ACS Poster Collection (2018) 39–58 Published 12/2018; Paper 3

IACI – A Human-Inspired Computational Architecture to Help Us Understand Visual Data Exploration

Fernanda Monteiro Eliott FERNANDA.M.ELIOTT@VANDERBILT.EDU
Electrical Engineering and Computer Science, Vanderbilt University, Nashville, TN, USA

Keivan Stassun KEIVAN.STASSUN@VANDERBILT.EDU
Physics and Astronomy, Vanderbilt University, Nashville, TN, USA

Maithilee Kunda MKUNDA@VANDERBILT.EDU
Electrical Engineering and Computer Science, Vanderbilt University, Nashville, TN, USA

Abstract

Herein we present IACI, a human-inspired computational architecture to help us understand how we bridge a dataset to our goals and background knowledge, translating it into a visual encoding. IACI combines visual perception, memory, domain-specific knowledge and reinforcement learning techniques to learn and visually encode data. IACI uses a dual mechanism of internally and externally depicting plots in its learning process. Our motivation comes from astronomy, which increasingly generates large datasets, fostering discovery opportunities, *e.g.* new planets or rare *phenomena* observations. Insights from a human study of real astronomers conducting an open-ended visual exploration of a large dataset informed IACI's design and output format.

1. Introduction

Suppose you are moving to a new city and looking for a home. You look for city data to help you decide what neighborhood to focus on. You find a dataset to explore with a clear goal in mind. What are the variables driving your visual data exploration? What makes you choose to combine particular attributes instead of others? What do you look for on the scatterplots you create? Why is the *number* of property violations different from the *number* of bus stations? Our previous knowledge, experiences and common sense imbue meaning to a dataset attributes. While visualizing data, we may continually check if what we see matches what we expect to be seeing. If not, anomalies may trigger curiosity, or suspicion about the data, or even about our own assumptions. The Ignorance Project from Rosling (2013) is an interesting experiment that shows how people's assumptions may not be matched by data. Now, suppose an astronomer has been visually exploring a dataset and suddenly makes a discovery: she sees a pattern and conceives a new equation, an alternative approach to measure something important within the astronomy literature. Something very much like this happened with the discovery described in Bastien et al. (2013). Fabienne Bastien was exploring data from the Kepler space observatory using a data visualization tool called Filtergraph

from Burger et al. (2013), when she saw a curious visual correlation. The pattern she saw resulted in a novel method for measuring the surface gravity of distant stars.

This visual discovery inspired us to investigate what happens when an astronomer visually explores a dataset. That framed our research problem: **"how to model and develop a human-inspired computational architecture to help us understand visual data exploration"?** Our goal is to build a human-inspired computational architecture that gets a dataset as input and uses its domain and visual knowledge to learn from the data and to output a sequence of plots. For the sake of simplicity, we are approaching the domain of Astronomy. Our computational architecture is called *IACI*[1]. In astronomy data exploration, it is not rare to find performance-driven tools, or impressive methods, such as in Shallue and Vanderburg (2018), aiming to help astronomers to discover new planets, or stars from massive amounts of data, etc. Our approach is different but complementary; IACI is human-inspired and design-driven, which we expect in the long run will lead to performance-enhancing tools of a different kind. Our expectations are: 1. To understand and rudimentary mimic some of the processes underlying human visualization of datasets (such as depicting plots in our thoughts or actually plotting and visualizing them). 2. From IACI's output, to produce plots clear and useful enough to assist people in exploring their data and, being a bit ambitious, add to the scientific discovery. Herein we describe IACI in a high-level, seeing that we are still designing it. We are aware of five challenges:

1. To define IACI's design;
2. To delimit an adequate output format;
3. To circumscribe a testing bed;
4. To outline metrics to measure IACI's functioning and results;
5. To distinguish metrics to assess at what level IACI rudimentary mimics humans.

In this paper, we address the challenges 1-3 at different levels. In Appendices A and B [2] we included a story-parallel between an astronomer and IACI visually exploring a dataset (see Figure 1), and a table with the terminology we used in IACI's design. IACI is composed of an Analysis System (AS), a Visual System (VS), and a system specifying IACI's motor capabilities, the Actuator (OR) – see Figure 3. Once a dataset is inputted, the Analysis System is activated and phase 1 begins. At the end of phase 1, a domain-driven sequence of plots is outputted by the Actuator, triggering the Visual System. Phase 2 begins when the Visual System visualizes the outputted plots. At the end of phase 2, the Actuator outputs a visual-driven sequence of plots and a decision log of both phases. Figure 3 shows IACI and Figure 2 depicts phases 1 and 2, dashed lines highlighting the internally visualized plots mechanism.

1. According to a Tupi-Guarani tribe narrative, Iaci is the god of the moon (Cascudo 2000), and there are several stories related to its legend. We mention two versions. First, Tupã created Coaraci (god of the sun) and then, to avoid the darkness of the night, Iaci. Iaci and the sun fell in love, but they could only meet during an eclipse. Another version tells the origin of Naia, the Victoria Amazonica flower, a large white water lily. Iaci would visit the Earth and take girls into the sky to transform them into stars. Naia, a Tupi-Guarani girl, searched for Iaci everywhere. One day, exhausted and sick from the search, Naia saw Iaci's reflection on the water and ran towards it, only to drown. Iaci saw it and decided to transform Naia into a star in the water, instead of in the sky.
2. Appendices are at `https://vanderbilt.app.box.com/s/trr5ulpfcpv9kfwcb2qdpihrmw0ortz6`

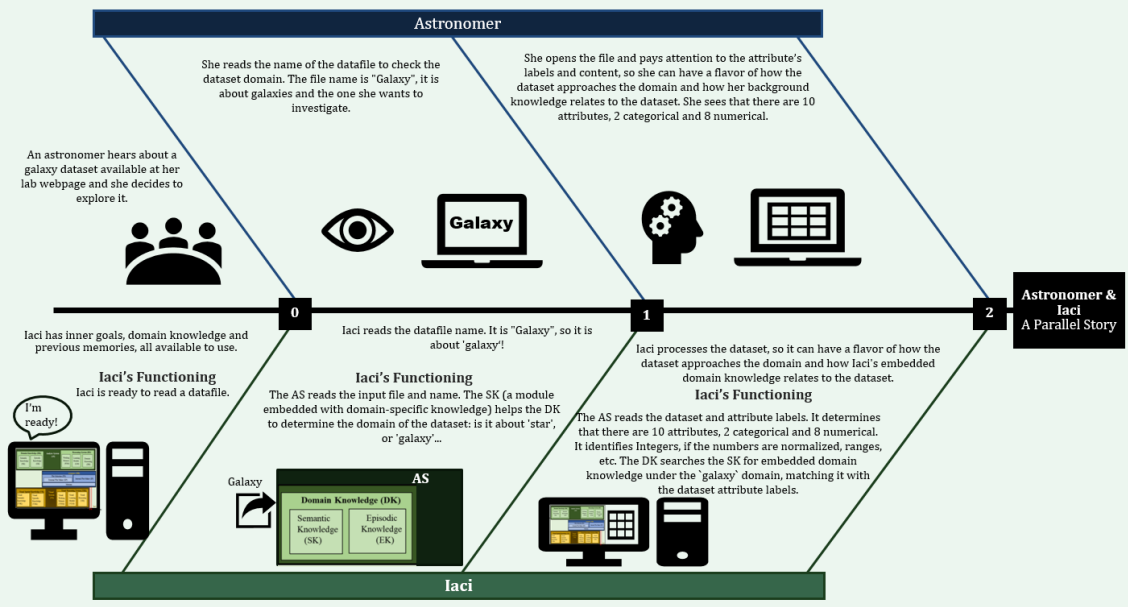


Figure 1. A figure depicting a story-parallel between an astronomer and IACI visually exploring a dataset.

