AutoSketch: VLM-Assisted Style-Aware Vector Sketch Completion

Hsiao-Yuan Chin[1*], I-Chao Shen[2*†], Yi-Ting Chiu[1], Ariel Shamir[3], Bing-Yu Chen[1†]

[1]National Taiwan University, Taiwan [2]The University of Tokyo, Japan
[3]Reichman University, Israel
([*]: Equal contribution. [†]: Joint corresponding author.)

SIGGRAPH ASIA 香港
2025 HONG KONG 15-18 DEC

Motivation

- Sketch has long been the key form of visual expression.
- Even people with little experience can easily sketch simple objects and ideas.

Motivation

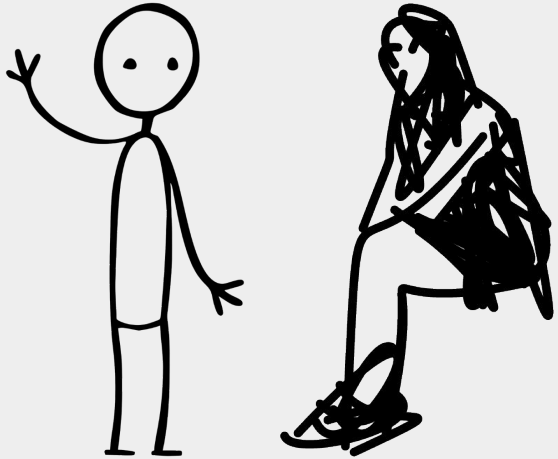
- When creating **complex scenes**, individuals often start with a rough or given partial sketch.

Partial sketch

I want that woman and a man chatting in the park...

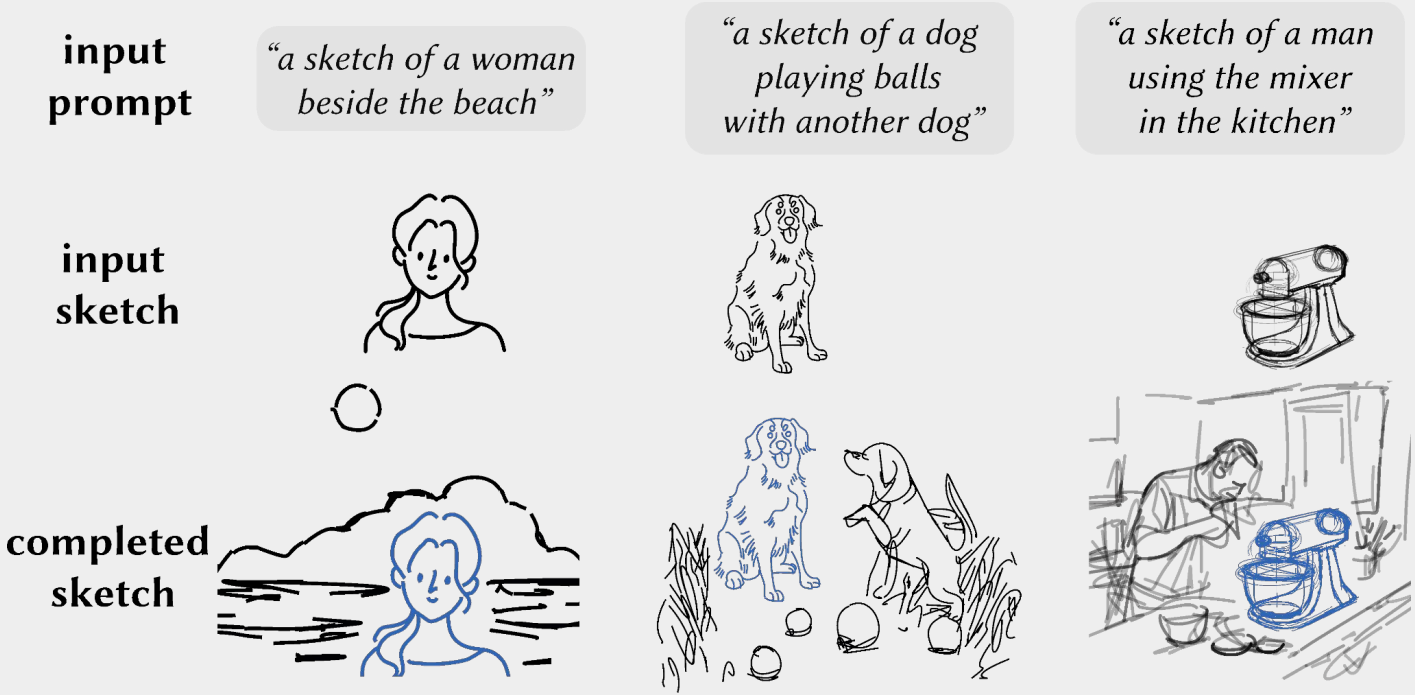
Motivation

- However, many struggle to complete it into a **final complex sketch** that maintains a **unique and consistent style**



