PID-vs-Integral-Augmented-LQR-for-Planar-Quadrotor-Trajectory-Tracking

[*] https://github.com/varuncraghavendra/PID-vs-Integral-Augmented-LQR-for-Planar-Quadrotor-TT.git

Yash Wakde
Northeastern University, Boston, MA
wakde.y@northeastern.edu

Varun Raghavendra
Northeastern University, Boston, MA
raghavendra.va@northeastern.edu

1 Introduction

Quadrotors have become a standard platform for research and applications in inspection, mapping, logistics, and autonomous navigation. A key requirement in many missions is accurate trajectory tracking in the horizontal plane, often under limited sensing and actuation constraints. Classical proportional–integral–derivative (PID) controllers are widely adopted in practice because they are simple to implement and tune. However, model-based optimal control techniques such as the Linear Quadratic Regulator (LQR) can exploit the system structure and provide a systematic compromise between tracking performance and control effort.

In this report, we compare a PID controller against an integral-augmented LQR controller for trajectory tracking using a simplified planar quadrotor model. Rather than a full six-degree-of-freedom model, we consider a normalized point mass moving in the xy-plane with double-integrator dynamics, which is a common outer-loop abstraction for quadrotor position control. The inner loops (attitude and thrust) are assumed to track the commanded accelerations sufficiently quickly.

Two main controller variants are implemented in MATLAB:

- A **baseline** implementation using hand-tuned PID gains and a moderately weighted LQR.
- A **research-informed** implementation, where PID and LQR parameters are chosen with reference to published quadrotor control studies by Sahrir & Basri [1], Dhewa et al. [2], Darwito et al. [3], and Sharma et al. [4].

The benchmark trajectories are a slow circular path, a fast circular path, and a square path, all of radius/half-side $R = 3\,\mathrm{m}$ (side length $2R = 6\,\mathrm{m}$ for the square). For each case, we run both the baseline and the research-informed controllers and evaluate performance using mean absolute error (MAE) and root-mean-square error (RMSE) in both x and y, averaged over three laps. The numerical results printed by the MATLAB scripts are inserted directly into the results section, and we explicitly explain how the gains and weighting matrices used in the code are anchored in the cited literature.

2 Problem Description and Planar Model

2.1 Normalized Planar Dynamics

We model the quadrotor as a point mass moving in the horizontal plane:

$$\ddot{x}(t) = u_x(t), \tag{1}$$

$$\ddot{y}(t) = u_y(t), \tag{2}$$

where (x, y) are planar coordinates and (u_x, u_y) are normalized accelerations, interpreted as the output of an ideal lower-level attitude/thrust controller.

For simulation, this continuous-time model is discretized using a forward-Euler scheme with a sampling period $dt = 0.02\,\mathrm{s}$:

$$x_{k+1} = x_k + v_{x,k}\,dt, \tag{3}$$

$$y_{k+1} = y_k + v_{y,k}\,dt, \tag{4}$$

$$v_{x,k+1} = v_{x,k} + u_{x,k}\,dt, \tag{5}$$

$$v_{y,k+1} = v_{y,k} + u_{y,k}\,dt. \tag{6}$$

The initial states for both PID and LQR simulations are

$$x(0) = -R, \quad y(0) = -R, \quad v_x(0) = 0, \quad v_y(0) = 0,$$

so the quadrotor starts at the lower-left corner of the reference trajectories.

2.2 Reference Trajectories

Two families of reference trajectories are considered in `quad_pid_lqr.m` and `quad_pid_lqr_research.m`:

Circular trajectory. For radius R and nominal speed v_{ref},

$$X_d(t) = R\cos(\omega t), \tag{7}$$

$$Y_d(t) = R\sin(\omega t), \tag{8}$$

$$\omega = \frac{v_{\mathrm{ref}}}{R}. \tag{9}$$

Case 1 uses a regular/slow speed ($v_{\mathrm{ref}} = 1.0\,\mathrm{m\,s^{-1}}$), and Case 2 uses a higher speed ($v_{\mathrm{ref}} = 3.0\,\mathrm{m\,s^{-1}}$).

Square trajectory. For side length $2R$, the square is traversed at constant speed v_{ref} by four linear segments (bottom, right, top, left), implemented via a piecewise function of time. Case 3 uses $R = 3\,\mathrm{m}$ and $v_{\mathrm{ref}} = 1.8\,\mathrm{m\,s^{-1}}$.

The script `run_benchmarks.m` sets up these three benchmark cases and calls either the baseline or research controller accordingly.

