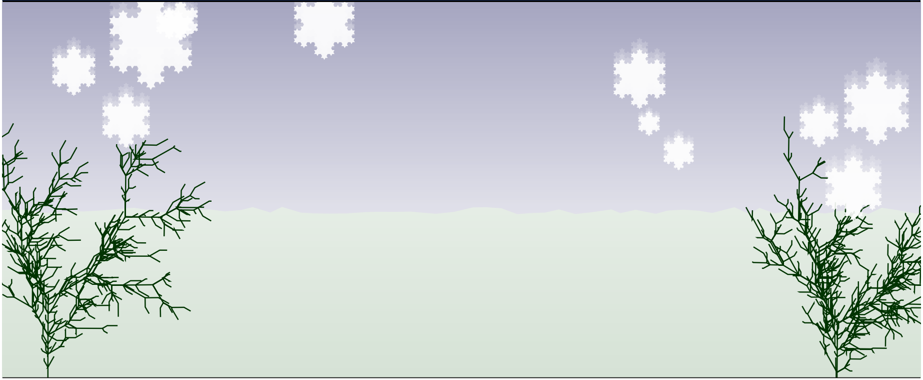
TikZero

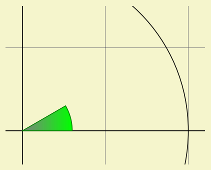
Zero-Shot Text-Guided Graphics Program Synthesis

Jonas Belouadi Eddy Ilg Margret Keuper Hideki Tanaka Masao Utiyama
Raj Dabre Steffen Eger Simone Ponzetto
University of Mannheim University of Technology Nuremberg NICT



Background

- Graphics programming languages offer advantages over low-level vector and raster image formats by representing visuals as high-level programs.



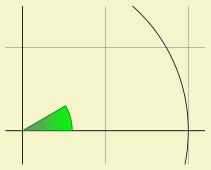
```
\begin{tikzpicture}[scale=3]
  \clip (-0.1,-0.2) rectangle (1.1,0.75);
  \draw[step=.5cm,gray,very thin] (-1.4,-1.4) grid (1.4,1.4);
  \draw (-1.5,0) -- (1.5,0);
  \draw (0,-1.5) -- (0,1.5);
  \draw (0,0) circle (1cm);
  \shadedraw[left color=gray,right color=green, draw=green!50!black]
    (0,0) -- (3mm,0mm) arc (0:30:3mm) -- cycle;
\end{tikzpicture}
```

Source: `https://tikz.dev`

UNIVERSITY OF MANNHEIM

Background

- Graphics programming languages offer advantages over low-level vector and raster image formats by representing visuals as high-level programs.
  - Preserve semantics, remain human-interpretable, and allow manual editing.

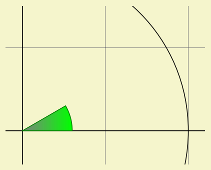


```
\begin{tikzpicture}[scale=3]
  \clip (-0.1,-0.2) rectangle (1.1,0.75);
  \draw[step=.5cm,gray,very thin] (-1.4,-1.4) grid (1.4,1.4);
  \draw (-1.5,0) -- (1.5,0);
  \draw (0,-1.5) -- (0,1.5);
  \draw (0,0) circle (1cm);
  \shadedraw[left color=gray,right color=green, draw=green!50!black]
    (0,0) -- (3mm,0mm) arc (0:30:3mm) -- cycle;
\end{tikzpicture}
```

Source: `https://tikz.dev`

Background

- Graphics programming languages offer advantages over low-level vector and raster image formats by representing visuals as high-level programs.
  - Preserve semantics, remain human-interpretable, and allow manual editing.
- These properties are valuable to the scientific research community, where specialized languages like *Ti*kZ (and many others) are popular.

