ButterFlight

Maya Plug-in Design Document

By: Cecilia Chen and Yiding Tian

Based on:

A Practical Model for Realistic Butterfly Flight Simulation.

Chen, Q., Lu, T., Tong, Y., Luo, G., Jin, X., and Deng, Z., 2022.
ACM Transactions on Graphics (TOG).

CIS 6600 — Advanced Topics in Computer Graphics and Animation

Instructor: Dr. Stephen Lane

Project Summary

Realistic butterfly flight animation is a recurring need in film, games, and virtual environments, yet no dedicated authoring tool exists for it. Hand-keying the erratic, noisy trajectories characteristic of real butterfly flight is prohibitively tedious, and CFD-based simulation is far too slow for production use—leaving artists with crude approximations that break the illusion of life.

ButterFlight addresses this gap by providing a Maya plug-in that generates physically plausible butterfly flight animations through a force-based simulation model. The design goals are to faithfully reproduce the characteristic features of real butterfly flight—including wing-abdomen coupling, aerodynamic lift and drag, and inherent trajectory noise—while keeping the interface simple enough for a generalist animator to use without any background in aerodynamics or simulation.

The primary target audience is VFX artists and technical directors working in film, games, or real-time virtual environments who need to populate a scene with one or more butterflies without spending hours on hand animation. Users are expected to have basic Maya modeling and rigging knowledge but require no knowledge of the underlying physics.

ButterFlight exposes a MEL-scripted UI panel from which the user can load a rigged butterfly mesh, set simulation parameters (flapping frequency, amplitude, wind direction and strength, swarm size), optionally attach the butterfly to a path curve, and bake the resulting motion as keyframed skeletal animation. Supported output modes include single-butterfly free flight, path following, wind interaction, and swarm aggregation. The final animation can be exported via Alembic or FBX for use in any downstream pipeline.

The core simulation is implemented as a Maya C++ plug-in (`.mll`) built against the Maya 2026 API. It runs the three-module algorithm from Chen et al. 2022: parametric maneuvering functions with wing-abdomen interaction, a force model combining quasi-steady aerodynamic forces with a curl-noise vortex force, and a maneuvering controller that decouples body translation from posture adjustment. The UI and scene setup utilities are written in MEL/Python.

The development schedule targets an alpha version (single-butterfly simulation with basic aerodynamic forces and MEL UI) by March 25th, a beta version (full force model, path following, and swarm support) by April 15th, and a final polished version with tutorials and demo scenes due May 11th.

1. Authoring Tool Design

1.1 Significance of Problem or Production/Development Need

Butterflies appear in a wide range of production contexts—background ambience in nature documentaries, hero insect shots in animated features, environmental atmosphere in video games and virtual reality, and educational simulations. Despite this demand, no dedicated authoring tool for butterfly flight animation exists in any major DCC package. Artists currently resort to one of three unsatisfactory options: (1) hand-keying every frame of wing and body motion, which is extremely time-consuming given the high flapping frequency ($\approx$9–11 Hz) and the erratic, noisy trajectories that characterize real butterfly flight; (2) looping a pre-made cycle animation, which produces robotic, repetitive motion with no dynamic response to the environment; or (3) outsourcing to a CFD-based simulation, which requires specialist knowledge, days of computation time, and produces results that are difficult to art-direct.

The fundamental challenge is that butterfly flight is governed by closely coupled wing-body interaction: the abdomen oscillates in counterphase to the wings, the thorax pitches with each stroke, and an inherent vortex-driven noise keeps the trajectory unpredictable. Without capturing all three effects simultaneously it is impossible to produce convincing butterfly animation in any setting, whether for a single hero butterfly or a swarm of hundreds. A tool that automates this physics while remaining controllable and fast enough for interactive use in Maya would directly address this production gap.

1.2 Technology

ButterFlight is based on the 2022 ACM SIGGRAPH paper *"A Practical Model for Realistic Butterfly Flight Simulation"* by Qiang Chen, Tingsong Lu, Yang Tong, Guoliang Luo, Xiaogang Jin, and Zhigang Deng. The paper proposes the first force-based model specifically designed for real-time butterfly flight simulation in graphics and animation applications. The approach is organized into three inter-connected modules, as illustrated in Figure 1.

