Pattern-Driven Navigation in 2D Multiscale Visual Spaces with Scalable Insets

Fritz Lekschas[1], Michael Behrisch[1], Benjamin Bach[3], Peter Kerpedjiev[2], Nils Gehlenborg[2] and Hanspeter Pfister[1]

[1]Harvard University
[2]Harvard Medical School
[3]University of Edinburgh

April 4, 2018

Abstract

We present *Scalable Insets*, a technique for interactively exploring and navigating large numbers of annotated patterns in multiscale visual spaces such as gigapixel images, matrices, or maps. Exploration of many but sparsely-distributed patterns in multiscale visual spaces is challenging as visual representations change across zoom levels, context and navigational cues get lost upon zooming, and navigation is time consuming. Our technique visualizes annotated patterns too small to be identifiable at certain zoom levels using insets, i.e., magnified thumbnail views of the annotated pattern. Insets support users in searching, comparing, and contextualizing patterns, while reducing the amount of navigation needed. They are dynamically placed either within the viewport or along the boundary of the viewport to offer a compromise between locality and context preservation. Annotated patterns are interactively clustered by location and type. They are visually represented as an aggregated inset to provide scalable exploration within a single viewport. A controlled user study with 18 participants found improved performance in visual search (up to 45% faster) and comparison of pattern types (up to 32 percentage points more accurate) compared to a baseline technique. A second study with 6 experts in the field of genomics showed that Scalable Insets are easy to learn and effective in a biological data exploration scenario.

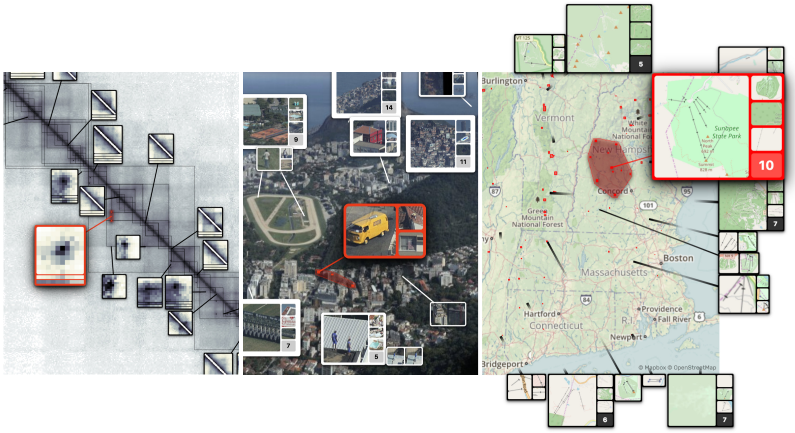

Figure 1: Scalable Insets applied to a genome interaction matrix [47], gigapixel image [8], and geographic map from Mapbox [37] and OpenStreetMaps [42] (left to right). Various annotated patterns are highlighted in red.

1 Introduction

Many large data sets, such as gigapixel images, geographic maps, or networks, require exploration of annotations at different levels of detail. We call an annotation a region in the visualization that contains some visual patterns (called *annotated pattern* henceforth) as seen in Figure 1. These annotated patterns can either be generated by users or derived computationally. However, annotated patterns are often magnitudes smaller than the overview and too small to be visible. This makes tasks such as exploration, searching, comparing, or contextualizing challenging, as considerable navigation is needed to overcome the lack of overview or detail.

Exploring annotated patterns in context is often needed to assess the relevance of patterns and to dissect important from unimportant regions. For example, computational biologists study thousands of small patterns in large genome interaction matrices [9] to understand which physical interactions between regions on the genome are the driving factor that defines the D structure of the genome. In astronomy, researchers are exploring and comparing multiple heterogeneous galaxies and stars within super high-resolution imagery [44]. In either case, inspecting every potentially important region in detail is simply not feasible.

Exploring visual details of these annotated patterns in multiscale visual spaces requires a tradeoff between several conflicting criteria. Annotated patterns must be visible for inspection and comparison (DETAIL). Enough of the overview needs to be visible to provide context for the patterns (CONTEXT). And any detailed representation of an annotated pattern must be close to its actual position in the overview (LOCALITY). Current interactive navigation and visualization approaches, such as focus+context, overview+detail, or general highlighting techniques (section 2), address some but not all of these criteria and become difficult as repeated viewport changes, multiple manual lenses, or separate views at different zoom levels are required, which stress the user's mental capacities.

