Design of an Offline Vision–Language Model for PCB Quality Inspection

1 Introduction

Printed Circuit Board (PCB) inspection in semiconductor manufacturing demands high accuracy, low latency, and strong reliability. Traditional computer vision pipelines rely on rigid rule-based or supervised object detection systems, which lack flexibility when inspectors need to ask ad-hoc questions about defects.

This document proposes a custom **Vision–Language Model (VLM)** design that enables inspectors to query PCB images in natural language while maintaining strict control over hallucinations, inference latency, and deployment constraints. The system is designed to operate fully offline and deliver responses in under two seconds per image.

The design is based on existing research literature, open-source model documentation, and logical system-level reasoning. No proprietary models or cloud-based services are assumed. :contentReference[oaicite:0]index=0

2 Problem Statement

The target system must satisfy the following constraints:

- Fully offline operation (no internet connectivity)
- Natural language query interface for inspectors
- Structured outputs including:
  - Defect locations (bounding boxes)
  - Defect types
  - Confidence scores
- Inference latency < 2 seconds per image
- Training data consists of:
  - 50,000 PCB images
  - Bounding box annotations
  - No question–answer pairs
- Generic VLMs exhibit unacceptable hallucination rates in this domain

3 Model Selection

3.1 Candidate Models

Model	Size	Speed	License	Notes
LLaVA	7B–13B	Medium	Open	Popular, well-documented
BLIP-2	3B–11B	Fast	Open	Older architecture
Qwen-VL	7B	Good	Open	Strong localization support
Custom VLM	Varies	Unknown	–	High development risk

3.2 Chosen Model: Qwen-VL (7B)

Qwen-VL (7B) is selected based on the following criteria:

1. Permissive open-source license suitable for commercial deployment
2. Native support for spatial reasoning and coordinate outputs
3. Manageable model size for industrial GPU deployment
4. Well-supported LoRA fine-tuning pipeline
5. More recent architecture compared to alternatives

LLaVA was considered as a secondary option, but Qwen-VL demonstrates better out-of-the-box grounding performance.

4 System Architecture

4.1 High-Level Pipeline

```
Image → Vision Encoder → Visual Adapter → Cross-Attention Fusion
      → Language Model → Structured Outputs
```

4.2 Component Breakdown

4.2.1 Vision Encoder

- Pretrained ViT or CLIP-based encoder from Qwen-VL
- Majority of layers frozen to reduce training cost
- Optional lightweight convolutional layers for small defect sensitivity

4.2.2 Visual Adapter

- Projects visual features into language embedding space
- Primary location for PCB-specific specialization
- Focuses on:
  - Micro-scratches
  - Solder bridges
  - Missing pads or components
  - Subtle color variations

4.2.3 Fusion Mechanism

Cross-attention is used to condition visual features on the query text, enabling spatially-aware reasoning such as region-specific defect queries.

4.2.4 Language Model

- Qwen 7B base model
- Minimal architectural changes
- Constrained output format enforced during decoding

4.2.5 Output Heads

The model produces structured outputs:

```
{
  "answer": "Two scratches detected",
  "defects": [
    {"type": "scratch", "bbox": [120,45,180,55], "confidence": 0.87},
    {"type": "scratch", "bbox": [300,200,350,210], "confidence": 0.92}
  ],
  "total_count": 2
}
```

5 Inference Optimization

Meeting the < 2 second latency constraint requires aggressive optimization.

5.1 Techniques

- **INT8 / INT4 Quantization** using GPTQ or `llama.cpp`
- **LoRA Fine-tuning** to reduce trainable parameters
- **Model Pruning** (20–30% where safe)
- **Knowledge Distillation** to a smaller ($\sim$2B) student model
- **TensorRT / ONNX Runtime** for deployment

5.2 Deployment Pipeline

```
Qwen-VL 7B
 → LoRA Fine-tuning
 → INT4 Quantization
 → TensorRT Conversion
 → Industrial GPU Deployment
```

6 Hallucination Mitigation

False positives are unacceptable in PCB inspection.

6.1 Mitigation Strategies

1. Domain-only fine-tuning (no generic images)
2. Explicit confidence calibration with temperature scaling
3. Training on "I don't know" responses
4. Fixed defect vocabulary
5. Mandatory bounding box grounding
6. Dual-pass verification
7. Confidence-based rejection thresholds

6.2 Training Loss

$$\mathcal{L} = \mathcal{L}_{text} + \mathcal{L}_{bbox} + 0.5\,\mathcal{L}_{conf} + 2.0\,\mathcal{L}_{halluc}$$

7 Training Strategy

7.1 Automatic QA Generation

QA pairs are synthesized from bounding box annotations:

- Counting questions
- Localization questions
- Yes/No defect presence
- Region-specific queries

Target scale: 10–20 QA pairs per image.

7.2 Multi-Stage Training

Phase	Epochs	Learning Rate	Purpose
Vision Warmup	5	10^{-4}	PCB adaptation
Main Training	3–5	10^{-4}	QA learning
Hallucination Reduction	1–2	10^{-5}	False positive control
Calibration	1	10^{-6}	Confidence tuning

8 Validation

8.1 Metrics

- Counting accuracy and MAE
- Bounding box IoU and mAP
- Hallucination rate
- Inference latency
- Human qualitative evaluation

8.2 Acceptance Criteria

Metric	Target	Minimum
Counting Accuracy	$> 85\%$	$> 75\%$
IoU@0.5	$> 80\%$	$> 70\%$
Hallucination Rate	$< 5\%$	$< 10\%$
Inference Time	< 1.5s	< 2s

9 Conclusion

This design outlines a practical, deployable offline VLM for PCB inspection that prioritizes reliability over generative freedom. The primary technical risks lie in achieving latency targets with INT4 quantization and ensuring QA generation quality, both of which can be mitigated through iterative experimentation.

Author's Note

The design presented in this document is based on literature review, analysis of existing open-source Vision–Language Models, and system-level reasoning.

I have not worked directly on implementing a full Vision–Language Model prior to this work. The architecture, training strategy, and optimization choices described here are derived from publicly available research papers, model documentation, and engineering trade-off analysis.

This document represents my best attempt at proposing a practical and deployable solution under the given constraints. Certain design decisions, hyperparameters, and performance assumptions would require empirical validation through implementation and experimentation.