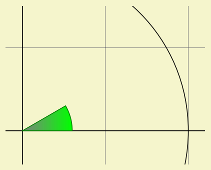


```
\begin{tikzpicture}[scale=3]
  \clip (-0.1,-0.2) rectangle (1.1,0.75);
  \draw[step=.5cm,gray,very thin] (-1.4,-1.4) grid (1.4,1.4);
  \draw (-1.5,0) -- (1.5,0);
  \draw (0,-1.5) -- (0,1.5);
  \draw (0,0) circle (1cm);
  \shadedraw[left color=gray,right color=green, draw=green!50!black]
    (0,0) -- (3mm,0mm) arc (0:30:3mm) -- cycle;
\end{tikzpicture}
```

Source: https://tikz.dev

Background

- Graphics programming languages offer advantages over low-level vector and raster image formats by representing visuals as high-level programs.
  - Preserve semantics, remain human-interpretable, and allow manual editing.
- These properties are valuable to the scientific research community, where specialized languages like TikZ (and many others) are popular.
- Graphics programming languages come with a steep learning curve motivating automated synthesis approaches (e.g., with text-guidance).

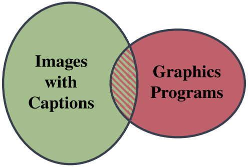


```
\begin{tikzpicture}[scale=3]
  \clip (-0.1,-0.2) rectangle (1.1,0.75);
  \draw[step=.5cm,gray,very thin] (-1.4,-1.4) grid (1.4,1.4);
  \draw (-1.5,0) -- (1.5,0);
  \draw (0,-1.5) -- (0,1.5);
  \draw (0,0) circle (1cm);
  \shadedraw[left color=gray,right color=green, draw=green!50!black]
    (0,0) -- (3mm,0mm) arc (0:30:3mm) -- cycle;
\end{tikzpicture}
```

Source: https://tikz.dev

The Problem

Although a large pool of potential training data exists for, current approaches are restricted to subsets based on input modality and training setup.

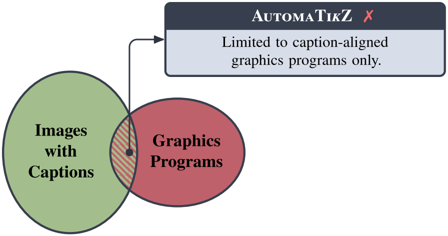


UNIVERSITY OF MANNHEIM

The Problem

Although a large pool of potential training data exists for, current approaches are restricted to subsets based on input modality and training setup.

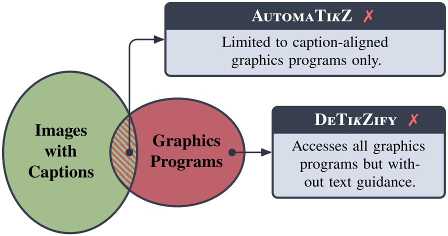
AutomaTikZ supports text-guidance but requires graphics programs paired with captions for training (scarce) resulting in limited performance.



The Problem

Although a large pool of potential training data exists for, current approaches are restricted to subsets based on input modality and training setup.

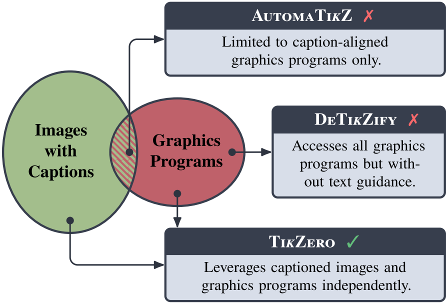
- AutomaTikZ supports text-guidance but requires graphics programs paired with captions for training (scarce) resulting in limited performance.
- DeTikZify generates programs from images (inverse graphics) with self-supervised training (can use lots of training data) but creating these visual inputs remains cumbersome



Our TikZero approach

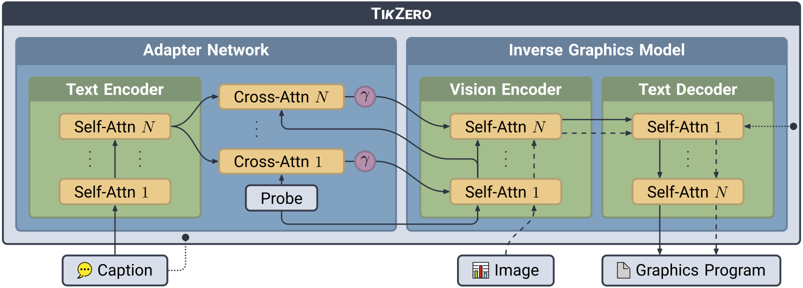
Idea

We decouple graphics program generation from text understanding, enabling independent training on graphics programs and captioned images without requiring paired data. We call our approach TikZero.