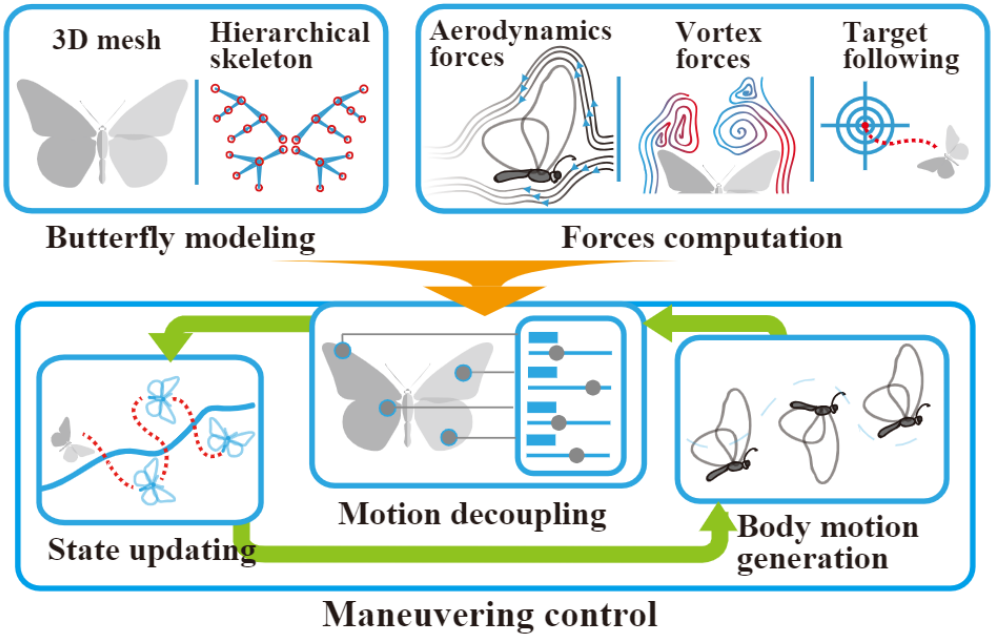

Figure 1: Pipeline overview of the Chen et al. 2022 approach. Starting from a rigged butterfly mesh, the system computes aerodynamic and vortex forces, then applies a maneuvering controller to generate body motion and trajectories (reproduced from Chen et al. 2022, Fig. 1).

The first module, **butterfly modeling**, represents the butterfly as a hierarchical skeleton (thorax as root, four wings, abdomen) driven by parametric maneuvering functions. Five joint angles—thorax pitch θ_β, fore-wing flap θ_γ, fore-wing feather θ_ζ, fore-wing sweep θ_ψ, and abdomen rotation θ_ϕ—are computed each frame via a cosine-based periodic function (Equation 1 of the paper) whose amplitude and frequency scale with the butterfly's current velocity. The abdomen is assigned a phase offset of $-180°$ relative to the wings to reproduce the counterphase oscillation observed in real Monarch butterflies [12]. Figure 2 illustrates the five joint angles and their geometric definitions.