2. Related Work and Background

Anscombe (1973) acknowledged the importance of visualizing data. In a similar direction, Matejka and Fitzmaurice (2017) developed a system that intakes a dataset and perturbs it towards a desired visual shape, still maintaining nearly the same statistical properties. If distinct datasets, whose statistical properties are very similar, can look so different on a scatterplot, we may wonder: is a scatterplot a reliable source for deducing the statistical properties of a dataset? Sher et al. (2017) conducted a study to investigate how reliable humans are to identify correlation indices through scatterplots, and their results suggest a lack of consistency in human judgments. Healey and Enns (2012) detail multiple attention and perceptual processing issues applicable to data visualization. Regardless our visual limitations, there are uncountable human discoveries from visually encoding data (consider our motivation: Bastien et al., 2013). Perhaps some discoveries only happened due to our limitations. Particularly for highly multivariate datasets, techniques may be required: 1. To reduce dimensionality and 2. To provide a more human-suitable visual encoding. According to Sedlmair et al. (2012), despite the importance of guidance through both techniques to data exploration, there is a lack of algorithms automatically addressing such guidance. The authors produced a "taxonomy of visual cluster separation factors". Working with synthetic and real datasets, they compared a human-made visual cluster identification with a non-visual, made by two different measures. Their results show a discrepancy between visual and non-visual cluster identification. Our expectation from IACI is that it should provide a cluster identification closer to humans than to the reported measures. By having cluster identification as goal and being fed with a dataset used in Sedlmair et al. (2012), we expect to compare IACI's cluster identification with theirs. Leman et al.

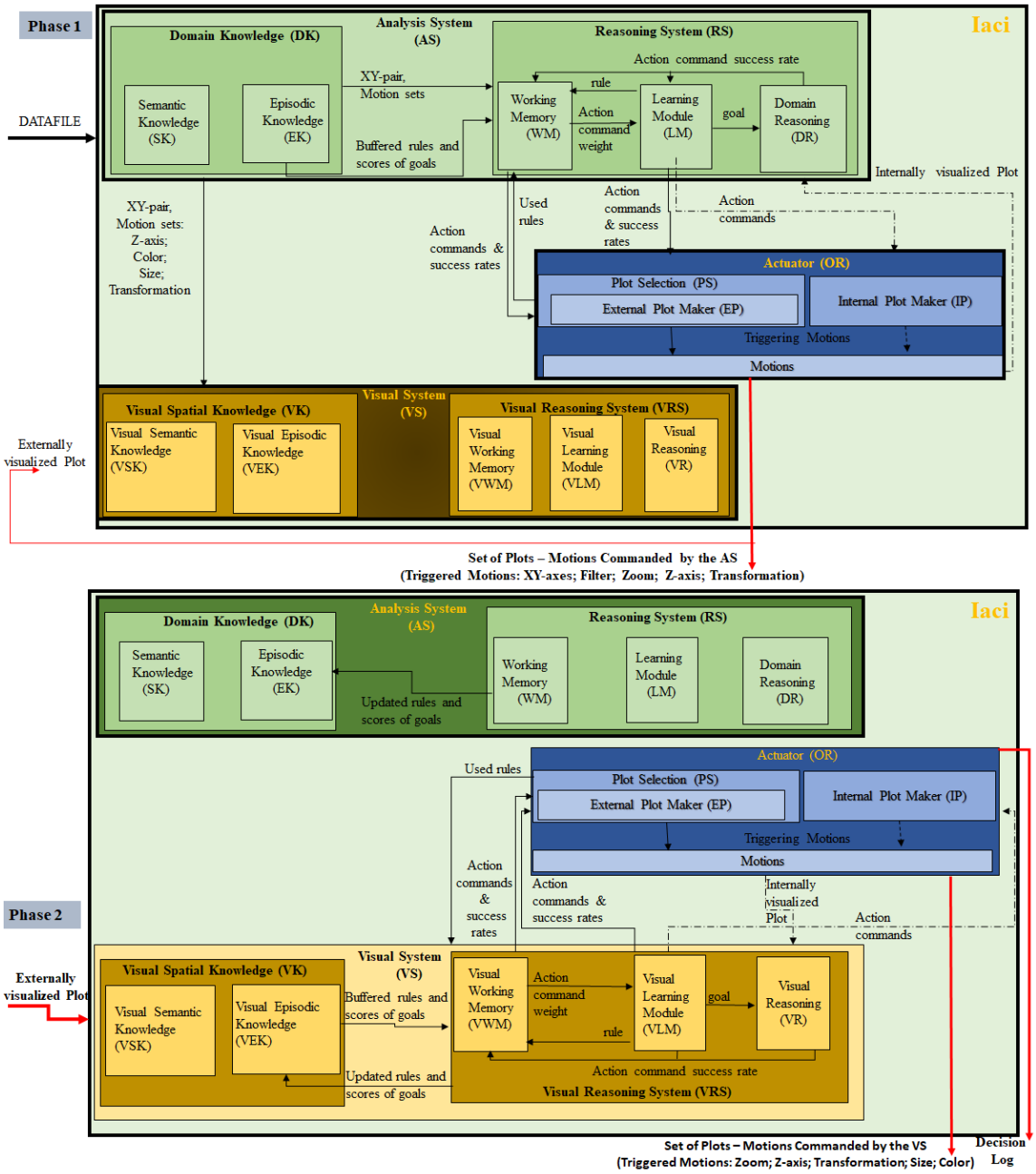


Figure 2. IACI and phases 1 and 2. Top – Phase 1, the Analysis System (AS) commands the Actuator (OR). Bottom – Phase 2, the Visual System (VS) commands the Actuator. IACI outputs plots in both phases, while a decision log is outputted at the end of phase 2.

(2013) describe a system for human interaction with data called "Visual to Parametric Interaction" (V2PI). V2PI addresses the issue of incorporating expert feedback into data visualizations with a system which seeks to break down barriers between mathematical and cognitive representations of data. V2PI is similar to our work for its focus on human cognitive responses to data visualizations and the importance of the judgment of trained experts in producing meaningful plots.

2.1 Background

We conducted a human study with experienced astronomers (graduate students). Participants were asked to make scatterplots while exploring an unfamiliar galaxy dataset from Berlind et al. (2006) using the Filtergraph visualization tool. Our preliminary results are described in Eliott et al. (2017): data collecting from screen recordings of each exploration session is anchored at the XY-pair of attributes, leading to the definition of *major observation (M)* and *minor observation (N)*. Whereas we defined M and N to collect our results, we got intrigued by how powerful those definitions are: 1. To standardize our results and reveal interesting patterns, which would be hidden otherwise, such as the circular walk across M (repetition of an XY-pair after a gap) and within M (plot repetition withing M). 2. To find a narrative-like skeleton under their exploration: seeing XY-pairs as a sort of character, and attributes (at color, size or Z-axis) or transformations brought to the scene (a scatterplot) as a change, a new perspective of the character. We observed a story-like sequence of plots develop as the participants chose to 'flip through' the dataset, setting up N at a brisk pace. Interestingly, Nersessian (1992) brings up that narrative-like reflections play an important role in scientists' thought experiments. We saw in M and N conceptual framework a promising human-inspiration skeleton to guide IACI's decision-making. See Figure 7 for a sequence of N's produced in our human study.

Definition. XY-pair of attributes are the attributes assigned to the XY-axes of a scatterplot, *e.g.*: getting a galaxy dataset and setting the Velocity attribute at the Y-axis and the Brightness attribute at the X-axis.