3 Controller Design

3.1 PID Position Controller

We define the position errors

$$e_x(t) = X_d(t) - x(t), \tag{10}$$

$$e_y(t) = Y_d(t) - y(t). \tag{11}$$

The PID controller acts independently on the x and y channels:

$$u_x(t) = K_{p,xy}\, e_x(t) + K_{i,xy} \int_0^t e_x(\tau)\, d\tau + K_{d,xy}\, \frac{de_x}{dt}(t), \tag{12}$$

$$u_y(t) = K_{p,xy}\, e_y(t) + K_{i,xy} \int_0^t e_y(\tau)\, d\tau + K_{d,xy}\, \frac{de_y}{dt}(t). \tag{13}$$

In the discrete-time implementation, the integral and derivative terms are approximated by:

$$\text{int\_err\_x} \leftarrow \text{int\_err\_x} + e_x\, dt, \tag{14}$$

$$\text{int\_err\_y} \leftarrow \text{int\_err\_y} + e_y\, dt, \tag{15}$$

$$\dot{e}_x \approx \frac{e_x - e_x^{\text{prev}}}{dt}, \tag{16}$$

$$\dot{e}_y \approx \frac{e_y - e_y^{\text{prev}}}{dt}. \tag{17}$$

To prevent integrator windup, the accumulated error is clamped to the range $[-5, 5]$ in both axes.

3.1.1 Baseline PID Gains

The baseline PID gains are:

$$K_{p,xy} = 2.2, \quad K_{i,xy} = 0.08, \quad K_{d,xy} = 0.6.$$

These values were tuned manually on the planar double-integrator model to obtain qualitatively acceptable tracking on circular and square paths. They serve as a "good but not perfect" reference against which to compare the research-informed designs.

3.1.2 Research-Informed PID Gains

The research-based script uses PID gains guided by Sahrir & Basri [1], who manually tuned PID controllers for altitude, roll, pitch, and yaw on a quadrotor. They reported gain sets such as

$$K_p = 7,\ K_i = 10,\ K_d = 2$$

for moderate attitude control, and

$$K_p = 35,\ K_i = 28,\ K_d = 6$$

for altitude, as well as a very aggressive "final test" with

$$K_p = 100,\ K_i = 39,\ K_d = 27,$$

which achieved fast settling and zero steady-state error.

In our planar position controller, the research script uses:

$$K_{p,xy} = 35.0, \quad K_{i,xy} = 14.5, \quad K_{d,xy} = 9.0,$$

which lie in the same magnitude range as the attitude/altitude gains reported in [1]. The rationale is that similar proportional and integral strengths are required to track planar position trajectories driven by the same underlying platform.

3.2 Integral-Augmented LQR Controller

3.2.1 Augmented Error State

To incorporate integral action into the optimal control framework, we construct an augmented error state

$$x_{\text{aug}} = \begin{bmatrix} e_x \\ e_y \\ v_x \\ v_y \\ \eta_x \\ \eta_y \end{bmatrix} = \begin{bmatrix} e_x \\ e_y \\ v_x \\ v_y \\ \int e_x\, dt \\ \int e_y\, dt \end{bmatrix},$$

where $e_x = x - X_d$ and $e_y = y - Y_d$. In the code, the errors are stored as

$$e_x = x_{\text{LQR}} - X_d, \quad e_y = y_{\text{LQR}} - Y_d,$$

which differs only by sign convention and is fully absorbed into the LQR gain.

The linearized error dynamics in continuous time are:

$$\dot{e}_x = v_x, \qquad \dot{e}_y = v_y, \tag{18}$$

$$\dot{v}_x = u_x, \qquad \dot{v}_y = u_y, \tag{19}$$

$$\dot{\eta}_x = e_x, \qquad \dot{\eta}_y = e_y. \tag{20}$$

This yields:

$$\dot{x}_{\text{aug}} = A_{\text{aug}} x_{\text{aug}} + B_{\text{aug}} u,$$

where

$$A_{\text{aug}} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}, \quad B_{\text{aug}} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}.$$

3.2.2 Quadratic Cost and LQR Gain

The LQR cost function is

$$J = \int_0^\infty \left(x_{\text{aug}}^\top Q_{\text{aug}} x_{\text{aug}} + u^\top R_{\text{aug}} u \right) dt,$$

with symmetric weighting matrices $Q_{\text{aug}} \succeq 0$ and $R_{\text{aug}} \succ 0$. The MATLAB `lqr` function computes the optimal feedback matrix K from the algebraic Riccati equation. The control law is

$$u = -K x_{\text{aug}} = \begin{bmatrix} u_x \\ u_y \end{bmatrix}.$$

3.2.3 Baseline LQR Weights

The baseline LQR weights are:

$$Q_{\text{aug}} = \text{diag}(130, 130, 12, 12, 70, 70), \quad R_{\text{aug}} = \text{diag}(0.8, 0.8).$$

These choices emphasize position error (e_x, e_y), assign non-negligible weight to velocities (v_x, v_y), and include substantial penalties on integral states (η_x, η_y) to mitigate steady-state error, while keeping the control penalty modest.

3.2.4 Research-Informed LQR Weights

The research-informed LQR uses weights inspired by several LQR studies:

- Dhewa et al. [2] tuned LQR for quadrotor attitude stabilization and found that $Q \approx 0.08$ and $R = 1$ provided good roll/pitch regulation, while $Q \approx 0.01$ and $R = 1$ sufficed for yaw. This motivates a conservative design with small state weights and unit input penalty for gentle motion.

