Lecture 12.

Language and Schema of Simulations

CS 222: AI Agents and Simulations
Stanford University

Joon Sung Park

Announcement

- Proposal Presentation Dates Announced!
- Please check the Announcements tab on Canvas for details. A template for the written proposal will be available soon.
- Presentation Format
  - 6-minute presentation
  - 2-minute Q&A
- Things to cover:
  - Motivation
  - Method
  - Hypothesis

Today:
How can we create tools to build simulations?

Importance of the right representation

H/T to Michael S. Bernstein, CS347 Lecture on Visualization

Cognitive amplification

- Visualization can help, but ultimately this power comes from better representation. By better understanding human cognition, we can design technology that makes us smarter.

- "The powers of cognition come from abstraction and representation: the ability to represent perceptions, experiences, and thoughts in some medium other than that in which they have occurred, abstracted away from irrelevant details." [Norman '94, Simon '81]

Simon, H. A. (1981). The Sciences of the Artificial (2nd ed.). MIT Press, p. 153.
Norman, D. A. (1994). Things That Make Us Smart: Defending Human Attributes in the Age of the Machine. Perseus Books, p. 47.

Example: Number scrabble [Simon 1988]

- **Take turns picking numbers in 1,2,3,4,5,6,7,8,9 without replacement**

- **Win if any *three* of your numbers add up to 15.**

- **It’s OK if you have extra numbers in your hand, as long as three of them add up to exactly 15.**

Simon, H. A. (1988). "The Science of Design: Creating the Artificial." Design Issues, 4(1/2), pp. 67-82.

Ready, set, go!

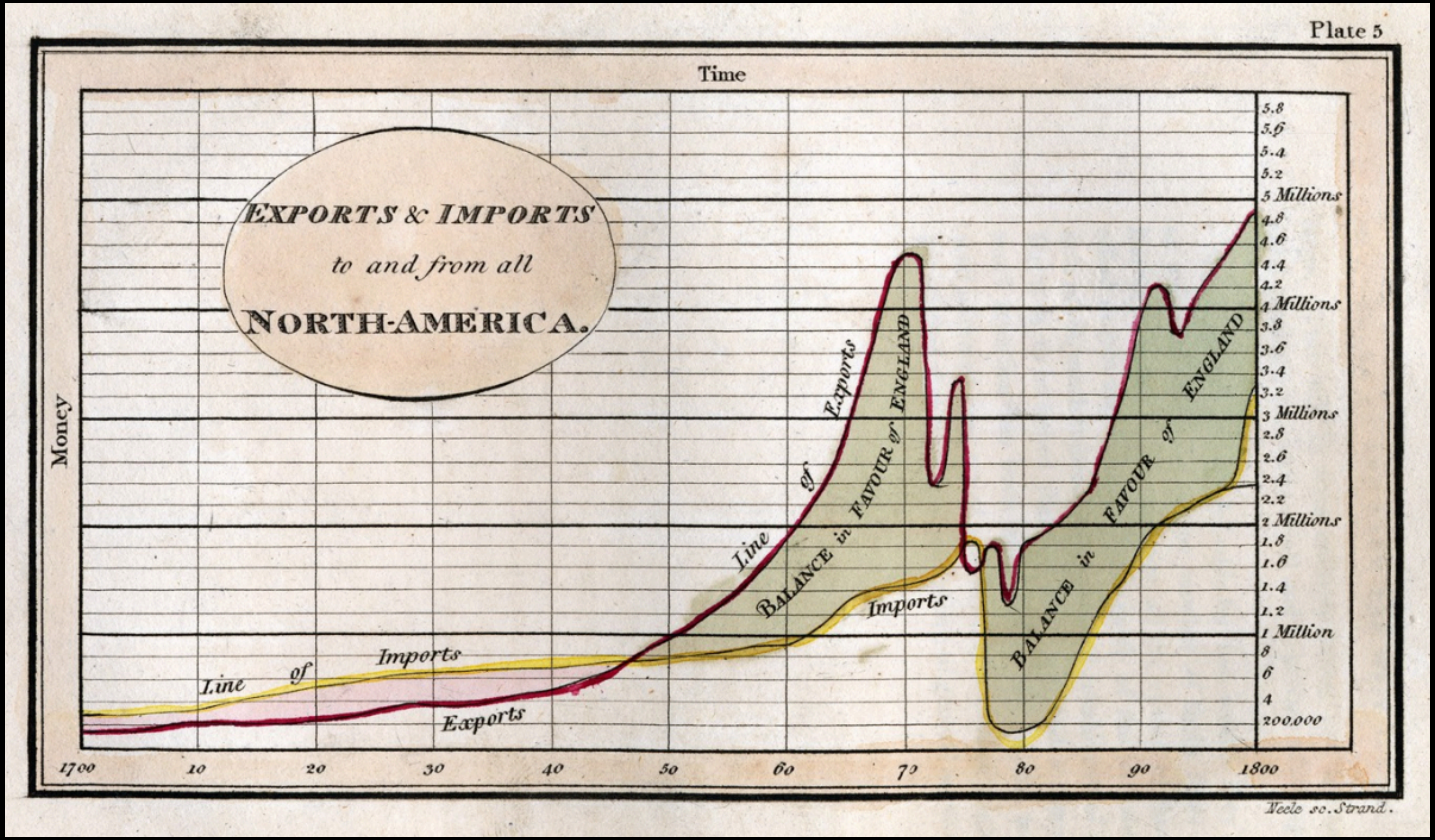
- I will show the series of moves from players A and B so far. Raise your hand when you know what B's best next move should be.

- A takes 4
  B takes 9
  A takes 2
  B takes 8
  A takes 5
  What should B do?

Re-encoding number scrabble

A 4	**B** 9	**A** 2
3	**A** 5	7
B 8	1	6

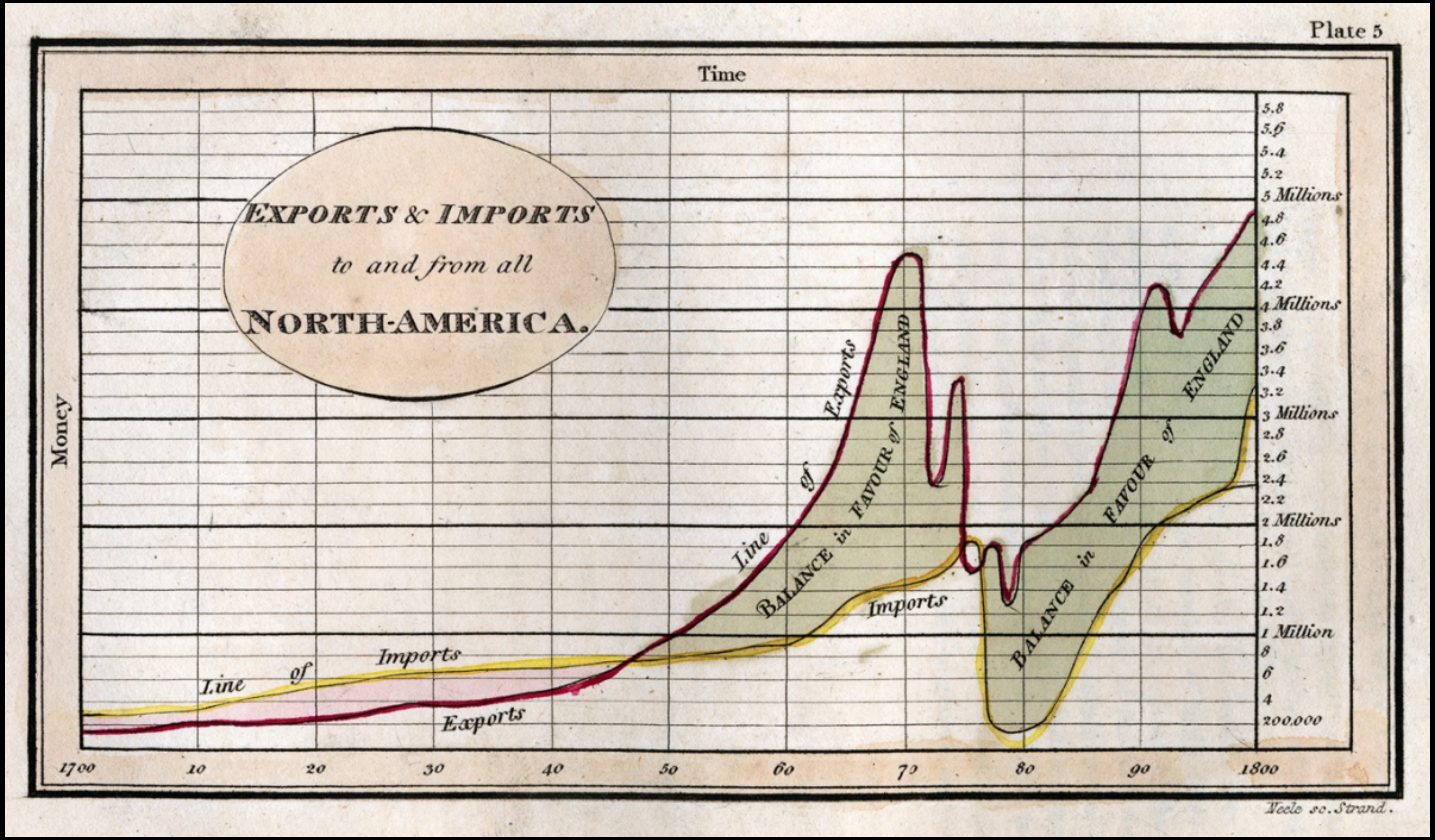
Representation: Changing representation to spatial tic-tac-toe board facilitates choice



