



УНИВЕРСИТЕТ ИТМО

Векторная графика и методы её обработки и генерации

Жарский Иван Александрович

Инженер в Лаборатории МЛ КТ ИТМО, техлид в Statanly Technologies

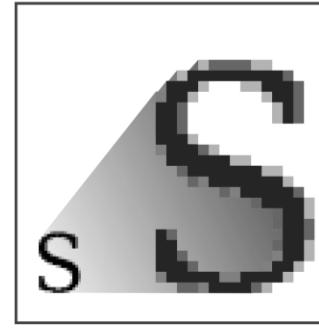
Растровое изображение

Матрица пикселей



Векторное изображение

- Будем рисовать упорядоченную последовательность геометрических фигур на холсте
- Каждая фигура задается своими уравнениями и параметрами
- Каждой фигуре можно указать её цветовые характеристики



РАСТР
.jpeg .gif .png



ВЕКТОР
.svg

$$\text{Окружность: } (x - x_0)^2 + (y - y_0)^2 = R^2$$

Векторное изображение

- Применение векторных изображений: логотипы, иллюстрации, иконки, текстовые шрифты и др.
- Преимущество: масштабирование без потери качества.



Векторное изображение формата SVG

```
1 v <svg width="400" height="400" viewBox="0 0 124 124">
2   <rect width="124" height="124" rx="24" fill="#F97316"/>
3   <path d="M19.375 36.7818V100.625C19.375 102.834 21.1659 104.625 23.375
104.625H87.2181C90.7818 104.625 92.5664 100.316 90.0466 97.7966L26.2034
33.9534C23.6836 31.4336 19.375 33.2182 19.375 36.7818Z" fill="white"/>
4   <circle cx="63.2109" cy="37.5391" r="18.1641" fill="black"/>
5   <rect opacity="0.4" x="81.1328" y="80.7198" width="17.5687"
height="17.3876" rx="4" transform="rotate(-45 81.1328 80.7198)"
fill="#FDBA74"/>
6 </svg>
7
```

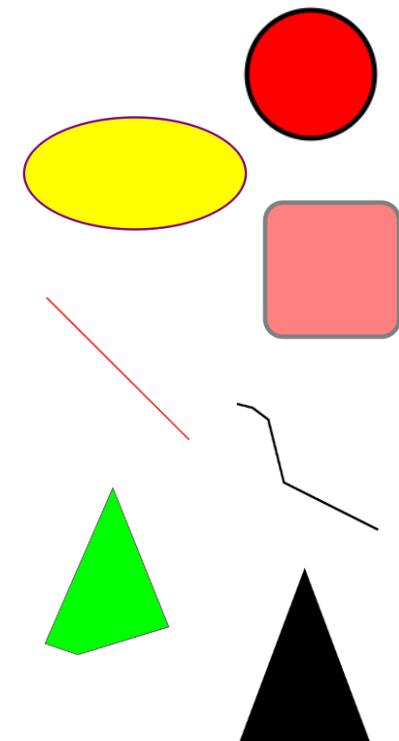
XML-разметка



Холст с фигурами

Базовые SVG фигуры

Tag	Атрибуты
<circle>	cx="50" cy="50" r="40"
<ellipse>	cx="200" cy="80" rx="100" ry="50"
<rect>	x="50" y="20" width="150" height="150" rx="20" ry="20"
<line>	x1="0" y1="0" x2="200" y2="200"
<polyline>	points="20,20 40,25 60,40 80,120 120,140 200,180"
<polygon>	points="220,10 300,210 170,250 123,234"
<path>	d="M150 0 L75 200 L225 200 Z"



Базовые цветовые характеристики

Тип	Атрибут	Значения
Заливка	fill	none / yellow / rgb(255, 255, 0) / #ffff00 / rgba(255, 255, 0, 0.8) / url(#id)
Обводка	stroke	
Жирность обводки	stroke-width	Число
Прозрачность	opacity	Число из [0; 1]

Тег <path>

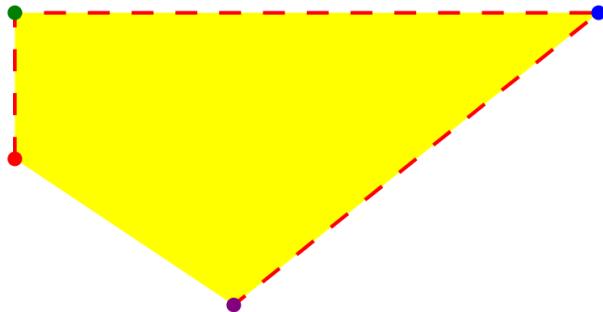
Команда	Имя	Пояснение
MoveTo	M/m	Откуда стартовать: “M 10,20”
ClosePath	Z/z	Соединить начало и конец: “Z”
LineTo	L/l; H/h (по x); V/v (по y)	Прямая линия к: “L 10,50”; “H 10”; “V 10”
QuadraticCurveTo	Q/q; T/t	Квадратичная кривая Безье: “Q 10,20 50,50”; “T 100,0”
CubicCurveTo	C/c; S/s	Кубическая кривая Безье: “C 10,20 30,40 50,50”; “S 25,50 100,0”

Тег <path> - M/V/H/L команды

```
<svg width="480" height="240" viewBox="0 0 1200 600">
  <g fill="black">
    <circle cx="0" cy="0" r="10"/>
    <circle cx="0" cy="600" r="10"/>
    <circle cx="1200" cy="0" r="10"/>
    <circle cx="1200" cy="600" r="10"/>
  </g>

  <path d="M200,300 v-200 h800 L500 500"
        fill="yellow" stroke="red"
        stroke-width="5" stroke-dasharray="30"/>

  <circle cx="200" cy="300" r="10" fill="red"/>
  <circle cx="200" cy="100" r="10" fill="green"/>
  <circle cx="1000" cy="100" r="10" fill="blue"/>
  <circle cx="500" cy="500" r="10" fill="purple"/>
</svg>
```



Тег <path> - квадратичная кривая

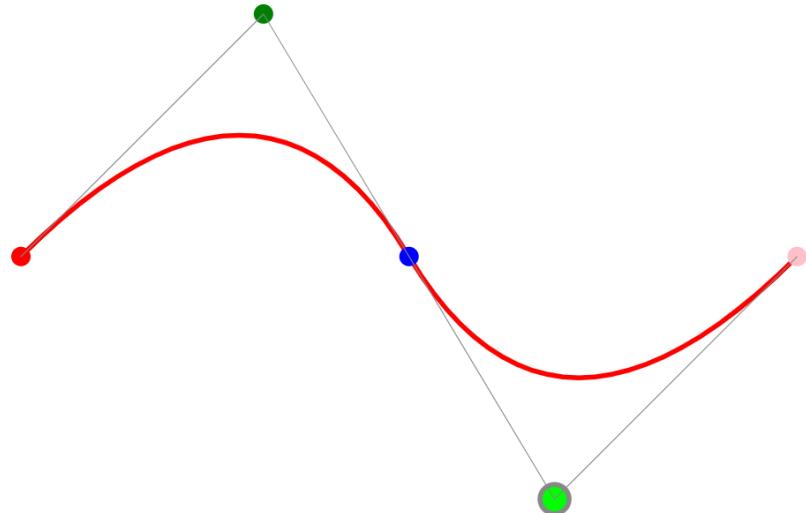
```
<svg width="480" height="240" viewBox="0 0 1200 600">
  <path d="M200,300 Q450,50 600,300 T1000,300"
        fill="none" stroke="red" stroke-width="5" />

  <circle cx="200" cy="300" r="10" fill="red"/>
  <circle cx="450" cy="50" r="10" fill="green"/>
  <circle cx="600" cy="300" r="10" fill="blue"/>

  <circle cx="1000" cy="300" r="10" fill="pink"/>

  <circle cx="750" cy="550" r="15"
        fill="lime" stroke="#888888" stroke-width="5"/>

  <path d="M200,300 L450,50 L600,300 L750,550 L1000,300"
        fill="none" stroke="#888888" stroke-width="1" />
</svg>
```



Тег <path> - кубическая кривая

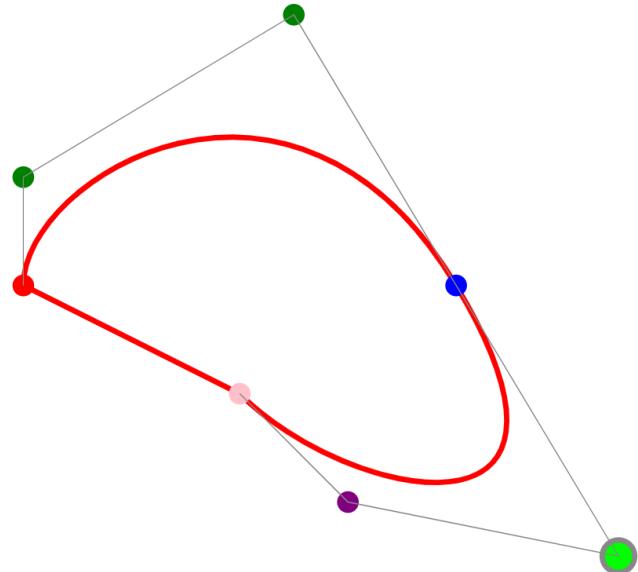
```
<svg width="480" height="240" viewBox="0 0 1200 600">
  <path d="M200,300 C 200,200 450,50 600,300 S 500,500 400,400 Z"
        fill="none" stroke="red" stroke-width="5" />

  <circle cx="200" cy="300" r="10" fill="red"/>
  <circle cx="200" cy="200" r="10" fill="green"/>
  <circle cx="450" cy="50" r="10" fill="green"/>
  <circle cx="600" cy="300" r="10" fill="blue"/>

  <circle cx="750" cy="550" r="15"
        fill="lime" stroke="#888888" stroke-width="5"/>

  <circle cx="500" cy="500" r="10" fill="purple"/>
  <circle cx="400" cy="400" r="10" fill="pink"/>

  <path d="M200,300 L200,200 L450,50 L600,300
         L750,550 L500,500 400,400"
        fill="none" stroke="#888888" stroke-width="1" />
</svg>
```



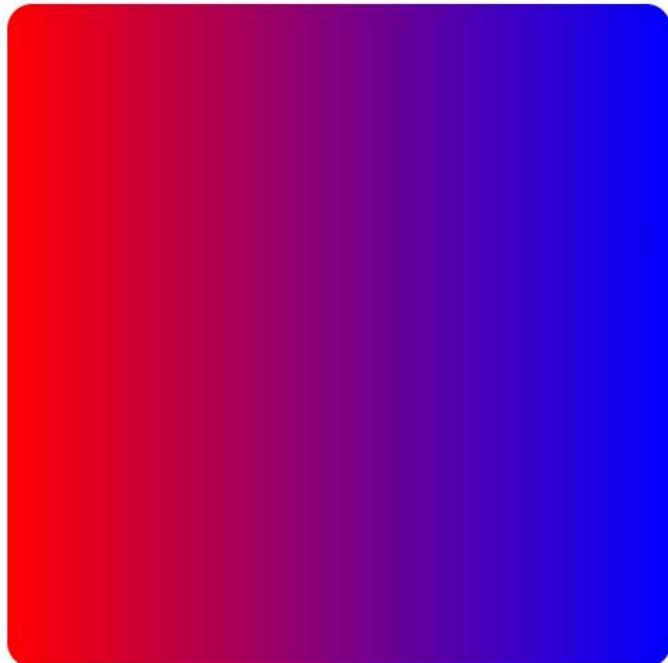
Другие теги

Tag	Описание
<g>	Группирует теги, задает общие атрибуты
<defs>	Позволяет задавать графические объекты для последующего использования
<linearGradient>	Задает линейный градиент. На тег ссылаются из атрибута fill="url(#id_name)". Дочерние теги <stop> задают используемые цвета и их границы.
<radialGradient>	То же самое, но градиент радиальный (круговой)
<style>	Такой же смысл как и в HTML
<text>	Текст

Линейный градиент

```
<svg width="400" height="400">
  <defs>
    <linearGradient id="Gradient1">
      <stop offset="0%" stop-color="red"/>
      <stop offset="100%" stop-color="blue"/>
    </linearGradient>
  </defs>

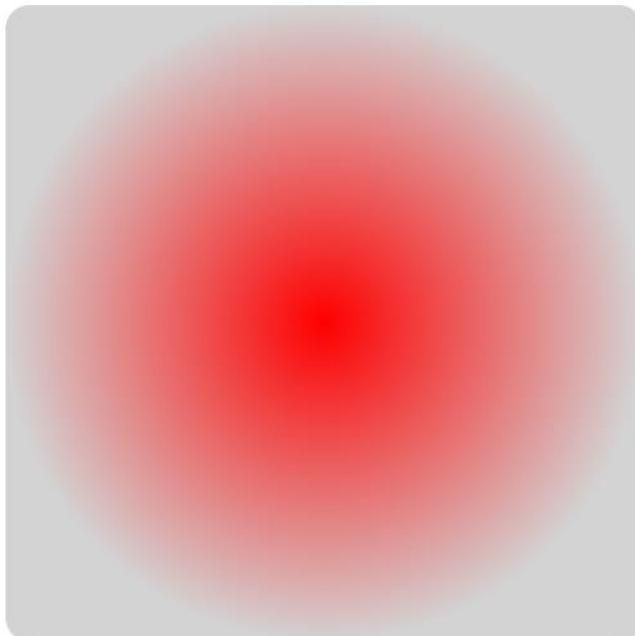
  <rect x="0" y="0" rx="15" ry="15"
        width="400" height="400" fill="url(#Gradient1)"/>
</svg>
```



Радиальный градиент

```
<svg width="400" height="400">
  <defs>
    <radialGradient id="Gradient1">
      <stop offset="0%" stop-color="red"/>
      <stop offset="100%" stop-color="lightgray"/>
    </radialGradient>
  </defs>

  <rect x="0" y="0" rx="15" ry="15"
    width="400" height="400" fill="url(#Gradient1)"/>
</svg>
```



<text> и <style>

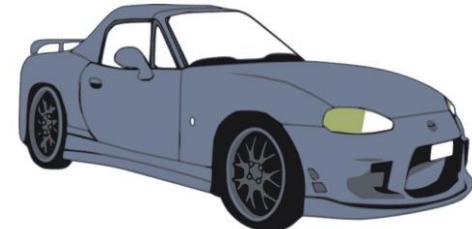
```
<svg viewBox="0 0 300 500" xmlns="http://www.w3.org/2000/svg">
  <style>
    .small { font: italic 13px sans-serif; }
    .heavy { font: bold 17px sans-serif; }
    .Rrrrr { font: italic 15px serif; fill: red; }
  </style>

  <path id="my_path" d="M 20,70 C 80,80 100,40 120,70"
        fill="none"/>
  <path id="my_path" d="M 20,70 C 80,80 100,40 120,70"
        fill="none" stroke="yellow"/>
  <text x="20" y="35" class="small">здесь</text>
  <text x="60" y="35" class="heavy">могла бы</text>
  <text x="30" y="55" class="small">быть</text>
  <text class="Rrrrr">
    <textPath href="#my_path">ваша реклама!</textPath>
  </text>
</svg>
```

Здесь **могла бы**
быть
ваша реклама!

