HW1: First-Order Boustrophedon Navigator

Andy Tsai
ctsai67@asu.edu
1233653410

February 3, 2026

1 Problem Statement

The objective of this assignment is to tune the PD controller gains and the pattern parameter (`spacing`) to achieve a precise and uniform boustrophedon pattern with minimal tracking error and smooth desired motion in ROS2 `turtlesim`.

Baseline (Default Settings)

All experiments start from the default controller and pattern parameters defined in the provided code. This configuration serves as the baseline reference for all tuning experiments.

```
class BoustrophedonController(Node):
    def __init__(self):
        super().__init__('lawnmower_controller')

        # Declare parameters with default values
        self.declare_parameters(
            namespace='',
            parameters=[
                ('Kp_linear', 1.0),
                ('Kd_linear', 0.1),
                ('Kp_angular', 1.0),
                ('Kd_angular', 0.1),
                ('spacing', 0.5)
            ]
        )
```

Baseline Results

For the baseline configuration, the final average cross-track error is **0.996**, and the maximum cross-track error reaches **2.111** (Figure 1). These relatively large errors mainly occur during turning maneuvers and line transitions (Figure 3), indicating that the default parameters do not provide accurate tracking for a uniform boustrophedon survey.

```
[boustrophedon_controller-2] [INFO] [1770173470.240712552] [lawnmower_controller]: Final average cross-track error: 0.996
[boustrophedon_controller-2] [INFO] [1770173470.241339898] [lawnmower_controller]: Maximum cross-track error: 2.111
```

Figure 1: Output for the baseline configuration, reporting the final average and maximum cross-track error.

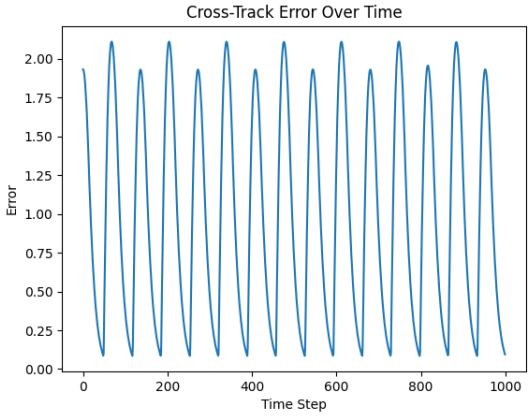

Figure 2: Cross-track error over time using the default parameter setting.

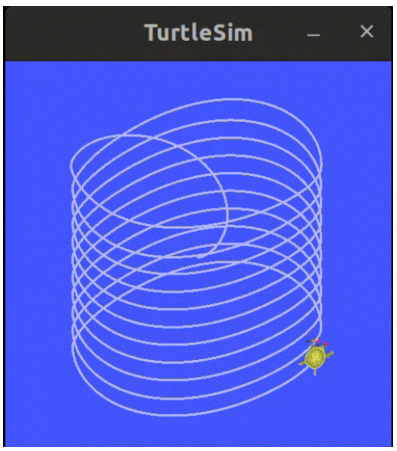

Figure 3: TurtleSim's trajectory under the default parameter setting.

2 Methodology

In this assignment, I used a one-parameter-at-a-time tuning strategy, where only a single parameter was modified while all other parameters were kept at their default values. This design makes it easier to attribute changes in the trajectory pattern and cross-track error (CTE) to a specific parameter, and to understand how each parameter contributes to the final behavior.

Parameter sensitivity tests

Starting from the default settings, I first tested the four PD controller parameters: `Kp_linear`, `Kd_linear`, `Kp_angular`, and `Kd_angular`. After the controller achieved stable and desired tracking behavior, I then adjusted the remaining pattern parameter, `spacing`, to improve coverage uniformity and survey efficiency. This ordering avoids confounding effects, since `spacing` changes the geometry of the desired path while the PD gains determine how well the turtle tracks that path.

To highlight the effect of each parameter, I evaluated scaled versions of the default values in separate trials (e.g., 10×). For example, `10xKp_linear` indicates that `Kp_linear` was increased to 10× its default value, while `Kd_linear`, `Kp_angular`, `Kd_angular`, and `spacing` remained unchanged. The same approach was applied to the other gains.

Final parameter selection

Final tuned parameters (`optimal_params`):

$$Kp\_linear = 1.2, \quad Kd\_linear = 0.02, \quad Kp\_angular = 9.0, \quad Kd\_angular = 0.03, \quad spacing = 0,5.$$

This configuration is used for the performance evaluation in the next section.