Exports and imports to and from all North America [Playfair 1786-1801]

User's task: Understand balance of trade between England and North America over time

Playfair, W. (1786). The Commercial and Political Atlas: Representing, by Means of Stained Copper-Plate Charts, the Progress of the Commerce, Revenues, Expenditure and Debts of England during the Whole of the Eighteenth Century. London: J. Debrett.



Exports and imports to and from all North America [Playfair 1786-1801]

Important information: Historical differences between exports and imports
Representation: Superimpose line charts of exports and imports to show historical pattern. Shade differences between lines to highlight balance against/in favor

On the theory of
scales of measurements
S. S. Stevens, 1946

Data Types

- **N - Nominal (labels)**
  Fruits: Apples, oranges…
  Operations: =, ≠

- **O - Ordered**
  Quality of eggs: Grade AA, A, B
  Operations: =, ≠, <, >

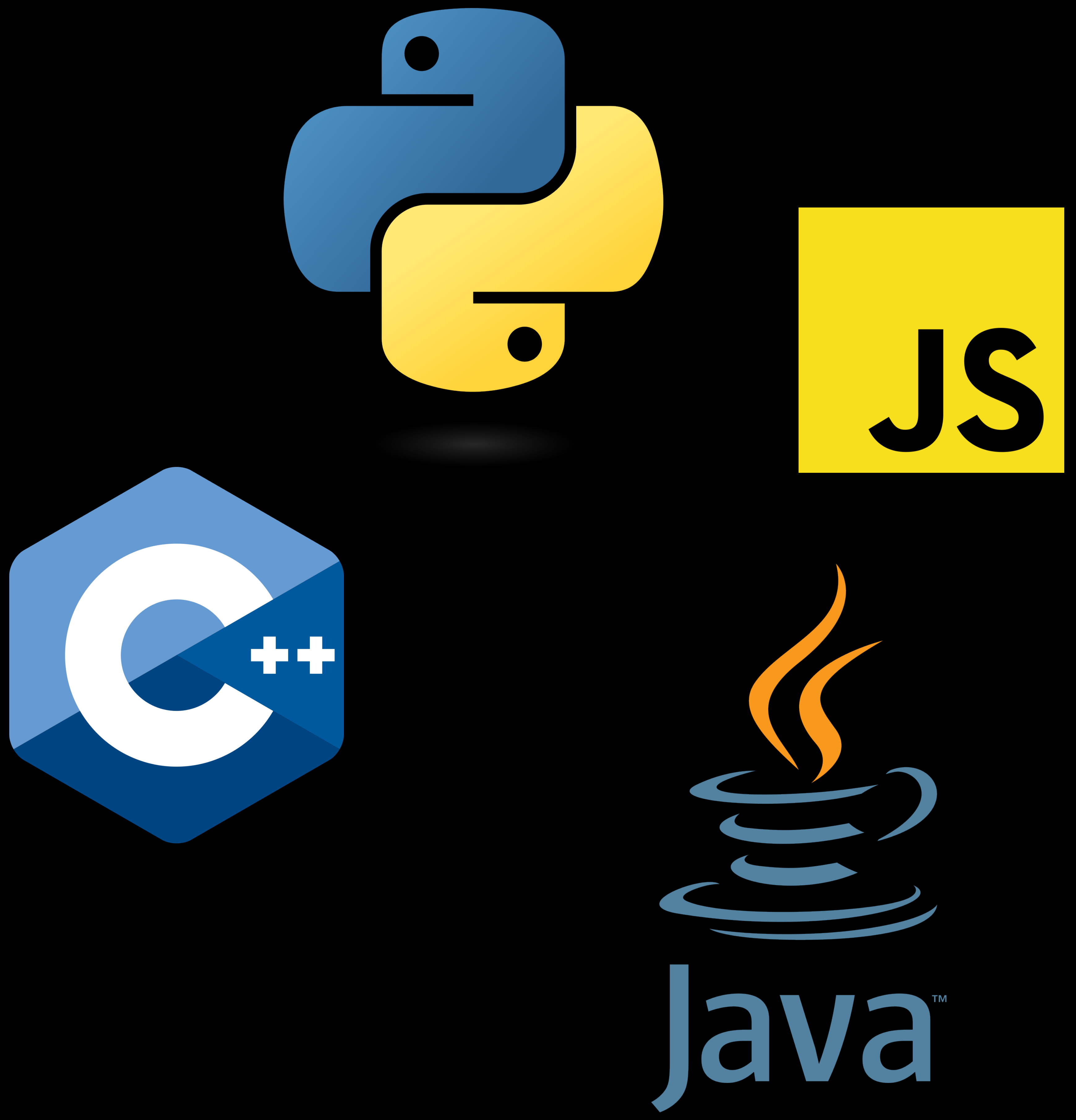
- **Q - Interval (location of zero arbitrary)**
  Dates: Jan, 19, 2016; Loc.:(LAT 33.98, LON-| | 8.45)
  Like a geometric point. Cannot compare directly
  Only differences (i.e. intervals) may be compared
  Operations: =, ≠, <, >, -

- **Q - Ratio (location of zero fixed)**
  Physical measurement: Length, Mass, …
  Counts and amounts
  Like a geometric vector, origin is meaningful
  Operations: =, ≠, <, >, -, ÷

Language and schema offer a point of view

Languages — where have you seen them?

- Programming languages are formal languages comprising a set of instructions that can be used to produce various types of output.