Сравнение растровых и векторных изображений

Критерий	Растровая графика	Векторная графика
Тип файла	JPG, JPEG, PNG и др.	SVG, EPS и др.
Внутреннее представление	RGB(A) матрица из пикселей	Геометрические фигуры, заданные XML-разметкой
Увеличение масштаба картинки	Явная видимость пикселей	Неограниченная масштабируемость без потери качества
Объем занимаемой памяти	Не зависит от сложности объектов, но зависит от разрешения	Простые фигуры: малый размер файла. Сложные и зашумленные фигуры – объем резко возрастает
Назначение	Фотографии	Графический дизайн, иконки, логотипы, текстовые шрифты, иллюстрации и прочее
Генерация в машинном обучении	Отлично изучена. Существует множество моделей для классификации и генерации растровых изображений.	Почти не изучена. Основные трудности в выборе количества генерируемых фигур и их сложности.



Векторная графика

Растровая графика

Где можно найти векторные изображения?

- <https://pictogrammers.com/library mdi/>
- <https://icons.getbootstrap.com/>
- <https://undraw.co/illustrations>
- <https://tablericons.com/>
- <https://freesvg.org/>
- <https://www.svgrepo.com/>
- <https://www.flaticon.com/free-icons/dataset>

Удобный онлайн редактор SVG: <https://www.svgviewer.dev/>

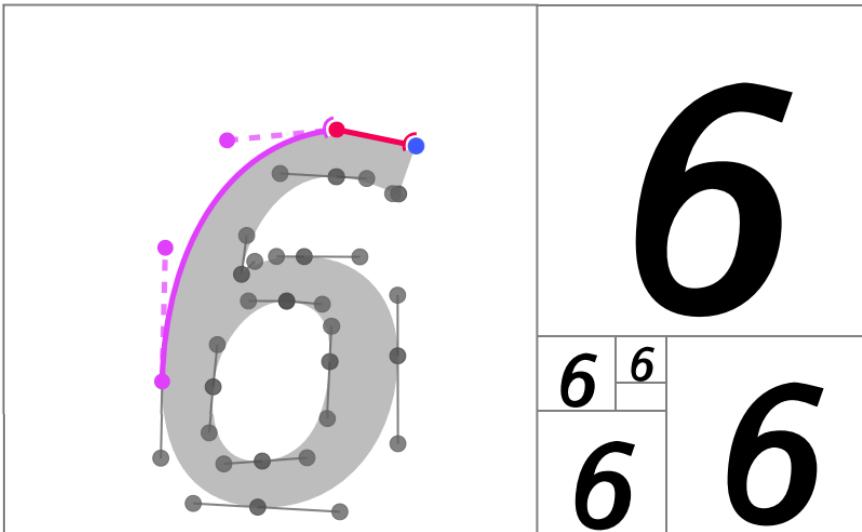
Методы обработки векторной графики

SVG-VAE (2019)

moveTo (15, 25)

lineTo (-2, 0.3)

cubicBezier (-7.4, 0.2) (-14.5, 11.7), (-12.1, 23.4)



Генерация шрифтов (0-9, a-z, A-Z)

Датасет: 14М символов.

4 типа команд:

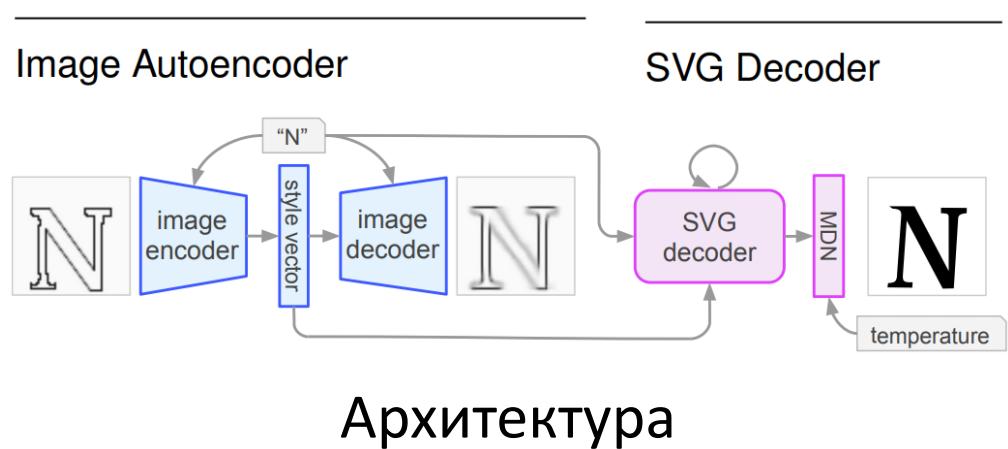
- moveTo
- lineTo
- cubicBezier
- EOS

Генерируем один символ = одну последовательность.
Суммарно команд не более 50.

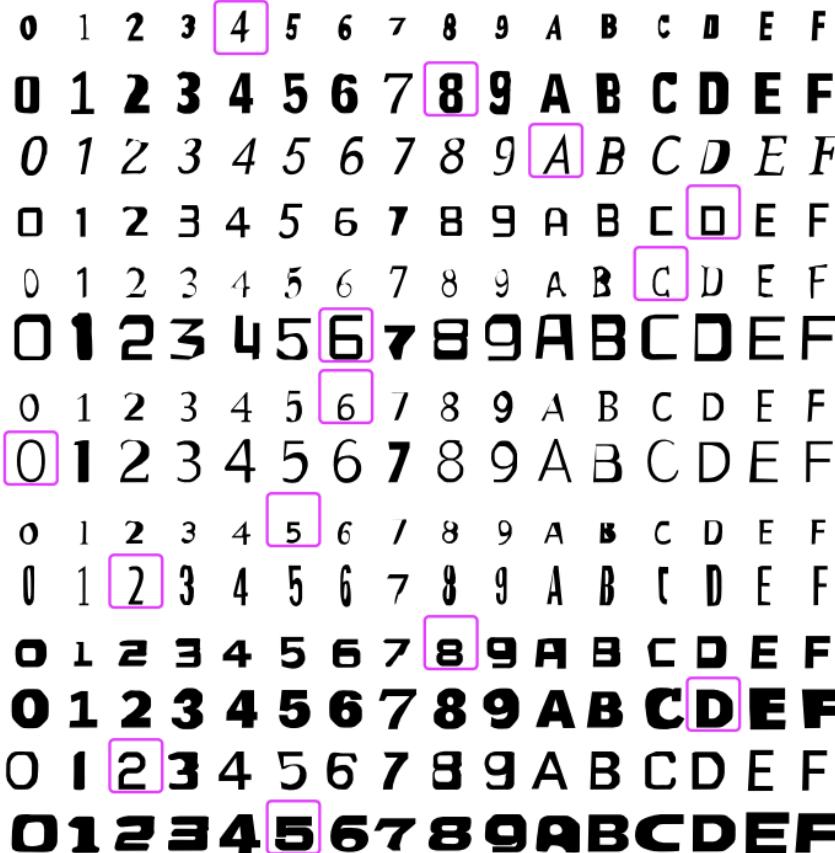
<https://arxiv.org/pdf/1904.02632.pdf>

SVG-VAE (2019)

- Скрытый слой CVAE – стиль шрифта.
- Декодер – стек 4 LSTM.
- MDN – Mixture Density Network – выдает список команд + их аргументов.
- Тренируем отдельно VAE, потом декодер.



SVG-VAE (2019)



Проверка стилистики скрытого слоя:

- Выбрали вектор стиля у конкретного символа (фиолетовый квадрат)
- По этому вектору декодер сгенерировал оставшиеся символы
- Итог: стиль *почти* однообразен.

Текущие проблемы

Хотим уметь генерировать более сложные SVG:

- Несколько последовательностей (путей)
- Добавить цветогенерацию
- Учитывать прозрачность (opacity) и т.п.

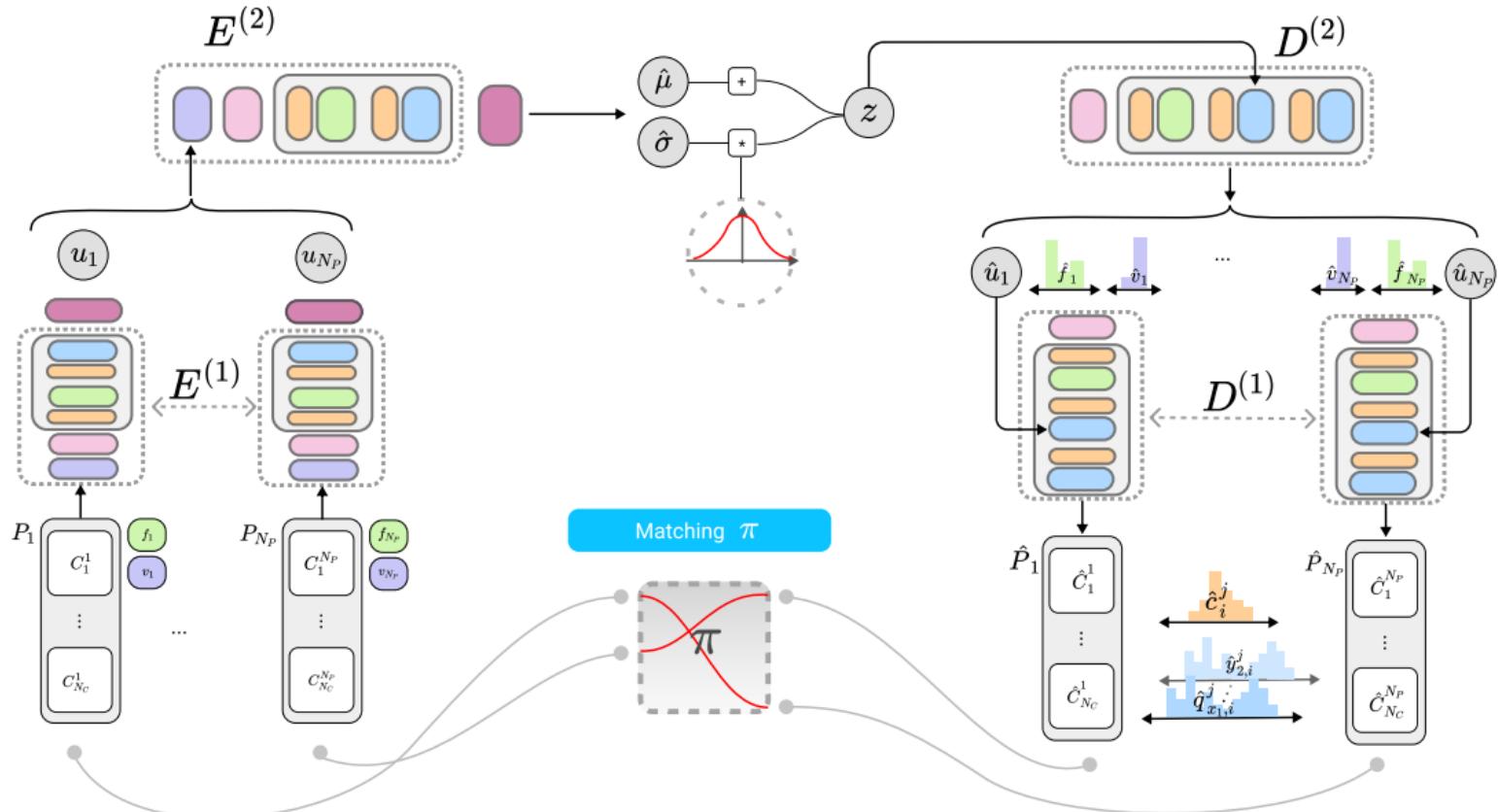
Хотим использовать подходы из NLP + трансформеров

DeepSVG (2020) - идея

Идея – двухэтапный VAE:

- закодируем каждый путь, получим набор внутренних представлений
- объединим эти представления и опять закодируем
- полученное – внутреннее представление всей картинки
- аналогично дважды декодируем
- Эмбеддинг команды:
 - Эмбеддинг для типа команды (<SOS>, M, L, C, Z, <EOS>)
 - Эмбеддинг для координат (вектор из 6 чисел)
 - Позиционное кодирование команд