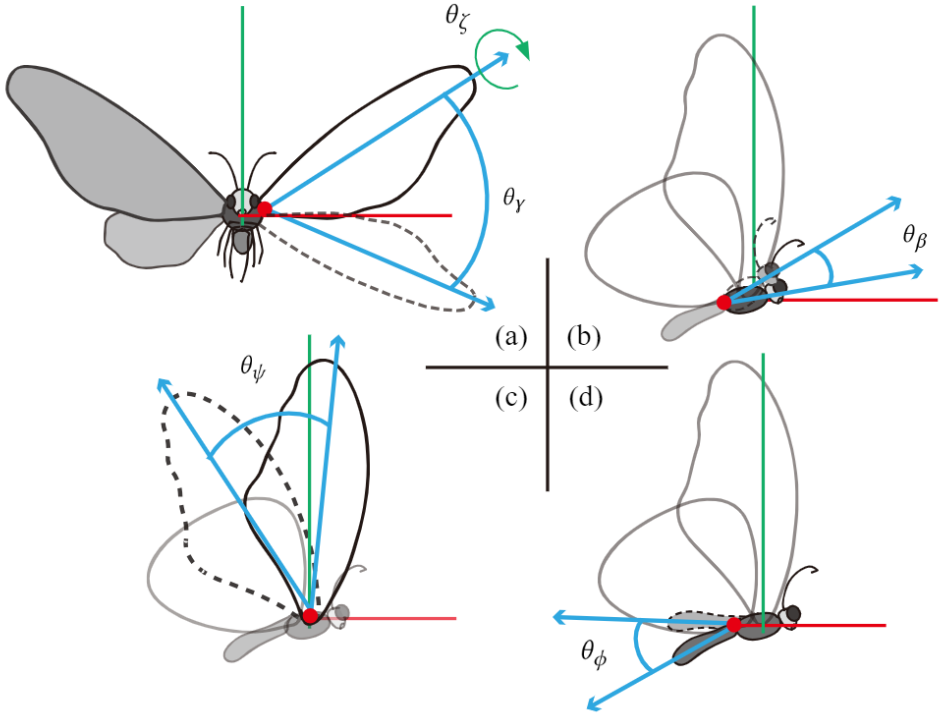

Figure 2: Joint angle definitions of the butterfly skeleton model. The thorax pitch angle θ_β and abdomen rotation θ_ϕ are shown in (d) and (b); the fore-wing flap angle θ_γ and feather angle θ_ζ are shown in (a); the sweep angle θ_ψ is shown in (c) (reproduced from Chen et al. 2022, Fig. 4).

The second module, **forces computation**, computes two forces acting on the butterfly at each time step. A simplified aerodynamic force (lift and drag per wing polygon, Equations 4–6) is derived from the quasi-steady theory of Ellington (1984), using empirical lift and drag coefficient polynomials as a function of the local angle of attack. A vortex force (Equation 7) is computed from a curl-noise field [8] seeded with Perlin noise [6] and applied to the thorax centre of mass, simulating the wake of the flapping wings and producing the inherent chaotic trajectory noise characteristic of real butterfly flight.

The third module, **maneuvering control**, decouples body translation from body posture. At each time step the composite force (aerodynamic + vortex + gravity + optional attraction toward a target or path keypoint) is integrated via Newton's second law to obtain velocity. Wing frequency and amplitude are then updated using a sliding-window smoother (Equation 12) to avoid abrupt changes across flapping cycles. Figure 3 shows the phase relationships of the five joint angles across one full flapping cycle.

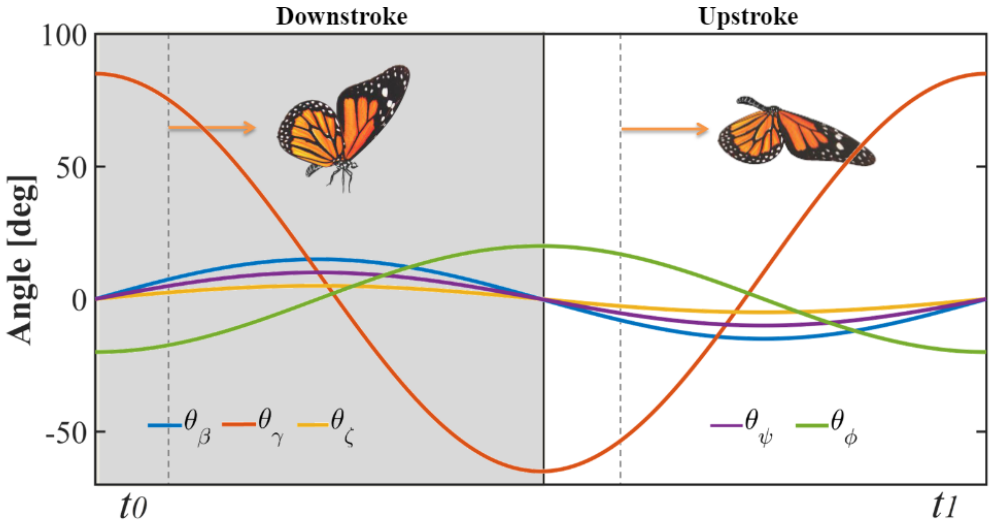

Figure 3: Phase shifts of all five maneuvering angles during one wing-flapping cycle (downstroke t_0 to upstroke t_1). The abdomen angle θ_ϕ is driven at $-180°$ phase offset relative to the flap angle θ_γ, reproducing counterphase oscillation (reproduced from Chen et al. 2022, Fig. 5).