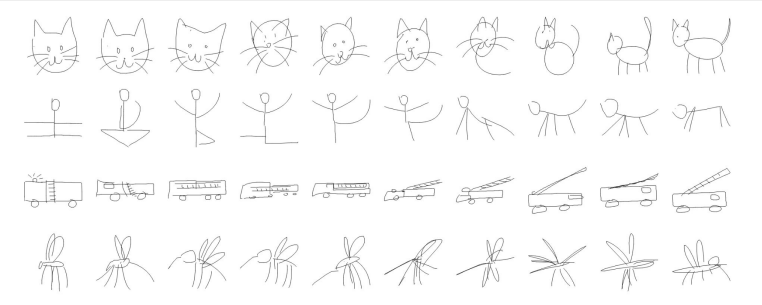
AutoSketch

- a novel style-aware vector sketch completion method that takes a text prompt and a partial sketch as input.

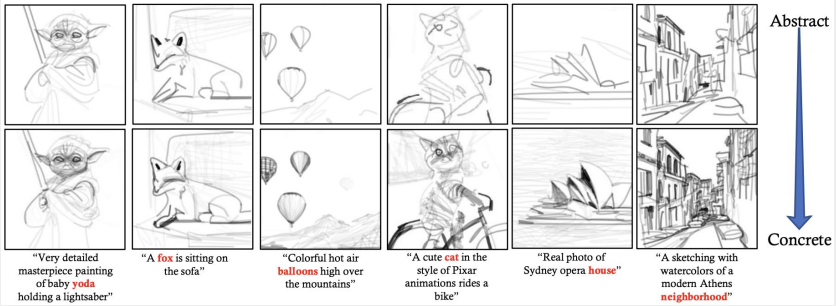


Limitation of Previous Works

- Existing studies used sketch datasets and various deep learning models
  - However, given **their reliance on these sketch datasets**, such methods generally generate sketches of only **simple objects**.
- **Synthesis-through-optimization** paradigm
  - These methods mainly focus on generating sketches that match the prompt content in **a predefined style,** while **neglecting the style of the input sketches.**



[Ha and Eck 2018]



DIffSketcher [Xing et al. 2023]

Method Overview

Input prompt

A sketch of a woman and a man chatting in the park

Input sketch

Stage 1: Content-centric sketch completion

Stage 2: VLM-based sketch style adjustment

Completed sketch

Method Overview

Input prompt

A sketch of a woman and a man chatting in the park

(a) Prompt augmentation

VLM

Augmented prompt

A sketch of a woman and a man chatting in the park, **sketchy, expressive, raw, gestural**

