# Insight: Interactive Machine Learning for Complex Graphics Selection

by

Sumit Gogia

B.S., Massachusetts Institute of Technology (2015)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2016



Author . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
Department of Electrical Engineering and Computer Science
May 20, 2016

Certified by . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
Dr. Mitchel Resnick
LEGO Papert Professor of Learning Research
Thesis Supervisor

Accepted by . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .
Dr. Christopher Terman
Masters of Engineering Thesis Committee

# Insight: Interactive Machine Learning for Complex Graphics Selection

by

Sumit Gogia

Submitted to the Department of Electrical Engineering and Computer Science
on May 20, 2016, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

## Abstract

Modern vector graphics editors support the creation of a wonderful variety of complex designs and artwork. Users produce highly realistic illustrations, stylized representational art, even nuanced data visualizations. In light of these complex graphics, *selections*, representations of sets of objects that users want to manipulate, become more complex as well. Direct manipulation tools that artists and designers find accessible and useful for editing graphics such as logos and icons do not have the same applicability in these more complex cases. Given that selection is the first step for nearly all editing in graphics, it is important to enable artists and designers to express these complex selections.

This thesis explores the use of interactive machine learning techniques to improve direct selection interfaces. To investigate this approach, I created Insight, an interactive machine learning selection tool for making a relevant class of complex selections: visually similar objects. To make a selection, users iteratively provide examples of selection objects by clicking on them in the graphic. Insight infers a selection from the examples at each step, allowing users to quickly understand results of actions and reactively shape the complex selection. The interaction resembles the direct manipulation interactions artists and designers have found accessible, while helping express complex selections by inferring many parameter changes from simple actions. I evaluated Insight in a user study of digital designers and artists, finding that Insight enabled users to effectively and easily make complex selections not supported by state-of-the-art vector graphics editors. My results contribute to existing work by both indicating a useful approach for providing complex representation access to artists and designers, and showing a new application for interactive machine learning.

Thesis Supervisor: Dr. Mitchel Resnick
Title: LEGO Papert Professor of Learning Research

# Acknowledgments

First and foremost, my thanks goes to Jennifer Jacobs, who has been an incredible help through every part of the work in this project, and in no small part provided my inspiration for pursuing it in the first place. She has been both an amazing mentor, and a wonderful friend.

I next want to thank Drs. Joel Brandt and Wilmot Li from Adobe Research, who have also provided guidance throughout this project, and in general taught me much about user interface design. I remember them telling me they wanted to instill in me an understanding of the need for interfaces that empower humans, and people to create them. I did not feel it was particularly important to me at the time, but I have become more than convinced.

Next are Dr. Mitchel Resnick and all of the members of the Lifelong Kindergarten Group. They have kindly provided me with a place to learn and explore this research to my heart's content, and have always made themselves available to support me.

Thanks goes to all of my friends from MIT's Japanese Lunch Table, for reminding me weekly of the beautiful feeling of having someone from an entirely different culture do their best to understand you.

My friends Parker, Lucy, and Min also deserve many wells of gratitude for constantly making me smile and look forward. I have no doubt that I would not have even written this thesis without the constant fun and whimsy of our times together.

I want to thank Cole, Max, DK, and Diana for straight up being my homeslices.

Finally, I thank my parents and sister for being supportive for all these years. I am glad to say that they have always tried to make my dreams come true.

THIS PAGE INTENTIONALLY LEFT BLANK

# Contents

# List of Figures

# Chapter 1

# Introduction

With remarkable advances in computation, digital art and design interfaces enable users to create increasingly complex compositions. However, as some of our tools begin to afford creation of this complexity, the other tools we use need to keep pace. An important case is *selection tools* in editors for vector graphics, a medium used ubiquitously in digital art and design. Selection specifies a *representation* of a set of objects to edit in the graphic. It is a preliminary step before every edit operation, making it important for it to be highly efficient and accessible. However, though the popular direct manipulation selection tools are very effective for simple artwork and designs, they are less effective in complex compositions:

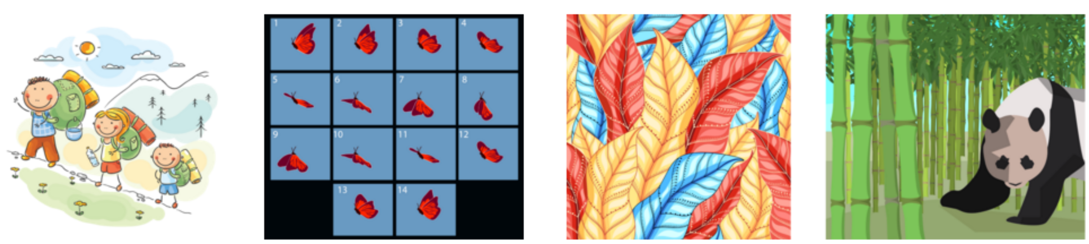


Figure 1-1: Instances of complex artwork where direct manipulation selection is less helpful. In each of these, there are sets of objects that stand out which cannot be individually clicked or region-selected easily. For example, in the panda image, selecting the leaves would be very tedious.

There are two fundamental reasons that direct manipulation tools are difficult to use in these cases. First, the representation users perceive for the selection is different from one which uses only screen position. Users can see shape similarity, contiguity, or resemblance to real-world objects as important to the selection structure. Second, to use the parametric representations we know for these other relationships we must specify many parameters, each taking many values. With direct manipulation tools, users adjust very small numbers of parameters with each action, and must spend time moving through continuous representations of parameters. Drawing a rectangle to specify $x$- and $y$-bounds requires users to spend time dragging, while the rectangle only has two corner points as parameters.

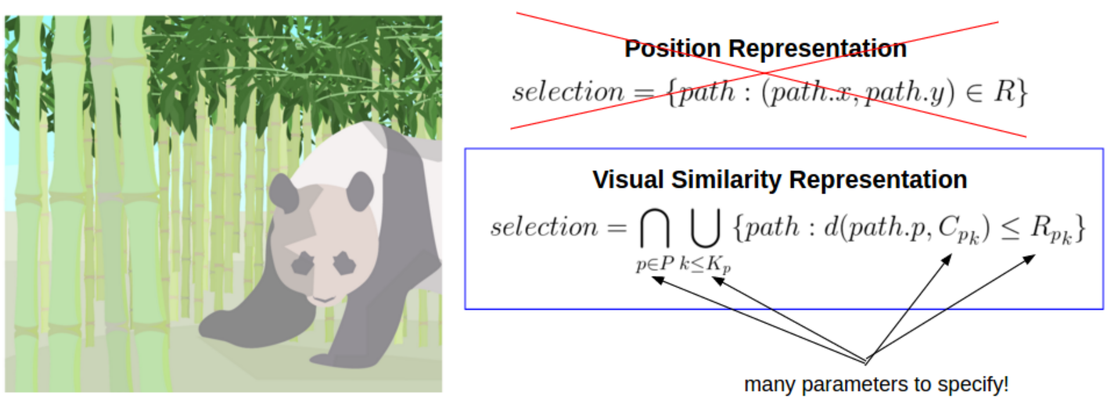



Figure 1-2: A visualization of the difficulties with direct manipulation selection tools. Note that we perceptually group the leaves because of their similarity in appearance, not their positions. Visual similarity is difficult to specify since it has many parameters.

Some approaches to managing this complexity involve leveraging the representational power of computers. We can add auxiliary data to that visible in the medium, such as explicitly marking groups and layers; or we can use automated processes to infer representation parameters from simple interactions, such as finding all objects that have similar shape under a stroke.

Unfortunately, each of these approaches has drawbacks given the practices digital artists and designers are familiar with. Auxiliary data requires users to keep track of a representation outside of the graphic. For example, in complex graphics where the number of groups and layers becomes large, users cannot remember what each of the groups is, and it becomes difficult to link them to the artwork. On the other hand, inferential tools tend to take control away from users' actions since they do not recognize users' goals effectively. When finding similarly-shaped objects under a stroke, an inferential system can easily be confused by seeing multiple sets of very similar objects.

In general, while these approaches potentially allow users to express complex representations, they are not accessible to artists and designers. This concern inspires my research question:

**Research Question**

*How do we design graphics selection tools that are both expressive and accessible to artists and designers?*

In this thesis, I present Insight, a vector graphics selection tool aimed at enabling artists and designers to make complex, expressive selections with simple, intuitive interactions. Insight uses an interactive machine learning approach which has seen increasing popularity for addressing complex decision tasks. Users iteratively provide examples of selection objects by clicking in the medium's visual representation, and Insight infers a selection of visually similar objects at each step. This incremental inference approach allows users to effectively shape the selection representation in a manner similar to direct manipulation, without the tedium of completely explicit specification.

I designed Insight's interface, developed its novel inference algorithm, and tested Insight in a user study of digital artists and designers. Users found the interaction simple and useful, discovering that they could effectively and easily make complex selections not supported by state-of-the art vector graphics editors. Though users did note that they could not control Insight as well as direct manipulation tools, they felt that the iterative example-based inference was important for navigating the selection representation. My work contributes to existing work by both indicating a useful approach for providing complex representation access to artists and designers, and showing another application for interactive machine learning.

The remainder of this thesis is structured as follows: Chapter 2 discusses background and related work, illuminating design goals for Insight. Chapter 3 explicitly lists the design principles obtained from examining works from Chapter 2. Chapter 4 describes in detail the design of Insight, including both the interaction and supporting inference algorithm. Chapter 5 details the user study performed to evaluate Insight, and the specific hypotheses it tested. Chapter 6 describes the results from the user study. Chapter 7 discusses the results and their relevance to Insight's specific design choices. Lastly, Chapter 8 is a conclusion to the work.

# Chapter 2

# Background and Related Work

In order to inform my development of Insight, I examined two major bodies of work. First, I examined literature and systems for editing digital graphics to acquire specific design goals. This exploration led me to the general approach of interactive machine learning. I then examined literature and systems in interactive machine learning as well to more particularly shape Insight's design.

## 2.1 Editing Digital Graphics

A variety of tools exist for enabling artists and designers to edit digital graphics. Those which provide the largest range of creative outcomes are creative code languages such as Processing [14] and [8]. These languages give users a library of data structures and functions useful for describing graphical content. Users can specify representations of their content and actions on it as needed, enabling selection and editing of the complex representations described in Section 1. Unfortunately, programming remains difficult for many artists and designers to adopt. They must go through an intermediate, abstract text interface to intermittently make changes to their graphic; and they must accustom themselves to structured problem-solving techniques in conflict with their exploratory practices [27]. These differences in practices are barriers that artists and designers frequently find difficult to overcome [11].


Figure 2-1: Examples of different types of tools for editing graphics, with mouse motion indicated by a dashed line. On the left is direct manipulation, in the middle is programming, and on the right is inference. Both programming and inference notably let users edit complex relationships, but it is often unclear to artists how actions will be realized on the graphic.

The tools common to digital graphic editing follow *direct manipulation* principles. In direct manipulation interfaces, users perform rapid, incremental, reversible actions directly on visual representations of objects of interest [21]. These interfaces typically allow users to interact directly with their artwork and see the results of their actions in real-time, reminiscent of the reflective conversation with medium that Schon notes is characteristic of the design process [20]. Examples of direct manipulation selection tools are those for specifying objects by clicking on or dragging regions around them. Tools such as the Magic Wand and grouping also use direct manipulation, but not on objects in the graphic, instead having the user work through other representations. Unfortunately, while these tools find great use for artists and designers, they are less useful when the graphic complexity increases and perceived object relationships are not in terms of position (see Section 1). Though there are multiple recent direct manipulation tools that enable people to create complex relationships like those produced with programming [23, 5, 19], these are not adapted for the case of selection.

A final approach that has found its way into graphics is *inference*. Rather than have the user go through a lengthy direct manipulation, parameters for actions on complex representations of the graphic are inferred from simple user actions. A good example of this approach which has found wide popularity is Content Aware Fill in Photoshop, with which the user can draw a region and have it filled to resemble surrounding content. The user does not need to understand things like boundary smoothness conditions that the tool might use to infer changes to the artwork. This approach has

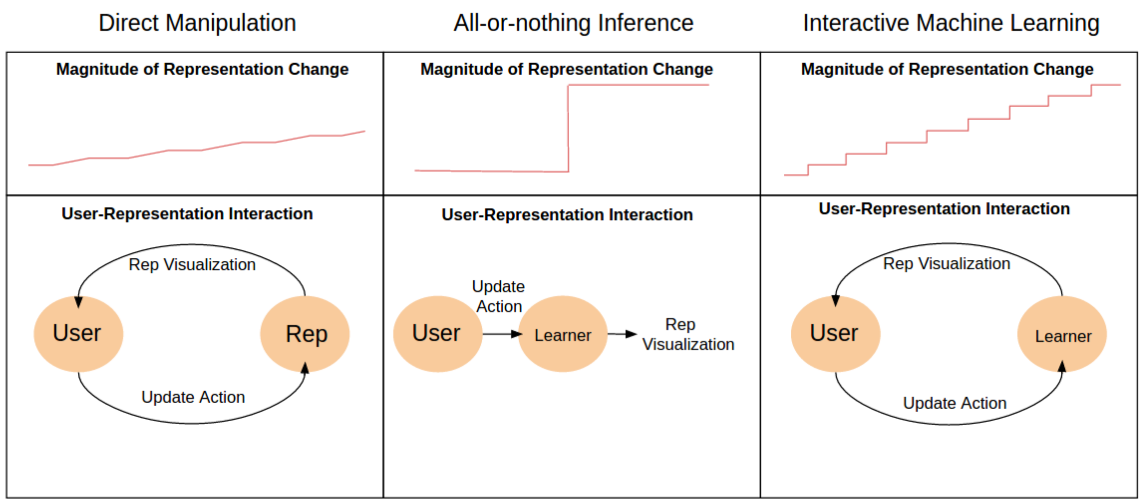



Figure 2-2: A visualization of significant differences between direct manipulation, all-or-nothing inference, and interactive machine learning. Direct manipulation has rapid feedback cycles directly with the representation but adjusts it slowly; all-or-nothing inference adjusts the representation quickly via a learner but with no feedback; and interactive machine learning adjusts the representation at a moderate speed via a learner with rapid feedback.

recently been seen to be effective in selection as well, with both Lazy Select [28] and Suggero [9] inferring objects that perceptually group from user strokes on a surface. However, despite their success, these approaches notably reduce the user's level of explicit control, an important quality afforded by direct manipulation.

