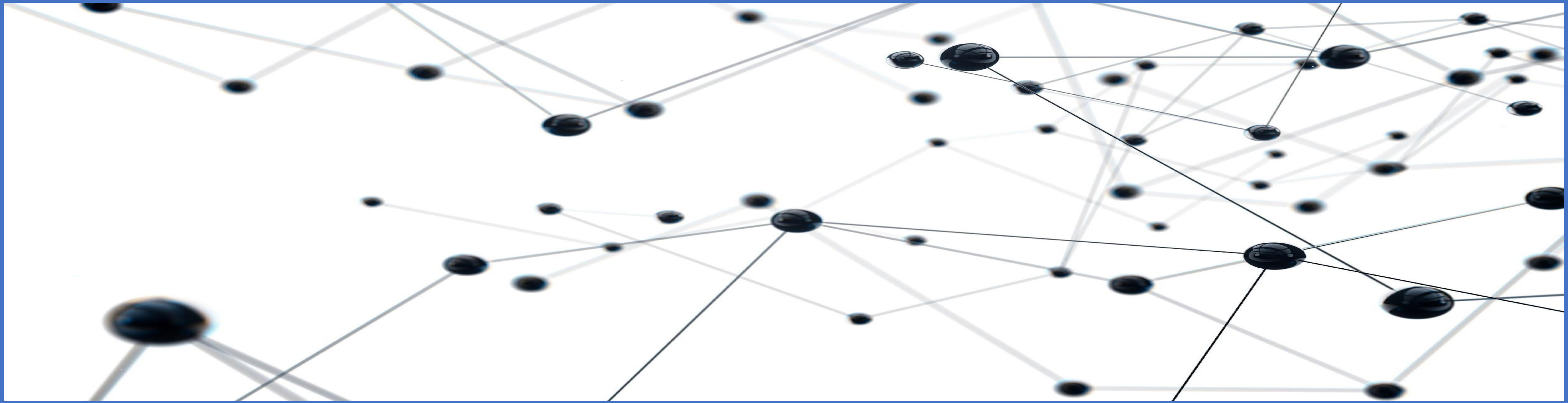
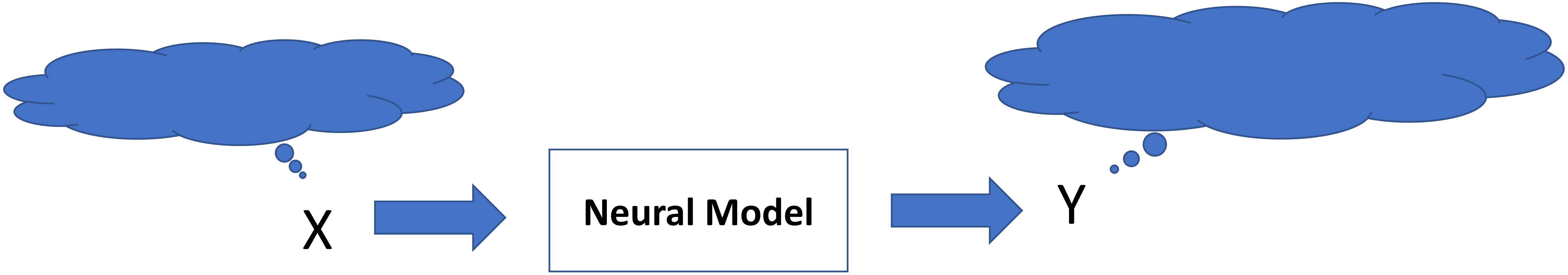


A Case for Neuro-Symbolic AI: 2 Instances of Image Generation (Vision) and Learning Generalizable Programs for Grounded Spatial Concepts (Robotics)

Parag Singla

IIT Delhi

Neural Models

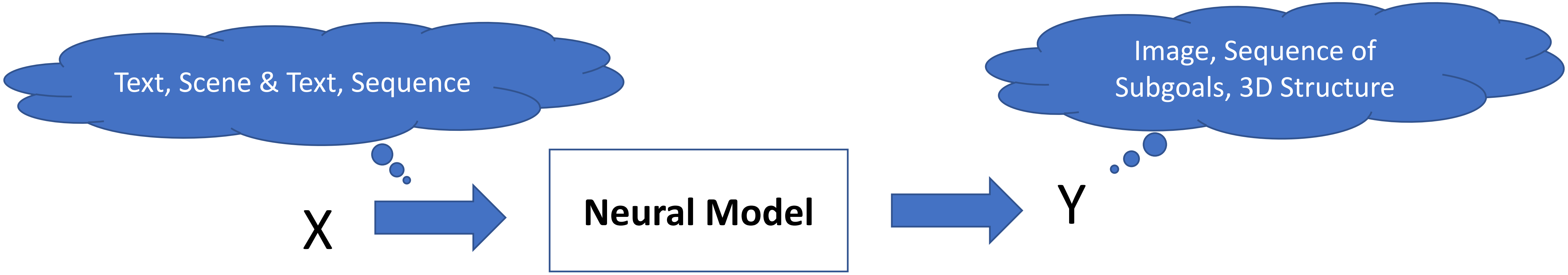


Seen as a function approximator.

$(f: X \rightarrow Y)$

Learn the model from Data: $\{X_i, Y_i\}_{i=1}^{m}$

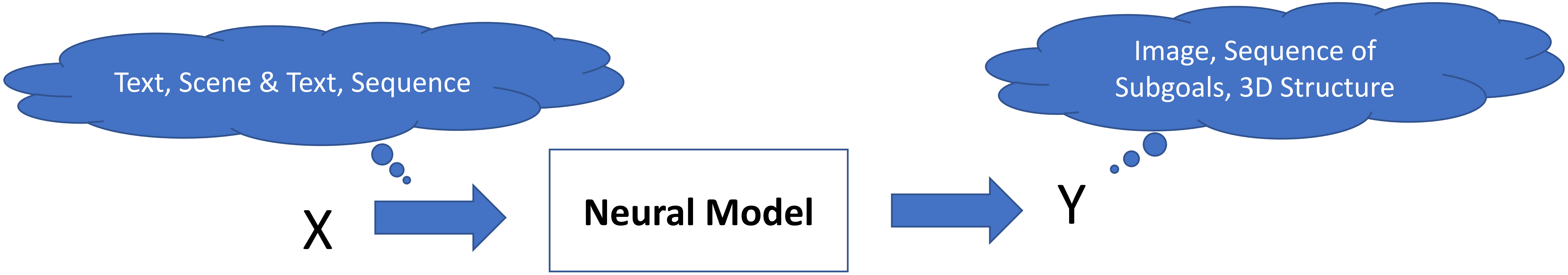
Neural Models: Input-Output can be Complex



X and Y can be quite complex:

1. Text Conditioned Image Generation (Input: Text, Output: Image)
2. Learning Generalizable Programs for Grounded Spatial Concepts (Input: Text, 3D Scene. Output: Sequence of Subgoals)
3. Protein Structure Prediction: (Input: Sequence of Amino Acids. Output: 3D Structure)

Neural Models: A Black Box Approach



Standard (Neural) Approach

Design a (black-box) neural network.
Throw in complex machinery. Lots of Data. Lots of Compute

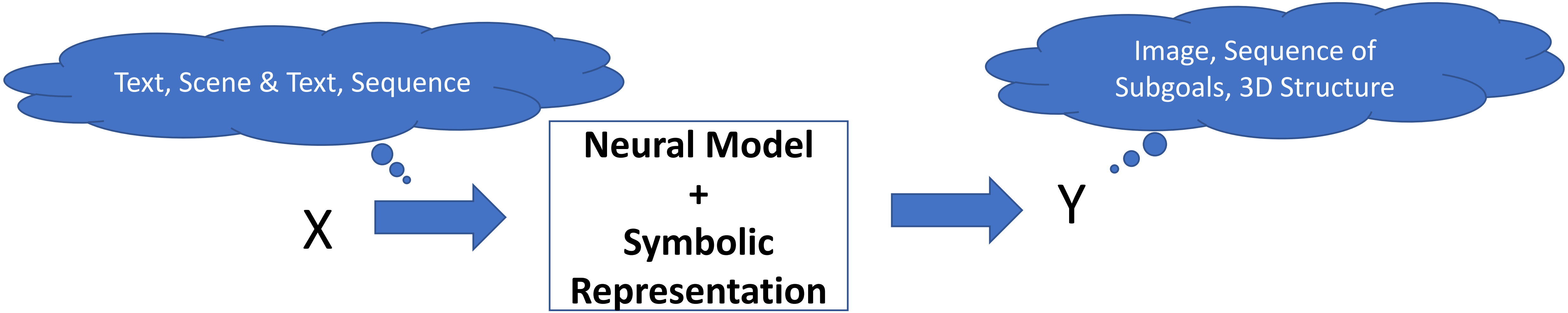
Problem?

(a) Does not encode (symbolic) information
(b) Sub-optimal predictions (does not generalize)
(c) Not Interpretable

Objective

(a) Encode (symbolic) structure explicitly
(b) Improve predictions
(c) A step towards interpretability

Neuro-Symbolic AI: Integrate with Symbolic Representations



Standard (Neural) Approach

Design a (black-box) neural network.

Throw in complex machinery. Lots of Data. Lots of Compute

What can we do differently ?

