CMPT 413/713: Natural Language Processing

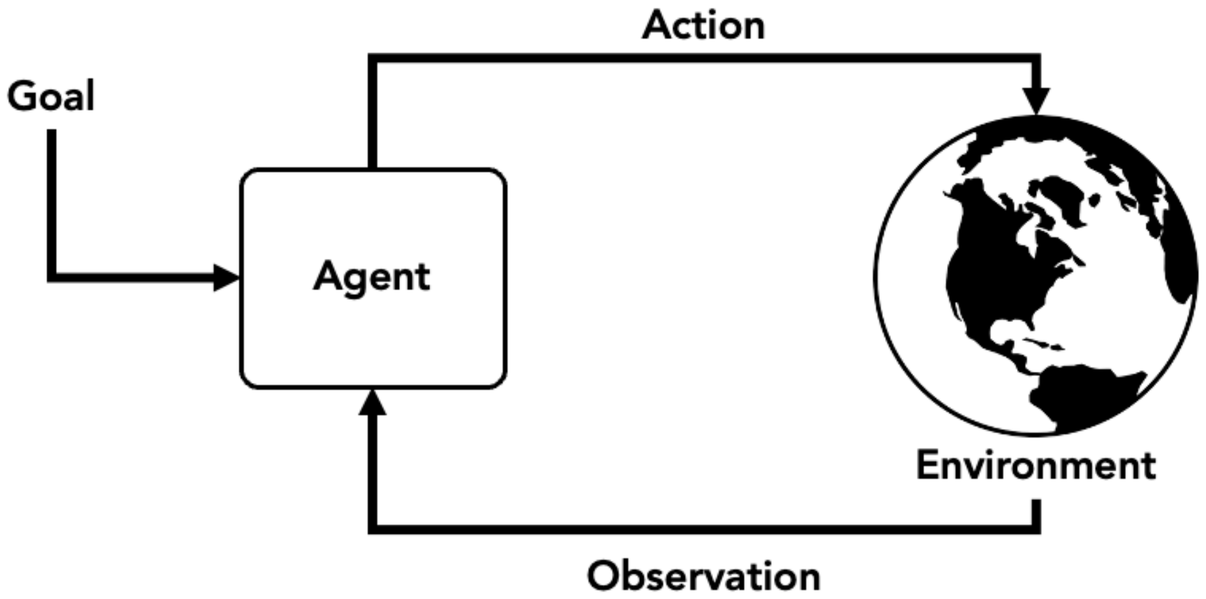
LLM agents

Spring 2024

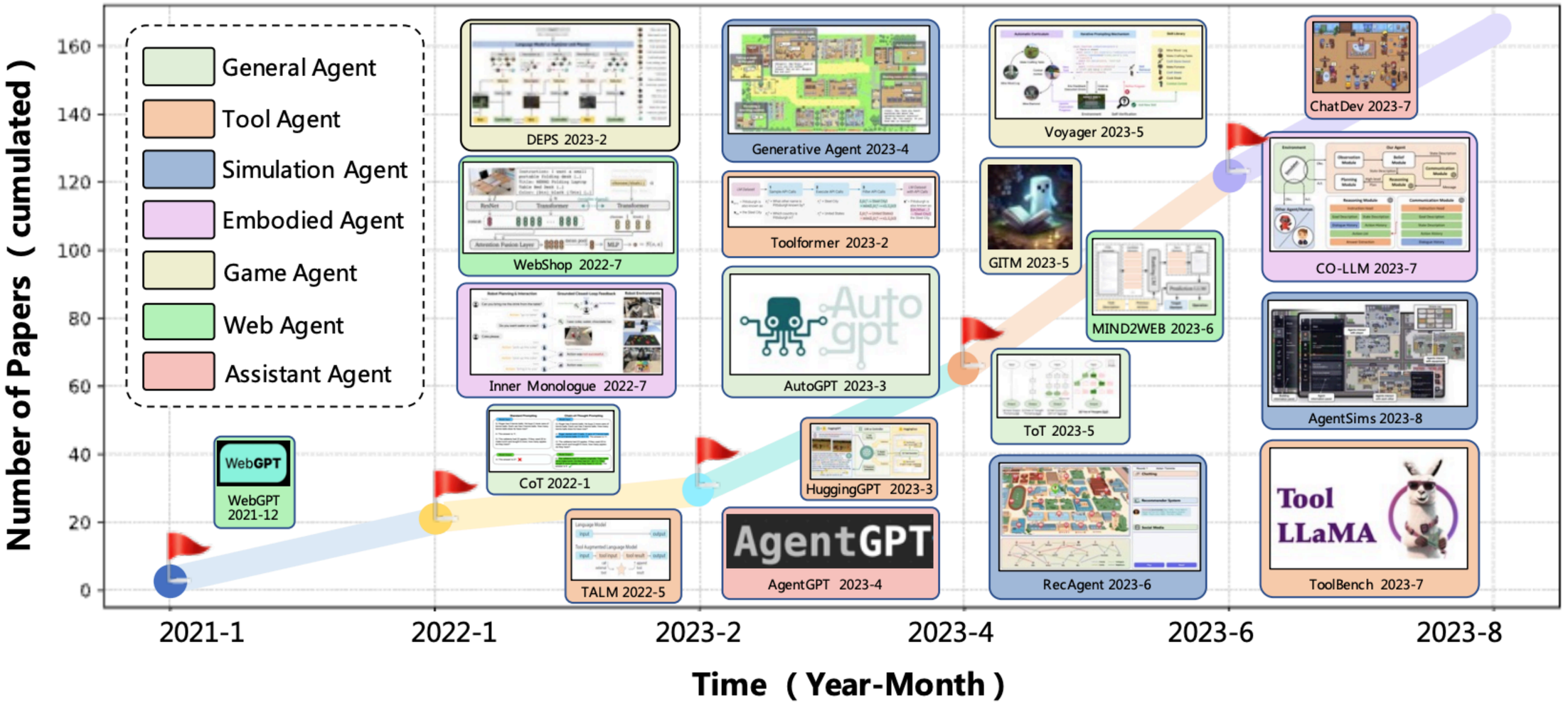
2024-03-27

LLMs as agents

- There is a lot of knowledge in LLMs
- But they can't "act"
- Can we leverage LLMs to build a "smart" agent that can interact with the environment to achieve given goals?



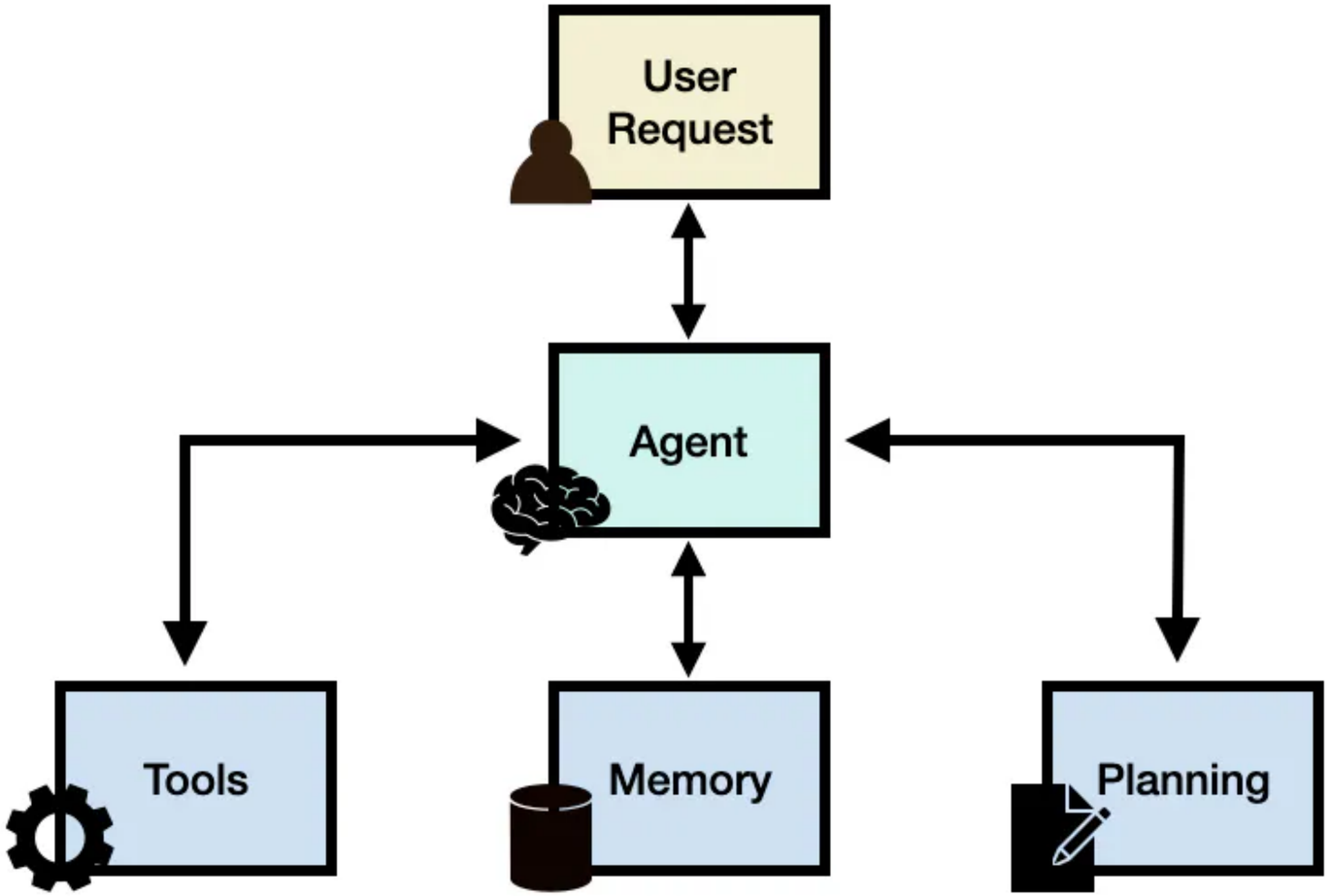
Lots of work!