Ideally, selection tools for art and design need to support a variety of selection types through interactions which are familiar and accessible for artists. Therefore, I rely on a combination of direct manipulation principles and inference-based approaches. These two find harmony in interactive machine learning, which I discuss in the next section.

## 2.2 Interactive machine learning

The inference-based approach I take categorizes as *interactive machine learning*. Interactive machine learning refers to an approach for letting users interactively train representation parameters with rapid, focused, and incremental model updates. Good examples of interactive machine learning systems are found in other selection environments. In text editing, LAPIS [12] allowed users to iteratively specify positive and negative text samples to infer all text samples satisfying a set of rules; Ritter and Basu created a system which similarly took positive and negative examples in a file browser and inferred a general file query [16]. For both systems, users interactively chose new examples as they saw the selection change, rather than specifying all examples at once. These sample systems show that the primary theme of interactive machine learning is *feedback*, since users shape the representation [1]. This is in contrast to all-or-nothing inference tools like Content-Aware Fill in Photoshop, where the user specifies a region to be filled once and then cannot *adjust* that filling afterwards.

Notably, interactive machine learning bears many similarities to direct manipulation: users manipulate a representation with quick and small *update actions*, adjusting these actions as they see the representation change. The significant difference is in the inferential nature of the representation updates themeselves: the results of actions are not explicitly understood by users prior to execution. The approach by nature posits that explicit action is not a requirement, and that only a general understanding of how actions are adjusting towards the target representation is important [3].

Conversely, since inference systems do not constrain the types of updates a user can make, it is very important to design interfaces and interactions that enable users to effectively enact changes. To this end, a variety of systems have explored different types of update actions and means to convey action results to users. Kulesza et al. showed that users could additionally use song ratings and explicit genre selection to improve an example-based music recommender, excited about the extra control [6]. Rosenthal and Dey showed that exposing lower level features such as uncertainty measures made users less effective at email-labeling [18].

I draw on a set of principles for designing interactive machine learning systems has been extracted in an excellent survey by Amershi et al. [1]. Many of these principles coincide with ideas of direct manipulation, such as support for representation assessment. Notably though, a number of these principles favor providing more control than the simple, concrete interaction of example provision. While further control can be helpful for artists and designers [15], the means for further control often involve users manipulating views of representations outside the original medium, reducing simplicity and concreteness important for this use case. Balancing these tensions is an important issue in this work, and results contribute to both discussions of general interactive machine learning systems design as well as that specific to artists and designers.

THIS PAGE INTENTIONALLY LEFT BLANK

# Chapter 3

# Design Goals

The discussions in the above sections imply a number of design goals for selection tools for artists. These goals primarily reflect general needs of selection to be fast and expressive, but also reflect particular needs of artists described in Chapter 2.

1. **Efficiency:** Can users make target selections quickly? Can they do so with few actions?
2. **Expressiveness:** Can users make a variety of selections they find relevant?
3. **Accessibility:** Do users feel like the system is easy to adopt and use?
4. **Exploration:** Can users make selections in multiple ways? Are actions incremental and reversible?
5. **Concreteness:** Is action directly on objects of interest? Do users need to interact with representations outside the medium?
6. **Control:** Can users predict the results of their actions? Can users react to the results of their actions?

THIS PAGE INTENTIONALLY LEFT BLANK

# Chapter 4

# System Description

Insight is implemented as an interactive machine learning tool for selecting visually similar paths in vector graphics environments. Users train the selection representation by iteratively clicking examples of paths they perceive as similar. They choose subsequent examples by evaluating a visualization of all currently selected paths. The example-based interaction makes Insight concrete, while small, reversible inference actions help provide control and promote exploration. Expression and efficiency are handled by inference: the visual similarity model allows for arbitrary sets of similarity cues at multiple ranges to determine the selection, while a small number of logarithmic-time tree searches forms the bulk of inference computation. The main tradeoff is between control and simplicity: rather than provide users with deeper access to the similarity representation, such as the ability to directly specify relevant cues, interaction is limited to providing examples.

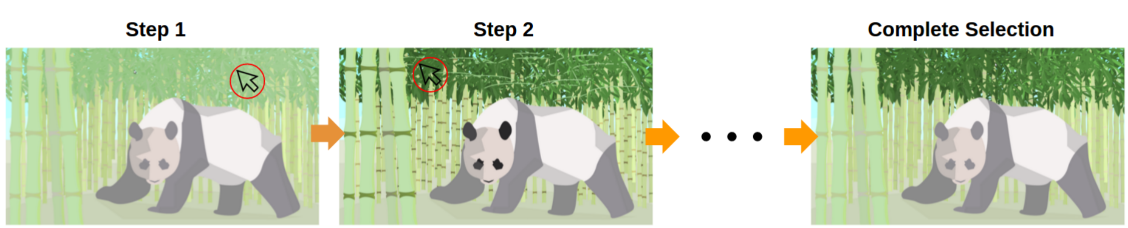



Figure 4-1: Sample usage of Insight. The pointer indicates the location of a click for providing an example. Examples are provided iteratively until the target selection is reached.

I describe the details of both Insight's interface and inference in the sections below, along with design alternatives, tradeoffs, and justification of how design choices address goals.

## 4.1 Interface

### 4.1.1 Example Provision

To describe the selection representation's parameters, users provide only positive examples of vector paths they believe describes the similarity. These examples are provided individually by clicking on example paths, and each example incrementally adjusts the representation. The inference algorithm is agnostic to the order of examples provided.

There are many reasons for using an example-based interaction. Foremost, it is concrete and similar to direct manipulation tools, since users can see and directly point to the example paths in the medium. Without an example-based approach, the user must resort to working in an abstract representation of the artwork, since properties such as color cannot be described in terms of artwork position. Secondly, example-based interaction naturally lends itself to representation refinement. The number of parameters in the visual similarity representation makes accurate inference difficult (see Section 4.2.2), showing a need for a refinement process. This inferential selection refinement is unique to Insight among vector graphics selection tools. Though LazySelect [28] and Suggero [9] also use inference and target complex selection representations, users are not able to adjust the inference.

The choice to have users provide examples by clicks one-by-one is based in the accuracy of clicking. Complex graphics can have many occlusions and high path density. Clicking unambiguously provides only examples, while marquees would inevitably include unwanted paths. This does not exclude use of regions to provide examples completely. Example provision could itself have an inferential step, where only paths very similar in the drawn region feed into the example-based inference. For this project, I was intent on exploring first the use of any inferential selection refinement, and so restricted to click-based example provision.

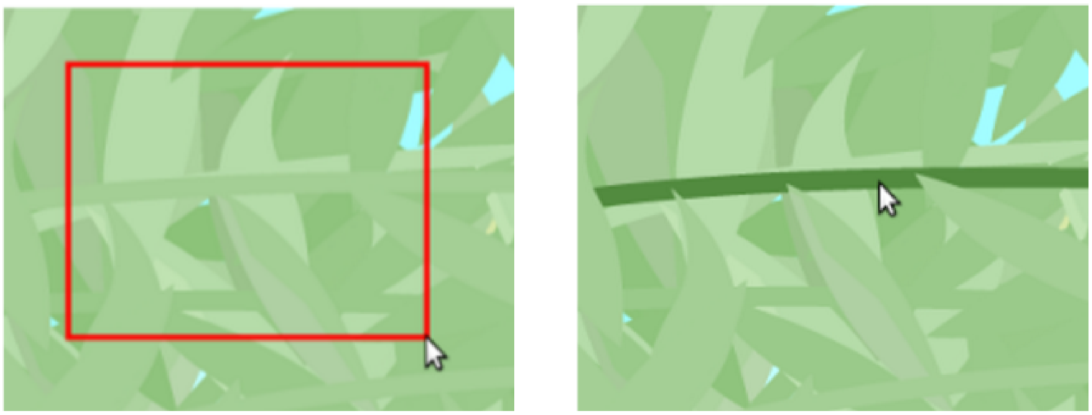


Figure 4-2: A depiction of region-based example provision. Trying to provide leaf examples by drawing a rectangle around them inevitably grabs an unwanted thin strand as well.

Inference being agnostic to example order is important to making the interaction concrete. This choice is different from tools such as LazySelect [28], which even uses parameters such as interaction speed to influence inference. However, if example order changed inference, it would mean that users' model for similarity needed to account for example order. Visual similarity descriptions from Feature Integration Theory maintain that perceived similarity is a *shared* property [26, 24, 25], indicating that an order-dependent model is at odds with user perception.

A last notable decision is the use of only positive examples. Many example-based interactive machine learning tools allow for both positive and negative examples, including selection tools in other environments. However, in the case of selection, the use of only positive examples is made possible due to non-examples serving implicitly as weak negative examples. This choice makes an important simplification to the interface at the cost of negative examples' ability to modify the representation differently. While negative examples could help to make inference converge more quickly, their usage would require the user to keep in mind a second example set. In addition, it is frequently more difficult for users to give helpful negative examples as opposed to positive examples [1].

### 4.1.2 Learning Feedback

The user receives feedback on the learned selection representation by directly viewing a visualization of the matching paths after each example. At the start of interaction with Insight, all vector paths in the graphic are rendered at 50% of their original lightness value. As examples are added, paths matching the selection have their lightness restored to normal, while paths removed from the selection are darkened again. This results in only paths in the selection having original lightness at any point of Insight's use. Users can further visualize the selection by toggling the visibility of selected paths.

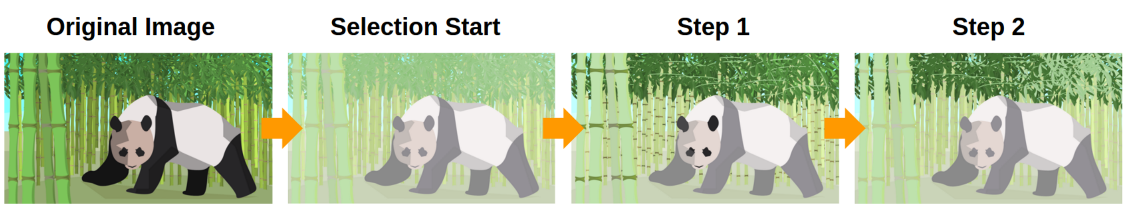



Figure 4-3: Depiction of selection visualization as examples are provided.

Providing feedback at each step is important to helping users understand the results of their actions [1]. Artists and designers look for feedback on all actions that change system state. Even for users who are only interested in seeing the results of inference after multiple examples, they can easily ignore the feedback since it does not interrupt example provision.

The feedback visualization shows all paths selected according to the current selection representation. This is in similar to previous selection tools [12, 16]. Particular to selection in editing environments is that it needs to be *exact*. Users perform edits on the specific items they want, and so want to know the selection exactly. Showing all paths selected allows users to quickly get a close understanding of representation accuracy, guiding them to provide more examples or examine accuracy more closely with visibility toggling.

Insight uses lightness to visualize the selection primarily due to complexities of graphics. Graphics can contain large numbers of paths in a variety of colors, packed together very closely. If outlines or bounding boxes like those in Adobe Illustrator are used, paths with similar colors lose visibility, particularly in graphics with large numbers of paths. Adjusting lightness helps distinguish selected paths for a wide variety of colors, and avoids adding any extra clutter when lots of paths are present. Despite this, the halving lightness adjustment is not appropriate for paths with very low lightness, since changes are minimal. The search for a better selection visualization remains an important issue.