In this paper, we describe *Scalable Insets*—a scalable visualization and interaction technique for interactively exploring and navigating large numbers of annotated patterns in multiscale visual spaces. Scalable Insets support users in early exploration through multifocus guidance by dynamically placing magnified thumbnails of annotated patterns as insets within the viewport (Figure 1). The design of Scalable Insets is informed by interviews with genomics experts, who are engaged in exploring thousands of patterns in genome interaction matrices. To keep the number of insets stable as users navigate, we developed a dynamic grouping and interactive placement of insets within the viewport, clustering patterns based on their location, type, and the user's viewport (subsection 4.3). The degree of clustering constitutes a tradeoff between CONTEXT and DETAIL. Groups of patterns are visually represented as a single aggregated inset to accommodate for DETAIL. Details of aggregated patterns are gradually resolved as the user navigates into certain regions. We also present two dynamic mechanisms (subsection 4.2) for placing insets either within the overview (Figure 1 left and center) or on the overview's boundary (Figure 1 right) to allow flexible adjustment to LOCALITY. With Scalable Insets, the user can rapidly search, compare, and contextualize large pattern spaces in multiscale visualizations.

We implemented Scalable Insets as an extension to HiGlass [29], a flexible web application for viewing large tile-based datasets. The implementation currently supports gigapixel images, geographic maps, and genome interaction matrices. We present two usage scenarios for gigapixel images and geographic maps (subsection 3.1) to demonstrate the functionality of Scalable Insets. Feedback from a qualitative user study with six computational biologists who explored genome interaction matrices using Scalable Insets shows that our technique is easy to learn and effective in biological data exploration. Scalable Insets simplify the interpretation of computational results through identification and comparison of patterns in context and across zoom levels. Results of a controlled user study with 18 novice users comparing both placement mechanisms of Scalable Insets to a standard highlighting technique show that Scalable Insets reduced the time to find annotated patterns by up to 45% at identical accuracy and improved the accuracy in comparing pattern types by up to 32 percentage points.

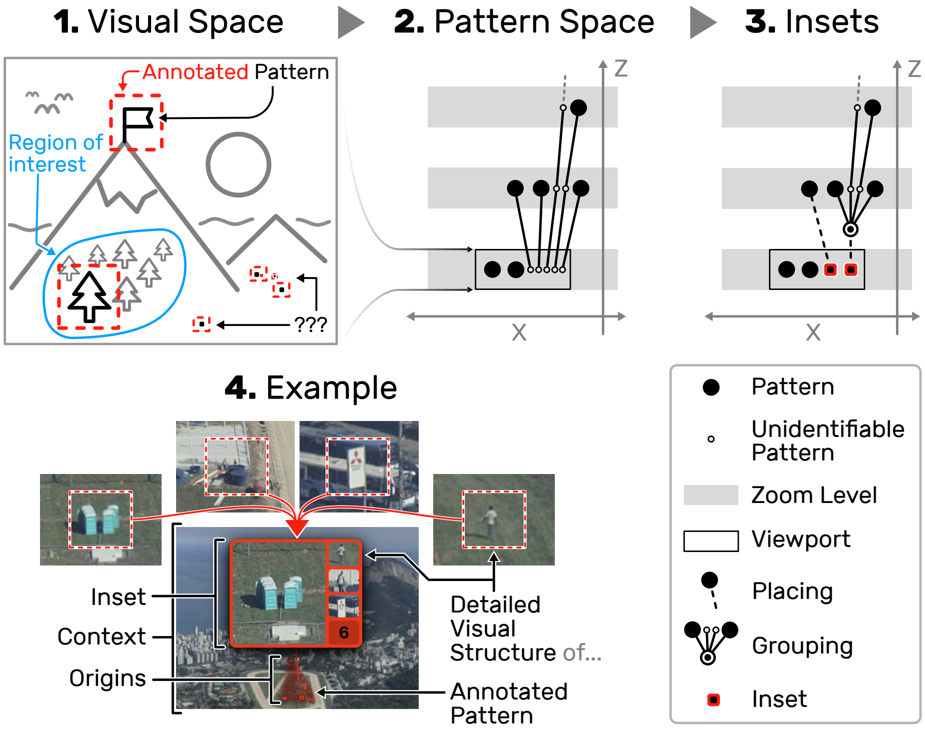

Figure 2: The core idea of the Scalable Inset. (1) A multiscale visual space with several annotated patterns, some of which are too small to be identifiable (indicated by "???"). (2) The virtual pattern space illustrated as a space-scale diagram, illustrating that some patterns are only identifiable at certain zoom levels. (3) To provide guidance, small patterns are placed as magnified insets into the current viewport. Scalability is ensured by dynamically grouping small patterns in close proximity and representing them as an aggregate as shown in (4).