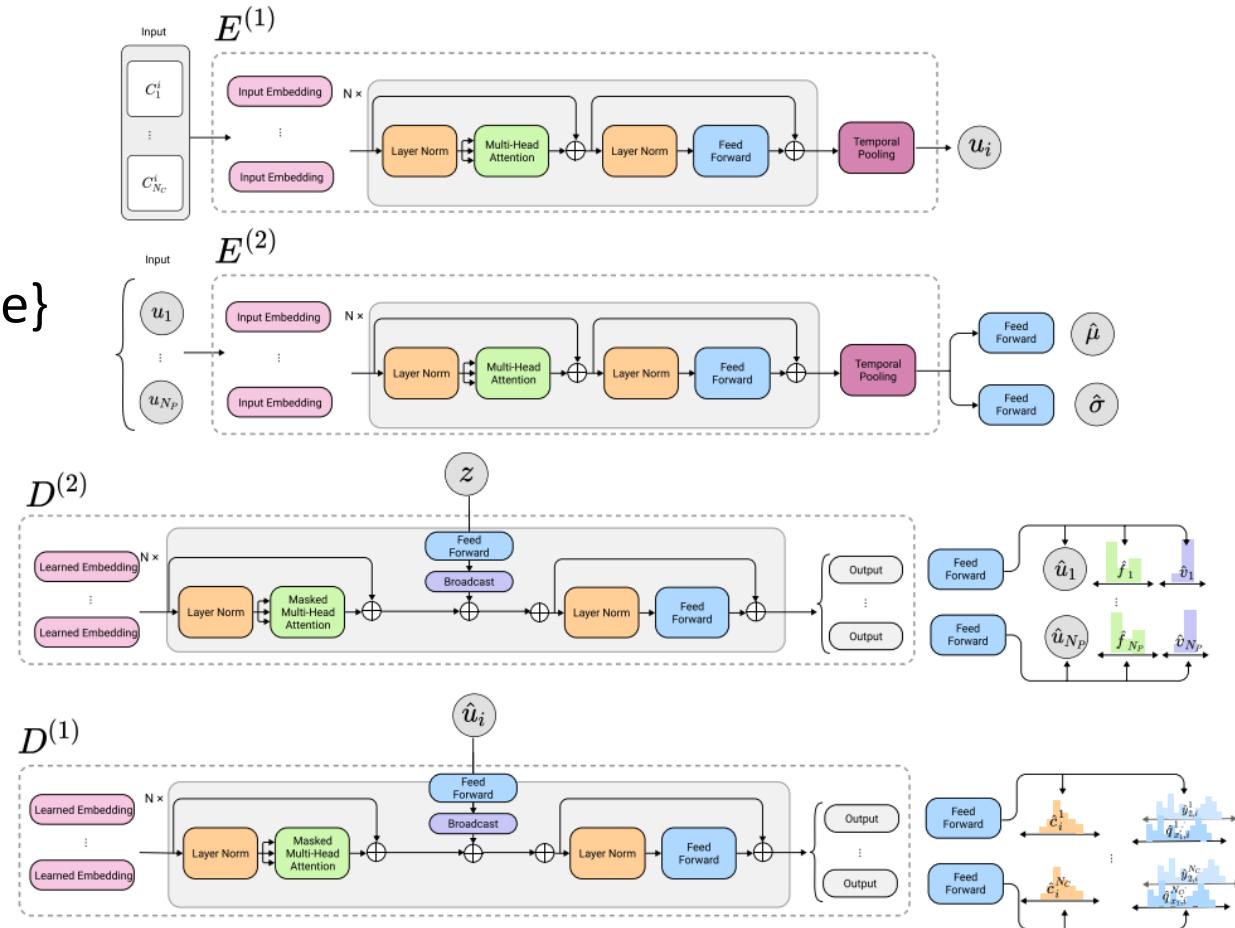
DeepSVG (2020) – общая архитектура



DeepSVG (2020) – подробная архитектура

Атрибуты тега path:

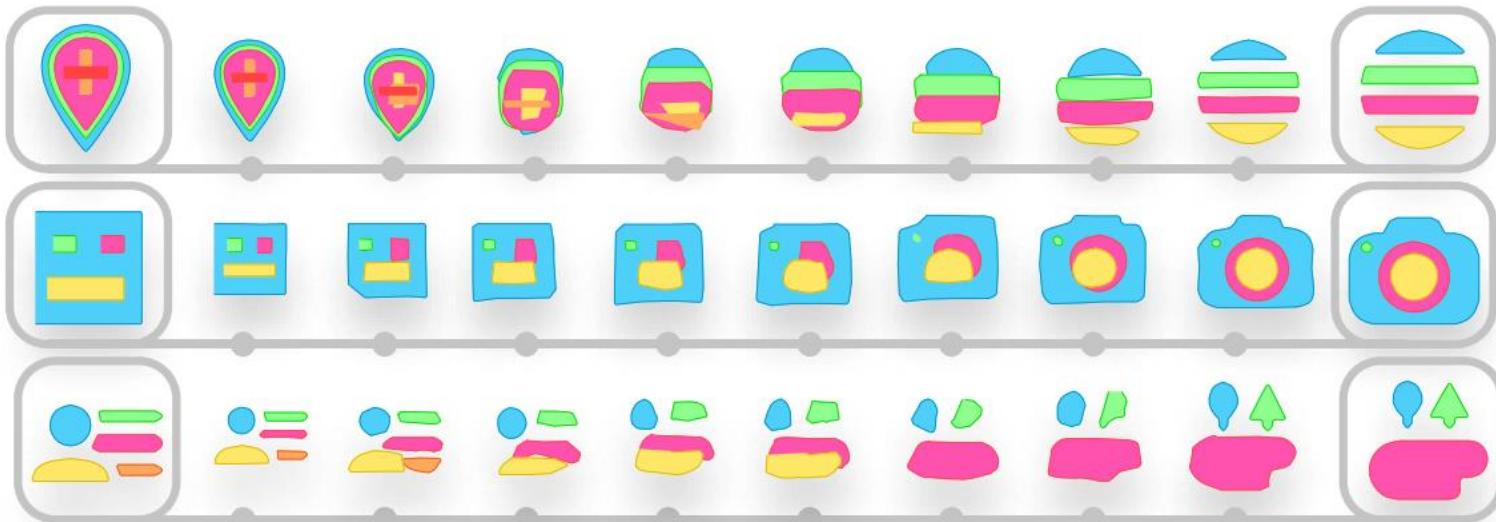
- команды (атр. d)
- fill = {outline; fill; erase}
- visibility = {0; 1}



DeepSVG (2020) - результаты

0 1 2 3 4 5 6 7 8 9

a b c d e f g h i j k l m n o p q r s t u v w x y z
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z



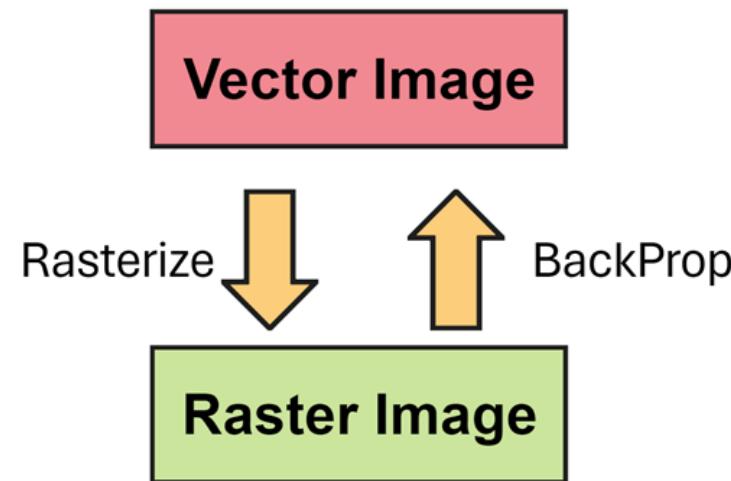
Текущие проблемы

Внутреннее ограничение на кол-во путей и команд.

Нужен датасет из векторных изображений.
Собрать его – дело непростое.

DiffVG (2020)

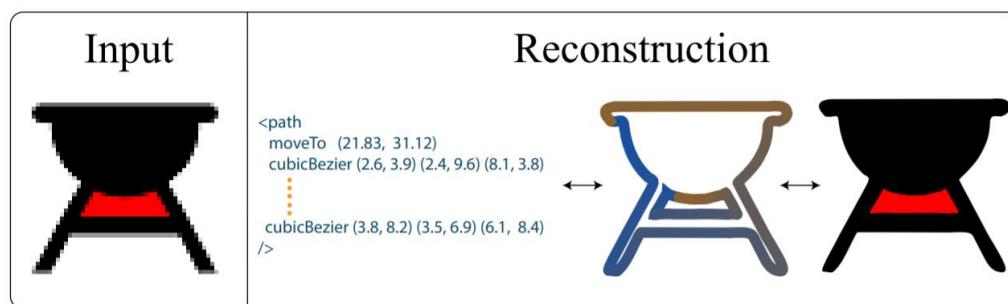
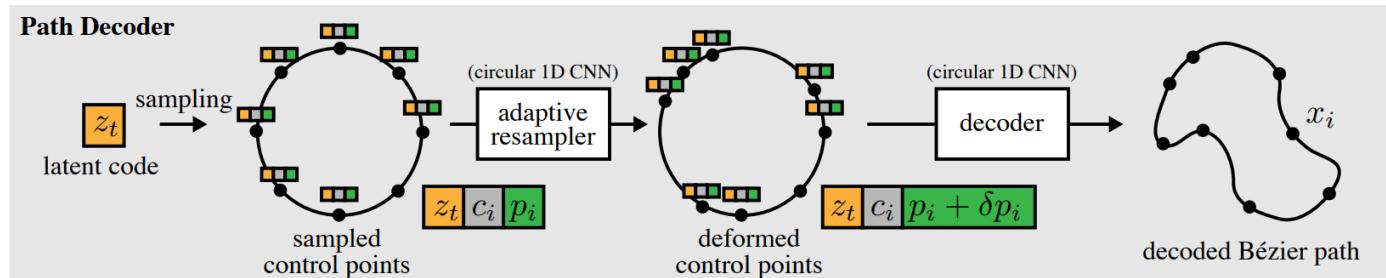
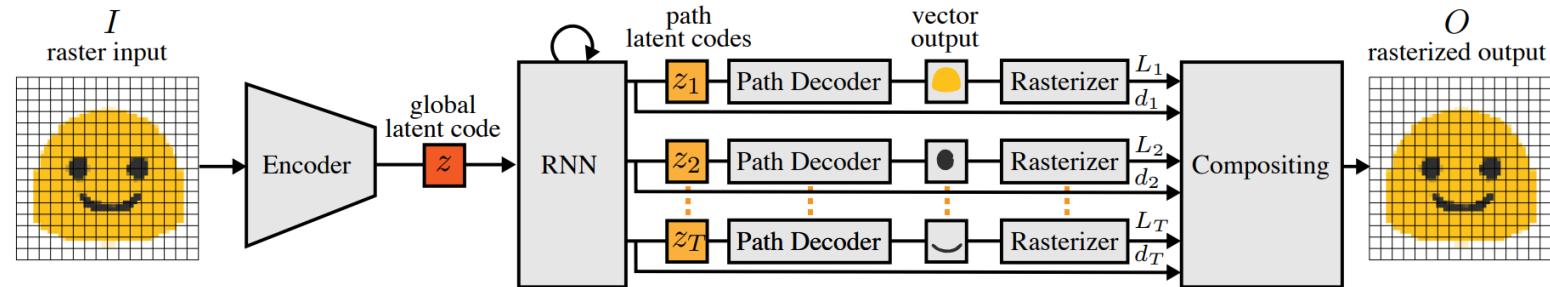
Дифференцируемый растеризатор векторной графики



Позволяет «связать» векторную и растровую графики через обратное распространение ошибки.

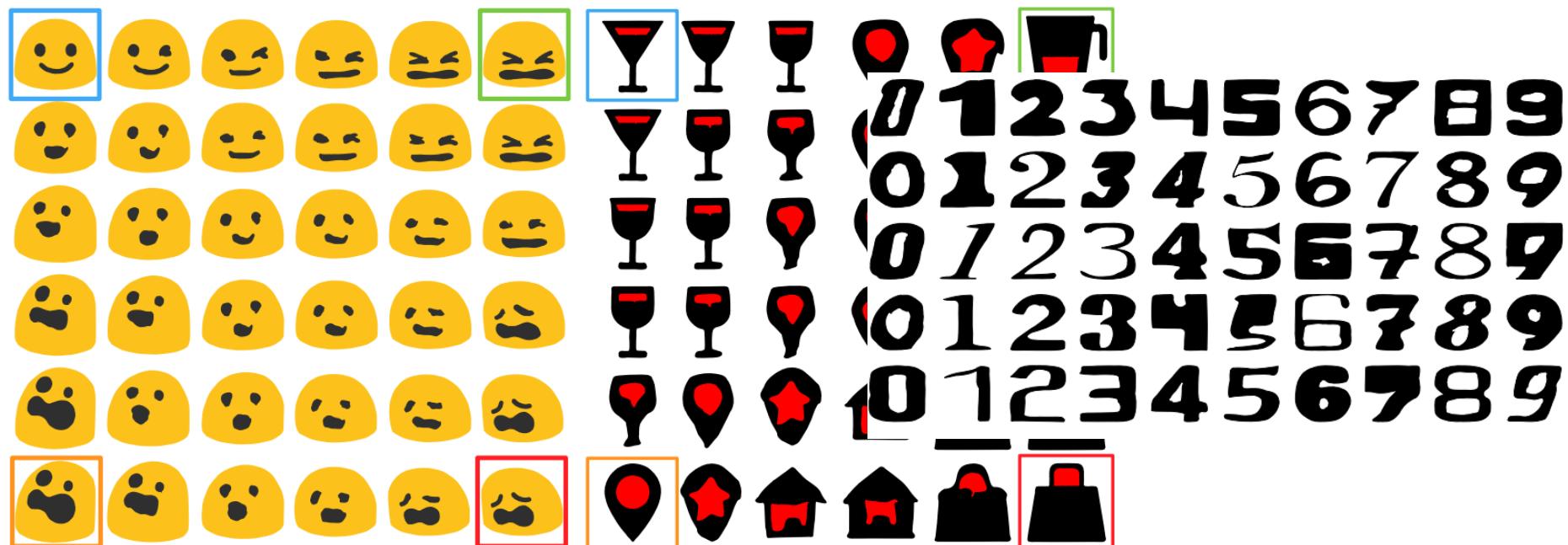
Можно сконструировать векторное изображение, растеризовать его и эту растеризацию использовать в функции потери.

