INTERNATIONAL JOURNAL OF ADVANCED RESEARCH
International Peer-Reviewed Research Journal
ISSN (Online): 3106-7980 | ISSN (Print): 3106-7972
Volume 3 • Issue 7 • November 2026

A LIGHTWEIGHT 6-DOF FLIGHT DYNAMICS MODEL WITH STABILITY-AUGMENTED EULER INTEGRATION FOR REAL-TIME DESKTOP TRAINING SIMULATORS

Hafiz Muhammad Farhan[*1], Kashif Saghar[2], Mati Ur Rab[3], Eman Murtaza[4], Sania Hafeez[5]

[*1] *Abasyn University, Islamabad, Pakistan*
[2] *CESAT, Islamabad, Pakistan*
[3] *CESAT, Islamabad, Pakistan*
[4] *FUI, Islamabad, Pakistan*
[5] *National University of Modern Languages, Pakistan*

[*1]Hafizfarhan0039@gmail.com, [2]kashif.saghar.v@nu.edu.pk, [4]emonmurtaza@gmail.com, [5]saniahafeez200@gmail.com

DOI: https://doi.org/

Keywords

Article History
Received: *25 June 2026*
Accepted: *16 September 2026*
Published: *02 November 2026*

Copyright @Author
Corresponding Author: *
Hafiz Muhammad Farhan

Abstract
Desktop flight simulators need physics that is cheap to run, simple to integrate, and numerically well-behaved under frame-rate-driven time steps. Certified-grade 6-DOF tools meet the last requirement by relying on aerodynamic datasets that hobbyist and student projects rarely have. This paper presents a lightweight 6-DOF flight dynamics engine for the .NET 8/WPF Flight Simulator Suite that computes forces and moments from only 14 coefficients. Its central idea is a stability-augmented semi-implicit Euler integrator: an attitude-proportional restoring moment and angular-rate damping together form a proportional-derivative (PD) attitude augmentation that keeps rotational motion bounded at variable time steps. We analyse the augmentation in closed form (natural frequency ≈ 0.9 rad/s, damping ratio ≈ 0.11) and validate the real implementation with 20 automated tests and five experiments: closed-form comparison, step response, perturbation recovery, a 2,000-run randomised robustness study, and timing. Steady states match theory to two decimals, all 1,000 disturbed zero-control runs return to level flight, no run produces a non-finite state at time steps up to 0.099 s, and one physics step costs 0.18 µs. The analysis also exposes a weakness: the shipped gains are lightly damped (70% overshoot), so we recommend retuning. The result is a low-cost, headless-testable model that complements, rather than replaces, certified-grade tools.

Index Terms–*Flight dynamics, 6-DOF simulation, real-time systems, stability augmentation, Euler integration, desktop training simulators.*

I. INTRODUCTION

Desktop flight simulators are a low-cost supplement to full-motion training devices [1], [6], but their value depends on a dynamics engine that responds to the pilot the way learners expect [6]. Two requirements pull in opposite directions. The engine must be cheap enough to share a processor with rendering and networking, and it must stay numerically well-behaved when the time step follows the frame rate

and when a user or a test script pushes the aircraft towards an upset.

Open-source flight dynamics models such as JSBSim [12] and the textbook formulations behind them [11] describe aerodynamics with per-aircraft coefficient or derivative tables. They are accurate when such data exist, but low-cost projects usually lack that data [2], [3] and the verification infrastructure of certified-grade efforts [5]. Existing low-cost work targets model identification [2] and hardware [3]; how a deliberately simple model stays stable at frame-rate time steps is left open. This paper addresses that gap.

Problem and scope. We ask whether a 14-coefficient 6-DOF model, integrated with semi-implicit Euler, stays well-behaved under variable time steps and is cheap enough for real-time use on a desktop processor. Explicit Euler has no inherent damping and accumulates error [4], and the engine was observed to drift in attitude under turbulence until a restoring moment was added. We study the flight dynamics engine of the Flight Simulator Suite only; the other components appear in Section IV as context.

Contributions.

- **Stability-augmented Euler integrator.** A PD-structured attitude augmentation (restoring moment plus rate damping) on a semi-implicit Euler integrator, with a closed-form analysis of its dynamics and time-step limits.
- **Compact aerodynamic parameterisation.** Forces and moments from 14 coefficients (Table VII), with no wind-tunnel data.
- **Modular code base with a clear API.** A UI-independent core library with Update(dt), control and state properties, and GetSensorData(), so the engine runs headless in tests.
- **Comprehensive validation.** 20 automated tests and five experiments on the real implementation: closed-form comparison, step response, ablation, a 2,000-run robustness study, and timing.

II. RELATED WORK

A. Low-cost 6-DOF models

Burston et al. derive 6-DOF model parameters for a small unmanned aircraft by reverse engineering and 3-D scanning, and check simulated trajectories qualitatively against a published reference model [2]. Rožić et al. couple a low-cost force-sensing sidestick to a non-linear 6-DOF model and evaluate it in a human-in-the-loop pitch-tracking study [3]. Both use simplified parametric models because a full aerodynamic database is unavailable, as we do.