(a) Devise ways to incorporate symbolic information in underlying representation

(b) Integrate with Existing Symbolic Solvers

Interpretability falls out as a side effect

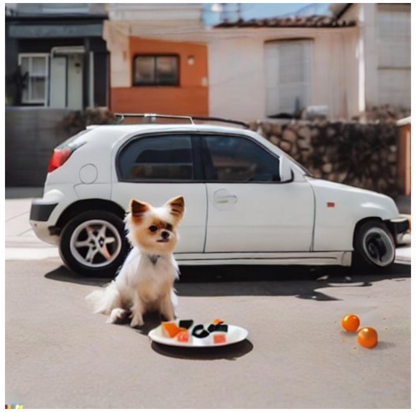
Neural Models + Symbolic Representation: 2 Instances

1. Vision: Text Conditioned Image Generation

Input:

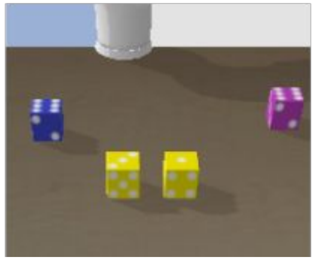
A tiny dog sitting next to a white car. Nearby, there is a plate of sushi and a few oranges scattered around

Output:



2. Robotics: Learning Generalizable Programs for Grounded Spatial Concepts

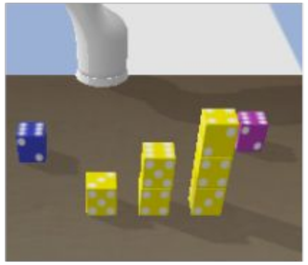
Input:



Construct a 3 step staircase using yellow blocks.

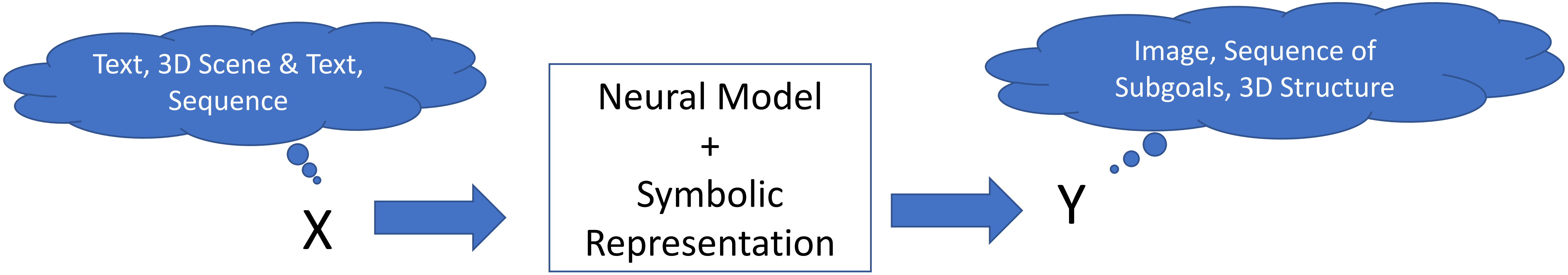
Output (truncated):

```
Place() ,
shift('top'),
Place (),
shift('top'),
Place()
```



Final scene, after executing command

Neural Models + Symbolic Representation: 2 Instances



Outline: Two of our Recent/Ongoing Works

1. Generate, Plan & Edit Framework for T2I Generation
2. Learning Generalizable Programs for Grounded Spatial Concepts

Future Directions

Outline of the Talk

- Motivation
- Text Conditioned Image Generation
- Learning Generalizable Programs for Grounded Spatial Concepts
- Other Directions

GraPE : A Generate-Plan-Edit Framework for Complex T2I-Generation

Goswami et al. [MMFM Workshop@CVPR 2025]
(In preparation for a Full Conf./Journal Submission)

Motivation

- Current SOTA image-generation models can generate photo-realistic images for diverse text-prompts



Images generated by Lumina-T2I model

Motivation [contd.]

- Current SOTA image-generation models can generate photo-realistic images for diverse text-prompts
- But ***fail at simple count, spatial and multi-object*** reasoning

A cat sitting on a sofa to the left of a dog sleeping on a couch in front of the t.v

A basket of fruits with 2 apples and 2 bananas

A man with an olive-green overcoat and blue hat walking in rain carrying a brown suitcase.

Our Approach : GraPE

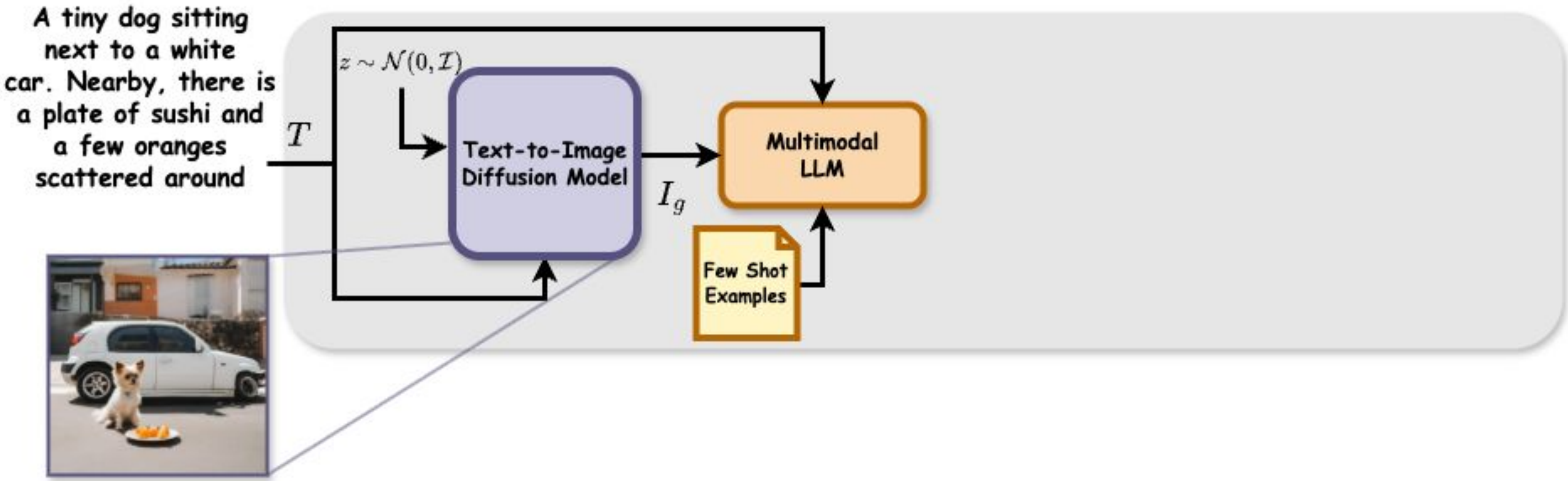
Decompose the task of complex imagex generation into three-steps:

- **Generate** : Use existing T2I model to Generate a base image for a given text prompt.
- **Plan** : Make pre-trained multi-modal language models (MLLMs) to:
  - Identify mistakes in the generated base image
  - Express corrections in terms of individual objects and their properties
  - Produce a sequence of corrective steps (edit-plan).
- **Edit** : Utilize existing (or fine-tuned) text-guided image editing models to sequentially execute the edit-plan over base image.