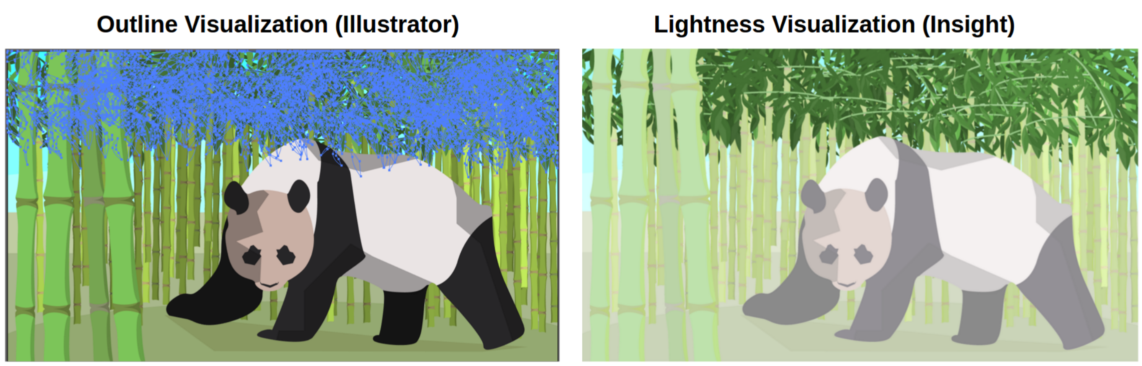



Figure 4-4: Depiction of differences between use of outline for selection visualization and lightness adjustment for selection visualization.

Allowing users to further visualize the selection by toggling the visibility of selected paths is an example of providing freedom to query representation at will, an important quality for interactive machine learning systems [1]. Though users already see the results of actions after each example, the visualization provided is for generally understanding the representation. It can be difficult in crowded graphics to clearly see which exact paths are selected. Toggling the visibility of paths helps users exactly recognize all selected paths.

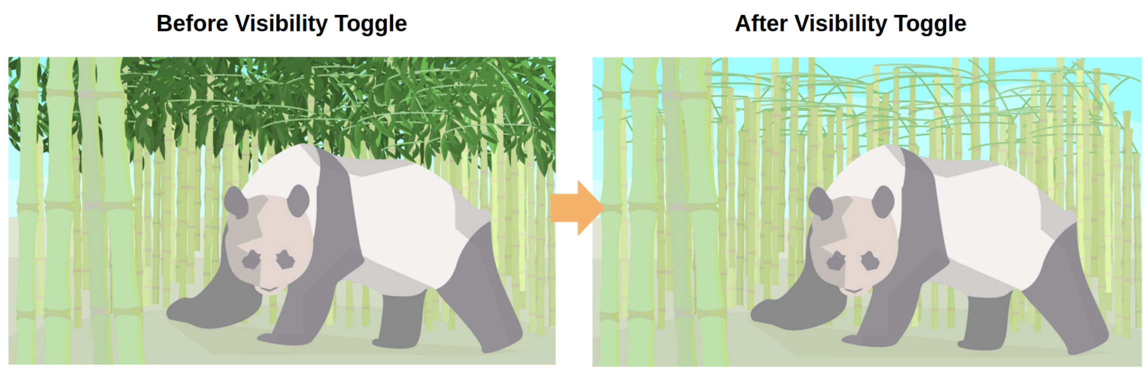



Figure 4-5: Depiction of selection visibility toggling. The selection becomes clearer when removed from the image.

### 4.1.3 Representation Exploration

Insight provides undo and redo operations to let the user regress or progress one example respectively, and then update the selection visualization. Both operations are triggered via key-presses, similar to existing vector graphics software. The history only contains the example sequence taken to obtain the current selection. If a user performs an undo operation and subsequently clicks a new example, the most recent example just previous to the undo is lost.

Undo and redo are provided to aid in representation exploration. By allowing users to traverse back through the example stream and provide other examples, Insight helps them try alternatives when they see unexpected results. On the other hand, not keeping track of the different alternative example streams for the user places burden on users to remember old example streams and their results. Features such as an undo tree or selection visualizations for older training states could aid in managing selection states, but would also complicate usage. Adding visualizations of selections for other example sets like in CueFlik [4] would be difficult, since graphics are complex. Notably, it is difficult to even do selection visualization in the main graphic view. Without such visualization, an undo tree would still require users to remember how their branches had performed, not relieving the original issue.

### 4.1.4 Refining Selection with Other Tools

Insight is explicitly designed to complement, rather than replace, other vector graphics selection tools. As with other selection tools, when users switch to different tools in the environment the selection is saved. Users can thus use other selection tools, such as manual click and lasso selection, to refine a selection produced by Insight. While parameters of the selection representation Insight trains are not adjusted, other tools describe their own selection representations and then combine with Insight's by add or remove set operations.

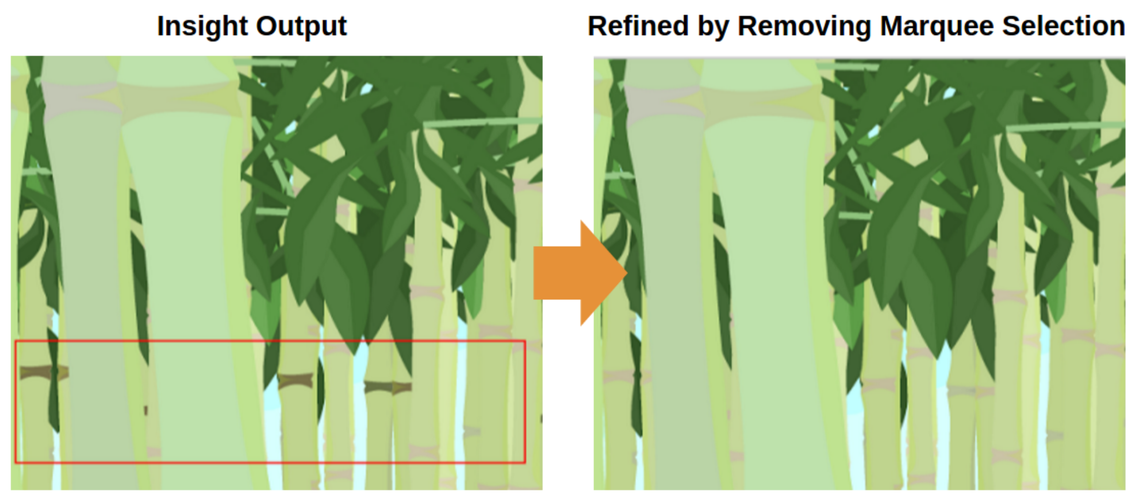



Figure 4-6: Depiction of selection refinement with manual tools. Here, a rectangular marquee is used to remove the small number of extra unwanted paths.

Allowing use of other selection tools to refine the selection is an example of providing means for inference critique [1], since users can adjust the end selection representation. Inference critique through other selection tools improves Insight's efficiency in cases where the inference gets very close to the target selection. Rather than continue providing examples with some uncertainty about when the target will be reached, it can be much faster to switch to a manual tool and remove or add the small difference.

Notably, other selection tools do not critique Insight's *learning*, as is the case with example provision. While more critiquing power could be given by allowing the user to manually specify certain parameters of the selection representation, this would require work in an abstract artwork representation, reducing concreteness. Enabling users to specify the importance of an example through other methods (i.e. the length of the click) would also add complexity to Insight's interaction set. Instead, Insight prioritizes speed, and simplicity.

## 4.2 Inference

Insight's inference consists of a selection representation and novel associated learning algorithm, which takes example paths and infers parameters for the selection representation. I chose to represent visual similarity for selection because of its expressiveness and relevance to artists and designers [13, 17, 9]. The representation I developed provides a high level of expression by allowing for multiple similarity cues and ranges described in Feature Integration Theory [26, 24, 25]. The learning algorithm finds the cues and ranges specified by the user through examples.

### 4.2.1 Similarity Cue Extraction

Paths in vector graphics are represented as sets of appearance properties, each with its own representation. For example, shape is represented as a Bezier curve, while fill color is represented as an RGB triplet. It is challenging to measure perceptual similarity from the default representations, so on activation Insight extracts needed similarity cues from all paths for a more convenient representation. Translating once helps avoid computational burden throughout the selection process.
The cues chosen for extraction along with their extraction means are shown below:

- **Shape:**
    1. Sample 64 evenly-spaced boundary points for the path using the Bezier curve.
    2. Compute squared distances from each point to the centroid.
    3. Extract rotation- and scale-invariant Fourier descriptor using the squared centroid distances.

- **Fill Color:**
  1. Convert the fill color to a LAB representation.
- **Stroke Color:**
  1. Convert the stroke color to a LAB representation.
- **Stroke Width:**
  1. Transfer directly from the path representation.
- **Scale:**
  1. Sample 64 evenly-spaced boundary points for the path using the Bezier curve.
  2. Compute eigenvalues and eigenvectors of the covariance matrix for boundary points.
  3. Extract principal axis and secondary axis of bounding ellipse with eigenvectors and eigenvalues.
- **Orientation:**
  1. Compute angle between principal and secondary axes obtained for scale.
  2. Break sign invariance by computing area above principle axis.
  3. Normalize to range $[-1, 1]$.
- **Position:**
  1. Transfer directly from the path representation.

The majority of properties here are directly relevant to visual similarity according to Feature Integration Theory [26, 24, 25], with position being the single exception. Insight supports position for a subtle reason: in most vector graphics editors, selection updates *by intersection* with new selections are not possible. While users can adjust selections by adding or removing new selections made with other tools, the lack of intersection updates makes it difficult to select paths by both position and other appearance properties. Ideally, position would not be included as a property in Insight. However, to support Insight's applicability for testing, position was included.

The specific representations chosen for each of the cues describe Euclidean spaces in which Euclidean distance correlates significantly with perceptual similarity. For example, shape similarity measured by distance in the Fourier domain is strongly correlated with perceived similarity [2]. This quality is important both for describing the selection representation and making parameter learning feasible.

It is important to point out that vector paths do support other properties, such as opacity. For this work, the focus is on providing a relevant and expressive similarity representation. The set of cues chosen is large enough to make the selection representation highly expressive (see Section 4.2.2), and so leaving out these extra properties does not reduce the utility of the system. In addition, Insight's inference is fully capable of supporting these other properties, as their processing is similar to the other properties in the chosen set. Opacity, for example, can be directly transferred just like stroke width.

### 4.2.2 Selection Representation

The selection representation is a parametric description of visually similar paths for a given image. Parameters are independent and variable. Different sets of values for the parameters correspond to the different ways people can interpret similarity. The range of values the parameters can take then describes all ways in which users can interpret similarity. When users provide path examples, values for all parameters are learned, fixing the representation to a particular interpretation of similarity the user intends to describe.

I first give an explicit mathematical description of this representation, then explain how it parameterizes visual similarity, and lastly describe how it supports the design goals of expression and relevance for artists.

Let $P \subset \{shape, fill, \ldots, scale\}$ be an arbitrary subset of similarity cues, and $\{C_{p_k}\}_{k=1}^{K_p}$ be an independent finite set of point clusters in $p$'s Euclidean space for each $p \in P$. Define the set of clusters for all cues in $P$ by:

$$C_P = \{\{C_{p_k}\} : p \in P \land k \leq K_p\}$$

The selection representation is then:

$$S(P, C_P) = \bigcap_{p \in P} \bigcup_{k \leq K_p} C_{p_k} \tag{4.1}$$

This representation describes visual similarity by encoding the rules from Feature Integration Theory. Namely, this representation allows for arbitrary subsets of similarity cues to be active in describing the similarity, and arbitrary ranges of similarity for each of the cues.

Activity of arbitrary subsets of similarity cues is described in the outer intersection. For the moment, take the inner union to describe a *similarity group*, a set of paths interpreted as similar for a given cue $p$. Then,

$$(\text{Overall Similar Paths}) = \bigcap_{p \in P} (\text{Similarity Group for } p) \tag{4.2}$$

describes the set of paths which are similar for each of the cues $p \in P$. Since $P$ is freely let to be an arbitrary subset of similarity cues, it follows that this intersection allows for similar paths to be expressed as paths that are similar for each of any subset of similarity cues. In terms of the graphic, paths could be considered similar if they are copies positioned differently, or if they just have the same shape and fill, or if they have only similar scale. During usage of Insight, the set of active cues $P$ is learned.

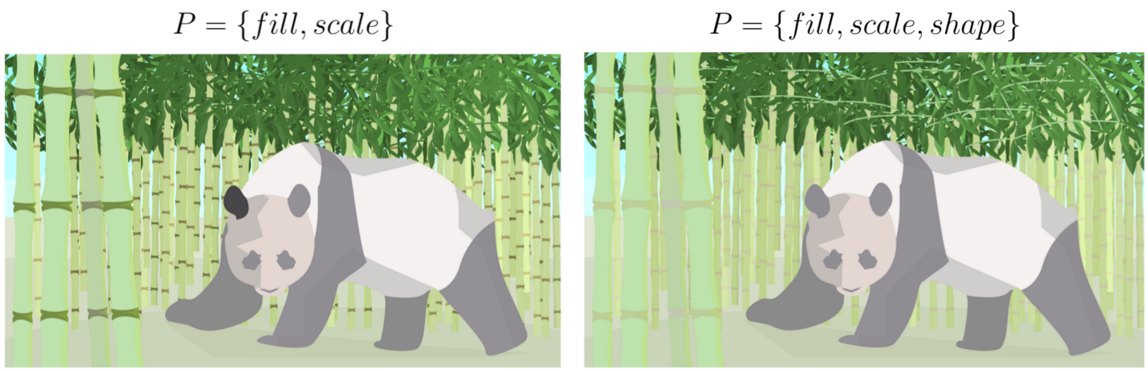



Figure 4-7: Depiction of similarity with different sets of similarity cues. On the left, only fill color and scale describe the similarity, while on the right shape is relevant as well. Generally, similarity can be determined by an arbtirary set of cues.