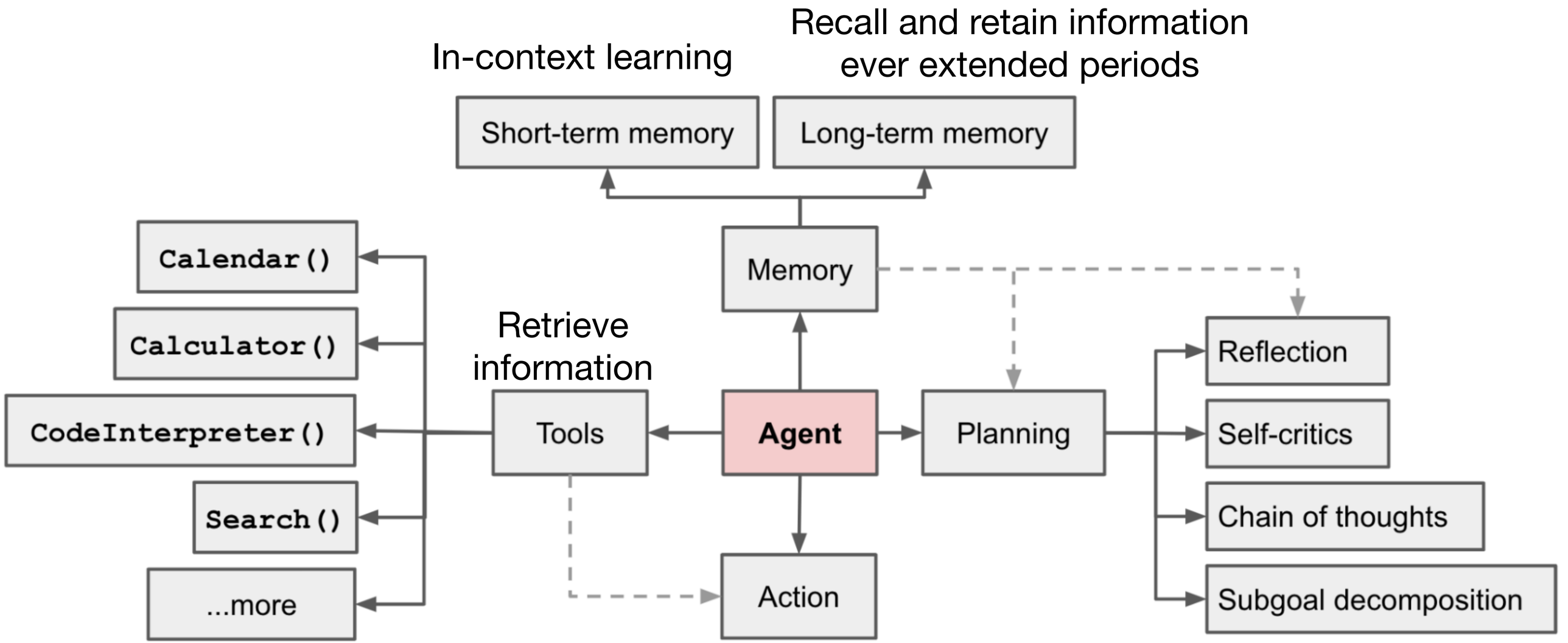
A Survey on Large Language Model based Autonomous Agents [Wang et al, 2023]

LLM Agent Framework

API calls to external services, expert modules, etc



https://www.promptingguide.ai/research/llm-agents



https://lilianweng.github.io/posts/2023-06-23-agent/

Planning

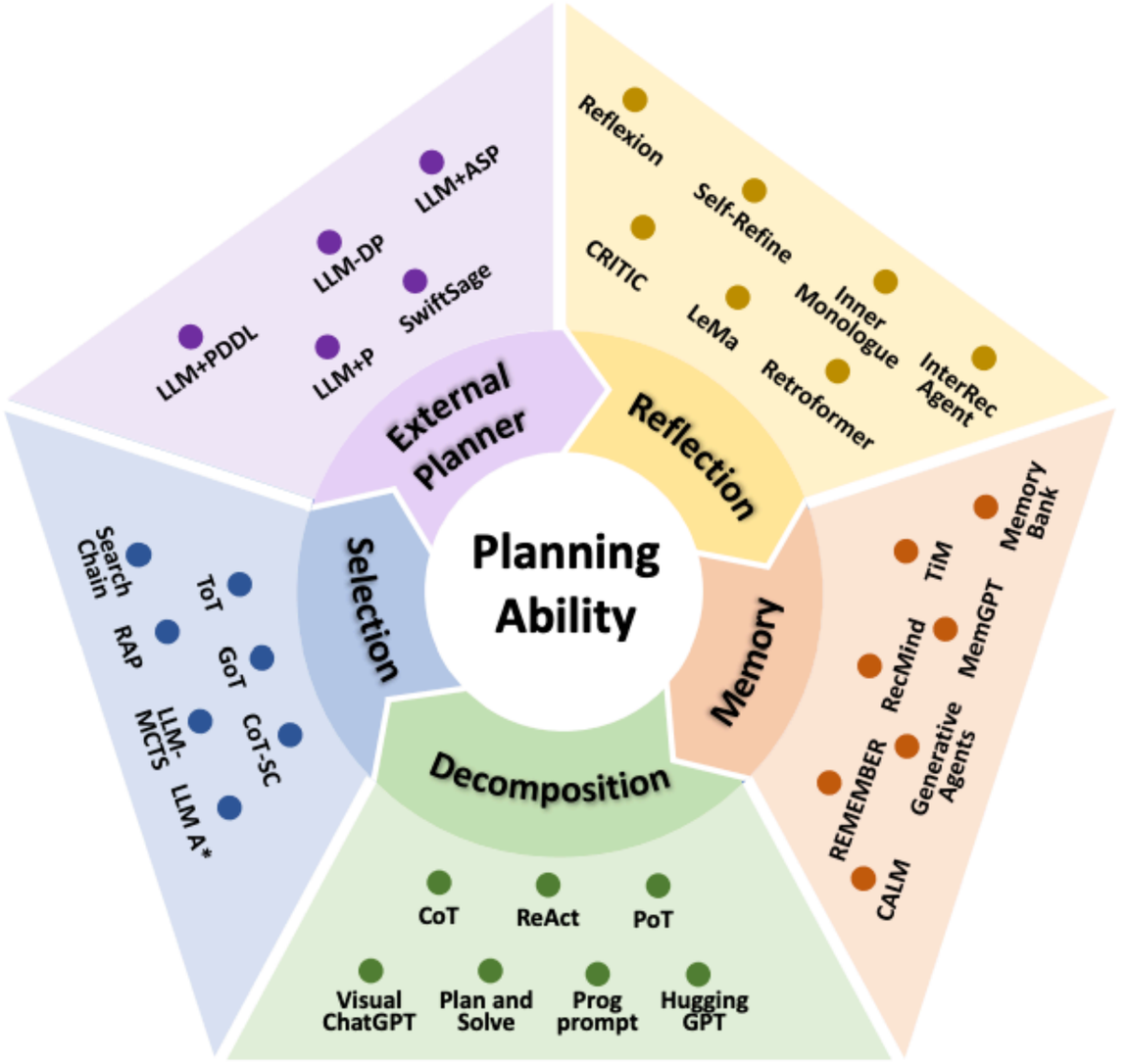
Planning

- What does the agent need to do to accomplish a specified goal?
- Low-level vs high-level planning
  - High-level plan: Identify subgoals for a long-horizon task
  - Low-level plan (sometime low-level control): Identify sequence of actions
- Traditionally use symbolic reasoning
  - Hard to recover from errors
  - Difficult to convert expert knowledge into planning languages such as PDDL (Planning Domain Definition Language)
- Use of LLM for planning
  - Lot of expert domain knowledge already encoded in language
  - Can we use LLMs to help us plan?

Taxonomy for planning with LLMs

Given environment and sequence of actions, and an overall task goal, identify subgoals and actions

- **Task decomposition** - figure out subgoals, do planning for subgoals if needed
- **Multi-plan selection** - generate multiple plans and then select one
- **External planner** - LLM used to formalized the problem which is passed to an external planner
- **Reflection and refinement** - After obtaining a plan, the LLM future reflects on the plan and refine it to fix any issues with the original plan
- **Memory-augmented planning** - Uses external memory to retrieve information (common sense knowledge, domain-specific knowledge, etc) and then determines plan based on that



Understanding the planning of LLM agents: A survey [Huang et al, 2024]

Self-Reflection

- ReAct: interleave thought, action, and observation
- Prompting to combine reasoning with actions

(1) Hotspot QA

Question: Aside from the Apple Remote, what other device can control the program Apple Remote was originally designed to interact with?

(1d) ReAct (Reason + Act)

Thought 1: I need to search Apple Remote and find the program it was originally designed to interact with.
Act 1: Search[Apple Remote]
Obs 1: The Apple Remote is a remote control introduced in October 2005 by Apple … originally designed to control the Front Row media center program …

Thought 2: Apple Remote was originally designed to control the Front Row media center program. I need to search Front Row next and find what other device can control it.
Act 2: Search[Front Row]
Obs 2: Could not find [Front Row]. Similar: ['Front Row Seat to Earth', 'Front Row Motorsports',' Front Row (software) ', …]

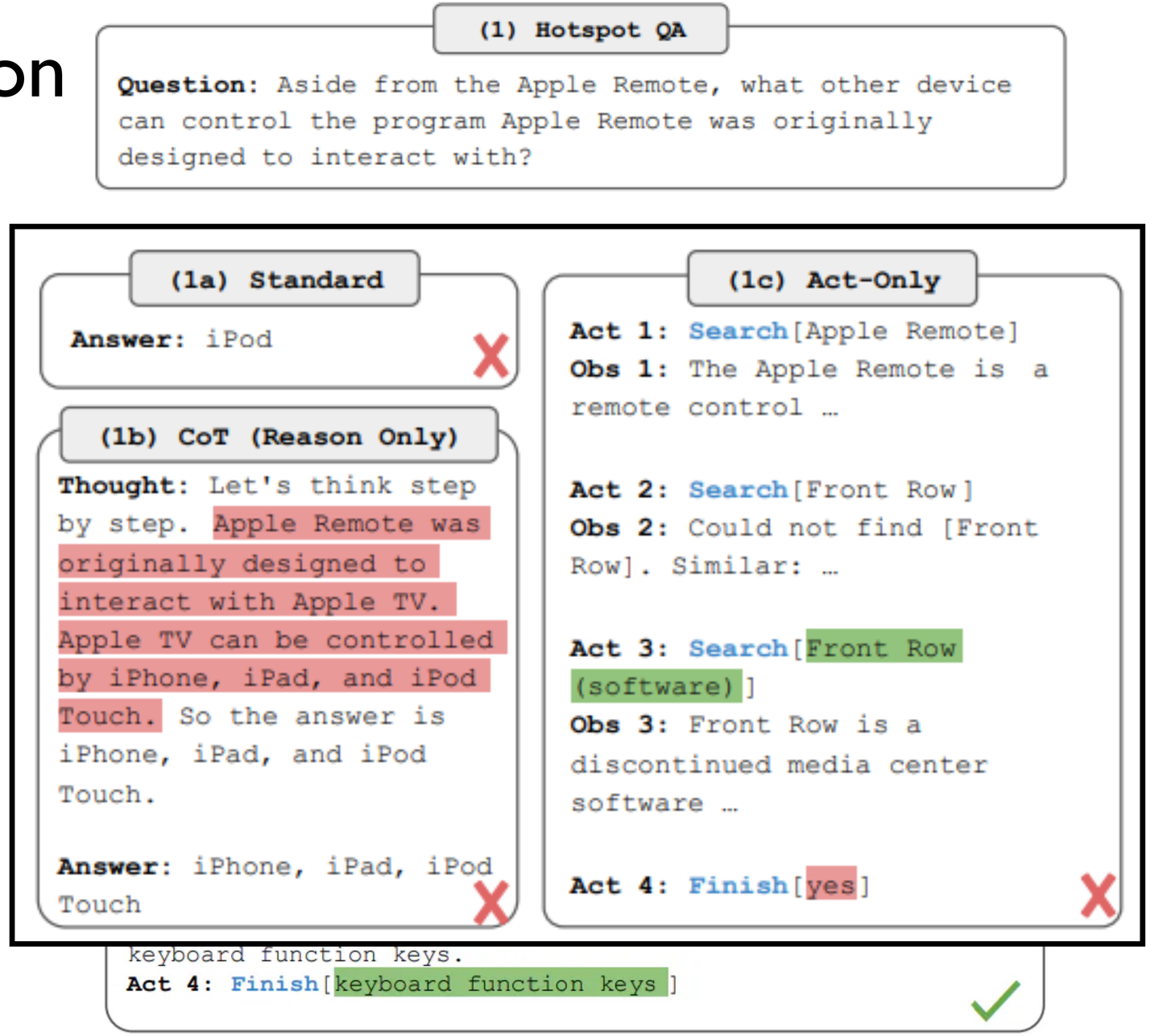
Thought 3: Front Row is not found. I need to search Front Row (software) .
Act 3: Search[Front Row (software)]
Obs 3: Front Row is a discontinued media center software …

Thought 4: Front Row (software) is controlled by an Apple Remote or the keyboard function keys. So the answer is keyboard function keys.
Act 4: Finish[keyboard function keys]