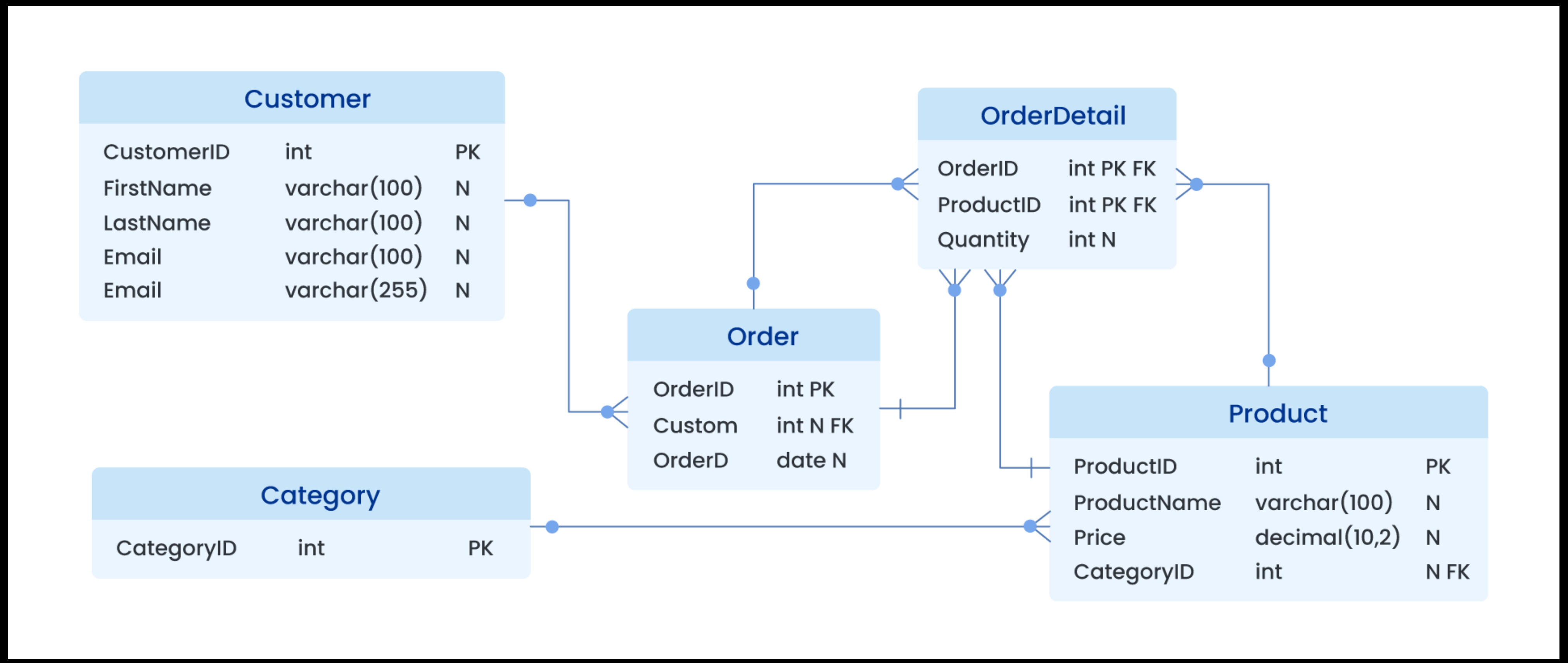


Languages — where have you seen them?

- Domain-Specific Languages (DSLs) are specialized programming languages designed for a specific domain, offering syntax and functions tailored to particular tasks.

Schemas — where have you seen them?

- Schemas define a structured layout or format for data, describing relationships between data types and fields



Customer		
CustomerID	int	PK
FirstName	varchar(100)	N
LastName	varchar(100)	N
Email	varchar(100)	N
Email	varchar(255)	N

OrderDetail	
OrderID	int PK FK
ProductID	int PK FK
Quantity	int N

Order	
OrderID	int PK
Custom	int N FK
OrderD	date N

Product		
ProductID	int	PK
ProductName	varchar(100)	N
Price	decimal(10,2)	N
CategoryID	int	N FK

Category		
CategoryID	int	PK

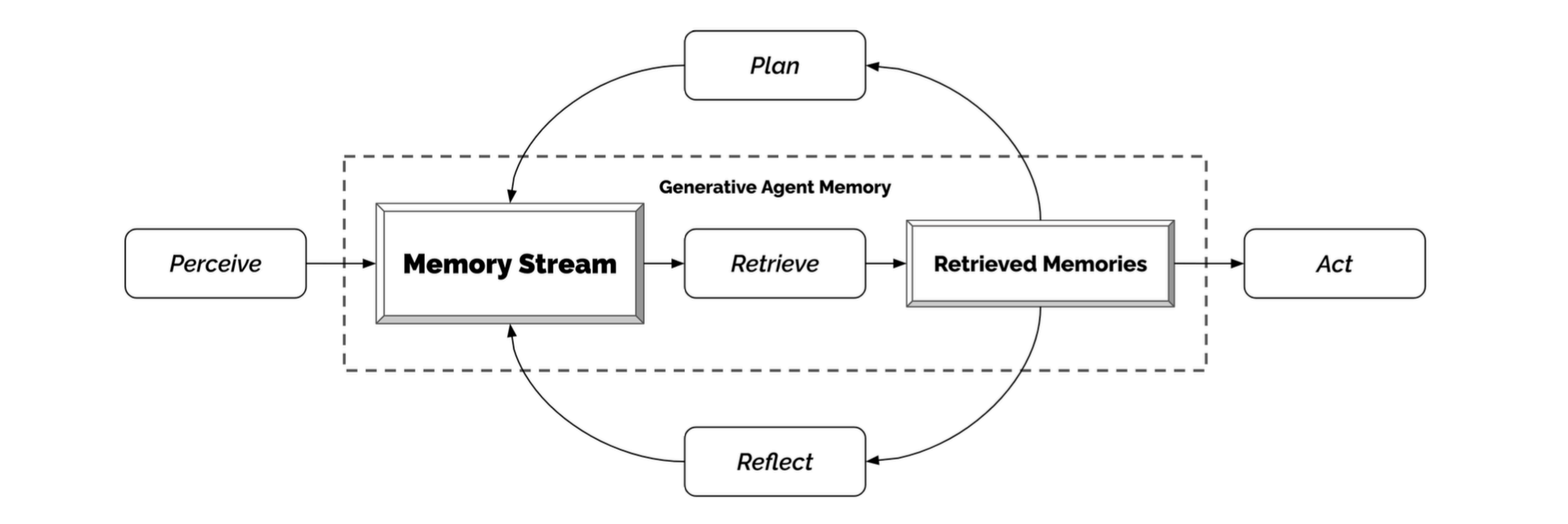
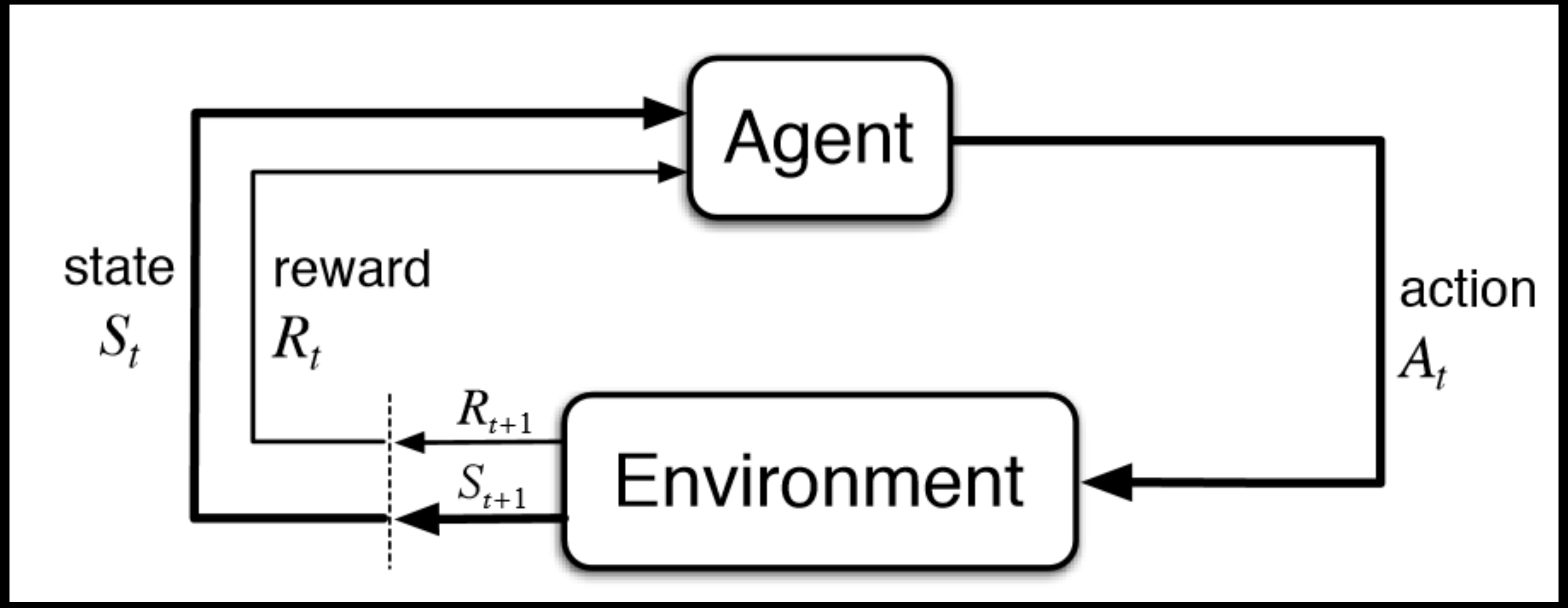


Figure 5: Our generative agent architecture. Agents perceive their environment, and all perceptions are saved in a comprehensive record of the agent's experiences called the memory stream. Based on their perceptions, the architecture retrieves relevant memories and uses those retrieved actions to determine an action. These retrieved memories are also used to form longer-term plans and create higher-level reflections, both of which are entered into the memory stream for future use.



