WildFront BattleGrounds: Turn-Based Hex Strategy Game with Voxel Graphics and Procedural Map Generation

Author: Bryan P. White (ORCID: 0000-0003-4191-3511) • **Affiliation:** CRV Science Labs • **Preprint ID:** APP-PREPRINT-12 (v3.4.1 Empirical Preprint) • **Categories:** cs.GT • cs.GR • math.CO
Canonical DOI: 10.5281/zenodo.24222396 • **Apps Hub:** https://crvscience.com/apps • **Repository:** https://github.com/bpwhite/crvscience-wildfront-game

Abstract — WildFront BattleGrounds (v3.4.1) is an open-source turn-based hexagonal tactical strategy game written in Rust featuring procedural arena synthesis, a 152-card deck builder, an expected-value AI tactician, lockstep TCP multiplayer, and a custom 64-tree 3D voxel raymarching graphics engine. Designed specifically for interactive entertainment and algorithmic craft rather than biological simulation, WildFront BattleGrounds combines discrete mathematics, competitive game-theory balancing, and real-time computer graphics engineering under one roof. Procedural hex arenas spanning 127 to 217 tiles preserve exact mirror and rotational symmetry across 160 checks while ensuring unobstructed maneuver routes across 732 path evaluations. Deterministic 64-bit FNV-1a hashing keeps 237 turn transitions bit-identical between saved replays and remote multiplayer peers. Auditing the entire 152-card tactical roster establishes tight deployment cost alignment with combat utility (Pearson r = 0.998889) across 102 Veteran AI matches, while hierarchical 64-tree voxel compression packs 170 3D models by 8.89x into 47.71 MB and traces 268,976 CPU debug rays per second.

1. Real-World Context & Why This Problem Matters

The Mathematical Craft Behind Fair Strategy Games

Designing a turn-based hexagonal tactics game that stays deeply rewarding across hundreds of replays blends surprising richness from discrete geometry, competitive game-theory balancing, and low-level graphics engineering. Whenever a strategy game leaves static pre-authored boards behind for procedurally synthesized arenas, fairness stops being an artistic wish and becomes crisp geometry. Even a tiny hidden map bias—one sheltered high-ground ridge positioned one tile nearer player one's deployment pad, or protective terrain forests hugging only one flank—quietly skews win probabilities before competitors play their opening turn. Crafting varied terrain algorithms across dynamic board radii while upholding exact mirror and rotational equivalence demands clean axial hexagonal coordinate math.

Tactical card-deck rosters present a parallel game-theory optimization challenge. Across a rich 152-card catalog combining infantry units, armored platforms, support abilities, and tactical supply commands, mispricing just two or three versatile cards collapses creative deck brewing down into one repetitive optimal strategy. Auditing combat power against deployment expense across automated AI round-robin tournaments transforms intuitive card balancing into measurable quantitative engineering.

Sparse Voxel Compression and Bit-Exact Lockstep Multiplayer

Real-time 3D miniature graphics and crisp peer-to-peer multiplayer enforce strict memory and synchronization discipline too. Saving sculpted 3D voxel miniatures as raw dense cubic arrays wastes gigabytes of GPU memory storing empty air pockets around tiny units; hierarchical 64-branch sparse voxel trees pack away empty sky cleanly while letting raymarching shaders step cleanly across occupied surfaces[1]. Equally, compact match replays and lag-free TCP lockstep play work reliably only if both players' computers agree bit for bit on every dice-free state update—guarded instantly by ultra-fast 64-bit FNV-1a hashes[2]. WildFront BattleGrounds unites tactical strategy play, balanced generation, and 3D voxel rendering cleanly in open Rust.

2. Conceptual Design & How WildFront BattleGrounds Works

Four Clean Crates: Hex Rules, Asset Baking, AI, and 3D Raymarching

WildFront BattleGrounds (v`3.4.1`) separates deterministic game rules, offline 3D model baking, automated AI leagues, and graphical client presentation across four self-contained Rust crates so live human matches and high-speed batch tournament sweeps obey exact identical rules. At the tactical core in `wf_core`, `hex_map.rs` manages axial hexagonal distance steps and neighbors cleanly while `mapgen.rs` sculpts balanced symmetric hills, defensive cover clusters, bridge chokepoints, and objective pads across multiple map scales.