ReAct: Synergizing Reasoning and Acting in Language Models [Yao et al. 2022]

Self-Reflection

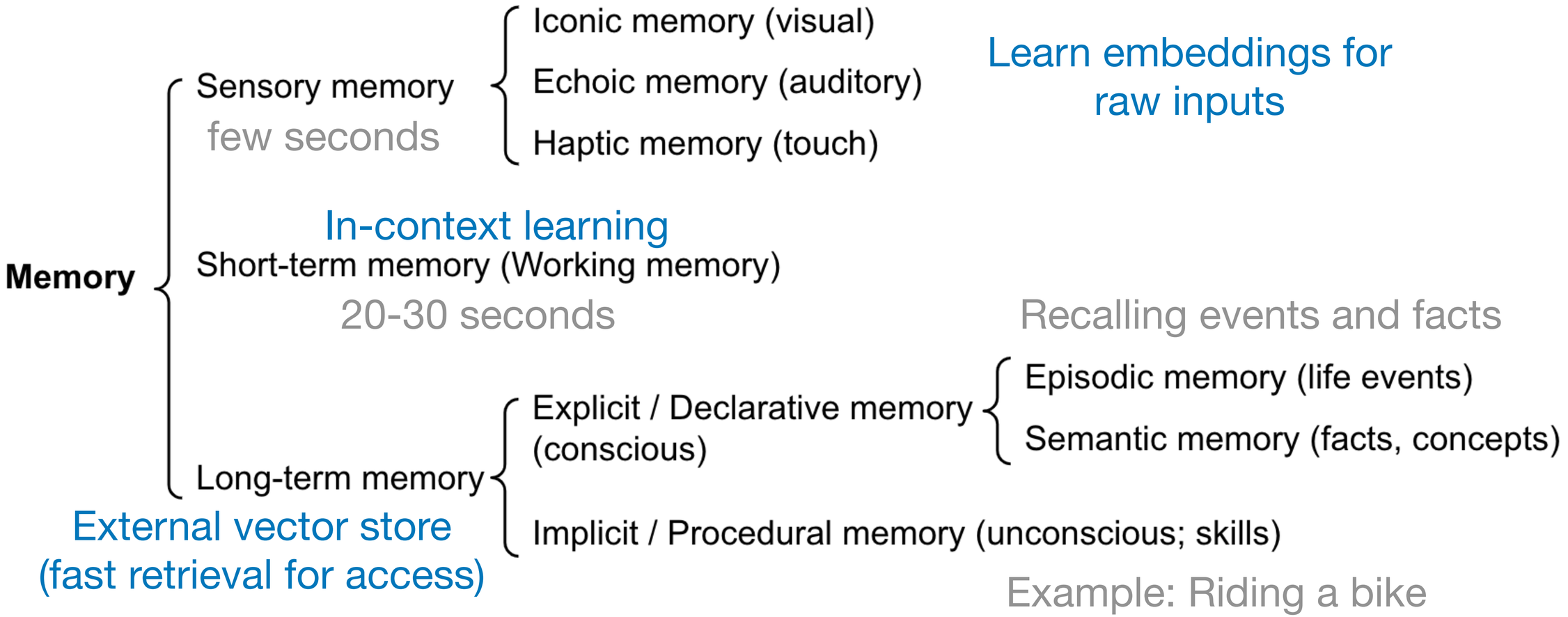
- Comparison with other ways of prompting



ReAct: Synergizing Reasoning and Acting in Language Models [Yao et al. 2022]

Memory

Types of memory



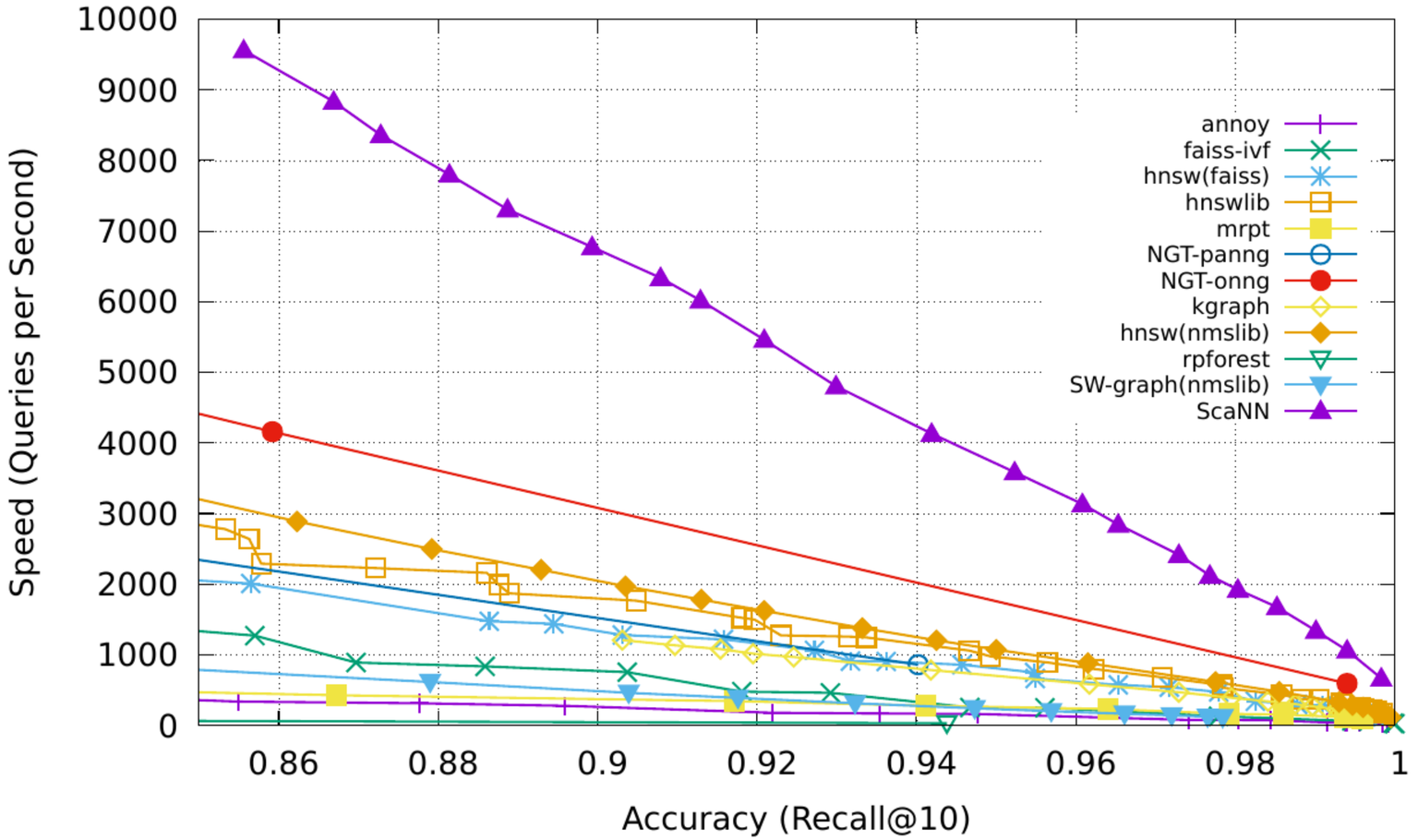
https://lilianweng.github.io/posts/2023-06-23-agent/

Efficient retrieval from memory

Approximate nearest neighbours (ANN) algorithms to return top k nearest neighbours using maximum inner product search (MIPS)

- LSH (Locality Sensitive Hashing) - hashing function so that similar inputs are mapped to same buckets with high probability
- ANNOY (ANN Oh Yeah) - Random projection trees where nodes splits input space into half
- HNSW (Hierarchical Navigable Small World)
  - Hierarchical layers of small-world graphs (points in the bottom layers)
  - Can be used with FAISS
- FAISS (Facebook AI Similarity Search) - vector quantization - partition vector space into clusters
- ScaNN (Scalable Nearest Neighbours) - Anisotropic vector quantization (quantize points while maintaining distances)

https://lilianweng.github.io/posts/2023-06-23-agent/



https://blog.research.google/2020/07/announcing-scann-efficient-vector.html

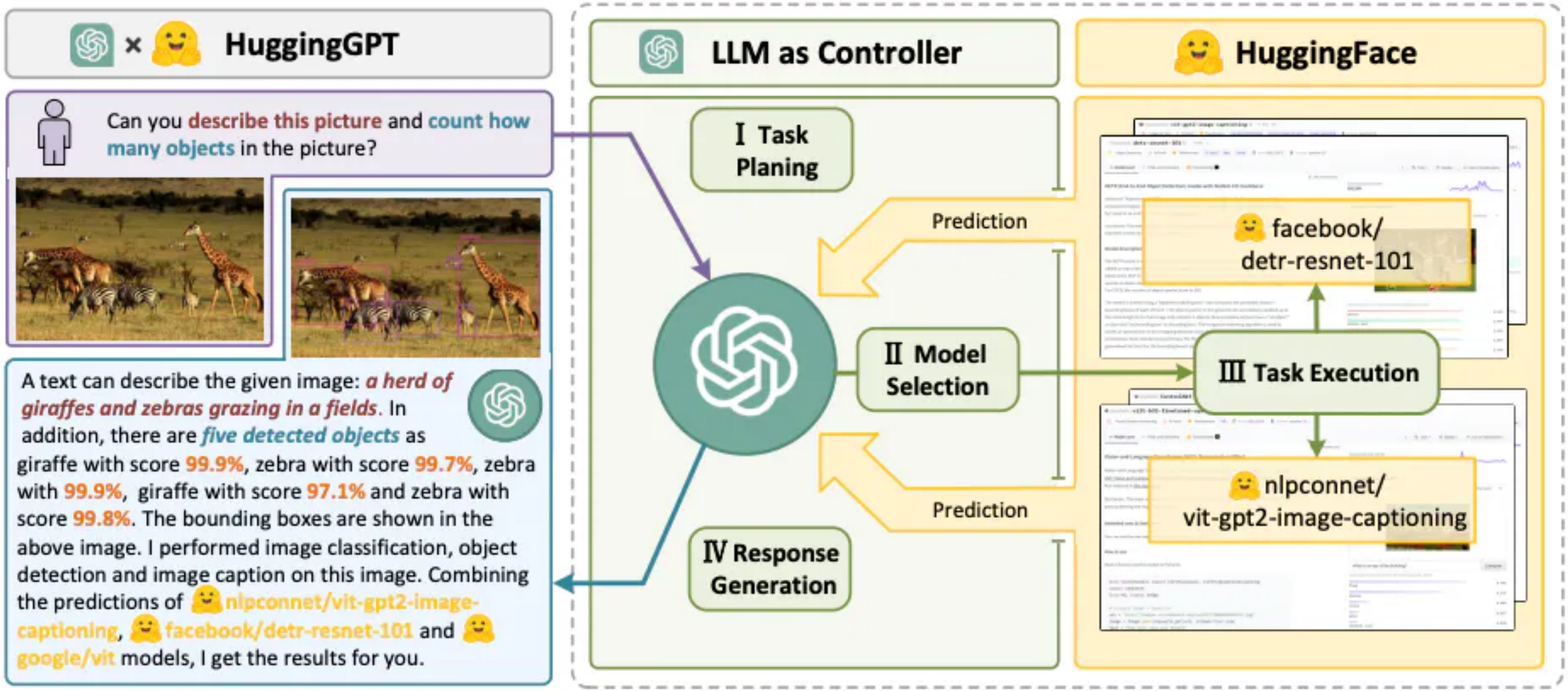
Note: nmslib focus is on non-metric spaces

Tool use

- API calls to external services (math calculator, currency converter, etc)
- Expert models that can be called