Input sketch

(b) Stroke optimization for completion

ControlNet

Image Guidance

(c) Style difference detection and code generation

Intermediate sketch

VLM

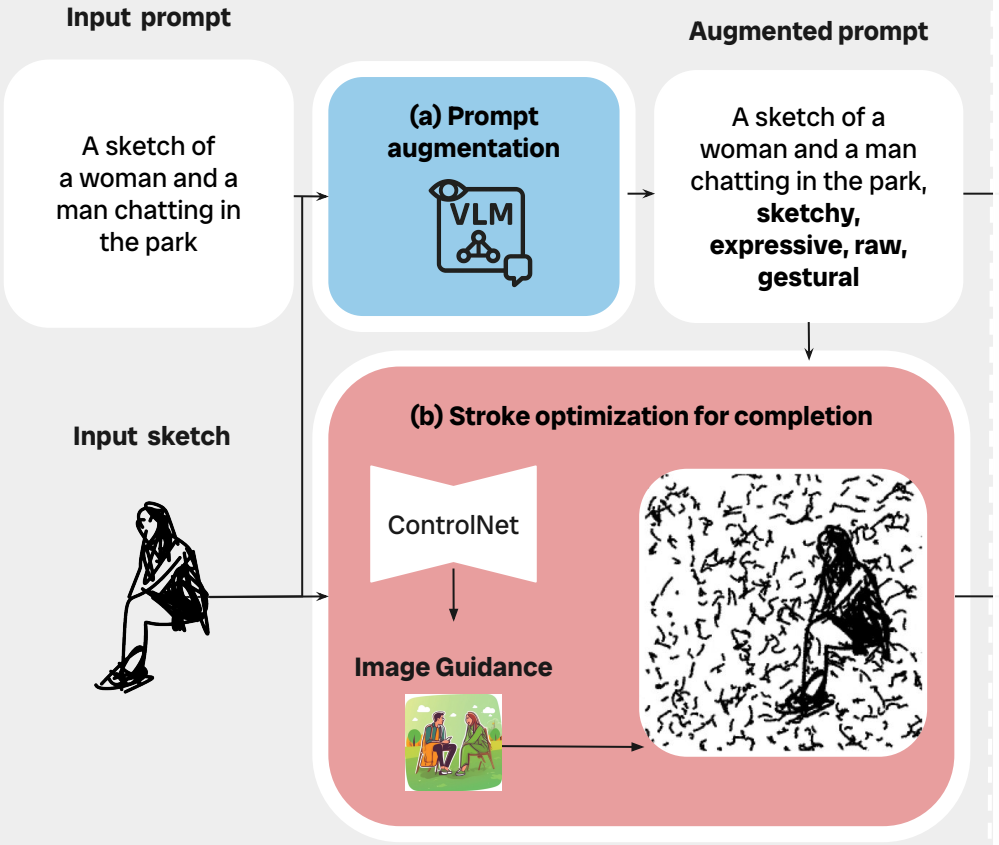
```
def adjust_stroke_style(path_data):
    parsed_path = parse_path(path_data)
    enhanced_segments = []
    for seg in parsed_path:
        if isinstance(seg, CubicBezier):
            # Simplify curves by splitting them
            midpoint = seg.point(0.5)
            enhanced_segments.append(Line(start=seg.start,\\
            end=midpoint))
            enhanced_segments.append(Line(start=midpoint,\\
            end=seg.end))
    # Smooth paths by reducing unnecessary points
    smoothed_segs = []
    for i in range(len(enhanced_segments) - 1):
        start = enhanced_segments[i].start
        end = enhanced_segments[i + 1].end
        smoothed_segs.append(Line(start=start, end=end))
    # Combine segments into an SVG path string
    enhanced_path = " ".join(seg.d() for seg in smoothed_segs)
    return enhanced_path
```