B. Desktop simulators in training

Reweti et al. report generally positive, though not universal, transfer of procedural and instrument skills from PC-based devices, as a complement to full-motion devices [1]. Fussell finds that engagement depends on how closely the simulator's behaviour matches learners' expectations of aircraft response [6]. This motivates a dynamics engine that is well-behaved rather than merely plausible-looking.

C. Verification of 6-DOF simulators

The NASA Engineering and Safety Center compared independently implemented 6-DOF tools on common check-cases, because open-literature verification data are scarce [5]. We adopt the idea at a small scale: simulation output is compared with closed-form solutions and with analytic limits.

D. Stability augmentation and integration

Classical theory separates static stability (the initial tendency to return to trim) from dynamic stability (whether the return is achieved and damped); aircraft with reduced static stability need artificial stability augmentation [7]. Our restoring moment is a simulation-side analogue of such augmentation. For integration, Euler schemes have a smaller stability region and larger error than fourth-order Runge-Kutta (RK4) but need one derivative evaluation per step instead of four [4].

E. Positioning against the state of the art

JSBSim [12] and textbook models [11] offer higher fidelity per aircraft but need that aircraft's data. Our model trades fidelity for a 14-coefficient description with an analysable stability bound (Table A).

Work	Focus	Model	Validation	Cost
Burston et al. [2]	Parameter identification for a UAV	Reverse-engineered parametric	Qualitative vs. reference model	Low
Rožić et al. [3]	Sidestick hardware	Non-linear 6-DOF	Human-in-the-loop tracking	Low
NASA NESC [5]	Cross-tool verification	Certified-grade tools	Check-case comparison	High
This paper	Numerical stability of a compact model	14 coefficients; semi-implicit Euler + PD augmentation	Closed form, ablation, 2,000-run robustness, timing	Low

TABLE A. Related low-cost and reference 6-DOF efforts compared with this paper.

III. NOTATION AND COORDINATE FRAMES

Two frames are used: a body frame at the centre of gravity with X forward, Y to the right and Z up, and a world frame whose Z axis is altitude. The implementation uses this Z-up convention rather than the north-east-down convention of many aerospace texts. Attitude is the Euler triple roll φ, pitch θ and yaw ψ, applied in yaw–pitch–roll order; p, q and r are the body rotation rates. Table I lists the symbols.

Symbol	Quantity	Frame / Units
x, y, z	Position	World frame, m
v	Velocity	World frame, m/s
φ, θ, ψ	Roll, pitch, yaw	Euler angles, rad
p, q, r	Roll, pitch, yaw rate	Body frame, rad/s
T, D, L, mg	Thrust, drag, lift, gravity	Force, N
ρ, P, T_{air}	Air density, pressure, temperature	ISA outputs
δ_a, δ_e, δ_r, δ_t	Aileron, elevator, rudder, throttle	Normalised input

TABLE I. Nomenclature of state and control variables.

IV. SYSTEM ARCHITECTURE

The Flight Simulator Suite is a Visual Studio solution of four projects built around one UI-independent core library (Fig. 1, Table II). The core holds all simulation logic and has no dependency on WPF, so the engine can run headless under test.

Project	Responsibility	Link to core
Core Engine	6-DOF dynamics, autopilot (PID), navigation, weather, logging, UDP	–
WPF UI (MVVM)	3-D view, OxyPlot graphs, Primary Flight Display	Reads state via view models
AI Grader	Heuristic scoring of UI layout, readability, colour, density	Screenshots only
UDP Sender Tool	Remote control and telemetry packets	UDP

INTERNATIONAL JOURNAL OF ADVANCED RESEARCH
International Peer-Reviewed Research Journal
ISSN (Online): 3106-7980 | ISSN (Print): 3106-7972
Volume 3 • Issue 7 • November 2026

TABLE II. The four projects of the Flight Simulator Suite.

Core API. The interface is small (Table III). A caller writes the controls and state on an Aircraft object, calls FlightDynamicsEngine.Update(dt) once per step, and reads the new state or GetSensorData(). The weather, autopilot, navigation, logging and UDP services share the same Aircraft object, so each can be tested on its own.

Member	Purpose
Aircraft.Throttle, Elevator, Aileron, Rudder	Control inputs
Aircraft.Position, Velocity, Attitude, AngularVelocity	State read by UI and logger
FlightDynamicsEngine.Update(dt)	Advance the state by dt seconds
FlightDynamicsEngine.GetSensorData()	Noisy instrument readings
HasCrashed, ResetCrashState()	Ground-impact status

TABLE III. Public API of the core library used by all other projects.

Presentation and input. The WPF layer follows the Model-View-ViewModel (MVVM) pattern [10]: view models expose bindable properties, the views contain no simulation logic, and the state is copied into those properties every tick. UDP packets arrive on background threads and are handed to the UI thread before they change the controls. This avoids WPF cross-thread errors and lets network and slider inputs drive the same state.