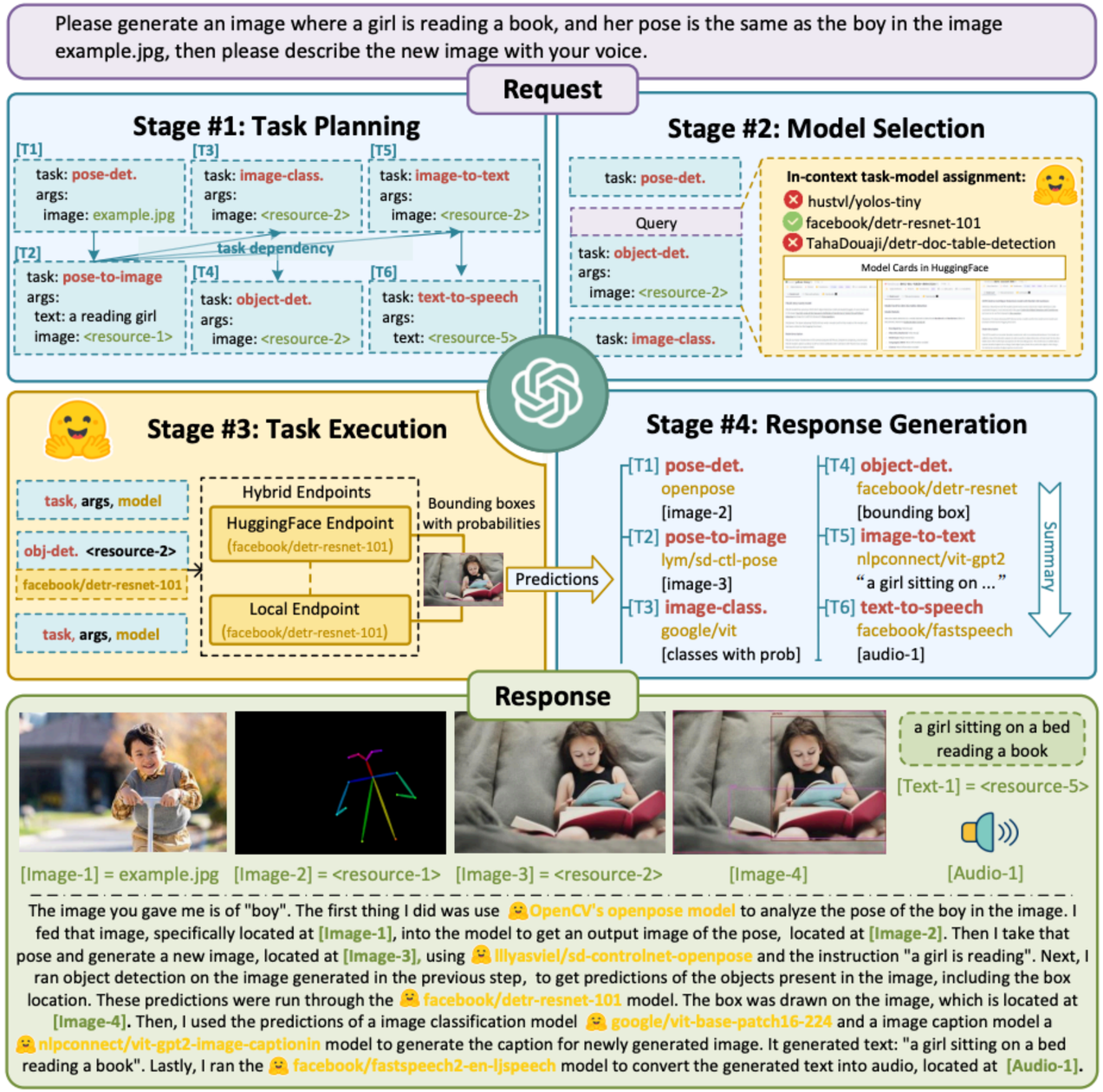
HuggingGPT: Task decomposition with model selection

1. Task planning
2. Model selection
3. Task execution
4. Response generation



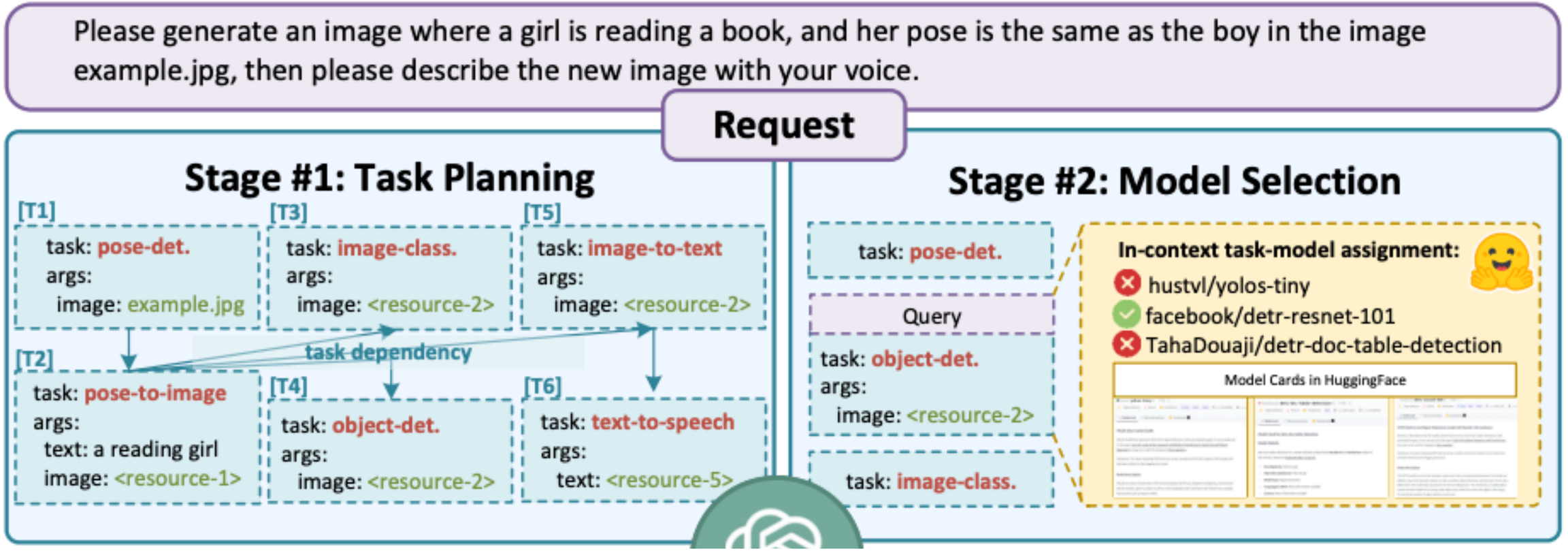
HuggingGPT: Solving AI Tasks with ChatGPT and its Friends in Hugging Face [Shen et al, 2023]

HuggingGPT



HuggingGPT: Solving AI Tasks with ChatGPT and its Friends in Hugging Face [Shen et al, 2023]

HuggingGPT



HuggingGPT: Solving AI Tasks with ChatGPT and its Friends in Hugging Face [Shen et al, 2023]

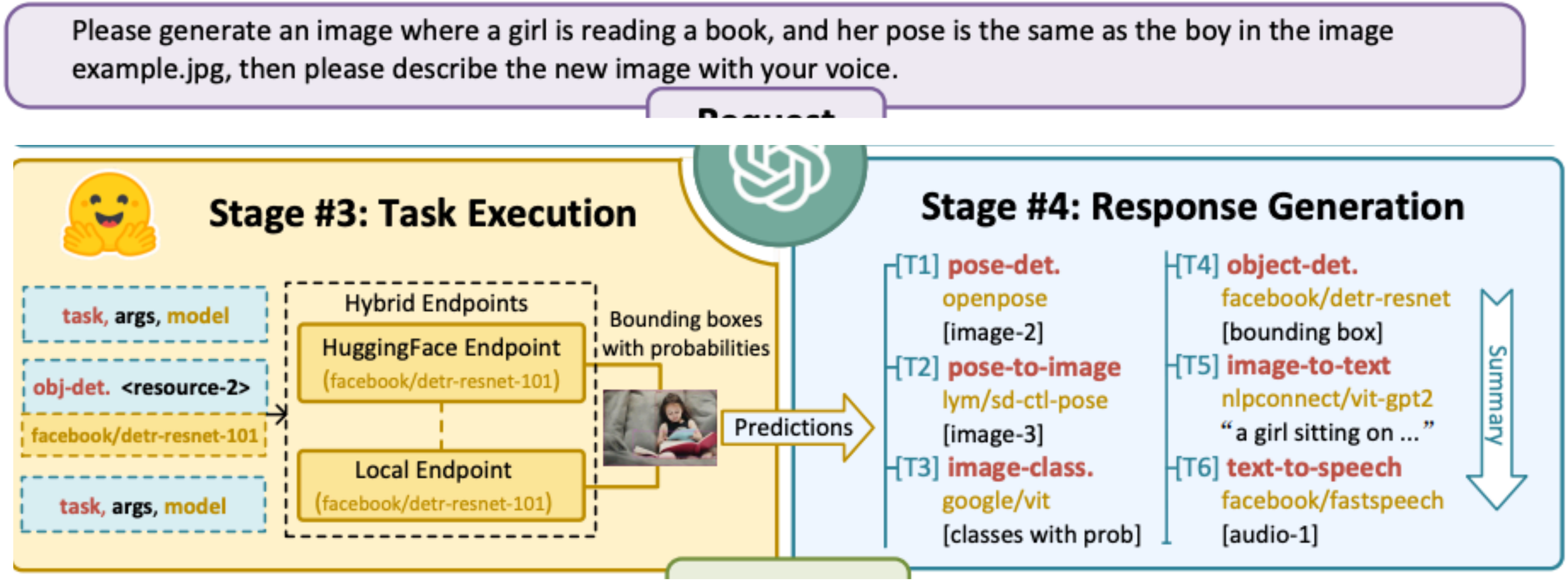
HuggingGPT

Task planning: figure out what task we want to solve, its id, **dependencies**, and **arguments** that are needed.