Completed sketch

Stage 1: Content-centric sketch completion

Stage 2: VLM-based sketch style adjustment

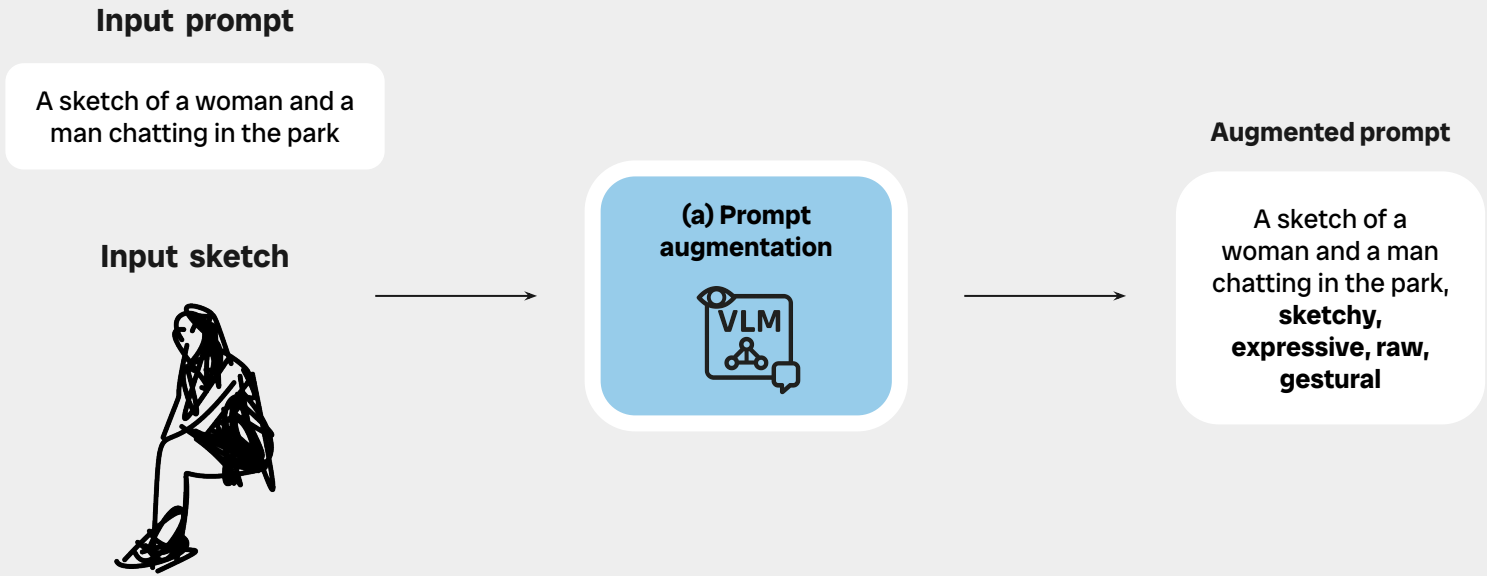
Method Overview



Stage 1: Content-centric sketch completion

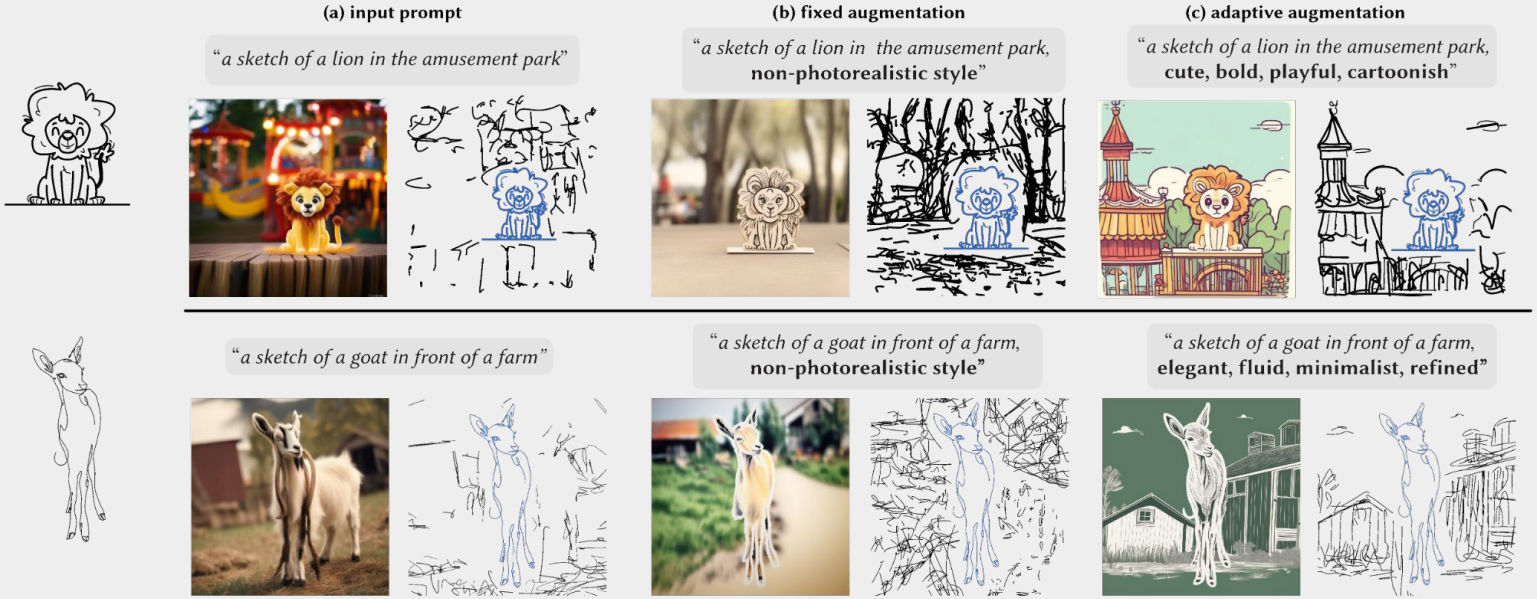
(a) Prompt Augmentation

- We render the input sketch and use a VLM to extract its global abstraction and local style cues.
- These descriptions are appended to the original prompt to form an augmented prompt.



(a) Prompt Augmentation

- We render the input sketch and use a VLM to extract its global abstraction and local style cues.
- These descriptions are appended to the original prompt to form an augmented prompt.



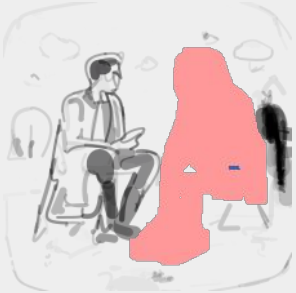
(b) Stroke Optimization for Completion

$$L_{\text{all}} = \alpha\left(1 - \text{sim}\left(\phi_{\text{vis}}(\mathcal{I}_{\text{sketch}}), \phi_{\text{vis}}(\mathcal{I}_{\text{guide}})\right)\right)$$
$$+\beta \text{LPIPS}(\mathcal{I}_{\text{sketch}}, \mathcal{I}_{\text{guide}}) + \gamma \sum_{x_k \in \mathbf{x}} \mathbf{1}\left[\mathbf{M}(x_k) = 1\right]$$