Separate tools. The AI Grader reads only UI screenshots, and the UDP Sender Tool talks to the core over UDP exactly as a remote machine would. Neither is evaluated further in this paper.

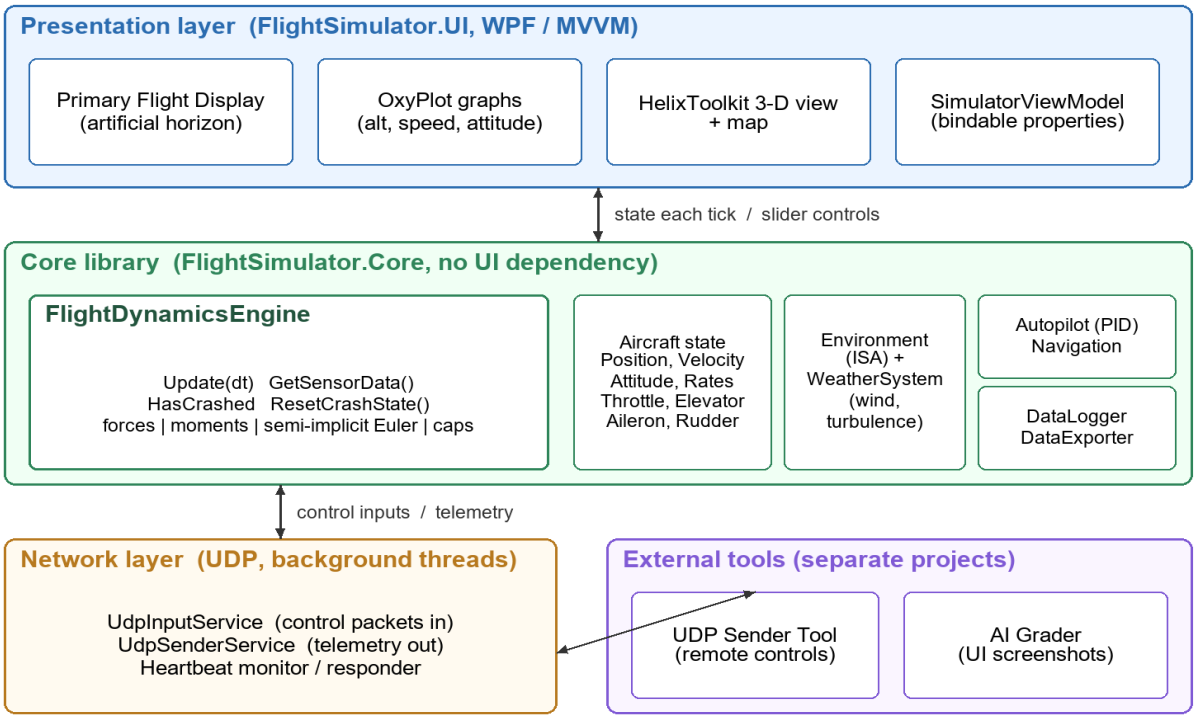


Fig. 1. Layered architecture. Arrows show data flow: state and controls between the core and the UI, and control packets and telemetry over UDP.

INTERNATIONAL JOURNAL OF ADVANCED RESEARCH
International Peer-Reviewed Research Journal
ISSN (Online): 3106-7980 | ISSN (Print): 3106-7972
Volume 3 • Issue 7 • November 2026

V. ATMOSPHERE MODEL

Drag and lift depend on air density, so the engine evaluates the International Standard Atmosphere (ISA) [9] once per step at the current altitude. The Environment class implements three layers: a troposphere to 11 km with a lapse rate of −6.5 K/km, an isothermal layer to 20 km, and a simplified exponential decay above. Pressure follows from the hydrostatic relation and density from $\rho = P/(R \cdot T)$. Within each layer ρ is smooth and falls monotonically with altitude, so the atmosphere adds no discontinuity to the force computation. The same pressure and temperature feed the sensor model, so forces and instrument readings share one atmosphere.

VI. FORCE MODEL

Four forces act on the aircraft. Thrust is a body-X force scaled by the throttle $\delta_t \in [0, 1]$:

$$T = \delta_t \cdot T_{max} \quad (1)$$

Drag opposes the airspeed vector relative to the wind, with air density ρ, airspeed v, drag coefficient C_d and reference area A:

$$D = \tfrac{1}{2} \rho v^2 C_d A \quad (2)$$

Lift acts along the body-up axis, so banking tilts it and turns lose height. Its coefficient is linear in pitch θ and elevator δ_e and is clamped to [0, 1.8], which prevents unbounded growth at extreme attitudes:

$$C_L = C_{L0} + k_{pitch}\,\theta + k_{elev}\,\delta_e \quad (3)$$

Lift follows $L = \tfrac{1}{2} \rho v^2 C_L A$. Gravity acts along world −Z. The forces are summed and divided by the mass to obtain the linear acceleration.

