ViperMesh Case Study Part One

Deterministic Blender agent harness benchmarked against the Anthropic x Blender MCP server baseline, with an EvoRig evidence loop.

6 of 7 live speed wins	2.534x mean speedup	+8.19 pts visual rubric lift	-90.91% local token reduction

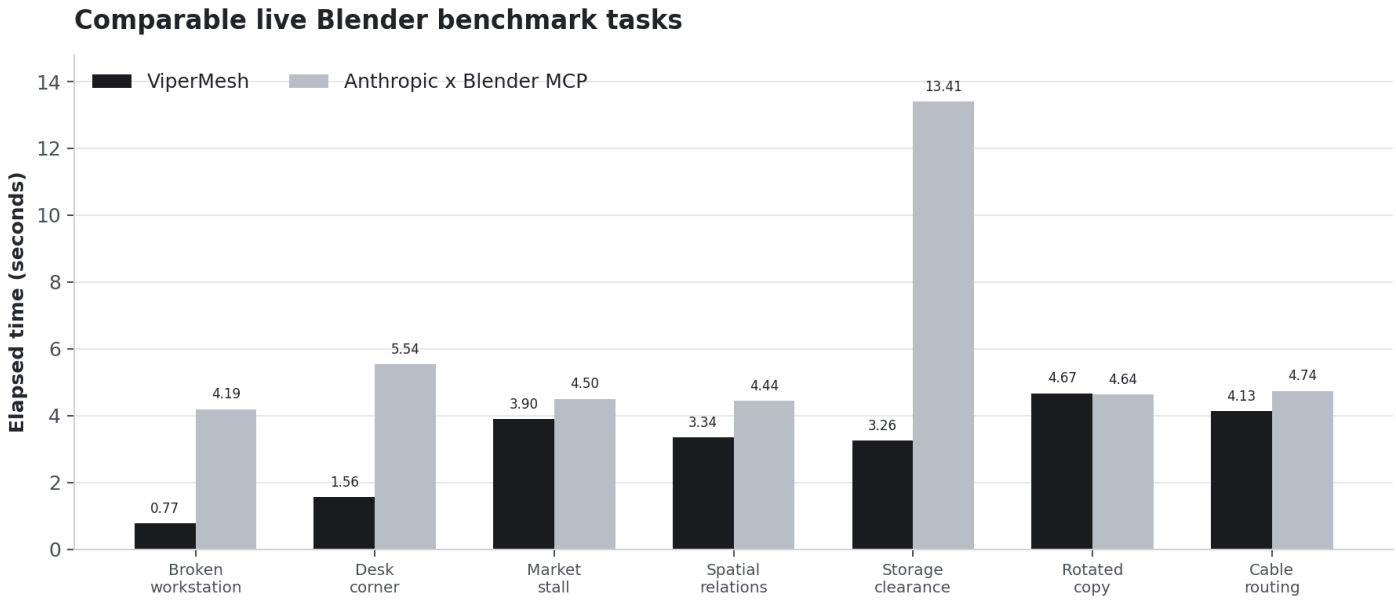
Executive Summary

ViperMesh is a deterministic Blender agent harness. It replaces avoidable broad execute_code calls with auditable tools, RAG-backed tool guidance, scene inspectors, render checks, and repeatable benchmark runners. EvoRig is the evidence loop that records artifacts and improvement decisions so harness changes can be tied to benchmark outcomes.

Both harnesses used OpenAI GPT 5.5 High as the acting model. The comparison therefore isolates the harness and tool surface rather than a model mismatch.

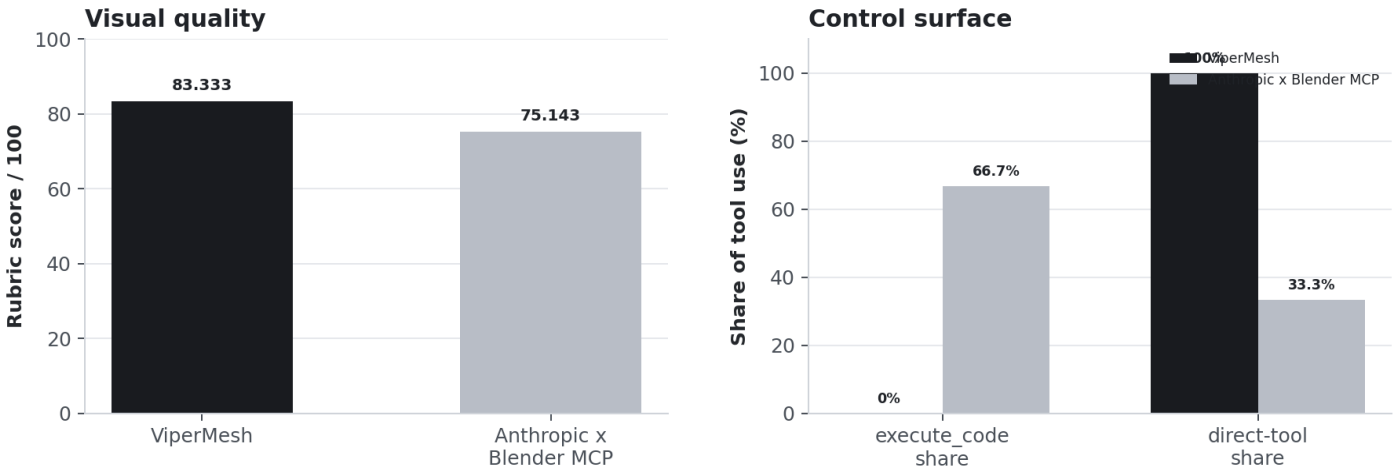
The current public result is scoped. ViperMesh was faster in 6 of 7 comparable live tasks, scored higher on a seven-pair neutral LLM visual rubric, and reduced local acting-agent tokens on one comparable Codex-operated token-window pair. Provider billing totals are not claimed.

