LTX-2 Video Generation — qvac-ext-stable-diffusion.cpp + Bare Addon

Title	LTX-2 Video Generation — qvac-ext-stable-diffusion.cpp + Bare Addon
Publish Date	Mar 24, 2026
Submission End Date	Jun 24, 2026
Reward	10’000 USDt

Problem Summary

What is the problem, deficiency, or gap which will be addressed by this project?

The LTX-2 family of models (specifically LTX 2.3) is a state-of-the-art open-weights video generation model (14B-parameter DiT, Gemma 3 text encoder, spatiotemporal Video-VAE) supporting text-to-video (T2V) and image-to-video (I2V). Its official stack is Python/PyTorch-only, making it impractical for edge and embedded use.
qvac-ext-stable-diffusion.cpp already provides ggml-based inference for image diffusion models (SD, Flux, Wan, etc.) across CPU, Vulkan, and Metal, but does not yet support any video diffusion. This grant funds the addition of LTX-2 T2V and I2V support to that fork, plus a Bare runtime addon to expose video generation to JavaScript applications in the QVAC ecosystem.

Impact / Reason Why

What is the metric or goal this bounty aims to improve?

- **Accessibility** — Local video generation on commodity hardware (macOS, Linux, Windows) without Python or vendor-locked GPU drivers.
- **Performance** — ggml quantisation (Q4–F16) reduces VRAM/RAM requirements for consumer devices.
- **Ecosystem growth** — Extends `qvac-ext-stable-diffusion.cpp` into video generation; ships a Bare addon following the `bare-llama-cpp` pattern.

Scope Items

What important items are in scope for the project?

- **Two PRs** — All work is delivered as two pull requests:
  - **PR 1** → tetherto/qvac-ext-stable-diffusion.cpp — C/C++ implementation (DiT, VAE encoder+decoder, text encoder, scheduler, conversion tooling, CLI, C API).

    - **PR 2** → Bare addon mono-repo *(repo link TBD)* — JavaScript bindings wrapping the C++ API, built with `bare-make`, with prebuilds for macOS ARM64, Linux x86-64, and Windows x86-64.
- **Generation modes** — Both **T2V** and **I2V**. I2V enables character consistency, scene bootstrapping from a reference frame, and static-image animation.
- **Model conversion** — Safetensors → GGUF at Q4_0, Q5_1, Q8_0, and F16.
- **Core model components** — DiT (14B video stream), Video-VAE encoder+decoder (32×32×8 per token), Gemma 3 text encoder, at least one noise scheduler (e.g. LinearQuadratic) + CFG guidance.
- **Backends** — **Vulkan** (Linux/Windows GPU), **Metal** (macOS GPU), **CPU AVX/AVX2/AVX512** (x86-64), **CPU NEON** (ARM64).
- **CLI** — Accepts a text prompt (and optionally an input image for I2V), outputs MP4 or raw frames.
- **C/C++ API** — Public header covering T2V and I2V inference.
- **Bare addon** — npm-publishable package exposing T2V and I2V to Bare JavaScript apps.
- **Docs** — Build instructions, usage examples, quantisation guidance, Bare addon API reference.
- **Tests** — Inference correctness (PSNR/SSIM vs. PyTorch reference for T2V and I2V), cross-platform CI (macOS ARM64, Linux x86-64, Windows x86-64), Bare addon smoke tests.
- **Benchmarks** — Wall-clock comparison against the PyTorch/Diffusers pipeline on identical hardware at F16. The ggml implementation must be at least **10% faster** *(⚠️ TBD — margin subject to revision)*. On macOS, Draw Things must also be used as a performance baseline for Metal inference, since it represents the current best-in-class on-device diffusion performance on Apple Silicon.

Scope Exclusions

What important items are out of scope for the project?

- Audio stream (5B audio DiT, Audio-VAE, Vocoder), training/fine-tuning, spatial upscaler, video-to-video, GUI/web UI.
- CUDA / ROCm / SYCL backends (may be contributed but are not acceptance criteria).
- `@qvac/sdk` integration — only the standalone Bare addon is required.

Deliverables

What are the outputs that must be submitted?

- **PR 1 — qvac-ext-stable-diffusion.cpp** — DiT, VAE, text encoder, scheduler, GGUF conversion scripts, CLI, C API header (`ltx2.h` or equivalent). Must follow the repository's existing structure and conventions.
- **PR 2 — Bare addon mono-repo** — JavaScript bindings, `bare-make` build, prebuilds for all three platforms, addon API docs with code samples.
- **GGUF weights** — Q8_0 and Q4_0 checkpoints published to a publicly accessible location (e.g. HuggingFace).
- **Test suite** — CI config (GitHub Actions or equivalent) covering all platforms + Bare addon smoke tests.

- **Benchmark report** — Tokens/s, total generation time, peak memory for ≥2 quantisation levels per backend. Head-to-head vs. PyTorch/Diffusers at F16, and vs. Draw Things on macOS Metal, demonstrating the required speed-up.

