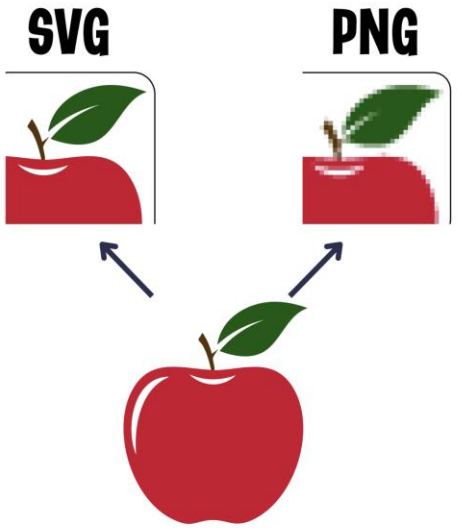
Week 6
Structured Vector Graphics Editing With VLMs

Matthew Perlman

James Beetham, Dr. Amrit Bedi, Dr. Mubarak Shah

Background

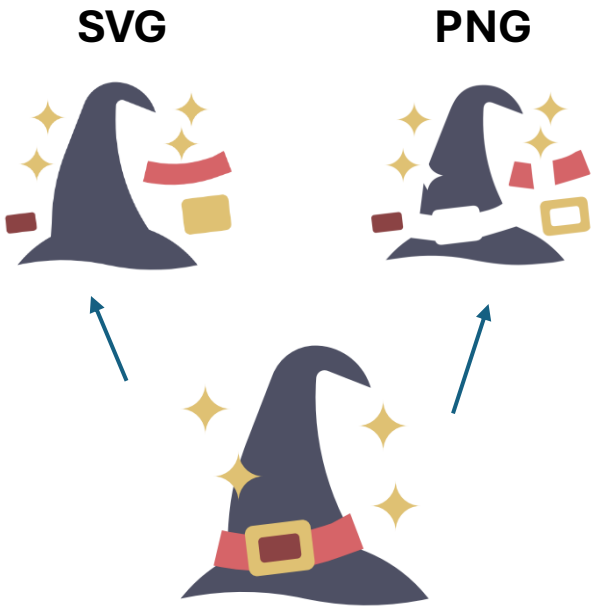
Why do we want to use SVG images instead of raster images?



Lossless Upscaling



Reduced Storage



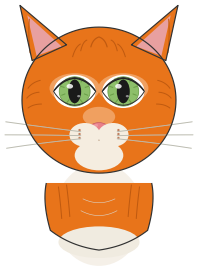
Human Editable

Existing Work

Combining the expressive power of frontier VLMs with diffusion models has "solved" generating rasters. For SVGs… not so much.



