast as possible a distance of 3m.
2. Similar to 1. But additionally a 360° turn at about 1m and 2m travelled.
3. During balancing an additional weight is added and the robot compensate it (stay/return).
4. Driving a figure 8 by combining two circles with 1m diameter each.

Challenge 1: Driving as fast as possible a distance of 3m:

- In the autonomous target block, two states are created inside a chart named as "**OP_mode_stop_and_go_5**". To run this chart, in "mode set logic block" modifications are made accordingly.
- From the below mentioned states it can be observed that after turn on the robot, it is stabilizing for 1.5 sec and later attaining some velocity with respect to time.
- The parameters are derived from the basic formula:
  - ❖ V=D/T

  Where;

  V= Velocity (m/s)
  D=Displacement (m)
  T= Time (Sec)

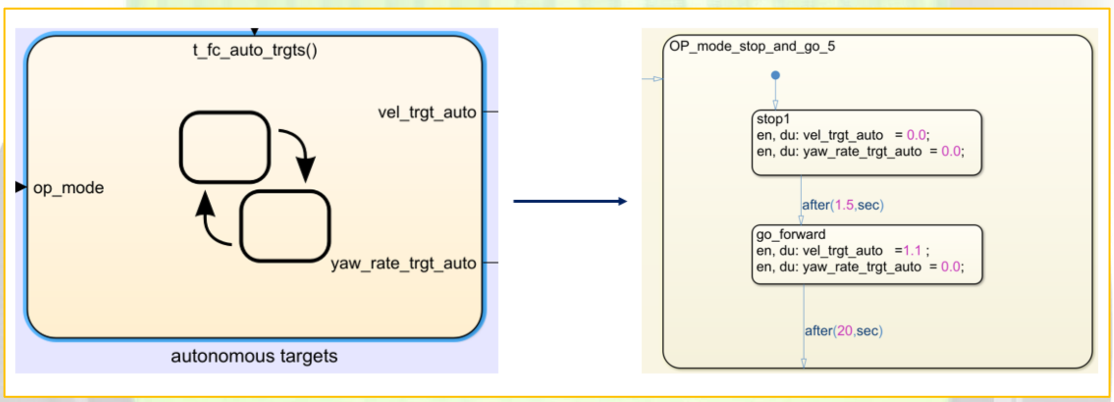


Fig: Autonomous targets for 3m run

- In the State control block , the integrator saturation limits are set to:
  - Upper limit: +1
  - Lower limit: −1

  This constrains the output signal range for safe motor control.
- Parameters in the yellow block are updated to:
  [-1.5, 12, 1.0] - tuning control response and acceleration behavior.
- The mean angle1 is set to:
  1*whl.rad - defining the wheel rotation reference for motion planning

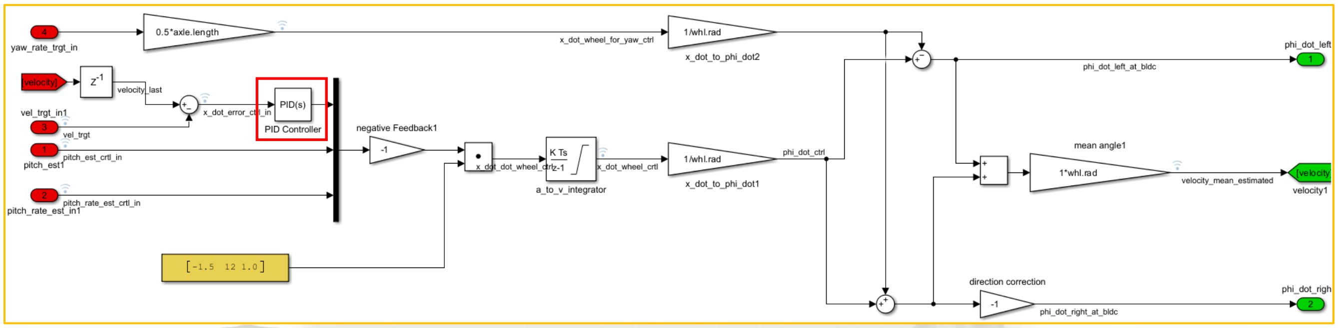


Fig: my control BLDC2 block diagram in state control block

- To enhance velocity control during the 3-meter sprint challenge, a PID controller is introduced into the motor control logic.
- The PID block is placed after the summation block in the "**my_control_BLDC2**" subsystem to fine-tune the velocity target signal.