We chose this paper because it is the only published method that addresses all three distinguishing features of butterfly flight (wing-abdomen coupling, aerodynamic force, and inherent trajectory noise) in a single, real-time capable framework. The algorithm is self-contained, mathematically explicit, and maps cleanly onto Maya's joint/keyframe animation system, making it an ideal candidate for implementation as an interactive authoring tool. The paper also includes quantitative parameter tables (Tables 2 and 3) for two butterfly species, providing concrete values to initialize our simulation.

1.3 Design Goals

ButterFlight addresses the production need by providing a Maya plug-in that automates the entire butterfly flight simulation pipeline—from joint-angle computation through trajectory integration to keyframe baking—behind a single, artist-friendly UI panel. The animator supplies a rigged butterfly mesh and a handful of high-level parameters; the plug-in handles all physics. The resulting output is standard Maya keyframe animation that can be refined by hand, cached to Alembic, or exported to any downstream pipeline without any dependency on the plug-in at render time.

1.3.1 Target Audience

The primary users of ButterFlight are **VFX artists and technical directors** in film, television, game development, and real-time virtual production who need to include one or more butterflies in a scene. This includes generalist animators who want a quick, convincing butterfly without manual keyframing, and TDs who need to populate a background environment with a swarm of butterflies in an automated, art-directable way. Secondary users are students and researchers who wish to explore physically-based insect animation.

Users are expected to have basic Maya literacy—they should know how to import meshes, work with joints and skinning, and use basic curve tools for path creation. No background in aerodynamics, physics simulation, or programming is required to operate the tool.

1.3.2 User Goals and Objectives

With ButterFlight, a user should be able to accomplish the following:

- **Single butterfly, free flight.** Simulate one butterfly flying with inherently noisy, physically plausible motion without specifying any path, suitable for ambient background use.
- **Single butterfly, path following.** Attach the simulation to a user-drawn NURBS curve so the butterfly broadly follows an art-directed trajectory while still exhibiting natural erratic deviations and wing-body dynamics.
- **Wind interaction.** Apply a constant or time-varying wind vector to the simulation and observe the butterfly attempt to recover stable flight, producing realistic spiraling responses.
- **Swarm simulation.** Instantiate multiple butterflies (up to $\sim$100 at interactive rates) that fly with collision-free, divergence-free curl-noise trajectories and individually-computed wing-body motion.
- **Parameter tuning.** Adjust species-specific physical parameters (wing area, body mass, flapping frequency range, vortex force gain) to produce different butterfly species or stylized variants.
- **Keyframe bake and export.** Bake the simulation result to Maya keyframes on the input rig, ready for render or export via Alembic or FBX.

The user should spend minimal time on setup—loading a rig, pressing *Simulate*, and baking should take no more than a few minutes for a single butterfly.

1.3.3 Tool Features and Functionality

ButterFlight exposes the following features:

- **Rig assignment.** The user selects a pre-rigged butterfly mesh from the Maya scene. The plug-in expects the rig to follow a standard joint-naming convention (`BF_thorax`, `BF_forewing_L/R`, `BF_hindwing_L/R`, `BF_abdomen`) so it can bind the correct simulation channels to the correct joints automatically.
- **Simulation mode selector.** Choose between *Free Flight*, *Path Following*, and *Swarm* modes from a drop-down menu.
- **Path curve input.** In Path Following mode, the user selects an existing NURBS curve from the scene as the attraction path.
- **Wind force control.** A direction vector and magnitude slider for a constant wind, plus a checkbox to enable time-varying (sinusoidal) wind.

- **Simulation parameter controls.** Numeric fields and sliders for: vortex force gain (η, $gain_{x,y,z}$), simulation duration (frames), playback frame rate, and sliding-window size k.
- **Swarm controls.** Number of agents, spatial spread at initialization, and inter-agent repulsion radius.
- **Simulate button.** Runs the C++ simulation for the specified duration and previews the result in the Maya viewport via a lightweight draw override.
- **Bake to Keyframes button.** Writes the simulation output as `setKeyframe` calls on all rig joints and the root transform, producing standard Maya animation curves.
- **Export.** After baking, the user can export to Alembic (`.abc`) or FBX through the standard Maya export dialogs; no special plug-in support is needed at this stage.