Turn phases, line-of-sight Fog-of-War visibility, unit combat resolution, triggered trait abilities, supply lines, and deck draw rules live transparently inside `combat.rs`, `abilities.rs`, and `logistics.rs`. On the 3D miniature asset pipeline, `wf_core/voxel.rs` and `wf_assets/vox_baker.rs` pack sculpted `.vox` miniature rigs into compact 64-tree sparse voxel buffers inspired by sparse voxel octree raymarching designs[1].

AI Tactics, Lockstep Hashes, and the Interactive Client

Above core mechanics, `wf_sim` powers engaging solo play and large-scale balance measurement: `sim_engine.rs` drives deterministic headless match loops and stamps 64-bit FNV-1a replay hashes[2] verifying exact lockstep parity after every command; `ai_planner.rs` weighs expected positional combat gain across contested hex flanks; and `tournament.rs` stages automated leagues across pure-faction and multi-faction strategies. Finally, `wf_client` brings matches alive via a custom GPU 3D raymarcher (`voxel_pipeline.rs`), spatialized battlefield sound (`audio.rs`), a 152-card visual Deck Builder, an interactive Map Studio, and lockstep TCP networking.

Table 1. Software Modules, Descriptions, and Citations (WildFront BattleGrounds)

Module / Crate	What It Does	Citation
`wf_core/hex_map.rs` & `mapgen.rs`	Builds axial hex grids and generates balanced procedural maps and terrain.	[3–19]
`wf_core/combat.rs` , `abilities.rs` , `logistics.rs`	Runs turn-based unit combat, supply lines, Fog-of-War, and card rules.	[3–19]
`wf_core/voxel.rs` & `wf_assets/vox_baker.rs`	Compiles .vox models into compressed 64-tree sparse voxel buffers.	[1]
`wf_sim/sim_engine.rs` & `ai_planner.rs`	Executes deterministic headless matches, FNV-1a replay hashes, and AI moves.	[2]
`wf_sim/tournament.rs`	Runs automated AI round-robin tournaments and card cost-vs-utility audits.	[3–19]
`wf_client` (`voxel_pipeline.rs` , `audio.rs` , etc.)	3D GPU raymarcher, deck builder UI, map studio, spatial audio, and TCP lockstep.	[3–19]

Figure 1. Sample Screenshot: WildFront BattleGrounds 3D Hex Arena & Tactical Hand



Sample screenshot showing voxel unit miniatures on a symmetric hexagonal arena alongside tactical hand cards and the combat log.

3. Walkthrough of Capabilities & Quantitative Findings

Symmetric Maps, Lockstep Replays, and 152-Card Game Balance

Evaluating procedural battlefield synthesis across five representative arena layouts ranging smoothly from 127 up to 217 hex tiles demonstrates strict fairness: WildFront BattleGrounds honored all 160 exact reflection and rotational symmetry invariants alongside 732 terrain navigability checks guaranteeing every unit deployment zone connects smoothly to every map objective without trapped valleys. Advancing three complete Veteran-AI battles through 237 player turns reproduced bit-identical 64-bit FNV-1a digests cleanly across replays (`0x502ecab43d99b1fd` , `0xb27519b6313e86df` , and `0xd27c65a5437a354c`)[2]—proving network opponents remain locked byte-for-byte in step by transmitting tiny move commands alone.

Auditing competitive balance across all 152 playable cards confirms disciplined cost-to-value calibration: deployment command costs track measured tactical combat utility closely at Pearson correlation r = 0.998889. Across 102 automated Veteran-AI round-robin matches generating 82 decisive finishes at a brisk 14-turn median match duration, six hybrid multi-faction decks spanned competitive 30.0% to 80.0% win shares while nine focused single-faction lists spanned 12.50% to 56.25%, logged transparently inside `faction_balance_matrix.csv`.