Definition. M is a grouping of contiguously viewed scatterplots (chronological sequence of N's) within which the attributes assigned to X and Y axes remain constant. When the attribute assigned to either the X or Y axis (or both) is changed, a new major observation begins.

Definition. N – each individual scatterplot generated within M is called N. When a new M begins, its first N is created: a scatterplot holding only the XY-pair of attributes, nothing else. Then, each time an action command is applied at the scatterplot, a new, subsequent N is created. We adjusted here the definition of N to accommodate IACI's design.

Definition. Motions get action commands and execute them to make a scatterplot. There are 7 motions: XY-axes, zoom, filter, Z-axis, color, size, transformation. Transformation is a multi-faceted motion in the sense that it teams up with another motion when triggered. *XY-axes* is the motion of setting up the XY-axes of a scatterplot, while *XY-pair* is an action command delivered to the XY-axes motion. In our human study, the Filtergraph visualization tool gets human action commands and then handles and coordinates motions to make scatterplots. However, IACI plays a dual role: as a user *and* as a visualization tool: the Actuator, IACI's motor system, is responsible of coordinating motions and making scatterplots.

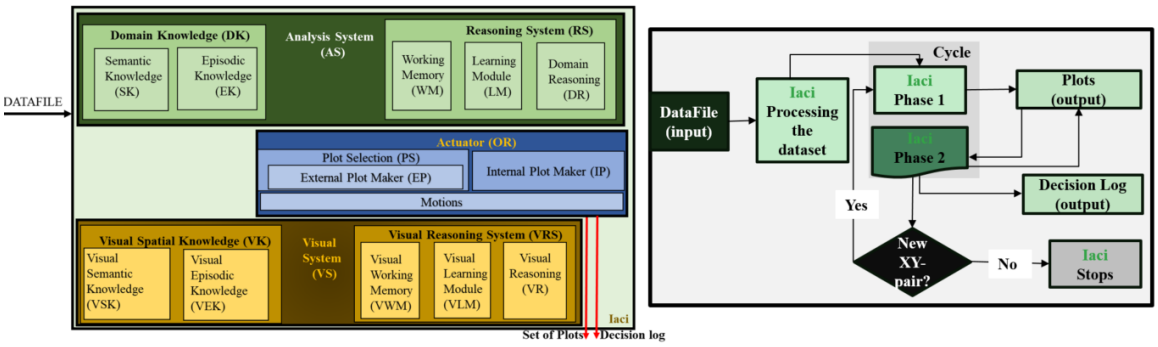


Figure 3. **Left:** IACI Computational Architecture. **Right:** IACI decision process and definition of simulation.

Definition. Action command is a specific instruction to a motion. Therefore, action commands are restrained by the scope of motions, which is: 1. defining the *XY-pair* (two specific attributes, or combination of attributes) to set the scatterplot XY-axes. 2. To use *color* on the point-glyph: IACI may use one attribute, or combinations of attributes. Once a point-glyph has color, IACI may change the palette of colors, such as set a 5-colors pallet instead of 2-color to fit best with the data. 3. To use *size* on the point-glyph: is similar to color but, instead of choosing a palette, IACI chooses a shape. 4. *zoom: e.g.*, zoom at the top 25% X-axis values; 5. *filter* the data: to simplify, IACI's filtering uses only categorical dataset attributes and respective categories. 6. *Z-axis:* use a specific attribute, or combination of attributes, to set the Z-axis. 7. To use a function to *transform* the X, Y, or Z axes, or color, or even the size values. Suppose we have the average velocity of different objects at the Z-axis. If we have the space and time attributes, we may transform the Z-axis by using $v = s/t$.

A limitation in IACI's current design is the lack of emotions, feelings and homeostatic goals. As Damásio (1994) and Damásio (2004) emphasize, emotions help us to establish priorities and our decision-making. As suggested in the astronomer-IACI parallel story[2], people usually frame real-time priorities and decision-making during their data exploration path. Of course, some decisions may be at random, or we may decide to follow a random data exploration strategy – in fact, one of our human study participants told us to be doing so. Still, emotions are likely to be playing their role in the decision of establishing no priorities and following a random strategy. We ask ourselves: *is it the case that the role of emotions over our participants' decision-making, and our own emotions while interpreting their output, spotlighted a story-like impression from M's and N's?*

3. IACI, a Human-Inspired Computational Architecture

IACI gets a dataset as input and uses reinforcement learning techniques (Sutton & Barto (1998)) to learn action commands and achieve goals – the learning state space is depicted in Figure 4. IACI's task is to read a datafile, define goals related to the inputted dataset, and learn action commands to output plots meeting a goal. Three main systems compose IACI, Figure 3: the Analysis System, the Visual System and the Actuator. Even though the Analysis System is domain-driven and the

Visual System is visual-driven, their structure is very much alike. For that, and because we are still designing the Visual System, we detail the functioning of the Analysis System only. However, as the contents of the systems are different, we provide clarifications. IACI is to learn from every dataset it "sees". A dataset leaves content in IACI's long-term memory in the format of **rules** in the Analysis System and of **image-like** rules in the Visual System. Rules are produced and updated through learning and, when IACI decides to stop the simulation, rules are embedded in IACI's long-term memory. IACI refines its action commands from learning within a simulation and across simulations. In Table 1 we present the terminology used to describe IACI.

IACI works in cycles of two phases (Figure 2): in phase 1, the Analysis System commands the Actuator; in phase 2, the Visual System commands the Actuator. The Analysis System produces and learns a_k, domain-driven action commands. The Visual System produces and learns a_i, visual-driven action commands. **Definition. Simulation** outlines at least one complete cycle. IACI reads a datafile, learns, outputs scatterplots (resuming phases 1 and 2) and stops. We depict a simulation and IACI's decision process in Figure 3.

IACI outputs scatterplots; therefore, the Actuator coordinates 7 built-in motions and 2 plot makers: the Internal Plot Maker (IP) and the External Plot Maker (EP). The External Plot Maker is part of the Plot Selection (PS), a bigger Module that gets action commands and success rates and selects action commands to be sent to the External Plot Maker. The External Plot Maker gets action commands and makes plots for external depiction, an externally visualized plot v_{out} (outputted plot). The Internal Plot Maker gets action commands and makes plots for internal depiction, internally visualized plot v_{int} (not-outputted). Action commands are motion instructions such as "zoom the top 25% X-axis values", or "use a specific attribute at the Z-axis and apply a log transformation at the Z-axis". If the Actuator receives action commands *and* a success rate, it delivers both to the Plot Selection; otherwise, the Actuator sends the action commands to the Internal Plot Maker.

Goals. IACI has two kinds of goals: domain-driven goals g_k (in the Analysis System) and visual-driven goals g_i (in the Visual System). IACI focus on goals while producing an action command, which, on its turn, generates v_{int}. Then, IACI matches v_{int} with the goal to calculate the action command success rate $v_{int:S}$. We may have goals beforehand. Thus, we select action commands towards a scatterplot accomplishing our goals; or goals may follow from visualizing the data. Actually, both may be happening at the same time: *goals→ action commands→ plots*, or *plots→ goals→action commands*. The Analysis System mimics mostly the 1rst approach, while the Visual System the 2nd. To keep goal consistency within a phase, goals are steady. IACI avoids changing goals at the same phase, though the Analysis System and the Visual System can replace a goal – observation of plots may drive us to think of new goals or, if experimenting on our data and realizing our goal does not fit well, we may adjust the goal to the data.