Intermediate sketch

Iteration **0**

Iteration **50**



Iteration **200**

Iteration **1000**

Method Overview

Input prompt

A sketch of a woman and a man chatting in the park

(a) Prompt augmentation

VLM

Augmented prompt

A sketch of a woman and a man chatting in the park, **sketchy, expressive, raw, gestural**

Input sketch

(b) Stroke optimization for completion

ControlNet

Image Guidance

Stage 1: Content-centric sketch completion

(c) Style difference detection and code generation

Intermediate sketch

VLM

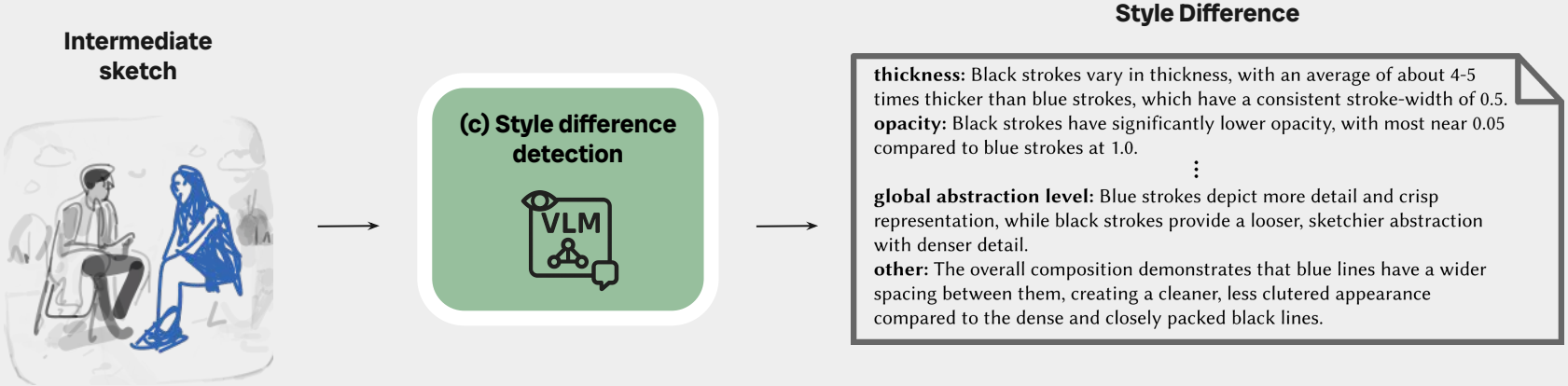
```
def adjust_stroke_style(path_data):
    parsed_path = parse_path(path_data)
    enhanced_segments = []
    for seg in parsed_path:
        if isinstance(seg, CubicBezier):
            # Simplify curves by splitting them
            midpoint = seg.point(0.5)
            enhanced_segments.append(Line(start=seg.start,\\
            end=midpoint))
            enhanced_segments.append(Line(start=midpoint,\\
            end=seg.end))
    # Smooth paths by reducing unnecessary points
    smoothed_segs = []
    for i in range(len(enhanced_segments) - 1):
        start = enhanced_segments[i].start
        end = enhanced_segments[i + 1].end
        smoothed_segs.append(Line(start=start, end=end))
    # Combine segments into an SVG path string
    enhanced_path = " ".join(seg.d() for seg in smoothed_segs)
    return enhanced_path
```