The possibility for arbitrary ranges of similarity for any similarity cue is described in the inner union. This is best seen by first replacing the union of clusters by a single cluster:

$$\text{(Similarity Group for } p) = \bigcup_{k \leq K_p} C_{p_k} = C_p \tag{4.3}$$

Each of the similarity cues takes values in a Euclidean space. Euclidean distance in these spaces corresponds to perceptual distance. Since a cluster $C_p$ for a cue $p$ contains all path values for the cue which group closely together in the space, it follows that an arbitrary cluster can describe an arbitrary range of similarity for the similarity cue. Concretely, a color cluster lets us describe paths similar for color as those very close to a particular shade of red, or as just generally green. Insight then must learn clusters for each cue.

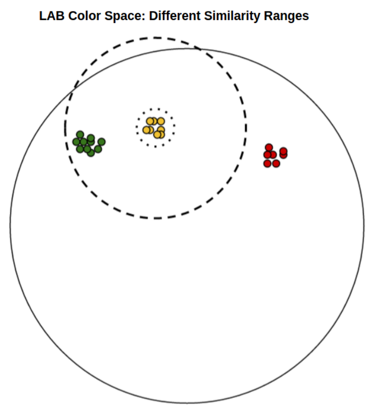



Figure 4-8: A visualization of different similarity ranges for a similarity cue. In this 2D LAB color space projection, depending on the user's intention either just the yellow items or both yellow and green items could be considered similar. These similarities would use different clusters.

It would appear that the previous two parts are enough to describe visual similarity according to Feature Integration Theory:

$$\text{(Overall Similar Paths)} = \bigcap_{p \in P} B_p \tag{4.4}$$

However, in practice, the Euclidean distance metrics used for each of the similarity cues do not perfectly correspond to perception. This can result in clustering metrics producing inaccurate clusters. People may also perceive similarity on axes such as lightness for color, which can have a haphazard shape in LAB space. For this reason, the description of similarity level for a similarity cue as a single cluster is relaxed to a flexible cluster union:

$$C_p \rightarrow \cup_{k \leq K_p} C_{p_k} \tag{4.5}$$

Such a finite cluster union exists to cover *any* set of values in the space. Thus, any set of paths can be considered similar for a given similarity cue. The consequence of adding this flexibility is that the parameters to be learned increases to include the number of clusters $K_p$ and the particular clusters. However, this relaxation is essential for providing representation of *all* interpretations of similarity.

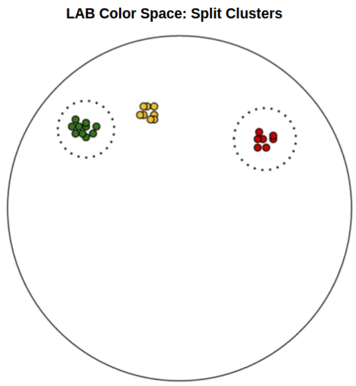



Figure 4-9: A visualization of a case where the user may need to use multiple clusters to describe the similarity. The user may find that red and green paths are both dark, but yellow is not.

It is good to review how this representation is relevant to this work. Firstly, it is clearly very expressive in that it supports a variety of interpretations of similarity. This is true not just in the sense that abstractly any set of paths can potentially be considered similar, but in the very real case of actual graphics. Graphics do contain paths which are distributed very freely in these spaces, requiring the flexibility in the representation.

The flexibility of this representation is also in contrast with previous work in Suggero [9] and LazySelect [28]. In both of these systems, similarity is described as a single cluster or cluster in a Euclidean space incorporating all cues. The assumption is that the similarity distance used corresponds well to visual perception, but to my knowledge there is no known metric that completely describes similarity between objects. If users are to be capable of describing similarity as they see fit, it must be the case that the representation allows for arbitrary interpretations. Of course, it is still necessary to make sure that *common* parameter values are accessible quickly. This is handled by appropriately biasing the learning algorithm.

### 4.2.3 Learning Algorithm

The learning algorithm is responsible for determining the parameters to the selection representation. It takes as input the graphic paths and examples, and outputs the following representation parameters:

1. The list of active cues $p$.

2. The number of clusters for each active cue, $\{K_p\}_{p \in P}$.

3. References to each of the clusters for each cue.

The primary goals for the learning algorithm are to ensure that interaction is efficient, and allow for access to the expressive range of the representation. These two goals are generally in conflict in learning systems. Ideally, users could describe all kinds of representations with very few examples. However, an example provides evidence for a large number of parameter sets. Uniquely identifying any of the parameter sets will require large numbers of examples. If the learning algorithm is to be efficient, it must then bias towards the parameter sets that are more frequent.

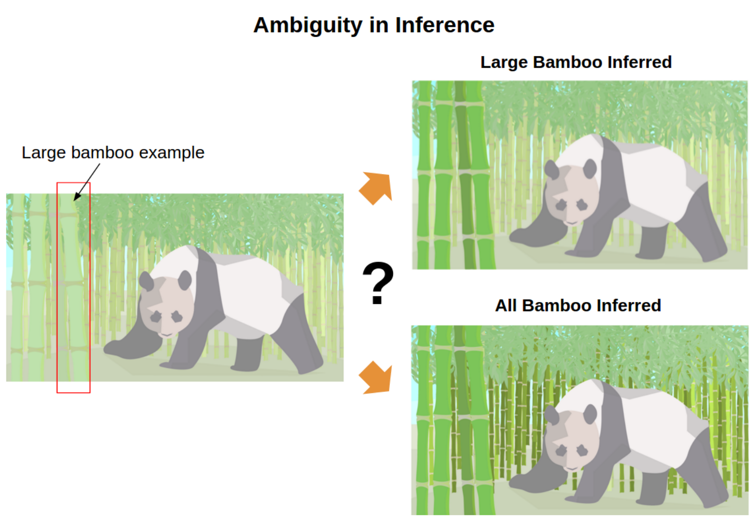



Figure 4-10: A depiction of the ambiguity problem in inference. A large bamboo example could be representative of only large bamboo, or all bamboo. In general, a very large number of interpretations are consistent with the example, so a biasing strategy must be carefully chosen.

The bias that Insight uses was informally determined by examining vector graphics and prior knowledge of visual perception. I noted that typical vector graphics, for each similarity cue, had sparsely-distributed and dense clusters. In the graphic this corresponded to very clear similarity groups, such as green paths versus red paths. This observation indicated that bias should be towards clusters which are *relatively dense*, those that have a large number of paths and small radius relative to a cluster covering all paths. The learning algorithm then tries to find clusters that have high relative density and contain examples for evidence.

The learning algorithm starts with hierarchically clustering the paths for each similarity cue. This is a first step in helping find important relatively dense clusters. Hierarchical clustering reduces the large set of possible clusters to check to a small set of important clusters. For points in a Euclidean space, hierarchical clustering produces a perceptually-organized hierarchy [10]. At higher levels of the hierarchy, clusters contain more points and cover larger spaces. At lower levels, clusters cover smaller spaces but contain fewer points. As levels increase, clusters are combined so that the distances between points in them are minimized, giving rise to the perceptual organization noted above.

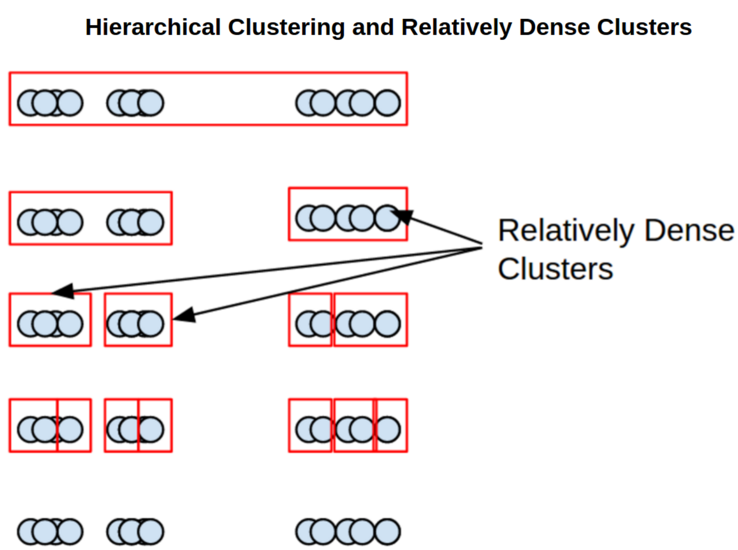



Figure 4-11: A depiction of hierarchical clustering, and the clusters with highest relative density.

While the cluster hierarchy gives us a reduced set of clusters to work with, the learning algorithm must figure out which levels of the hierarchy are appropriate according to the example evidence. This is dependent on both the relative densities of the clusters in the hierarchy, as well as the number of examples in the clusters. Insight scores each of the clusters according to this intuition to help make this decision. The *score* for a cluster $C$ is given by:

$$score(C) = (N(C) - R(C)) \times \left(\frac{2}{3}E(C)\right) \tag{4.6}$$

where $N(C)$ is the relative number of paths in the cluster, $R(C)$ is the relative radius of the cluster, and $E(C)$ is the number of examples in the cluster:

$$N(B) = \frac{\text{(number of paths in cluster)}}{\text{(number of paths in graphic)}} \tag{4.7}$$

$$R(B) = \frac{\text{(radius of cluster)}}{\text{(radius of cluster containing all paths)}} \tag{4.8}$$

$$E(B) = \text{(number of examples in cluster)} \tag{4.9}$$

The scoring function describes the intuition on relative density and example evidence given above. For a high-scoring cluster, we want a larger number of contained paths and lower radius. The linear scaling with each of these terms is reasonable since we do not want to prioritize score for any *change* in radius or number of paths. The terms are normalized according to the global path count and radius of the graphic so that weights for each are not scale-dependent. Lastly, the score grows multiplicatively with the number of examples, to capture the idea that each example provides equivalent evidence for the same cluster. Notably, having many examples will push even clusters with low relative density to high score, manifesting the importance of evidence to learning.

Insight takes the clusters for a given cue to be the disjoint clusters that produce the highest aggregate score. The enforces each example to contribute to only one selected cluster, since disjoint clusters cannot contain the same path by definition. With this choice, if examples aggregately provide good evidence for a cluster it will be selected, while smaller clusters will be selected if the larger clusters' lower relative densities do not compensate the greater evidence. There is an important subtlety here though: targeting aggregate score actually supports dense clusters that are far apart in the hierarchy. This means that selecting both red objects and green objects is considered advantageous to just selecting generally warm-colored objects given red and green evidence. This is where the importance of a flexible representation becomes clear. If a user is giving both red and green examples, the idea that they perceive similarity for these examples specifically should be supported. Their examples should be more important than the general metric for the similarity cue, since that metric as we noted is imperfect.

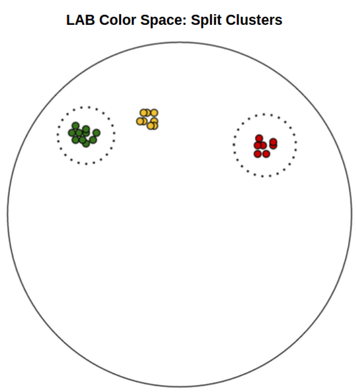



Figure 4-12: A visualization of a case where targeting aggregate score is important. The user has only given red and green examples, so it is more likely they want red and green objects as opposed to red, green, and yellow.

To find the clusters for a cue giving highest aggregate score, Insight climbs the hierarchy from each example, both scoring clusters and keeping track of best-seen clusters. The pseudocode for this procedure is given below:

**Algorithm 1** Maximizing Aggregate Ball Score for Cue

**Require:**

All example-containing clusters have initialized empty dictionary *collect*
All example-containing clusters have initialized example count *excount*
All example-containing clusters have initialized seen example count $seen = 0$
Initialized empty dictionary *best*

```
for all example in examples do
    cluster ← getLowestBall(example)
    while cluster exists and cluster.seen < cluster.excount do
        cluster.seen ← cluster.seen + 1
        if cluster.seen == cluster.excount then
            score ← clusterScore(cluster)
            cluster.collect ← update(cluster.collect, prev_cluster.collect)
            if score >= average(cluster.collect) then
                cluster.collect ← {cluster.id : score}
        if cluster.parent is null then
            best.update ← update(best, cluster.collect)
```

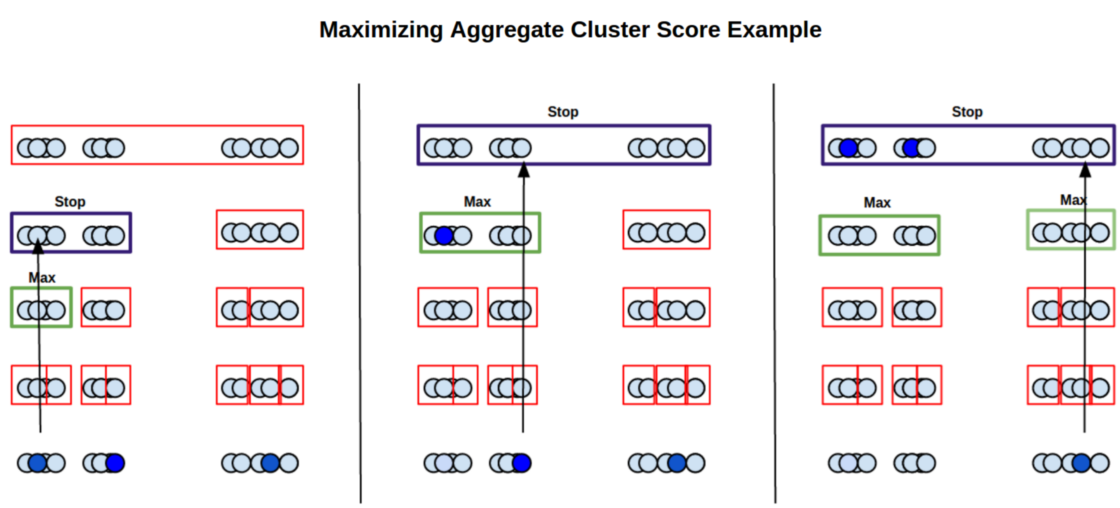



Figure 4-13: An example execution of the climbing algorithm. Examples climb until they see a cluster that has not collected all contained examples (Stop). At each step they track the best cluster they've seen (Max), replacing children clusters as necessary. At the end the best clusters for all examples are aggregated.