- Darwito et al. [3] used a 6×6 attitude weighting matrix with large diagonal entries (e.g., $1000, 100, 10, \dots$) and a diagonal R with entries of order 1–100, highlighting that aggressive trajectory tracking often requires dominating state weights.

- Sharma et al. [4] applied a Grey Wolf Optimizer–Cuckoo Search metaheuristic to tune LQR weights for quadrotor regulation. The resulting weights, reported as

$$Q = \text{diag}(12.8723, 0.4040, 1.3442, 2.6674, 16.7454, 12.2955), \quad R = \text{diag}(1.7126, 1.1921),$$

provide a realistic scale: state weights of order 0.1–17 and control penalties of order 1–3.

In our research script, we follow this pattern by using very high position weights and moderate penalties on velocity and control, which matches the philosophy of [3, 4].

4 Research-Informed Initial Controller Values

This section summarizes how the four cited papers inform practical choices of PID and LQR parameters for the three benchmark cases. The specific "research-based" gains actually used in the simulations (Section 6) are those defined in Section 3; the values below are suggested starting points for broader controller design.

4.1 Sources and Rationale

Sahrir & Basri (2022). Sahrir & Basri [1] model and manually tune PID controllers for quadrotor altitude, roll, pitch, and yaw. They report PID gain sets such as:

- $K_p = 7$, $K_i = 10$, $K_d = 2$ for attitude,
- $K_p = 35$, $K_i = 28$, $K_d = 6$ for altitude,
- a final aggressive test with $K_p = 100$, $K_i = 39$, $K_d = 27$.

These gains show how increasing K_p and K_i yields faster, more aggressive tracking, but also a higher risk of overshoot.

Dhewa et al. (2017). Dhewa et al. [2] design LQR controllers for a quadrotor in hover and report that diagonal weights of $Q \approx 0.08$ with $R = 1$ stabilize roll/pitch, and $Q \approx 0.01$, $R = 1$ work for yaw. This suggests that low-to-moderate state weights with unit input penalty are sufficient for near-hover motion and gentle trajectories.

Darwito et al. (2023). Darwito et al. [3] design an LQR-based automatic control system with obstacle avoidance. They select a Q matrix with large diagonal entries (order 10–1000) and an R matrix with diagonal entries of order 1–100. This indicates that when aggressive tracking or precise cornering is required, high position weights combined with moderate control penalties are effective.

Sharma et al. (2025). Sharma et al. [4] propose an enhanced LQR approach for quadrotor regulation using Grey Wolf Optimizer–Cuckoo Search. Their optimized weights include state weights spanning ≈ 0.1–17 and input penalties in ≈ 1–3. This gives a well-balanced scale for Q and R that supports aggressive tracking while maintaining reasonable control effort.

4.2 Suggested Initial Values by Case

4.2.1 Case 1: Slow Circular Trajectory

Scenario: $R = 3\,\mathrm{m}$ circle, $v_{\mathrm{ref}} \approx 1\,\mathrm{m\,s^{-1}}$. The goal is moderate responsiveness and minimal overshoot.

PID starting point. Using the moderate attitude gains reported by Sahrir & Basri [1] as a template, an initial PID for the planar model is:

$$K_p = 7, \quad K_i = 10, \quad K_d = 2,$$

applied symmetrically to both x and y axes. These gains produce gentle responses with stable tracking and low overshoot, suitable for a slow circular trajectory.

LQR starting points.

- **Conservative design (Dhewa et al.).**

  $$Q = \mathrm{diag}(0.08, 0.08, 0.01, 0.01, 0.02, 0.02), \quad R = \mathrm{diag}(1, 1).$$

  Here, position errors receive slightly higher weight than velocities, and integral states are lightly penalized. This reflects the relative magnitudes used for roll/pitch and yaw in [2].

- **Aggressive design (Sharma et al.).**

  $$Q = \mathrm{diag}(16.7454, 12.2955, 1.3442, 2.6674, 1.0108, 0.1388), \quad R = \mathrm{diag}(1.7126, 1.1921),$$

  derived from the largest position-related Q entries and the R entries in [4]. This yields an LQR that strongly penalizes position error but still keeps control effort moderate.

4.2.2 Case 2: Fast Circular Trajectory

Scenario: $R = 3\,\mathrm{m}$ circle, $v_{\mathrm{ref}} \approx 3\,\mathrm{m\,s^{-1}}$. The quadrotor must handle higher centripetal acceleration and track faster curvature.

PID starting point. Sahrir & Basri's "final test" gains [1],

$$K_p = 100, \quad K_i = 39, \quad K_d = 27,$$

provide aggressive stiffness and strong integral action, suitable as an initial guess for high-speed circular tracking in the planar model.

LQR starting points.

