Quantum Circuit Vision

Cost-Aware Evaluation of Visual AI Agents for Quantum Code Generation

Dongping Liu[†,1] **Aoyu Zhang**[†,1] **Luyao Zhang**[†,2,*]

[1]Amazon Web Services [2]Duke Kunshan University [*]Corresponding to: lz183@duke.edu

[†]*Authors are listed in alphabetical order (last name, then first name).*

***Acknowledgments**: We thank the anonymous reviewers of the KDD Workshop on Evaluation and Trustworthiness of Agentic AI.*

KDD 2026 August 9–13, 2026 Jeju, Korea

QCV · Agentic AI · Cost-aware

Motivation & Research Questions (RQs)

Quantum computing is scaling, yet every circuit diagram in papers, textbooks, and courses must still be translated into code by hand — a bottleneck slowing scientific progress.

We ask whether multimodal agents can **read circuit images** and output **executable, unitary-verified** code — and at what cost.

- **RQ 1**: Can agents translate circuit **images** into correct SDK code?
- **RQ 2**: What is the **cost–accuracy** trade-off across model tiers?
- **RQ 3**: Does analyze-then-code (TV / CoT) improve visual understanding?

Datasets

QCV fills a gap left by *circuit-only* corpora (e.g., NTangled, QDataSet, MNISQ) and *non-quantum* vision→code suites (e.g., Vis.→Code): it pairs **multimodal diagrams** with **executable ground truth** and **objective verification** (unitary fidelity $F \geq 0.99$). In the taxonomy, QCV stands out on **dataset infra.** (governance + versioning), **cost/deployment** (latency + billing logs), and **trust/monitoring** (annotated failure taxonomy).

Circuit Categories (13)

Group	Examples
Basic	Hadamard, Bell, Toffoli
Intermediate	QFT, Grover, teleportation
Advanced	QAOA, VQE, Bernstein–Vazirani
Blockchain	BB84, VRF, consensus-style protocols
A: Gate coverage	CRy, CCZ, iSWAP (broad gate families)
B: Qubit scaling	GHZ-10, QFT-5, ring entanglers (4–10 qubits)
C: Classical	Deutsch–Jozsa, Simon, Shor
D: Variational	hardware-efficient, UCCSD-style ans”atze
E: Error correction	Shor-9, Steane-7, surface-code patch
F: Quantum ML	QCNN, QGAN, kernel methods
G: Blockchain extensions	E91, voting, auction protocols
H: Visual variants	barrier/decomposed/reversed (layout robustness)
I: BTC security	Shor vs ECDSA, Kyber-like PQC defense

Methodology

Two visual modes. BV (image→code) and TV (analyze-then-code; CoT).

Objective verification (no LLM-as-judge):

$$F = \frac{|\mathrm{Tr}(U_{\mathrm{gt}}^{\dagger} U_{\mathrm{gen}})|}{2^n} \geq 0.99$$

Three levels: **syntax** → **execution** → **unitary fidelity**. Unitary comparison is SDK-agnostic in principle (e.g., Qiskit Aer, Cirq, PennyLane).

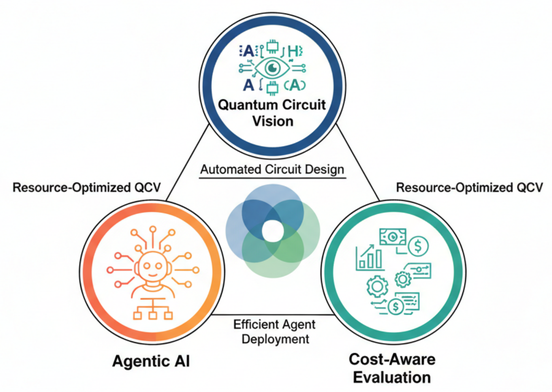
Verification Pipeline

Circuit diagram (PNG)
↓
MMLLM agent: BV / TV mode
↓
Generated SDK code (Braket / Qiskit)
↓
✓ **1.** Syntax check (compile)
↓
✓ **2.** Execution check (runs)
↓
✓✓ **3.** Unitary fidelity $F \geq 0.99$

Objective pipeline: no human review and no LLM-as-judge.