Language and schema offer a perspective on the future we are heading toward.

The clearer and more prescient this perspective is, the more powerful the language and schema become.

SQL: We ought to support dynamic data manipulation that can be authored by non-experts in SQL.

- SQL was developed as a declarative language, allowing users to specify what data they want without dictating how to retrieve it.
  - This design simplifies data retrieval for users who may not be experts in database optimization.

- SQL provides fundamental operations—Create, Read, Update, Delete (CRUD)—to manage data.
  - CRUD operations give SQL the flexibility needed for *dynamic data manipulation*. This approach makes SQL adaptable, allowing users to manage data throughout its lifecycle within a single language.

Chamberlin, D. D., & Boyce, R. F. (1974). "SEQUEL: A Structured English Query Language." Proceedings of the 1974 ACM SIGFIDET Workshop on Data Description, Access and Control, pp. 249-264.

HTML5: we ought to support multimedia content and accommodate various screen sizes.

- HTML5 introduced native <audio> and <video> elements to support multimedia content directly within the browser, without requiring plugins like Flash.
  - These elements were added to streamline multimedia integration, reduce dependency on third-party plugins, and improve performance on mobile and desktop devices. Native support makes multimedia content more accessible and SEO-friendly.

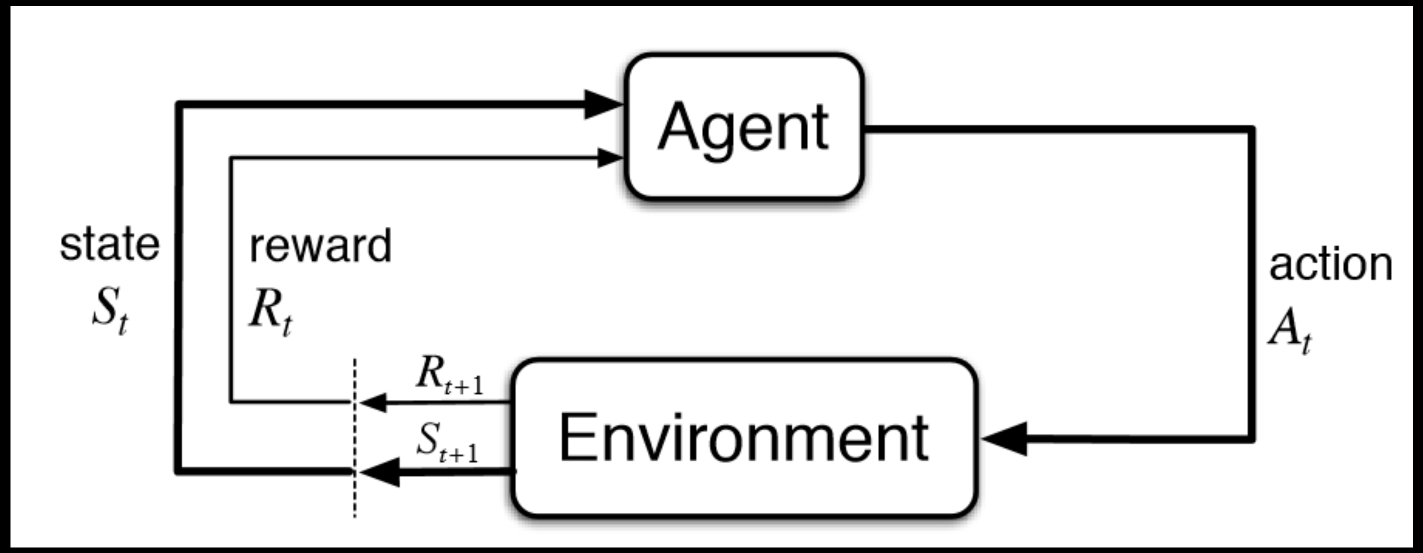
"New open standards created in the mobile era, such as HTML5, will win on mobile devices"

- Steve Jobs, 2010

- HTML5 supports responsive design principles, enabling web pages to adapt to various screen sizes and devices.
  - With the rise of mobile devices, consistency across screens became essential. HTML5's responsive capabilities allow websites to deliver a seamless experience across desktops, tablets, and smartphones.

Hickson, I., et al. (2014). "HTML5: A vocabulary and associated APIs for HTML and XHTML." W3C Recommendation. World Wide Web Consortium (W3C).

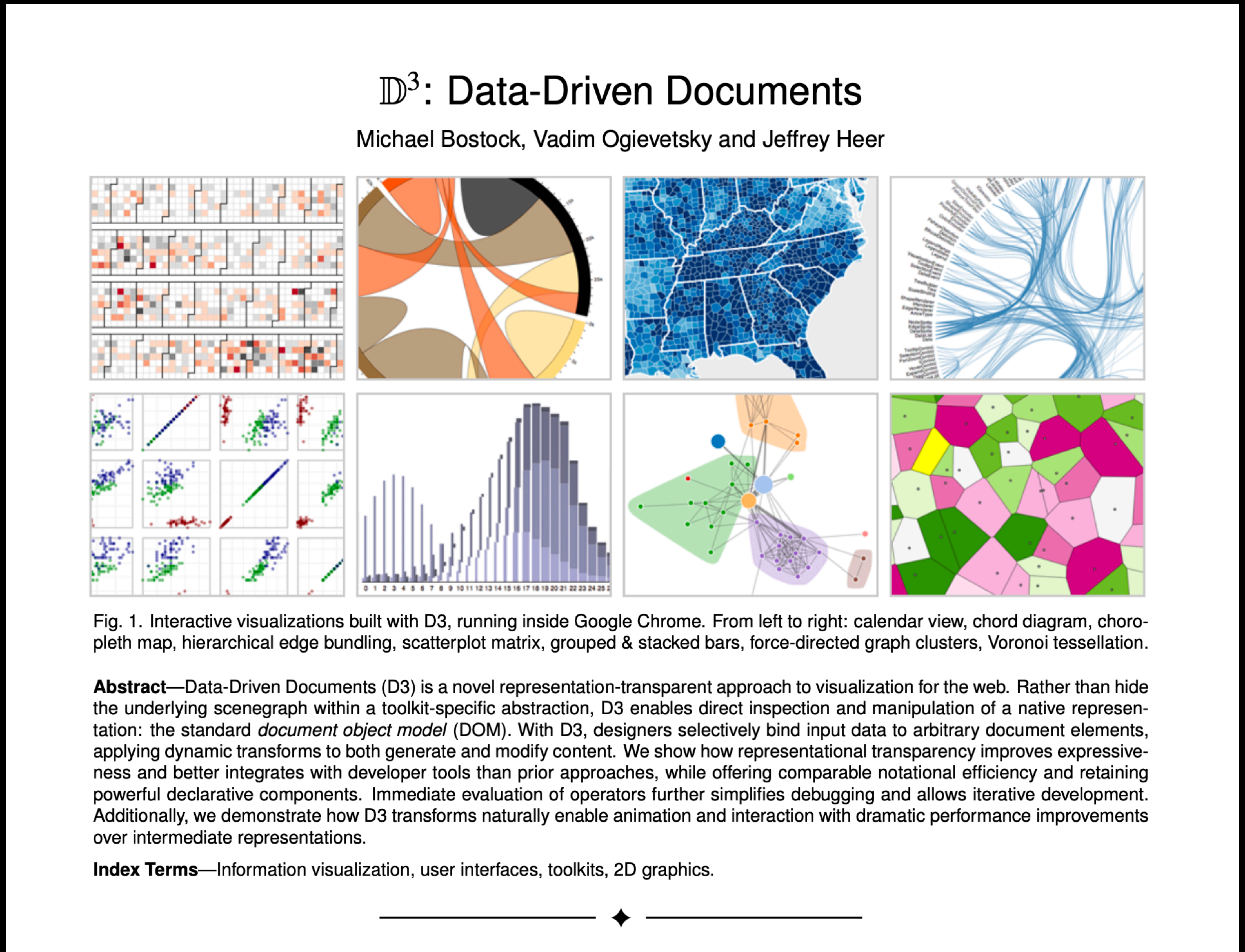
RL: agents should be described as an interplay between the environment and their actions, with agents receiving rewards from the environment.