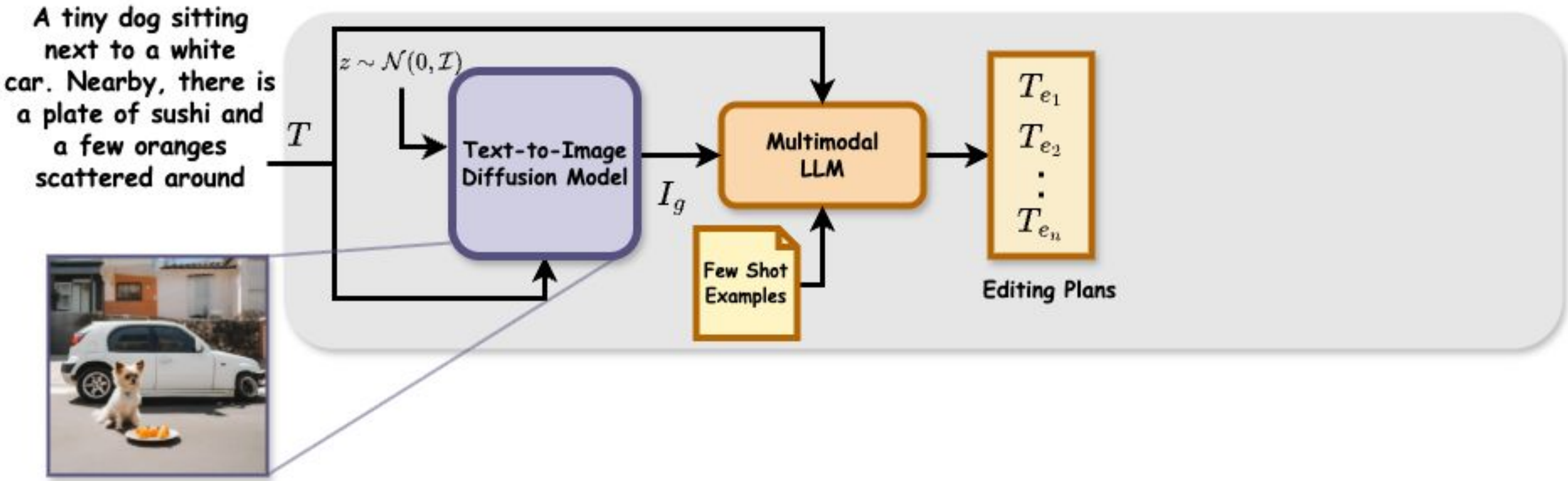
Our Approach : GraPE

Our Approach : GraPE

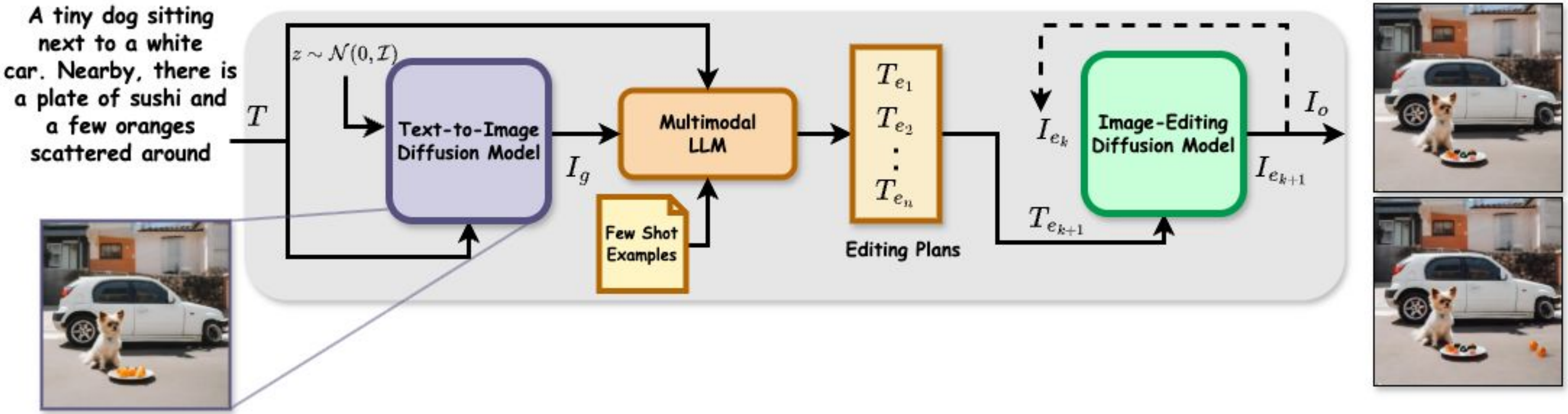
Our Approach : GraPE



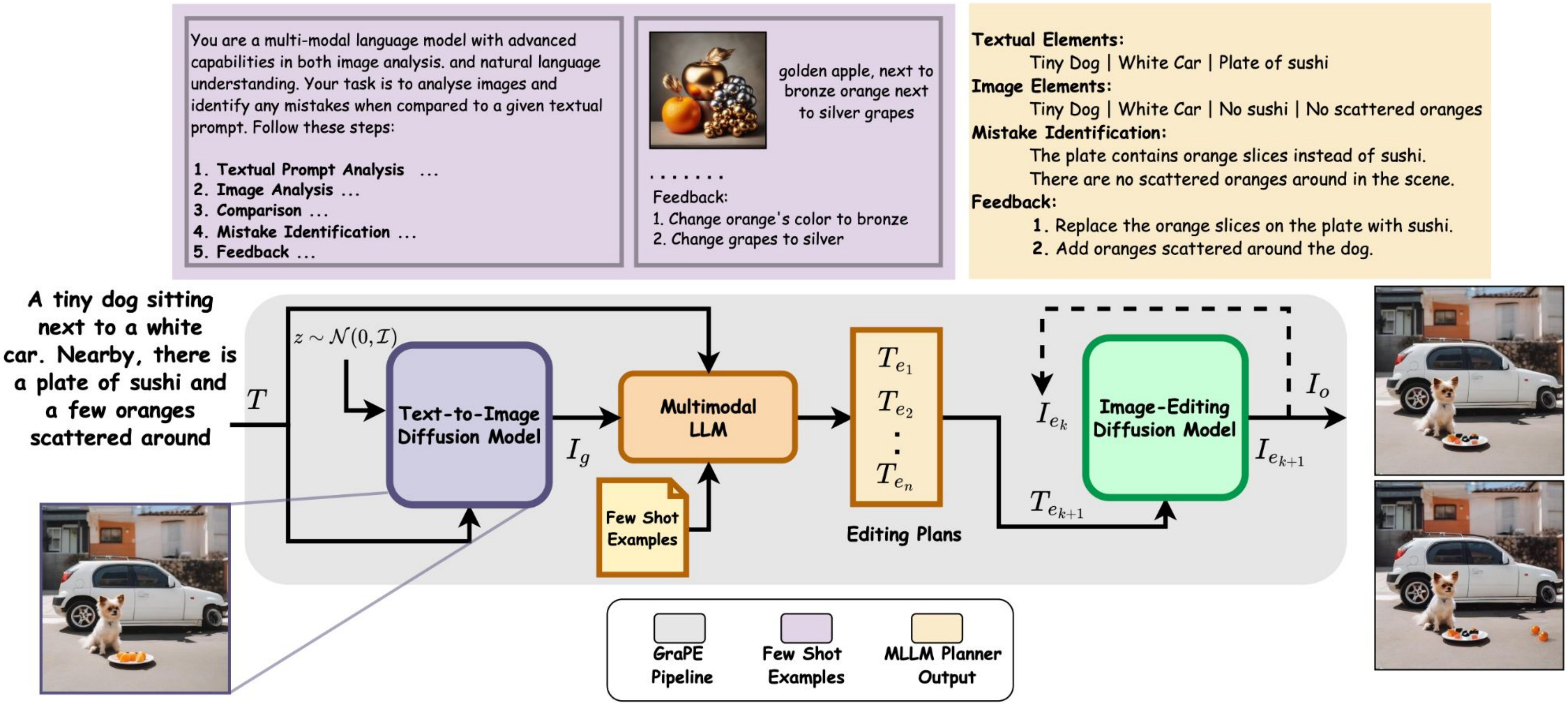
Our Approach : GraPE



Our Approach : GraPE



Our Approach : GraPE



Why GraPE works - inside the Planner

- Prompting style for multi-modal model impacts its ability to generate concrete and concise plans.
- Our Planner has 4 steps:
    - ***Analysing Textual Elements*** : Extract high level object-attribute pairs from text-prompt.
    - ***Analysing Image Elements*** *:* Generate an object-centric description of image as a function of entities present
    - ***Error Identification*** : Express mistakes in terms of the extracted entities
    - ***Feedback Generation*** *:* Generate actionable feedback in the form of image-editing instructions
- The structured framework not only enhances the performance but also makes the entire approach more interpretable

Why GraPE works - inside the Editor

- Suboptimal performance of Image Editing Models
  - Attributed to the CLIP encoder used as text-encoder and it’s known issues in handling compositionality [Yuksekgonul et al. ICLR’23]
- Trained an image-editing model based on Pixart-alpha (T2I) model
  - Uses an enhanced text-encoder (T5) [Raffel et al. JMLR’20]
  - Use high quality high-quality object and reasoning centric data for training
  - Our Proposed Model is called PixEdit.

Benchmark and Evaluation Metric

Approach evaluated using a variety of benchmarks :

- **T2I-Compbench [Huang et al. NeurIPS'23]** :
  - Well established benchmark to evaluate compositional capabilities
  - Considers categories such as color, shape, texture, spatial, non-spatial and complex prompts.
- **ConceptMix [We et al. NeurIPS'24]** :
  - Provides controllable compositionality levels.
  - Evaluate using $K \in [1, 3, 5, 7]$
  - Each prompt includes at least one object paired with K additional visual concepts.
- **FlickrBench :**
  - Sampled from existing flickr-30k [Plummer et al. ICCV'15] dataset.
  - The human generated prompts offer a well-balanced mix of compositionality and realism.

Evaluation Metric : Based on Visual-Question Answering (VQA) Framework

- QA pairs generated by LLMs based on text-prompt and images are evaluated by MLLMs like GPT-4o.
- We follow the complete framework proposed in DSG [Cho et al. ICLR'24]

Related Works

- Self-Correcting LLM-controlled Diffusion Models (SLD) [Wu et al. CVPR’24]
    - A framework to perform self-correction on generated images.
    - Operates on layout-space generated using LLMs or object-detectors.
    - Struggles with generation and control of complex layouts

- ReflectionFlow [Zhuo et al. ICCV’25]
    - Trains an open-source Qwen-7B MLLM to generate textual-feedback
    - Uses the feedback to perform correction using fine-tuned T2I model

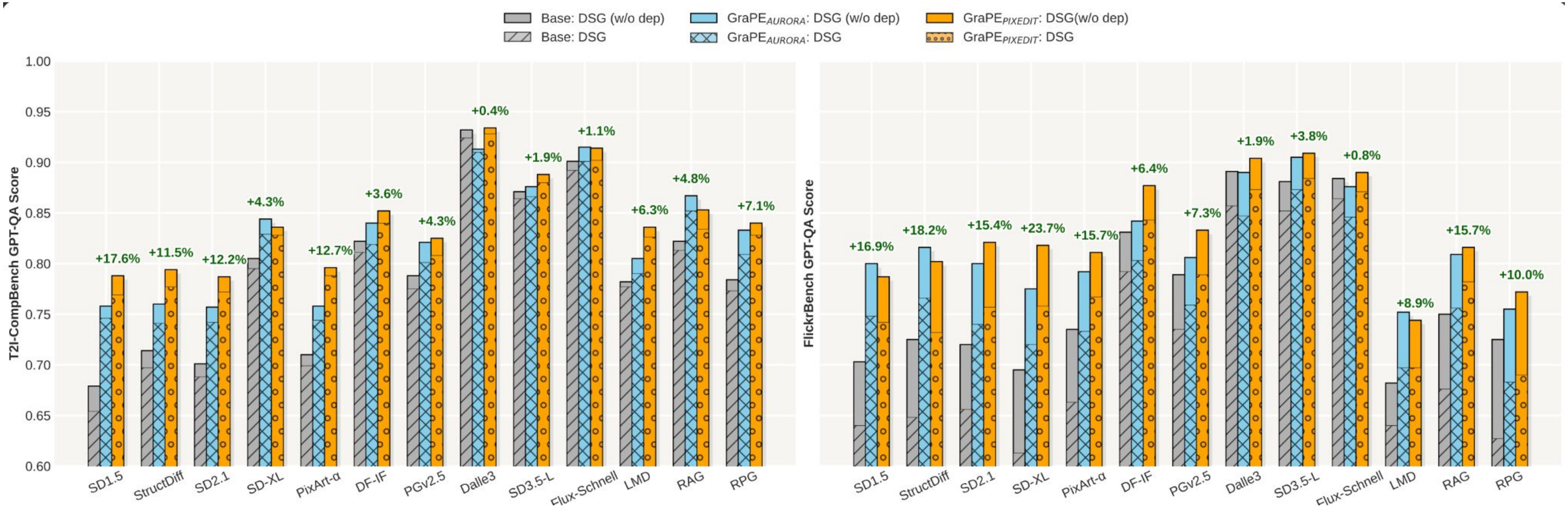
Extensive comparison against these methods : GraPE consistently outperforms across benchmarks and conditioning text.