Completed sketch

Stage 2: VLM-based sketch style adjustment

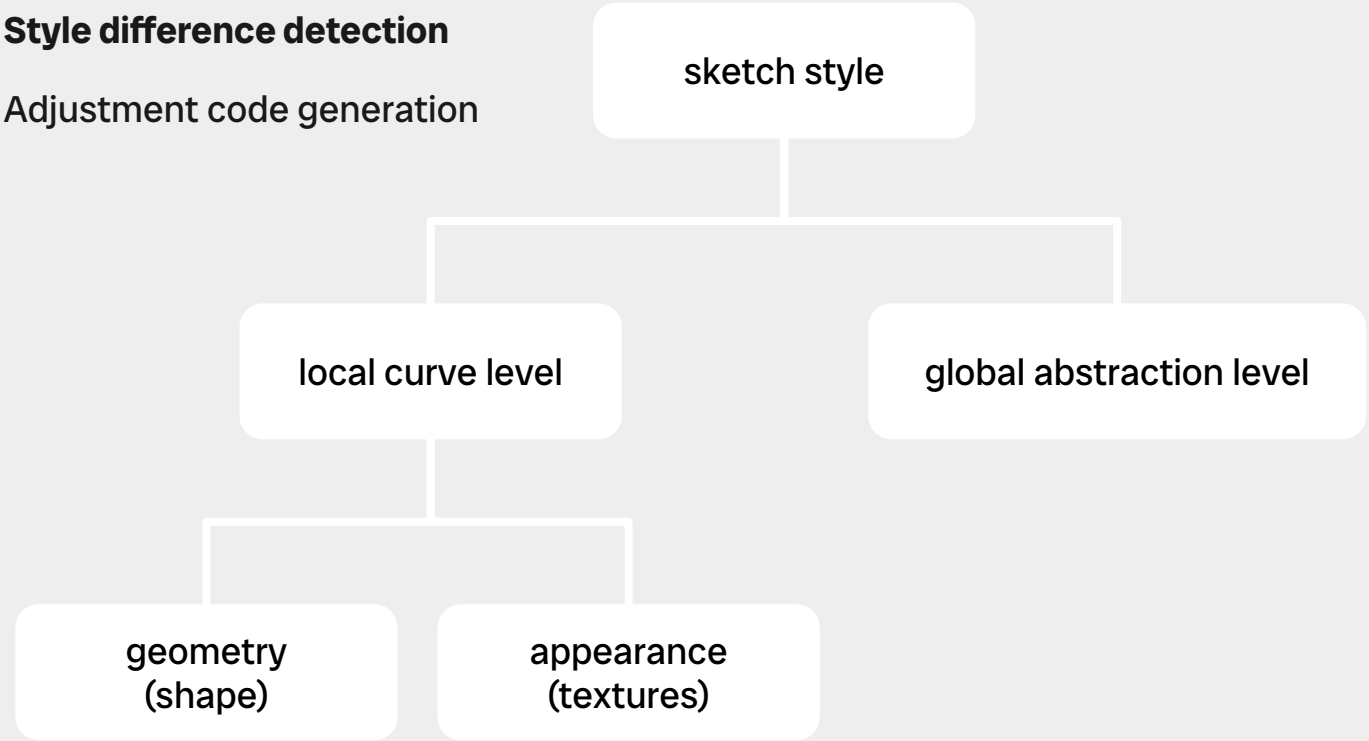
(c) VLM-based Sketch Style Adjustment

- Iterative style differences detection and code generation process
  - **Style difference detection**
  - Adjustment code generation



(c) VLM-based Sketch Style Adjustment

- Iterative style differences detection and code generation process
  - **Style difference detection**
  - Adjustment code generation



(c) VLM-based Sketch Style Adjustment

SIGGRAPH ASIA 2025 香港
HONG KONG 15-18 DEC

- Iterative style differences detection and code generation process
  - Style difference detection
  - **Adjustment code generation**

Intermediate sketch

Augmented prompt

A sketch of a woman and a man chatting in the park, **sketchy, expressive, raw, gestural**

Style Difference

thickness: Black strokes vary in thickness, with an average of about 4-5 times thicker than blue strokes, which have a consistent stroke-width of 0.5.
opacity: Black strokes have significantly lower opacity, with most near 0.05 compared to blue strokes at 1.0.
⋮
global abstraction level: Blue strokes depict more detail and crisp representation, while black strokes provide a looser, sketchier abstraction with denser detail.
other: The overall composition demonstrates that blue lines have a wider spacing between them, creating a cleaner, less clutered appearance compared to the dense and closely packed black lines.

→

(c) Adjustment code generation

VLM

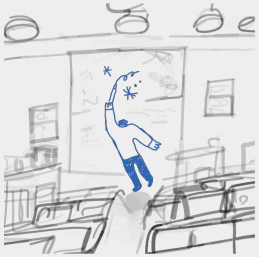
→

```python
def adjust_stroke_style(path_data):
    for elem in parent.findall('.//svg:path', namespace):
        style = elem.attrib
        if 'stroke' in style and style['stroke'] != 'rgb(51, 102, 178)':
            # Get current attributes
            stroke_width = float(style.get('stroke-width', 2.0))
            stroke_opacity = float(style.get('stroke-opacity', 1.0))
        # 1. Remove extreme attributes
        if stroke_opacity < 0.12:
            parent.remove(elem)
            continue
        # 2. Adjust stroke width
        style['stroke-width'] = str(max(0.5, min(stroke_width,
        given_sketch_style['stroke-width'] * 0.9)))
        # 3. Adjust opacity
        style['stroke-opacity'] = str(min(max(stroke_opacity, 0.95), 1.0))
        # 4. Adjust path to increase distance (shift position slightly)
        d = style.get('d', '')
        new_d = re.sub(
            r"([MLC])\s*(-?\d+\.?\d*)\s*(-?\d+\.?\d*)",
            lambda match: f"{match.group(1)}
            {float(match.group(2)) + 2} {float(match.group(3)) + 2}", d)
        style['d'] = new_d
```