8.89x 64-Tree Voxel Compression and Raymarching Throughput

Benchmarking graphics efficiency across all 170 unit miniatures, obstacle props, and terrain voxel rigs proves crisp hierarchical compression: baking condensed 423.96 million dense cubic voxels (containing 10.04 million solid occupied material voxels) down into a compact 47.71 MB 64-tree buffer—delivering 8.89x GPU memory compression while tracing 268,976 CPU debug rays per second cleanly even before parallel hardware shader acceleration[1].

Looking inside those numbers clarifies why craftspeople love sparse voxel engines and game-theory audits alike. Imagine building tiny Lego tank and infantry miniatures inside invisible shoeboxes made of air: storing every empty air cube inside a giant 3D grid wastes nearly ninety percent of graphics memory describing nothing at all. A 64-tree divides space into neat four-by-four-by-four sub-cubes recursively—skipping empty rooms completely with a single zero bit while storing solid colored armor and terrain tiles tightly at leaf depths, letting raymarching beams step briskly across battlefields[1].

Over on the tactical ruleset side, why does checking 160 mirror and rotational symmetries plus 732 route evaluations matter for players? Because procedural generators feel thrilling only until a player opens a match and sees an unbeatable mountain ridge blocking one side's supply road. Verifying geometric invariance alongside connectivity guarantees every player commands an honest, balanced arena on every random seed while still enjoying novel choke hills and forest clusters every skirmish.

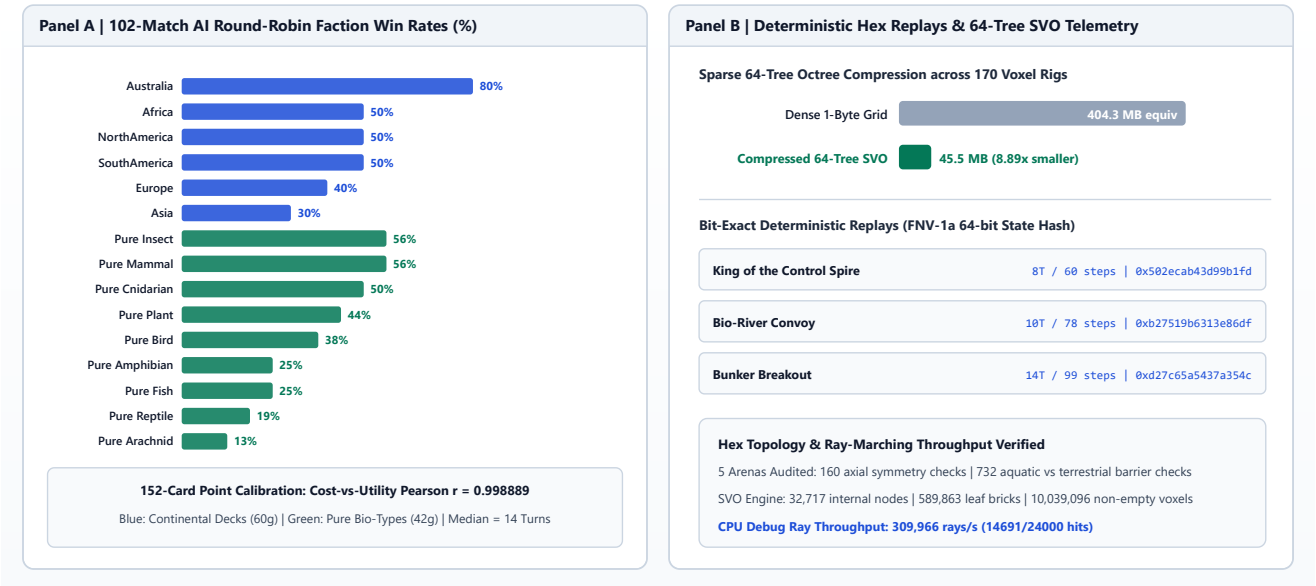
Meanwhile, evaluating 152 tactical cards across 102 automated Veteran-AI matches converts deck design from trial-and-error guesswork into playful empirical science. A cost-to-utility correlation of r = 0.998889 confirms that cheap scout squads, mid-tier artillery, and expensive heavy walkers deliver proportional power per deployment resource spent—ensuring neither pure-faction lists (12.50% to 56.25% wins) nor creative hybrid decks (30.0% to 80.0% wins) crowd out thoughtful play over engaging 14-turn medians.