Given that this is the bulk of the computation of the algorithm, confirming the efficiency of this procedure is important. The runtime complexity of this procedure is $O(ME \log N)$, where $M$ is the number of cues, $E$ is the number of examples, and $N$

is the number of paths. Both $M$ and $E$ are small in a computational complexity context. In this study $M = 7$ and $E$ empirically is noted to be upper bounded by 50. $N$ can be large, on the order of 10000 in some cases, but since complexity is logarithmic in $N$ this is not significant. The procedure thus maintains real-time application of the learning algorithm.

The highest aggregate scores and corresponding clusters are saved for the purpose of determining relevant similarity cues. As noted above, since the scoring function is invariant to scale of values for a cue, the aggregate scores for each cue can be compared. Insight chooses the relevant cues using these aggregate scores by:

1. Clustering the aggregate scores into 2 clusters using $k$-means, and picking the cues from the cluster with higher mean (note these are clusters for cue scores, not path values for a cue).

2. Thresholding for cues with aggregate score greater than $t_w$.

The first allows for selecting cues that notably stand out, while the second enforces selecting cues which have a high score regardless. The second condition is for robustness against a failure case in the first: one cue gets an incredibly high score because all paths are extremely close. In this case, the one cue will be selected and keep getting even greater score than the rest, since examples supporting old clusters just keep increasing. If someone has given lots of examples for a cluster in a cue, the cue should be selected regardless if other cues are getting increasing scores. The issue is visualized below:

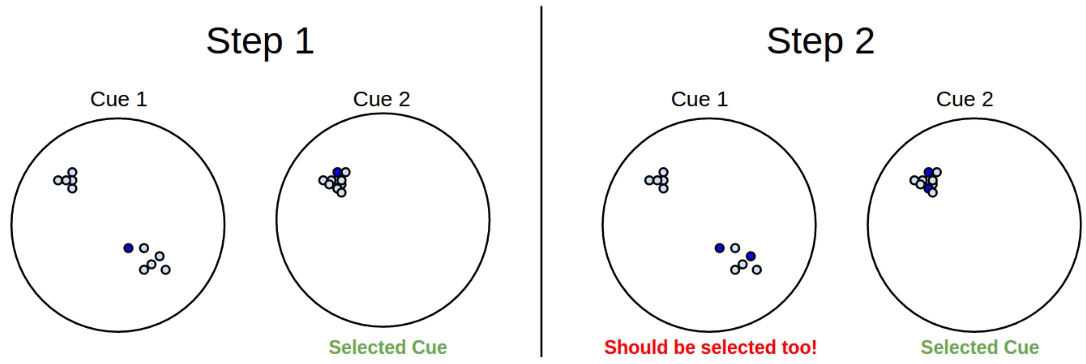



Figure 4-14: A visualization of the scoring issue. If using only 2-mean clustering, the right cue will always be selected while the left will not. The problem is solved by forcing selection of cues with scores past a threshold.

After the relevant cues are learned, all parameters for the selection representation have been computed, and the paths matching the selection are derived. The paths are passed to the interface for visualization and the learning algorithm is completed.

THIS PAGE INTENTIONALLY LEFT BLANK

# Chapter 5

# Evaluation

To evaluate Insight, I developed a set of hypotheses for usage to analyze Insight's performance with respect to the design goals in Chapter 3. To explore these hypotheses, I conducted a user study of people with prior digital design and illustration experience. Participants performed a variety of selection tasks with both Insight and Adobe Illustrator, used as a control because of its role as a predominant commercial vector graphics editor. Participants answered survey questions designed to describe subjective experience with both (see Appendix B, C, D). Their answers and recorded performance during tasks provided the data for exploring hypotheses and understanding Insight's effectiveness.

## 5.1 Hypotheses

1. **Efficiency:** I hypothesized that users generally would be quicker at selecting visually similar paths. Given that Insight biases towards path sets relatively dense in cues, I expected Insight would prove particularly quick for selections in graphics with repetitive elements. I also expected that Insight would be quicker in images with lots of paths or occlusions, since in these cases position-based selection tools would be less helpful.

2. **Expressiveness:** I hypothesized that users would be *able* to make selections

satisfying a wide variety of interpretations of visual similarity, meaning I expected users to be able to select paths similar to multiple degrees for many similarity cue subsets, in a variety of images. I expected that even when the interpretation of similarity did not get high priority from the learning algorithm's bias, Insight would help users get closer to the selection.

3. **Accessibility:** I hypothesized users would find example-based selection refinement easy to use and adopt, and that the choice to use positive examples only would particularly help.

4. **Exploration:** I hypothesized that users would find it easy to try alternative example sequences with Insight by using undo and redo features to decide whether different examples could help them make selections. However, I also hypothesized that when large numbers of user-provided examples resulted in an unsatisfactory selection.

5. **Control:** I expected that users would be able to predict results of inference enough to be able to move towards target selections and that any unexpected results would be minor, enabling users to react effectively. Conversely, I hypothesized that for images with fewer paths, users would prefer to use standard direct manipulation tools rather than rely on Insight's inference.

## 5.2 Participants

For the study, I solicited 14 participants with digital design or illustration experience from a university community. Participants ranged in age from 18 to 45 and were primarily female (85%). All had significant experience with Adobe Illustrator to ensure they were familiar with the selection tools used as a control condition. Most participants used Illustator at least weekly (71%), with the remainder reporting monthly usage. 13 (93%) also had experience with programming or procedural art. I checked this factor to help understand whether users' familiarity with working in abstract representations might affect their usage of tools in the study.

## 5.3 Procedure

The study was conducted using the *selection laptop*: a 2015 Dell XPS 15 with a mouse for performing selections, and the *survey laptop*: a 13-inch 2013 Macbook Air for both visualizing selection task targets and answering survey questions. Both the participant and the experimenter were in the same room, with the experimenter answering questions about procedure and interface, and facilitating tasks such as opening images. Participants performed 10 selection tasks with target selections based on visual similarity. The target selections were chosen to describe a broad diversity of visual similarities, with images that varied in style and use case. Selection tasks were performed both with Adobe Illustrator and a test environment for Insight implemented with Paper.js [7], providing a control condition. For each selection task, participants went through the following steps:

1. The participant viewed the target selection on the survey laptop.

2. The participant attempted to perform the selection in Adobe Illustrator on the selection laptop.

3. The participant attempted to perform the selection in the Insight test environment on the selection laptop.

4. The participant answered a series of survey questions about the task. Surveys contained attitudinal questions relating to the evaluation hypotheses using 5-point Likert scales, 5 being the optimal response. Attitudinal questions included those for ease-of-use, confidence, predictability, and reactability.

To determine when selections were completed, participants were required to yes-no query the experimenter, ensuring that they were responsible for interpreting inference results. Participants could also choose to stop if they felt the task was too difficult, or after 90 seconds into the task. The experimenter aided only in moving between survey pages, opening images, and reminding users about controls (for example the keys which controlled undo, zoom, pan, etc.). At the end of all selection tasks, participants were asked to provide detailed feedback about the entire experience through open-ended survey questions. To help examine all hypotheses, data about selection actions in the test environment were logged, and video of the screen was recorded.

### 5.3.1 Conditions

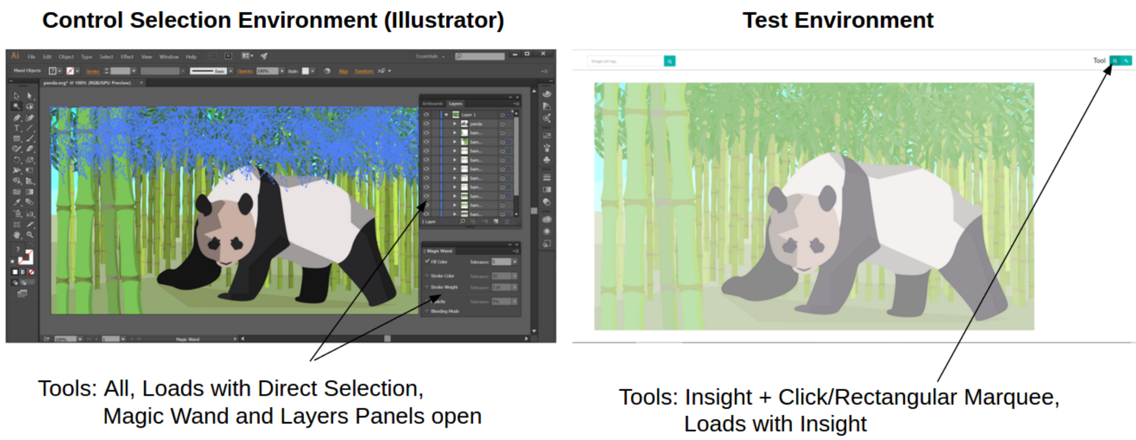



Figure 5-1: Images of both the control interface, Adobe Illustrator, and the Insight test environment. Participants had access from image load to Magic Wand and groups in Illustrator, starting in position-based select. The test environment started in Insight.

Users performed selection tasks in both Adobe Illustrator and a test environment for Insight, providing a standard for comparison. I chose to evaluate Insight in comparison to Adobe Illustrator because it is one of the most predominant commercial vector graphics editors, and was the editor most used by the study participants.

No simplification was made to Adobe Illustrator's interface. Participants had full access to all tools in Adobe Illustrator, including groups and layers defined by users who had created the task images. This choice was made to help understand the importance of working directly in the graphic. If available tools were restricted to click and lasso as in other studies [22, 9, 28], it would be difficult to say whether this was important or not. In addition, this choice allowed users to use all the toolset familiar to them, helping to better gauge Insight's accessibility.

The test environment included just two tools: Insight and a manual tool for click and rectangular marquee selection. The manual tool was included with Insight to better model Insight's expected use in a full vector graphics editor, one which allows for critiquing the selection produced with other tools (see Section 4.1.4). The Insight test environment automatically recorded all user examples and selections for analysis following the study.

### 5.3.2 Selection Tasks

The selection tasks consisted of 10 different target selections, each on a different image. A team of 4 researches including myself selected and evaluated a diversity of composition types and image styles to ensure an expressive range of visual similarity was tested. Some images are shown below in this section, with the rest available in Appendix A.

Selected composition types included static backgrounds like those used in video games, animation assets, and data visualizations. Selected style types included flat and minimalist, sketch-like, and realistic.

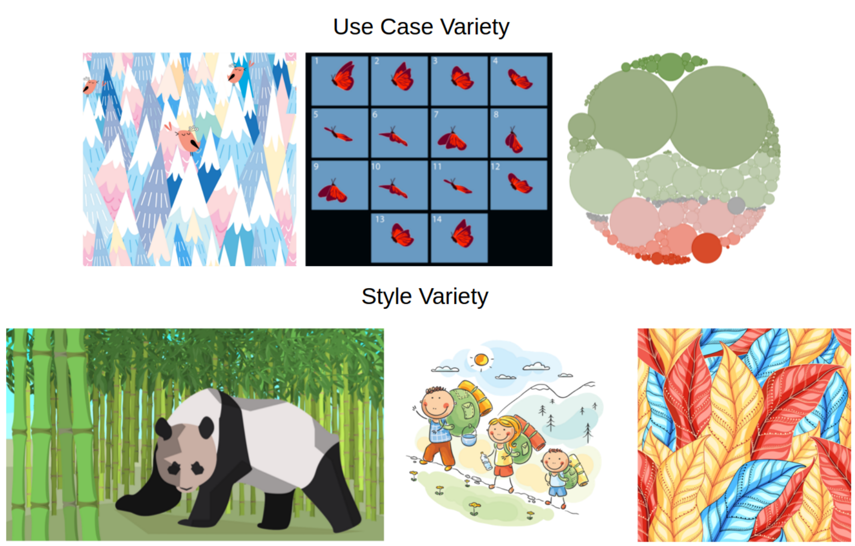



Figure 5-2: Examples of task images from the different composition types and style types.