3 Performance

The tuned controller achieves low average and maximum cross-track error while maintaining smooth motion and uniform coverage. These results demonstrate that the selected PD gains and spacing provide a good balance between tracking accuracy and motion stability.

Cross-track error

Figure 4 shows the cross-track error over time using the final tuned parameters. The final average cross-track error is **0.061**, and the maximum cross-track error is **0.180**, indicating accurate path tracking throughout the survey.

```
[boustrophedon_controller-2] [INFO] [1770169578.943878462] [lawnmower_controller]: Final average cross-track error: 0.061
[boustrophedon_controller-2] [INFO] [1770169578.944177940] [lawnmower_controller]: Maximum cross-track error: 0.180
```

Figure 4: Output for the final tuned parameters, reporting the final average and maximum cross-track error.

Trajectory tracking

The resulting trajectory using the final parameters is shown in Figure 6. Compared to the baseline result, the turtle follows a more uniform boustrophedon (lawnmower) pattern with consistent line spacing and clean corner transitions.

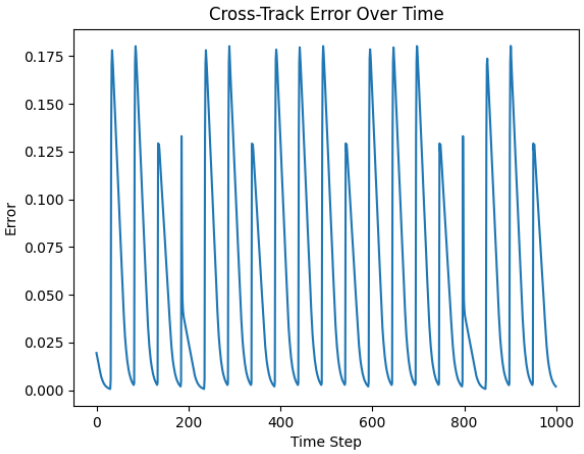

Figure 5: Cross-track error over time using the final tuned parameters.

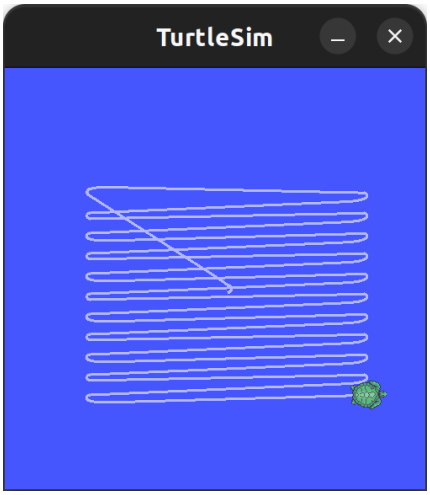

Figure 6: TurtleSim trajectory using the final tuned parameters.

Velocity profiles

Figure 7 shows the linear and angular velocity profiles during the survey. The linear velocity remains smooth and consistent during straight segments, while the angular velocity exhibits sharp but controlled changes during turns. This behavior suggests stable cornering without excessive oscillation.

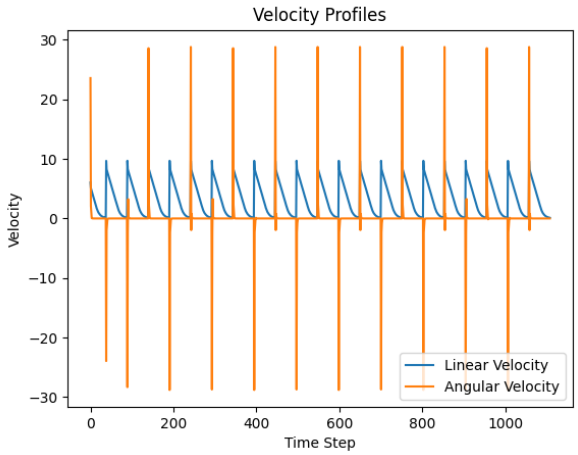

Figure 7: Linear and angular velocity profiles using the final tuned parameters.

4 Challenges encountered

Challenges

- **Excessive gains lead to oscillatory or unstable behavior.** When the angular derivative gain was increased to 10× the default value (`10xKd_angular`), the turtle exhibited strong oscillations during turning, resulting in an unstable trajectory. Similarly, increasing the angular proportional gain (`10xKp_angular`) produced sharper corners as expected, but also introduced noticeable oscillations near turns.
- **Overly aggressive linear gains distort path tracking.** With `10xKd_linear`, the turtle failed to follow the desired boustrophedon pattern and showed oscillatory behavior when approaching waypoints. In the `10xKp_linear` case, the turtle repeatedly circled around the first waypoint instead of progressing along the planned path. These results indicate that excessive linear gains can destabilize waypoint convergence and prevent proper path execution.