How good does it perform? [contd.]

Method	Concept K=1		Concept K=3		Concept K=5		Concept K=7	
	Base	+$\text{GraPE}_{\text{PixEdit}}$	Base	+$\text{GraPE}_{\text{PixEdit}}$	Base	+$\text{GraPE}_{\text{PixEdit}}$	Base	+$\text{GraPE}_{\text{PixEdit}}$
Stable-Diffusion v1.5 (Rombach et al. 2022)	0.808 $\pm$0.009	**0.892** $\pm$0.002	0.606 $\pm$0.018	**0.752** $\pm$0.002	0.497 $\pm$0.010	**0.689** $\pm$0.002	0.450 $\pm$0.005	**0.610** $\pm$0.005
Structure Diffusion (Feng et al. 2023a)	0.823 $\pm$0.002	**0.885** $\pm$0.004	0.606 $\pm$0.002	**0.723** $\pm$0.005	0.542 $\pm$0.014	**0.703** $\pm$0.006	0.447 $\pm$0.001	**0.571** $\pm$0.002
Stable Diffusion v2.1 (Rombach et al. 2022)	0.833 $\pm$0.002	**0.868** $\pm$0.005	0.639 $\pm$0.014	**0.737** $\pm$0.002	0.579 $\pm$0.012	**0.687** $\pm$0.005	0.466 $\pm$0.002	**0.626** $\pm$0.002
SD-XL (Podell et al. 2024)	0.848 $\pm$0.010	**0.877** $\pm$0.005	0.708 $\pm$0.018	**0.780** $\pm$0.007	0.635 $\pm$0.014	**0.729** $\pm$0.004	0.520 $\pm$0.003	**0.628** $\pm$0.002
PixArt-α (Chen et al. 2024b)	0.813 $\pm$0.010	**0.872** $\pm$0.010	0.668 $\pm$0.011	**0.722** $\pm$0.002	0.649 $\pm$0.011	**0.742** $\pm$0.002	0.507 $\pm$0.001	**0.625** $\pm$0.002
DeepFloyd IF (at StabilityAI 2023)	0.883 $\pm$0.009	**0.915** $\pm$0.000	0.680 $\pm$0.016	**0.765** $\pm$0.007	0.663 $\pm$0.014	**0.745** $\pm$0.002	0.583 $\pm$0.002	**0.662** $\pm$0.006
PlaygroundV2.5 (Li et al. 2024)	0.908 $\pm$0.010	**0.955** $\pm$0.004	0.737 $\pm$0.023	**0.792** $\pm$0.009	0.658 $\pm$0.015	**0.721** $\pm$0.002	0.540 $\pm$0.003	**0.640** $\pm$0.005
Dalle3 (Betker et al. 2023)	0.947 $\pm$0.002	**0.953** $\pm$0.002	0.832 $\pm$0.012	**0.861** $\pm$0.003	0.812 $\pm$0.014	**0.832** $\pm$0.004	0.728 $\pm$0.006	**0.737** $\pm$0.004
Stable Diffusion v3.5 Large (stability.ai 2024)	0.927 $\pm$0.005	**0.948** $\pm$0.002	0.815 $\pm$0.002	**0.817** $\pm$0.002	0.803 $\pm$0.003	**0.831** $\pm$0.004	0.759 $\pm$0.004	**0.784** $\pm$0.005
Flux-schnell (Labs 2024)	0.902 $\pm$0.002	**0.918** $\pm$0.002	0.820 $\pm$0.003	**0.864** $\pm$0.001	0.786 $\pm$0.005	**0.804** $\pm$0.008	0.775 $\pm$0.004	**0.779** $\pm$0.004
LMD (Lian et al. 2023)	0.855 $\pm$0.004	**0.873** $\pm$0.002	0.711 $\pm$0.008	**0.773** $\pm$0.006	0.643 $\pm$0.011	**0.725** $\pm$0.009	0.591 $\pm$0.002	**0.668** $\pm$0.009
RAG (Chen et al. 2024c)	0.815 $\pm$0.009	**0.866** $\pm$0.003	0.668 $\pm$0.007	**0.718** $\pm$0.005	0.665 $\pm$0.002	**0.721** $\pm$0.004	0.520 $\pm$0.003	**0.628** $\pm$0.001
RPG (Yang et al. 2024)	0.696 $\pm$0.015	**0.845** $\pm$0.008	0.694 $\pm$0.002	**0.715** $\pm$0.007	0.583 $\pm$0.002	**0.688** $\pm$0.005	0.388 $\pm$0.007	**0.467** $\pm$0.005

Table 1: Results on Concept-mix benchmark $\text{GraPE}_{\text{PixEdit}}$.

How good does it perform?



Results of GraPE when applied over images generated by various T2I models on T2I-Compbench and FlickrBench

Another Perspective via Test-Time Scaling

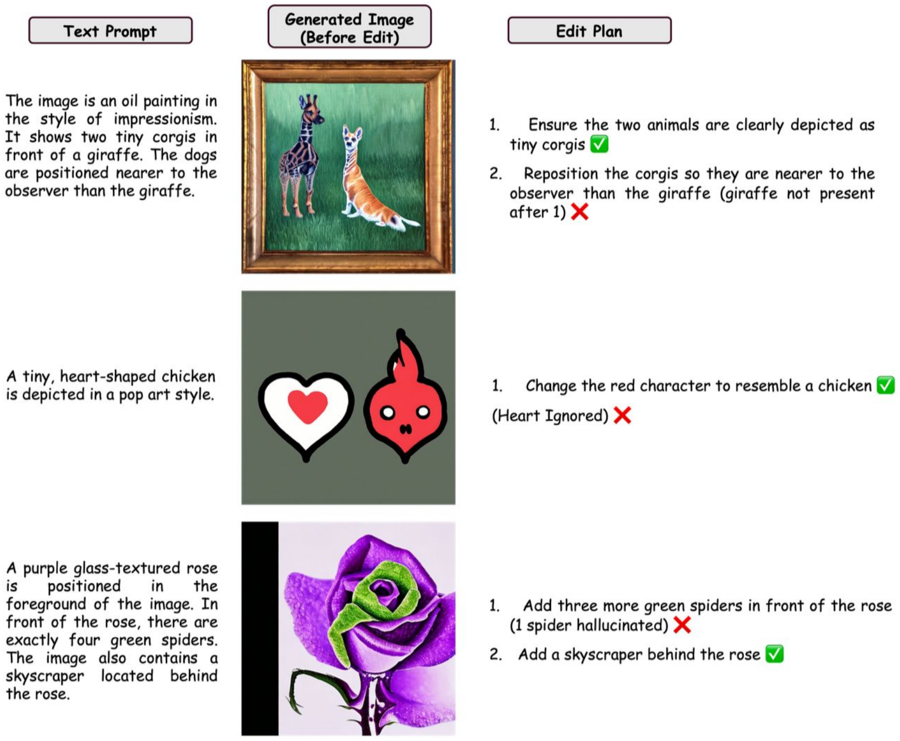
- Test-Time Scaling (TTS): Scaling is guided by the mistakes identified by the planner, and then using the editing model to correct detected errors.

- Two views of GraPE in this context:
    - ***Pure :*** Comparing GraPE directly against the TTS methods while being considerably faster.
    - ***Hybrid :*** Combine GraPE with existing TTS methods where at each editing step we use existing verifier to select best possible candidate among N images.