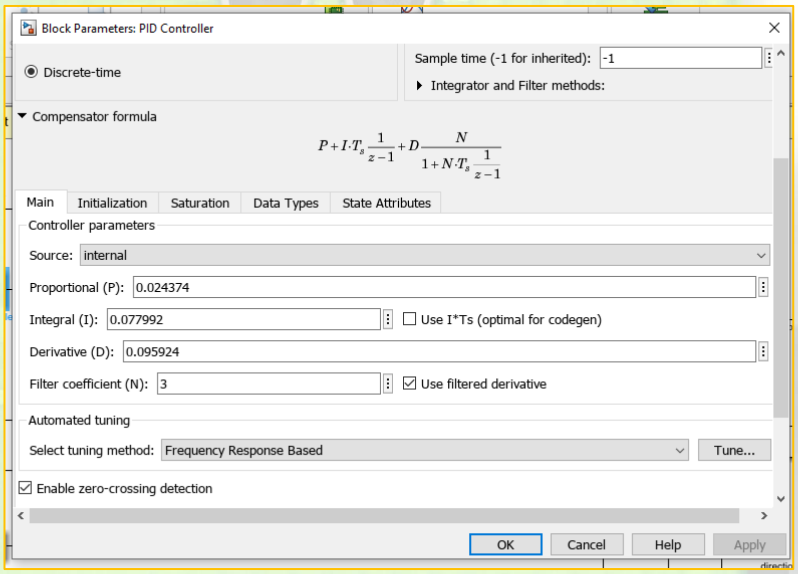


Fig: PID controller values

- The controller is tuned using the Frequency Response Based method, which ensures stability and responsiveness across a range of operating conditions.
- A 3-meter track is laid out using a measuring tape. Markers are placed at 1m, 2m, and 3m intervals to monitor progress and turning points.

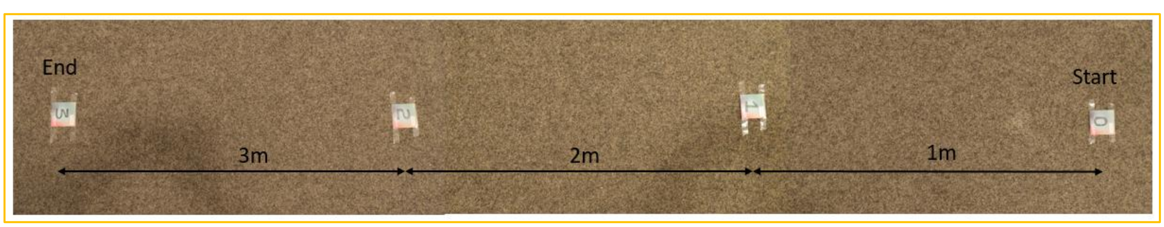


Fig: 3m track to run the robot

- The control algorithm and autonomous logic were successfully deployed to the two-wheel robot.

- Upon activation, the robot executed the programmed sequence and completed the 3-meter run in approximately 4 seconds

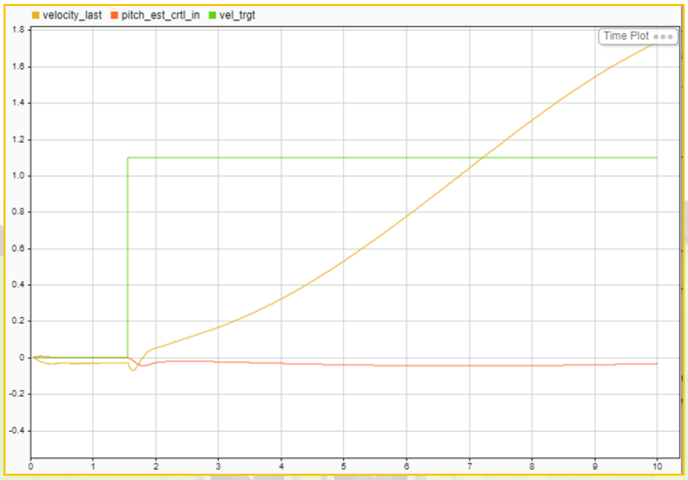


Fig: Graph for various parameters

Performance Analysis:

- *Velocity_last* - actual velocity of the robot
- *pitch_est_ctrl_lin* - estimated pitch angle from the controller
- *vel_tgt* - target velocity set by the autonomous block

These signals confirm that the robot maintained stability while accelerating toward the target, validating both the control logic and tuning strategy.

- **A video demonstration of the 3-meter sprint is available here: https://drive.google.com/file/d/1zpq5zcaVpGd0MC8EgwzDZjcBtcITYLC5/view?usp=sharing**

Challenge 2: Similar to 1. But additionally a 360° turn at about 1m and 2m travelled:

Extend the 3-meter sprint challenge by adding two 360° turns one after traveling 1 meter and another after 2 meters. The robot must maintain stability and resume motion after each turn.

- A dedicated Stateflow chart named “**OP_mode_stop_and_go_3**” is created within the autonomous target block.
- The chart includes multiple states to manage:
  - Linear motion
  - Pause and stabilization
  - 360° rotation
  - Resumption of motion

Control Logic:

- After traveling 1 meter, the robot pauses for 0.1 seconds to stabilize before executing the first 360° turn.

- The same logic is applied after the 2-meter mark, ensuring consistent behavior.
- Velocity is calculated using the same relation as in Challenge 1.
- At the 3-meter endpoint, the robot enters a final stop state with a brief stabilization period to complete the task smoothly.

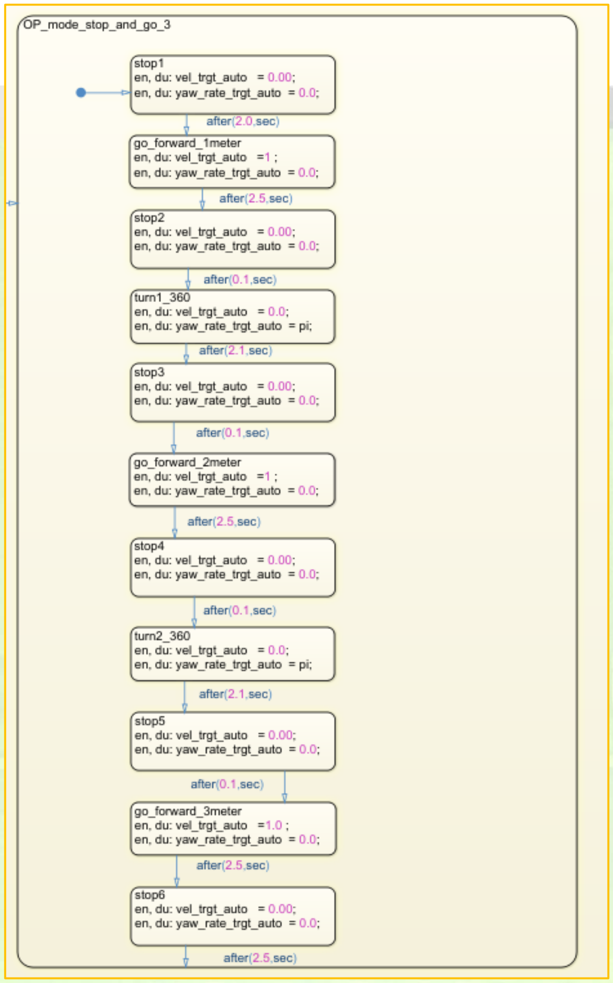


Fig: Autonomous targets

- In the **my_control_BLDC2** block:
  - **Integrator saturation limits** were set to **+1** and **−1** to constrain motor output.
  - Control parameters were updated to: [-1.5, 1.2, 1.0] — optimizing responsiveness and turn precision.
  - **Mean angle** was defined as $\pi/4$ rad to guide rotational behaviour.