Im2Vec (2021) – векторизатор растровой графики

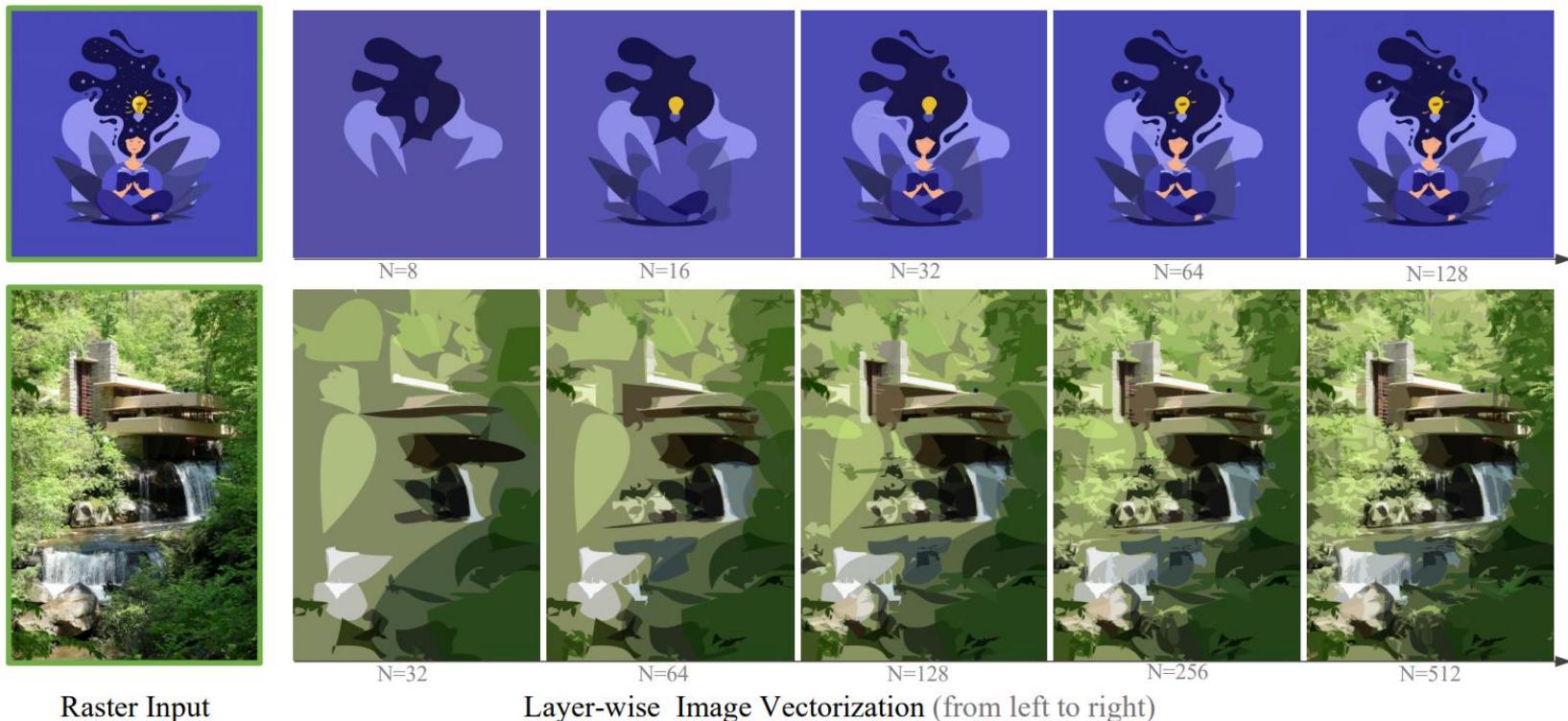


<https://arxiv.org/pdf/2102.02798.pdf>

Im2Vec (2021) – результаты



LIVE (2022) – векторизатор раcтровой графики



<https://arxiv.org/pdf/2206.04655.pdf>

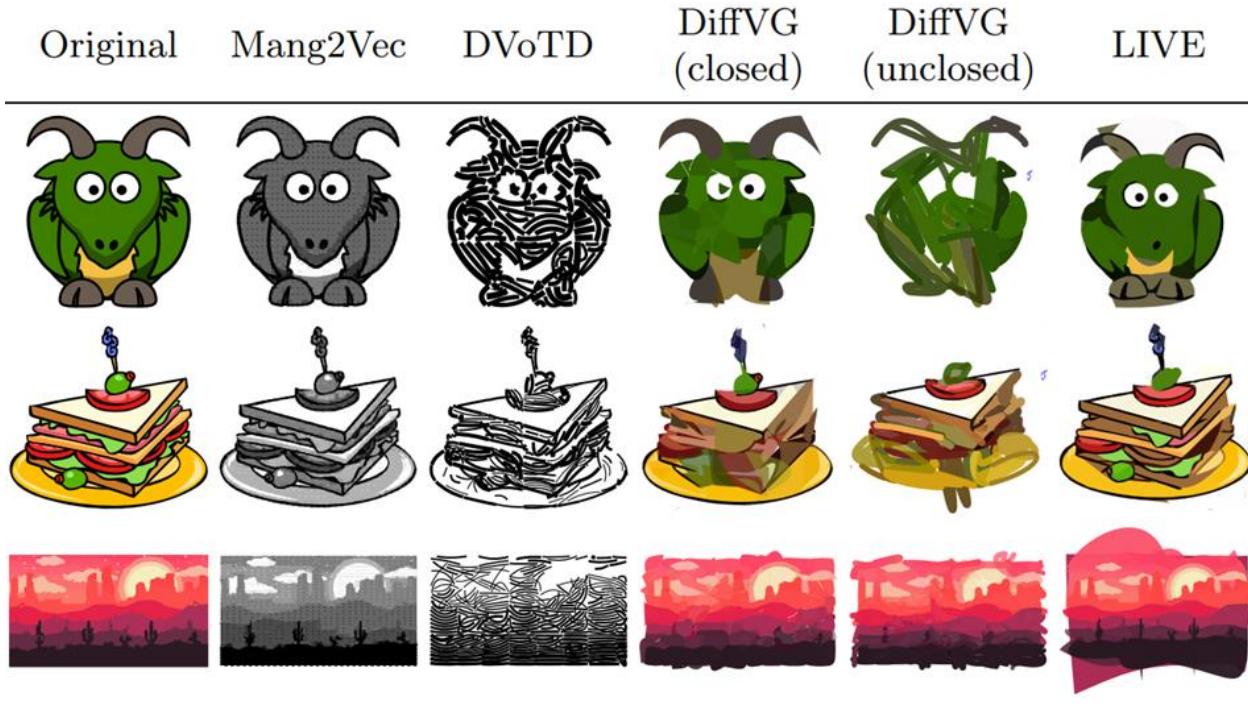
Достоинства:

- Введен лосс на самопересечение фигур (*)
- Можно получать векторизации картинок с заданным количеством фигур

Недостатки:

- Работает чрезвычайно долго

Чем плоха векторизация?

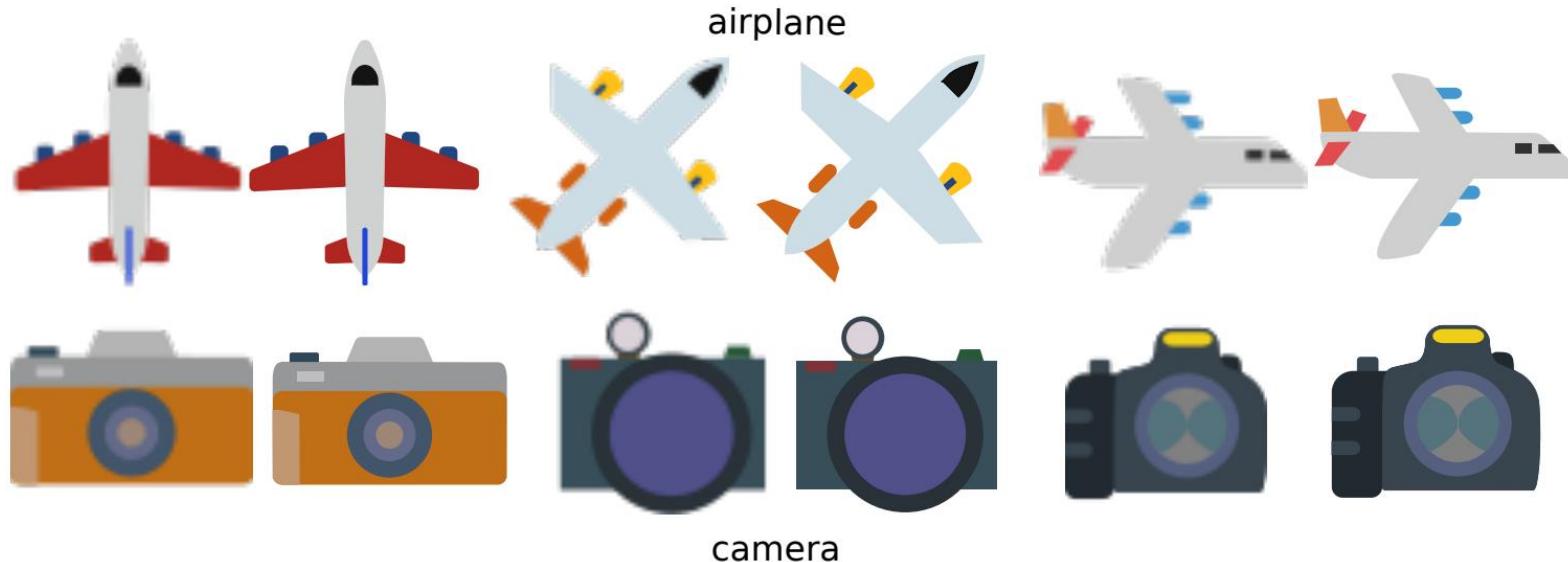


Проблемы:

- Большое время обработки;
- используется очень много фигур;
- плохое качество;
- нужна настройка гиперпараметров => невозможно автоматизировать;
- нужно предобучение.

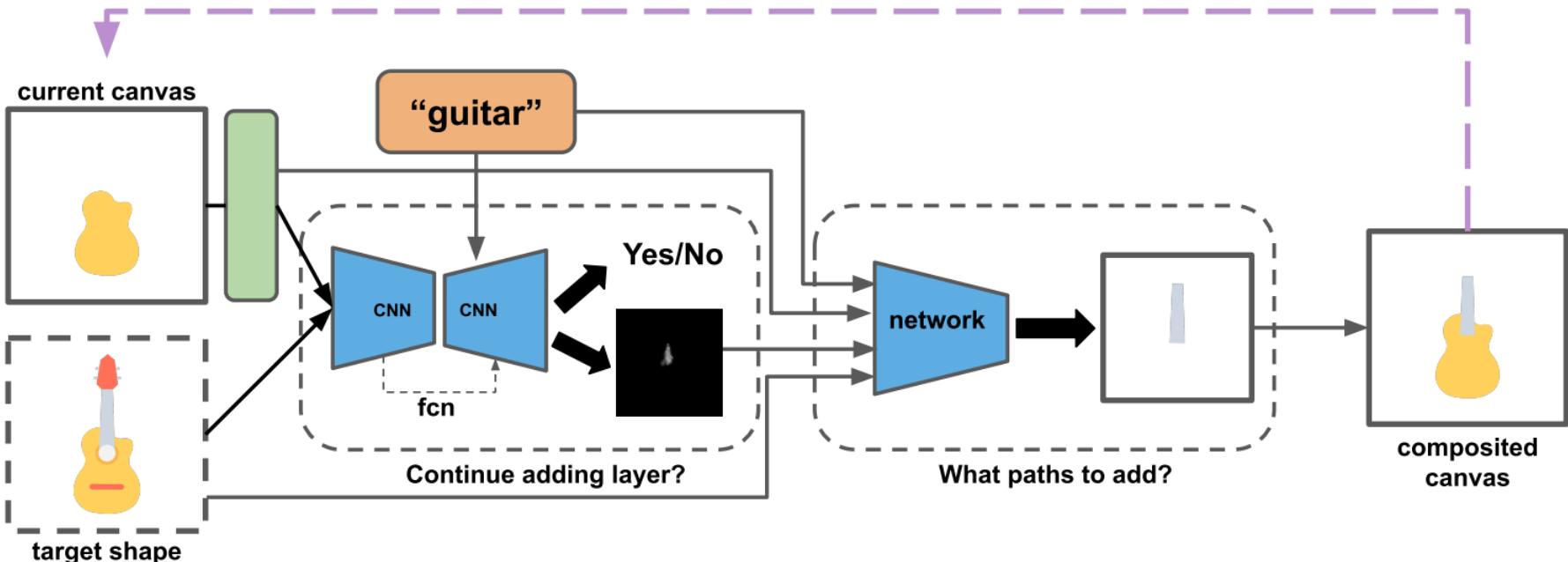
ClipGen (2021)

Векторизатор растровой графики с указанием класса



<https://arxiv.org/pdf/2106.04912.pdf>

ClipGen (2021) – архитектура



EvoVec (2024)

Векторизация изображений с помощью эволюционных алгоритмов

Алгоритм:

1. Инициализируем изображение с помощью SVGTracer
2. Эволюционным алгоритмом удаляем лишние фигуры, сегменты кривых, исправляем контуры фигур

EvoVec

Исходное изображение	LIVE (N=64)	DiffVG (N=512)	SvgTracer	EvoVec	Метрика
					Результат
	51,5 м.	45,4 м.	7,5 с.	24,5 м.	Время работы
	4760	4255	1780	1547	Значение ф. отбора
	64	512	1790	985	Кол-во путей

https://link.springer.com/chapter/10.1007/978-3-031-70085-9_24

LIVBOC (2025)

Layerwise Image Vectorization Via Bayesian-Optimized Contours

Векторизатор изображений, аналог LIVE.

