

# Appendix C — Track Records of Computing Power Usage and Team

---

## AuraSense Limited | AISS Application 2026

---

### 1. Track Record of Computing Power Usage

---

#### Current Development Environment

AuraSense has developed and benchmarked the entire SFSVC engine using GitHub Codespaces (cloud VM — 4-core x86-64 with AVX2, 16 GB RAM, no GPU). Despite the limited environment, the team achieved production-level performance metrics, demonstrating efficient use of computing resources and strong optimisation skills.

#### Benchmark Results (Codespaces VM, No GPU)

| Metric           | Result    | Notes                    |
|------------------|-----------|--------------------------|
| Frames Processed | 1,127     | Full demo.mp4 @ 30fps    |
| Throughput       | 125.5 fps | 6 processing lanes       |
| P50 Latency      | 0.399 ms  | End-to-end control path  |
| P95 Latency      | 0.559 ms  | Stable across runs       |
| P99 Latency      | 0.637 ms  |                          |
| Spike Events     | 6,840,849 | Deterministic output     |
| Sparsity         | 93.76%    | Consistent across 2 runs |
| Run Variance     | <2.7% P95 | High reproducibility     |

## Optimisation History (Demonstrating Efficient Resource Use)

| Phase    | Optimisation                 | Latency Impact | Method                           |
|----------|------------------------------|----------------|----------------------------------|
| Baseline | Scalar C++ implementation    | P95: ~50 ms    | Naive per-pixel loop             |
| Phase 1  | AVX2 SIMD frame copy         | P95: -5%       | 256-bit register memcpy          |
| Phase 2  | AVX2 delta spike computation | P95: -40%      | 32 pixels per SIMD op            |
| Phase 3  | Fixed-point SIMD resize      | P95: -10%      | Eliminated floating-point resize |
| Phase 4  | Lock-free queue + multi-lane | P95: ~0.56 ms  | Concurrent 6-lane pipeline       |

## Why GPU Computing Is Now Needed

The current engine operates entirely on CPU with hand-tuned parameters. To advance to production-grade with learned models, GPU computing is essential for:

1. **Training neuromorphic spike encoders** — temporal convolutional models over 1,000+ frame sequences, batch size 32–64, mixed precision (FP16/FP32)
2. **Fine-tuning crack perception models** — CNN-based models on crack datasets (100+ GB)
3. **YOLOv8-nano training** — Object detection model for semantic crack labelling and severity classification
4. **Large-scale simulation sweeps** — 100+ parameter combinations to validate robustness under varying conditions

Estimated GPU requirement: ~3,600 GPU-hours over 6 months (2 GPU cards).

## 2. Track Record of the Team

### Technical Achievements

| Achievement               | Evidence  |
|---------------------------|---|
| Complete SFSVC C++ engine | 20+ source files, 5,000+ lines of production C++                  |
| AVX2 SIMD optimisation    | <code>compute_delta_spikes_avx2()</code> — 32 pixels per register |

| ACHIEVEMENT                       | EVIDENCE   |
|-----------------------------------|--|
| Multi-rate four-lane architecture | Biologically-inspired, from hard-RT control to YOLO semantics                              |
| Lock-free concurrent pipeline     | <code>lockfree_queue.h</code> — wait-free multi-producer/single-consumer                   |
| Python SDK via pybind11           | <code>rt_core_pybind.cpp</code> — seamless ML integration                                  |
| Failsafe subsystem                | <code>failsafe.cpp</code> + <code>degraded_mode_policy.cpp</code>                          |
| Provisional patent filed          | Spike-based neuromorphic video encoding method   |
| Professional website              | <a href="http://www.aurasensehk.com">www.aurasensehk.com</a> — live product marketing site |
| Comprehensive documentation       | Datasheet, deployment checklist, pilot onboarding guide                                    |

## Codebase Summary (GitHub: AuraSense-SFSVC)

Source Code: 20+ C++/Python files  
 Build System: CMake with AVX2 auto-detection  
 Testing: Automated benchmarks + validation suite  
 Documentation: Technical datasheet, benchmark reports, pilot guides  
 CI/CD: GitHub Actions (planned)

## Key Publications & Recognition

- Research papers in preparation — targeting publication in IEEE/ACM conferences on neuromorphic computing and edge AI as part of this project

## Open Source Engagement

- AuraSense SFSVC repository hosted on GitHub
- Technical documentation and benchmark data publicly available
- Demonstrates transparency and reproducibility of results

---

AuraSense Limited — Demonstrating efficient, world-class R&D output with minimal resources