- The diagram clearly separates the agent and the environment as distinct entities, often showing two main blocks.
  - This separation highlights the core RL concept that the agent interacts with an environment, learning from the outcomes of its actions.
- Arrows are typically used to show the action going from the agent to the environment and reward and/or observation/state coming back to the agent.
  - Distinct arrows reinforce the idea of causality and feedback central to RL.

Sutton, R. S., & Barto, A. G. (1998). "Reinforcement Learning: An Introduction." MIT Press, Cambridge, MA.

Discussion 1. D3



$\mathbb{D}^3$: Data-Driven Documents

Michael Bostock, Vadim Ogievetsky and Jeffrey Heer

Fig. 1. Interactive visualizations built with D3, running inside Google Chrome. From left to right: calendar view, chord diagram, choropleth map, hierarchical edge bundling, scatterplot matrix, grouped & stacked bars, force-directed graph clusters, Voronoi tessellation.

Abstract—Data-Driven Documents (D3) is a novel representation-transparent approach to visualization for the web. Rather than hide the underlying scenegraph within a toolkit-specific abstraction, D3 enables direct inspection and manipulation of a native representation: the standard *document object model* (DOM). With D3, designers selectively bind input data to arbitrary document elements, applying dynamic transforms to both generate and modify content. We show how representational transparency improves expressiveness and better integrates with developer tools than prior approaches, while offering comparable notational efficiency and retaining powerful declarative components. Immediate evaluation of operators further simplifies debugging and allows iterative development. Additionally, we demonstrate how D3 transforms naturally enable animation and interaction with dramatic performance improvements over intermediate representations.

Index Terms—Information visualization, user interfaces, toolkits, 2D graphics.

Bostock, M., Ogievetsky, V., & Heer, J. (2011). D3: Data-Driven Documents. IEEE Transactions on Visualization and Computer Graphics, 17(12), 2301-2309. [pdf]

Discussion 2. Generative Agents

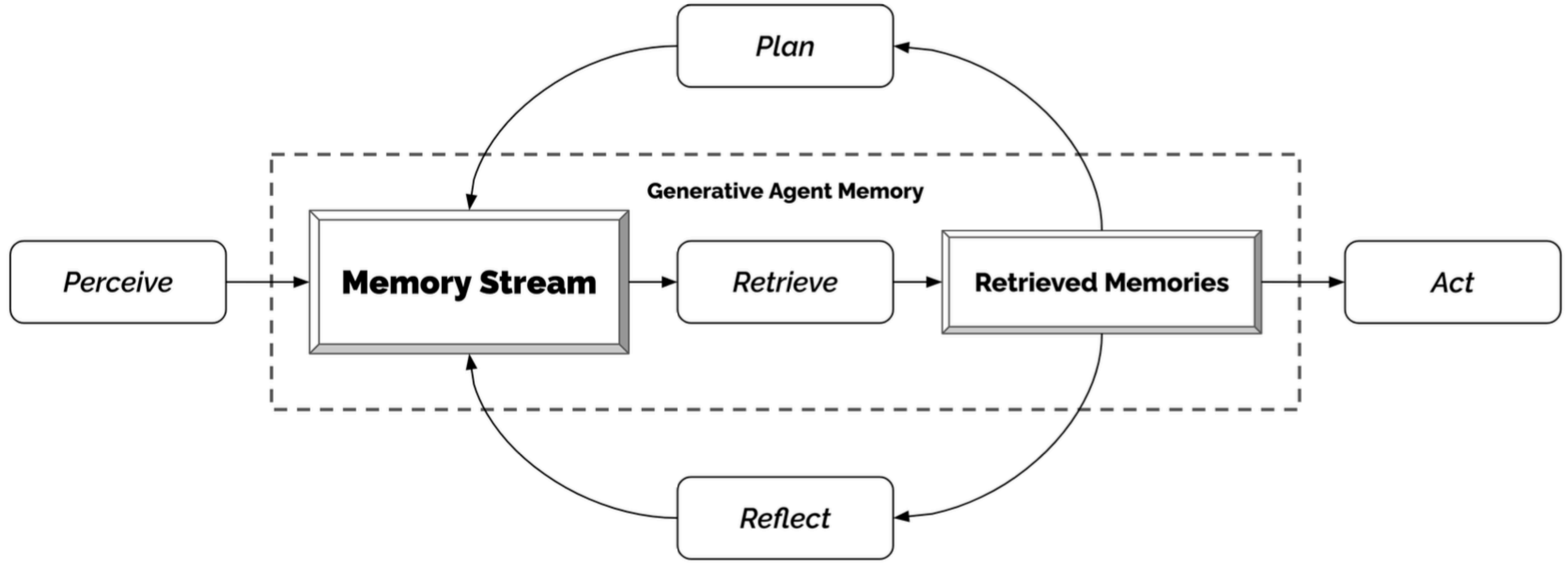
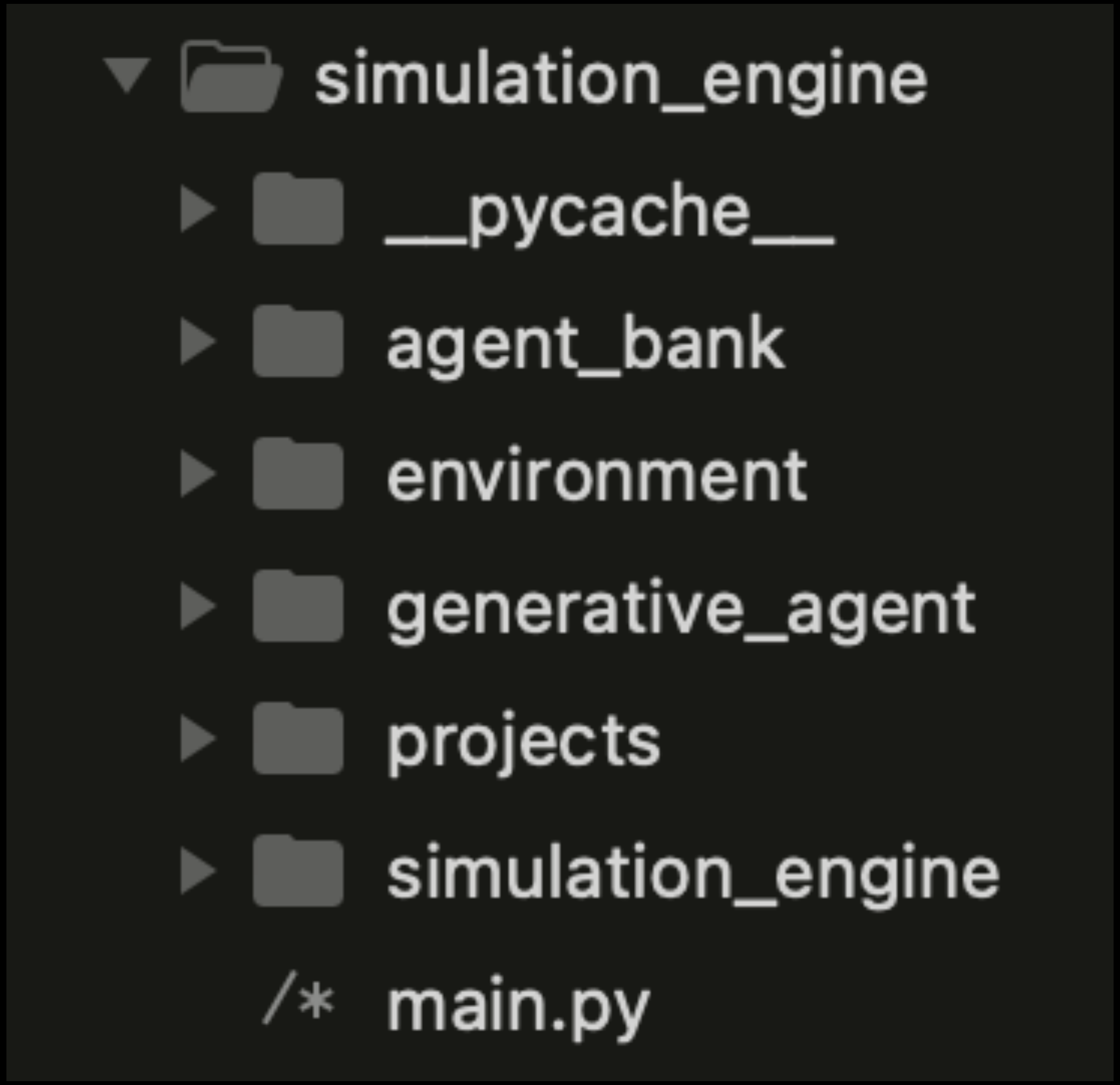


Figure 5: Our generative agent architecture. Agents perceive their environment, and all perceptions are saved in a comprehensive record of the agent's experiences called the memory stream. Based on their perceptions, the architecture retrieves relevant memories and uses those retrieved actions to determine an action. These retrieved memories are also used to form longer-term plans and create higher-level reflections, both of which are entered into the memory stream for future use.