1.3.4 Tool Input and Output

Input:

- A Maya scene containing one or more rigged butterfly meshes, with joints named according to the ButterFlight convention.
- (Optional) A NURBS curve in the scene to serve as the flight path.
- (Optional) A wind vector specified through the UI.
- Simulation parameters set through the UI panel (simulation mode, duration, vortex force parameters, swarm count).

Output:

- Maya keyframe animation curves on all rig joints (rotation) and the root transform (translation and rotation), covering the simulated frame range.
- The output is entirely standard Maya animation data—no custom nodes are left in the scene after baking, and the animated rig can be rendered, referenced, exported, or hand-edited like any other Maya animation.

1.4 User Interface

ButterFlight's entire user-facing interface is a single floating panel window inside Maya, opened by sourcing the MEL script `butterFlight_ui.mel` and calling `butterFlightUI`. The panel is implemented as a scrollable `columnLayout` containing eight collapsible `frameLayout` sections—one per logical parameter group—followed by three color-coded action buttons at the bottom. No new menus are added to the main Maya menu bar; the tool is fully self-contained in the floating window. Figure 4 shows the panel as loaded in Maya 2026.

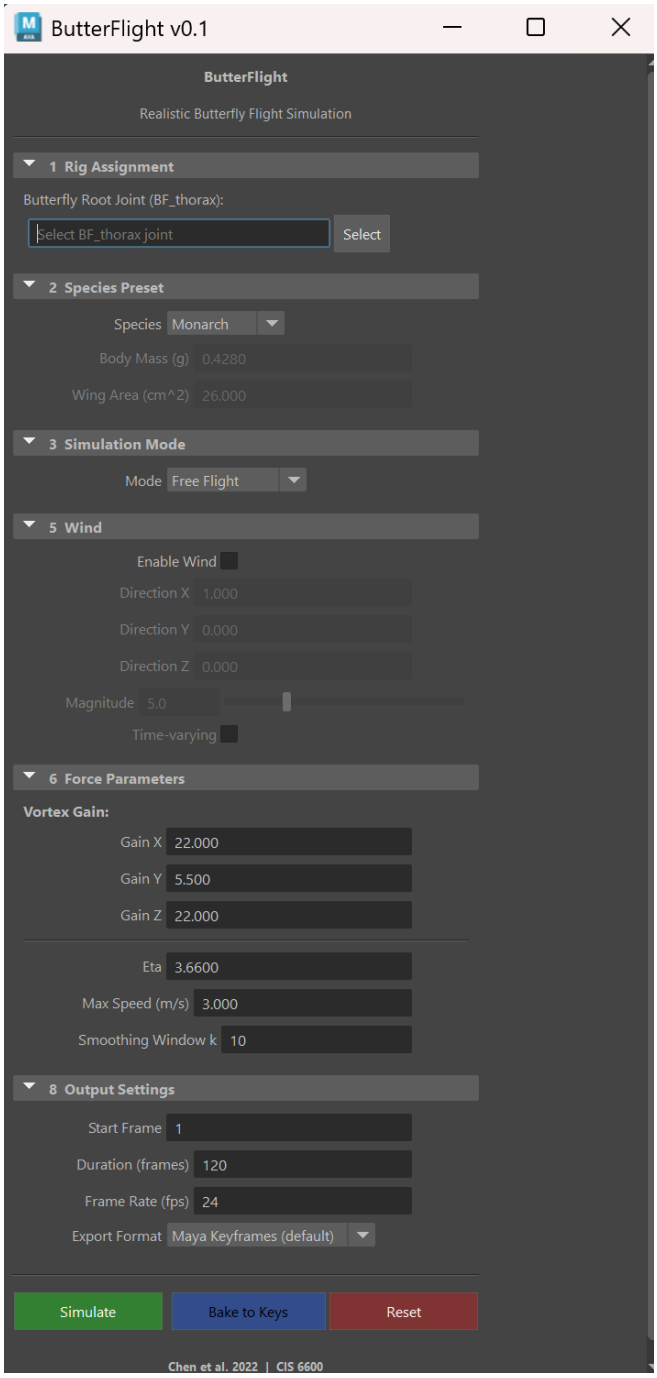

Figure 4: ButterFlight v0.1 UI panel running inside Maya 2026. Sections 4 (Path Settings) and 7 (Swarm Settings) are hidden by default and become visible when the corresponding simulation mode is selected.