Since the Analysis System and the Visual System autonomously define goals, g_k and g_i are likely to be distinct in a cycle. However, it may happen for goals present in both systems: suppose on cycle 5, $g_k = g_i =$*monotonic plots*. The Visual System uses visual variables and a visual memory gallery of monotonic plots to attain g_i. On the other hand, the Analysis System may use

the monotonic scagnostics measure (one out of 9) from Wilkinson et al. (2005) and Wilkinson et al. (2006), to get a score of υ_{int} for monotonic, defining the $\upsilon_{int:S}$.[3]

3.1 IACI and Major and Minor Observations Framework

We use the M and N framework in IACI's design. This is a choice of design, but it makes sense to explore an XY-pair at a brisk space and then change to another XY-pair (as we saw in our study; see Figure 7), instead of choosing every scatterplot parameters at the same time (without fixing the XY-pair). Each complete cycle creates M and the sequence of N is given by the outputs from phases 1 and 2. When phase 2 is complete, IACI may decide to look at a new XY-pair and restart phase 1, or to stop (see Figure 3). IACI reads a dataset, autonomously decides what XY-pairs to look at (with regard to defining the number of M's and of cycles) and stops when all XY-pairs have been visualized. At the beginning of phase 1, the Analysis System triggers the XY-axes motion by delivering an XY-pair action command to the Actuator. The XY-axes motion is kept running the same XY-pair until the end of phase 2. In every plot (υ_{int} and υ_{out}), the first triggered motion is the XY-axes to execute the action command of setting the XY-pair. IACI never repeats an XY-pair in the same simulation (no circular walk).

IACI reads and processes a dataset. Then, **phase 1** begins: the Analysis System uses its inherited background knowledge and memory from past datasets to calculate the current XY-pair of attributes and a list of attributes *per* motion which makes sense to combine and informs the Visual System. The Analysis System learns to produce domain-driven action commands by following g_k. At the end of phase 1, the Actuator follows action commands from the Analysis System and outputs a set of υ_{out}. The Visual System externally visualizes the set υ_{out}, marking the beginning of **phase 2:** to learn and refine action commands seeking g_i, the Visual System uses a) υ_{out} from phase 1; and b) a list of attributes per motion provided by the Analysis System. The Visual System uses its visual memory and internal variables to define g_i and selects action commands accordingly. At the end of phase 2, the Actuator follows action commands from the Visual System and outputs a set of υ_{out} and a decision log of phases 1 and 2.

In contrast to phase 2, all point-glyph from phase 1 (in both, υ_{int} and υ_{out}) have no color or size. As Figure 4 (right) shows, the Visual System and the Analysis System trigger different sets of motions. In phase 1, the Analysis System chooses the XY-pair and works through transformation, Z-axis, zoom and filter. In phase 2, the Visual System triggers color, size, transformation, Z-axis and zoom. The Visual System and the Analysis System may have distinct goals but both are synchronized regarding the current XY-pair and the set of attributes to use motions with. The Actuator gets action commands from both, the Analysis System and the Visual System, and makes scatterplots by coordinating motions to execute action commands. In case of motions triggered by both systems, the flavor of action commands should be very different, as a result of the systems distinct goals and contents. Of course, splitting the motions reached by each system is a choice of design. We made the two groups considering what motion can favour each system more.

3. We may use the scagnostics measures to define some of g_k and g_i. That case, the $\upsilon_{int:S}$ for g_k comes from the R cran package from Wilkinson & Anand (2015).

3.2 Phase 1 and the Analysis System (AS)

The **Analysis System (AS)** is composed of two systems: the Domain-Knowledge (DK) and the Reasoning System (RS). The former is related to IACI's long-term memory and the latter to its working memory and learning. The Domain-Knowledge is responsible of providing g_k and the Reasoning System of providing a_k and $v_{int:S}$. **IACI may be applied to fields other that astronomy. To accomplish this, the Domain-Knowledge is to be replaced by a field-related knowledge.** While looking at a dataset and knowing what the attributes mean, an analyst has expectations about the dataset objects and relationships among them. Expectations contribute to visual data exploration; regardless of whether they are met, it is interesting to investigate how people visually encode responses to their thoughts (our human study contributes on that). In case we are not familiar with a dataset, we may generalize from previous experiences while getting to know the dataset. The Domain-Knowledge is intended to mimic, in a simplistic sense, using domain-knowledge and previous experiences to investigate data while defining goals and combinations of attributes to plot.

Once IACI reads a datafile, the Analysis System works to put the dataset into context – this step is done only once by simulation. The Domain-Knowledge identifies categorical and numerical attributes. The former are inspected to enumerate data filtering options. The latter are categorized: whether they belong to the $\mathbb{Z}$ set, if they are normalized, *etc*. IACI gets the dataset name and labels of attributes and matches them with its inherited astronomy knowledge to pinpoint the dataset domain d^D. The Domain-Knowledge stores specific domain knowledge linked to key labels of attributes – *e.g.* for d^D =galaxy, key labels such as Vel for velocity, and so on. Of course, one requirement for IACI to work properly is an accurate identification of the dataset name and attribute labels. For instance, velocity related data must have *vel* as part of its attribute label. The same way, a dataset named 'galaxy' should refer to a galaxy domain. Even for humans, a self-speakable label facilitates data manipulation. The Domain-Knowledge is embedded with ways to deal with homonyms. **IACI handles dataset produced from discrete observations on multiple objects differently from continuous views about the same object.**

3.2.1 Domain-Knowledge (DK)

The Domain-Knowledge is equipped with two embedded modules: the Semantic Knowledge (SK) and the Episodic Knowledge (EK). The Semantic Knowledge is supposed to rudimentary mimic the background and foundation knowledge experts learn from studying astronomy. The Semantic Knowledge imbues IACI with rule-based knowledge on astronomy objects, relations, equations and principles, and keeps a matching between knowledge and key labels of attributes. While the Semantic Knowledge is hard-wired and unchangeable, the Episodic Knowledge keeps experience-based knowledge, supposed to mimic the experience experts get from data exploration. The Episodic Knowledge keeps rule-based knowledge dynamically updated within and across simulations. Once d^D is determined and the dataset processed, the Domain-Knowledge starts searching for background knowledge related to the dataset. It searches the Semantic Knowledge and the Episodic Knowledge to determine a set of XY-pairs of attributes (one attribute by axis, or attributes combined at the same axis) from the dataset. Then, for each XY-pair, the Domain-Knowledge:

- Determines a set of attributes (from the dataset) for each motion: for filter, transformation, Z-axis, color and size. The set of attributes for transformation are tied to functions and motions to team-up with – *e.g.* use the average velocity equation at the Z-axis. Sets of motion attributes and the XY-pair are fed into the Visual System;
- Establishes a set of suitable zooming options;
- Items above define a_k^M: the number of available action commands for the current XY-pair;
- Determines a set of g_k.

Note that the set of g_k is based on the astronomy literature. That means the effect of a goal in IACI's data exploration varies according to the dataset (*i.e.* distinct datasets may approach differently the same domain). Sets of g_k are embedded in the Semantic Knowledge, while their scores in the Episodic Knowledge. A score may start from zero or slightly above zero (if the goal is usually followed among the astronomy literature for the XY-pair). Through learning, IACI tries at least one g_k by XY-pair. The Reasoning System updates the score of g_k when used by the Learning Module, and sends it back to the Episodic Knowledge at the end of the simulation. If g_k consistently provides low scores for an XY-pair during learning, IACI updates the score of g_k and changes the goal. *From a human perspective, it is like checking a hypothesis and moving on.* Since the rank provides ways for IACI to explore or to exploit goals, the Learning Module uses the scores and internal learning variables to dynamically rank goals.