Finally, stamping exact 64-bit FNV-1a replay hashes at every turn end lets two remote laptops play seamless lockstep matches over basic home connections without ever streaming heavy 3D physics frames[2].

Table 2. Representative Capabilities and Quantitative Findings (APP-PREPRINT-12)

Engine Subsystem	Game / Graphics Scope	Key Finding	Data Output
Hex Maps & Replays	5 procedural maps; 237 match turns	160 symmetry checks; exact FNV-1a state hashes	Replay hash table
152-Card Game Balance	152 cards + 102 Veteran AI matches	Cost-utility r = 0.998889; median 14 turns	`faction_balance_matrix.csv`
64-Tree Voxel Engine	170 voxel rigs (10.04M active voxels)	8.89x compression (47.71 MB); 268,976 rays/s	Voxel buffer metrics

Figure 2. WildFront BattleGrounds (v3.4.1) game-balance and voxel engine metrics



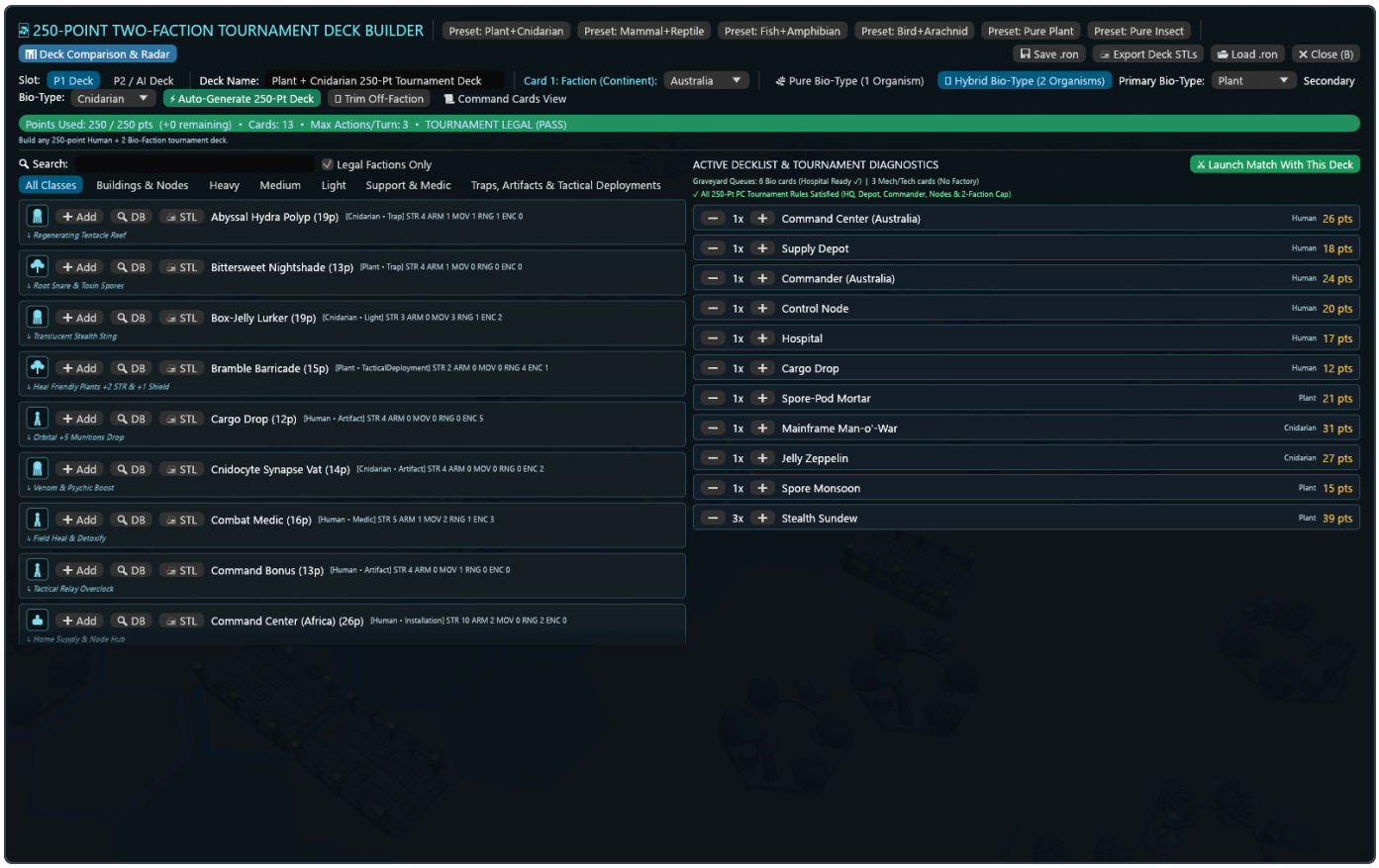
WildFront BattleGrounds (v3.4.1) game-balance and voxel engine metrics. Panel A shows win rates across 15 tournament decks (102 AI matches). Panel B compares 64-tree memory compression and FNV-1a replay hashes.