	Prompt	
Task Planning	#1 Task Planning Stage - The AI assistant performs task parsing on user input, generating a list of tasks with the following format: [{"task": task, "id", task_id, "dep": dependency_task_ids, "args": {"text": text, "image": URL, "audio": URL, "video": URL}}]. The "dep" field denotes the id of the previous task which generates a new resource upon which the current task relies. The tag "<resource>-task_id" represents the generated text, image, audio, or video from the dependency task with the corresponding task_id. The task must be selected from the following options: {{ *Available Task List* }}. Please note that there exists a logical connections and order between the tasks. In case the user input cannot be parsed, an empty JSON response should be provided. Here are several cases for your reference: {{ *Demonstrations* }}. To assist with task planning, the chat history is available as {{ *Chat Logs* }}, where you can trace the user-mentioned resources and incorporate them into the task planning stage.	
	Demonstrations	
	Can you tell me how many objects in e1.jpg?	[{"task": "object-detection", "id": 0, "dep": [-1], "args": {"image": "e1.jpg" }}]
	In e2.jpg, what's the animal and what's it doing?	[{"task": "image-to-text", "id": 0, "dep":[-1], "args": {"image": "e2.jpg" }}, {"task":"image-cls", "id": 1, "dep": [-1], "args": {"image": "e2.jpg" }}, {"task":"object-detection", "id": 2, "dep": [-1], "args": {"image": "e2.jpg" }}, {"task": "visual-quesrion-answering", "id": 3, "dep":[-1], "args": {"text": "what's the animal doing?", "image": "e2.jpg" }}]
	First generate a HED image of e3.jpg, then based on the HED image and a text "a girl reading a book", create a new image as a response.	[{"task": "pose-detection", "id": 0, "dep": [-1], "args": {"image": "e3.jpg" }}, {"task": "pose-text-to-image", "id": 1, "dep": [0], "args": {"text": "a girl reading a book", "image": "<resource>-0" }}]

HuggingGPT: Solving AI Tasks with ChatGPT and its Friends in Hugging Face [Shen et al, 2023]

HuggingGPT

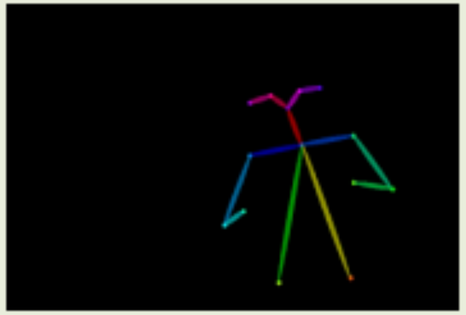


HuggingGPT: Solving AI Tasks with ChatGPT and its Friends in Hugging Face [Shen et al, 2023]

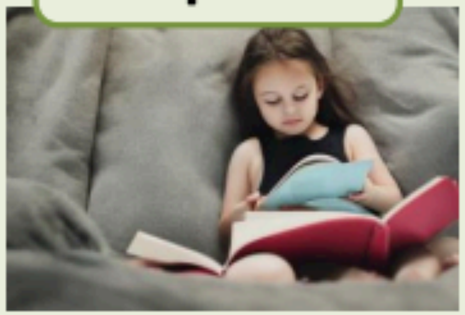
HuggingGPT

Response

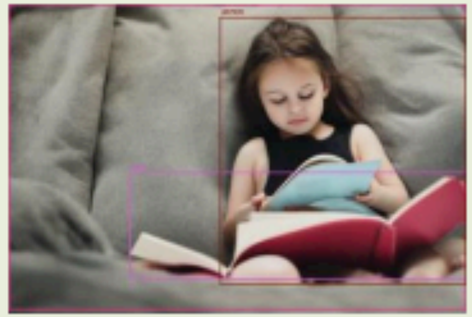
[Image-1] = example.jpg



[Image-2] = <resource-1>



[Image-3] = <resource-2>



[Image-4]

a girl sitting on a bed reading a book

[Text-1] = <resource-5>

[Audio-1]

The image you gave me is of "boy". The first thing I did was use 🤗OpenCV's openpose model to analyze the pose of the boy in the image. I fed that image, specifically located at [Image-1], into the model to get an output image of the pose, located at [Image-2]. Then I take that pose and generate a new image, located at [Image-3], using 🤗 lllyasviel/sd-controlnet-openpose and the instruction "a girl is reading". Next, I ran object detection on the image generated in the previous step, to get predictions of the objects present in the image, including the box location. These predictions were run through the 🤗 facebook/detr-resnet-101 model. The box was drawn on the image, which is located at [Image-4]. Then, I used the predictions of a image classification model 🤗 google/vit-base-patch16-224 and a image caption model a 🤗 nlpconnect/vit-gpt2-image-captionin model to generate the caption for newly generated image. It generated text: "a girl sitting on a bed reading a book". Lastly, I ran the 🤗 facebook/fastspeech2-en-ljspeech model to convert the generated text into audio, located at [Audio-1].

HuggingGPT: Solving AI Tasks with ChatGPT and its Friends in Hugging Face [Shen et al, 2023]

HuggingGPT

Model selection: figure out what model to invoke

	Prompt
Model Selection	#2 Model Selection Stage - Given the user request and the call command, the AI assistant helps the user to select a suitable model from a list of models to process the user request. The AI assistant merely outputs the model id of the most appropriate model. The output must be in a strict JSON format: {"id": "id", "reason": "your detail reason for the choice"}. We have a list of models for you to choose from {{ *Candidate Models* }}. Please select one model from the list.
	Candidate Models
	{"model_id": model id #1, "metadata": meta-info #1, "description": description of model #1} {"model_id": model id #2, "metadata": meta-info #2, "description": description of model #2} . . . . {"model_id": model id #K, "metadata": meta-info #K, "description": description of model #K}

HuggingGPT: Solving AI Tasks with ChatGPT and its Friends in Hugging Face [Shen et al, 2023]

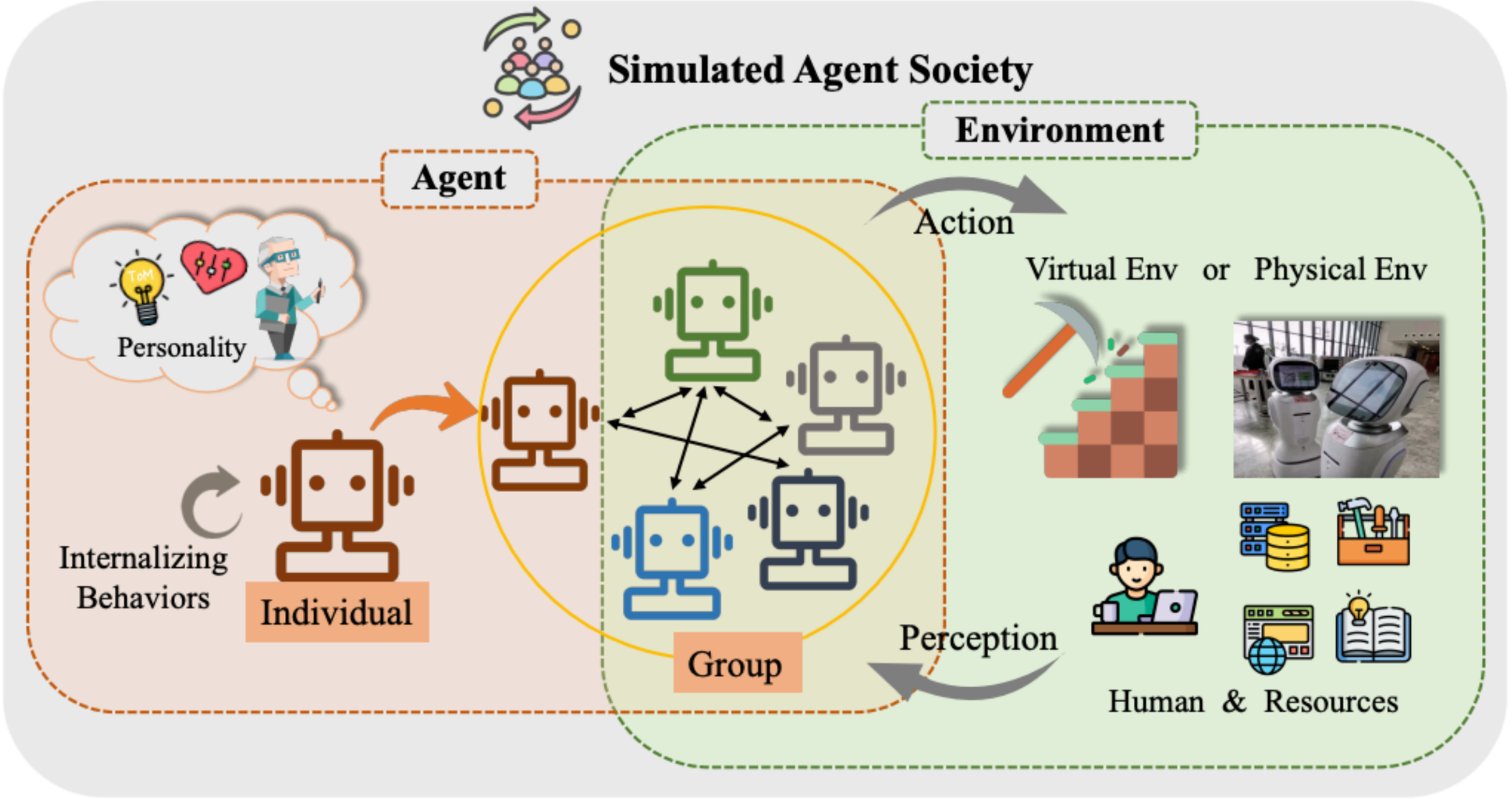
HuggingGPT

Response generation: respond to user the process and results

	Prompt
Response Generation	#4 Response Generation Stage - With the input and the inference results, the AI assistant needs to describe the process and results. The previous stages can be formed as - User Input: {{ *User Input* }}, Task Planning: {{ *Tasks* }}, Model Selection: {{ *Model Assignment* }}, Task Execution: {{ *Predictions* }}. You must first answer the user's request in a straightforward manner. Then describe the task process and show your analysis and model inference results to the user in the first person. If inference results contain a file path, must tell the user the complete file path. If there is nothing in the results, please tell me you can't make it.

HuggingGPT: Solving AI Tasks with ChatGPT and its Friends in Hugging Face [Shen et al, 2023]

Virtual worlds



The Rise and Potential of Large Language Model Based Agents: A Survey [Xi et al, 2023]

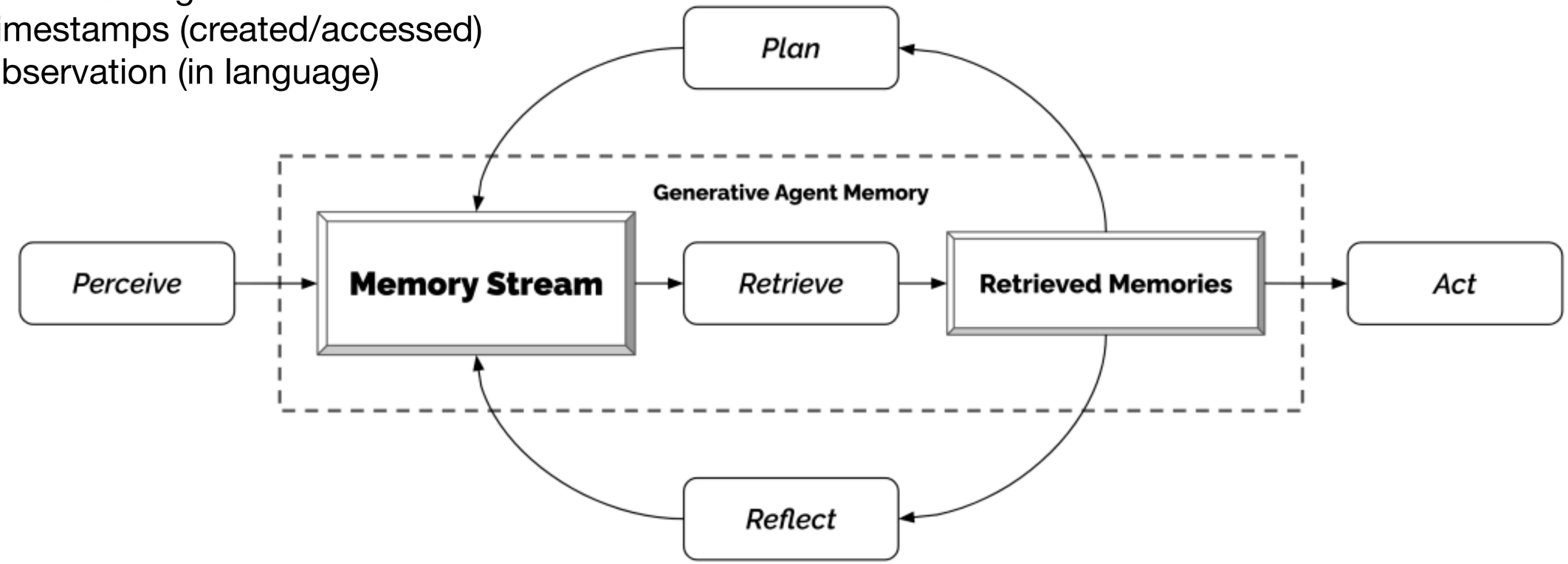
Generative Agents

Generative Agents: Interactive Simulacra of Human Behavior [Park et al, 2023]