GPT-5.5



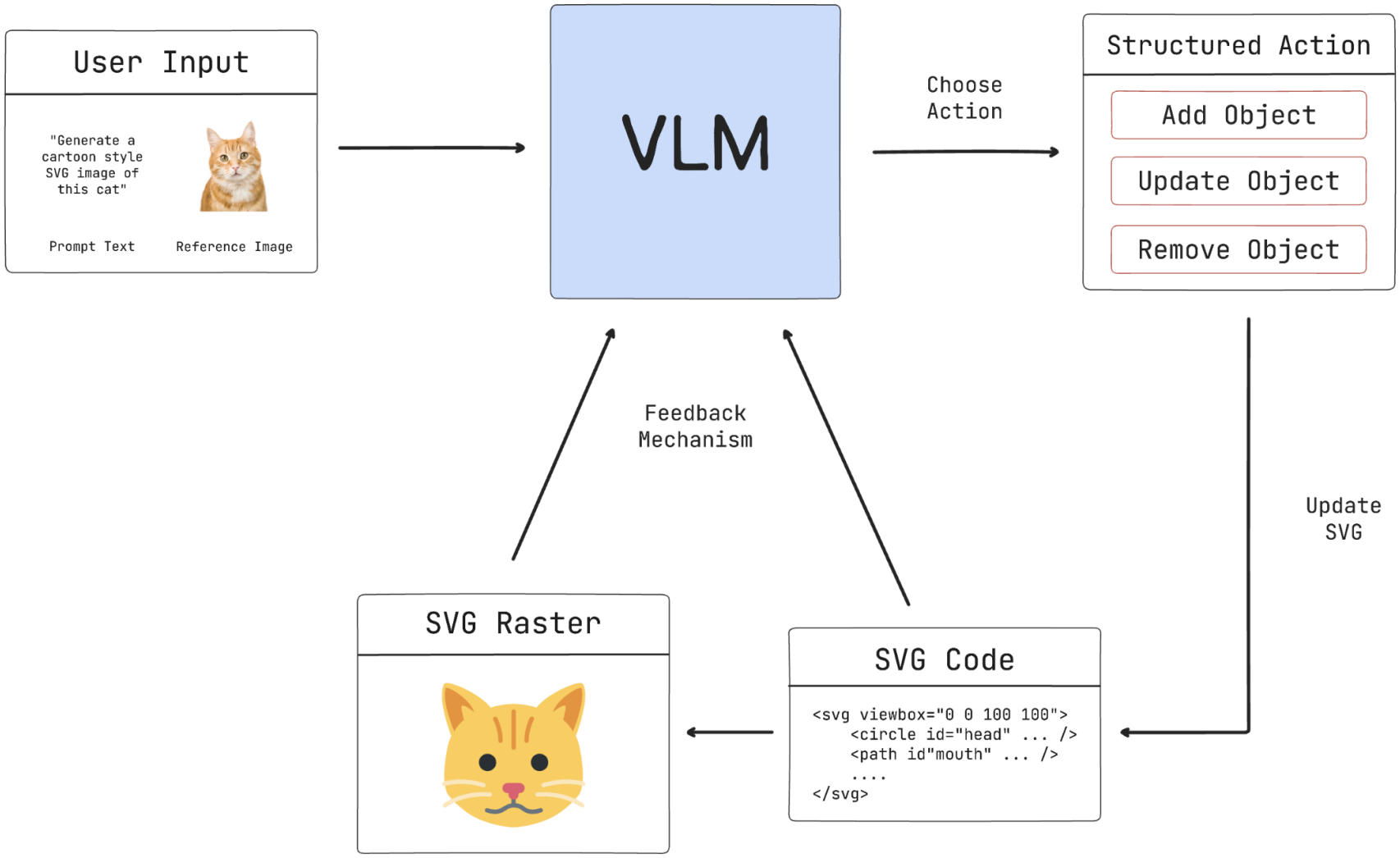
Sonnet-4.6

Dedicated raster-to-vector models perform better, but are **limited** to mostly icons and logos, often produce **overly complex** code, and require **separate training** pipelines.

Why Off-the-Shelf VLMs Fail

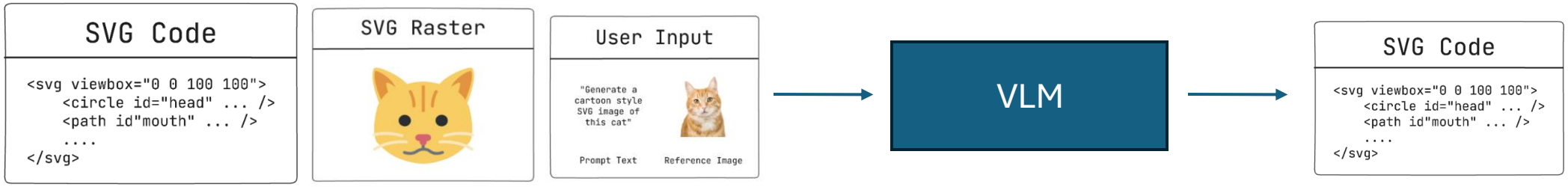
- Smaller models **lack reasoning skill** needed to produce full SVG code
- VLMs **never see** the result, they guess how code looks
- They produce code autoregressively, making choices **irreversible**