Semantic knowledge (SK)

The Semantic Knowledge incorporates five categories linked to key labels of attributes:

1. *General Overview* on astronomy objects and relations among them. Objects such as stars, planets and galaxies;
2. *Event Oriented Overview*: focus on specific objects and draws some of the most common events associated with them. For example, what may happen to a star? It may explode, incorporate another star, etc;
3. *Astronomy equations* related to the topics above;
4. *Representative Examples*: keeps representative study case examples of events related to the objects. For example, what are the values that, without a doubt, one can tell that a star has died?
5. *Unusual Examples*: keeps representative study case examples that seem to contradict the theory. We believe those examples are crucial to trigger interesting plots.

Episodic knowledge (EK) The Episodic Knowledge stores experience-based rules from previous simulations. After processing a dataset, the Episodic Knowledge buffers into the Reasoning System scores of goals and every rule related to d^D – mimicking a process of triggering memories. Rules not related to d^D are kept unaltered inside the Episodic Knowledge (not triggered memories). At the beginning of the very 1rst IACI's simulation, the Episodic Knowledge has got only scores of goals, none experience yet. When the learning starts, the Reasoning System creates new rules and updates buffered rules. At the end of a simulation, every rule and goal scores are sent back to the Episodic Knowledge.

3.2.2 Reasoning System (RS)

The Reasoning System is composed of: Working Memory (WM), Learning Module (LM) and Domain Reasoning (DR). The Reasoning System is responsible of IACI's domain-related learning (Learning Module) of action commands, and of IACI's reasoning (Domain Reasoning) to evaluate action commands. The Working Memory keeps a working memory of the learning process and updates past memories (buffered rules and score of goals). Both, the Learning Module and the Working Memory send action commands to the Actuator to produce υ_{out}. However, the Plot Selection outputs them in a different fashion (see Figure 6), using as assumption that our response to flash of memories is faster than to the learning's (while we are exploring and building new memories).

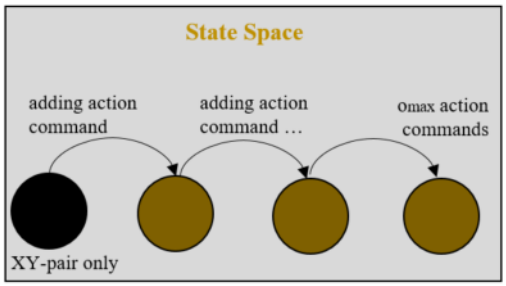
IACI's learning refines action commands aiming to produce υ_{int} with high success rates for a goal. When IACI stops the learning process, it has found a high success sequence of action commands o_a^M (or as high as it can be, since a goal may not fit well with the data, but still be usable). Then, o_a^M and respective $\upsilon_{int:S}$ are sent to the Actuator. The Actuator delivers both to the Plot Selection which, on its turn, sends o_a^M to the External Plot Maker. The External Plot Maker coordinates motions to execute action commands and outputs one υ_{out} by action command (translating internally visualized plots into externally visualized plots).

Domain Reasoning (DR) The Domain Reasoning has internal mechanisms to match goals and plots to provide a success rate of action commands. The Domain Reasoning gets υ_{int} from the Actuator and g_k from the LM to provide $\upsilon_{int:S}$. If υ_{int} has zero point-glyph, it is considered an invalid plot: $\upsilon_{int:S} < 0$.

Learning Module (LM). We are considering two learning approaches: 1. Learn from motions or 2. Learn from action commands. An issue with *1* is that shortening the decision space to one motion still leaves the decision of what action command to apply; a subsystem would have to pick an action command from the selected motion. For simplicity, we are adopting approach **2**. We may be using the model-based reinforcement learning algorithm R-max from Brafman & Tennenholtz (2002) in a fixed-sum game, modeling it as a Markov Decision Process, in which the Analysis System plays against a stationary adversary with only a single action at each state and rewards ranging between zero and R_{max} (reward values built from $\upsilon_{int:S}$).

If we merge phases 1 and 2, allowing the Analysis System and the Visual System to work at the same time, we may consider a variation. We can model it as a stochastic game in self-play – one agent representing the Visual System and the other representing the Analysis System, both agents competing to decide what action command goes to υ_{in}. At each state, the winner is who chooses the action command with the highest $\upsilon_{int:S}$. This case, υ_{out} results from the Visual System's and Analysis System's conjoint work. A homeostatic system can play a role in here, impacting priorities for a visual (Visual System) or a domain-oriented (Analysis System) decision. Another reinforcement learning algorithm we think may be interesting is the WoLF-phc from Bowling & Veloso (2001).

Think about a scatterplot. What is the maximum number of action commands you still consider to be possible to follow? To mimic a human limitation in IACI's design, we defined α, the maximum allowed number of action commands *per* υ_{int}. Duo our human limitations, it becomes hard to understand a plot after adding many action commands at the same scatterplot. In fact, in our human study, the session with higher average of action commands by scatterplot reached a value



Motion	VS	AS
XY-axes		V
Filter		V
Zoom	V	V
Z-axis	V	V
Transformation	V	V
Size	V	
Color	V	

Figure 4. **Left:** State space defined by the maximum number of action commands for the current XY-pair. **Right**: set of motions triggered by the Analysis System (AS) and by the Visual System (VS).

below 4. Suppose we transform the X-axis values equalizing all the point-glyph values. From a visual perspective, there is only one point-glyph. **But what is the maximum number of action commands a single scatterplot can hold?** It depends upon motion definition and the dataset, but it may be infinite. If restraining the filter to categorical attributes only, the number of filtering action commands is finite and given by the universe of the dataset categories. Once set an XY-pair, we may add a Z-axis, set color and size (so far, 4 action commands). We can filter or zoom the data until there is only one instance left. Suppose $\{\text{X-axis value} = \text{Y-axis value} = 1\}$: it can continuously be transformed if multiplied by $\mathbb{N}$ numbers, one multiplication (and plot) at a time – of course, the story is different for a zero value instead of 1.

The Learning Module is activated every time a new XY-pair is delivered to the Reasoning System: it is time for the Learning Module help IACI to produce N. The Learning Module uses the motion sets and a_k^M, both sent by the Domain-Knowledge to choose action commands. The Learning Module uses the scores of goals and its internal learning variables to dynamically rank goals and choose the current g_k. The ranking takes into account: how easy it is to obtain a high success rate for that goal; in how many simulations the goal has been followed. A dynamic ranking enhances the diversity of υ_{int}, even if inputting multiple times the same dataset. The Analysis System has an embedded mechanism to handle dataset repetition and increase the diversity through augmenting all exploratory parameters. **In the future, we aim replacing the ranking process by a Homeostatic module mimicking emotions and feelings to prioritize goals and systems**. We may be using Eliott & Ribeiro (2015) as inspiration.