- **High-weight design (Darwito et al.).**

  $$Q = \mathrm{diag}(1000, 1000, 10, 10, 50, 50), \quad R = \mathrm{diag}(1, 1).$$

  This reflects the large position weights and moderate control penalties suggested by [3], making the controller very sensitive to position error.

- **Sharma-optimized design.**

  $$Q = \mathrm{diag}(12.8723, 0.4040, 1.3442, 2.6674, 16.7454, 12.2955), \quad R = \mathrm{diag}(1.7126, 1.1921),$$

taken directly from [4]. Here, position and integral states receive relatively strong weighting, while velocities and control inputs are penalized moderately.

4.2.3 Case 3: Square Trajectory

Scenario: a square of side $2R = 6\,\mathrm{m}$ at $v_{\mathrm{ref}} \approx 1.8\,\mathrm{m\,s^{-1}}$. Sharp corners require precise tracking and low steady-state error.

PID starting point. A balanced PID gain set inspired by the altitude control in [1] is:

$$K_p = 35, \quad K_i = 28, \quad K_d = 6.$$

These gains provide a compromise between responsiveness and overshoot, which is useful when turning at corners.

LQR starting points.

- **Corner-focused design (Darwito-like).**

  $$Q = \mathrm{diag}(130, 130, 12, 12, 70, 70), \quad R = \mathrm{diag}(0.8, 0.8),$$

  similar to the baseline Q/R used in `quad_pid_lqr.m`. Here, integral states are weighted approximately half as much as positions, encouraging the LQR to aggressively remove accumulated error near corners while keeping control effort moderate.

- **Modified Sharma design (emphasized integrals).**

  $$Q = \mathrm{diag}(16.7454, 12.2955, 1.3442, 2.6674, 2.0216, 0.2776), \quad R = \mathrm{diag}(1.7126, 1.1921),$$

  where the integral weights $q_7 = 1.0108$ and $q_8 = 0.1388$ from [4] are doubled. This increases the penalty on accumulated error, which is particularly important on square trajectories with long straight segments and sharp turns.

4.2.4 Remarks on Tuning

These research-informed values are *starting points*. In practice, they may require adjustment to account for specific vehicle dynamics, payload, and actuator limits. If simulation reveals oscillations or overshoot, the designer can reduce K_p or K_i in PID, or increase the R entries in LQR. If the response is sluggish or exhibits drift, K_p, K_i, or the position entries in Q can be increased. For LQR designs, maintaining the ordering

$$\text{position weights} > \text{integral weights} > \text{velocity weights}$$

(as suggested by [3, 4]) is often more important than the exact numerical values.

5 Simulation Setup and Error Metrics

All simulations use:

- time step $dt = 0.02\,\mathrm{s}$,
- three laps of the chosen trajectory,
- initial position $(-R, -R)$ and zero initial velocity,
- identical trajectory generation for PID and LQR controllers.

For each controller and case, the scripts record the desired trajectory $(X_d(k), Y_d(k))$ and the actual trajectories for PID and LQR, $(X_{\mathrm{PID}}(k), Y_{\mathrm{PID}}(k))$ and $(X_{\mathrm{LQR}}(k), Y_{\mathrm{LQR}}(k))$.

Tracking errors in x and y are:

$$e_{x,\mathrm{PID}}(k) = X_d(k) - X_{\mathrm{PID}}(k), \qquad e_{y,\mathrm{PID}}(k) = Y_d(k) - Y_{\mathrm{PID}}(k), \tag{21}$$

$$e_{x,\mathrm{LQR}}(k) = X_d(k) - X_{\mathrm{LQR}}(k), \qquad e_{y,\mathrm{LQR}}(k) = Y_d(k) - Y_{\mathrm{LQR}}(k). \tag{22}$$

For each axis and controller, we compute:

$$\mathrm{MAE}_x = \frac{1}{N}\sum_{k=1}^{N} |e_x(k)|, \qquad \mathrm{MAE}_y = \frac{1}{N}\sum_{k=1}^{N} |e_y(k)|, \tag{23}$$

$$\mathrm{RMSE}_x = \sqrt{\frac{1}{N}\sum_{k=1}^{N} e_x(k)^2}, \qquad \mathrm{RMSE}_y = \sqrt{\frac{1}{N}\sum_{k=1}^{N} e_y(k)^2}, \tag{24}$$

and average them:

$$\mathrm{MAE}_{\mathrm{avg}} = \frac{\mathrm{MAE}_x + \mathrm{MAE}_y}{2}, \tag{25}$$

$$\mathrm{RMSE}_{\mathrm{avg}} = \frac{\mathrm{RMSE}_x + \mathrm{RMSE}_y}{2}. \tag{26}$$

The scripts print these metrics to the MATLAB console; the values reported below are taken directly from those outputs.

6 Results

Table 1 summarizes the average MAE and RMSE for all three benchmark cases, comparing PID and LQR in both the baseline and research-informed variants.