1.4.1 GUI Components and Layout

The panel is divided into eight numbered, collapsible sections plus an action-button row. Table 1 summarises each section; a detailed description follows.

#	Section name	Key controls
1	Rig Assignment	Text field + *Select* button for `BF_thorax`
2	Species Preset	Drop-down (Monarch / Swallowtail / Custom); mass, wing area
3	Simulation Mode	Drop-down (Free Flight / Path Following / Swarm)
4	Path Settings	Curve picker + follow-strength slider (conditional)
5	Wind	Enable check; direction XYZ; magnitude slider; time-varying
6	Force Parameters	Vortex gain X/Y/Z; η; max speed; smoothing window k
7	Swarm Settings	Agent count; spawn spread; repulsion radius (conditional)
8	Output Settings	Start frame; duration; fps; export format
Simulate **Bake to Keys** **Reset**		

Table 1: ButterFlight GUI sections and their primary controls.

Section 1 — Rig Assignment. A `textFieldButtonGrp` displays the name of the currently assigned root joint. The *Select* button reads the active Maya selection, validates that the chosen node is a joint and that its name begins with the `BF_` prefix, and writes it into the field. A warning dialog is shown if the prefix check fails, allowing the user to override or cancel.

Section 2 — Species Preset. An `optionMenuGrp` offers three entries: *Monarch*, *Swallowtail*, and *Custom*. Selecting a named species auto-populates the body mass and wing-area fields with the values from Tables 2 and 3 of Chen et al. [1] (Monarch: 0.428 g, 26 cm^2; Swallowtail: 0.34 g, 28 cm^2) and disables those fields to prevent accidental edits. Selecting *Custom* re-enables both fields for free entry.

Section 3 — Simulation Mode. An `optionMenuGrp` controls the overall simulation behavior. Switching modes triggers the `bfUpdateMode` callback, which shows or hides Sections 4 and 7: Section 4 (Path Settings) is visible only in *Path Following* mode; Section 7 (Swarm Settings) is visible only in *Swarm* mode.

Section 4 — Path Settings (conditional). A second `textFieldButtonGrp` with a *Select* button lets the user pick an existing NURBS curve from the Maya scene to serve as the flight path. A `floatFieldGrp` sets the path-following strength (0–1).

Section 5 — Wind. A `checkBoxGrp` enables the wind subsystem. When checked, three `floatFieldGrp` controls for the wind direction vector (X, Y, Z) and a `floatSliderGrp` for magnitude become active. An additional checkbox enables time-varying (sinusoidal) wind. All five controls are disabled when wind is off, preventing irrelevant input.

Section 6 — Force Parameters. Six numeric controls expose the aerodynamic and vortex parameters that are otherwise fixed inside the C++ solver. Vortex gain in each axis ($gain_x = 22.0$, $gain_y = 5.5$, $gain_z = 22.0$) and the curl-noise turbulence scale $\eta = 3.66$ are pre-set to the Monarch defaults from the paper and are editable for fine-tuning. Maximum flight speed (m/s) and sliding-window size k (default 10 frames) complete this section.

Section 7 — Swarm Settings (conditional). Three `intFieldGrp` / `floatFieldGrp` controls set the number of simulated agents, their spatial spread at initialization (in metres), and the inter-agent repulsion radius used to keep butterflies from overlapping.

Section 8 — Output Settings. Integer fields for start frame and simulation duration (in frames), an integer field for frame rate, and an `optionMenuGrp` for export format (*Maya Keyframes*, *Alembic (.abc)*, *FBX (.fbx)*) define how the baked animation will be saved.

Action buttons. Three full-width buttons occupy the bottom row:

- **Simulate** (green): Collects all parameters, validates the rig field, and issues the C++ plug-in command to run the simulation for the specified duration.
- **Bake to Keys** (blue): Calls `bakeResults` on the entire rig hierarchy across the simulated frame range, writing standard Maya keyframe curves. If an Alembic or FBX export format is selected, the appropriate export command is also triggered.
- **Reset** (red): Restores all controls to their default values, clears the rig and path fields, and re-applies the Monarch species preset.