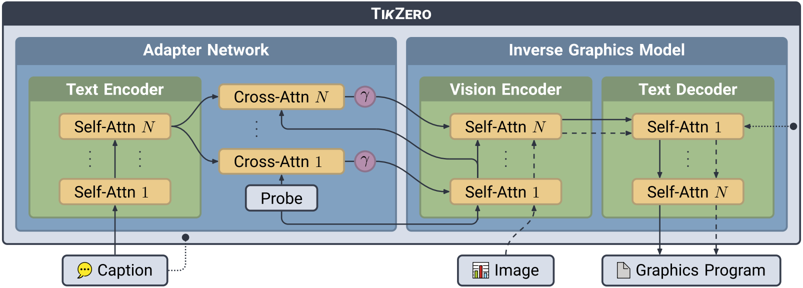
UNIVERSITY OF MANNHEIM

Architecture



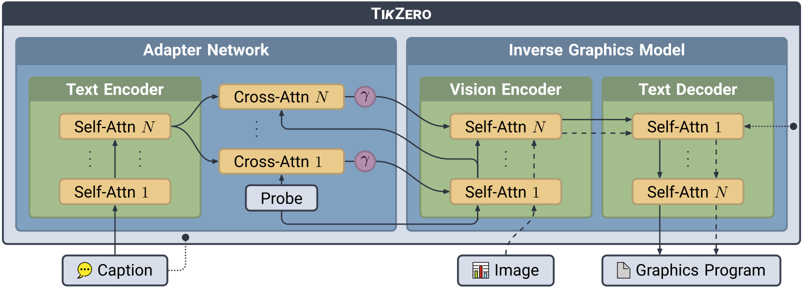
1. We train an inverse graphics model conditioned on image patch embeddings from a vision encoder (akin to DeTikZify).

Architecture



1. We train an inverse graphics model conditioned on image patch embeddings from a vision encoder (akin to DeTikZify).
2. We then train an adapter network that generates synthetic image patch embeddings from captions. This adapter training relies solely on captioned images, effectively circumventing resource limitations.

Architecture



1. We train an inverse graphics model conditioned on image patch embeddings from a vision encoder (akin to DeTikZify).
2. We then train an adapter network that generates synthetic image patch embeddings from captions. This adapter training relies solely on captioned images, effectively circumventing resource limitations.
3. Later, we combine this approach with end-to-end fine-tuning (TikZero+).

UNIVERSITY OF MANNHEIM

Training Data

Inverse Graphics Model Training Data

We systematically extract Ti*k*Z graphics programs from online sources. Whenever possible we also extract captions to support our claims. From over 450k instances, fewer than 170k include captions, underscoring the challenges discussed.

Training Data

Inverse Graphics Model Training Data

We systematically extract TikZ graphics programs from online sources. Whenever possible we also extract captions to support our claims. From over 450k instances, fewer than 170k include captions, underscoring the challenges discussed.

Adapter Training Data

We leverage the 6.4 million scientific caption-image pairs from ArxivCap for adapter training.

Automatic Evaluation

We employ the following automatic evaluation metrics for quantitative evaluation:

Automatic Evaluation

We employ the following automatic evaluation metrics for quantitative evaluation:

Image Similarity DreamSim (DSim); Kernel Inception Distance (KID)

Automatic Evaluation

We employ the following automatic evaluation metrics for quantitative evaluation:

Image Similarity DreamSim (DSim); Kernel Inception Distance (KID)

Caption Similarity CLIPScore (CLIP)

Automatic Evaluation

We employ the following automatic evaluation metrics for quantitative evaluation:

Image Similarity DreamSim (DSim); Kernel Inception Distance (KID)

Caption Similarity CLIPScore (CLIP)

Code Similarity CrystalBLEU (cBLEU); TeX Edit Distance (TED)

Automatic Evaluation

We employ the following automatic evaluation metrics for quantitative evaluation:

Image Similarity DreamSim (DSim); Kernel Inception Distance (KID)

Caption Similarity CLIPScore (CLIP)

Code Similarity CrystalBLEU (cBLEU); TEX Edit Distance (TED)