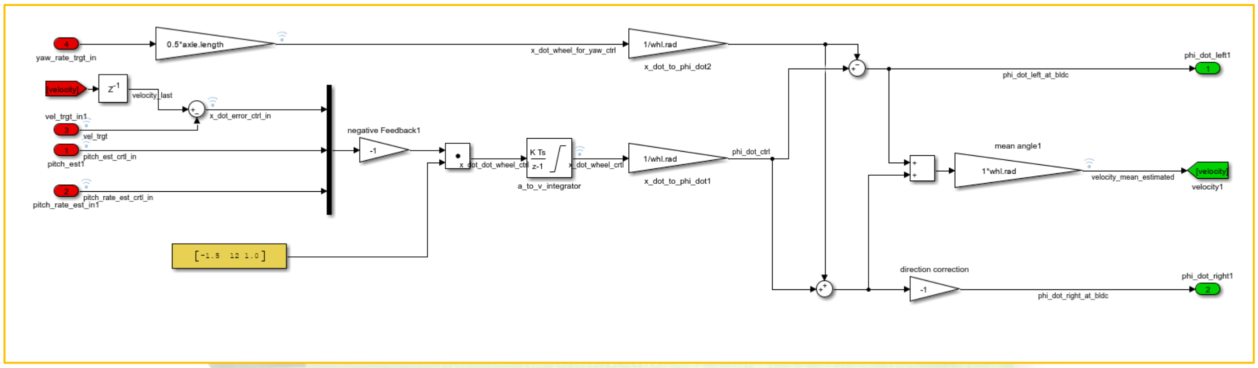


Fig: my control BLDC2 block diagram in state control block

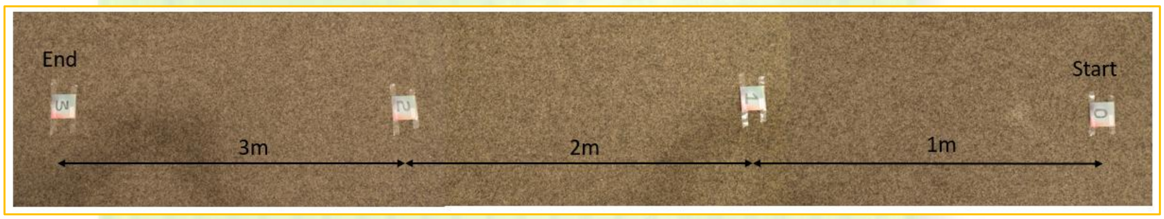


Fig: Linear and 360° turn path

- The control logic for Challenge 2 was successfully **deployed to the two-wheel robot**.
- The robot executed the programmed sequence, completing the task in approximately **13 seconds**.

Fig: Graph for various parameters

The graph displays key control and sensor parameters over time, including:

- Velocity tracking
- Pitch estimation
- Target velocity signals

These plots confirm that the robot maintained stability during both linear motion and rotational transitions.

- **Watch the robot complete Challenge 2 here:** https://drive.google.com/file/d/1M5yEZeqM_xkaHVH9_e3tv-ttZL6tiLvd/view?usp=sharing

Challenge 3: During balancing an additional weight is added and the robot compensate it (stay/return).

- Unlike previous challenges, this task uses a single balancing state implemented in the **challenge_weights** chart.
- The robot operates in neutral mode, with velocity and yaw rate targets set to zero, focusing solely on position control.

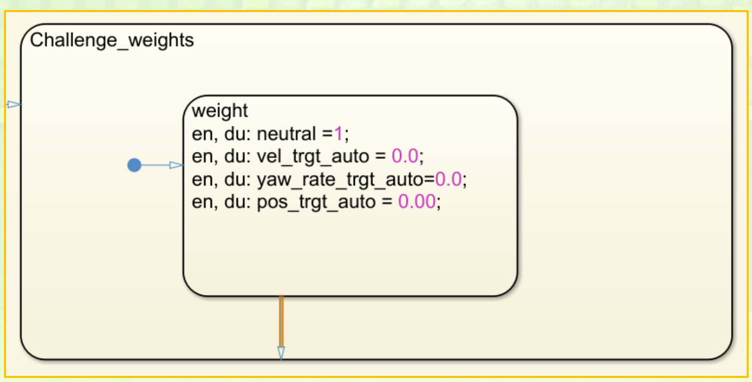


Fig: Autonomous targets configuration

Controller Adjustments:

- In the my_control_BLDC2 block:
  - Integrator saturation limits are set to +1 and −1 to constrain motor output.
  - Control parameters are updated to:
    - [-2.4, 14, 1.0] — tuned for enhanced stability under load.
  - Mean angle1 is defined as **0.5*whl.rad** to reflect the shifted center of mass.
- A switch block is introduced to toggle between velocity-based and position-based control depending on the task.

PID Tuning Method:

- The "**Ziegler–Nichols method"** is used to tune the PID controller:
  - Provides a systematic approach based on system response
  - Does not require a precise model, making it ideal for real-world balancing tasks

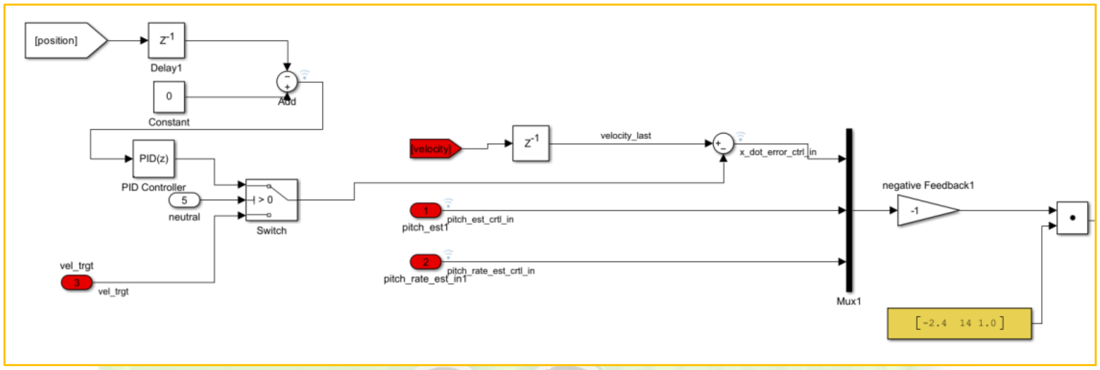
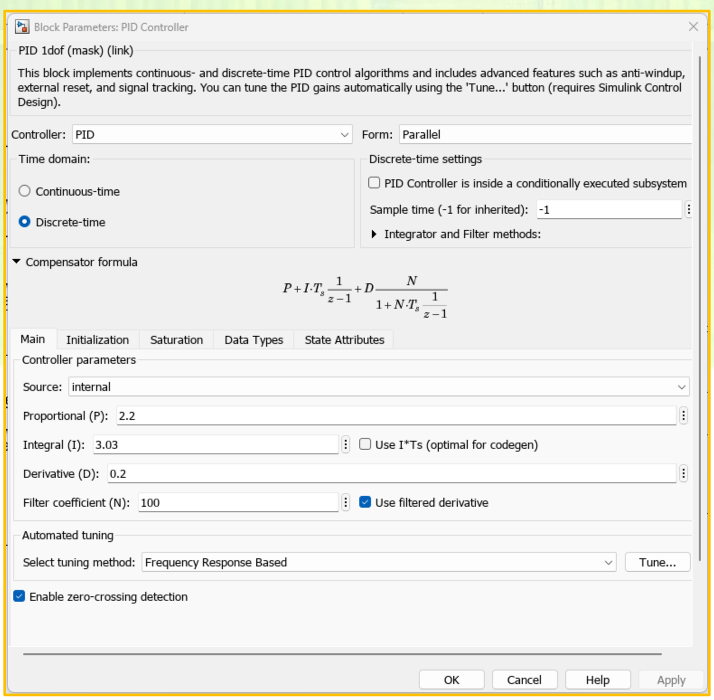


Fig: my control BLDC2 block diagram in state control block

PID Parameters

- Proportional (P): 2.2
- Integral (I): 3.03
- Derivative (D): 0.2
- Filter Coefficient (N): 100
- Tuning Method: Frequency Response Based



$$P + I \cdot T_s \frac{1}{z-1} + D \frac{N}{1 + N \cdot T_s \frac{1}{z-1}}$$

Fig: PID controller values

Fig: Weights

- The combined weight of all accessories is approximately 400 grams. During testing, the robot was able to maintain balance under this load. However, when the weight exceeded 400 grams, the robot became unstable and was unable to compensate effectively.