When the Learning Module is activated, at each time step one action command is selected to be added on υ_{int}. The number of action commands on υ_{int} increases according to: $o^M = \{1; 2; 3; ...; o_{max}\}$. Figure 5 depicts how υ_{int} is made through learning. For comparison with a human output, Figure 7 shows a sequence of plots within M produced by an astronomer during our pilot study. IACI uses the Analysis System to define o_{max} according to the current XY-pair, but $o_{max} <= \alpha$. **We expect $\alpha <= 20$: in our human study, the higher session average of action commands *per* scatterplot is below 4 (including visual action commands).**

The first υ_{int} of every M is a scatterplot with only the XY-pair. The 2nd υ_{int} is made by adding another action command. The 3rd υ_{int} has $o^M = 3$ (the current and the previous, inherited action commands), and so on. IACI keeps learning from adding action commands until $o^M = o_{max}$. The

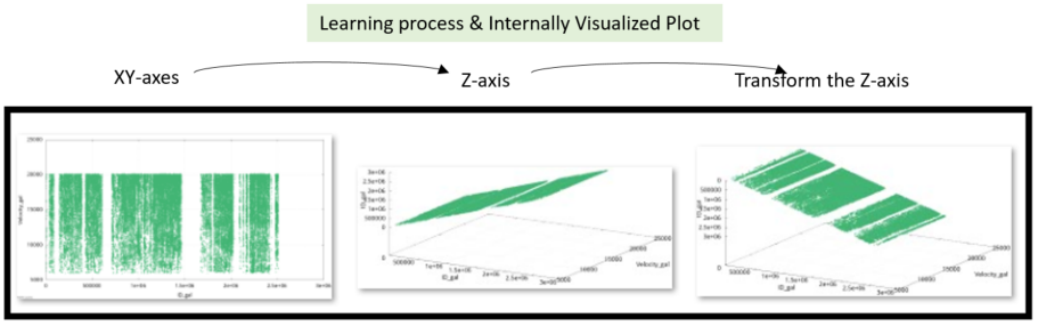


Figure 5. Learning process: action commands from the Learning Module (LM) chronologically generating an υ_{int}, one plot *per* action command.

learning state space is defined by o^M – Figure 4 depicts the state space and transitions while Figure 5 shows the generation of υ_{int}. After selecting a new action command and transitioning to a new state, the Analysis System sends the action commands to the Actuator. The Actuator makes υ_{int} and sends it to the Domain Reasoning. The Domain Reasoning uses g_k to calculate $\upsilon_{int:S}$ and sends it to to the Learning Module. There is a possibility of using different goals for different states (humans may change goals within M). The Learning Module calculates the reward value using: 1.$\upsilon_{int:S}$; 2.the current state and o_{max}; 3. a weight ϱ_{gk}^M from the Working Memory (see the Working Memory for details). With the reward value, IACI learns how good the current a_K is given g_k and the current combinations on υ_{int}. Suppose $o_{max} = 3$ and $a_k^M = 5$. In the first state, a_K =XY-pair; in the 2nd state, there are 4 possible a_K; in the 3rd state, there are 3 possible a_K (we don't allow a_k repetition): 12 possible combinations of action commands for 3-state space with 5 possible action commands. The ordering of action commands is important because different ordering may lead to distinct υ_{int}. Of course, $o_{max} <= a_k^M$.

Definition: Learning loop. Once υ_{int} has o_{max} action commands, IACI restarts the learning loop. IACI goes back to $o^M = 1$, υ_{int} with only the XY-pair. Note that from $o^M > 1$, the $\upsilon_{int:S}$ results from a chronological combination of action commands. IACI restarts the learning loop until it has explored a_k^M action commands and converged to a sequence $o_a^M = \{a_{k:1}; ...; a_{k:max}\}$ of action commands (or if a maximum number of learning loops is reached). In case a_K produces a plot with no point-glyph (invalid action command for the current data), the Domain Reasoning delivers $\upsilon_{int:S} < 0$, the learning variables are updated, and IACI restarts the learning loop. If the average value of $\upsilon_{int:S}$ for o_a^M is very low, IACI chooses another goal and restarts the learning loop. At the end of the learning loop, the Learning Module uses o_{max}, and the average value of $\upsilon_{int:S}$ (for o_a^M) to update the score of g_k. The Learning Module sends o_a^M and success rate to the Actuator, the Actuator to the Plot Selection, which, finally, delivers every action command from the Learning Module to the External Plot Maker. The External Plot Maker outputs a sequence of o_{max} υ_{out}, from $a_{k:1}$ to $a_{k:max}$: each υ_{out}s inheriting previous action commands.

Working Memory (WM)

The Episodic Knowledge buffers a set of rules ϱ and scores of goals into the Working Memory at the beginning of a simulation. Then, the Working Memory keeps track of the learning loops

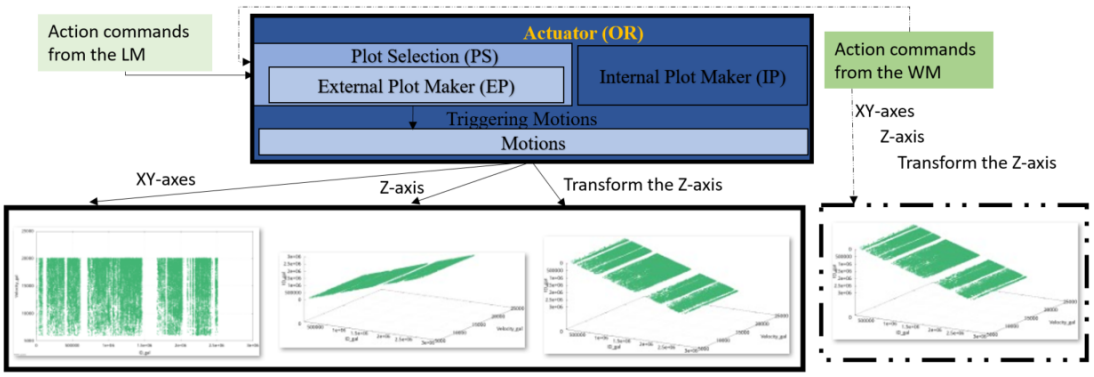


Figure 6. The Plot Selection (PS) uses action commands from the Learning Module (LM) and from the Working Memory (WM) in a different fashion. Example of 3 action commands: $o_a^M = \varrho_a^M[1] = \{$X-axis=IDgal, Y-axis=Velocitygal; Z-axis=IDgal; Transform the Z-axis, Reverse Function$\}$. The Plot Selection commands the External Plot Maker (EP) to produce one υ_{out} *per* action command from o_a^M; starting from the 2nd, each υ_{out} inherits previous action commands. **Dashed Lines:** The Plot Selection commands the External Plot Maker to produce one υ_{out} (with all action commands) from $\varrho_a^M[1]$.

to create new rules and update ϱ, registering IACI's learning and data exploration. At the end of the simulation, all rules are sent back to the Episodic Knowledge. We are using the Alec architecture's cognitive system from Gadanho (2003) as inspiration do design IACI's rule-system – Alec's cognitive system is influenced by Clarion from Sun & Peterson (1998).

The Working Memory helps the Learning Module to use past memories (from the same d^D, $XY - pair$ and g_k, **but not necessarily** the same inputted dataset) to diversify action command choice – also a way for IACI to generalize from previous to similar experiences. During the learning loop, if the Working Memory already has a rule (from simulations other than the current) matching the current set of action commands and goals, it informs the Learning Module by sending $\varrho_{gk}^M \approx 0.98$, so the Learning Module may converge to another set of action commands but with a similar $\upsilon_{int:S}$. In case there isn't such a rule, $\varrho_{gk}^M = 1$.

Creating Rules. The Working Memory keeps track of the learning loops: if a_k generated a high $\upsilon_{int:S}$, the Working Memory records and transforms that into a rule ϱ^M. Each ϱ keeps: 1. rule ID; 2. d^D; 3. XY-pair; 4. g_k; 5. all action commands currently at $\upsilon_{int:S}$; 6.number of action commands at υ_{int}; 7. number of times the rule has been used by the Plot Selection to output υ_{out}, starting from zero; 8. how old the rule is (simulation ID); 9. success rate: a measure taking into account 6., 7. and 8. Rules with more action commands tend to have higher success rates. **Deleting Rules.** At the end of a simulation, the Working Memory searches for similar rules and deletes the ones with smaller number of action commands.