Measured Benchmark Performance



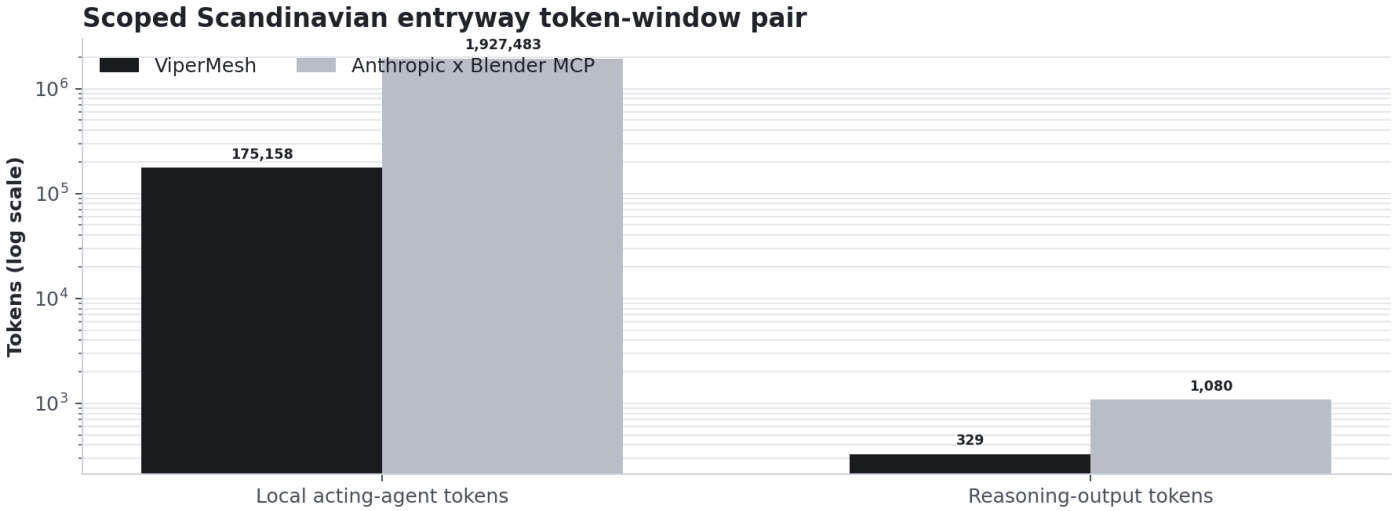
Each pair is a measured live or agentic Blender task. The Anthropic x Blender MCP baseline was 34 ms faster on the rotated-copy task; ViperMesh was faster on the other six comparable tasks.

Quality And Control Surface



Visual quality was evaluated through a neutral, non-biased LLM rubric across seven live render pairs and four criteria. The control-surface chart covers 11 latest-per-label comparisons. It shows ViperMesh moving work from generated Python toward deterministic tools, not a general claim that arbitrary code is never needed.

Token Evidence



The token chart is logarithmic so both systems remain visible. It represents one comparable Scandinavian entryway Codex token-window pair: 175,158 versus 1,927,483 local acting-agent tokens and 329 versus 1,080 reasoning-output tokens. These are local telemetry measurements, not final provider invoices.

Architecture And Evidence Method

Task prompts and image references are prepared with benchmark context, passed to the ViperMesh MCP gateway and Blender addon, then inspected for scene, grounding, render, spatial relation, and command-trace evidence. The result is normalized into comparison artifacts. EvoRig tracks the operational map, observations, and candidate improvements outside the addon.

Evidence class	Current status
Live and agentic elapsed benchmark pairs	7 comparable tasks
Latest-per-label comparison traces	11 of 11 complete
Neutral LLM visual rubric	7 render pairs, 4 criteria
Provider-billable token totals	Not yet complete

Interview Explanation

I built ViperMesh because general Blender MCP agents can hide essential behavior inside broad generated code. The harness introduces deterministic tools and inspectors so performance, reliability, and scene evidence can be examined rather than inferred from a render alone. EvoRig turns that development process into a repeatable evidence loop.

Resume-Ready Claims

Built and benchmarked a deterministic Blender MCP harness that was faster in 6 of 7 comparable live tasks, reduced execute_code dependency by 66.7 percentage points across 11 selected comparisons, and achieved 83.333 versus 75.143 on a scoped neutral LLM visual-quality rubric. Implemented local token-window telemetry showing a 90.91% reduction in local acting-agent token usage on a comparable live pair.

Sources: ViperMesh case-study evidence bundle, current comparison suite summaries, command traces, live visual review artifacts, and scoped Codex token-window telemetry. Repository: github.com/Ker102/ViperMesh