Example. With the values of Table VII at 1,000 m (ρ = 1.112 kg/m³) and v = 100 m/s, Eq. (2) gives $D \approx 4.45$ kN and the lift relation gives $L \approx 44.5$ kN. Level flight therefore needs a mass of about 4.5 t and a throttle of 0.22; the validation experiments below use 4,535 kg.

VII. MOMENT MODEL

Rotational motion is driven by three additive moments (Fig. 2): control-surface moments, rate damping, and the restoring moment described next. Control moments map aileron, elevator and rudder linearly to roll, pitch and yaw moments:

$$M_{ctrl} = (k_a\,\delta_a,\ k_e\,\delta_e,\ k_r\,\delta_r) \quad (4)$$

Rate damping opposes the current rotation rates, as aerodynamic damping does on a real airframe:

$$M_{damp} = (-c_p\,p,\ -c_q\,q,\ -c_r\,r) \quad (5)$$

Without damping, a constant aileron input would accelerate the roll rate without bound, because nothing else opposes an established rotation. Angular acceleration is the moment divided by the inertia of each axis.

A. Modelling Trade-offs: Euler Angles and Linear Moments

Attitude uses Euler angles rather than quaternions. Quaternions avoid gimbal lock near $\theta = \pm 90°$, but a training simulator rarely approaches vertical pitch, and Euler angles keep the logged state human-readable. Control moments are linear and independent per axis rather than a coupled stability-derivative model [11]. This is easier to tune and to keep stable, at the cost of cross-coupling effects such as adverse yaw. Both choices trade aerodynamic realism for simplicity and predictability.

VIII. STABILITY-RESTORING MOMENT

The restoring moment opposes roll and pitch deviations from level flight:

$$M_\varphi = -k_\varphi\,\varphi, \quad M_\theta = -k_\theta\,\theta \quad (6)$$

Together with rate damping, it forms a PD attitude law: the restoring term is proportional (P) and the damping term is derivative (D). For one axis with inertia I, restoring gain k, damping coefficient c, control gain g and control input δ, the attitude θ then obeys

$$I\,\ddot{\theta} + c\,\dot{\theta} + k\,\theta = g\,\delta \quad (7)$$

a damped second-order system with natural frequency and damping ratio

$$\omega_n = \sqrt{(k/I)}, \quad \zeta = c \,/\, (2\sqrt{(k\,I)}) \quad (8)$$

and a steady-state attitude $\theta_{ss} = (g/k)\,\delta$. Table IV evaluates these for the shipped gains. Yaw has no restoring moment, only damping.

Axis	k	c	I	ω_n	ζ	Period	θ_{ss}/δ
Roll	4000	1000	5000	0.894	0.112	7.07 s	1.25 rad
Pitch	6000	1500	8000	0.866	0.108	7.30 s	1.33 rad

TABLE IV. Second-order properties of the roll and pitch axes (k in N·m/rad, c in N·m·s/rad, I in kg·m²).

Why the term is needed.

Rate damping slows motion but cannot return an angle to level, so a disturbance would leave a permanent attitude error, as the ablation experiment below shows. The restoring term was added in commit 825a275 after the aircraft drifted in attitude under turbulence. The same commit scaled the turbulence rate impulse by Δt, which removed a second, time-step-dependent source of the drift.

Relation to real stability augmentation.

A certified augmentation system feeds back gyro measurements through a validated control law. The restoring moment here is a physical-model term tuned for plausible handling, not a control law, and carries no certification claim.