→

Completed sketch

Comparison with Existing Methods

(a) input sketch & augmented prompt	(b) SketchDreamer	(d) DiffSketcher	(f) Gemini Co-Drawing	(g) Ours (first stage)	(h) Ours
 "*a sketch of a man in the classroom,* **whimsical, minimal, expressive, dynamic**"					
 "*a sketch of a woman beside the beach* **minimalistic, clean, expressive, modern**"					
 "*a man using the mixer in the kitchen* **minimalist, detailed, precise, clean**"					

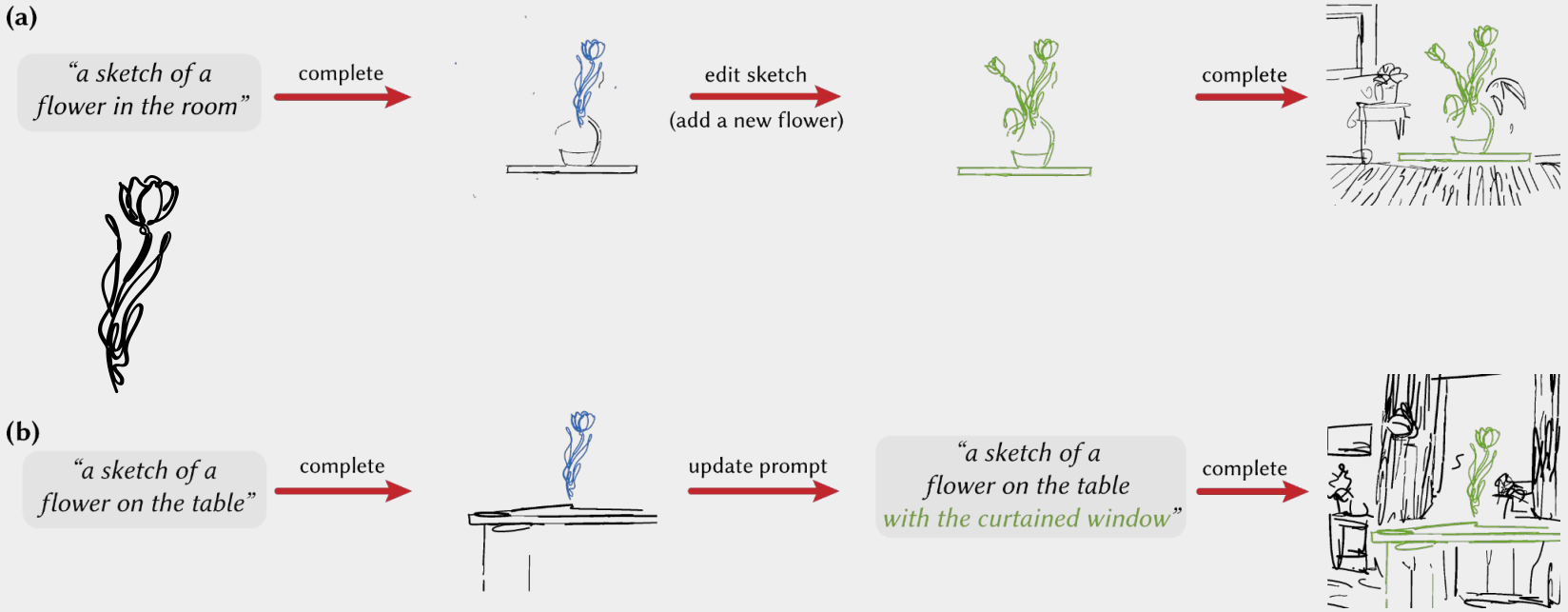
Comparison with Existing Methods

- User evaluation results

	Style			Content		
	Ours	Others	neither	Ours	Others	neither
(a) vs. SketchDreamer	**94.29**	5.36	0.36	**86.43**	3.21	10.36
(b) vs. DiffSketcher	**92.14**	7.50	0.36	**55.71**	40.36	3.93

Diverse Sketch Scenario

- Iterative sketch completion : sketches with different prompts, or distinct sketches



Generalization of VLMs

partial sketch

(a) GPT-4o

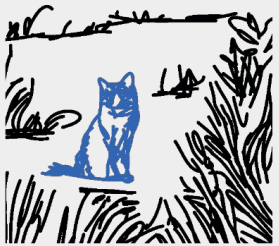
"*a sketch of a cat beside the river in the grass,* **bold, sketchy, expressive, simplistic**"

(b) Gemini

"*a sketch of a cat beside the river in the grass,* **simple, minimalist, expressive, dynamic**"