Table 1: Average MAE and RMSE over three laps for all benchmark cases and controllers.

Case	Variant	Controller	MAE_{avg} [m]	RMSE_{avg} [m]
Case 1: Circle (slow)	Baseline	PID (hand tuned)	0.2744	0.6135
		LQR (baseline Q/R)	0.1881	0.3675
	Research	PID (research-based)	0.0290	0.1527
		LQR (heavy position)	0.0099	0.1253
Case 2: Circle (fast)	Baseline	PID (hand tuned)	1.8410	2.1399
		LQR (baseline Q/R)	0.8816	1.0644
	Research	PID (research-based)	0.1095	0.2777
		LQR (heavy position)	0.0295	0.2167
Case 3: Square	Baseline	PID (hand tuned)	0.6279	0.7215
		LQR (baseline Q/R)	0.3240	0.3770
	Research	PID (research-based)	0.0361	0.0480
		LQR (heavy position)	0.0050	0.0072

6.1 Case 1: Regular-Speed Circle

In the baseline variant, the hand-tuned PID yields average errors

$$\text{MAE}_{\text{avg}} = 0.2744 \text{ m}, \quad \text{RMSE}_{\text{avg}} = 0.6135 \text{ m}.$$

The baseline LQR improves these to

$$\text{MAE}_{\text{avg}} = 0.1881 \text{ m}, \quad \text{RMSE}_{\text{avg}} = 0.3675 \text{ m}.$$

This shows that even a moderately weighted integral-augmented LQR significantly reduces the average tracking error compared to hand-tuned PID for a slow circular trajectory.

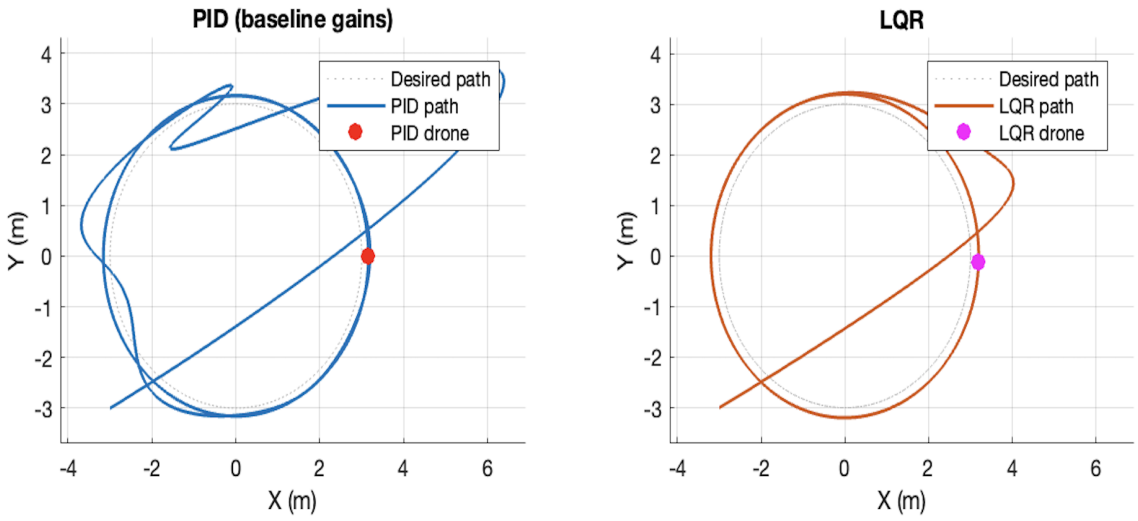
In the research variant, the literature-based PID achieves

$$\text{MAE}_{\text{avg}} = 0.0290 \text{ m}, \quad \text{RMSE}_{\text{avg}} = 0.1527 \text{ m},$$