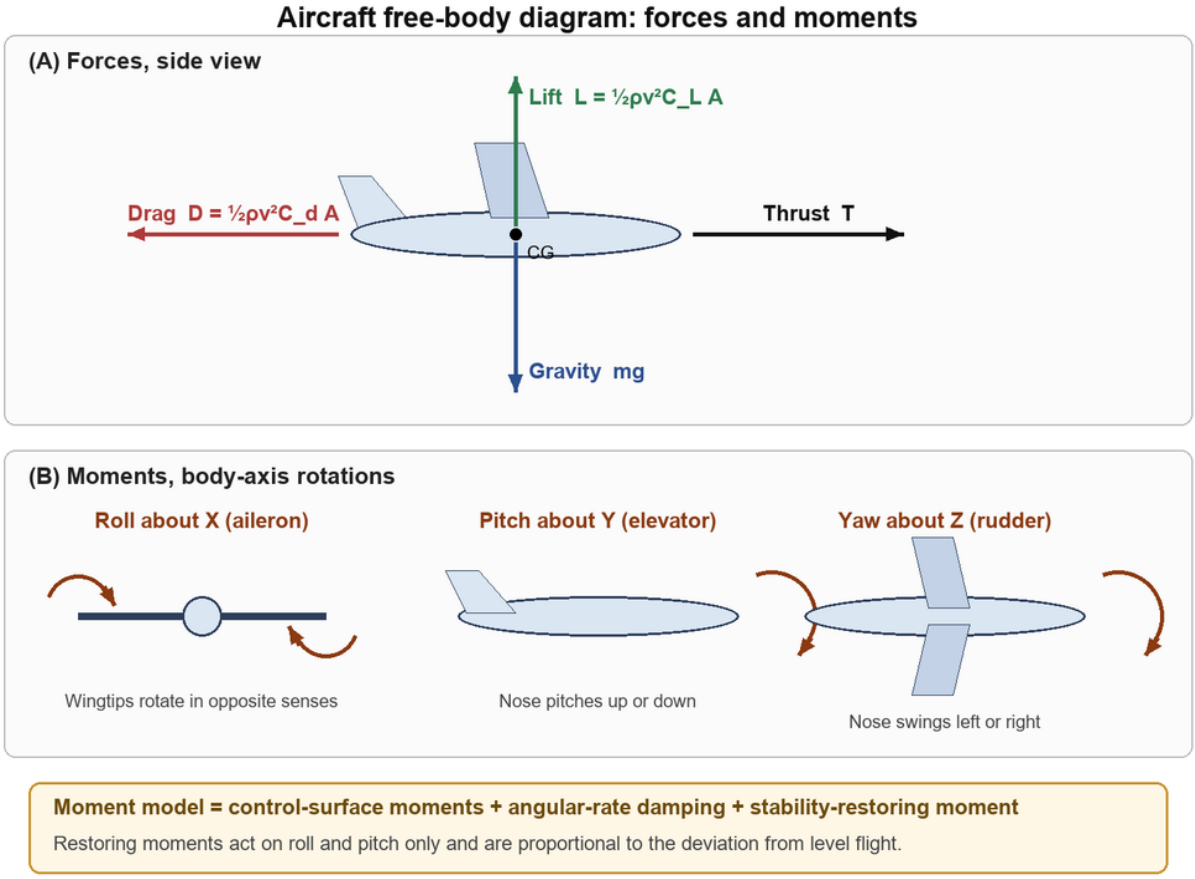


Fig. 2. Aircraft free-body diagram. (A) Forces resolved in the side view: thrust, drag, lift, and gravity. (B) Moments shown as three separate standard aerospace views, roll about the longitudinal axis (front view), pitch about the lateral axis (side view), and yaw about the vertical axis (top view), each driven by its corresponding control surface.

INTERNATIONAL JOURNAL OF ADVANCED RESEARCH
International Peer-Reviewed Research Journal
ISSN (Online): 3106-7980 | ISSN (Print): 3106-7972
Volume 3 • Issue 7 • November 2026

IX. NUMERICAL INTEGRATION AND SIMULATION LOOP

The engine uses semi-implicit (symplectic) Euler: the velocity is updated first and the new velocity updates the position, and likewise for angular rate and attitude:

$$v_{n+1} = v_n + a_n \Delta t, \quad x_{n+1} = x_n + v_{n+1} \Delta t \qquad (9)$$

This costs one force evaluation per step, like explicit Euler, but does not add energy to oscillatory motion. After each step, airspeed is capped at 280 m/s, angular rates at 2 rad/s per axis, pitch at ±90° and roll at ±180°. These caps and the lift clamp bound the state reachable in one step.

The application loop targets 60 Hz and measures Δt as the elapsed wall-clock time. If Δt reaches 0.1 s, for example after a debugger pause, the frame is skipped, so the integrator never sees a step of 0.1 s or more. Δt therefore varies from frame to frame instead of being fixed as in the accumulator pattern of [8], and stability must hold over a range of Δt. Fig. 3 shows the per-frame structure.