To capture a range of visual similarity, images were manually examined for sets of similar paths that stood out for editing. Conscious effort was taken to ensure that many different sets of similarity cues were active for the tasks, but selections which I could not understand in terms of Feature Integration Theory were not ignored.


Figure 5-3: Examples of selection tasks. All paths not in the selection have lightness halved.

Inevitably there is some bias in a few people choosing selection tasks. To help account for this issue, I included questions about task relevance in surveys. No participants felt that the tasks were not typical, though some did note that they personally worked on smaller-scale designs.

# Chapter 6

# Results

Overall, participants found Insight to be useful. They noted that it could be helpful when creating their own work (mean: 3.86, std: 0.83), and would particularly find use when editing vectors they did not create themselves (mean: 4.50, std: 0.5). Multiple recognized specific use cases without prompt, and requested it be made available to them in standard editors. The major reasons participants gave for Insight's usefulness directly match with the goals implicit in my research question. Users found that they could *express* new and relevant selections, perform visual similarity selections *efficiently*, and do so in a way that *accessible* to them. In the following sections, I present more detailed evidence for how each of these goals was met.

## 6.1 Expression

I examined users' ability to express selections by looking at the tasks participants were able to complete. Histograms depicting both number of users that completed each task as well as number of tasks completed by each user are shown below.

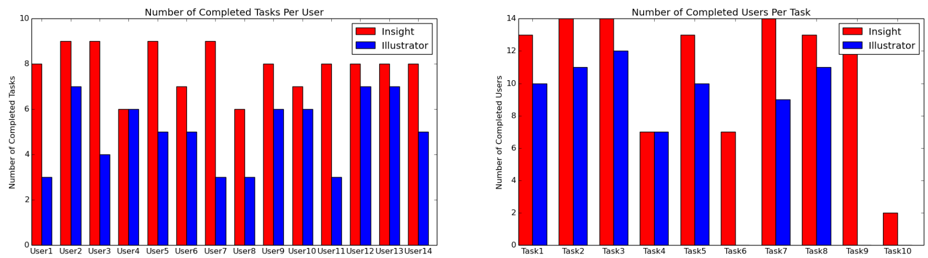



Figure 6-1: Histograms for number of users that completed each task, and number of tasks each user completed.

All users were able to make a majority of the selections in the tasks, often making nearly all (mean: 7.9, std: 0.83). While none were able to make all selections, this was a marked improvement over usage of Illustrator (mean: 5.0, std: 0.16). On an individual level as well, no participant was generally *less* able to make selections with Insight.

The histograms also reveal information about *when* Insight helps express selections. Particularly, we see that for images with repetitive shapes, such as in **Task 2** and **Task 5**, Insight works well. These include tasks which were impossible for users to make with Illustrator, such as **Task 6** and **Task 9**. For images as in **Task 4** Insight does not provide much benefit. The paths have similarity cue values that are dispersed comparatively evenly in the cue spaces, with a few paths each for a wide variety of colors, and a few paths each for many different shapes.

**Task 4:** All paths filling sleeping bags

**Task 5:** Red, orange, yellow hands (no holes)

**Task 9:** All leaves (no thin strands)

Another trend is that participants were better able to complete tasks that should have been doable in Illustrator. **Task 5** is a good example of this: it was possible to use the Magic Wand to make three individual selections and combine them, but users either missed this or found it too error-prone. In general, it appeared that users' expressive ability with Insight was not tied only to it smartly finding similar paths, but due to features of the interaction such as simplicity of example provision.

## 6.2 Efficiency

I took two approaches to measuring Insight's efficiency. First, I compared the amount of time users spent on tasks completed in both Insight and Illustrator. Second, I measured the time users spent on all tasks completed in Insight. The first measurement helps for comparing Insight's interaction paradigm to standard tools', while the second is useful for checking whether Insight is generally efficient as a selection tool. Graphs for both of these measurements are shown below:

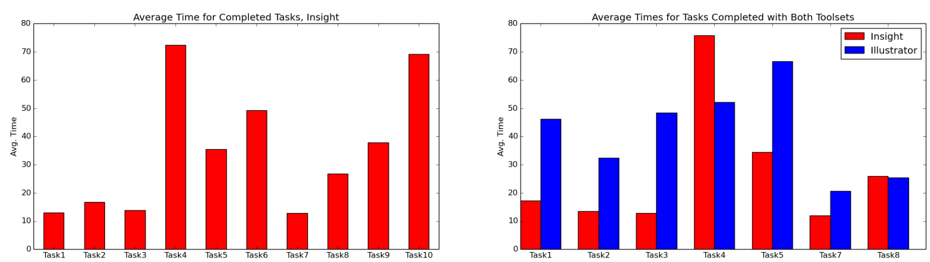
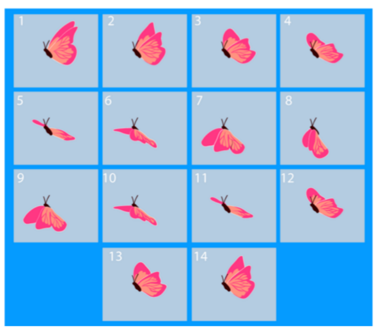


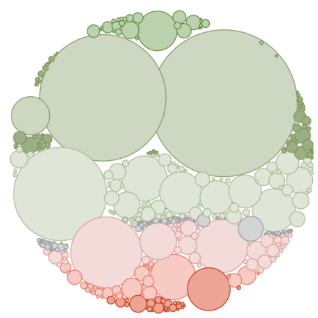

Figure 6-2: Histograms for average time users spent on tasks completed with Insight, and average time users spent on tasks completed with both Insight and Illustrator.

The comparison with Illustrator again highlights the idea that Insight can be more accessible than the standard tools. Despite users having familiarity with tools in Illustrator, with very little learning time they were able to be faster with Insight. I noticed that even when users could use a similar number of actions in Insight and Illustrator, as in **Task 2** and **Task 3**, use of Insight was substantially faster. The only task where this was the exception was **Task 4**. As noted in Section 6.5 above, **Task 4** has paths that are distributed very evenly in the spaces for similarity cues.

While comparisons with Illustrator show that Insight enables faster selection, it is important to see that Insight is still not as efficient as would be expected of selection tools. For some tasks Insight took on average between 10 and 20 seconds, but on others Insight took upwards of 50 seconds. 10-20 seconds may be acceptable for a selection tool, particularly when users have had no prior exposure to it, but I find greater than this to be concerning given the frequency of selection. Multiple participants confirmed this by noting that if they were creating the artwork, they would still try to organize selections with groups and layers for quick reference.



**Task 2:** Butterfly bodies, antennae



**Task 3:** Small green ("medium positive") circles

**Task 4:** All paths filling sleeping bags

## 6.3 Accessibility

Accessibility was measured by Likert ratings for ease-of-use and confidence. In the second survey about overall experience, participants indicated that they both were confident (mean: 4.21, std: 0.58) and found Insight easy to use (mean: 4.29, std: 0.70). Again, we also measured these for each task individually to help understand more specifically how Insight helped:

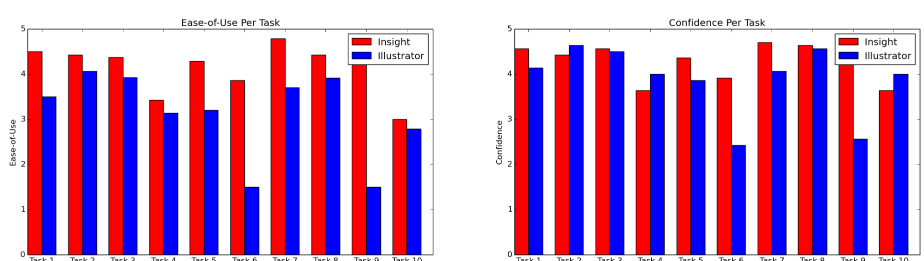



Figure 6-3: Histograms for average ease-of-use and confidence per task.

Examining the histograms, we see that generally participants found Insight and Illustrator similarly accessible, with ease-of-use and confidence for both tending to be high. Given that Illustrator consists of a set of tools adopted into common practice, this indicates that Insight is perceived as accessible to artists and designers as well. The fact that Insight was provided to users with only a few minutes of instruction compared with their months or years of experience only further supports this idea.

The exceptions to these trends are with **Task 6** and **Task 9**, where Insight was noted to be significantly more accessible. These were images both with very large numbers of paths, and similarity representations with cues that Illustrator did not support. Importantly, this is where we see Insight's added levels of expression. With Insight's accessibility in these cases, we see an accomplishment of the goal set in my research question: to provide expressive selection tools accessible to artists and designers.

**Task 6:** Small patterns on red, yellow leaves

**Task 9:** All leaves (no thin strands)

## 6.4 Control

Control was measured by Likert ratings for predictability and reactability. In the second survey about overall experience, participants indicated that they found Insight moderately predictable (mean: 3.50, std: 0.98) and reactable (mean: 3.43, std: 0.90). Again, we also measured these for each task individually to help understand more specifically how Insight helped. Importantly, reactability and predictability for

Illustrator were measured for usage of non-positional tools only, since positional tools are considered to be clear:

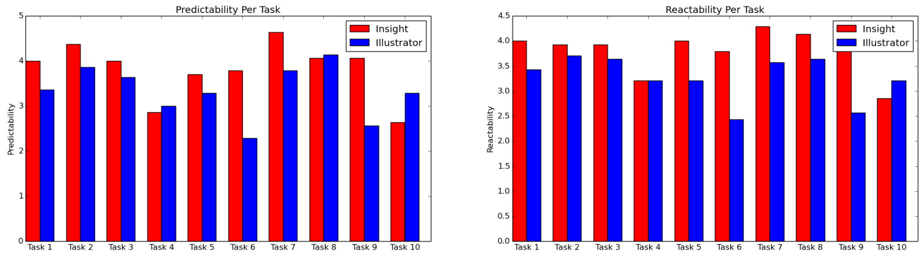



Figure 6-4: Histograms for predictability and reactability of Insight and Illustrator.

Participants overall found Insight and Illustrator to both be similarly reactable and predictable. While neither was extremely reactable or predictable like positional tools are, participants tended to agree that the results of their actions were understandable. This is an interesting result, since in Illustrator all actions are explicit, while Insight *infers* selections.

Insight generally had a very small edge over Illustrator, but substantial differences were visible for **Task 6** and **Task 9**. As noted in results for accessibility (Section 6.3), both images had very large numbers of paths, and similarity respresentations with cues Illustrator did not support. Participants noted that these images were particularly difficult to work with in Illustrator because the group and layer hierarchy was far too complex to examine.

Lastly, just as Insight was found to be less efficient and expressive for **Task 4**, it was found to be less predictable and reactable as well. Participants stated that Insight inferred selections which did not appear similar to them, and changed rapidly throughout the interaction.

**Task 4:** All paths filling sleeping bags

**Task 6:** Small patterns on red, yellow leaves

**Task 9:** All leaves (no thin strands)

## 6.5 Exploration

Exploration was measured by counting the undos used by each participant, and the number of examples used on average for each task. The first gives a straightforward measure of whether users explored alternatives, while the second gives a measure of whether participants explored the model by incremental updates. Histograms for both are shown below:

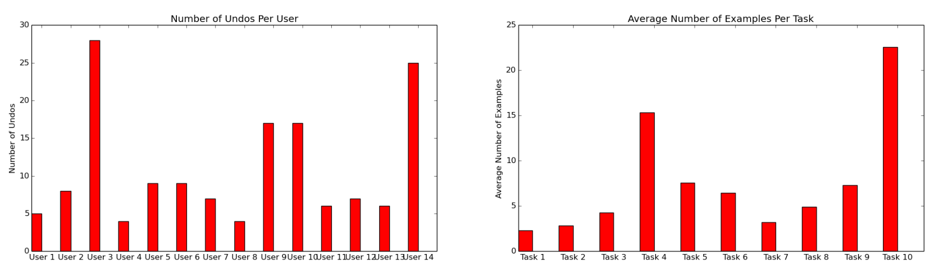



Figure 6-5: Histograms for number of undos used by each user, and average number of examples for each task.

The graphs give a general indication that Insight supports exploration. All users tried to provide different examples using undo, with some using it heavily. I observed that the difference in these participants was generally that those who used undo less instead relied on providing extra examples, while those who used undo more were more careful about picking examples they perceived to be relevant.

Participants appeared comfortable with adding examples in all tasks. I observed that users very rarely restarted usage of Insight, instead preferring to continue adding examples until inference was correct. This does not necessarily indicate that Insight provided small, incremental updates. In fact, participants felt that this was not true in some cases, such as **Task 4** and **Task 10**, where as noted in Section 6.4 inferred selections changed rapidly. However, it does suggest that participants found adding examples to be natural for exploring the visual similarity representation, continuing to add examples even in the cases they found inference poor.

**Task 4:** All paths filling sleeping bags

**Task 10:** All buildings except gray back row, all faces

# Chapter 7

# Discussion

My main goal in this work was to see if interactive machine learning could provide interaction benefits important for selection in graphics. Here I discuss how this general approach, and particular design choices unique to Insight, were important to this goal. During these discussions, I also note how results indicate relevance of interactive machine learning principles for artists and designers.

