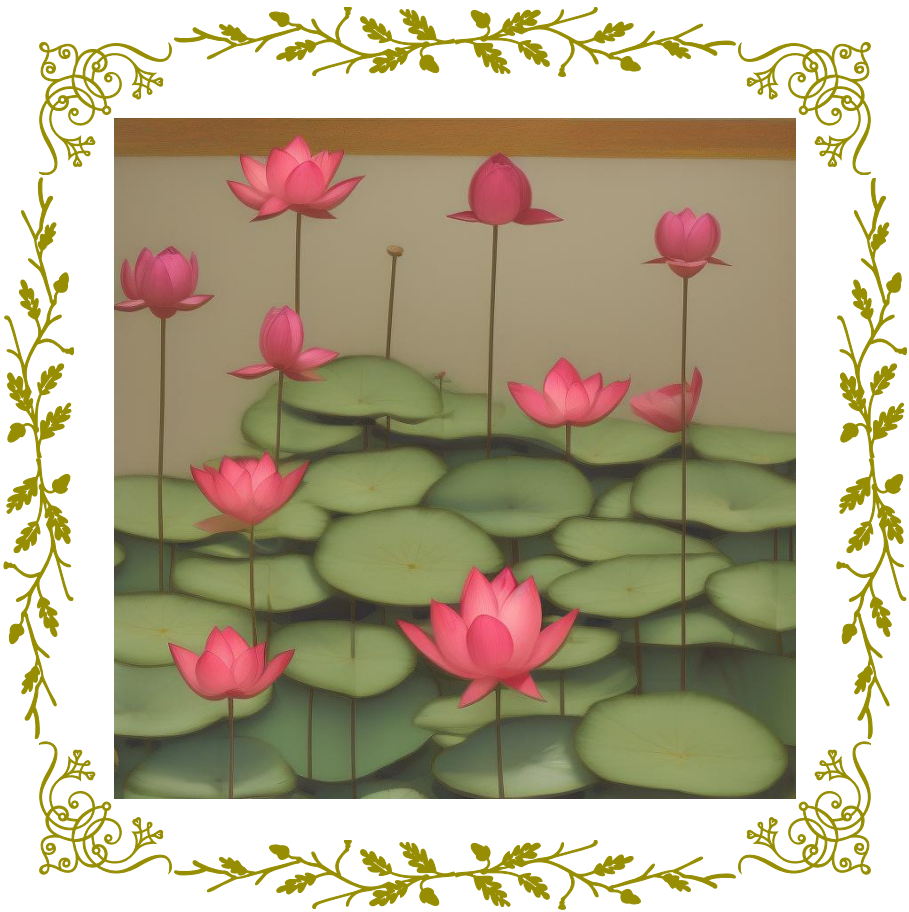


REPORT ON THE GENERATION

Executed on October 1, 2024



Essential paths of the run

result: ./outputs/results/text_to_image_final.png
inputs: ./outputs/input/
logs: ./outputs/logs/

Elapsed Time: ~7 seconds

- Domain classifier 1.22 seconds
- Problem+Plan generation ... 0.91 seconds
- Code execution 6.89 seconds

1 Inputs Provided by the User

User Query

Build a pictorial presentation of the flower after deciphering the audio
./data/audio/audio_4.wav

2 Inputs Provided by the User

User Text

```
{
  "instruction": "Convert the audio to text and then generate an image",
  "input_text": null,
  "question": null,
  "url": "./data/audios/audio_4.wav",
  "data_dict": {},
  "categories": []
}
```

3 PDDL Domain

3.1 Domain Classification

['audio', 'image_generation']

3.2 Domain Merging

```
(define (domain audio-image-generation)
  (:requirements :typing :strips)
  (:types text image audio)
  (:predicates
    (IsText ?input_text - text)
    (SpeechRecognition ?input_audio - audio)
    (GenerateImageFromText ?input_text - text)
    (GenerateAudioFromText ?input_text - text)
    (IsImage ?input_image - image)
    (ConvertAudioToAudio ?input_audio - audio)
    (RefineImage ?input_image - image ?input_text - text)
    (IsAudio ?input_audio - audio)
  )
  (:action text_to_audio ; given a piece of text, convert it to audio
    :parameters (?input_text - text)
    :precondition (IsText ?input_text)
    :effect (GenerateAudioFromText ?input_text)
  )
  (:action audio_to_audio ; convert a given audio to another audio
    :parameters (?input_audio - audio)
    :precondition (IsAudio ?input_audio)
    :effect (ConvertAudioToAudio ?input_audio)
  )
  (:action audio_to_text ; given an audio, transcribe it to text
    :parameters (?input_audio - audio)
    :precondition (IsAudio ?input_audio)
    :effect (SpeechRecognition ?input_audio)
  )
)
```

```

    (:action text_to_image ; given a piece of text, generates an image
      :parameters (?input_text - text)
      :precondition (IsText ?input_text)
      :effect (GenerateImageFromText ?input_text)
    )
    (:action image_refinement ; given an image, we try to refine it.
      Image-to-image
      :parameters (?input_image - image ?input_text - text)
      :precondition (and (IsImage ?input_image)(IsText ?input_text))
      :effect (RefineImage ?input_image ?input_text)
    )
  )
)