2 Related Work

In the following sections we review various techniques that focus on certain aspects of navigation in large-scale visualizations and highlight challenges for multifocus exploration of many small patterns.

Navigation Pan and zoom [18] techniques are common for navigating large multiscale visual spaces. Although these techniques are often applied they can require a large amount of mental effort as either context or details are lost and navigating to distant regions can be time consuming [17]. Other techniques take advantage of the underlying data structure to simplify navigation. For example, Link Sliding and Bring & Go developed by Moscovich et al. [41] implement navigation along an underlying network and provide context to navigational options through direct visualization of this network on demand. Similarly, Dynamic Insets [19] utilizes an underlying graph topology to dynamically place off-screen annotations as insets at the boundary of the view and thereby provides guidance on where to navigate next. However, many data sets do not provide such graph structures or navigation along links is not desirable.

Highlighting Highlighting the presence of details is commonly used to alleviate the lack of navigational cues [26]. For reviews on general highlighting techniques see [21, 31, 35, 48, 54]. Irrespective of the navigation technique or goal, knowing details about the outcome of an interaction upfront can prevent spending unnecessary time on interacting with an information space. For instance, Scented Widgets [57] embed visual cues and simple visualizations into user interface elements. This can shorten the interaction time given that some details about the outcome of the interaction are known upfront. Ip and Varshney [23] describe a salience-based technique for guiding users in gigapixel images. They utilize color to highlight regions of high salience. While this is very effective to provide visual cues, these cues do not enable the user to get an understanding of the detailed visual structure of the highlighted regions without having to manipulate the viewport. Scalable Insets draws on these ideas to display the details of annotated patterns across scales.

Aggregation and Simplification Scalability for large visualizations can be achieved through aggregation and simplification of sets of elements. For instance, ZAME [12] is a matrix visualization technique that presents a visual summary of multiple cells at a higher level. The summary is composed of an embedded visualization showing the distribution of aggregated cell values. Milo et al. [40] describe network motifs, which are repetitive network patterns, to facilitate a more concise view of large networks through visual simplification [10]. Van de Elzen and van Wijk [55] integrate the ideas of ZAME and Network Simplification into a very concise overview of large networks where nodes are aggregated into small statistical summary diagrams. This works well to gain an overview but the detailed visual structure of patterns is lost, which applies to many semantic zoom interfaces.

Overview+Detail To address the problem of lost context or missing details [26] one can juxtapose multiple panels at different zoom levels. The separation between these panels provides flexibility but comes at the cost of divided user attention [39]. For example, in Poly-Zoom [25] different zoom levels are organized hierarchically and appear as separate panels, which limits the number of regions that the user can simultaneously focus on. TrailMap [58] shows thumbnails of previously visited locations in a map visualization to support rapid bookmarking and time traveling through the exploration history. Hereby, separate panels work well as the user has already seen each location in detail before it appears as a thumbnail, but it is not clear how efficient such an approach would be for guidance to new locations. HiPiler [33] supports exploration and comparisons of many patterns through interactive small multiples of many small patterns in a separated view. While this approach works well for out-of-context comparison, it has not been designed for guided navigation within the viewport of a multiscale visualization.

Focus+context Focus+context techniques show details in-place, while maintaining a continuous relationship to the context, often via distortion. The most common type of these techniques are lenses that can be moved and parameterized by the user (see [53] for a comprehensive review). For example, Table Lens [46] utilizes a degree of interest [17] approach to enlarge or shrink cells within a spreadsheet visualization. Similarly, Rubber Sheet [49] and JellyLens [45] are distortion-based techniques that enlarge areas of interest in a visualization. Mélange [13] takes a different approach by folding unfocused space into a 3rd dimension like paper. Unlike our method, these techniques benefit from maintaining a continuous view at the expense of direct context around a focus region and limiting the ability to support simultaneous focus on many regions.

Detail-In-Context Hybrid approaches magnify annotated patterns as insets within the image but offside their original location [5]. For example, Pad [43] and Pad++ [2] were one of the first tools to visualize details through magnified insets. Detail Lenses [27] put emphasis on several regions as insets that are arranged along the inner border of a map visualization. The insets are only loosely connected to their origin through sequential ordering and sparse leader lines. This ensures that the center of the map visualization is not occluded. This works well for up to a dozen insets without dynamic exploration. DragMag [56] extended the concept of Pad into a hybrid approach where magnified insets can either be placed within the image or on the outside border. Our new technique complements efforts in this area. In particular, we build upon the idea of DragMag [56] and extend it to multifocus scenarios that require automated grouping and placement to scale beyond a handful of insets.