Generative Agents

Memory Stream

- List of memory objects
- Each containing:
  - Timestamps (created/accessed)
  - Observation (in language)



Generative Agents: Interactive Simulacra of Human Behavior [Park et al, 2023]

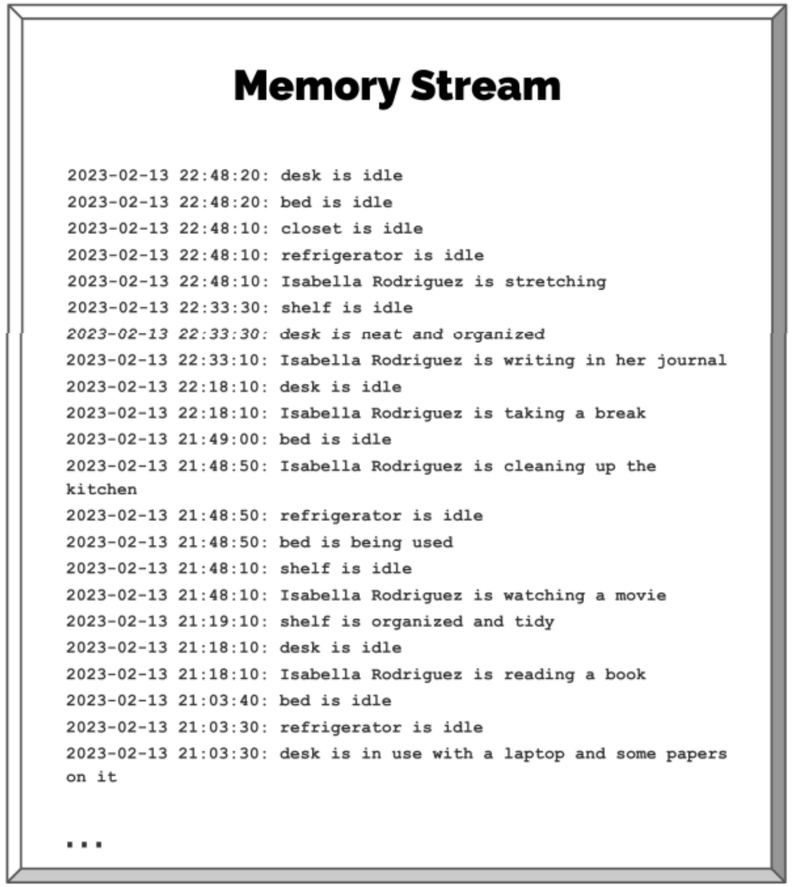
Generative Agents

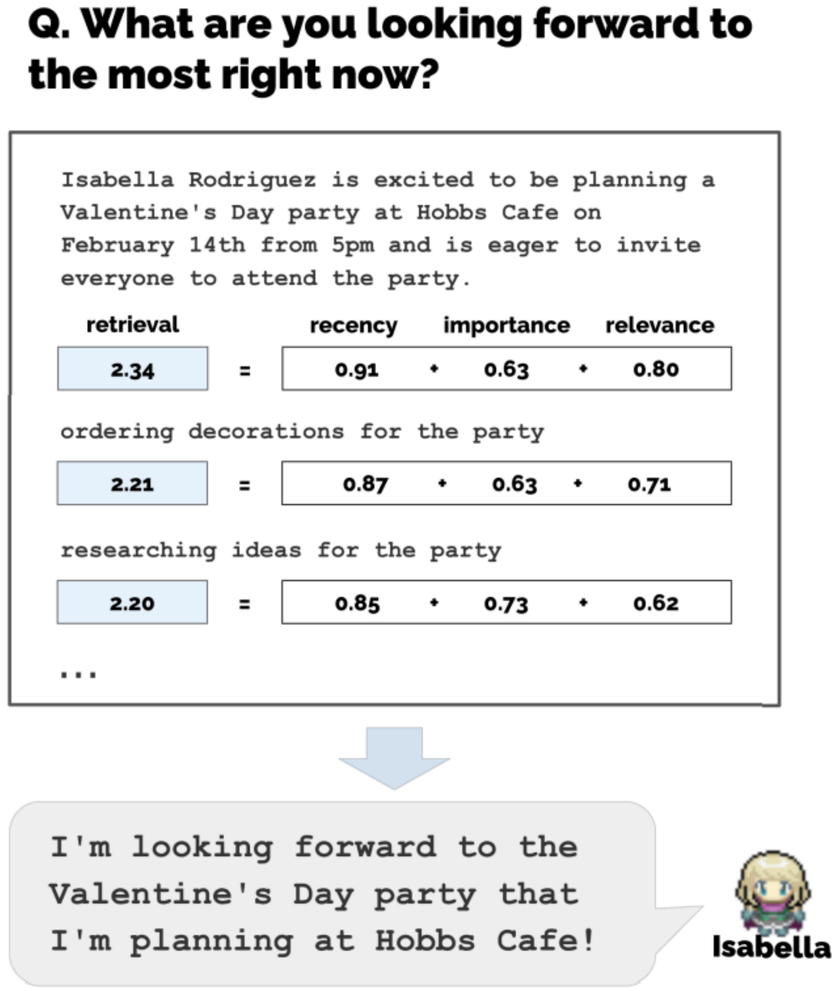
Retrieval extracts subset of memories based on:

Recency: exponential decay

Importance: ask LLM to output score

Relevance: cosine similarity of memory and query embeddings





Generative Agents: Interactive Simulacra of Human Behavior [Park et al, 2023]

Generative Agents

Reflection: additional memory that is synthesized from previous memories (generated periodically)

- Prompt LLM with 100 most recent observations
- Use to to generate 3 questions from which relevant memories are extracted
- LLM then prompted to extract insights from the memories

```
Statements about Klaus Mueller
1. Klaus Mueller is writing a research paper
2. Klaus Mueller enjoys reading a book
on gentrification
3. Klaus Mueller is conversing with Ayesha Khan
about exercising [...]
What 5 high-level insights can you infer from
the above statements? (example format: insight
(because of 1, 5, 3))
```

Generative Agents: Interactive Simulacra of Human Behavior [Park et al, 2023]

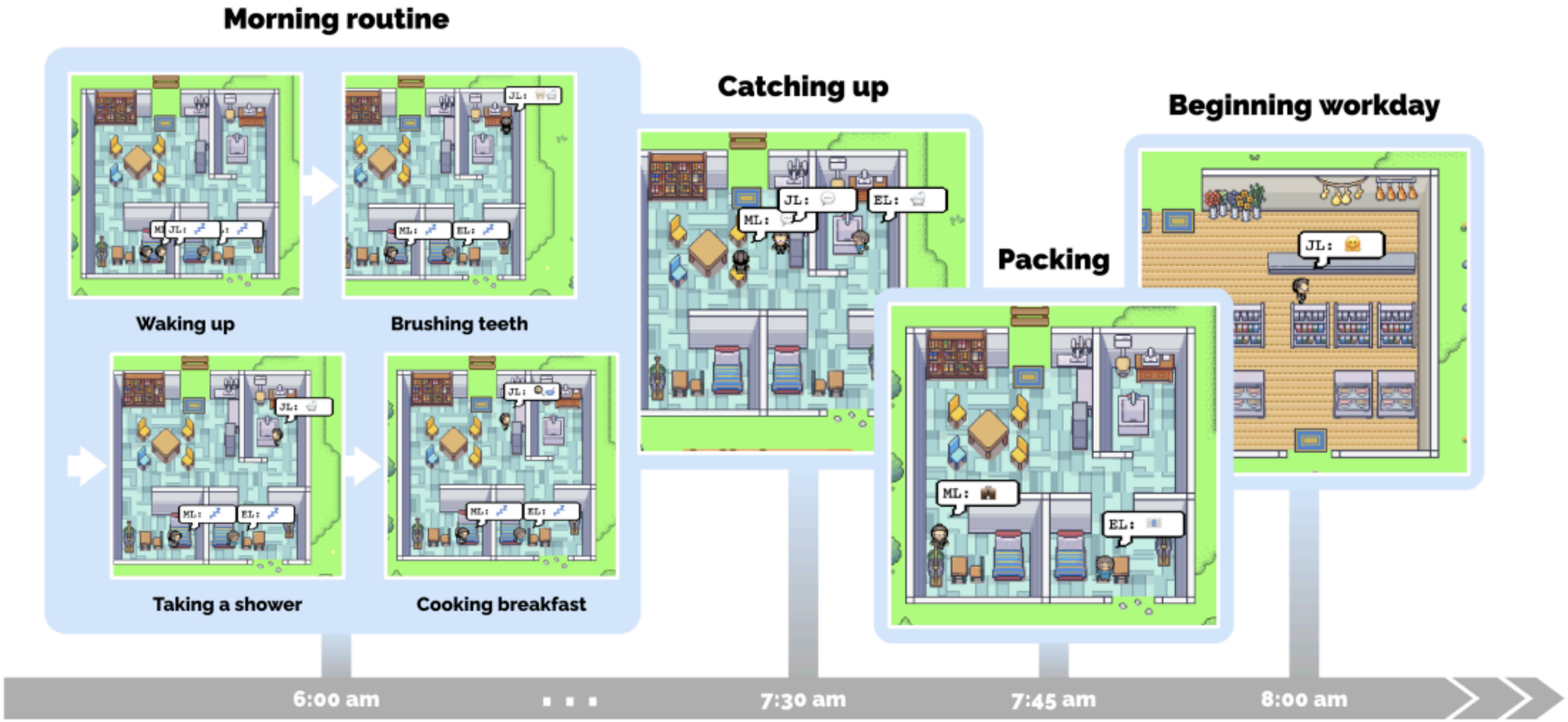
Generative Agents

Planning and reacting: converts memories and observations into actions

- Generates rough-plan from agent's summary description and summary of previous day and has LLM complete is
- Converse as they interact with each other (conditioned on memories about each other)
- LLM then prompted to extract insights from the memories