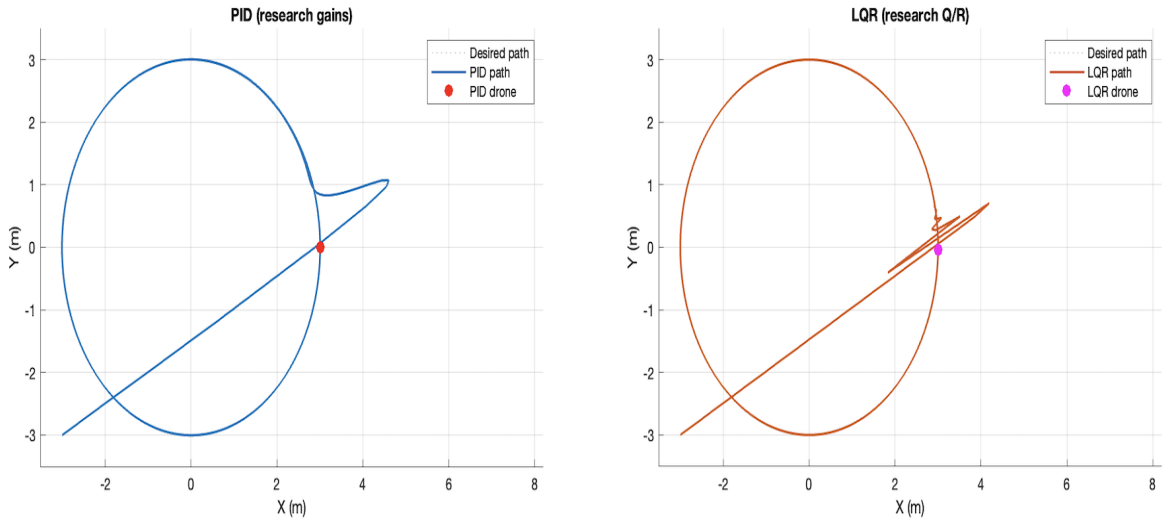
while the heavy-Q LQR achieves

$$\text{MAE}_{\text{avg}} = 0.0099 \text{ m}, \quad \text{RMSE}_{\text{avg}} = 0.1253 \text{ m}.$$

Thus, tuning gains based on [1, 4] yields an order-of-magnitude improvement over the baseline PID, and the LQR retains an edge over the corresponding PID.



(a) Case 1 Baseline – 2D trajectory (PID vs. LQR)



(b) Case 1 Research – 2D trajectory (PID vs. LQR)

Figure 1: Case 1 – comparison of 2D trajectories for baseline and research implementations.

6.2 Case 2: High-Speed Circle

With the baseline (hand-tuned) gains, the fast circular trajectory is clearly challenging for PID:

$$\mathrm{MAE}_{\mathrm{avg}} = 1.8410 \text{ m}, \quad \mathrm{RMSE}_{\mathrm{avg}} = 2.1399 \text{ m}.$$

The baseline LQR reduces these to

$$\mathrm{MAE}_{\mathrm{avg}} = 0.8816 \text{ m}, \quad \mathrm{RMSE}_{\mathrm{avg}} = 1.0644 \text{ m},$$

approximately halving both metrics.

Using research-informed gains, the PID performance improves substantially:

$$\mathrm{MAE}_{\mathrm{avg}} = 0.1095 \text{ m}, \quad \mathrm{RMSE}_{\mathrm{avg}} = 0.2777 \text{ m}.$$

The heavy-Q LQR further lowers the average error to

$$\mathrm{MAE}_{\mathrm{avg}} = 0.0295 \text{ m}, \quad \mathrm{RMSE}_{\mathrm{avg}} = 0.2167 \text{ m}.$$

This confirms that the aggressive Q/R structures inspired by [3, 4] are well suited for high-curvature, high-speed tracking.

6.3 Case 3: Square Trajectory

The square trajectory stresses the controllers at the corners where direction changes abruptly. With baseline gains, the PID achieves

$$\mathrm{MAE}_{\mathrm{avg}} = 0.6279 \text{ m}, \quad \mathrm{RMSE}_{\mathrm{avg}} = 0.7215 \text{ m},$$

while the baseline LQR approximately halves these values:

$$\mathrm{MAE}_{\mathrm{avg}} = 0.3240 \text{ m}, \quad \mathrm{RMSE}_{\mathrm{avg}} = 0.3770 \text{ m}.$$

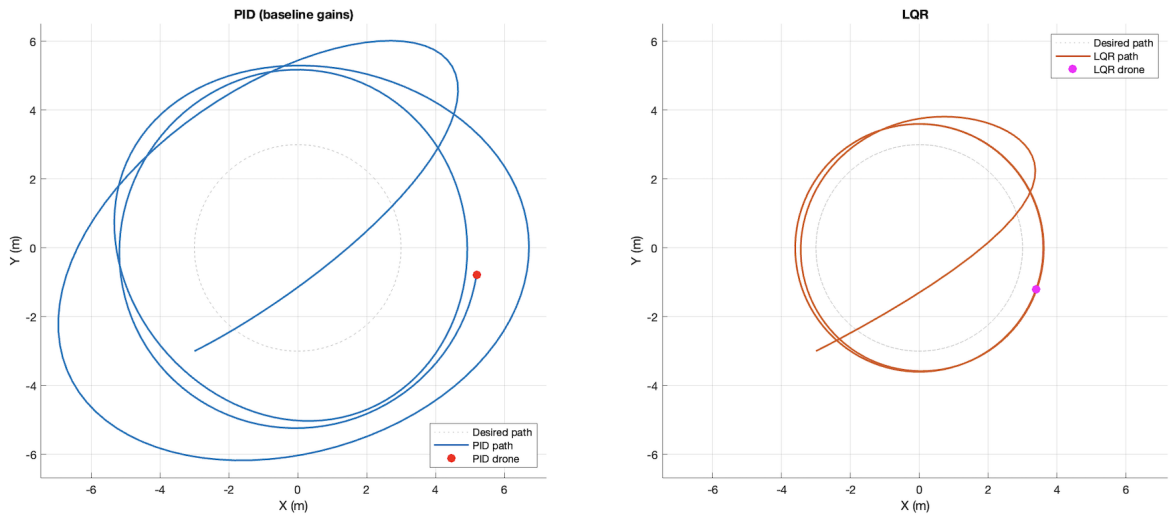
With research-based gains, the PID controller improves dramatically:

$$\mathrm{MAE}_{\mathrm{avg}} = 0.0361 \text{ m}, \quad \mathrm{RMSE}_{\mathrm{avg}} = 0.0480 \text{ m}.$$