Solutions / tuning interpretation

From the challenges and my observations, I summarized the role of each gain as follows:

- `Kp_linear`: determines how strongly the turtle drives toward a waypoint (and whether it can reliably reach the waypoint).
- `Kd_linear`: affects the damping near a waypoint, i.e., how much the turtle oscillates when it gets close to the target.
- `Kp_angular`: controls how aggressively the turtle turns, which directly affects corner sharpness and heading alignment.
- `Kd_angular`: provides damping during turning, reducing oscillations and overshoot when cornering.

5 Comparison

High angular gains

Increasing angular gains primarily affects *turning behavior*. With higher `Kp_angular` (as shown in Figure. 8), the turtle produces sharper corners and faster heading correction, but oscillations appear near turns, as reflected by fluctuations in `/cmd_vel/angular/z`. When `Kd_angular` is increased excessively (as shown in Figure. 9), strong oscillatory behavior dominates the motion, resulting in unstable trajectories that deviate significantly from the desired boustrophedon pattern. These results indicate that while angular gains are essential for cornering performance, overly aggressive values reduce stability.

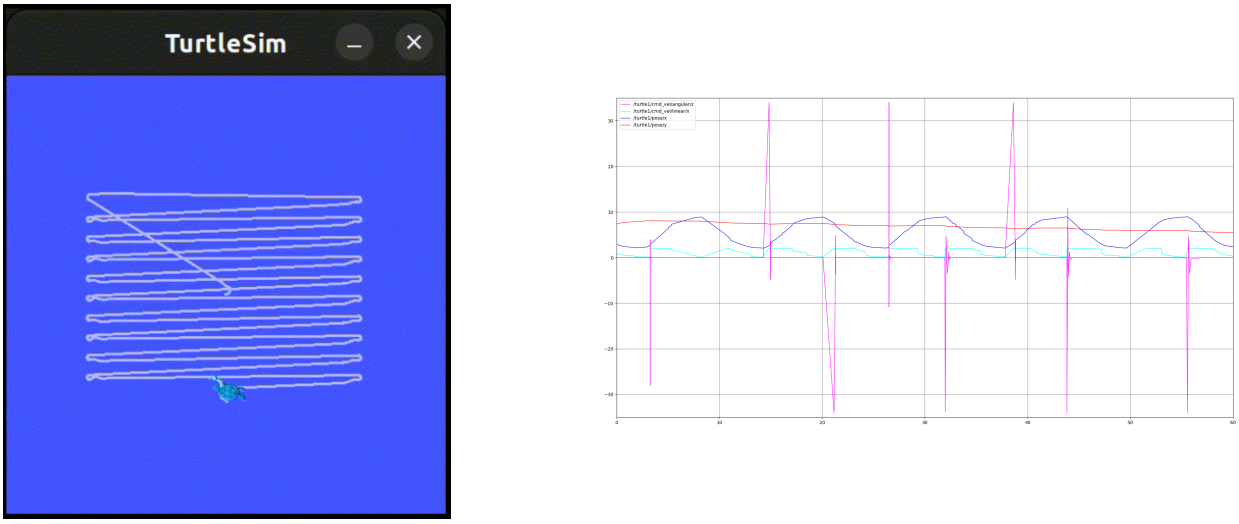

Figure 8: Trajectory (left figure) and rqt_plot (right figure) results for the `10xKp_angular` configuration.