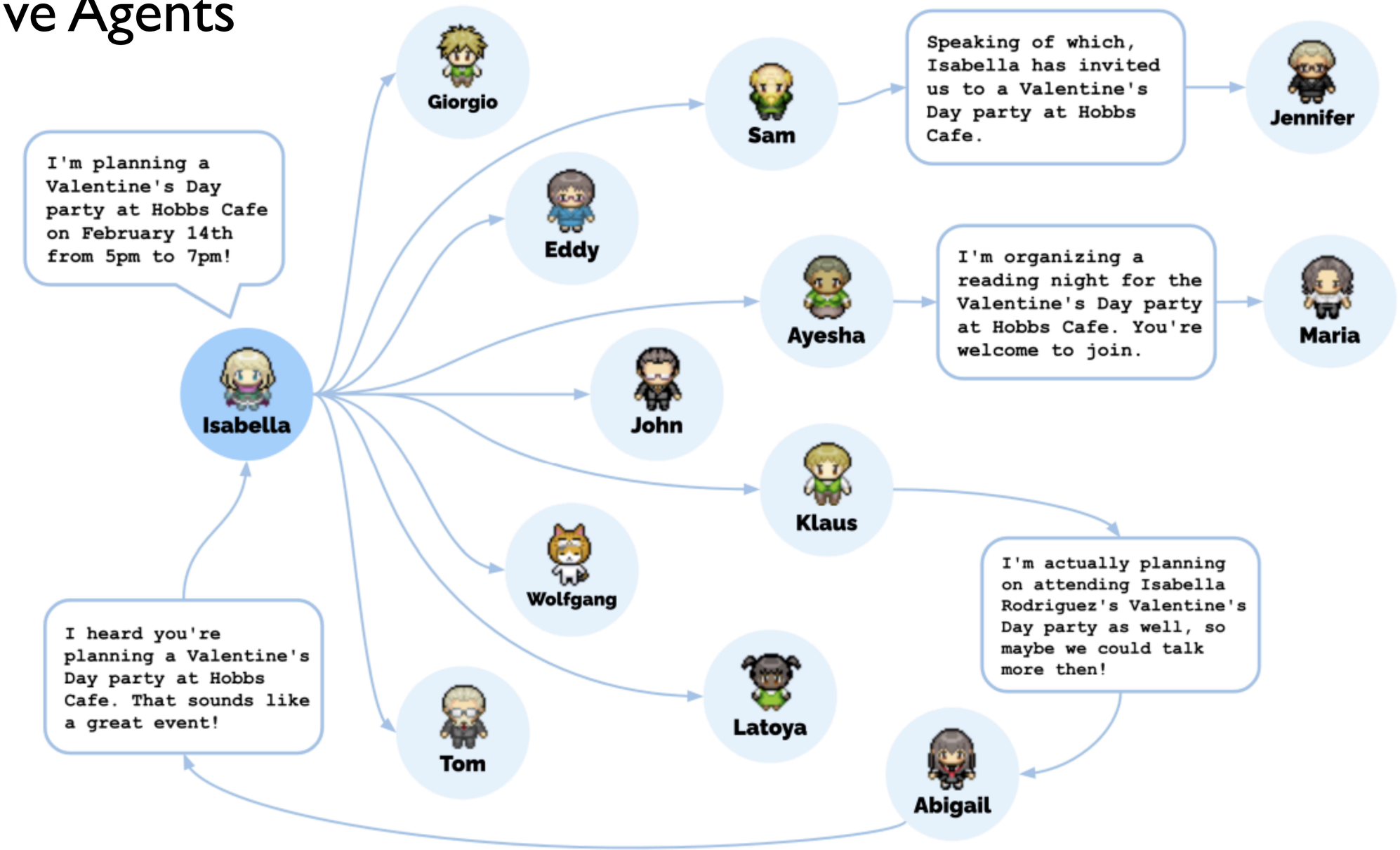
Generative Agents: Interactive Simulacra of Human Behavior [Park et al, 2023]

Generative Agents



Generative Agents: Interactive Simulacra of Human Behavior [Park et al, 2023]

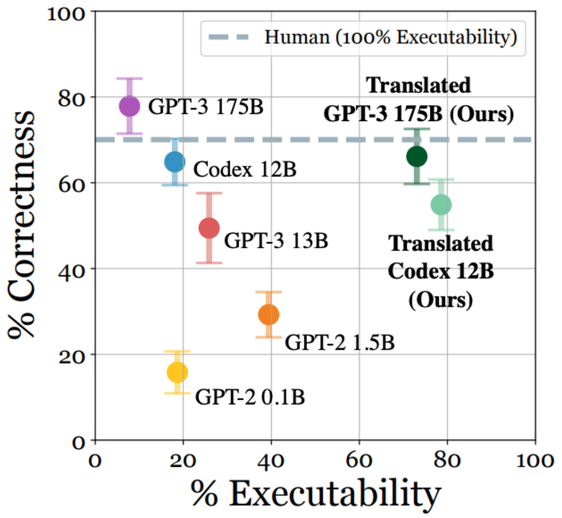
Generative Agents



Generative Agents: Interactive Simulacra of Human Behavior [Park et al, 2023]

Agents in more realistic scenarios (robotics and embodied AI)

Planning via large language models



Human

Task: Throw away paper
Step 1: Walk to home office
Step 2: Walk to desk
Step 3: Find desk
Step 4: Turn to desk
Step 5: Find chair
Step 6: Sit on chair
Step 7: Find check
Step 8: Grab check
Step 9: Squeeze check
Step 10: Stand up
Step 11: Walk to trashcan
Step 12: Put check on trashcan

GPT-2 1.5B

Task: Brush teeth
Step 1: Go to bathroom

GPT-3 175B

Task: Throw away paper
Step 1: Walk to home office
Step 2: Walk to wastebasket
Step 3: Find wastebasket
Step 4: Grab wastebasket
Step 5: Walk to desk
Step 6: Drop paper in wastebasket

Codex 12B

Task: Brush teeth
Step 1: Walk to bathroom
Step 2: Walk to sink
Step 3: Find toothbrush
Step 4: Pick up toothbrush
Step 5: Put toothbrush in mouth
Step 6: Move brush around mouth for two minutes
Step 7: Spit out toothpaste and brush into sink
Step 8: Turn on water in sink and rinse brush for one minute
Step 9: Turn off water in sink and return brush to cupboard

Translated GPT-3 175B

Task: Throw away paper
Step 1: Walk to home office
Step 2: Walk to table
Step 3: Find table
Step 4: Turn to table
Step 5: Find paper
Step 6: Grab paper
Step 7: Walk to trashcan
Step 8: Open trashcan
Step 9: Put paper on trashcan
Step 10: Close trashcan

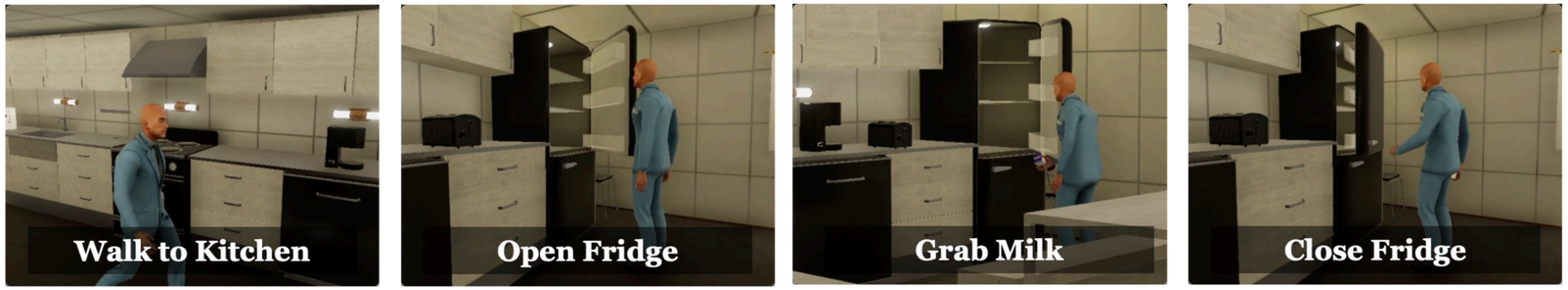
Translated Codex 12B

Task: Brush teeth
Step 1: Walk to bathroom
Step 2: Open door
Step 3: Walk to sink
Step 4: Put pot on sink
Step 5: Put brush on toothbrush
Step 6: Turn to toothpaste
Step 7: Put toothpaste on toothbrush
Step 8: Put teeth on toothbrush

Throw away paper

Brush teeth

Task: Get Glass of Milk

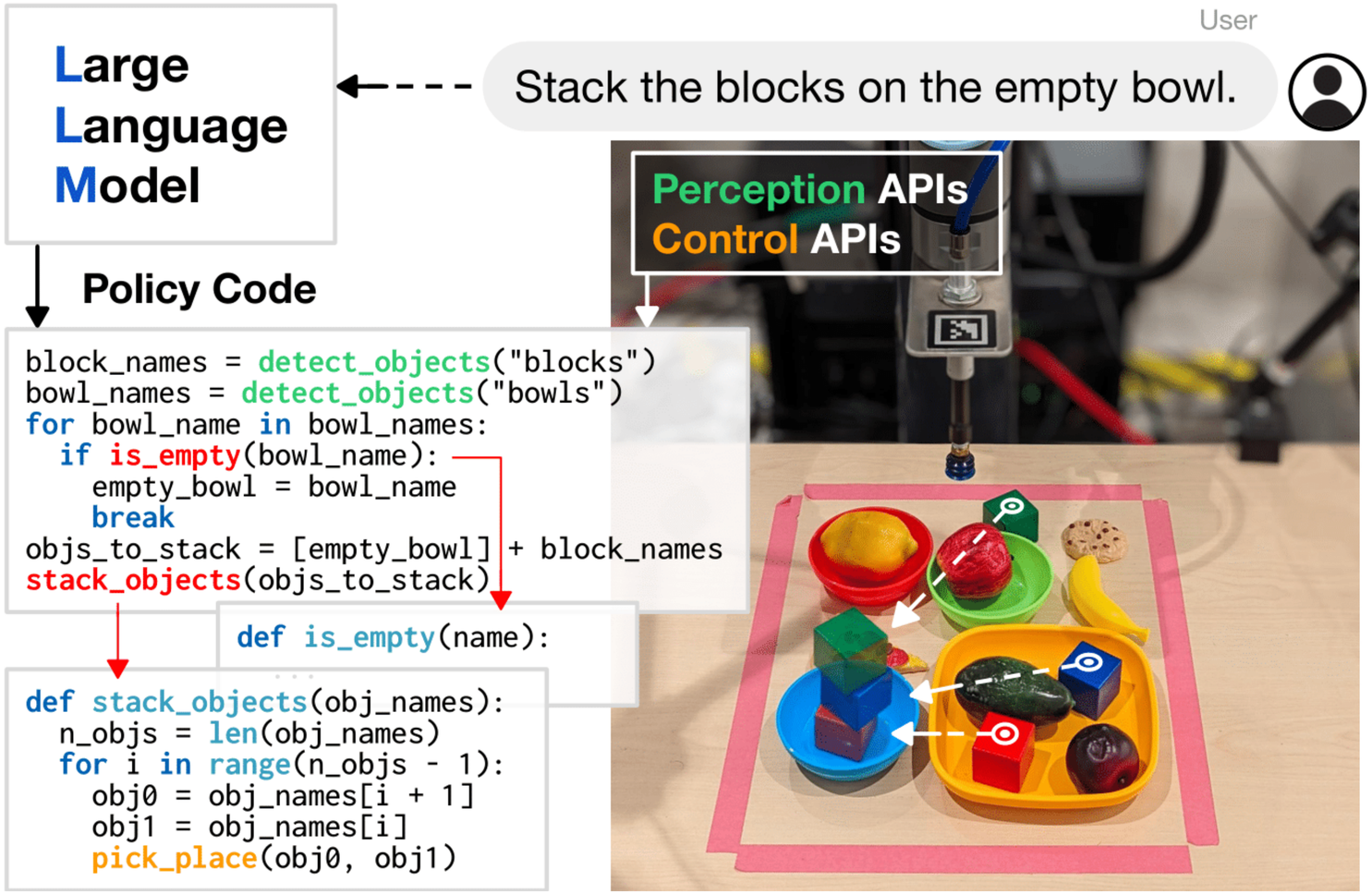


Get Glass of Milk

Language Models as Zero-Shot Planners: Extracting Actionable Knowledge for Embodied Agents [Huang et al. ICML 2022]

Berkeley UNIVERSITY OF CALIFORNIA CMU https://wenlong.page/language-planner/ Google Brain