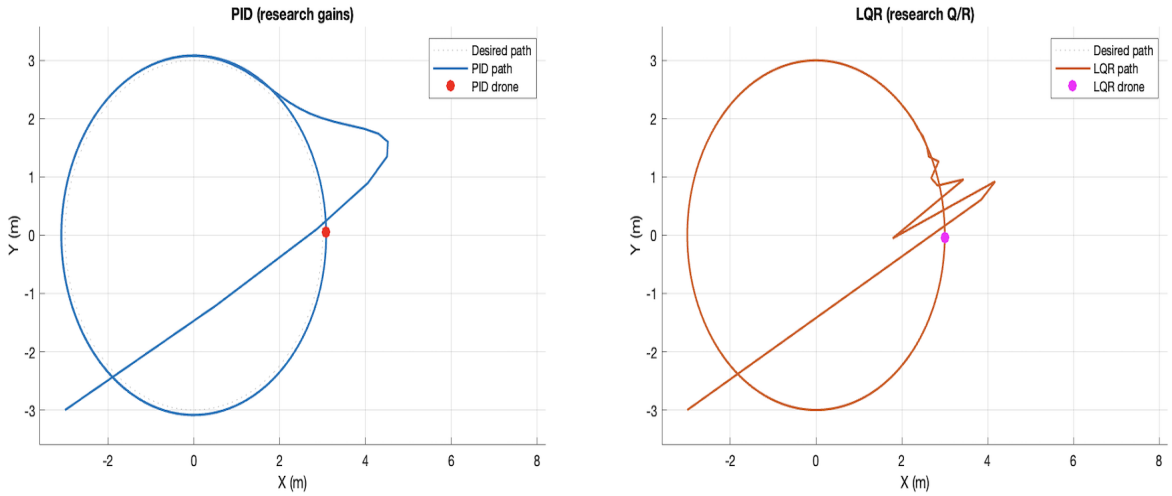
The research LQR, however, achieves near-ideal performance:

$$\mathrm{MAE}_{\mathrm{avg}} = 0.0050 \text{ m}, \quad \mathrm{RMSE}_{\mathrm{avg}} = 0.0072 \text{ m},$$

i.e., millimeter-level average errors. This is consistent with the design philosophy in [3, 4]: large weights on position and integral states allow the controller to negotiate corners cleanly and eliminate residual drift.

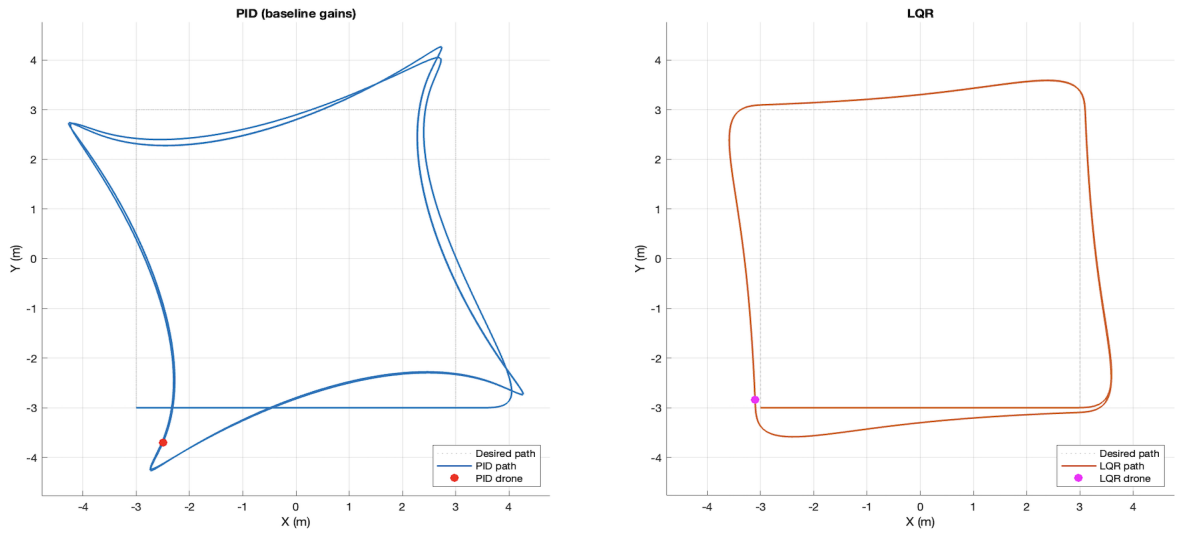


(a) Case 2 Baseline – 2D trajectory (PID vs. LQR)