## 7.1 Inference helps manage complexity in selection

The particular tasks in which Insight was found to be most useful were those in which images had thousands of crowded vector paths. In the corresponding images, the vector paths to select conflicted with other paths in many similarity cues, and the group hierarchies were extremely large. Results show that in all metrics, Insight substantially outperformed Illustrator for these tasks.

I extract from these results that explicit selection tools become difficult to use when working with complex artwork. When the selection representation contains many parameters, and the spaces for those parameters take many values, explicitly specifying all parameters takes a large amount of interaction. In the tasks, when users applied the Magic Wand the large number of parameters became the cues, while with groups and layers finding the exact group index parameter for the desired group became impractical.

With Insight, participants were able to specify complex representations with a simple action.

## 7.2 Artists and designers explore the representation space

The results show that artists and designers both continue providing examples and use undo to explore the selection representation. At any point in the interaction, participants could restart the inference. However, they preferred options for incrementally adjusting the selection representation and trying alternative example sequences. The conclusion I draw from this is that artists and designers *do* value the ability to incrementally adjust their work in inferential selection as well, and find both iterative example-based interaction and undo in Insight helpful for this goal.

5 participants also confirmed the importance of exploration by describing an additional feature they wanted in Insight: the ability to remove examples. In tasks where participants used large numbers of examples, they realized that they may have specified a poor example early on in the interaction, and that adjusting by removing that example would be more helpful than restarting the inference. Critical here is that participants stated they did not feel the need for such a feature until first adding examples with the tool.

## 7.3 Benefits of working directly on the graphic

In Section 6.5 and Section 6.2, I noted that even in tasks where the selection was simple to perform with the tools in Illustrator, participants were slightly more effective and efficient with Insight. I observe that this is a subtle benefit that comes from working entirely in the medium. It appears that moving back-and-forth between the graphic representation and the abstract representation presents a small but important amount of overhead.

It was also important that participants could visualize the selection clearly and directly on the graphic. This was confirmed by issues participants found with the lightness-based selection visualization. Though participants could examine the visualization closely and recognize the selection, they understood it much better when they used the visibility toggle, and so used it extremely frequently. By completely removing the selection from the graphic, the visibility toggle very explicitly made the selection clear.

On the other hand, no participant generally felt that abstract representations were particularly inaccessible except in very complex artwork. Even then, the difficulty they noted was not with the fact that tools used representations outside of the graphic, but with the loss of control they experienced. Participants went to groups for these tasks, but noted that it was impossible to tell how selecting any of those groups from the panel would impact the artwork, since participants had not created them themselves. As opposed to not being able to link an abstract representation to the graphic, participants were concerned that they had not defined the components of that abstract representation. While this acceptance of abstract representations may align with the programming experience participants indicated, the conlusion I draw from these results then is that concreteness is only important to the extent that it is explicit.

## 7.4 Control has primary importance in inferential selection

Over all other factors I measured, participants' feelings of control were most closely tied to accessibility. As noted just above in Section 7.3, participants only found significant difficulty with groups because they had no control over the creator's grouping. Many participants were confident that if they had been constructing the artwork, they would have structured the group hierarchy so that they could perform the tasks. However, this only indicates the importance of control for artists. Realistically, many images (such as in **Task 9**) could not support multiple important selections if groups or layers were used, since paths in them could not share $z$-index. Furthermore, in professional design users must frequently edit work others have provided to them, making it impossible for users to define their own structure. A tool like Insight would be necessary for these cases.

Importance of control was similarly true with Insight's usage. Though participants did not feel that Insight was extremely controllable like positional selection, for most tasks they generally agreed it was predictable and reactable, and were accordingly able to perform the tasks efficiently. Generally, they understood the small changes from Insight's incremental inference. When the model changes were dramatic however, as in **Task 4**, participants felt they had less control, and accordingly both accessibility and efficiency dropped. Our results then showed a clear agreement with the main idea of interactive machine learning approaches: having small, incremental inference updates to a representation is important.

Surprisingly, multiple participants also asked for more complex features to obtain greater control. Contrary to my expectation that simplicity should be prioritized, participants indicated that they would like to have features such as a dialog box for marking explicit similarity cues, and the option to provide negative examples. This confirms the results in other domains using interactive machine learning: prioritize control (see Section 2.2). An interesting direction for future work is then to try providing additional means of control to artists and see how usability changes.

## 7.5 Learning for visual similarity is hard

All tasks contained selections for visually similar paths, but Insight was significantly more effective in task images with repetitive elements and pattern.

The first issue I observe is with the representation of visual similarity. There are cases where Euclidean distances for similarity cue representations do not closely resemble perceptual similarity. A good example of this is with **Task 10**, where participants selected buildings with multiple faces. Participants recognized faces as being the same shape because they processed perspective, while Insight does not take perspective into account. A substantial improvement to be made would be in developing better metrics for individual cues.

A second important issue is with Insight's biasing strategy. The biasing strategy is generally one of Insight's strengths. I noted that traditional machine learning algorithms were not helpful for the case of Insight since the generalizations they made with few examples were fairly arbitrary, making the selections visualized seem uncontrollable. By biasing towards relatively dense clusters, Insight helped users access repetitive elements quickly, a generally important indication of visual similarity. However, as seen in **Task 4**, this works poorly in images with paths distributed sporadically in each similarity cue. A possible adjustment for the learning algorithm that could help would be to weight examples more heavily. This would cause less relatively dense clusters to be selected more quickly so long as users were repeatedly providing evidence.

A final possibility for learning visual similarity is to *learn the representation itself.* With tools such as deep learning, it may be possible to directly feed a number of real-world visually similar path sets and obtain a classifier that understands what visual similarity is without prior modeling. The same could be true of selections in vector graphics in general. It would be interesting to pursue this direction, though certainly collection of the required amount of selection data would be challenging.

# Chapter 8

# Conclusion

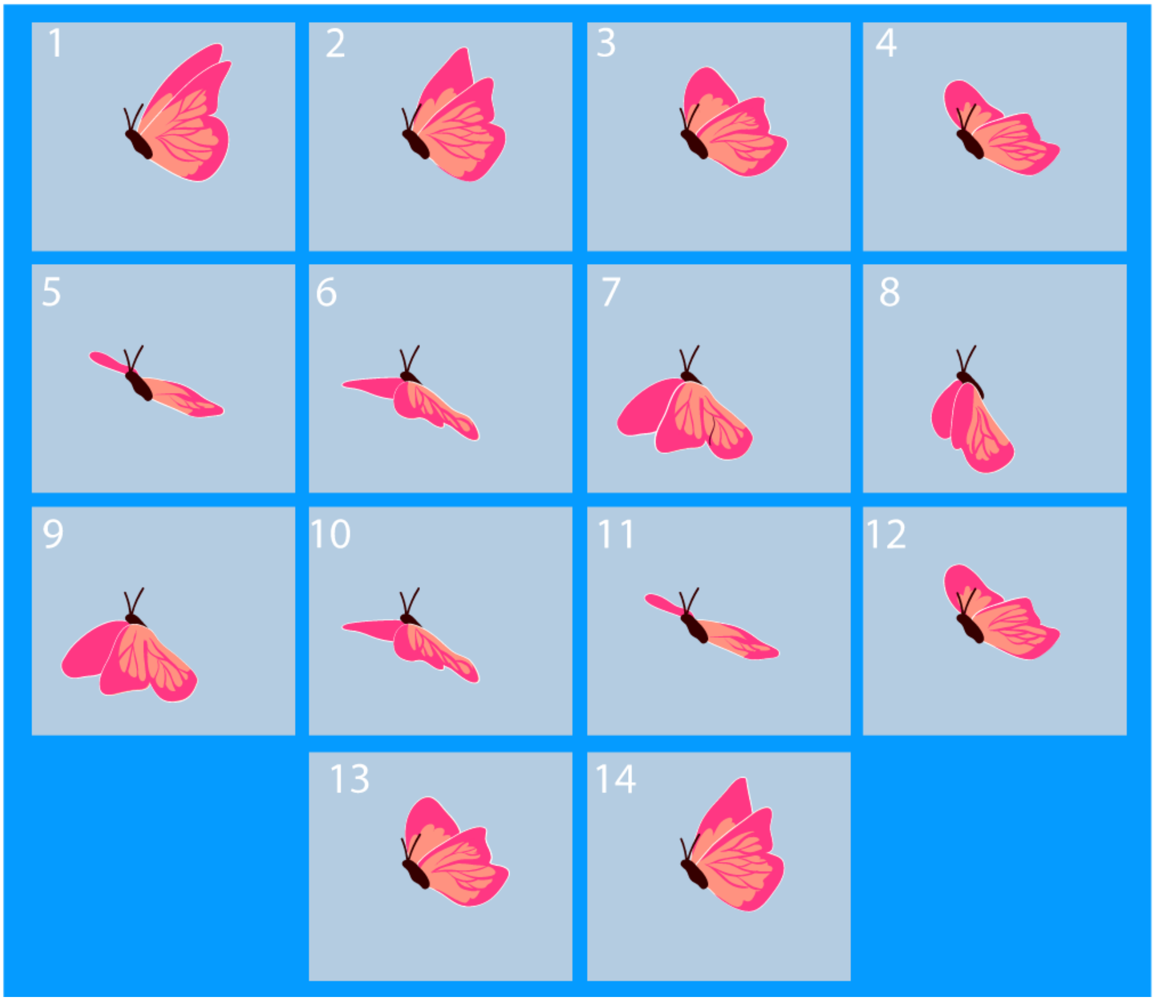
Through my development and evaluation of Insight, I show a way to use interactive machine learning to enable artists and designers to easily perform complex selections in vector graphics. My results indicate that many of the qualities artists and designers have been noted to value in their direct manipulation tools are true for graphics selection as well, and that Insight's interaction set is a means of providing them.

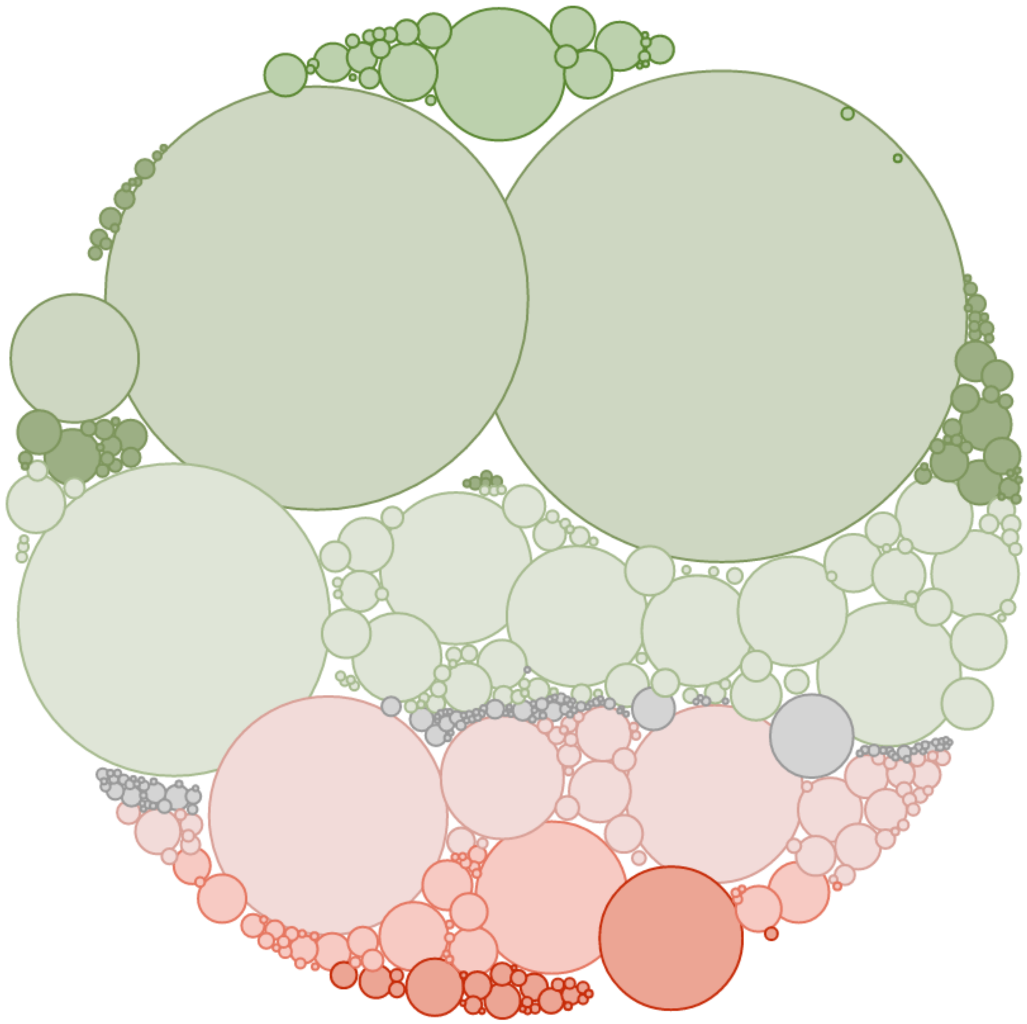
However, Insight is just a starting point in a broad space of possibilities. Interactive machine learning could be explored for other important selection models or even complex editing operations. It is an interaction paradigm which might be used to enable artists and designers to produce a much broader variety of creative outcomes. Studying how these tools can affect the creative process is a wonderful goal for future research.