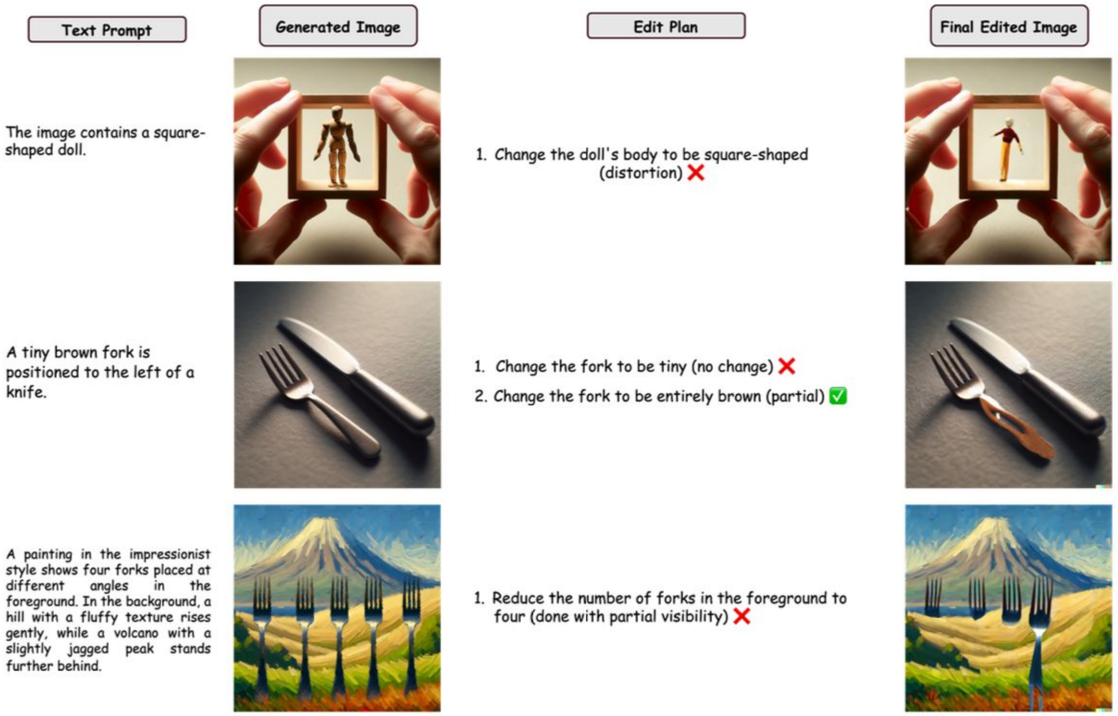
Method	N_c	Accuracy)		Time (sec.)
		K=5	K=7	
Flux-sch.	8	0.799	0.750	28.8
Flux-sch.	16	0.803	0.759	61.2
Flux-sch.+ GraPE	1	**0.804**	**0.779**	**25.2**
Flux-sch.	32	0.819	0.785	126.0
Flux-sch.+ GraPE(*hybrid*)	32*	**0.822**	**0.814**	**104.4**
SD v3.5L	8	0.811	0.765	133.2
SD v3.5L	16	0.819	0.775	270.0
SD v3.5L+ GraPE	1	**0.831**	**0.784**	**25.5**
SD v3.5L	32	0.833	0.784	543.6
SD v3.5L+ GraPE(*hybrid*)	32*	**0.850**	**0.799**	**82.8**

Table 2: Comparison of GraPE with SOTA T2I models in pure/hybrid TTS. Reported time is sec./sample and averaged over 100 samples. N_c in hybrid-TTS refers to the total scaling budget s.

Failure Cases : Visual Samples



Errors caused by incorrect Planning



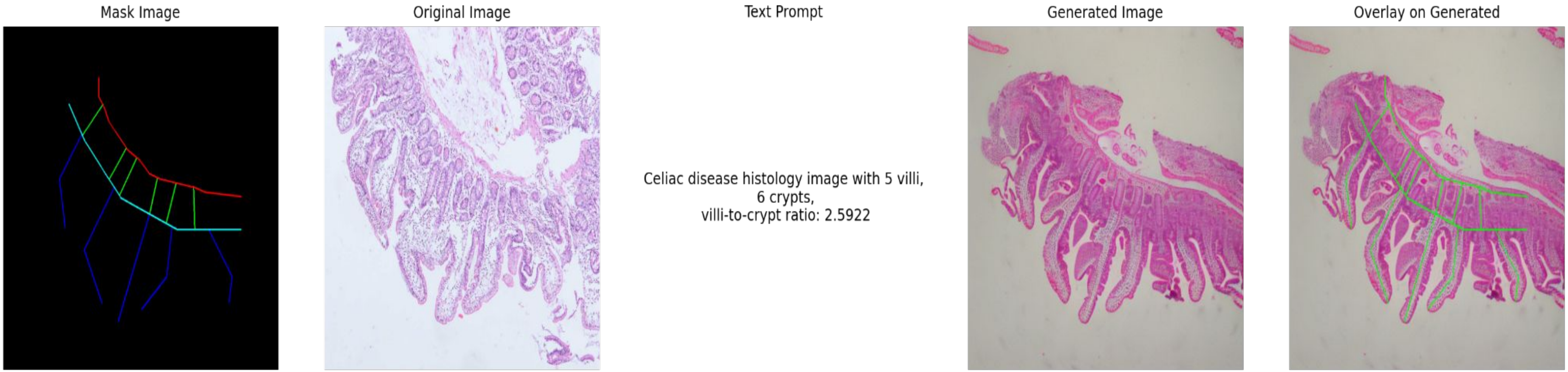
Editing model failure

Generation Biological Data

Work being done by other colleagues in the group

- Controlled Data Generation
  - For medical images annotated data is limited
  - Can you generate medical images, where which follows annotation pattern

For example: Histopathology Images for duodenum biopsy using for identification of Celiac Disease



Tyagi et.al. : Controlled Histopathology Image generation for Celiac Disease

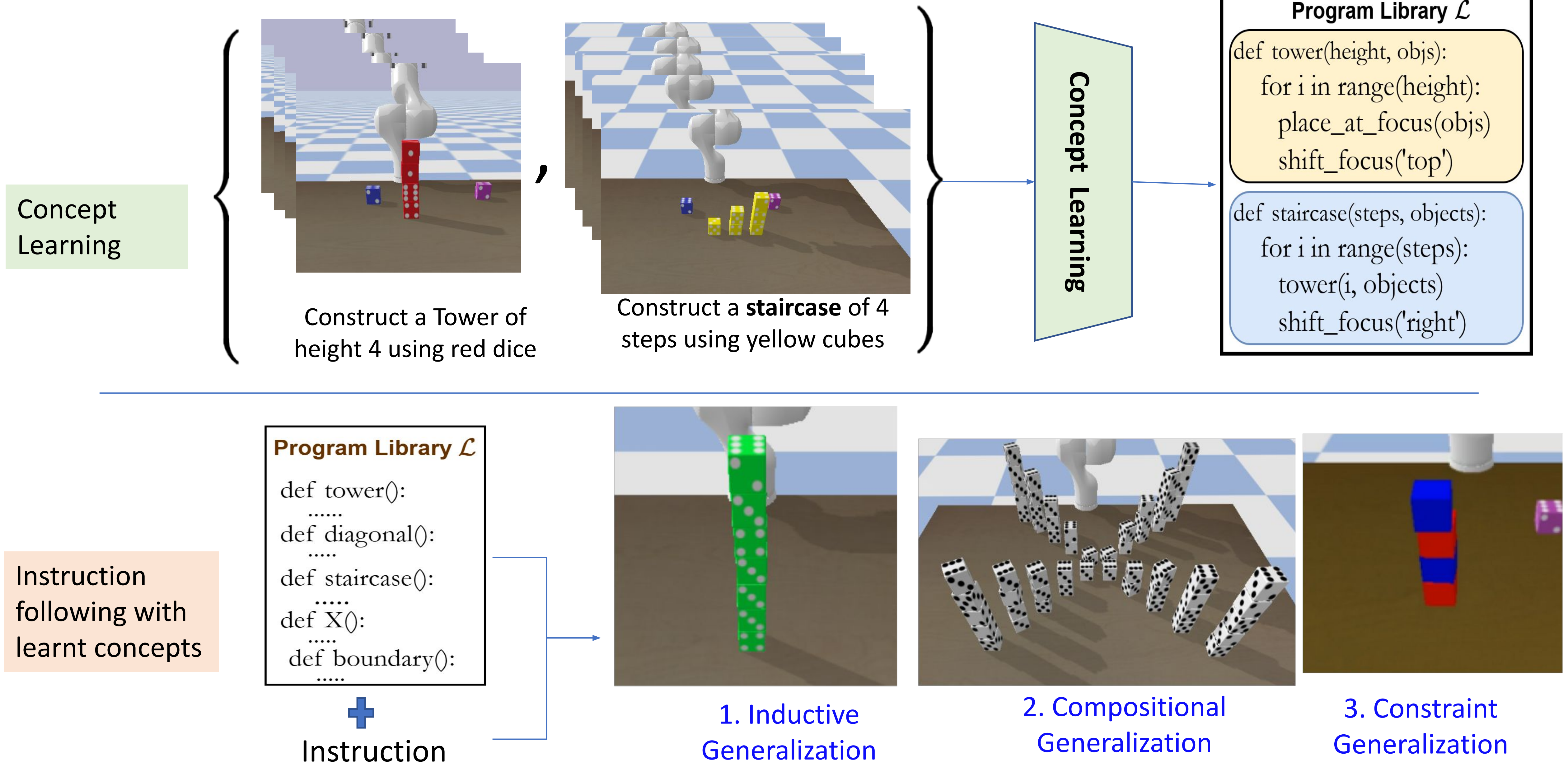
Outline of the Talk

- Motivation
- Text Conditioned Image Generation
- Learning Generalizable Programs for Grounded Spatial Concepts
- Other Directions

Learning Generalizable Programs for Grounded Spatial Concepts using MCTS for Plan Discovery & Lifting via LLMs

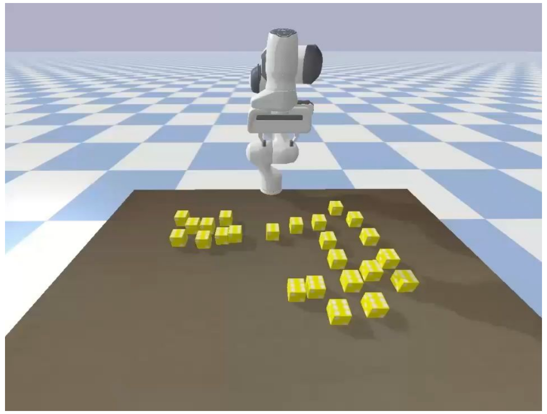
Namasivayam et al. [ICML Workshop 2025]
(In Submission for a full Conference Publication)

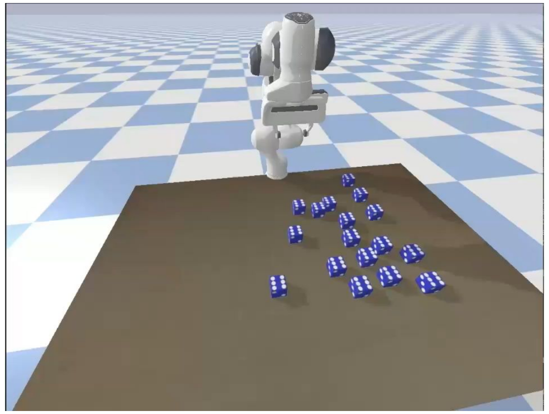
Problem Statement



Motivation

- **Spatial abstractions** are pervasive in human-robot communication
  - Rows, columns, spatial assemblies.