J. S. Park, J. C. O'Brien, C. J. Cai, M. R. Morris, P. Liang, M. S. Bernstein, Generative agents: Interactive simulacra of human behavior, in Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology (ACM, 2023).

Agent, population, environment

Assignment 1 repo



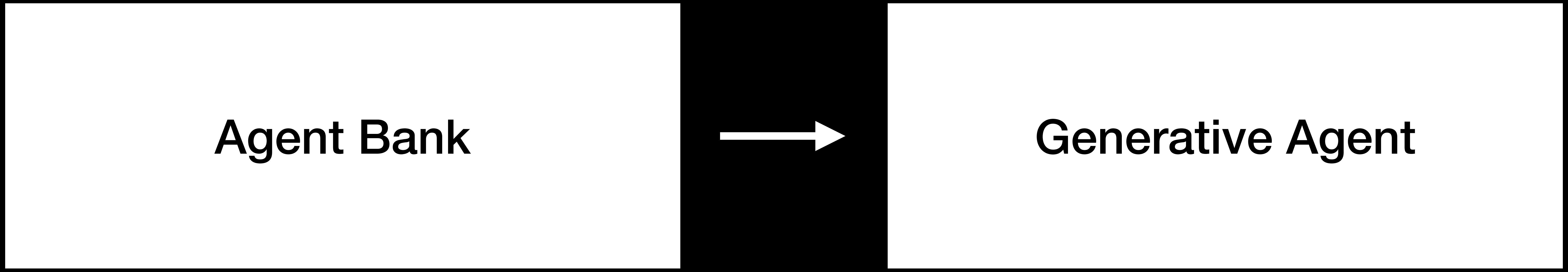
```
▼ simulation_engine
    ▶ __pycache__
    ▶ agent_bank
    ▶ environment
    ▶ generative_agent
    ▶ projects
    ▶ simulation_engine
    /* main.py
```

Assignment 1 repo offers the following point of view

How can we provide representations for creating robust and replicable simulations?

- **We should freeze the “agent memories and architectures.”**
- **We should standardize the implementation of the environment.**