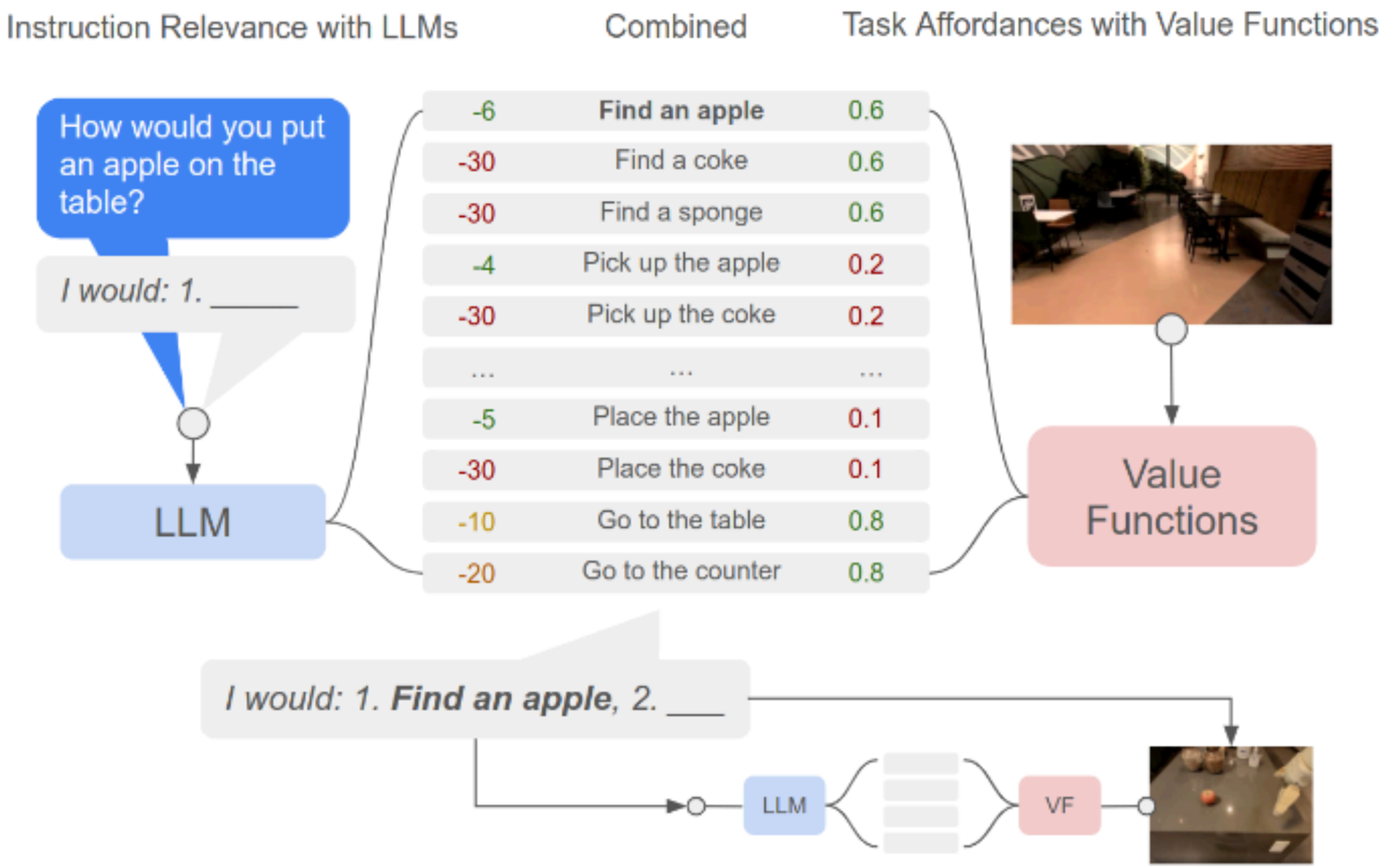
Control by code generation using LLMs



Code as Policies: Language Model Programs for Embodied Control [Liang et al. 2022]
https://code-as-policies.github.io/

Google

Combining perception with planning



Use perception to determine what is possible

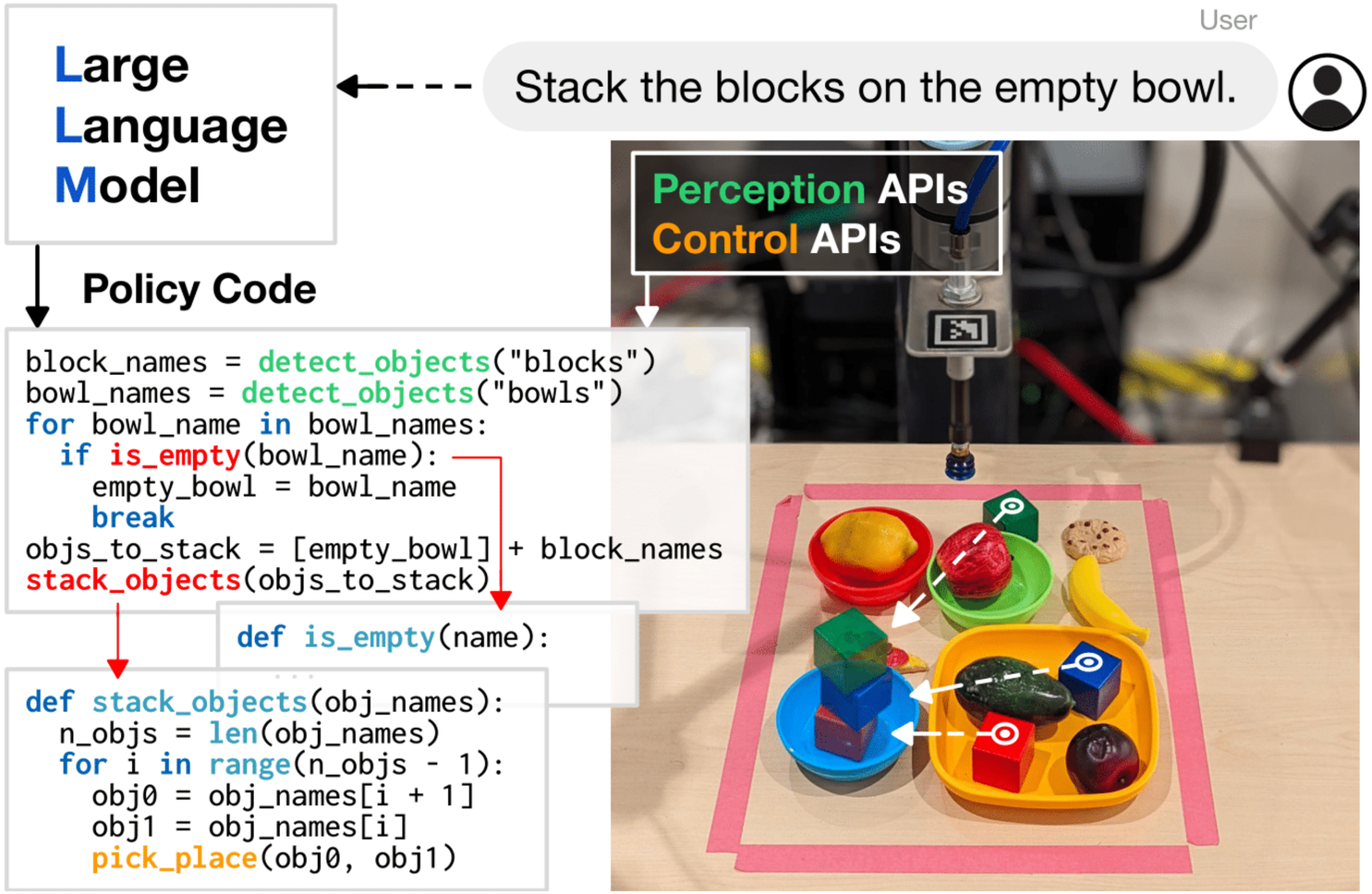
Do As I Can, Not As I Say: Grounding Language in Robotic Affordances [Ahn et al. CORL 2022]

https://say-can.github.io/

Everyday Robots

Google

Control by code generation using LLMs



Code as Policies: Language Model Programs for Embodied Control [Liang et al. 2022]
https://code-as-policies.github.io/

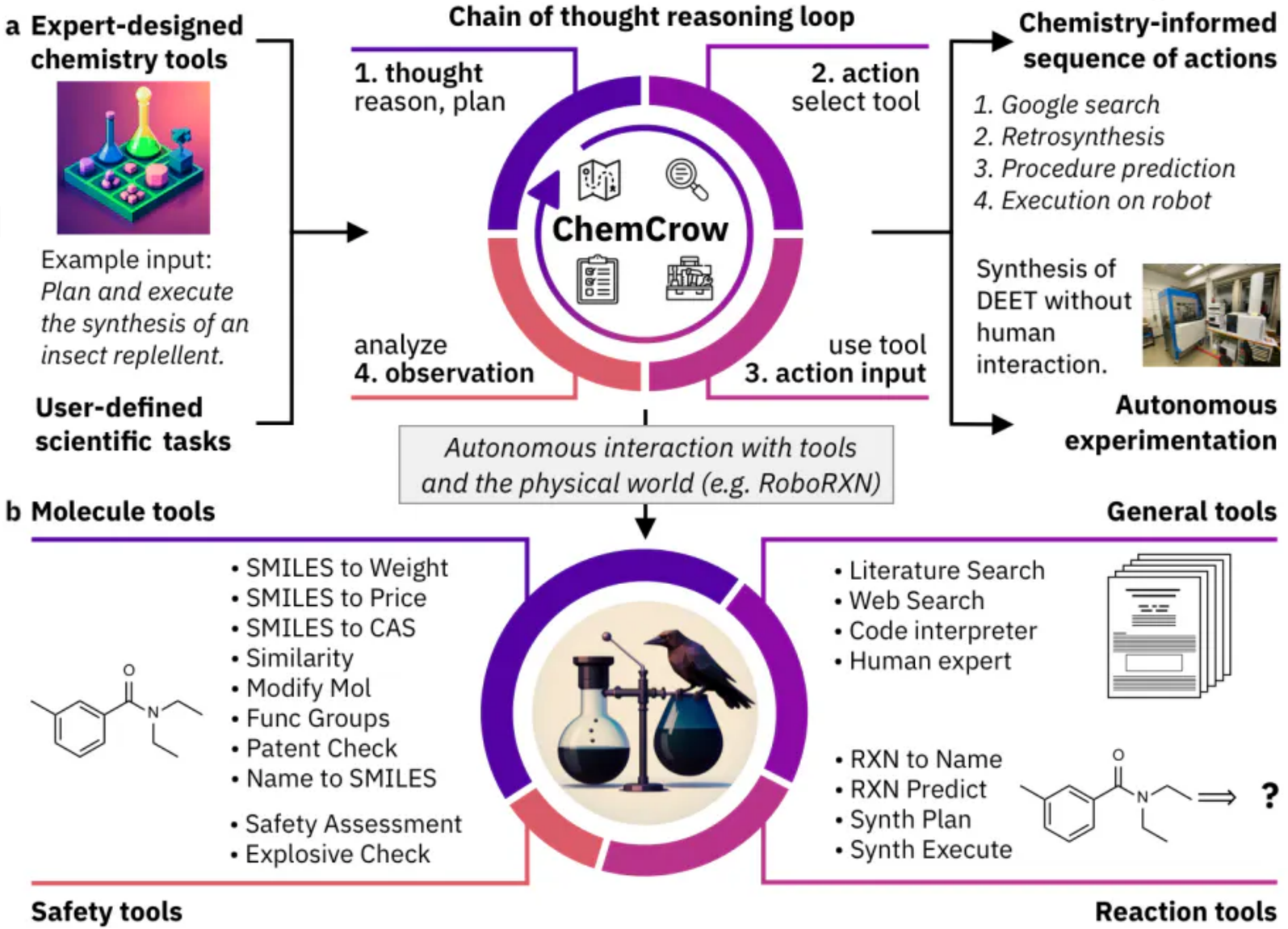
Google

Practical applications

ChemCrow: LLM powered chemistry engine

List of tools with name, description, expected input and output

Use ReAct style prompting: thought, action, action input, observation



ChemCrow: Augmenting large-language models with chemistry tools [Bran et al, 2023]