After the Learning Module has delivered o_a^M to the Actuator, the Working Memory searches through its rules and selects some of them to be sent to the Actuator. The Working Memory makes a vector ϱ_a^M of rule-based action commands, with each rule separately kept in a vector's position. Each position holds a sequence of rule-based action commands and success rate. The Plot Selection uses the success rate to decide which positions of ϱ_a^M to follow and sends that to the External Plot

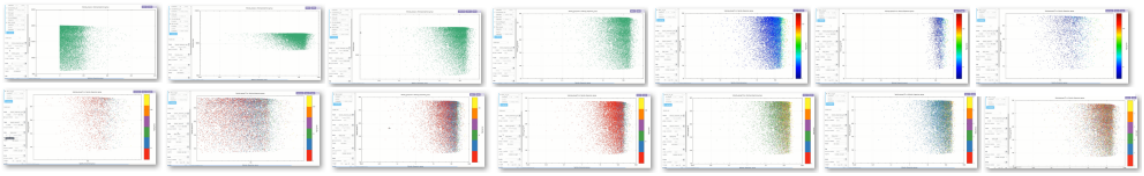


Figure 7. A sequence of plots from our pilot study. In chronological order, from left to right: plots showing 14 N's within one M.

Maker. The Plot Selection informs the ϱ_a^M chosen positions to the Working Memory, so it can update the rule's statistics of use. *Note that ϱ_a^M holds sequences of action commands from distinct simulations, and even datasets. Suppose IACI reads two different datasets sharing the same d^D and key attributes. If IACI produced the same XY-pair and g_k while reading each dataset, it may have produced rules from learning through both datasets.*

3.3 Phase 2 and the Visual System (VS)

We are still in the progress of designing the Visual System. Therefore, we describe it briefly. At the beginning of a simulation, the Analysis System feeds the Visual System with the current sets of motions. The Visual System is free to choose action commands, but not to choose motion defining attributes. When IACI starts outputting υ_{out} from the Analysis System, the Visual System is activated: it externally visualizes the plots. Phase 2 begins, IACI shifts from a domain-oriented to a visual oriented approach. Analog imagery may help IACI to match goals and plots: we are studying how to embed IACI's Visual System with a visual analogical reasoning (Kunda et al. (2013) and McGreggor et al. (2014)). The Visual System combines the externally visualized plots with the motion sets sent by the Analysis System to learn action commands meeting g_i. The Visual System uses each υ_{out} as starting point: it internalizes the externally visualized plots and provides a visually modified version of it. The Visual System adds its own action commands a_i by learning to fulfill visual goals. This is how plots get color or size: though the action of the Visual System. The Visual System is composed of the Visual Knowledge (VK) and the Visual Reasoning System (VRS).

3.3.1 Visual Knowledge (VK)

The Visual Knowledge is composed of two modules: the Visual Semantic Knowledge (VSK) and the Visual Episodic Knowledge (VEK). To explore data, it may be valuable to search for mathematical patterns, for example, to look at a scatterplot and, by using a visual memory, to compare its appearance with familiar distributions of data points. We intend to mimic a visual search for mathematical patterns in the Visual Semantic Knowledge. The Visual Semantic Knowledge also holds a visual plot gallery matching visual goals and visual assumptions, such as examples of 'monotonic plots'. The Visual Semantic Knowledge is equipped with knowledge about palettes of colors and shapes to use with size. Described in Eilbert et al. (2018), we conducted a 2nd human study to get insights on how to design the Visual Knowledge. We collected scatterplots from astronomy. Then, for each scatterplot, we compared scagnostics values given by people with the ones provided

by the scagnostics CRAN package from Wilkinson & Anand (2015). We expect to collect scatterplots from the astronomy literature (followed by astronomy interpretations) to feed IACI's Visual Semantic Knowledge. The same way as the Analysis System keeps experience-based knowledge in the format of rules, the Visual Episodic Knowledge keeps a experience-based *visual* memory of υ_{in}, with rules keeping images of plots.

3.3.2 Visual Reasoning System (VRS)

The Visual Reasoning System is composed of the Visual Working Memory (VWM), the Visual Learning Module (VLM) and the Visual Reasoning (VR). During learning, the Visual Working Memory may decide to keep some of υ_{int} into its visual gallery by making new rules. We intend to employ pixel-oriented techniques, such as in Keim (2000), in the Visual Reasoning to provide $\upsilon_{int:S}$.

3.4 Actuator (OR)

A plot may be internally visualized by the Analysis System and by the Visual System or outputted, externally visualized by the Visual System. The Actuator is composed of Internal and External Plot Makers, which get action commands and coordinate motions to make plots. The Plot Selection outputs a decision log and uses $\upsilon_{int:S}$ to select a set of action commands to be sent to the External Plot Maker, generating υ_{out}. Note that most of the υ_{int} are discarded during the learning loop and never get to be outputted. Action commands lacking a success rate are sent to the Internal Plot Maker; and sent to the Plot Selection otherwise. The Plot Selection sends to the External Plot Maker every action command delivered by the Learning Module. On the other hand, if it came from the Working Memory, it decides what to output and does it in a different fashion, see Figure 6. Every υ_{out} from the same cycle is considered an N belonging to the same M. At the end of the simulation, the Actuator uses the success rates to output a decision log for every M produced within the simulation.

4. Assessment of IACI Computational Architecture

A possible approach to assess IACI is by comparing the astronomers' plots and decisions from our study with IACI's plots and decision log. A simpler approach to test incomplete pieces of IACI is to define IACI's Domain-Knowledge using the Hertzsprung-Russell (HR) Diagram; embedding IACI's Semantic Knowledge with the HR's established parameters on stars, such as spectral class, surface temperature, luminosity, absolute magnitude, etc. IACI's goals may be to test relationships in a star dataset: *e.g.* groupings into different star classifications, or evolution routes of stars between groups.

5. Final Remarks

We described IACI, a human-inspired computational architecture to help us understand how we visually encode datasets. We are aware that IACI should be able to capture a narrative-like mechanism while choosing what to plot (to somehow mimic the role that narrative-like reflections play in

Term	Name
d^D	Dataset domain
AS	Analysis System
DK	Domain-Knowledge
SK	Semantic Knowledge
EK	Episodic Knowledge
RS	Reasoning System
WM	Working Memory
LM	Learning Module
DR	Domain Reasoning
VS	Visual System
VK	Visual Knowledge
VSK	Visual Semantic Knowledge
VEK	Visual Episodic Knowledge
VRS	Visual Reasoning System
VWM	Visual Working Memory
VLM	Visual Learning Module
VR	Visual Reasoning
OR	Actuator
PS	Plot Selection
IP	Internal Plot Maker
EP	External Plot Maker
Motions	XY-axes, transformation, color, size, zoom, filter, Z-axis
XY-pair	Pair of attributes used by the XY-axes motion
M	Major Observation
N	Minor Observation
g_k	Domain-driven goal determined by the AS
g_i	Visual-driven goal determined by the VS
υ_{out}	Externally visualized (outputted) plot
υ_{int}	Internally visualized plot
$\upsilon_{int:S}$	Action command success rate
a_k	Action command from the AS
a_i	Action command from the VS
a_k^M	Number of action commands available for the current XY-pair
α	Maximum allowed number of action commands *per* υ_{int}
o_{max}	Maximum number of action commands for the current M
o^M	Defines the learning state space
$o_a^M = \{a_{k:1}; ...; a_{k:max}\}$	Sequence of action commands from the LM
ϱ_a^M	Vector of rule-based sequences of action commands from the WM
ϱ	Set of rules buffered by the EK into the WM
ϱ_{gk}^M	A weight value sent by the WM to the LM