Efficiency Mean Token Efficiency (MTE)

Automatic Evaluation

We employ the following automatic evaluation metrics for quantitative evaluation:

Image Similarity DreamSim (DSim); Kernel Inception Distance (KID)

Caption Similarity CLIPScore (CLIP)

Code Similarity CrystalBLEU (cBLEU); TeX Edit Distance (TED)

Efficiency Mean Token Efficiency (MTE)

Mean Similarity average of the above (AVG)

Results

Models	DSIM↑	KID↓	CLIP↑	cBLEU↑	TED↓	MTE↑	AVG↑
IDEFICS 3 (8B)	45.475	11.426	14.327	0.656	63.175	69.558	66.628
AUTOMATIKZ (13B)	46.033	**1.294**	3.955	0.386	**62.24**	**85.866**	63.093
AUTOMATIKZ$_{v2}$ (VLM)	38.313	33.203	0.775	0.328	76.985	21.595	0.0
AUTOMATIKZ$_{v2}$ (LLM)	50.548	3.491	**15.766**	0.658	62.307	81.775	82.375
TIKZERO	**52.829**	5.103	10.051	**1.603**	65.51	82.291	**85.599**

- On average, TIKZERO outperforms AUTOMATIKZ$_{v2}$ (baseline trained on supervised subset of our data) and additional baselines.

Additional Results (TiKZero+)

Models	DSim↑	KID↓	CLIP↑	cBLEU↑	TED↓	MTE↑	AVG↑
Qwen2.5 Coder (32B)	54.473	5.493	24.87	0.285	59.856	97.269	48.593
GPT-4o	**56.464**	2.844	**31.787**	0.327	**58.511**	**97.675**	79.019
TiKZero	52.829	5.103	10.051	1.603	65.51	82.291	14.658
TiKZero+	56.295	**1.831**	24.177	**1.988**	59.008	93.058	**87.043**

- TiKZero+ outperforms TiKZero by a huge margin showing that subsequent supervised fine-tuning is beneficial.

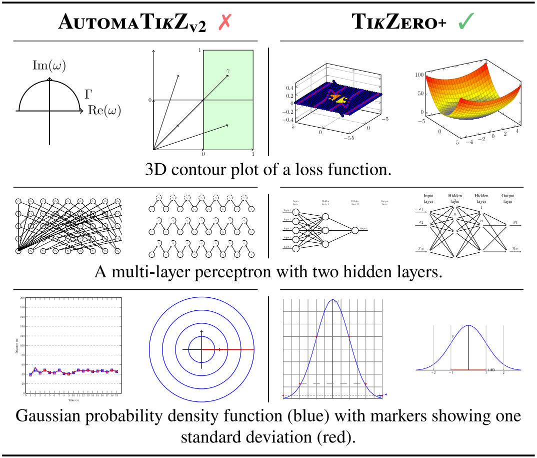
Additional Results (TikZero+)

Models	DSim↑	KID↓	CLIP↑	cBLEU↑	TED↓	MTE↑	AVG↑
Qwen$_{2.5}$ Coder (32B)	54.473	5.493	24.87	0.285	59.856	97.269	48.593
GPT-4o	**56.464**	2.844	**31.787**	0.327	**58.511**	**97.675**	79.019
TikZero	52.829	5.103	10.051	1.603	65.51	82.291	14.658
TikZero+	56.295	**1.831**	24.177	**1.988**	59.008	93.058	**87.043**

- TikZero+ outperforms TikZero by a huge margin showing that subsequent supervised fine-tuning is beneficial.
- It now also outperforms GPT-4o and other much larger and commercial models on average and comes close on key metrics DreamSim and CLIPScore.

UNIVERSITY OF MANNHEIM

Examples



Qualitative comparison of our TikZero approach (last two columns) and the end-to-end trained baseline AutomaTikZ$_{v2}$. Our method generates outputs that more closely follow the given captions.

UNIVERSITY OF MANNHEIM

Interested? There's more!

In our paper, we evaluate additional baselines, including reasoning models; uncover that end-to-end trained baselines prioritize copying strings over better visuals; and evaluate the inverse graphics performance of our model. Our code, datasets, and select models are publicly available.