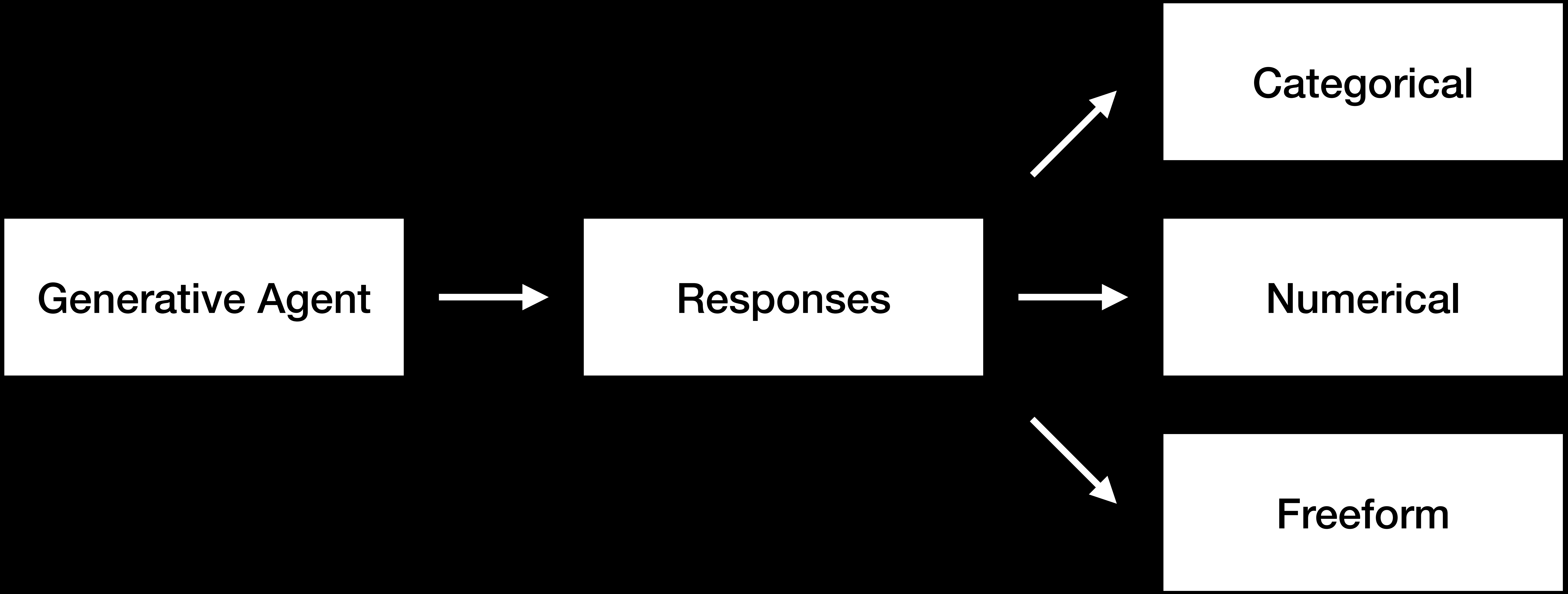
Assignment 1 repo



Agent Bank

Generative Agent

Assignment 1 repo



Generative Agent

Responses

Categorical

Numerical

Freeform

Environment

Survey

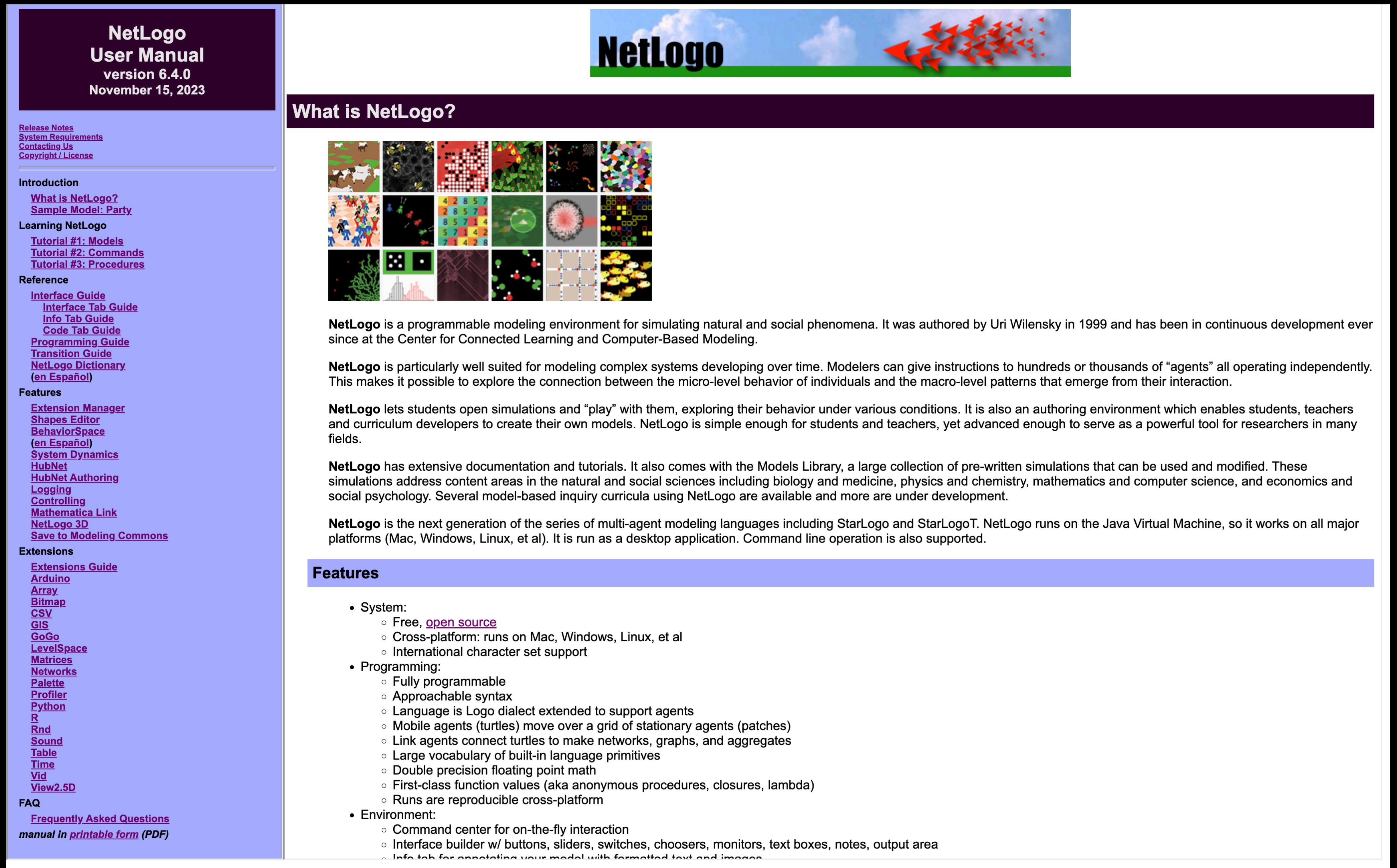
Interview

Network

Primary repo:

https://github.com/joonspk-research/gabm-stanford-main

Other examples of languages and schemas for simulations



**NetLogo
User Manual
version 6.4.0
November 15, 2023**

- Release Notes
- System Requirements
- Contacting Us
- Copyright / License

Introduction
- What is NetLogo?
- Sample Model: Party

Learning NetLogo
- Tutorial #1: Models
- Tutorial #2: Commands
- Tutorial #3: Procedures

Reference
- Interface Guide
  - Interface Tab Guide
  - Info Tab Guide
  - Code Tab Guide
- Programming Guide
- Transition Guide
- NetLogo Dictionary
(en Español)

Features
- Extension Manager
- Shapes Editor
- BehaviorSpace
(en Español)
- System Dynamics
- HubNet
- HubNet Authoring
- Logging
- Controlling
- Mathematica Link
- NetLogo 3D
- Save to Modeling Commons

Extensions
- Extensions Guide
- Arduino
- Array
- Bitmap
- CSV
- GIS
- GoGo
- LevelSpace
- Matrices
- Networks
- Palette
- Profiler
- Python
- R
- Rnd
- Sound
- Table
- Time
- Vid
- View2.5D

FAQ
- Frequently Asked Questions

manual in printable form (PDF)

What is NetLogo?

NetLogo is a programmable modeling environment for simulating natural and social phenomena. It was authored by Uri Wilensky in 1999 and has been in continuous development ever since at the Center for Connected Learning and Computer-Based Modeling.

NetLogo is particularly well suited for modeling complex systems developing over time. Modelers can give instructions to hundreds or thousands of "agents" all operating independently. This makes it possible to explore the connection between the micro-level behavior of individuals and the macro-level patterns that emerge from their interaction.

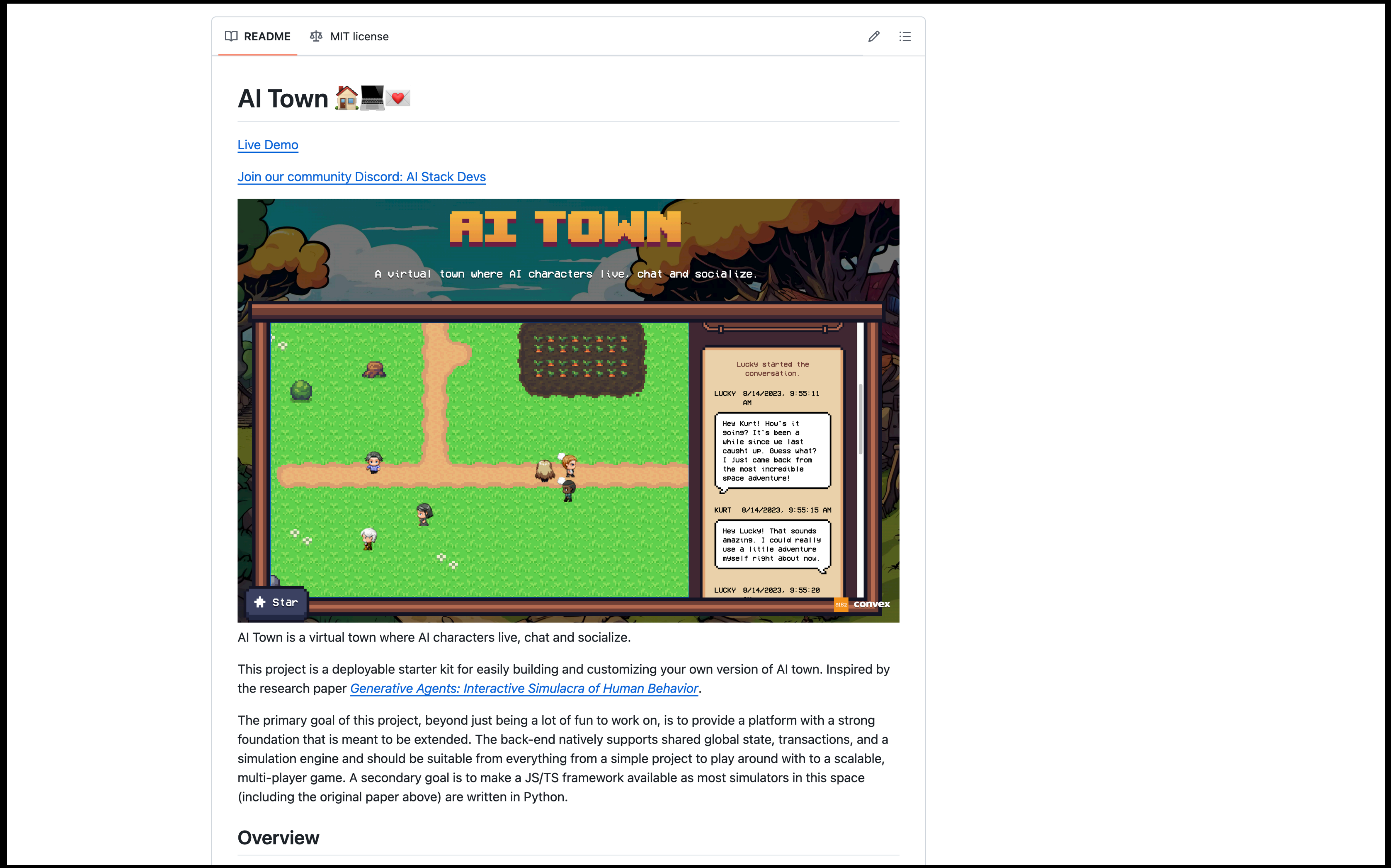
NetLogo lets students open simulations and "play" with them, exploring their behavior under various conditions. It is also an authoring environment which enables students, teachers and curriculum developers to create their own models. NetLogo is simple enough for students and teachers, yet advanced enough to serve as a powerful tool for researchers in many fields.

NetLogo has extensive documentation and tutorials. It also comes with the Models Library, a large collection of pre-written simulations that can be used and modified. These simulations address content areas in the natural and social sciences including biology and medicine, physics and chemistry, mathematics and computer science, and economics and social psychology. Several model-based inquiry curricula using NetLogo are available and more are under development.

NetLogo is the next generation of the series of multi-agent modeling languages including StarLogo and StarLogoT. NetLogo runs on the Java Virtual Machine, so it works on all major platforms (Mac, Windows, Linux, et al). It is run as a desktop application. Command line operation is also supported.