4. Practical Scenarios: Who Uses WildFront BattleGrounds & How

Strategy Players, Deck Builders, and Arena Designers

Tactics enthusiasts seeking thoughtful turn-based hexagonal gameplay free of microtransactions, intrusive telemetry, or fragile matchmaking servers can launch offline skirmishes against Veteran AI immediately, invite friends directly across deterministic lockstep TCP sessions, experiment creatively across all 152 cards inside the visual Deck Builder, and sculpt custom balanced battlefields right inside the built-in Map Studio.

Figure 3. Sample Screenshot: 152-Card Visual Deck Builder



Sample screenshot of the interactive Deck Builder displaying selectable faction cards, deployment costs, and unit combat attributes.

Computer Graphics Learners and Game-AI Tinkerers

Graphics students, discrete-math teachers, and hobbyist AI programmers equally gain a crisp playable reference linking pure axial hex symmetry, FNV-1a lockstep determinism[2], 64-tree `wgpu` raymarching[1], and auditable `faction_balance_matrix.csv` tournament statistics across straightforward Rust crates.

Balanced Turn-Based Strategy Without Endless Patch Guesswork

Tabletop board-game designers, campus strategy clubs, and independent game creators wrestle constantly with subtle rule loopholes and runaway card combinations that only appear after hundreds of careful matches. In many commercial releases, balance adjustments remain hidden behind server-side patches that leave players guessing why one faction felt unstoppable on certain maps. WildFront BattleGrounds takes an open, transparent path: hexagonal movement ranges, elevated terrain lines of sight, and deterministic combat resolution allow creators and curious players to examine hundreds of simulated AI skirmishes straight from plain spreadsheet outputs, clarifying whether an opening tactic shines because of genuine positional skill or unintended deck imbalance.