3 Scalable Insets: Overview

The design of Scalable Insets is driven by the three functional requirements for detail-in-context methods. It provides a dynamic tradeoff for DETAIL, CONTEXT and LOCALITY to overcome the issues one would run into with naive approaches (Figure 3). In the following we give an overview of the technique and demonstrate the visualization and guidance aspects. Technical details on how we achieve this tradeoff are given in section 4.

In a multiscale visual space (Figure 2.1) that contains several *annotated patterns* (henceforth just called *patterns*), not all patterns will be identifiable at every zoom level. To this end, the notion of identifiability can be described as patterns being fully contained in the viewport and having a minimal output resolution, e.g., 24 × 24 pixels. Whenever a pattern is identifiable we are able to perceive its *detailed visual structure* (Figure 2.4). This setup lets us imagine a virtual pattern space (Figure 2.2), which describes when a pattern is visible or how much zooming is needed to identify its detailed visual structure. To reduce the interaction and navigation time to assess these visual details of patterns, we extract thumbnails of unidentifiable patterns at the zoom level that renders the pattern identifiable and place these thumbnails within the current viewport as insets. The number of displayed insets can be limited by clustering several closely-located patterns into a group (Figure 2.3) and representing this group as a single inset (Figure 2.4). Together with a dynamic placement strategy that avoids occlusion of insets and patterns, this enables Scalable Insets to provide guidance for high numbers of patterns while reducing navigation time.

3.1 Usage Scenarios

The two following usage scenarios, on a gigapixel image and geographic map application, depict typical exploration tasks, such as searching, comparing, and contextualizing patterns, and focus likewise on overview and detail. A third, more complex exploration scenario on genome interaction matrices with computational biologists is presented in subsection 6.2.

Exploring Gigapixel Images We show a photograph of Rio de Janeiro [8, 52] with a resolution of 454330 × 149278 pixels (Figure 4.1) in which local users have annotated 924 patterns, including birds, people, or cars. Some

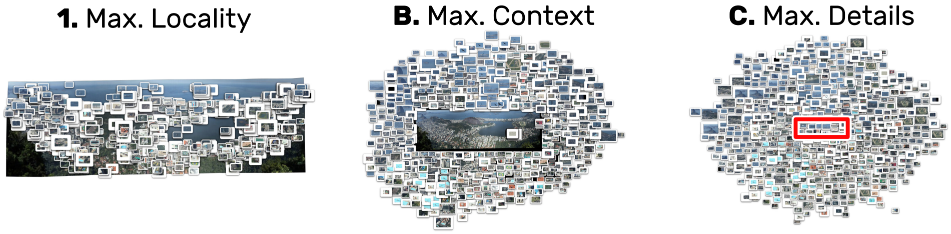

Figure 3: Three approaches exemplifying naive optimization of (1) Locality, (2) Context, and (3) Detail only. The red rectangle in (C) indicates the size of the occluded image for reference.