- Aim: How to enable the robot to understand:
  - “Robot, build a tower of size 5 with blue blocks”
  - ”Create a row with alternating blocks”
  - “Construct X whose diagonals are replaced with staircases”

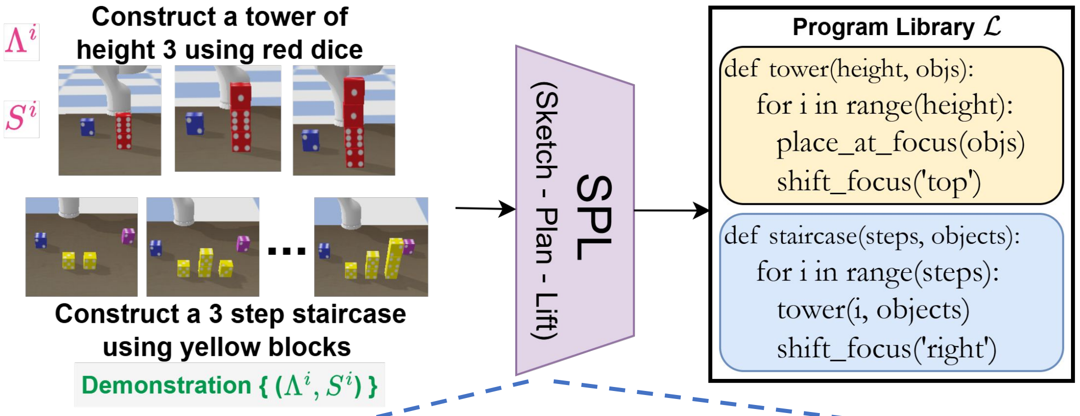
- **Technical Challenges**
  - Directly converting instructions to programs does not scale with complex reasoning.
  - Pure Learning from Demonstrations approaches learn fixed concepts.

This work fills the following technical Gap:

1. How to learn a program representation for concepts from simple demonstrations as a library.
2. How to adapt them online to perform complex tasks.

Technical Approach : Concept Learning

Aim: Learn programmatic representation for concepts
- Factored as Sketch - Plan - Lift



Sketch (LLM)

$\Lambda^i \longrightarrow$ Tower (height = 3, filter(red,dice))

Prompt (Sketch)

#You are a Language Reasoner to parse the instructions into a sequence of function calls. The output should be a sequence of function calls written in a single line, separated by semicolons.",

"# importing available functions",
"from visual_operators import filter",
"from primitives import assign_focus, shift_focus, place_at_focus

"#function signature of the imported functions",
"filter(color, cube) #filter objects that are cubes and color",
"assign_focus(at_obj_loc) #assigns the focus to the location of the object",
"shift_focus(dir) #moves the focus in the given dir",
"place_at_focus(obj) #keeps the object obj at the head",

"#Examples:",
"#Instruction: Move the green block to the left of the red dice"
"assign_focus(at_obj_loc = filter(red, dice)); shift_focus(left); place_at_focus(obj = filter(green, cube))",

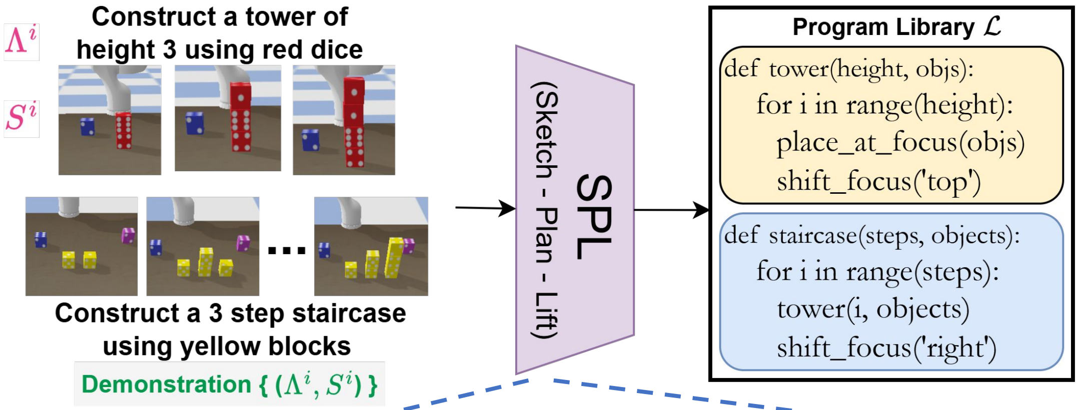
"#Instruction: Move the green block to the left of the red dice and the yellow block to the top of the green block",
"assign_focus(at_obj_loc = filter(red, dice)); shift_focus(left); place_at_focus (obj = filter(green, cube)); assign_focus(at_obj_loc = filter(green, cube)); shift_focus(top); place_at_focus(obj = filter(yellow, cube))",

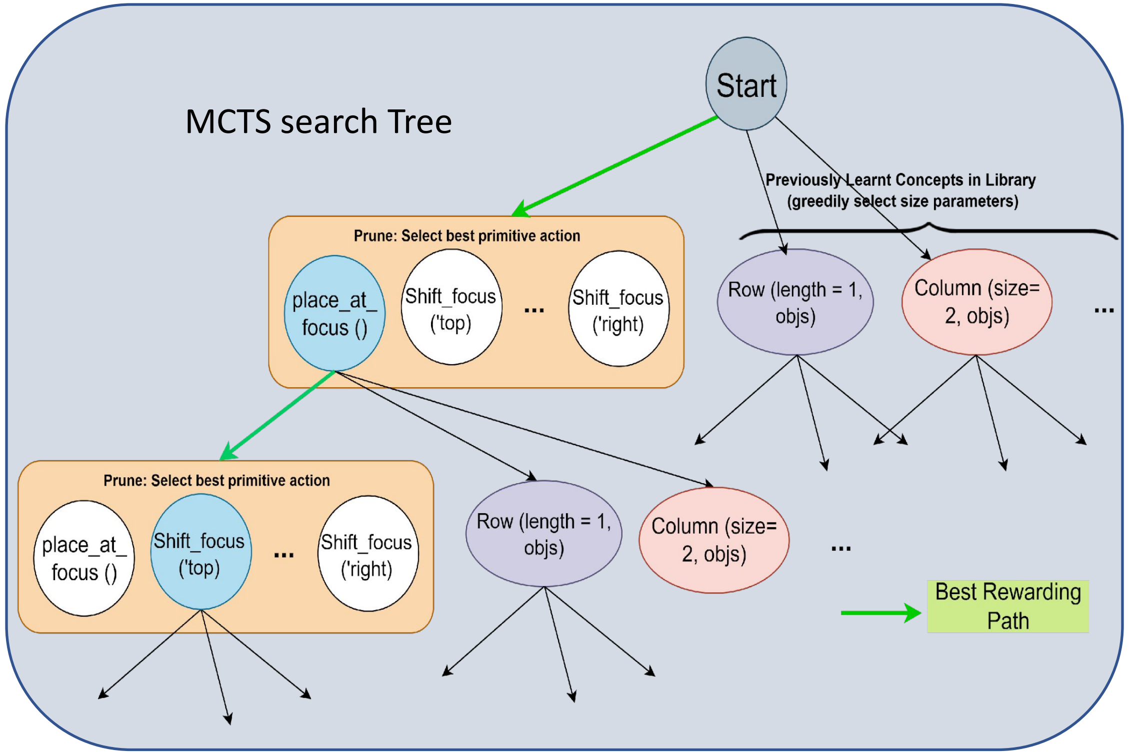
"#Instruction: Construct a Row of length 4 with the yellow cubes",
"Row(length = 4, objects = filter(yellow, cube))"

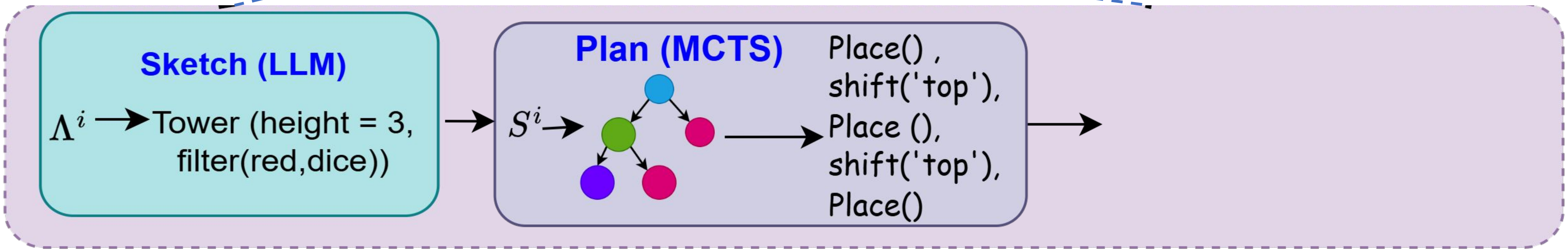
Instruction: Construct a tower of height 3 using red dice
???