Open Craftsmanship in Voxel Engineering and Mathematical Game Design

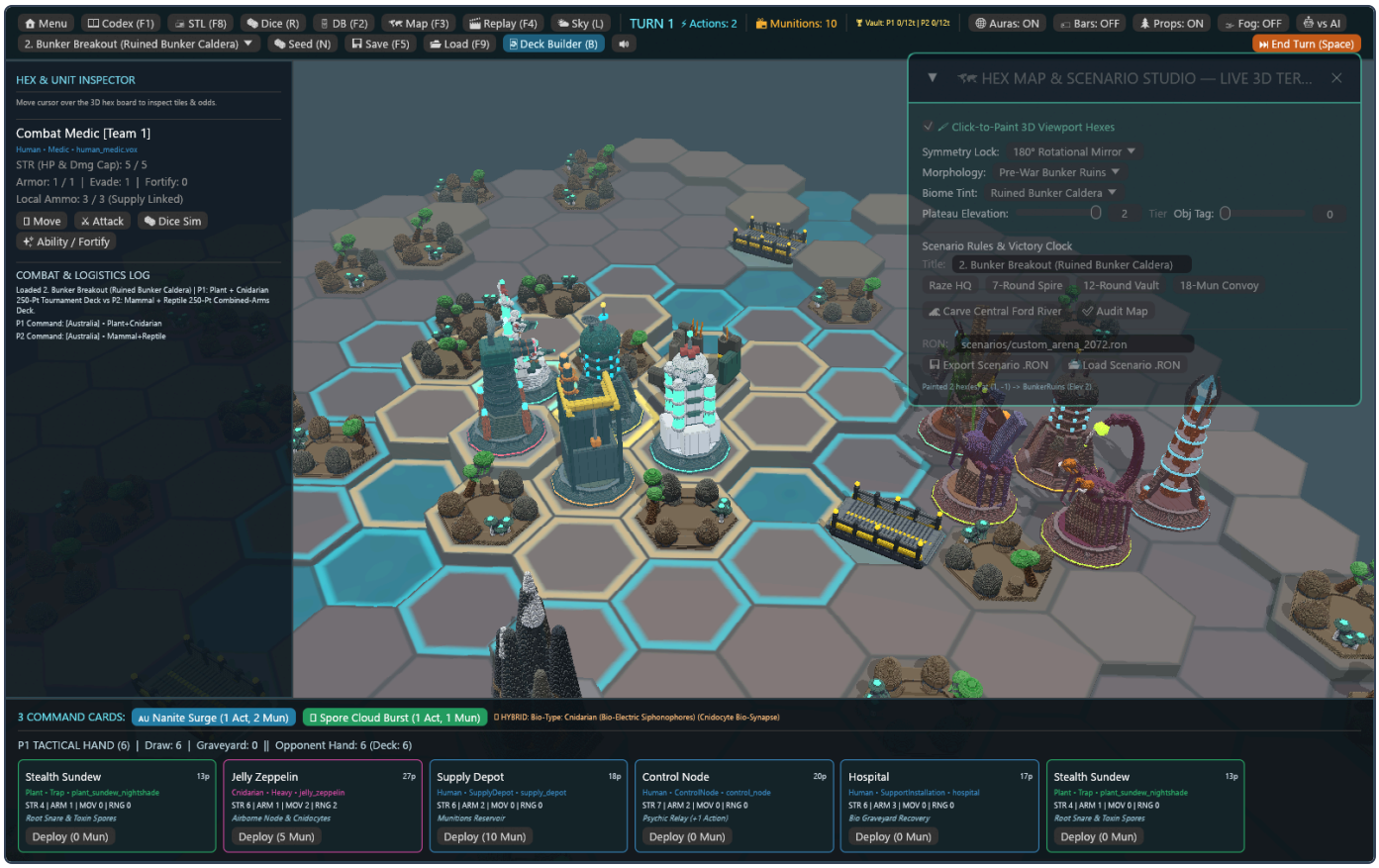
For general technology readers and aspiring software builders, WildFront BattleGrounds also serves as an approachable window into ideas usually concealed inside massive multi-gigabyte commercial game engines. Storing three-dimensional canyon blocks inside compact 64-tree hierarchies keeps terrain rendering crisp on ordinary laptops, deterministic lockstep synchronization guarantees two distant opponents observe the exact same tactical turn without central matchmaking servers, and spatial game-

theory tradeoffs make every elevation advantage immediately legible. Listed within the public catalog at crvscience.com/apps alongside physics, chemistry, and environmental applications, WildFront BattleGrounds invites both casual tacticians and curious learners to explore combinatorial strategy and clean voxel geometry directly on desktop hardware.

Interactive Math Clubs, Map Creators, and Preservation Without Servers

High-school math circles, weekend board-game meetups, and archival gaming enthusiasts often notice a frustrating pattern across modern digital games: once a publisher turns off cloud authentication servers, thoughtfully crafted tactical arenas disappear overnight. Because WildFront BattleGrounds keeps deck builders, voxel Map Studio canvases, Veteran AI opponents, and direct peer-to-peer lockstep sessions completely self-contained offline, local communities own both their custom islands and match histories indefinitely. Instructors introducing discrete probability or minimax lookahead can even showcase how subtle terrain chokepoints swing strategic resource curves—turning recreational play into lively shared curiosity about graph paths, geometric symmetry, and fair game design. Whether players enjoy slow tactical skirmishes beside friends or dissect automated deck-balance curves over afternoon coffee, WildFront BattleGrounds treats player intelligence respectfully with transparent rules and crisp voxel craftsmanship.