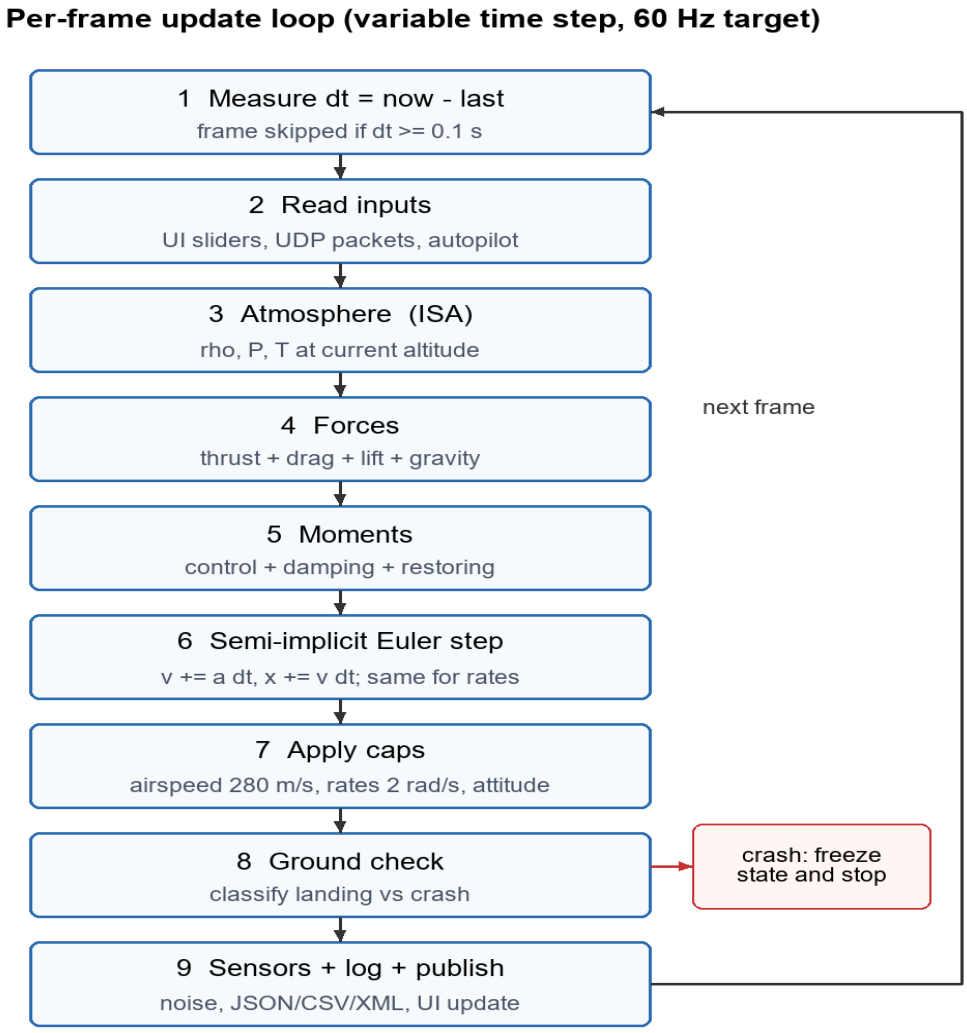


Fig. 3. Per-frame update loop of the implementation (SimulatorViewModel.cs, lines 658–727): variable time step, frames with Δt ≥ 0.1 s skipped.

Every stage is closed-form arithmetic on a fixed-size state, so the cost per step is independent of simulation length and of the inputs. The validation section reports the measured cost.

X. GROUND-IMPACT CLASSIFICATION

When the aircraft reaches the ground, a rule-based classifier compares sink rate and attitude with fixed limits:

$$\text{landing if } |v_z| \le 5\ m/s \ \wedge\ |\varphi| \le 60° \ \wedge\ |\theta| \le 60° \quad (10)$$

and crash otherwise. After a crash the state is frozen until ResetCrashState() is called. The rule is deterministic, so every flight ends in a well-defined outcome, and it reuses the state vector of Table I without a separate ground model.

XI. SENSOR SIMULATION AND DATA LOGGING

GetSensorData() adds zero-mean Gaussian noise to the true state: σ = 2 m for GPS position, 0.1 m/s² for acceleration, 0.01 rad/s for angular rate, 0.5 m for altitude and 0.2 m/s for airspeed, with smaller noise on magnetic field, temperature and pressure. This makes instruments look imperfect, and it gives downstream consumers such as the AI Grader a realistic display. The data logger records the state each step, and flights can be exported as JSON, CSV or XML. Because the ground truth is logged, step responses and recoveries can be analysed offline without extra instrumentation.

XII. IMPLEMENTATION

The suite is written in C# for .NET 8. OxyPlot draws the telemetry graphs, HelixToolkit the 3-D view, and MathNet.Numerics supplies the random distributions used by the sensor model. The AI Grader references ML.NET as an extension point for a future trained scoring model; no such model has been trained. Tests use xUnit, Moq and FluentAssertions. The test project contains 20 test cases covering the physics engine, the PID autopilot and navigation, and all 20 pass.

XIII. VALIDATION AND RESULTS

A. Method

All experiments call the real FlightDynamicsEngine from the core library. They ran in a Release build on .NET 8 on an Intel Core i5-1235U laptop. Unless stated, the aircraft has mass 4,535 kg and flies at 1,000 m and 100 m/s with throttle 0.22, Δt = 1/60 s, and altitude held fixed so that only attitude dynamics are exercised. For the closed-form comparison, a one-axis copy of the rotational equations was first checked against the engine: over 100 s of roll recovery the two differ by less than 10^{-5} degrees.

B. Closed-form agreement and integration accuracy

For an initial roll of 15° and no control, Eq. (7) has an exact solution. Table V gives the largest roll error over 30 s for the engine's semi-implicit scheme, explicit Euler and RK4. At 60 Hz the engine's error is 0.11° (0.7% of the amplitude). The error grows roughly linearly with Δt, reaching 0.67° at 0.1 s. Explicit Euler is neutrally stable at $\Delta t = 2\zeta/\omega_n \approx 0.25$ s and diverges beyond it, while the semi-implicit scheme stays bounded at every step tested up to 1 s. RK4 is far more accurate at every step.