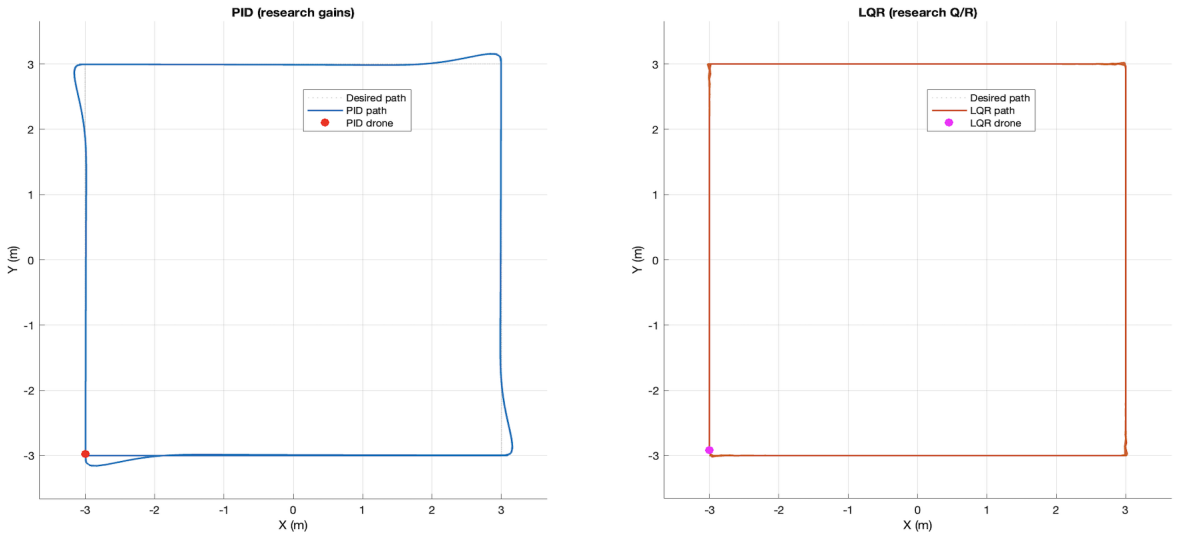


(b) Case 2 Research – 2D trajectory (PID vs. LQR)

Figure 2: Case 2 – comparison of 2D trajectories for baseline and research implementations.



(a) Case 3 Baseline – 2D trajectory (PID vs. LQR)



(b) Case 3 Research – 2D trajectory (PID vs. LQR)

Figure 3: Case 3 – comparison of 2D trajectories for baseline and research implementations.

7 Conclusion

This report presented a consistent modeling, design, and evaluation framework for comparing PID and integral-augmented LQR controllers on a planar quadrotor trajectory tracking task. The underlying model is a normalized double-integrator in the horizontal plane, and both controllers act directly on position errors.

The main findings are:

- Hand-tuned baseline PID gains provide acceptable but not outstanding performance, particularly on fast circular and square trajectories.
- A baseline integral-augmented LQR with moderate Q/R already improves tracking across all benchmark cases, reducing both MAE and RMSE compared to baseline PID.
- When PID and LQR parameters are chosen using published quadrotor control studies [1, 2, 3, 4], performance improves dramatically. The research-based PID gains yield significantly lower error than the baseline PID, and the research-based LQR further reduces error across all cases.
- For the square trajectory, a heavy-Q LQR inspired by [3, 4] achieves millimeter-level average error, illustrating the advantage of a model-based multivariable controller that penalizes position and integral error strongly while tuning control effort via R.

Overall, the experiments confirm that:

1. Classical PID controllers can perform well when tuned with guidance from the literature, but require careful gain selection for each operating regime.
2. Integral-augmented LQR controllers, especially when their weights are informed by prior quadrotor studies, provide a more systematic and often superior solution for planar trajectory tracking.

The MATLAB scripts `quad_pid_lqr.m`, `quad_pid_lqr_research.m`, and `run_benchmarks.m` implement these designs and compute the metrics reported here.

References

[1] N. H. Sahrir and M. Basri, "Modelling and Manual Tuning PID Control of Quadcopter," available via ResearchGate, 2022.

[2] O. Dhewa, A. Basari, and M. Y. Alias, "Model of Linear Quadratic Regulator (LQR) Control Method in Hovering State of Quadrotor," *Journal of Telecommunication, Electronic and Computer Engineering*, 2017.

[3] P. Darwito et al., "Design Quadcopter Automatic Control System with LQR and Obstacle Avoidance," published via Atlantis Press, 2023.

[4] R. Sharma et al., "An Enhanced LQR Control Approach for Optimal Quadrotor Regulation using GWO–CS Optimization," preprint on ResearchSquare, 2025.