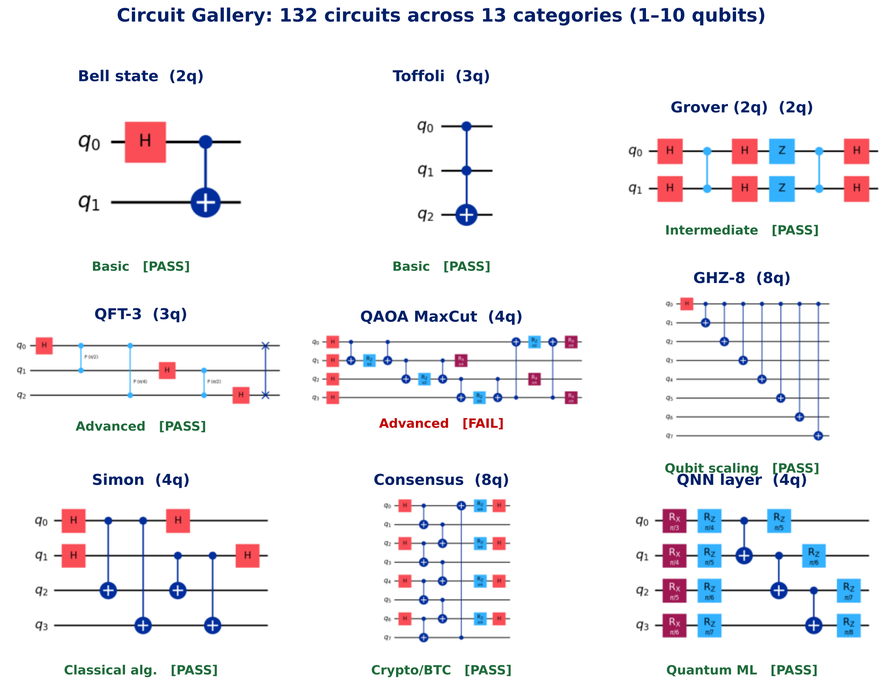
Key Contributions

- **Open Assets — QCV-Datasets**: 132 circuits, 13 categories, 5 modalities. Released on Hugging Face & GitHub with Datasheet, `CITATION.cff`, Croissant-RAI, DOI.
- **Cost-Aware Framework**: model-agnostic & SDK-agnostic evaluation; cascade routing cuts cost by 62% (84% pass @ 38% cost).
- **Agentic AI for Quantum Computing Evaluation**: $n=5$ trials, paired tests, TOST on core subset; full-set results ($n=1$) are indicative; failure analysis + transparent limitations guide future research.



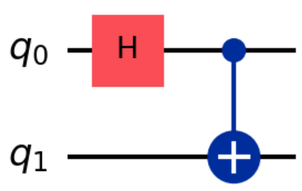
Interidisiplionly Research Framework

Circuit Gallery (132 circuits, 1–10 qubits)



Regular structure can pass even at 8 qubits (**GHZ-8, Consensus PASS**), while a smaller instance (**QAOA, 4q**) can fail — *structure, not size, determines success*.

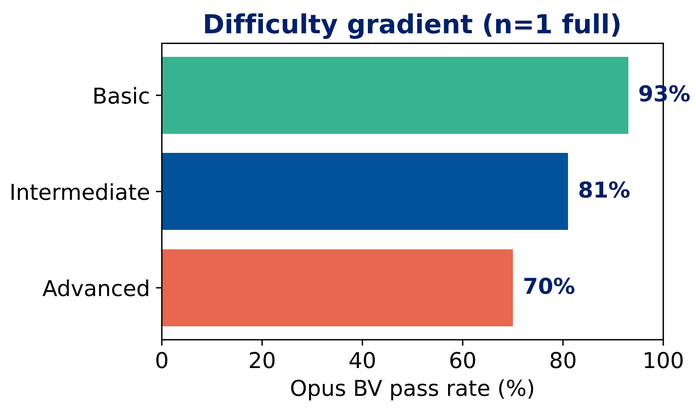
Example: Image → Verified Code




```
from braket.circuits
  import Circuit
circuit = Circuit()
circuit.h(0)
circuit.cnot(0, 1)
```

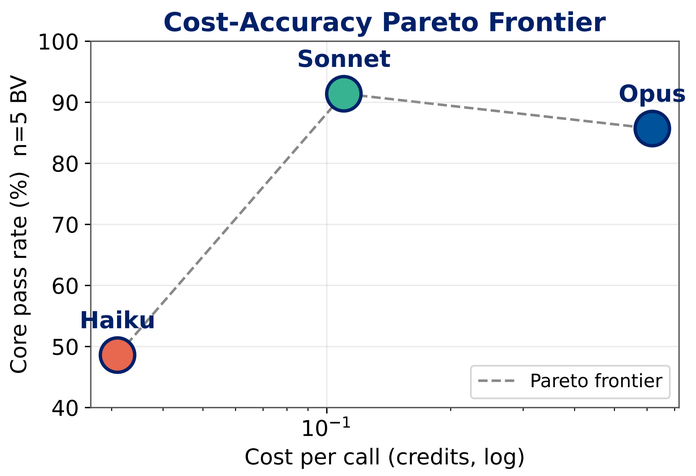
Input (Bell)
Verified by **unitary fidelity** ($F \geq 0.99$) as an objective oracle.

Difficulty Gradient



A clear gradient: Basic **93%** → Intermediate **81%** → Advanced **70%** (n=1, Opus BV, full set). Hard algorithms and cryptographic protocols remain challenging, while basic gates are near-solved.