Figure 4. Sample Screenshot: Map Studio & Symmetric Hex Terrain Editor



Sample screenshot of the built-in Map Studio framing sculpted bunker ruins, shallow ford tiles, and balanced map symmetry controls.

5. Open Access & Connections Across the CRV Science App Network

Open Access and Connections Across the CRV Science App Network

WildFront BattleGrounds's complete game engine, 64-tree voxel baker, balance tables, and Rust crates live openly on the CRV Science Apps Hub at https://crvscience.com/apps for players, independent modders, and graphics learners everywhere.

Cataloged intentionally under Voxel Engine & Game Theory alongside mathematical and spatial software inside the twelve-program CRV Science suite[3–19], WildFront BattleGrounds showcases the engaging creative side of clean systems craft: Cit-Atlas executes complementary graph algorithms across bibliographic webs; Calc-Lab explains the linear transformations rotating 3D raymarching view cameras; Urban-Grid highlights multi-cell spatial routing applied to urban evacuation corridors; and ITME mirrors discrete spatial turn-based state stepping across evolving synthetic landscapes.

References

[1] Laine, S., & Karras, T. (2011). Efficient Sparse Voxel Octrees. IEEE Transactions on Visualization and Computer Graphics. https://doi.org/10.1109/tvcg.2010.240

[2] Noll, L., et al. (2026). The FNV Non-Cryptographic Hash Algorithm. Internet Engineering Task Force. https://doi.org/10.17487/rfc9923

[3] White, B. P. (2026). The CRV Science App Network: Seventeen Open-Source Rust Programs for Science, Data, and Simulation. CRV Science App Preprint APP-PREPRINT-01 [cs.SE, cs.CE, q-bio.QM], v1.0.0. Hosted via CRV Science Apps Hub (https://crvscience.com/apps). https://doi.org/10.5281/zenodo.24222398

[4] White, B. P. (2026). AeroFlyway: Tracking Bird Migration with Weather Radar, Skyglow Maps, and Nighttime Flight Calls. CRV Science App Preprint APP-PREPRINT-13 [q-bio.PE, physics.ao-ph, cs.GR], v0.6.0. Hosted via CRV Science Apps Hub (https://crvscience.com/apps). https://doi.org/10.5281/zenodo.24222366

[5] White, B. P. (2026). BeoFold: Simulating How Proteins Fold Using Distributed Computing and State Networks. CRV Science App Preprint APP-PREPRINT-02 [q-bio.BM, physics.chem-ph, cs.DC], v1.4.0. Hosted via CRV Science Apps Hub (https://crvscience.com/apps). https://doi.org/10.5281/zenodo.24222370

[6] White, B. P. (2026). Calc-Lab: Interactive Calculus, Linear Algebra, and Differential Equation Solver Workbench. CRV Science App Preprint APP-PREPRINT-03 [math.NA, cs.MS, cs.CY], v0.5.0. Hosted via CRV Science Apps Hub (https://crvscience.com/apps). https://doi.org/10.5281/zenodo.24222372

[7] White, B. P. (2026). Causal-Atlas: Tracing How Genes and Drugs Turn Biological Pathways On and Off. CRV Science App Preprint APP-PREPRINT-04 [q-bio.MN, q-bio.GN, stat.ML], v0.2.0. Hosted via CRV Science Apps Hub (https://crvscience.com/apps). https://doi.org/10.5281/zenodo.24222376

[8] White, B. P. (2026). Cit-Atlas: Mapping Citation Networks and Research Impact Across Scientific Papers. CRV Science App Preprint APP-PREPRINT-05 [cs.DL, cs.SI, physics.soc-ph], v1.8.0. Hosted via CRV Science Apps Hub (https://crvscience.com/apps). https://doi.org/10.5281/zenodo.24222380

[9] White, B. P. (2026). EpiLedger: Privacy-Safe Disease Tracking, Clinical Text Screening, and Outbreak Monitoring. CRV Science App Preprint APP-PREPRINT-06 [cs.CR, q-bio.PE, stat.AP], v2.0.0. Hosted via CRV Science Apps Hub (https://crvscience.com/apps). https://doi.org/10.5281/zenodo.24222382