1.4.2 User Tasks

The following operations are available through the ButterFlight panel. Each corresponds to one or more GUI controls described in Section 1.4.1.

- **Assign Rig.** The user selects the `BF_thorax` root joint in the Maya viewport and clicks the *Select* button in Section 1. The plug-in validates the naming prefix and records the joint handle. This operation must be performed before any simulation can run.
- **Choose Species Preset.** The user picks *Monarch*, *Swallowtail*, or *Custom* from the Section 2 drop-down. Named species auto-fill body mass and wing area with paper-derived values and lock those fields; *Custom* leaves both fields editable.
- **Select Simulation Mode.** The Section 3 drop-down switches between *Free Flight*, *Path Following*, and *Swarm* modes. The selection controls which additional sections (4 or 7) are shown and which simulation code path is activated on *Simulate*.
- **Set Path Curve (Path Following mode only).** The user selects an existing NURBS curve in the viewport and clicks *Select* in Section 4 to designate it as the flight path. A follow-strength slider controls how tightly the butterfly tracks the curve.
- **Enable and Configure Wind.** Checking the *Enable Wind* box in Section 5 activates direction and magnitude controls. The user sets a world-space direction vector and a magnitude (m/s), and can optionally enable time-varying sinusoidal wind to simulate gusts.
- **Tune Force Parameters.** Section 6 exposes the six low-level numerical parameters of the simulation. Expert users or TDs may adjust vortex gains ($gain_{x,y,z}$), the curl-noise scale η, maximum flight speed, and the sliding-window size k to achieve different flight characters.
- **Configure Swarm (Swarm mode only).** Section 7 lets the user specify the number of agents, their spatial spread radius at spawn, and the inter-agent repulsion radius to prevent overlap.
- **Set Output Settings.** Section 8 defines the simulation time range (start frame, duration in frames, and frame rate) and the desired export format (Maya keyframes, Alembic, or FBX).

- **Simulate.** Clicking the green *Simulate* button collects all parameters, validates the rig assignment, and calls the C++ plug-in command to run the physics simulation. The resulting joint trajectories are previewed in the Maya viewport but not yet committed as keyframes.
- **Bake to Keyframes.** Clicking *Bake to Keys* converts the live simulation into standard Maya `animCurve` nodes on every rig joint across the specified frame range. If Alembic or FBX is selected, the appropriate export command is triggered immediately after baking.
- **Reset.** The red *Reset* button clears all input fields and restores every control to its factory default, including re-applying the Monarch species preset and hiding the conditional sections.

Prior knowledge required. Users must be comfortable with standard Maya operations: importing and referencing scene assets, selecting and naming joints, creating NURBS curves with the *EP Curve* or *CV Curve* tools, and exporting scenes via Alembic or FBX. No knowledge of aerodynamics, numerical integration, or MEL/Python programming is required to operate the tool; the force parameters in Section 6 carry sensible species-tuned defaults and need not be touched for typical use.

1.4.3 Work Flow

Figure 5 shows the complete ButterFlight workflow as a flowchart. The user sets up the scene and parameters, runs the simulation, then bakes and optionally exports the final animation. The steps below describe a typical single-butterfly session; the Swarm workflow differs only in Section 7 being visible and the bake step writing curves for all agents.

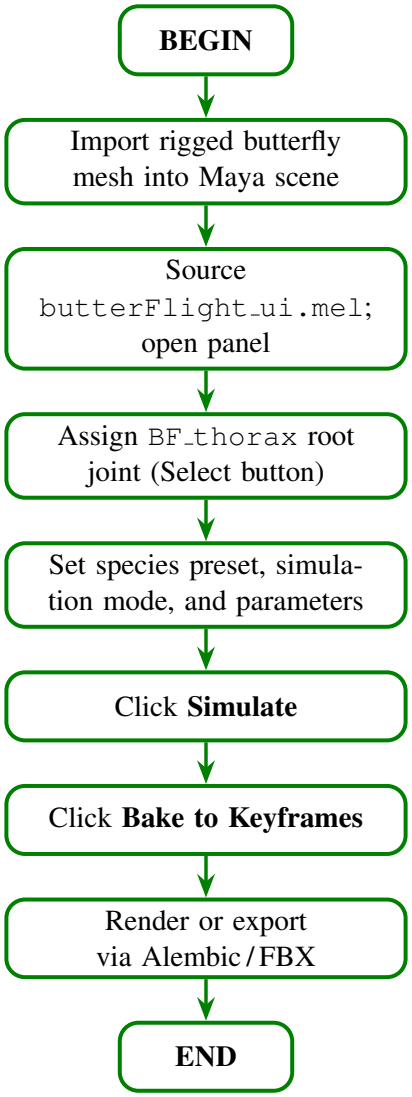

Figure 5: ButterFlight workflow. After opening the panel and assigning a rig, the user configures species, mode, and simulation parameters, runs the simulation, then bakes the result to standard Maya keyframes and optionally triggers an Alembic or FBX export.