Подход:

1. Проработан процесс инициализации путей – совершается Байесовской оптимизацией

$$\mathcal{L}_{\text{Bayesian}} = \mathcal{L}_{\text{Reconstruction}} + \lambda \mathcal{L}_{\text{Overlap}} + \gamma \mathcal{L}_{\text{Coverage}}$$

2. Итеративная оптимизация путей с помощью diffvg

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{reconstruction}} + \alpha \mathcal{L}_{\text{smooth}} + \beta \mathcal{L}_{\text{overlap}}$$

<https://www.scitepress.org/Papers/2025/133810/133810.pdf>

LIVBOC сравнение

Initial image	LIVE	LIVBOC (Ours)	Metrics
			Visualization of Vector Compactness
	0.0216	0.0033	Reconstruction Loss
	0.9310	0.9841	SSIM
	16,911	5,433	Time(s)
	42	41	File Size(KB)
	64	32	Paths

LIVBOC

Достоинства:

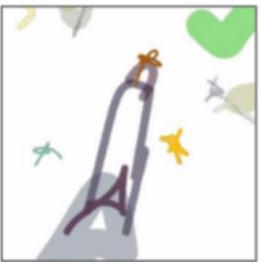
- Точнее и более оптимально производит векторизацию
- Работает примерно в два раза быстрее LIVE

Недостатки:

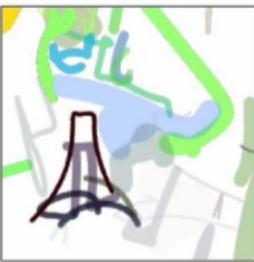
- Всё еще работает довольно долго

CLIPDraw (2021)

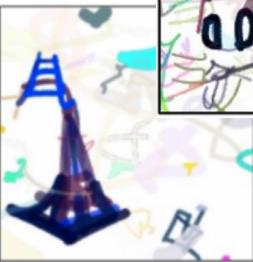
Генерация SVG изображения по входному тексту



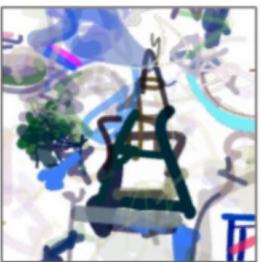
16 Strokes



32 Strokes



64 Strokes



128 Strokes



256 Strokes



512 Strokes

“A drawing of a cat”.



“A painting of a sunset”.



“Underwater”.



“Sheep wearing a top hat”.

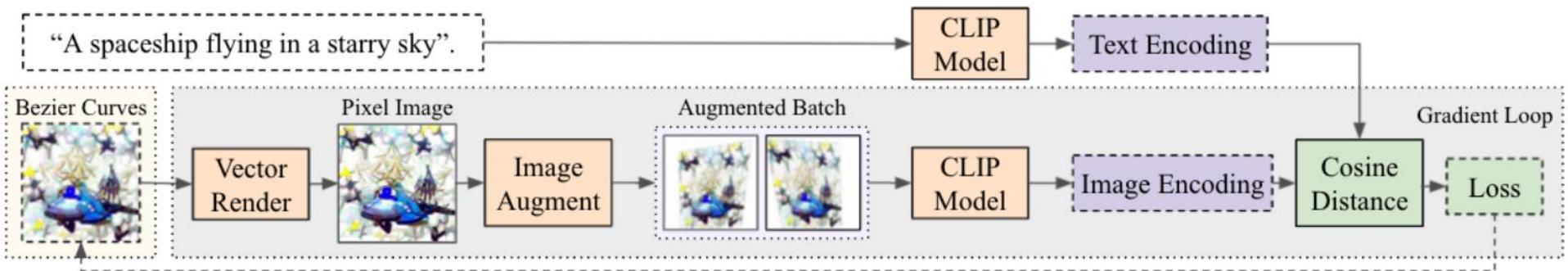


Параметры:

- ЧИСЛО КРИВЫХ
- КОЛ-ВО ИТЕРАЦИЙ
- ЖИРНОСТЬ ОБВОДКИ (нет fill)

<https://arxiv.org/pdf/2106.14843.pdf>

CLIPDraw (2021) - архитектура



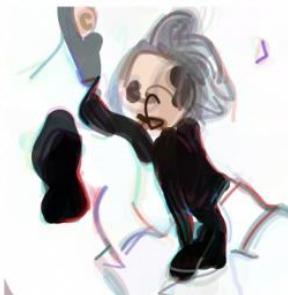
StyleCLIPDraw (2022)

Генерация изображения по тексту согласно стилю

Стиль:



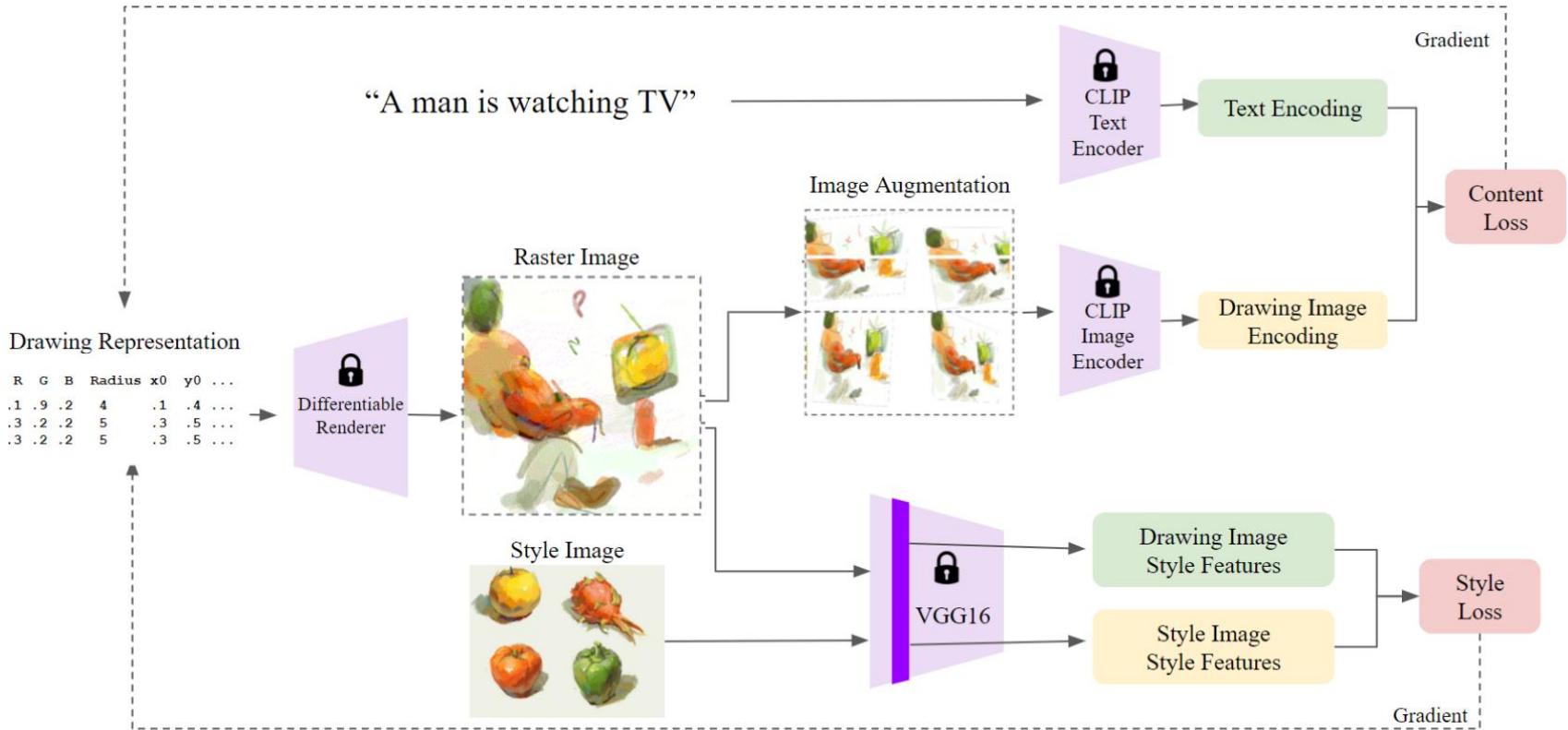
Результат:



Текстовый запрос: «Танцующий Альберт Эйнштейн»

<https://arxiv.org/pdf/2202.12362.pdf>

StyleCLIPDraw (2022) - архитектура



VectorNST (2023)

Перенос стиля в векторной графике



Контент



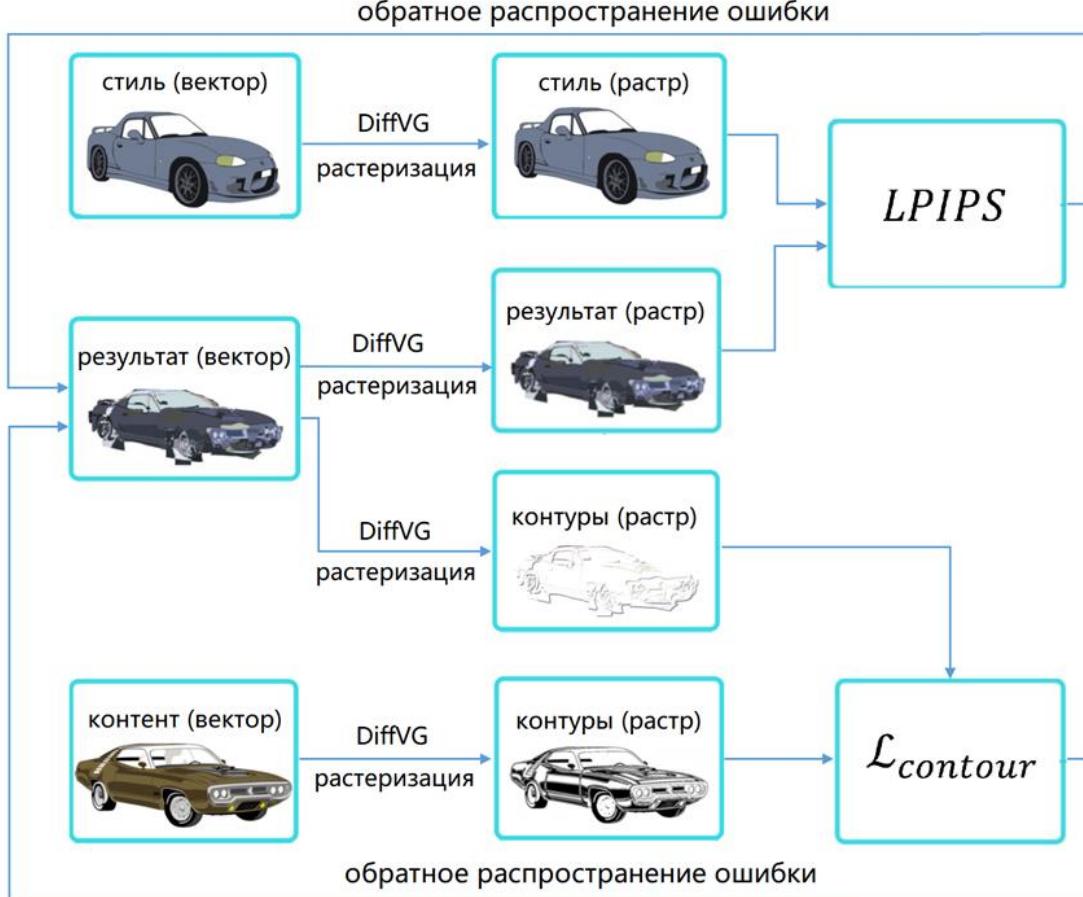
Стиль



Результат

<https://arxiv.org/pdf/2303.03405.pdf>

VectorNST - перенос стиля для векторной графики



Итеративный подход

Jarsky, I. ; Efimova, V. ; Chebykin, A. ; Shalamov, V. and Filchenkov, A. (2024). **Neural Style Transfer for Vector Graphics**. In Proceedings of the 19th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 3: VISAPP; ISBN 978-989-758-679-8; ISSN 2184-4321, SciTePress, pages 686-693. DOI: 10.5220/0012438200003660

VectorNST - результаты

Контент



Стиль



Результат



DeepVecFont (2021)

Few-shot генерация векторных шрифтов

Вход: 4 символа одного стиля.

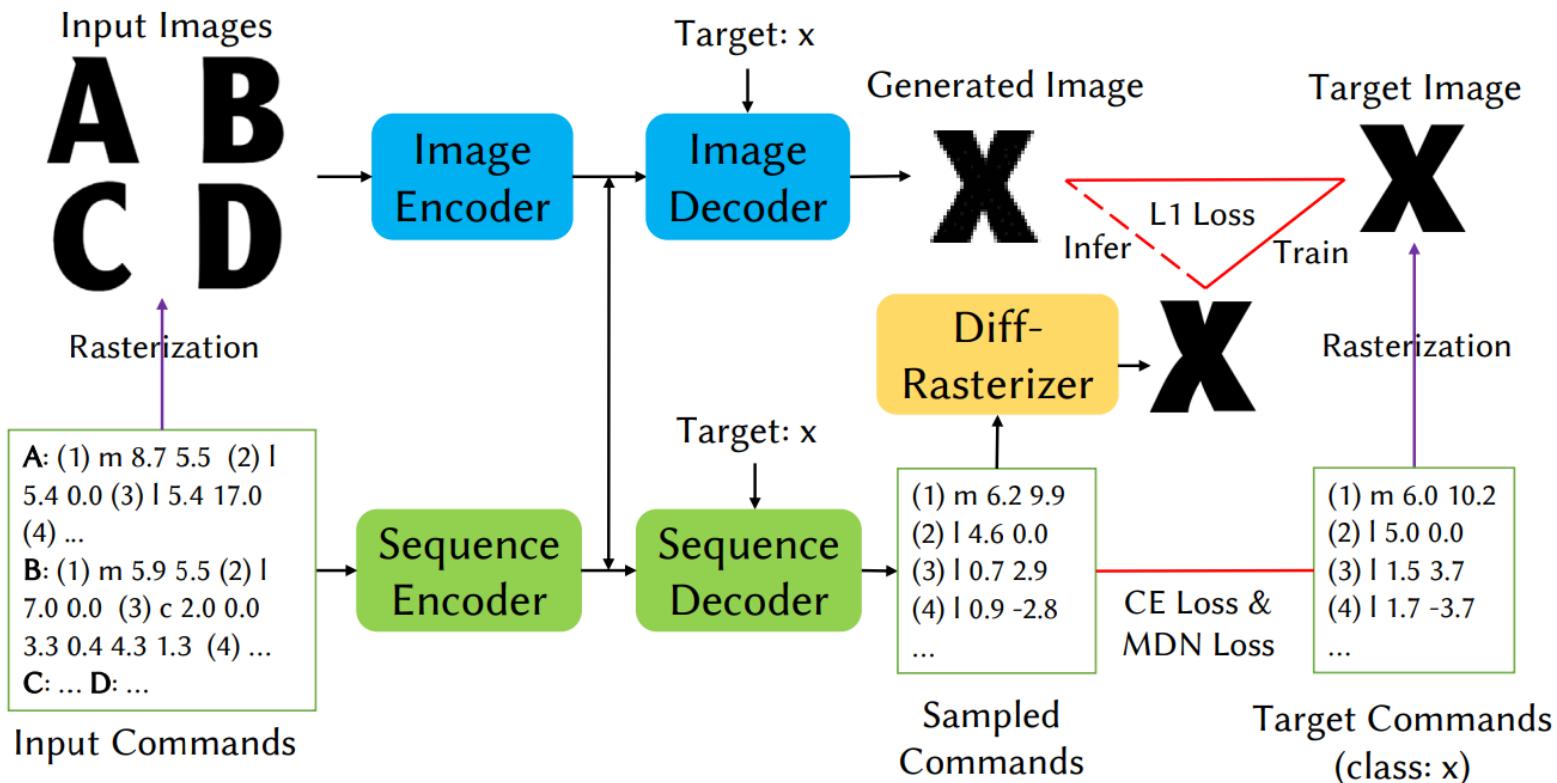
Выход: набор символов a-zA-Z с данным стилем.