- **Watch the robot complete Challenge 3 here**:

  https://drive.google.com/file/d/1enrcaShX-5f-Rv6xtj_MOBAMjCTFglsM/view?usp=sharing

Challenge 4: Driving a figure 8 by combining two circles with 1m diameter each:

The aim of this challenge is to program the two-wheel robot to drive in a figure 8 trajectory, formed by two connected circles of 1 meter diameter each. This task tests the robot's ability to perform continuous curved motion, maintain stability, and return to its starting point.

- A Stateflow chart named **"OP_mode_stop_and_go_4".** is created within the autonomous target block.
- The chart defines five states to manage motion:
  - **Stop state** for initialization and stabilization
  - **Turn states** for semicircular paths (anti-clockwise and clockwise)
  - **Forward states** to connect the two circles smoothly
  - **Final stop state** for stabilization at the end

- The following formulas are used to calculate angular velocity and linear speed:
  - $\omega = \theta/t$
  - $v = \omega . r$

  Where; ω = Angular velocity (rad/sec)
  θ = angular displacement (rad) ($\theta=2\pi$)
  t = Time required to complete one circle (sec)
  v = Linear speed (m/s)

  These parameters ensure smooth circular motion and accurate path tracking.

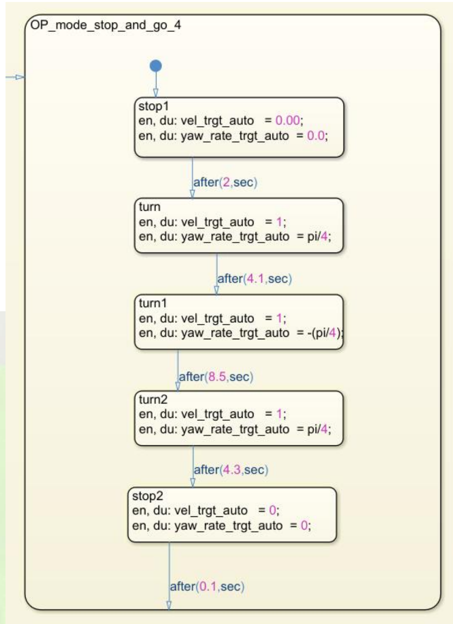


Fig: Autonomous targets

Controller Adjustment:

- Integrator saturation limits are set to +1 and −1.
- Control parameters are tuned to **[-1.5, 12, 1.0]** for stable circular motion.
- Mean angle1 is defined as **1*whl.rad** to synchronize wheel rotation with the circular trajectory.