2. Authoring Tool Development

2.1 Technical Approach

2.1.1 Algorithm Details

2.1.2 Maya Interface and Integration

2.1.3 Software Design and Development

2.2 Target Platforms

2.2.1 Hardware

2.2.2 Software

2.3 Software Versions

2.3.1 Alpha Version Features (first prototype)

2.3.2 Beta Version Features

2.3.3 Description of Demos and Tutorials

3. Work Plan

3.1 Tasks and Subtasks

3.2 Milestones

3.2.1 Alpha Version

3.2.2 Beta Version

3.3 Schedule

4. Related Research

References

[1] CHEN Q., LU T., TONG Y., LUO G., JIN X., DENG Z.: A Practical Model for Realistic Butterfly Flight Simulation. *ACM Transactions on Graphics (TOG)* 1, 1 (2022), 12 pages.

[2] ELLINGTON C. P.: The aerodynamics of hovering insect flight. I. The quasi-steady analysis. *Philosophical Transactions of the Royal Society of London. B, Biological Sciences* 305, 1122 (1984), 1–15.

[3] ELLINGTON C. P.: The aerodynamics of hovering insect flight. V. A vortex theory. *Philosophical Transactions of the Royal Society of London. B, Biological Sciences* 305, 1122 (1984), 115–144.

[4] REYNOLDS C. W.: Flocks, herds and schools: A distributed behavioral model. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques* (1987), pp. 25–34.

[5] BETTS C. R., WOOTTON R. J.: Wing shape and flight behaviour in butterflies (Lepidoptera: Papilionoidea and Hesperioidea): a preliminary analysis. *Journal of Experimental Biology* 138, 1 (1988), 271–288.

[6] PERLIN K.: Improving noise. In *ACM Transactions on Graphics (TOG)*, Vol. 21. ACM, 2002, pp. 681–682.

[7] WU J., POPOVIĆ Z.: Realistic modeling of bird flight animations. *ACM Transactions on Graphics (TOG)* 22, 3 (2003), 888–895.

[8] BRIDSON R., HOURIHAM J., NORDENSTAM M.: Curl-noise for procedural fluid flow. *ACM Transactions on Graphics (TOG)* 26, 3 (2007), 46–es.

[9] WILSON T., ALBERTANI R.: Wing-flapping and abdomen actuation optimization for hovering in the butterfly *Idea leuconoe*. In *52nd Aerospace Sciences Meeting* (2014), 0009.

[10] WANG X., JIN X., DENG Z., ZHOU L.: Inherent noise-aware insect swarm simulation. In *Computer Graphics Forum*, Vol. 33. Wiley Online Library, 2014, pp. 51–62.

[11] WANG X., REN J., JIN X., MANOCHA D.: BSwarm: biologically-plausible dynamics model of insect swarms. *Proceedings of the 14th ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2015), 111–118.

[12] SRIDHAR M., KANG C.-K., LANDRUM D. B.: Instantaneous lift and motion characteristics of butterflies in free flight. In *46th AIAA Fluid Dynamics Conference* (2016), 3252.

[13] CHEN Q., LUO G., TONG Y., JIN X., DENG Z.: Shape-constrained flying insects animation. *Computer Animation and Virtual Worlds* 30, 3–4 (2019), e1902.

[14] SRIDHAR M., KANG C.-K., LEE T.: Geometric formulation for the dynamics of monarch butterfly with the effects of abdomen undulation. In *AIAA Scitech 2020 Forum* (2020), 1962.

[15] WILL H.: Dynamic Bone. Unity Asset Store, 2020. `https://assetstore.unity.com/packages/tools/animation/dynamic-bone-16743`.