Идея – два аспекта использования входной информации:

- последовательность команд
- растеризованное векторное изображение

<https://arxiv.org/pdf/2110.06688.pdf>

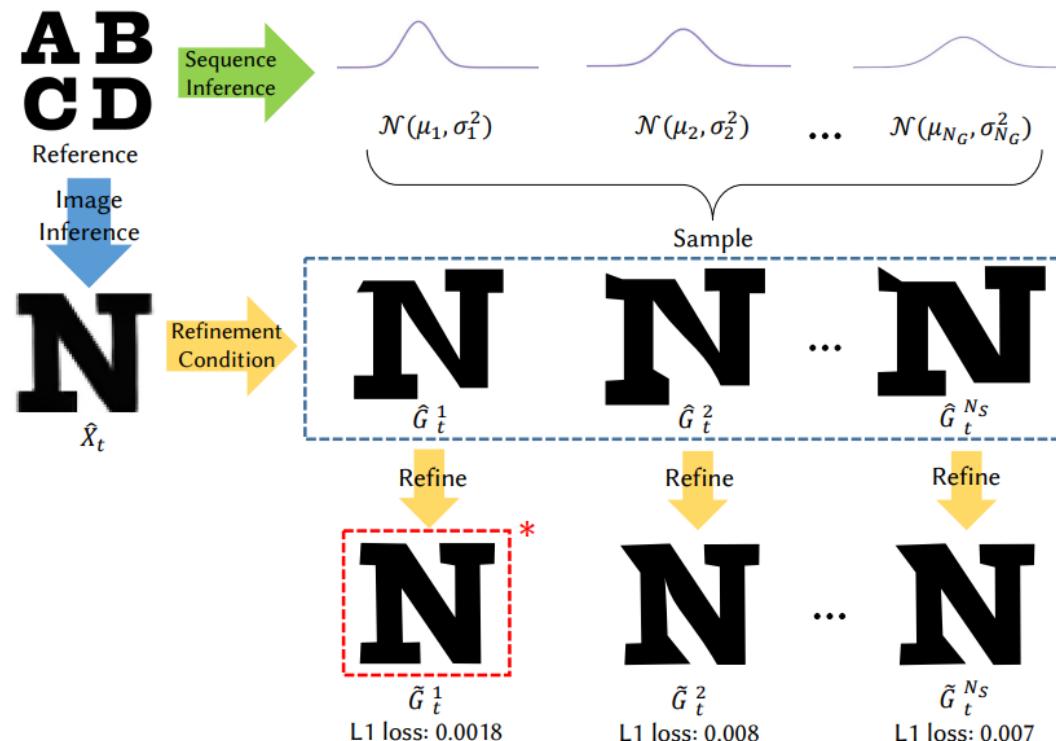
DeepVecFont (2021) – архитектура



DeepVecFont (2021) – архитектура

Refinement:

Итеративно оптимизируем
с помощью DiffVG
координаты полученных
векторных изображений
(тип команд зафиксирован)



DeepVecFont (2021) - результаты

DeepVecFont
(w/o refinement)

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
a b c d e f g h i j k l m n o p q r s t u v w x y z

DeepVecFont
(w/ refinement)

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
a b c d e f g h i j k l m n o p q r s t u v w x y z

Ground Truth

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
a b c d e f g h i j k l m n o p q r s t u v w x y z

A B Г M R Ш а б м н о р

Synthesized Vector Font (w/o refinement)

П В Г М Р Ш а б м н о р

Synthesized Vector Font (w/ refinement)

A B Г M R Ш а б м н о р

Synthesized Glyph Images

A B Г M R Ш **а б** м н о р

Ground-Truth Font

CoverGAN - генератор векторных обложек

Требования к сервису

Входные параметры:

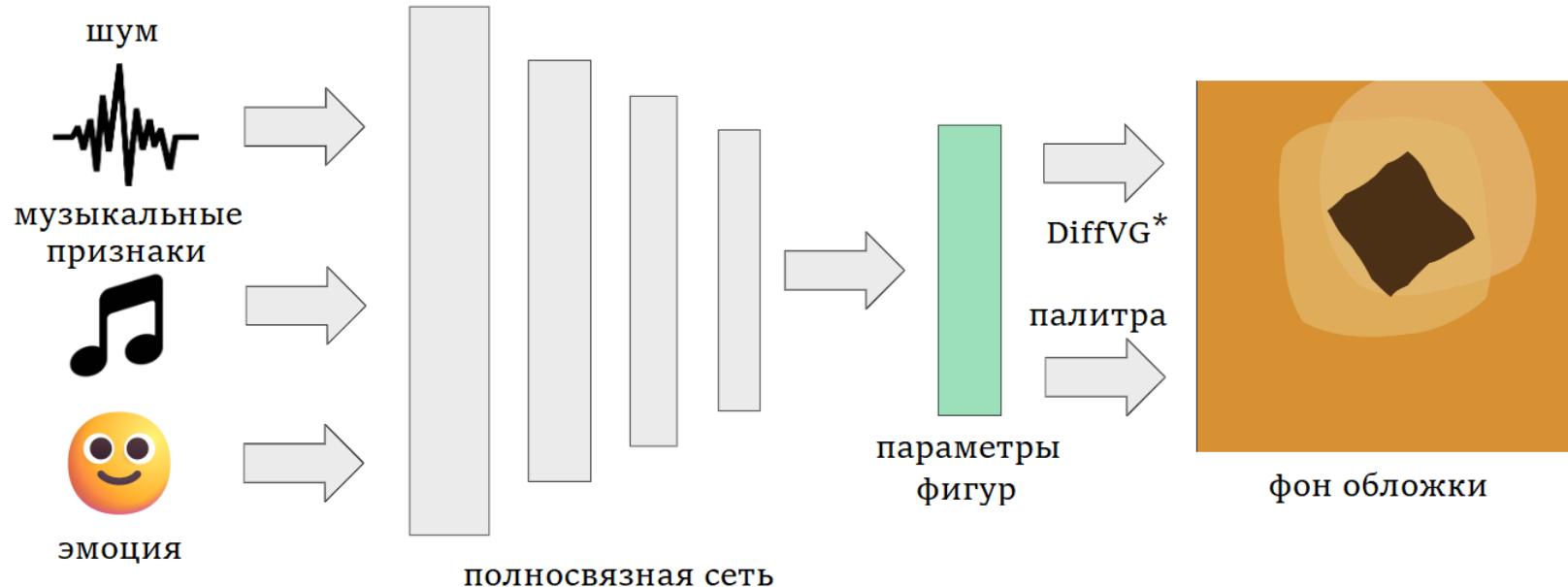
- Трек музыкальной композиции.
- Имя исполнителя и композиции.
- Вызываемая эмоция.

Параметры выходной обложки:

- Формат SVG.
- Подпись музыкальной композиции на обложке.



Архитектура генератора CoverGAN (2022)



* DiffVG – дифференцируемый растеризатор векторной графики

Примеры обложек CoverGAN (2022)



VectorFusion (2022)

Генерация изображения по тексту



<https://arxiv.org/pdf/2211.11319.pdf>

VectorFusion (2022) - метод

A panda rowing a boat in a pond.



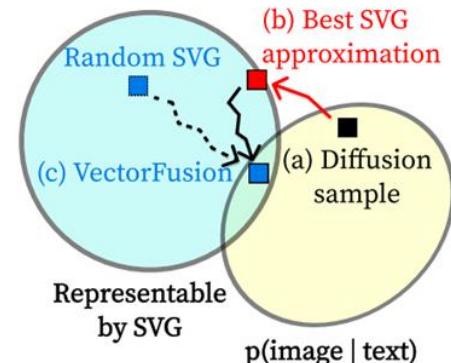
(a) Sample raster image
with Stable Diffusion



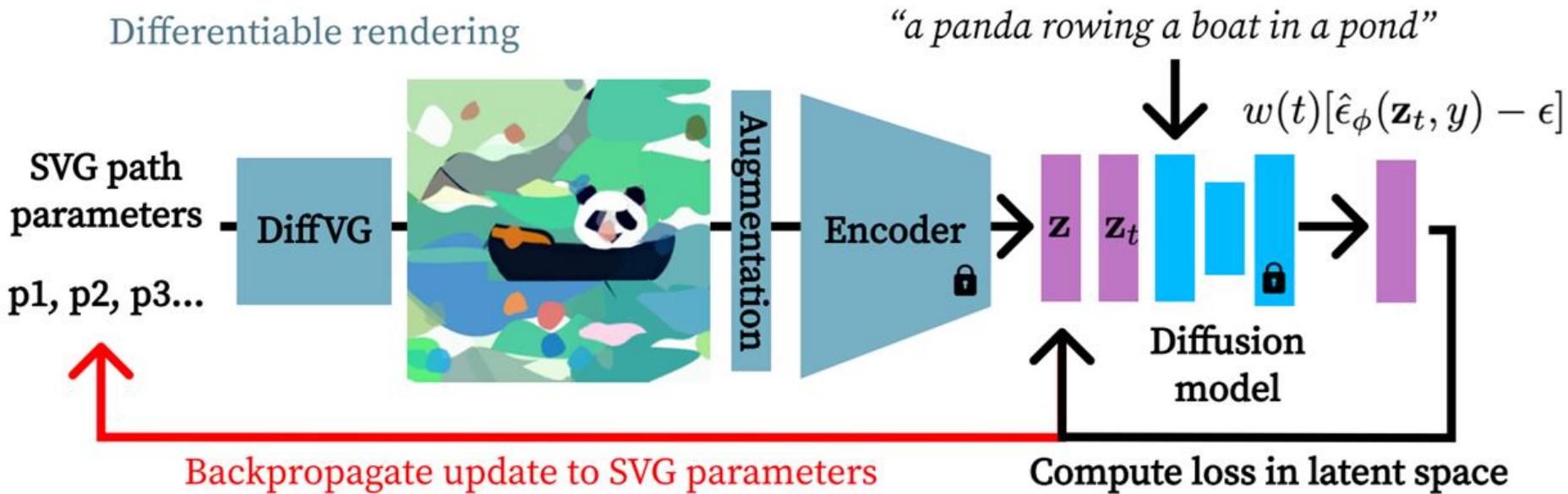
(b) Convert raster image to a vector



(c) VectorFusion: Fine tune
by latent score distillation



VectorFusion (2022) - Finetune блок



SVGDreamer - генерация SVG по тексту с диффузионной моделью



"Darth Vader with lightsaber"; **Pixelart**



"Sonic"; **Pixelart**



"Pikachu, childish and fun"; **Pixelart**



"Polar bear"; **Low-poly**



"Bald eagle"; **Low-poly**



"Scarlet macaw"; **Low-poly**



"Wolf. flat 2d vector."; **Low-poly**



"Abstract Vincent van Gogh Oil Painting Elephant"; **Painting**



"The image captures the essence of Vincent van Gogh, colorful world he painted"; **Painting**