Technical Approach : Concept Learning

Aim: Learn programmatic representation for concepts
- Factored as Sketch - Plan - Lift



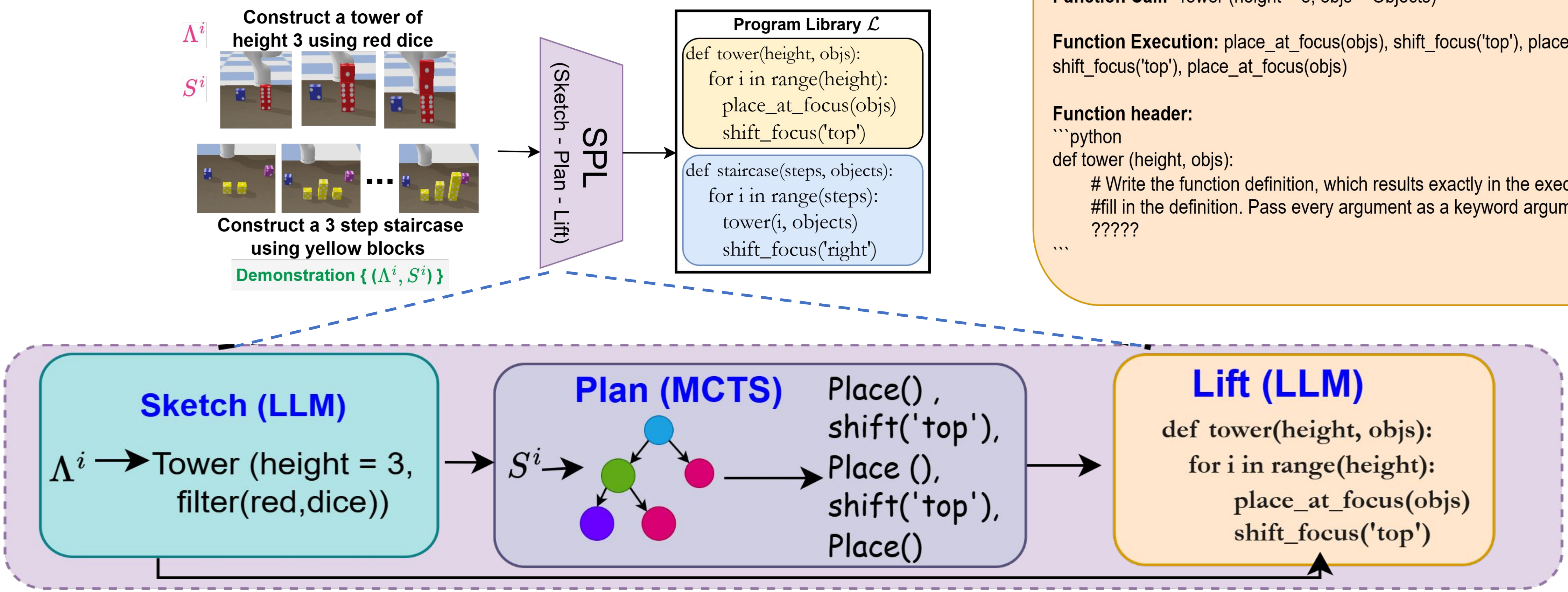




Technical Approach : Concept Learning

Aim: Learn programmatic representation for concepts

- Factored as Sketch - Plan - Lift



Prompt
(Lift)

You are a Code writer who takes instances of function execution and reconstructs the function.

Function Call: Tower (height = 3, objs = Objects)

Function Execution: place_at_focus(objs), shift_focus('top'), place_at_focus(objs), shift_focus('top'), place_at_focus(objs)

Function header:

````
```python
def tower (height, objs):
    # Write the function definition, which results exactly in the execution steps provided.
    #fill in the definition. Pass every argument as a keyword argument
    ?????
```
````

Complex Instruction Following using Learnt concepts

Prompt
(Inference)

You are a code writer which write Python code for a given language instruction. You are given a library of functions. Now, given a new instruction, you need to compose or combine the functions from the library to write code that satisfies the given instruction. Clearly understand the function semantics in the library and write your code. Modify the function in the library appropriately if needed.

You should:

1) Make sure you don't keep two blocks in the same position.

2) Head is passed by value. So after constructing a structure, the head will automatically reset to its original position by default.

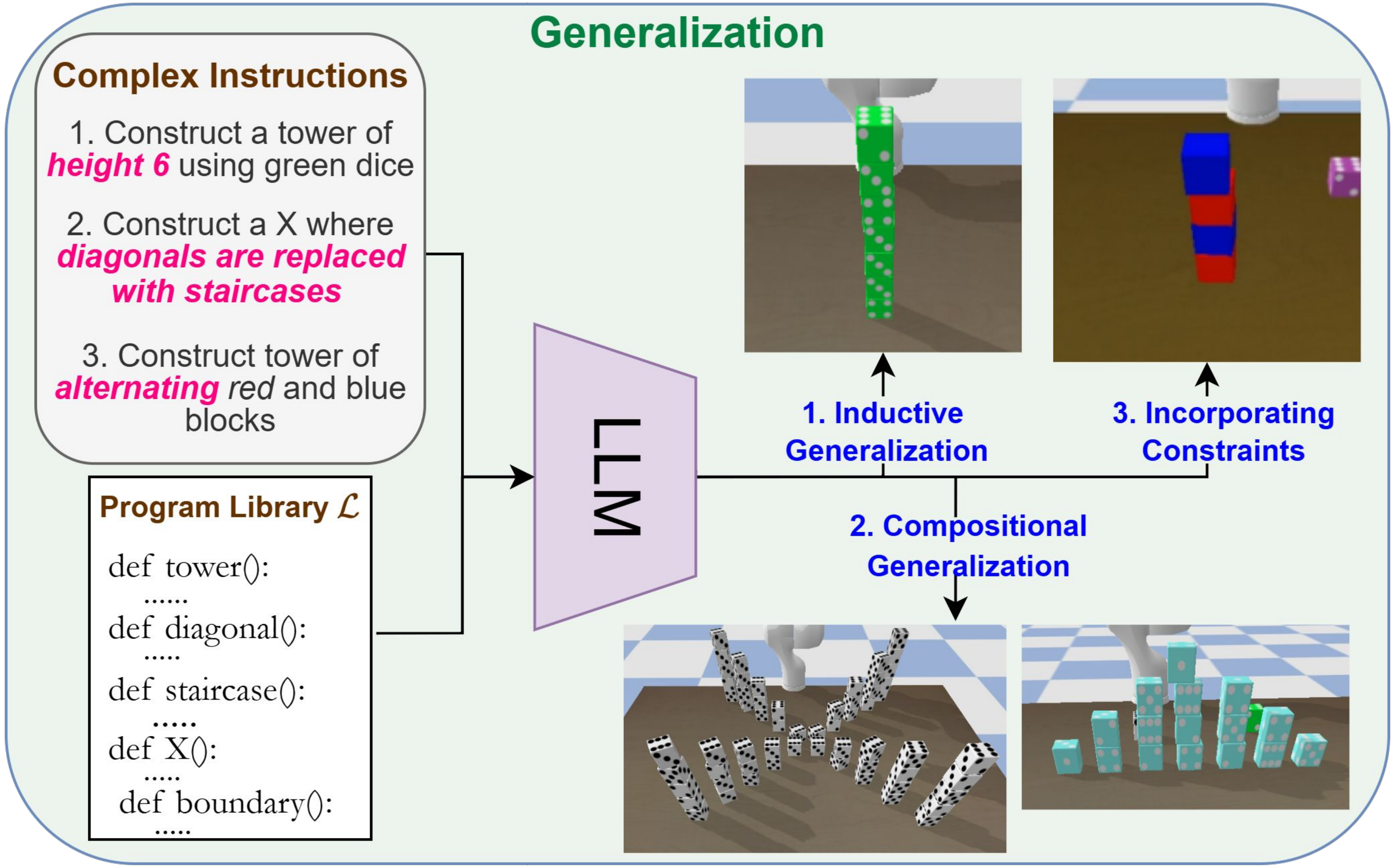
Output only the code within ``` ```

```
``` Library

def row( length, objs):    def tower (height, objs):    def diagonal_45(height, objs):
 .....                      .....                         .....

                           def X(height, objs):         def Staircase(height, objs):
........                    .....                         .....
```
```

Instruction: Construct a X where diagonals are replaced with staircases of size 5 increasing outwards

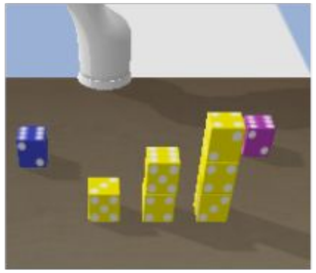


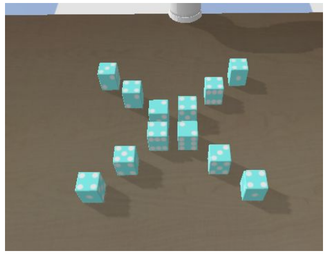
The learnt representations enable online generalization from very simple instructions to very complex instructions