(c) Qwen3

"*a sketch of a cat beside the river in the grass,* **minimalist, bold, stylized, abstract**"



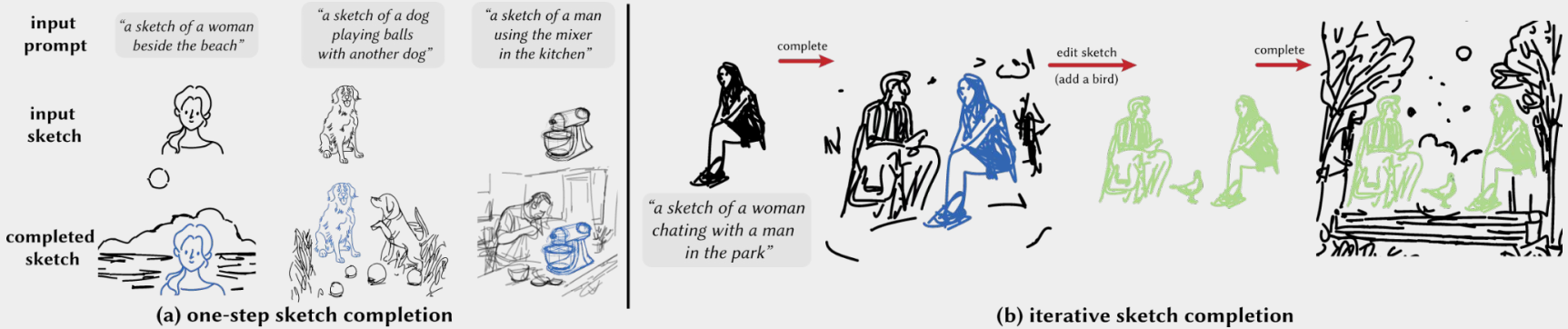
augmented prompt

guidance image

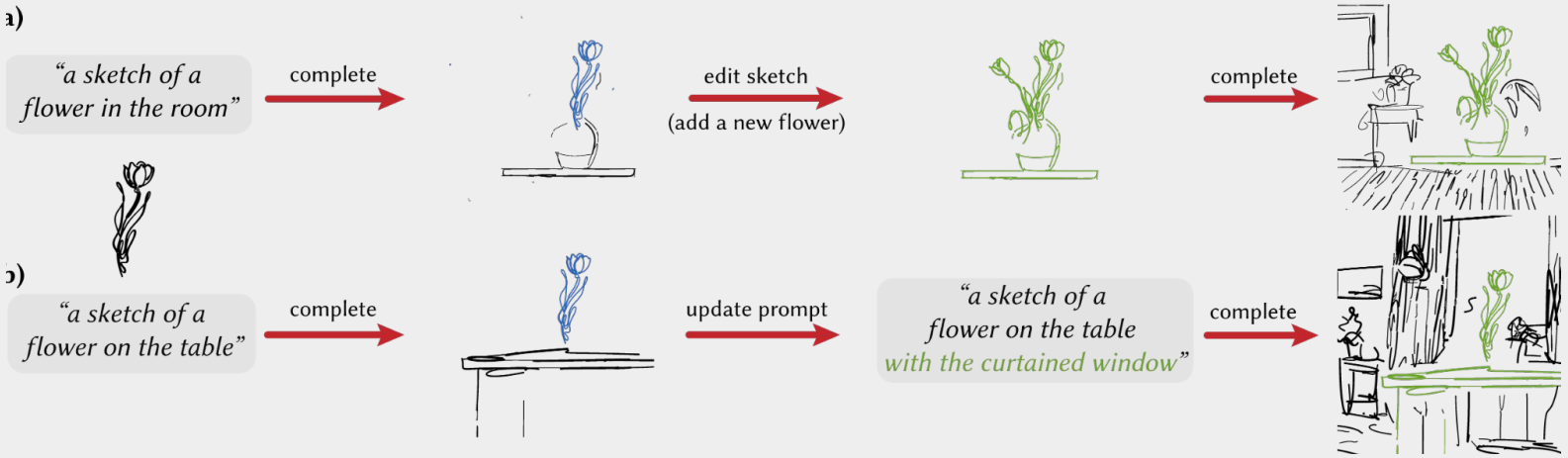
completed sketch

Conclusion

- AutoSketch, a **style-aware vector sketch completion method** that accommodates diverse sketch styles by **leveraging a pretrained VLM.**



Thank You


a)
"a sketch of a flower in the room"
complete
edit sketch
(add a new flower)
complete
b)
"a sketch of a flower on the table"
complete
update prompt
"a sketch of a flower on the table with the curtained window"
complete