THIS PAGE INTENTIONALLY LEFT BLANK

# Appendix A

# Selection Tasks


1
2
3
4
5
6
7
8
9
10
11
12
13
14




FOOD
ZOO

# Appendix B

# Demographic Survey

Demographics

What is your age?

- 18 to 24
- 25 to 34
- 35 to 44
- 45 to 54
- 55 to 64
- 65 to 74
- 75 or older

What is your gender?

- Female
- Male

What is your profession?

Please rate your experience with vector graphics editors on a scale of 1-5, with 1 being none and 5 being expert.

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| Adobe Illustrator | | | | | |
| Inkscape | | | | | |
| Omnigraffle | | | | | |
| CorelDRAW | | | | | |

How frequently do you use the following vector graphics editors?

| | Daily | Weekly | Monthly | Never |
|---|---|---|---|---|
| Adobe Illustrator | | | | |
| Inkscape | | | | |
| Omnigraffle | | | | |
| CorelDRAW | | | | |

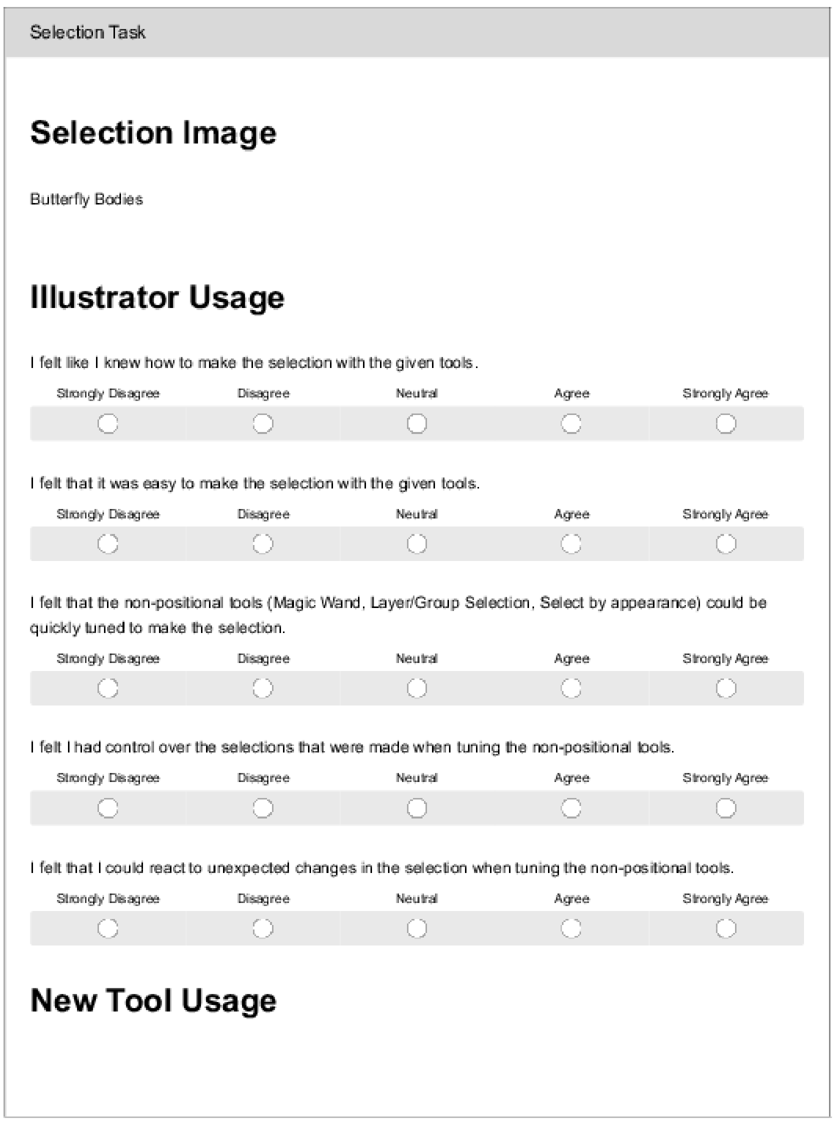
Do you have experience with programming or making procedural art?

◯ Yes ◯ No

THIS PAGE INTENTIONALLY LEFT BLANK

# Appendix C

# Selection Task Survey


Selection Task

## Selection Image

Butterfly Bodies

## Illustrator Usage

I felt like I knew how to make the selection with the given tools.

| Strongly Disagree | Disagree | Neutral | Agree | Strongly Agree |
|---|---|---|---|---|
| ○ | ○ | ○ | ○ | ○ |

I felt that it was easy to make the selection with the given tools.

| Strongly Disagree | Disagree | Neutral | Agree | Strongly Agree |
|---|---|---|---|---|
| ○ | ○ | ○ | ○ | ○ |

I felt that the non-positional tools (Magic Wand, Layer/Group Selection, Select by appearance) could be quickly tuned to make the selection.

| Strongly Disagree | Disagree | Neutral | Agree | Strongly Agree |
|---|---|---|---|---|
| ○ | ○ | ○ | ○ | ○ |

I felt I had control over the selections that were made when tuning the non-positional tools.

| Strongly Disagree | Disagree | Neutral | Agree | Strongly Agree |
|---|---|---|---|---|
| ○ | ○ | ○ | ○ | ○ |

I felt that I could react to unexpected changes in the selection when tuning the non-positional tools.

| Strongly Disagree | Disagree | Neutral | Agree | Strongly Agree |
|---|---|---|---|---|
| ○ | ○ | ○ | ○ | ○ |

## New Tool Usage

# Appendix D

# Overall Feedback Survey

## Overall Experience

**Please describe your overall experience using the new selection tool, using your experience with Illustrator as a baseline.**

1. I felt like I knew how to make the selection with the given tools.

| Strongly Disagree | Disagree | Neutral | Agree | Strongly Agree |
|---|---|---|---|---|
| ○ | ○ | ○ | ○ | ○ |

2. I felt that it was easy to make the selection with the given tools.

| Strongly Disagree | Disagree | Neutral | Agree | Strongly Agree |
|---|---|---|---|---|
| ○ | ○ | ○ | ○ | ○ |

3. I felt that the inference tool understood the selection with few examples.

| Strongly Disagree | Disagree | Neutral | Agree | Strongly Agree |
|---|---|---|---|---|
| ○ | ○ | ○ | ○ | ○ |

4. I felt that I had control over the generalizations the inference tool made.

| Strongly Disagree | Disagree | Neutral | Agree | Strongly Agree |
|---|---|---|---|---|
| ○ | ○ | ○ | ○ | ○ |

5. I felt that I could react to unexpected generalizations the inference tool made.

| Strongly Disagree | Disagree | Neutral | Agree | Strongly Agree |
|---|---|---|---|---|
| ○ | ○ | ○ | ○ | ○ |

6. Do your answers to the previous questions vary significantly between different images? If so, please explain.

7. Did you feel like the images were organized (layers, groups) in a way similar to how you would have organized them?

| Strongly Disagree | Disagree | Neutral | Agree | Strongly Agree |
|---|---|---|---|---|
| ○ | ○ | ○ | ○ | ○ |

8. Did difference from expected organization make some images difficult to work with? Do you feel like your expected organization would have helped? If so, please explain.

9. I feel like I would use this tool to edit artwork others have created.

| Strongly Disagree | Disagree | Neutral | Agree | Strongly Agree |
|---|---|---|---|---|
| ○ | ○ | ○ | ○ | ○ |

10. I feel like I would use this tool when I am creating artwork.

| Strongly Disagree | Disagree | Neutral | Agree | Strongly Agree |
|---|---|---|---|---|
| ○ | ○ | ○ | ○ | ○ |

11. Do you feel like your answers to the previous two questions are specific to particular subsets of vector artwork? If so, please explain.

12. How do you feel the new selection tool compares to other selection tools? Does it make other forms of selection unnecessary, or can you achieve similar results with existing tools? Does it compliment existing selection tools, or is it redundant? Please explain.

13. Are there any other comments you have that you'd like to share?

THIS PAGE INTENTIONALLY LEFT BLANK

# Bibliography

[1] Saleema Amershi, Maya Cakmak, W Bradley Knox, and Todd Kulesza. Power to the people: The role of humans in interactive machine learning. American Association for Artificial Intelligence, 2014.

[2] James M Cortese and Brian P Dyre. Perceptual similarity of shapes generated from fourier descriptors. *Journal of Experimental Psychology: Human Perception and Performance*, 22(1):133, 1996.

[3] Jerry Alan Fails and Dan R Olsen Jr. Interactive machine learning. In *Proceedings of the 8th international conference on Intelligent user interfaces*, pages 39–45. ACM, 2003.

[4] James Fogarty, Desney Tan, Ashish Kapoor, and Simon Winder. Cueflik: interactive concept learning in image search. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 29–38. ACM, 2008.

[5] Rubaiat Habib Kazi, Fanny Chevalier, Tovi Grossman, and George Fitzmaurice. Kitty: Sketching dynamic and interactive illustrations. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*, UIST '14, New York, NY, USA, 2014. ACM.

[6] Todd Kulesza, Simone Stumpf, Margaret Burnett, and Irwin Kwan. Tell me more?: the effects of mental model soundness on personalizing an intelligent agent. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 1–10. ACM, 2012.

[7] J. Lehni and J. Puckey. Paper.js, 2011.

[8] Z. Lieberman, T. Watson, and A. Castro. openframeworks, 2015.

[9] David Lindlbauer, Michael Haller, Mark Hancock, Stacey D Scott, and Wolfgang Stuerzlinger. Perceptual grouping: selection assistance for digital sketching. In *Proceedings of the 2013 ACM international conference on Interactive tabletops and surfaces*, pages 51–60. ACM, 2013.

[10] Christopher D Manning, Prabhakar Raghavan, Hinrich Schütze, et al. *Introduction to information retrieval*, volume 1. Cambridge university press Cambridge, 2008.

[11] M. McCullough. *Abstracting Craft: The Practiced Digital Hand.* The MIT Press, Cambridge, Massachusetts, 1996.

[12] Robert C Miller and Brad A Myers. Multiple selections in smart text editing. In *Proceedings of the 7th international conference on Intelligent user interfaces*, pages 103–110. ACM, 2002.

[13] Liangliang Nan, Andrei Sharf, Ke Xie, Tien-Tsin Wong, Oliver Deussen, Daniel Cohen-Or, and Baoquan Chen. *Conjoining gestalt rules for abstraction of architectural drawings*, volume 30. ACM, 2011.

[14] C. Reas and B. Fry. Processing, 2004.

[15] Mitchel Resnick, Brad Myers, Kumiyo Nakakoji, Ben Shneiderman, Randy Pausch, Ted Selker, and Mike Eisenberg. Design principles for tools to support creative thinking. 2005.

[16] Alan Ritter and Sumit Basu. Learning to generalize for complex selection tasks. In *Proceedings of the 14th international conference on Intelligent user interfaces*, pages 167–176. ACM, 2009.

[17] Erich Rome. *Simulating perceptual clustering by gestalt principles.* Citeseer, 2001.

[18] Stephanie L Rosenthal and Anind K Dey. Towards maximizing the accuracy of human-labeled sensor data. In *Proceedings of the 15th international conference on Intelligent user interfaces*, pages 259–268. ACM, 2010.

[19] Toby Schachman. Alternative programming interfaces for alternative programmers. In *Proceedings of the ACM international symposium on New ideas, new paradigms, and reflections on programming and software*, pages 1–10. ACM, 2012.

[20] Donald A Schon. Designing as reflective conversation with the materials of a design situation. *Research in Engineering Design*, 3(3):131–147, 1992.

[21] Ben Shneiderman. The future of interactive systems and the emergence of direct manipulation. In *Proc. Of the NYU Symposium on User Interfaces on Human Factors and Interactive Computer Systems*, pages 1–28, Norwood, NJ, USA, 1984. Ablex Publishing Corp.

[22] Sara L Su, Sylvain Paris, and Frédo Durand. Quickselect: History-based selection expansion. In *Proceedings of Graphics Interface 2009*, pages 215–221. Canadian Information Processing Society, 2009.

[23] Ivan E. Sutherland. Seminal graphics. chapter Sketchpad: a Man-machine Graphical Communication System, pages 391–408. ACM, New York, NY, USA, 1998.

[24] Anne Treisman. Perceptual grouping and attention in visual search for features and for objects. *Journal of Experimental Psychology: Human Perception and Performance*, 8(2):194, 1982.

[25] Anne Treisman. Features and objects in visual processing. *Scientific American*, 255(5):114–125, 1986.

[26] Anne M Treisman and Garry Gelade. A feature-integration theory of attention. *Cognitive psychology*, 12(1):97–136, 1980.

[27] Sherry Turkle and Seymour Papert. Epistemological pluralism: Styles and voices within the computer culture. *Signs*, 16(1):128–157, 1990.

[28] Pengfei Xu, Hongbo Fu, Oscar Kin-Chung Au, and Chiew-Lan Tai. Lazy selection: A scribble-based tool for smart shape elements selection. *ACM Transactions on Graphics (TOG)*, 31(6):142, 2012.