Qualitative Results: Imagined plan for complex structures

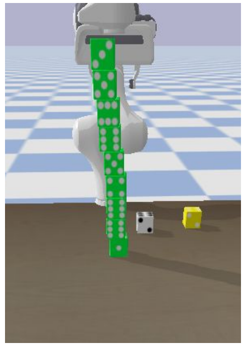
Human Demonstrated Concepts

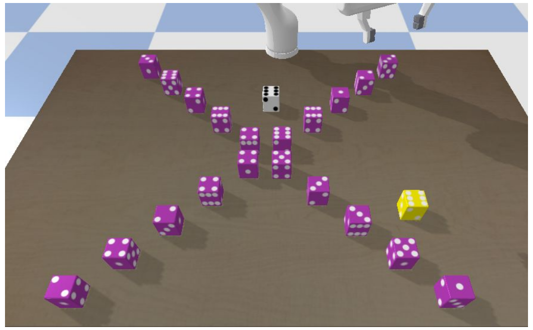

Tower of height 3


Staircase with 3 steps


X of size 3 using blue dice

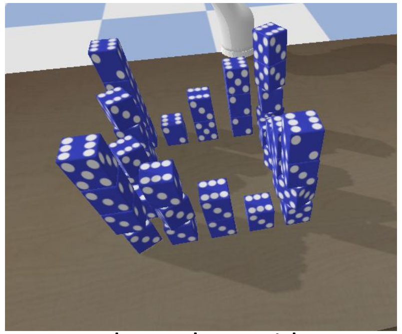
Inductive Generalization

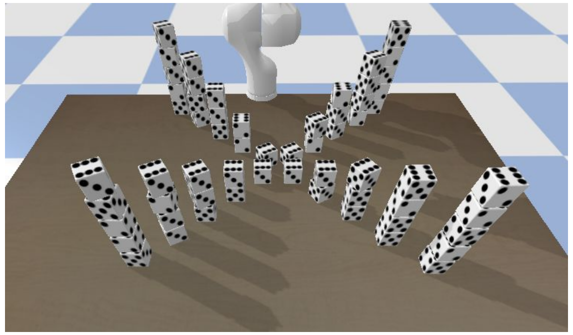

Tower of height 9


X of size 5

Can plan Larger instances than what is demonstrated

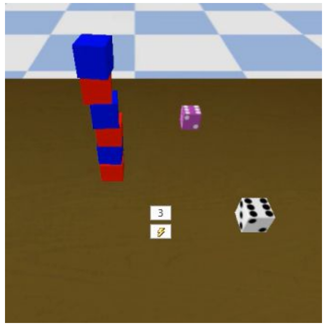
Compositional Generalization

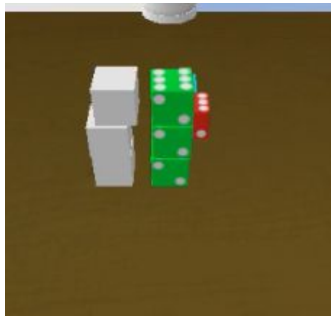

Boundary whose sides are replaced with staircases


X, whose diagonals are replaced with outward increasing staircases

Can compose demonstrated structures to plan complex assemblies construction

Constraint Generalization


Tower of *alternating* red and blue blocks


White tower whose *height is same* as green tower

Can incorporate novel visual/spatial constraints

Results

Concept Learning: Our method learns complex concepts effectively from human demonstration

Model	Simple			Complex		
	Acc.↑	IoU↑	MSE↓	Acc.↑	IoU↑	MSE ↓
SPL(Ours)	**1.00**	**0.96**	**0.01**	**0.83**	**0.85**	**2.06**
CaP (VLM)	0.33	0.75	4.33	0.00	0.50	7.29
CaP+VRF	0.66	0.75	6.8	0.16	0.46	12.0
CaP (LLM)	0.78	0.89	1.36	0.00	0.28	13.5
SD+G	NA	0.74	1.42	NA	0.61	2.43
SD	NA	0.49	1.48	NA	0.46	3.71

Inductive Generalization: SPL shows stronger inductive generalization over the learnt concepts

Model	Simple			Complex		
	IoU↑	R.D↓	MSE↓	IoU↑	R.D↓	MSE↓
SPL (Ours)	**0.89**	**7.27**	**0.43**	**0.80**	**5.74**	**1.49**
CaP (VLM)	0.58	23.33	13.2	0.29	41.64	10.9
CaP (LLM)	0.78	12.61	5.51	0.13	53.87	19.1
SD+G	0.27	63.25	6.21	0.15	74.72	14.2
SD	0.24	51.84	6.86	0.15	67.67	11.6

Dataset: Generated our own benchmark
Related Work: CaP: Code As Policies [Liang et al. 2022]. SD: Struct Diffusion [Liu et al. 2022]

Grounded in Demonstration: Performance retained with randomized concept names — even when LLMs cannot use world knowledge.

Model	Simple			Complex		
	Acc.↑	IoU↑	MSE↓	Acc.↑	IoU↑	MSE↓
SPL(Ours)	**0.88**	**0.86**	**1.74**	**0.78**	**0.78**	**3.93**
CaP (VLM)	0.23	0.71	3.92	0.00	0.09	21.29
CaP (LLM)	0.67	0.78	3.16	0.00	0.00	22.73

Robotics: Surgical Applications

1. Replace blocks by actual objects in a medical environment
2. Interpretable - human in the loop execution
3. Error Recovery (Work published at [IROS 2024])

Outline of the Talk

- Motivation
- Weakly Supervised Image Editing
- Error Recovery in Neuro-Symbolic Robotic Manipulation
- Solving Hard Combinatorial NP Hard Problems
- Other Directions

Other on-Going Directions

Theme based Game Generation

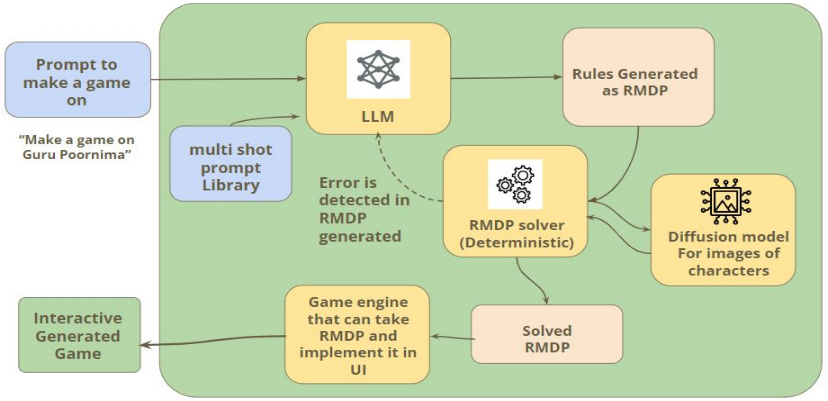
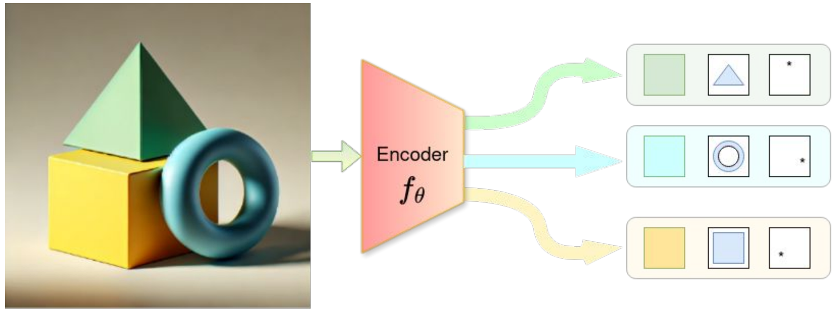


Image Generated using a T2I Model

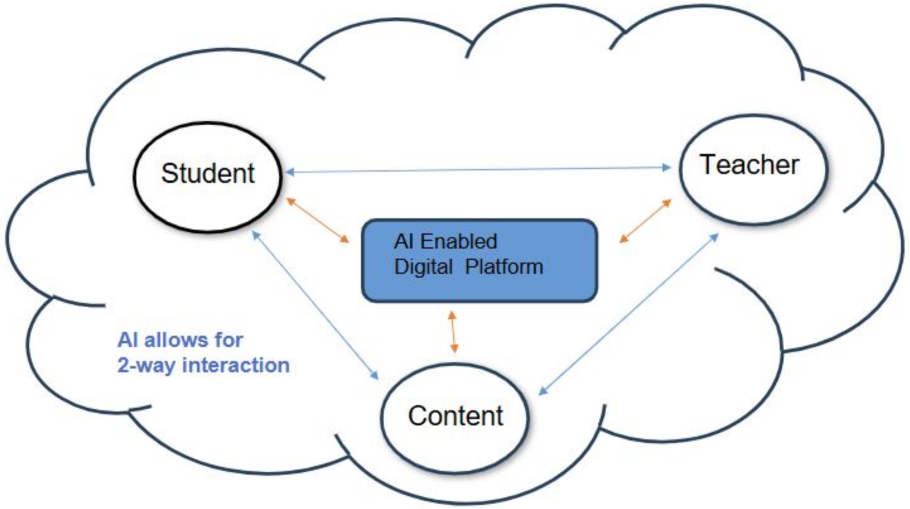
Attribute Disentanglement



Generating Pedagogical Content

Image Generated using Google Gemini

AI in Education



Collaborators (Colleagues & PhD Students)*

Prathosh A. P (IISc. Bangalore)

Rohan Paul (IIT Delhi)

Ashish Goswami (PhD Student)

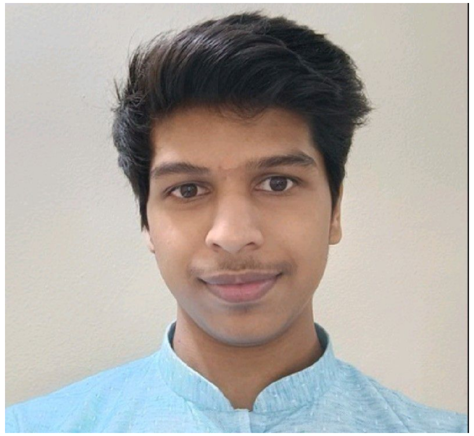
Namasivayam K (PhD Student)

*For two works presented in this talk.

Collaborators (Past UG/PG Students)*

Harman Singh

Satyam Modi



Santhosh Deshineni

Vishal Bindal

Gurarmaan Singh

Harsh Vora

Sachit Sachdeva

Divyanshu Agarwal

Himanshu Singh

Arnav Tuli

Sujeet Lahane

*For two works presented in this talk.

Questions and Discussion