Features

- System:
  - Free, open source
  - Cross-platform: runs on Mac, Windows, Linux, et al
  - International character set support
- Programming:
  - Fully programmable
  - Approachable syntax
  - Language is Logo dialect extended to support agents
  - Mobile agents (turtles) move over a grid of stationary agents (patches)
  - Link agents connect turtles to make networks, graphs, and aggregates
  - Large vocabulary of built-in language primitives
  - Double precision floating point math
  - First-class function values (aka anonymous procedures, closures, lambda)
  - Runs are reproducible cross-platform
- Environment:
  - Command center for on-the-fly interaction
  - Interface builder w/ buttons, sliders, switches, choosers, monitors, text boxes, notes, output area



AI Town 🏠💻💌

Live Demo

Join our community Discord: AI Stack Devs

AI TOWN
A virtual town where AI characters live, chat and socialize.
Lucky started the conversation.
Star
a16z
convex

LUCKY 8/14/2023, 9:55:11 AM

Hey Kurt! How's it going? It's been a while since we last caught up. Guess what? I just came back from the most incredible space adventure!

KURT 8/14/2023, 9:55:15 AM

Hey Lucky! That sounds amazing. I could really use a little adventure myself right about now.

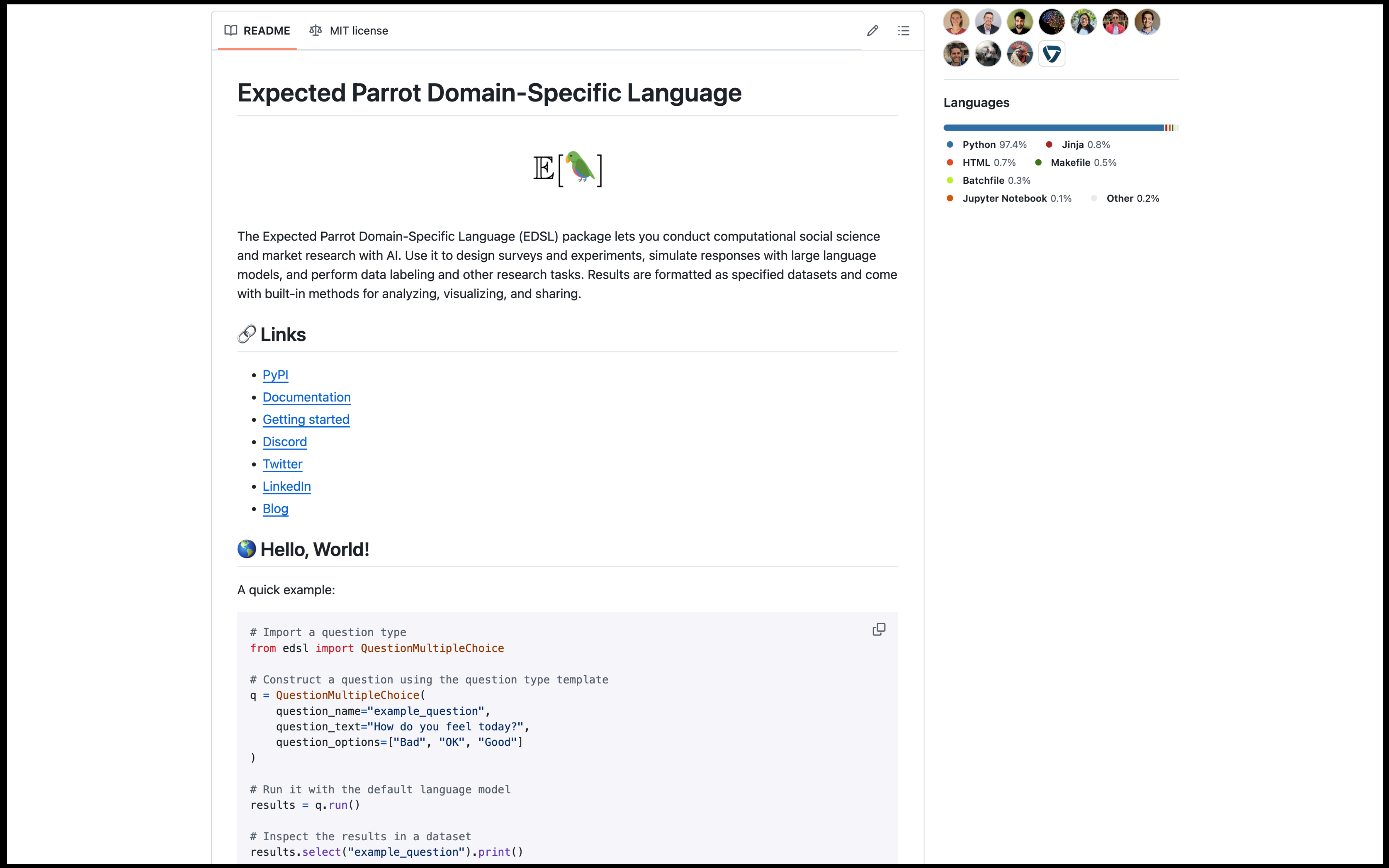
LUCKY 8/14/2023, 9:55:20

AI Town is a virtual town where AI characters live, chat and socialize.

This project is a deployable starter kit for easily building and customizing your own version of AI town. Inspired by the research paper *Generative Agents: Interactive Simulacra of Human Behavior*.

The primary goal of this project, beyond just being a lot of fun to work on, is to provide a platform with a strong foundation that is meant to be extended. The back-end natively supports shared global state, transactions, and a simulation engine and should be suitable from everything from a simple project to play around with to a scalable, multi-player game. A secondary goal is to make a JS/TS framework available as most simulators in this space (including the original paper above) are written in Python.

Overview



Expected Parrot Domain-Specific Language

The Expected Parrot Domain-Specific Language (EDSL) package lets you conduct computational social science and market research with AI. Use it to design surveys and experiments, simulate responses with large language models, and perform data labeling and other research tasks. Results are formatted as specified datasets and come with built-in methods for analyzing, visualizing, and sharing.

🔗 Links

- PyPI
- Documentation
- Getting started
- Discord
- Twitter
- LinkedIn
- Blog

🌎 Hello, World!

A quick example:

```
# Import a question type
from edsl import QuestionMultipleChoice

# Construct a question using the question type template
q = QuestionMultipleChoice(
    question_name="example_question",
    question_text="How do you feel today?",
    question_options=["Bad", "OK", "Good"]
)

# Run it with the default language model
results = q.run()

# Inspect the results in a dataset
results.select("example_question").print()
```

Languages

- ● Python 97.4%
- ● Jinja 0.8%
- ● HTML 0.7%
- ● Makefile 0.5%
- ● Batchfile 0.3%
- ● Jupyter Notebook 0.1%
- ● Other 0.2%

Quick demo

References

- Michael S. Bernstein, CS347 Lecture on Visualization
- Simon, H. A. (1981). The Sciences of the Artificial (2nd ed.). MIT Press, p. 153.
- Norman, D. A. (1994). Things That Make Us Smart: Defending Human Attributes in the Age of the Machine. Perseus Books, p. 47.
- Simon, H. A. (1988). "The Science of Design: Creating the Artificial." Design Issues, 4(1/2), pp. 67-82.
- Playfair, W. (1786). The Commercial and Political Atlas: Representing, by Means of Stained Copper-Plate Charts, the Progress of the Commerce, Revenues, Expenditure and Debts of England during the Whole of the Eighteenth Century. London: J. Debrett.
- Chamberlin, D. D., & Boyce, R. F. (1974). "SEQUEL: A Structured English Query Language." Proceedings of the 1974 ACM SIGFIDET Workshop on Data Description, Access and Control, pp. 249-264.
- Hickson, I., et al. (2014). "HTML5: A vocabulary and associated APIs for HTML and XHTML." W3C Recommendation. World Wide Web Consortium (W3C).

References

- Sutton, R. S., & Barto, A. G. (1998). "Reinforcement Learning: An Introduction." MIT Press, Cambridge, MA.
- Bostock, M., Ogievetsky, V., & Heer, J. (2011). D3: Data-Driven Documents. IEEE Transactions on Visualization and Computer Graphics, 17(12), 2301-2309.
- J. S. Park, J. C. O'Brien, C. J. Cai, M. R. Morris, P. Liang, M. S. Bernstein, Generative agents: Interactive simulacra of human behavior, in Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology (ACM, 2023).

CS 222: AI Agents and Simulations

Stanford University

Joon Sung Park