Acceptance Criteria / Definition of done

What needs to be achieved for this project to be considered completed?

- ☐ GGUF conversion works at Q4_0, Q5_1, Q8_0, and F16.
- ☐ CLI generates coherent ≥2s video from a text prompt (T2V) and from a text prompt + image (I2V) on every required backend.
- ☐ Output quality ≥ 25 dB PSNR (or ≥ 0.85 SSIM) vs. PyTorch reference at F16, for both T2V and I2V.
- ☐ End-to-end wall-clock at least **10% faster** than PyTorch/Diffusers on identical hardware and precision *(⚠️ TBD — applicants should document methodology so the threshold can be re-evaluated)*. Draw Things on macOS Metal serves as an additional performance baseline.
- ☐ Clean compilation (`-Wall -Wextra`) on Clang, GCC, and MSVC.
- ☐ C API usable by an external program without the CLI.
- ☐ Bare addon loads in a Bare session, exposes T2V and I2V via a JavaScript API, and produces valid output.
- ☐ Both PRs follow their target repository's conventions and require only minor modifications before merge.
- ☐ All tests pass in CI across all three platforms.
- ☐ A reviewer can build from source and generate a video within 15 minutes using the README.

Milestones

What are the intermediary checkpoints necessary to track project progress?

ID	Description	Deliverables	Reward
M1	Model conversion & scaffolding	• GGUF conversion tool integrated into the fork, CI skeleton, model loads on CPU. • **Delivers** — Conversion scripts, F16 GGUF checkpoint, buildable project, CI green on Linux x86-64.	1’000 USDt
M2	Core CPU inference	• End-to-end T2V and I2V on CPU (AVX x64 + NEON ARM64). • **Delivers** — CLI produces valid video	3’000 USDt

		from both T2V and I2V. Q4 and Q8 checkpoints.	
M3	GPU backends & benchmarks	• Vulkan and Metal backends functional for T2V and I2V, performance tuning. • **Delivers** — CLI runs on Vulkan and Metal. Benchmark report (vs. PyTorch and vs. Draw Things on Mac).	2’500 USDt
M4	Bare addon, docs & tests	• C API, Bare addon with prebuilds, full test suite, documentation, final polish. • **Delivers** — Both PRs ready for review. Bare addon package with prebuilds. CI green on all platforms.	3’500 USDt

Success Indicators / Key Results

What are the key results that help us measure if this project was a success?

- ☐ Build-to-first-video in under 15 minutes from the README.
- ☐ T2V and I2V each complete within 5 minutes for a 2s clip at 512×768 on Apple M2 Pro (Metal) or equivalent Vulkan GPU.
- ☐ ≥10% faster than PyTorch/Diffusers at F16 on identical hardware *(⚠️ TBD)*.
- ☐ Peak memory with Q4_0 ≤ 12 GB RAM (CPU) / ≤ 10 GB VRAM (GPU).
- ☐ Two independent reviewers confirm output quality is comparable to the PyTorch pipeline.
- ☐ Bare addon usable in fewer than 20 lines of JavaScript.

Applicants Requirements

What requirements that applicants must meet?

- C/C++ systems programming and CMake experience.
- Familiarity with ggml / llama.cpp / stable-diffusion.cpp internals (prior contributions preferred).
- Understanding of diffusion architectures (DiT, VAE, schedulers, CFG).
- Experience with Bare or Node.js native addons (N-API / bare-addon / node-gyp).
- Access to ≥2 of the required backends (e.g. Apple Silicon Mac + Linux x86-64 with Vulkan GPU).
- Weekly progress updates in English via GitHub issues/PRs.

Reward & Payment Schedule

What are the payout structure and conditions?

Total: 10 000 USDt, four milestone-based installments:

- **M1** — 1 000 USDt
- **M2** — 3 000 USDt
- **M3** — 2 500 USDt
- **M4** — 3 500 USDt

Each milestone is reviewed within 5 business days. Payment released after reviewer approval. No partial payouts.

Resources & Links

What are additional useful links and resources for applicants?

- qvac-ext-stable-diffusion.cpp — Target repo for PR 1.
- Bare addon mono-repo — *(link TBD)* — Target repo for PR 2.
- LTX-2 GitHub — Official source and model cards.
- LTX-2 on HuggingFace — Pre-trained weights.
- LTX-Video paper (arXiv:2501.00103) — Original architecture.
- LTX-2 Technical Report (arXiv:2601.03233) — LTX-2 architecture.
- HuggingFace Diffusers — LTX-2 — Reference Python implementation (T2V + I2V).
- stable-diffusion.cpp upstream — Upstream project.
- llama.cpp / ggml — Tensor library.
- GGUF Specification — File format docs.
- bare-addon template — Addon template.
- bare-llama-cpp — Reference Bare addon.
- Bare runtime — Target JS runtime.
- Draw Things — macOS Metal performance baseline.