"A speeding Lamborghini"; **Sketch**



"An airplane"; **Sketch**

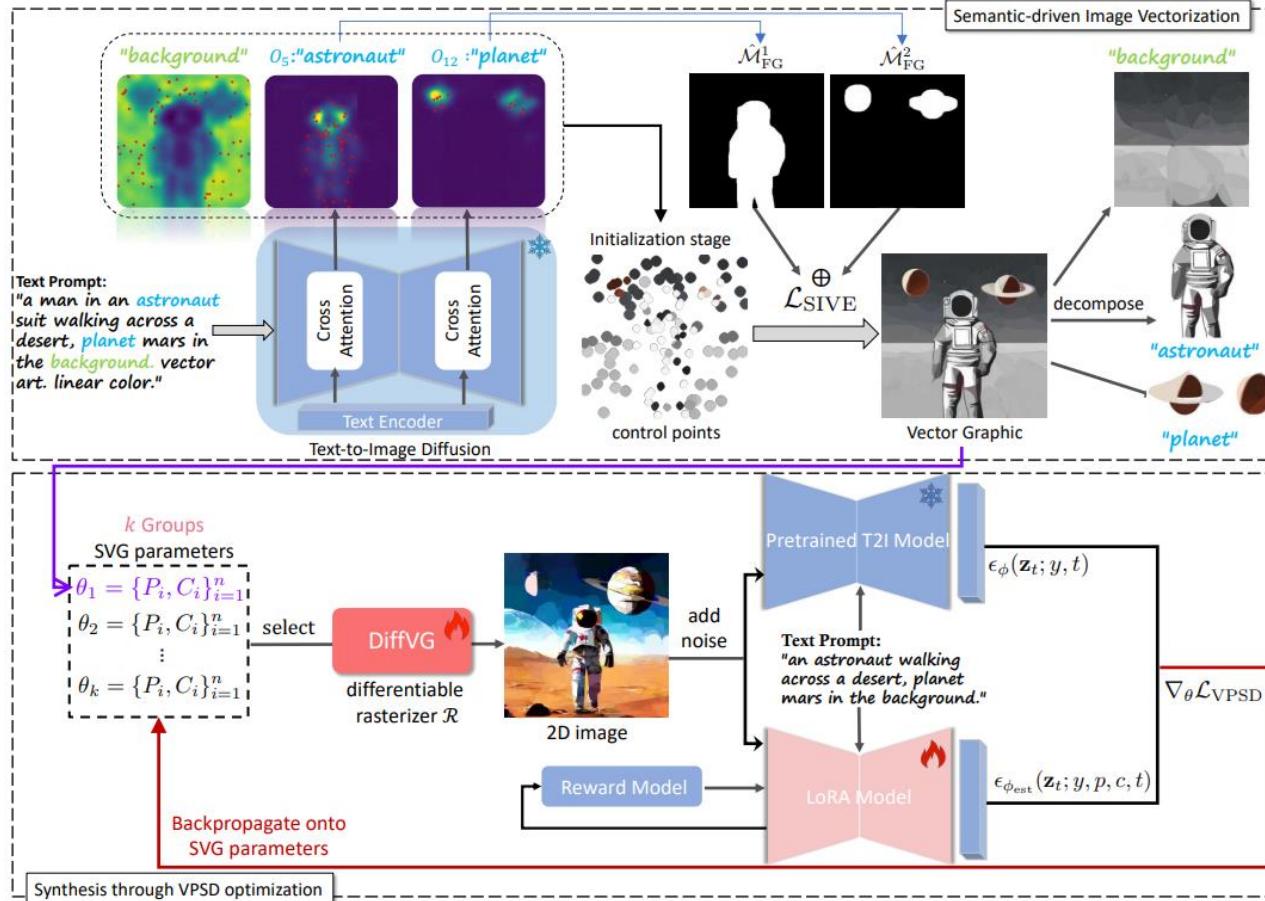


"Black and white, simple horse flash tattoo"; **Sketch**

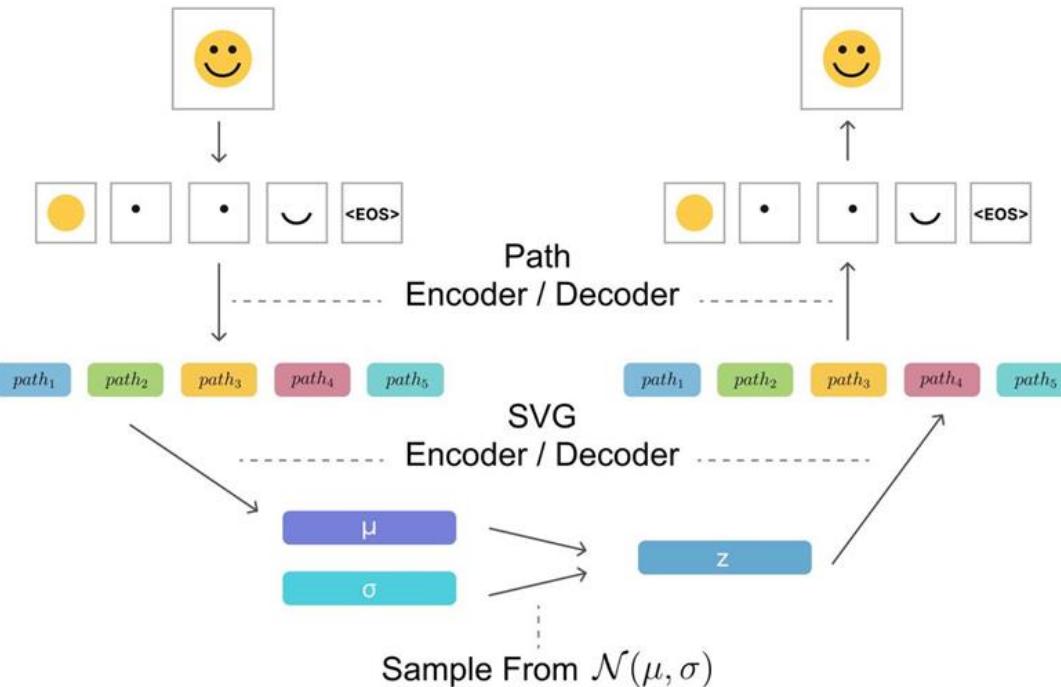


"Big Wild Goose Pagoda"; **Ink**

SVGDreamer - генерация SVG по тексту с диффузионной моделью



VectorWeaver - безусловная генерация SVG с диффузионной моделью



Архитектура: двухэтапный VAE-трансформер



Сгенерированные изображения

ChatGPT (2022)

Нестандартный способ генерации

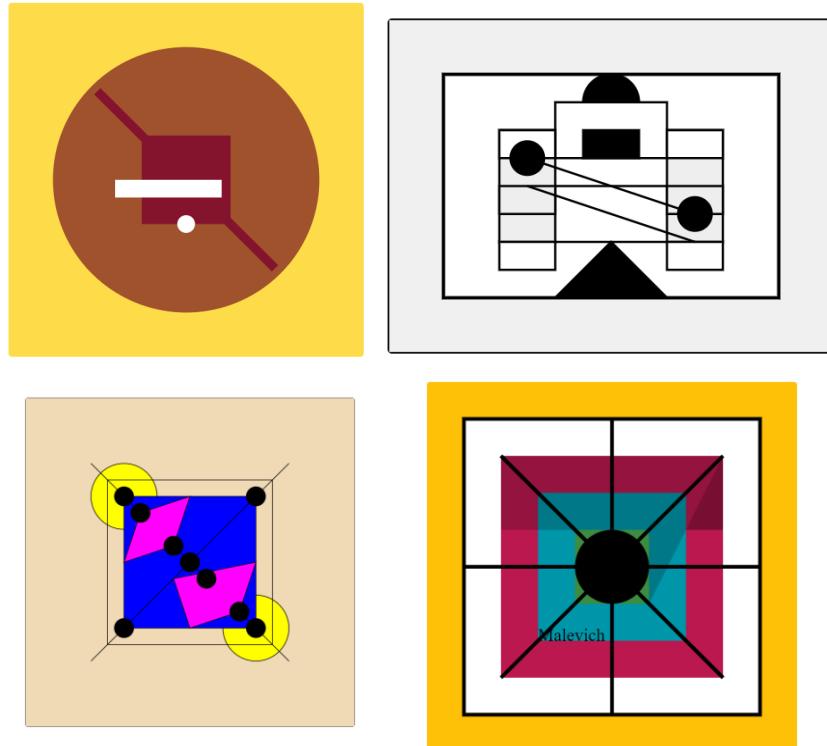
Запрос:

Using the SVG format, output a schematic reproduction of Malevich.
[I want to see a frame, objects, Malevich-like details].

Put the output in the code block. Use vibrant colors.

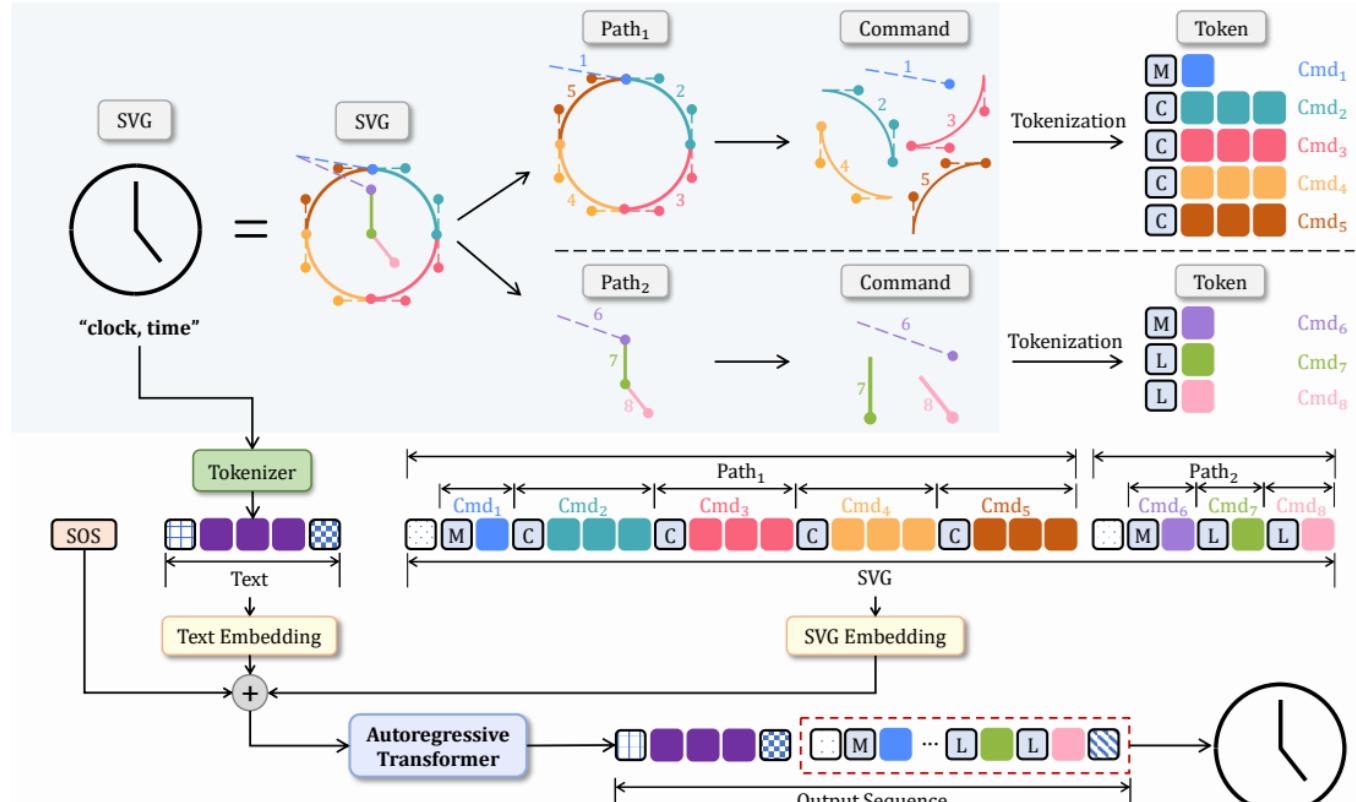
Don't forget
`xmlns="http://www.w3.org/2000/svg"` after
svg tag.

Use 20 lines of code.



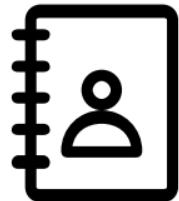
<https://neural.love/blog/chatgpt-svg>

IconShop - генерация SVG по тексту с использованием LLM



(+) Concatenate [M] Move To [L] Line To [C] Cubic Bézier [BOP] BOP [EOS] EOS [CLS] CLS [SEP] SEP [SOS] Start of Sequence

IconShop - генерация SVG по тексту с использованием LLM



phone-book,
agenda, contact



phone, device,
sun, weather



umbrella, cloud



An icon of a black
summer T-shirt.



A mop used to
clean the floor.



A calendar with
a mountain in
the middle.



kite, fly,
game, festival



rabbit, bunny,
animal, Easter



airplane, plane,
travel, flight



The night sky has a
half-moon and a star.



A bridge built
over the river.



An SVG depicts a
flask used in a
medical laboratory.



hot-air-balloon



tree, garden,
nature, ecology



car, drive, vehicle,
transportation



A bulb is producing
light.



An image of a pair of
fashion eyeglasses.



An image of an
air-conditioner.

Leveraging LLMs for SVG processing: A Review

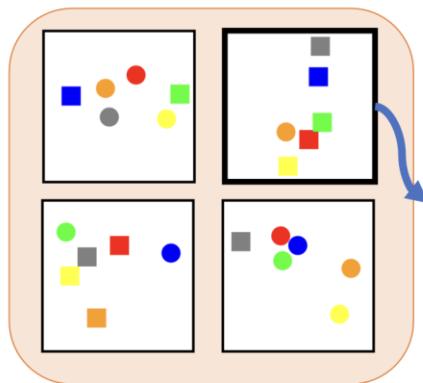
Задачи в обработке SVG

- **Генерация** (Generation; text -> svg)
 - Синтез SVG с нуля по текстовому промпту/безусловно
- **Векторизация** (Vectorization; png -> svg)
 - Синтез SVG с нуля по входному изображению
- **Редактирование** (Editing; text + svg -> svg)
 - Изменить входную SVG на основе текстового промпта
- **Анализ** (Understanding; svg -> text)
 - Выявить основной смысл/структуру из SVG (например, понять, что изображено на графике)

Бенчмарки для оценки LLM

- **Image-text bridging** – understanding
- **SVGEeditBench** – edits like rotate, crop, color change
- **SVG Taxonomy** – chart-based reasoning
- **VGBench** – both generation & understanding
- **SGP-Bench** – evaluates semantic robustness

Image-text bridging

**Unary:**

Q: What is the shape of the green object?
A: Rectangle

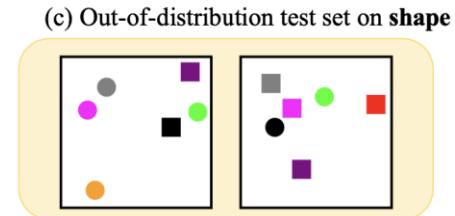
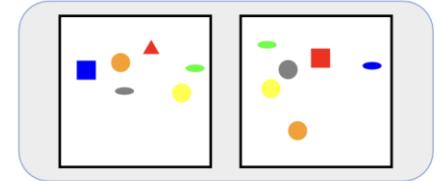
Binary:

Q: How many objects have the same shape as the red object?
A: 4

Ternary:

Q: Is there an object lies on the line connecting the blue and gray objects?
A: Yes

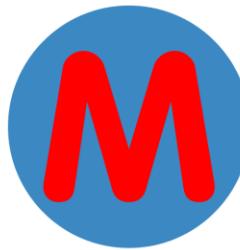
(b) Sample questions from in-distribution test set



Примеры задач на понимание

SVGEeditBench

іТМО

<p>Query: The following code is the SVG code for the emoji 'circled latin capital letter m'.</p>	<p>Task [change color]: Please generate an SVG code that changes the part of the emoji with a #FFF color to red.</p>	<p>Task [set contour]: Please generate an SVG code that draws a black line around the part of the emoji with a #FFF color.</p>	<p>Task [upside down]: Please flip this emoji upside down.</p>	<p>Task [transparency]: Please make this emoji transparent by half.</p>	<p>Task [crop to half]: Please trim the right half and keep the left half.</p>
<p>Query SVG:</p> 	<p>Ground truth:</p> 	<p>Ground truth:</p> 	<p>Ground truth:</p> 	<p>Ground truth:</p> 	<p>Ground truth:</p> 

Примеры задач на редактирование

SVG Taxonomy

іТМО

Input Text prompt:

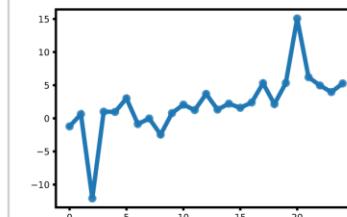
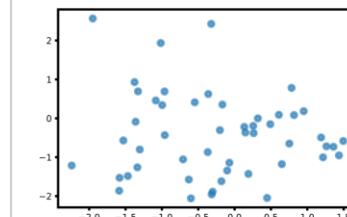
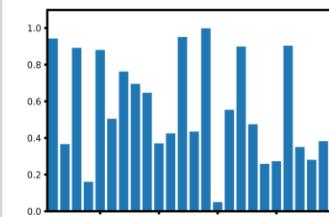
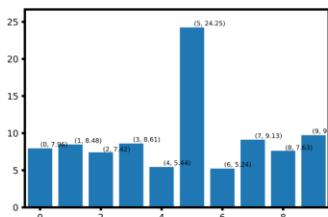
<input>: An SVG scatter plot with n points. The points are not labeled with their coordinates. The axes, title, legends, and other unnecessary elements are omitted for simplicity.