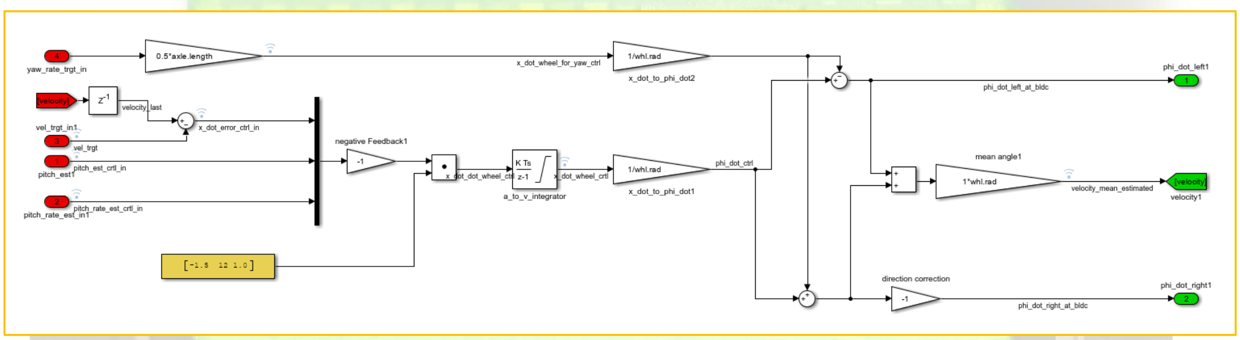


Fig: my control BLDC2 block diagram in state control block

- The robot begins at the start point, executes a semicircle in the anti-clockwise direction, and reaches the center of the figure 8.
- It then transitions into a **clockwise semicircle** to complete the second loop.
- Finally, the robot returns to the **starting position**, stabilizes, and stops.

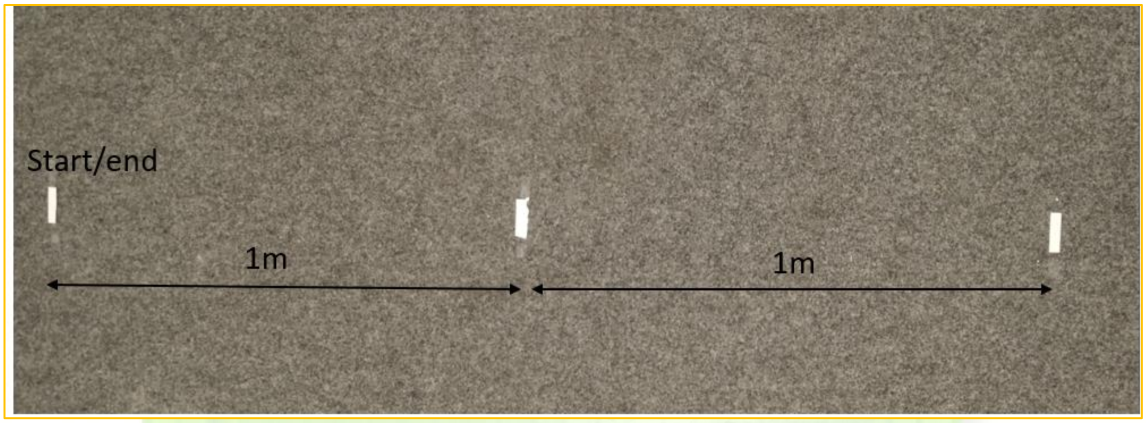


Fig: Robot running path for making figure 8

Performance:

- The robot successfully completed the figure 8 trajectory in approximately 19 seconds.
- Graphs of velocity, yaw control, and pitch estimation confirm stable execution throughout the task.
- This challenge demonstrates the robot’s ability to perform complex closed-loop trajectories, combining precise control, stability, and smooth transitions between curved paths.

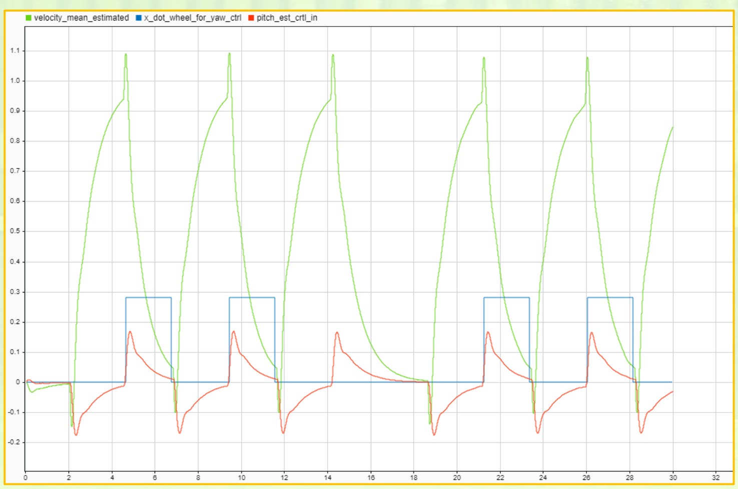


Fig: Graph for various parameters

- **Watch the robot complete Challenge 4 here**: https://drive.google.com/file/d/1wT2oj48leuTUbESjGRwrRz_fl2aziLON/view?usp=sharing