Our Solution – Iterative Refinement

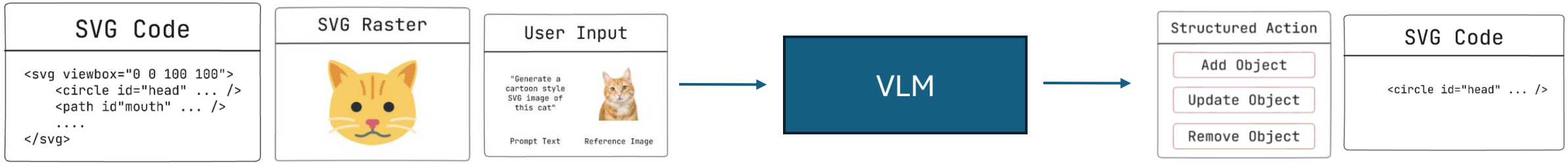


Pipeline Variants

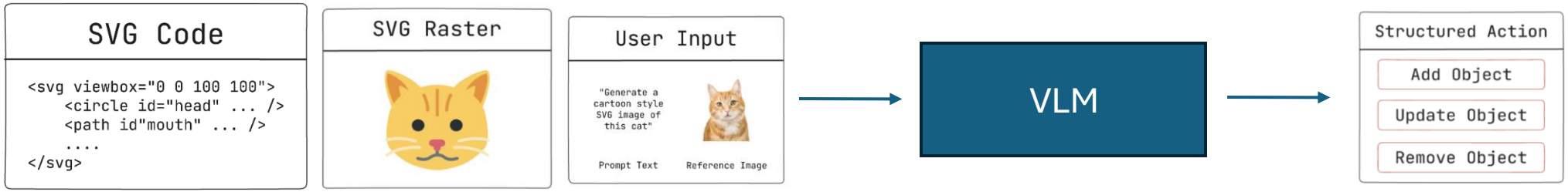
Unstructured Actions



Semi-Structured Actions



Structured Actions



Structured Outputs + Constrained Decoding

- Models natively produce text, but we want to **enforce structure**:
  - Enumerated Options (ex: "True" or "False", nothing else allowed)
  - JSON Template (ex: { name: string, age: number })
  - Context-Free-Grammar (ex: code syntax, XML, nested data)
- Modern language models support structured outputs implemented through a constrained decoding engine

Progress This Week

- Enforcing **structured outputs** for better compilation rates
  - Created JSON Schemas
  - Created EBNF Context-Free Grammar for SVG syntax
  - Extended OpenAI API to leverage schemas
  - Extended local VLLM models to leverage CFGs
- **Improved evaluation** scores
  - Results on Qwen3.5 with structured decoding
  - Still running experiments on other settings

Example EBNF Context-Free Grammar

```
svg ::= "<svg width=\\\"500\\\" height=\\\"500\\\" xmlns=\\\"http://www.w3.org/2000/svg\\

element ::= rect | circle | ellipse | line | polyline | polygon | path | text-elem

rect ::= "<rect" svgws x-attr svgws y-attr svgws width-attr svgws height-attr ( svgws fi

circle ::= "<circle" svgws cx-attr svgws cy-attr svgws r-attr ( svgws fill-attr )? ( svgw

ellipse ::= "<ellipse" svgws cx-attr svgws cy-attr svgws rx-attr svgws ry-attr ( svgws f

line ::= "<line" svgws x1-attr svgws y1-attr svgws x2-attr svgws y2-attr ( svgws stroke-

polyline ::= "<polyline" svgws points-attr ( svgws fill-attr )? ( svgws stroke-attr )? (

polygon ::= "<polygon" svgws points-attr ( svgws fill-attr )? ( svgws stroke-attr )? ( s

path ::= "<path" svgws d-attr ( svgws fill-attr )? ( svgws fill-rule-attr )? ( svgws str

text-elem ::= "<text" svgws x-attr svgws y-attr ( svgws font-size-attr )? ( svgws font-f
```

Image-to-SVG Results: Model × Pipeline

Model	Generation Method	Compilation Rate	CLIP Alignment
(Last Week) Unconstrained Qwen2.5-3B	Single-Turn	0.18	0.216
	Structured	**0.25**	0.188
	Semi-Structured	0.01	0.168
	Unstructured	0.03	**0.221**
Constrained Qwen3.5-2B	Single-Turn	0.27	0.569
	Structured	0.15	0.548
	Semi-Structured	**0.30**	0.468
	Unstructured	0.07	**0.628**
Constrained Qwen3.5-9B	Single-Turn	**1.00**	0.248
	Structured	0.53	**0.657**
	Semi-Structured	0.26	0.502
	Unstructured	0.44	0.661

Next Steps

- Modify schemas/grammars to achieve **100% compilation** rate
- **Reduce generation time** when using structured outputs
- Collect **qualitative examples** and **finalized evaluation** results
- Continue drafting **paper**
  - Pacific Graphics poster due July 15
  - SIGGRAPH Asia due July 27