```

4 PDDL Problem

```

(define (problem audio_to_text-text_to_image-problem)
  (:domain audio-image-generation)
  (:objects input_audio - audio input_text - text)
  (:init (IsAudio input_audio) (IsText input_text))
  (:goal (and (SpeechRecognition input_audio) (GenerateImageFromText input_text)))
)

```

PLANNER_OUTPUT_START

predicate REFINEIMAGE is declared to use unknown or empty type IMAGE

predicate ISIMAGE is declared to use unknown or empty type IMAGE

warning: parameter ?INPUT_IMAGE of op IMAGE_REFINEMENT has unknown or empty type IMAGE. skipping op --- OK
Match tree built with 4 nodes.

PDDL problem description loaded:

Domain: AUDIO-IMAGE_GENERATION

Problem: AUDIO_TO_TEXT-TEXT_TO_IMAGE-PROBLEM

#Actions: 4

#Fluents: 4

Goals found: 2

Goals_Edges found: 2

Starting search with 1-BFWS...

--[2 / 0]--

--[1 / 0]--

--[1 / 1]--

--[0 / 0]--

--[0 / 1]--

Total time: 0.000115

Nodes generated during search: 9

Nodes expanded during search: 2

Plan found with cost: 2

Fast-BFS search completed in 0.000115 secs

PLANNER_OUTPUT_END

PLAN_START

(TEXT_TO_IMAGE INPUT_TEXT)

(AUDIO_TO_TEXT INPUT_AUDIO)

PLAN_END

5 PDDL Plan

```
(TEXT_TO_IMAGE INPUT_TEXT)
(AUDIO_TO_TEXT INPUT_AUDIO)
```

6 Model Selection

The Domain Classifier and the generated Plan lead us to short list the following models:
Given the user's preferences:

```
{
  "licenses": [
    "any_license"
  ],
  "size": "no_constraints",
  "benchmarks": {
    "text_to_image": {},
    "audio_to_text": {}
  }
}
```

We selected the following model(s):

Task: audio_to_text
Model: openai/whisper-large-v2

—

Task: text_to_image
Model: stabilityai/stable-diffusion-2-1

7 Python Code Generation

SELECTED_MODEL_START

Task: audio_to_text

Model: openai/whisper-large-v2

SELECTED_MODEL_END

```
def transcribe_audio(audio_file_path, model_path="/path/to/model"):

    processor = WhisperProcessor.from_pretrained(model_path)
    model = WhisperForConditionalGeneration.from_pretrained(model_path)
    model.config.forced_decoder_ids = None

    waveform, sampling_rate = torchaudio.load(audio_file_path)
    sample = dict(array=waveform[0].numpy(), sampling_rate=16000)
    input_features = processor(sample["array"], sampling_rate=sample["sampling_rate"],
    ↪ return_tensors="pt").input_features
    predicted_ids = model.generate(input_features)

    transcription = processor.batch_decode(predicted_ids, skip_special_tokens=True)[0]
    return transcription
```

INTERMEDIATE_OUTPUT_START

./outputs/results/audio_to_text_intermediate.txt

INTERMEDIATE_OUTPUT_END

SELECTED_MODEL_START

Task: text_to_image

Model: stabilityai/stable-diffusion-2-1

SELECTED_MODEL_END

```
def run_model(prompt="An astronaut riding a green horse",
↪ model_path="/data/small_plms/stable-diffusion-xl-base-0.9", use_safetensors=True,
↪ torch_dtype=torch.float16):
    pipe = DiffusionPipeline.from_pretrained(model_path, torch_dtype=torch_dtype,
↪ use_safetensors=use_safetensors, variant="fp16")

    if torch.__version__ < '2.0':
        pipe.enable_xformers_memory_efficient_attention()

    pipe = pipe.to("cuda")

    images = pipe(prompt=prompt).images[0]

    return images
```

FINAL_OUTPUT_START

./outputs/results/text_to_image_final.png

FINAL_OUTPUT_END