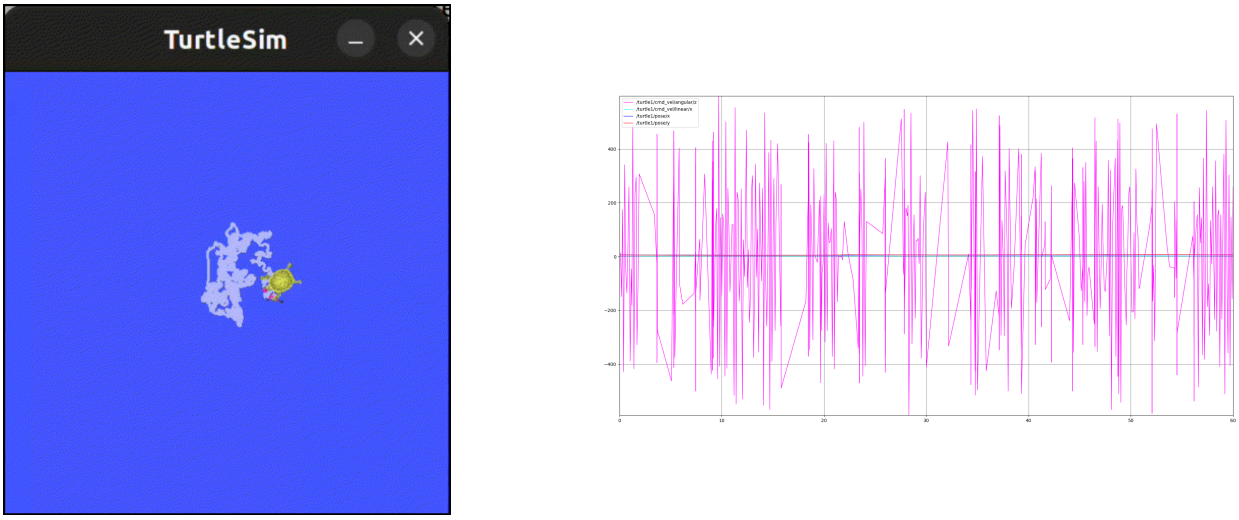

Figure 9: Trajectory (left figure) and rqt_plot (right figure) results for the `10xKd_angular` configuration.

High linear gains

Linear gains mainly influence *waypoint convergence and straight-line motion*. In the high `Kp_linear` case (as shown in Figure. 10), the turtle fails to progress along the planned path and instead circles around the first waypoint, indicating instability in waypoint attraction. With excessively large `Kd_linear` (as shown in Figure. 11), oscillations occur near waypoints, preventing smooth convergence and distorting the overall path. These behaviors show that aggressive linear gains can disrupt path execution even if angular control is adequate.

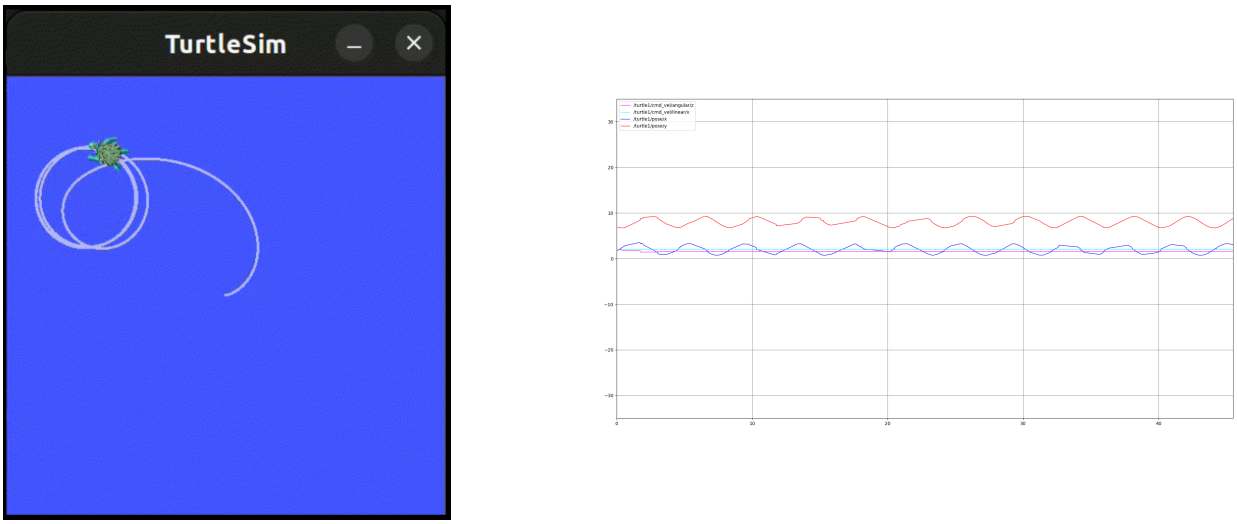

Figure 10: Trajectory (left figure) and rqt_plot (right figure) results for the `10xKp_linear` configuration.