Cost–Accuracy Frontier



All three models lie on the **Pareto frontier**: Haiku minimizes cost, Opus maximizes accuracy, and **Sonnet** is the balance point (∼97% of Opus accuracy at ∼18% of the cost). The Sonnet–Opus gap is **not significant** (paired t: p=0.083; Wilcoxon: p=0.057) — “optimal” depends on preference.

BV vs TV: A Rescue Example

TV (analyze-then-code) can catch SDK-specific pitfalls that BV(image→code) misses. Example: a **controlled-R_y** gate.

BV — Fail

```
circuit = Circuit()
circuit.h(0)
circuit.cry(0, 1,
  angle=1.57)
```

`.cry()` does not exist in the Braket SDK ⇒ runtime error

TV — Pass

```
# analysis: controlled-Ry
# = decompose to CNOT + Ry
circuit.ry(1, 0.785)
circuit.cnot(0, 1)
circuit.ry(1, -0.785)
circuit.cnot(0, 1)
```

analysis → valid decomposition, F=1.0

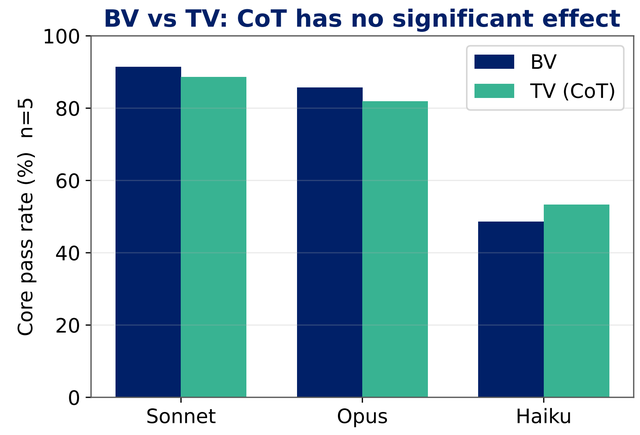
Analysis yields a valid **decomposition** into Braket primitives. Aggregate CoT impact remains **not significant** ($p > 0.18$; TOST: BV ≈ TV).

Results & Findings

■ Accuracy ■ Cost ■ Deployment & Failure

CoT has no significant effect

Overall, analyze-then-code (TV / CoT) does not significantly improve accuracy vs BV. Equivalence testing (TOST, ±10pp) supports BV ≈ TV (see table + $p > 0.18$ overall).



Circuit **depth** (p<0.001) — not qubit count (p=0.20) — predicts failure (LR, 13.2% dev., strongest tested).

Full Benchmark (pass rate %)

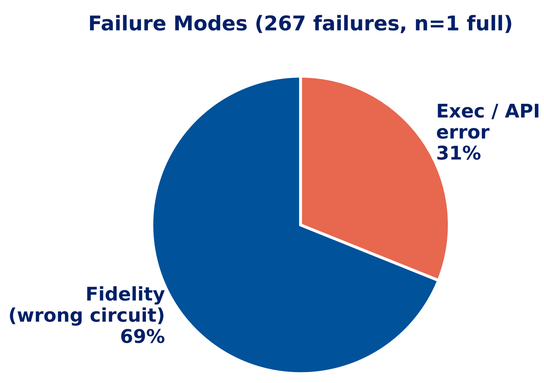
	Full (n=1)		Core (n=5)				
Model	BV	TV	BV	±σ	TV	±σ	**Note**
Opus	79	77	85.7	5.8	81.9	7.8	priciest
Sonnet	77	75	**91.4**	5.2	88.6	4.3	sensible default
Haiku	43	47	48.6	4.0	53.3	9.2	cheapest

Cost & CoT Equivalence (BV mode)

Model	cr/call	cr/correct	Δ(TV−BV)	TOST p
Haiku	0.031	0.072	+4.8	0.33 (n.s.)
Sonnet	0.110	0.142	−2.9	0.001 ✓
Opus	0.618	0.778	−3.8	0.037 ✓

Opus & Sonnet: TV is *statistically equivalent* to BV (±10pp). Sonnet is the cost-efficiency sweet spot.

Failure Analysis



45/132 are solved by all 6 configs; **18** defeat all (Shor, HHL, QAOA, surface code, crypto). Failures are mainly **semantic** (wrong unitary), not syntax.

Cascade Routing

Haiku→Sonnet→Opus yields **84% pass at 38% of single-model cost (n=1, indicative)**. **Routing** > **prompting** as the main cost lever.

Conclusion & Resources

We introduce a **cost-aware, unitary-verified** benchmark for visual AI agents on quantum code. **Sonnet** is the cost-efficiency sweet spot on the frontier; **routing** beats prompting (84% pass @ 38% cost); and circuit **depth** predicts failure. To handle volatile pricing, we report **tokens** and **price** layers separately. Data, code, and cost logs are released.

GitHub · HF Dataset

BV = Basic Vision (direct image→code); TV = Thinking Vision (chain-of-thought analyze-then-code); CoT = Chain of Thought; EDA = Electronic Design Automation; MLLM = Multimodal Large Language Model.

When AI reads Quantum?