Table 1. Terminology – IACI Computational Architecture

scientists' thought experiments, as Nersessian (1992) emphasized). Similarly, outputs should easily be seen as a narrative. We understand this is not a trivial problem and we are approaching M and N as an initial attempt to mimic a narrative-like data-exploration and visualization – simulation of emotions and feelings can add into that goal. Another point to consider is our personal preferences and human variability. For example, someone may be biased to apply log transformations while another person may find color a powerful tool towards insights. As Ziemkiewicz et al. (2013) and

Green et al. (2010) point out, even personality may influence our visual exploration. We observed in our human study an unexpected use of motions, such as using lack of color to filter the data. Using motions in a manner other than the designed, is a very human thing to do, and certainly plays a role in innovation and creativity. IACI does not mimic any aspect of that. Another limitation is that IACI does not remove action commands from a plot, only adds – we chose to do so to simplify our research problem. The Human inspiration to develop and test IACI may contribute to the development of Measuring Data Skills tests; for example, helping to frame a visual taxonomy for equation and number distribution comprehension and manipulation. Measuring Data Skills tests can help, for example, in the recruitment of people in the autism spectrum.

Acknowledgements

We would like to thank both Joseph Eilbert for his contributions to this project and Dr. Andreas Berlind who generously shared the galaxy dataset we used in our study. We also would like to thank the Vanderbilt Center for Autism & Innovation. This research makes use of Filtergraph, an on-line data visualization tool developed at Vanderbilt University through the Vanderbilt Initiative in Data-intensive Astrophysics (VIDA).

References

Anscombe, F. J. (1973). Graphs in statistical analysis. *The American Statistician*, *27*, 17–21.

Bastien, F., Stassun, K. G., Basri, G., & Pepper, J. (2013). An observational correlation between stellar brightness variations and surface gravity. *Nature*, *500*, 427–430.

Berlind, A., et al. (2006). Percolation galaxy groups and clusters in the SDSS redshift survey: Identification, catalogs, and the multiplicity function. *The Astrophysical Journal Supplement Series*, *167*, 1.

Bowling, M., & Veloso, M. (2001). Rational and convergent learning in stochastic games. *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence* (pp. 1021–1026). Seattle, WA: Lawrence Erlbaum.

Brafman, R. I., & Tennenholtz, M. (2002). R-MAX – A general polynomial time algorithm for near-optimal reinforcement learning. *Journal of Machine Learning Research*, *3*, 213–231.

Burger, D., Stassun, K., Pepper, J., Siverd, R., Paegert, M., De Lee, N., & Robinson, W. (2013). Filtergraph: An interactive web application for visualization of astronomy datasets. *Astronomy and Computing*, *2*, 40–45.

Cascudo, L. d. C. (Ed.). (2000). *Dicionário do folclore Brasileiro*, volume 1. Instituto Nacional do Livro.

Damásio, A. (2004). *Looking for Spinoza: Joy, sorrow, and the feeling brain*. London: Random House.

Damásio, A. R. (1994). *Descartes' error: Emotion, rationality and the human brain*. New York: Putnam's Sons.

Eilbert, J., Peters, Z., Eliott, F. M., Stassun, K., & Kunda, M. (2018). Shapes in scatterplots: Comparing human visual impressions and computational metrics. *Proceedings of the Fortieth Annual Meeting of the Cognitive Science Society*. Madison, WI: Cognitive Science Society.

Eliott, F., & Ribeiro, C. (2015). Moral behavior and empathy modeling through the premise of reciprocity. *Proceedings of the First International Conference on Human and Social Analytics*. St. Julians, Malta: Huso.

Eliott, F., Stassun, K., & Kunda, M. (2017). Visual data exploration: How expert astronomers use flipbook-style visual approaches to understand new data. *Proceedings of the Thirty-Ninth Annual Meeting of the Cognitive Science Society*. London: Cognitive Science Society.

Gadanho, S. (2003). Learning behavior-selection by emotions and cognition in a multi-goal robot task. *Journal of Machine Learning Research*, *4*, 385–412.

Green, T., Jeong, D., & Fisher, B. (2010). Using personality factors to predict interface learning performance. *Proceedings of the Forty-Third Hawaii International Conference on System Sciences* (pp. 1–10). Honolulu, HI: IEEE.

Healey, C., & Enns, J. (2012). Attention and visual memory in visualization and computer graphics. *IEEE Transactions on Visualization and Computer Graphics*, *18*, 1170–1188.

Keim, D. (2000). Designing pixel-oriented visualization techniques: Theory and applications. *IEEE Transactions on Visualization and Computer Graphics*, *6*, 59–78.

Kunda, M., McGreggor, K., & Goel, A. (2013). A computational model for solving problems from the Raven's Progressive Matrices intelligence test using iconic visual representations. *Cognitive Systems Research*, *22*, 47–66.

Leman, S. C., House, L., Maiti, D., Endert, A., & North, C. (2013). Visual to parametric interaction (v2pi). *PLoS ONE*, *8*.

Matejka, J., & Fitzmaurice, G. (2017). Same stats, different graphs: Generating datasets with varied appearance and identical statistics through simulated annealing. *Proceedings of the 2017 Conference on Human Factors in Computing Systems* (pp. 1290–1294). Denver, CO: Association for Computing Machinery.

McGreggor, K., Kunda, M., & Goel, A. (2014). Fractals and ravens. *Artificial Intelligence*, *215*, 1–23.

Nersessian, N. (1992). In the theoretician's laboratory: Thought experimenting as mental modeling. *Proceedings of the Biennial Meeting of the Philosophy of Science Association* (pp. 291–301). The University of Chicago Press.

Rosling, H. (2013). The Ignorance Project. Retrieved July 9, 2018, from `https://www.gapminder.org/ignorance/`.

Sedlmair, M., Tatu, A., Munzner, T., & Tory, M. (2012). A taxonomy of visual cluster separation factors. *Computer Graphics Forum* (pp. 1335–1344). Wiley Online Library.

Shallue, C., & Vanderburg, A. (2018). Identifying exoplanets with deep learning: A five-planet resonant chain around Kepler-80 and an eighth planet around Kepler-90. *The Astronomical Journal*, *155*, 94.

Sher, V., Bemis, K., Liccardi, I., & Chen, M. (2017). An empirical study on the reliability of perceiving correlation indices using scatterplots. *Computer Graphics Forum*, *36*, 61–72.

Sun, R., & Peterson, T. (1998). Autonomous learning of sequential tasks: Experiments and analyses. *IEEE Transactions on Neural Networks*, *9*, 1217–1234.

Sutton, R., & Barto, A. (1998). *Reinforcement learning: An introduction*. Cambridge: MIT Press.

Wilkinson, L., & Anand, A. (2015). *Package* `scagnostics`. Package maintainer: Urbanek, S.

Wilkinson, L., Anand, A., & Grossman, R. (2005). Graph-theoretic scagnostics. *Proceedings of the 2005 IEEE Symposium on Information Visualization*. Minneapolis, MN: IEEE.

Wilkinson, L., Anand, A., & Grossman, R. (2006). High-dimensional visual analytics: Interactive exploration guided by pairwise views of point distributions. *IEEE Transactions on Visualization and Computer Graphics*, *12*, 1363–1372.

Ziemkiewicz, C., Ottley, A., Crouser, J., Yauilla, A. R., Su, S., Ribarsky, W., & Chang, R. (2013). How visualization layout relates to locus of control and other personality factors. *IEEE Transactions on Visualization and Computer Graphics*, *19*, 1109–1121.