Δt (s)	Semi-implicit (engine)	Explicit Euler	RK4
1/120	0.055	0.19	1×10^{-9}
1/60	0.11	0.38	2×10^{-8}
0.1	0.67	2.8	3×10^{-5}
0.25	1.7	14 (stability limit)	1×10^{-3}
0.5	3.5	diverges	0.017
1.0	8.3	diverges	0.28

TABLE V. Maximum roll-angle error (degrees) over 30 s against the closed-form solution, 15° initial roll.

C. Step response and perturbation recovery

A 0.1 elevator step settles at 7.64° and a 0.1 aileron step at 7.16°, both equal to the predicted θ_{ss} to two decimals. Both responses overshoot by about 70% (peak 13.1° and 12.2°) and reach 90% of the final value in 1.8 s. From an initial roll of 15° with zero control, the aircraft first crosses level after 1.9 s, undershoots by 70%, and settles within 10% of the initial error after 21.8 s; the pitch axis behaves alike (1.95 s, 71%, 22.7 s). A 45° disturbance gives the same normalised response. The final error is below 0.001° in every case. This confirms the predicted damping ratio of about 0.11: the augmentation guarantees recovery, but the response is lightly damped.

D. Ablation: damping without restoring moment

With the restoring gain set to zero in the one-axis copy, a 15° roll stays at 15.00° after 120 s, and a roll-rate kick of 0.5 rad/s leaves a permanent offset of 143°. With the restoring moment the same kick decays to 0.000°. Rate damping alone therefore cannot keep attitude near level.

E. Robustness

We ran 2,000 flights of 120 s, each with a random time step drawn from [1/240, 0.099] s at every step, a random wind of up to ±30 m/s per horizontal axis, random initial roll (±170°), pitch (±80°) and rates (±2 rad/s), and, in half of the runs, random control inputs. Across all runs no state became non-finite, and no airspeed, rate or attitude limit was exceeded. In the 1,000 zero-control runs, every aircraft returned to within 2° of level flight within 120 s.

F. Real-time feasibility

One Update() call, including the ISA evaluation, took 0.184 µs on average over 3 million steps, and GetSensorData() took 3.0 µs. At the 60 Hz target, the physics uses 0.001% of the 16.7 ms frame budget. A 100 Hz or 1 kHz update would use about 0.002% or 0.02% of one core. The frame rate of the application is therefore limited by rendering and loop pacing, not by the dynamics.

G. Ground impact

Touchdown tests at sink rates of 2, 4.9 and 5.2 m/s classify as landing, landing and crash. At 2 m/s, bank or pitch of 50° gives a landing and 65° gives a crash. All seven cases match Eq. (10).

Test	Result
Closed-form comparison	Steady states exact; 0.11° error at 60 Hz
Step response	≈70% overshoot; 90% rise 1.8 s
Perturbation recovery	Level in 1.9 s; settles in ≈22 s; final < 0.001°
Ablation (no restoring)	15° stays 15°; rate kick leaves 143°
Robustness (2,000 runs)	0 non-finite; 0 limit violations; 1,000/1,000 recovered
Timing	0.18 µs per step
Ground impact	7/7 cases as specified
Automated tests	20/20 pass

TABLE VI. Summary of validation results.

XIV. DISCUSSION

A. Strengths

- **Real-time feasibility.** At 0.18 µs per step the engine could run at refresh rates far above 60 Hz, such as 100 Hz or 1 kHz, with negligible load.
- **Numerical robustness.** No failures in 2,000 randomised flights with steps up to 0.099 s.
- **Analysability.** Eqs. (7) and (8) predict frequency, damping and steady state, and the simulation matches them.
- **Modularity.** The small API makes the engine testable without a UI.

B. Trade-offs and comparison with the state of the art

Semi-implicit Euler is first-order accurate: its error at 60 Hz is 0.11°, against negligible error for RK4. RK4

needs four force evaluations per step, so by our estimate it would cost under 1 µs per step, still negligible. We kept Euler for simplicity, and because the augmentation, not the integrator order, provides the stability. Table-driven models such as JSBSim [12] offer higher fidelity but need per-aircraft data. Our 14 coefficients can be set by hand: Eq. (8) turns a target frequency and damping ratio directly into gains.

C. Practical relevance

The low-cost solution is feasible: it needs no aerodynamic database, runs at a negligible fraction of a frame, and survives randomised abuse. It adds value for students, hobbyists and test automation, who get a deterministic, headless, instrumented model with closed-form tuning rules and a measured stability margin. It does not suit certified training credit, because it has not been compared with flight-test data.

D. Limitations and threats to validity

- **Damping.** The shipped gains give $\zeta \approx 0.11$, hence 70% overshoot and about 22 s to settle. Gains for $\zeta = 0.7$ would be $c \approx 6{,}300$ N·m·s/rad (roll) and 9,700 N·m·s/rad (pitch) by Eq. (8); we have not tested them.
- **Fidelity.** No comparison with flight-test data or an independent reference implementation has been made. Axes are decoupled, there is no yaw restoring moment, and Euler angles have a gimbal-lock limit.
- **Scope of tests.** The rotational tests hold altitude fixed, use constant wind rather than the weather system's turbulence, and were timed on one machine for the physics step only; UI frame pacing was not measured.
- **Provenance.** The account of the original drift comes from the repository history, not from a reproduced failure.