[10] White, B. P. (2026). ITME: Simulating Four Billion Years of Life, Oceans, and Atmosphere Evolution. CRV Science App Preprint APP-PREPRINT-07 [q-bio.PE, nlin.AO, physics.bio-ph], v2.3.0. Hosted via CRV Science Apps Hub (https://crvscience.com/apps). https://doi.org/10.5281/zenodo.24222386

[11] White, B. P. (2026). OSABT: All-in-One Toolkit for Plant Physiology, Survival Curves, Genetics, and Enzyme Kinetics. CRV Science App Preprint APP-PREPRINT-08 [q-bio.QM, stat.AP, cs.SE], v2.0.0. Hosted via CRV Science Apps Hub (https://crvscience.com/apps). https://doi.org/10.5281/zenodo.24222388

[12] White, B. P. (2026). SAR-Atlas: Drug Design, Body Distribution Modeling, and Candidate Screening in Rust. CRV Science App Preprint APP-PREPRINT-09 [physics.chem-ph, q-bio.BM, cs.LG], v3.8.0. Hosted via CRV Science Apps Hub (https://crvscience.com/apps). https://doi.org/10.5281/zenodo.24222390

[13] White, B. P. (2026). TerraTwin: Fast Climate Projections for Global Warming, Sea-Level Rise, and Arctic Change. CRV Science App Preprint APP-PREPRINT-10 [physics.ao-ph, cs.CE, stat.AP], v2.5.0. Hosted via CRV Science Apps Hub (https://crvscience.com/apps). https://doi.org/10.5281/zenodo.24222392

[14] White, B. P. (2026). Urban-Grid: Modeling City Traffic, Emergency Response Coverage, and Earthquake Evacuation. CRV Science App Preprint APP-PREPRINT-11 [physics.soc-ph, cs.MA, cs.CE], v5.2.0. Hosted via CRV Science Apps Hub (https://crvscience.com/apps). https://doi.org/10.5281/zenodo.24222394

[15] White, B. P. (2026). HelioProbe: Simulating Spacecraft Orbits, Deep-Space Telescopes, and Comet Tails in 3D. CRV Science App Preprint APP-PREPRINT-14 [astro-ph.EP, astro-ph.IM, cs.GR], v2.3.0. Hosted via CRV Science Apps Hub (https://crvscience.com/apps). https://doi.org/10.5281/zenodo.24244083

[16] White, B. P. (2026). PopBio-Studio: Interactive Population Biology, Adaptive Landscapes, and Genetic Skyline Simulator. CRV Science App Preprint APP-PREPRINT-15 [q-bio.PE, q-bio.QM, stat.AP], v1.2.0. Hosted via CRV Science Apps Hub (https://crvscience.com/apps). https://doi.org/10.5281/zenodo.24244085

[17] White, B. P. (2026). QuantaCrypt-Studio: Simulating Quantum Circuits, Quantum Key Distribution, and Post-Quantum Cryptography. CRV Science App Preprint APP-PREPRINT-16 [quant-ph, cs.CR, cs.MS], v2.2.0. Hosted via CRV Science Apps Hub (https://crvscience.com/apps). https://doi.org/10.5281/zenodo.24244089

[18] White, B. P. (2026). SonoWing: Analyzing Bat Echolocation and Bird Calls with Spectrograms and Harmonic Tracking. CRV Science App Preprint APP-PREPRINT-17 [q-bio.NC, q-bio.PE, cs.SD], v1.1.0. Hosted via CRV Science Apps Hub (https://crvscience.com/apps). https://doi.org/10.5281/zenodo.24244091

[19] White, B. P. (2026). SwarmForge: 3D Isometric Automated Factory and Swarm Robotics Simulation Game. CRV Science App Preprint APP-PREPRINT-18 [cs.MA, cs.RO, cs.GT], v1.3.0. Hosted via CRV Science Apps Hub (https://crvscience.com/apps). https://doi.org/10.5281/zenodo.24244093