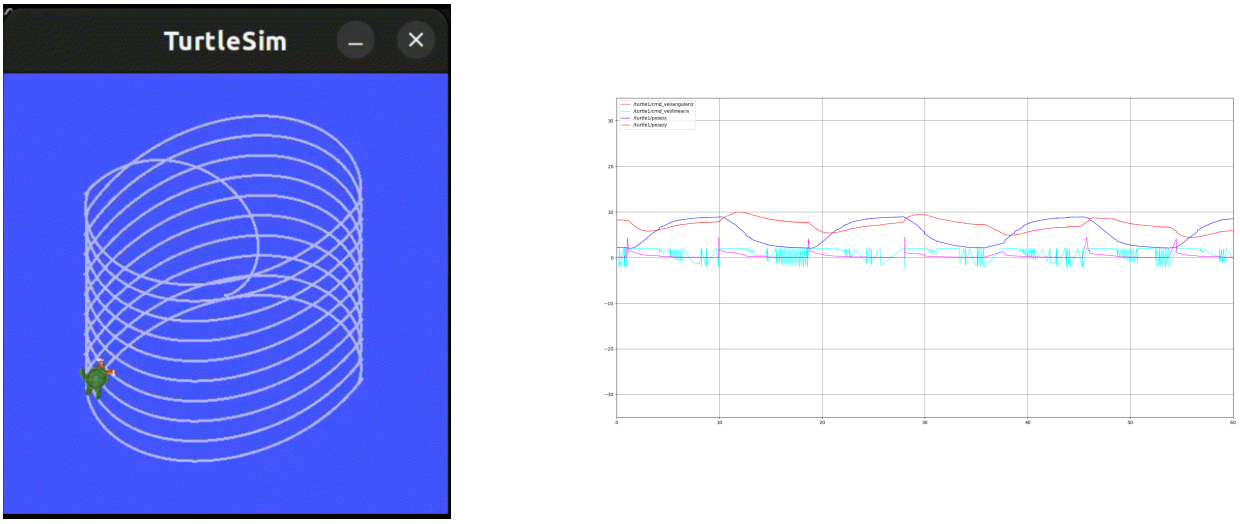

Figure 11: Trajectory (left figure) and rqt_plot (right figure) results for the `10xKd_linear` configuration.

Effect of spacing

Changing the `spacing` parameter directly modifies the geometry of the desired boustrophedon path. Increasing the spacing results in wider separation between survey lines and improved coverage efficiency. However, the pose and velocity signals remain similar to other configurations, confirming that spacing primarily affects path geometry rather than controller stability or tracking accuracy.

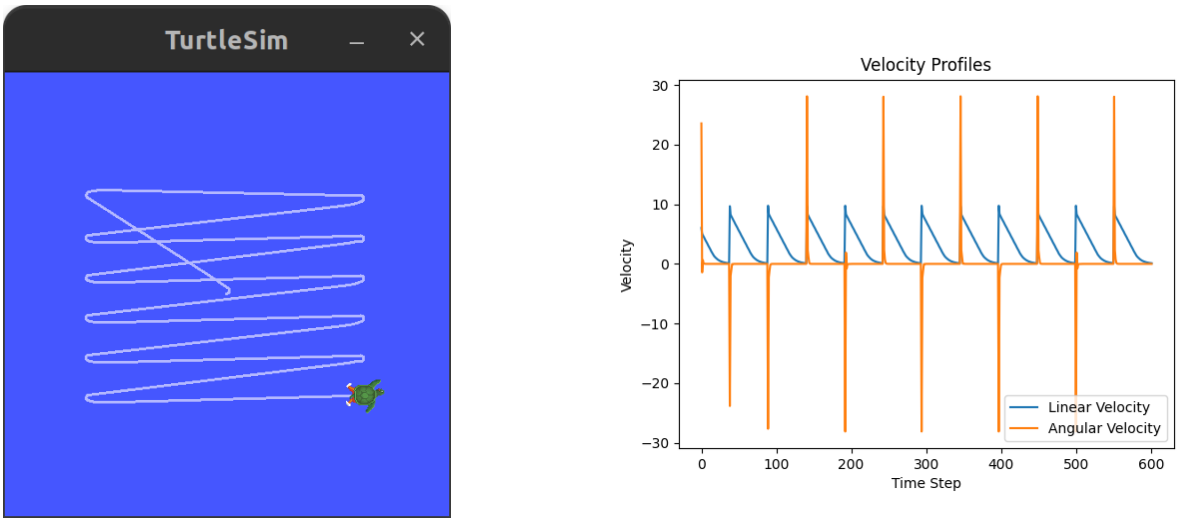

Figure 12: Trajectory (left figure) and velocity profile (right figure) results for the `2xSpacing` plus `optimal` configuration.

6 Github Repository

All results were saved in my GitHub repository and filed into `media/` folders named by the parameter setting (e.g., `default_params`, `10xKp_linear`, `10xKd_angular`, `2xSpacing`, etc.), enabling direct comparison across experiments.