XV. CONCLUSION AND FUTURE WORK

A 14-coefficient 6-DOF model with a PD attitude augmentation on semi-implicit Euler integration is numerically well-behaved at variable time steps and costs 0.18 µs per step. Closed-form analysis and tests on the real implementation agree, and they also show that the shipped gains are too lightly damped for good handling.

Future work: (i) retune the damping to $\zeta \approx 0.7$ and repeat the experiments; (ii) compare the engine with an otherwise identical RK4 variant, to separate the benefit of the restoring moment from that of integrator order, and with a reference implementation or flight data in the spirit of [5]; (iii) measure frame times in the full application; (iv) test with the weather system's turbulence; and (v) evaluate quaternion attitude.

XVI. APPENDIX: PARAMETERS

Table VII lists the constants of the force, moment and ground-impact models as implemented. Mass is set per aircraft preset.

Parameter	Value	Role
A	20 m²	Reference area (Eq. 2)
C_d	0.04	Drag coefficient (Eq. 2)
C_{L0}, k_{pitch}, k_{elev}	0.4, 3.0 rad^{-1}, 0.15	Lift model (Eq. 3)
C_L limits	[0, 1.8]	Lift clamp
T_{max}	20 kN	Maximum thrust (Eq. 1)
k_a, k_e, k_r	5000, 8000, 3000 N·m	Control moment gains (Eq. 4)
c_p, c_q, c_r	1000, 1500, 2000 N·m·s/rad	Rate damping (Eq. 5)
k_φ, k_θ	4000, 6000 N·m/rad	Restoring gains (Eq. 6)
I_x, I_y, I_z	5000, 8000, 10000 kg·m²	Inertia
v_{max}, ω_{max}	280 m/s, 2 rad/s	Caps
$v_{z,max}$, φ_{max}, θ_{max}	5 m/s, 60°, 60°	Impact limits (Eq. 10)
Δt	measured, < 0.1 s	Integration step

TABLE VII. Model parameters (14 force and moment coefficients plus inertia, caps and limits).

XVII. REFERENCES

[1] S. Reweti, A. Gilbey, and L. Jeffrey, "Efficacy of Low-Cost PC-Based Aviation Training Devices," Journal of Information Technology Education: Research, vol. 16, pp. 127-142, 2017.

[2] M. T. Burston, R. Sabatini, R. Clothier, A. Gardi, and S. Ramasamy, "Reverse Engineering of a Fixed Wing Unmanned Aircraft 6-DoF Model for Navigation and Guidance Applications," Applied Mechanics and Materials, vol. 629, pp. 164-169, 2014.

[3] P. Rožić, M. Vrdoljak, K. Krajček Nikolić, and J. Ivošević, "Low-Cost Strain-Gauge Force-Sensing Sidestick for 6-DoF Flight Simulation: Design and Human-in-the-Loop Evaluation," Sensors, vol. 25, no. 14, art. 4476, 2025.

[4] W. Yenesew, M. Habtamu, and B. Basaznew, "Numerical methods for solving second-order initial value problems of ordinary differential equations with Euler and Runge-Kutta fourth-order methods," Frontiers in Applied Mathematics and Statistics, vol. 10, art. 1360628, 2024.

[5] R. Shelton and M. Madden, "Standard Check-Cases for Six-Degree-of-Freedom Flight Vehicle Simulations," NASA Engineering and Safety Center (NESC) Academy technical presentation, Apr. 21, 2015.

[6] S. G. Fussell, "Exploring the adoption of desktop simulators in pilot training: An ethnographic approach," presented at the conference proceedings on flight simulation and training research, 2022.

[7] Embry-Riddle Aeronautical University, "Aircraft Stability & Control," in Introduction to Aerospace Flight Vehicles (open textbook), ch. on stability and control, eaglepubs.erau.edu, 2023.

[8] G. Fiedler, "Fix Your Timestep!," Gaffer On Games, [Online]. Available: https://www.gafferongames.com/post/fix_your_timestep/.

[9] International Civil Aviation Organization, "Manual of the ICAO Standard Atmosphere, extended to 80 kilometres (262 500 feet)," Doc 7488, 3rd ed., 1993.

[10] J. Smith, "WPF Apps With The Model-View-ViewModel Design Pattern," MSDN Magazine, vol. 24, no. 2, Feb. 2009.

[11] B. L. Stevens, F. L. Lewis, and E. N. Johnson, Aircraft Control and Simulation: Dynamics, Controls Design, and Autonomous Systems, 3rd ed. Hoboken, NJ, USA: Wiley, 2016.

[12] J. Berndt, "JSBSim: An Open Source Flight Dynamics Model in C++," in Proc. AIAA Modeling and Simulation Technologies Conf. and Exhibit, Providence, RI, USA, Aug. 2004, AIAA Paper 2004-4923.

International Journal of Advanced Research