<output>: SVG code only and no other textual response

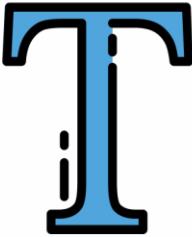
Instructions:

1. Identify the one outlier in the scatter plot, defined as data points that significantly deviate from the clusters.
2. Reconstruct the SVG scatter plot, coloring the outlier with a different color compared to the rest of the points.
3. Omit the axes, title, legends, and any other unnecessary elements in the reconstructed SVG.
4. Ensure that the reconstructed SVG contains only the data points, with the same number of points as the input SVG.
5. Include the necessary shape definitions in the reconstructed SVG code. Please provide the complete SVG code for the scatter plot with the outliers colored differently, without any additional textual response.

Plot examples:

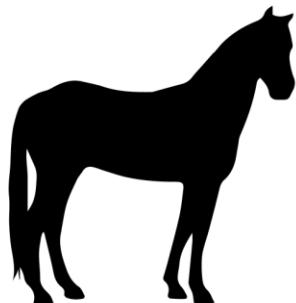


Примеры задач на понимание графиков

VGQA			VGen
Question: 	Question: 	Question: 	Caption: The image is a simple black and white silhouette of a bottle with a triangular label. The bottle has a narrow neck and a wider base, resembling a typical glass bottle shape. The label has a prominent downward-pointing arrow in the center. The silhouette is stylized and lacks detail, making it suitable for a logo or an icon representing a bottle.
What is the dominant color of the text icon in the SVG image?	What does this SVG image primarily represent?	What emotion or concept is this SVG image primarily representing?	Ground truth image: 
Answer: A. Green B. Yellow <u>C. Blue</u> D. Red	Answer: A. A mobile phone <u>B. A playing card</u> C. A book D. A photo frame	Answer: A. Freedom <u>B. Love or Romance</u> C. Migration D. Solitude	

Примеры задач на понимание и генерацию

Question:



What type of animal is represented by the object in the image?

Question:



What object is depicted in the image?

Question:



What is the person in the image doing?

Question:



What is the color of the object in the image?

Question:



How many legs does the object have?

Answer:

- A. Horse
- B. Cat
- C. Dog
- D. Elephant

Answer:

- A. Book
- B. Newspaper
- C. Notebook
- D. Magazine

Answer:

- A. Running
- B. Walking
- C. Sitting
- D. Jumping

Answer:

- A. Blue
- B. Green
- C. Yellow
- D. Pink

Answer:

- A. 4
- B. 6
- C. 10
- D. 8

Примеры задач на понимание

Сравнительная таблица

Model	Generation VGen	Editing SVGEedit	Understanding		Avg	O 	R 	C 	MoE	Model	Generation VGen	Editing SVGEedit	Understanding		Avg	O 	R 	C 	MoE
			VGQA	SGP									VGQA	SGP					
IconShop	0.816	-	-	-	-	✓				LLama3.2-1B	0.805	0.073	0.220	0.168	0.357	✓			
Codestral-2501	0.889	0.901	0.273	0.484	0.723		✓			LLama3.2-3B	0.834	0.063	0.193	0.406	0.399	✓			
Deepseek-R1	0.957	<u>0.926</u>	0.587	0.710	0.844	✓	✓			LLama3.3-70B	0.863	0.460	0.347	0.558	0.592	✓			
Deepseek-R1-LLama-70B	0.861	0.751	0.387	0.521	0.689	✓	✓			LLama4-Maverick	0.911	0.805	0.450	0.627	0.752	✓			✓
Deepseek-R1-Qwen-1.5B	∅	0.109	0.067	0.134	-	✓	✓			Mistral-NeMo-12B	0.848	0.267	0.253	0.419	0.484	✓			
Deepseek-R1-Qwen-32B	0.858	0.721	0.407	0.401	0.661	✓	✓			Mistral-Small-24B	0.866	0.722	0.187	0.491	0.642	✓			
Deepseek-v3	0.923	0.642	0.460	0.606	0.699	✓				Qwen-2.5-7B	0.865	0.081	0.367	0.475	0.456	✓			
Gemma-2-9B	0.857	0.084	0.060	0.429	0.395	✓				Qwen-2.5-72B	0.872	0.533	0.372	0.567	0.625	✓			
Gemma-3-27B-it	0.898	0.605	0.390	0.565	0.660	✓				Qwen-2.5-Coder-32B	0.890	0.545	0.333	0.532	0.623	✓			✓
gemini-2.5-flash-preview	0.940	0.899	0.680	0.459	0.803					Qwen-3-235B-A22B	0.927	0.818	0.500	0.567	0.760	✓			✓
gemini-2.5-pro-preview	<u>0.961</u>	0.882	<u>0.727</u>	<u>0.765</u>	<u>0.863</u>					Qwen-3-30B-A3B	0.897	0.833	0.380	0.500	0.723	✓			✓
GPT-4.1	0.960	0.869	0.593	0.721	0.829	✓	✓			Qwen-3-32B	0.901	0.792	0.460	0.530	0.729	✓			
GPT-4.1-mini	0.944	0.849	0.560	0.671	0.803	✓	✓			Qwen-3-8B	0.889	0.419	0.450	0.435	0.584	✓			
GPT-4.1-nano	0.925	0.591	0.293	0.516	0.640	✓	✓			Qwen-Max	0.906	0.599	0.440	0.528	0.663				
GPT-4o	0.947	0.827	0.653	0.670	0.812					Qwen-QwQ-32B	0.905	0.651	0.400	0.535	0.675	✓	✓		✓
GPT-4o-mini	0.910	0.727	0.380	0.574	0.707					Qwen-Turbo	0.858	0.574	0.296	0.477	0.606				
GPT-o4-mini	0.949	0.914	0.700	0.696	0.857	✓	✓												

Сравнение производительности различных моделей в тестах.

В последних столбцах приведены атрибуты модели:

O (open-source), R (reasoning), C (code-oriented), MoE (Mixture of Experts).

Визуальные результаты

іТМО

Input Text prompt

Iconshop

Deepseek-v3

GPT-4.1

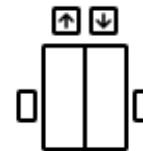
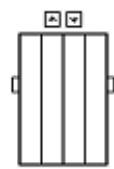
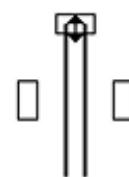
Deepseek-R1

Gemini-2.5-pro

The image shows a stylized icon of a greeting card or invitation...



The image is a simple line drawing of an elevator...

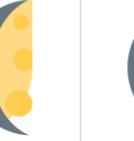
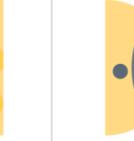
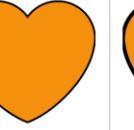
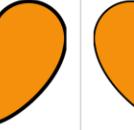
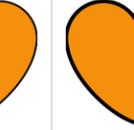


The image is a simple black and white line drawing of a character's face...



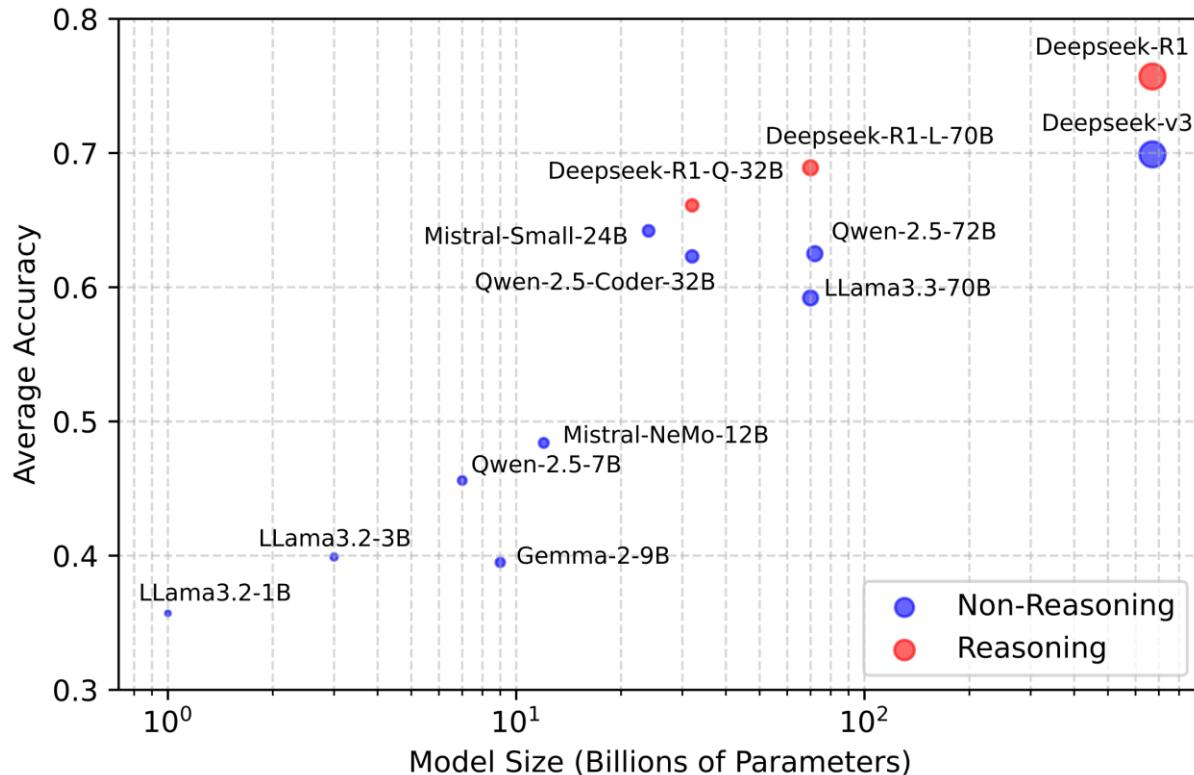
Случайно выбранные изображения, созданные самыми успешными моделями

Визуальные результаты

Input Text prompt	Ground truth	Deepseek-v3	Codestrail-2501	Deepseek-R1	Qwen-Max
The following code is the SVG code for the emoji 'waxing gibbous moon symbol'. Please trim the right half and keep the left half.					
The following code is the SVG code for the emoji 'ram'. Please make this emoji transparent by half.					
The following code is the SVG code for the emoji 'orange heart'. Please generate an SVG code that draws a black line around the part of the emoji with a #F4900C color.					
The following code is the SVG code for the emoji 'input symbol for latin small letters'. Please flip this emoji upside down.					

Сравнение разных задач из SVGEeditBench

Сравнение



Сравнение reasoning моделей (и их дистилятов) и non-reasoning моделей

Выводы

Общие проблемы:

- долгая генерация
- слишком сложные изображения, в частности из-за векторизации
- DiffVG имеет различные ограничения по использованию
- нет модели для упрощения SVG
- авторы часто черрипикуют результаты, и они плохо воспроизводимы
- нет стандарта как сравнивать качество векторных изображений, т.к. критериев очень много

Выводы

- **SVG-VAE** – VAE для генерации шрифтов (результаты не очень).
- **DeepSVG** – двухэтапный VAE для разных задач, но генерация простых шрифтов всё еще не очень.
- **DiffVG** – растеризатор векторной графики. Очень полезный оператор, но имеет свои косяки при использовании.
- **Img2Vec** – VAE векторизатор растровой графики, большие проблемы с репродуктивностью результатов. Идея: инициализация замкнутой фигуры с последующей деформацией.
- **LIVE** – векторизатор растровой графики с итеративным подходом оптимизации. Идея: постепенно добавляем новые фигуры.
- **EvoVec** – векторизатор растровой графики с итеративным эволюционным подходом
- **LIVBOC** – векторизатор растровой графики, улучшение LIVE

Выводы

- **CLIPDraw** – генерация векторного изображения по тексту. Изображения довольно абстрактные, генерация занимает много времени
- **StyleCLIPDraw** – генерация изображения по тексту и стилю
- **DeepVecFont** – Few-shot генерация векторных шрифтов. Идея: использовать информацию как из растеризованного изображения, так и из последовательности путей и команд.
- **CoverGAN** – генерация векторных музыкальных обложек. Изображения довольно абстрактные, полученную связь музыки и обложки сложно интерпретировать
- **VectorNST** – перенос стиля с векторного/растрового на векторное изображение
- **LLMs for SVG** – нестандартный подход для генерации SVG текста с помощью GPT модели

Выводы

- **VectorFusion** – генерация изображения по тексту путем векторизации + оптимизации
- **SVGDreamer** – многоступенчатая генерация по тексту с использованием векторизации и последующей оптимизации
- **VectorWeaver** – безусловная генерация изображения с помощью латентной диффузии
- **IconShop** – авторегрессионная генерация по тексту с использованием трансформер-декодера

Спасибо за внимание!

www.ifmo.ru