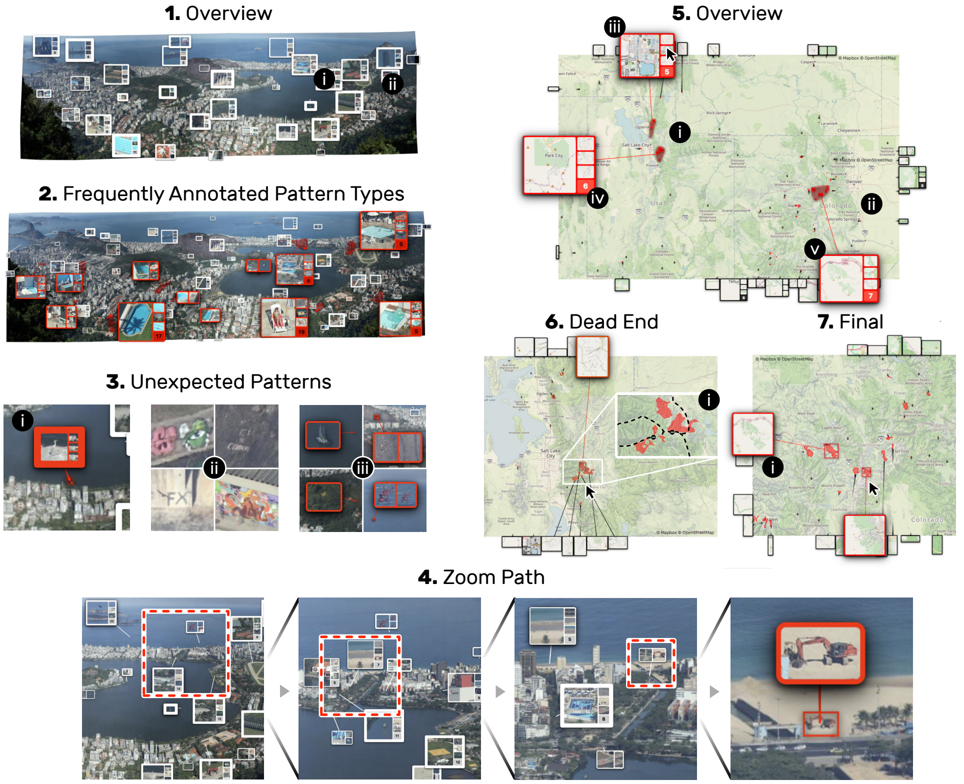

Figure 4: Demonstration of the Scalable Insets approach on a gigapixel image of Rio de Janeiro [8] by *The Rio de Janeiro - Hong Kong Connection* [52] and ski areas around Utah and Colorado shown on a map from Mapbox [37] and OpenStreetMap [42]. The screenshots illustrate how Scalable Insets enables pattern-driven exploration and navigation at scale; details are explained in subsection 3.1.

of these patterns are close together, e.g., on the same street, while others are isolated in the sea. However, most of them are not identifiable without considerate pan and zoom. We follow a hypothetical journalist who is writing an article about unseen aspects of Rio de Janeiro, which requires finding, comparing, and localizing the annotated patterns to assess "Which neighborhoods are particularly interesting to viewers?", "What kind of patterns are most frequently annotated?", and "Are there any unexpected patterns given their location?".

As shown in Figure 4.1, Scalable Insets places insets for annotated patterns too small to be identifiable within the viewport. Given their high number, Scalable Insets clusters many patterns that are in close proximity to each other, into groups, and presents these groups as aggregated insets. In this example, the number of insets ranges between 25-50 depending on the region being explored, which provides a good tradeoff between the DETAIL and CONTEXT criteria. The size of the insets ranges from 32 to 64 pixels (for the longer side) to provide a notion of depth and the popularity of patterns (determined by view statistics) is mapped onto the border size such that thicker borders indicate higher popularity.

The journalist starts by examining the entire picture to gain an overview. At a first glance, Scalable Insets reveals a relatively equal distribution of annotated patterns, with higher frequency in areas of man-made constructions (Figure 4.1) The journalist immediately finds popular pattern types as they are represented by the large image in an inset. The journalist notices a relatively high frequency of annotated swimming pools (Figure 4.2), which are scattered across the entire image but appear to be more frequently found in the left part of the image. Upon hovering over an inset, its origin is highlighted through small red dots and a red hull in the overview, which localizes the inset. Via a click, insets are scaled up and more details of the represented patterns are seen. This enables the journalist to quickly identify a bird (Figure 4.3i) sitting on a locally known rock and to find interesting street art (Figure 4.3ii), some of which are located in areas that are not easily accessible by foot. Also, with Scalable Insets, several patterns located in monotone regions (Figure 4.3iii), such as the sea, are explorable with minimal pan and zoom. As the journalist inspects a specific region more closely, aggregated insets of groups of patterns gradually split into smaller and smaller groups (Figure 4.4), presenting more details while maintaining a relatively fixed number of insets.

Exploring Geographic Maps Our second scenario involves the exploration of ski areas in a geographic map from Mapbox [37] and OpenStreetMaps [42]. We obtain an estimation of the location of the world's ski areas by analyzing *aerialway* annotations [30] of the raw data from OpenStreetMap as these annotations describe potential paths of ski lifts, slopes, and other aerial ways.

The user sets out to find, compare, and localize *interesting* ski areas. In this scenario, interest is defined by the size of individual ski areas and the size of multiple ski areas within close proximity. Localization of ski areas is important to determine the accessibility and proximity by car between multiple closely-located ski areas. This time, insets are shown *outside* the map to provide full access of information in the map, such as streets, cities, mountains and other important geographical information needed for localization.

The user starts exploring around Utah and Colorado (Figure 4.E). The map shows two regions with several closely-located ski areas nearby Salt Lake City (Figure 4.5i) and Denver (Figure 4.5ii). Upon scaling up an inset, the user can explore size and shape of up to four representative ski areas among a group. This allows for fast comparison of the ski areas without the need to navigate. For example, the user compares three promising groups of several ski areas (Figure 4.5iii, 5iv, and 5v). Through hovering over different images in an aggregated inset, shown in Figure 4.5iii, the user identifies that this group

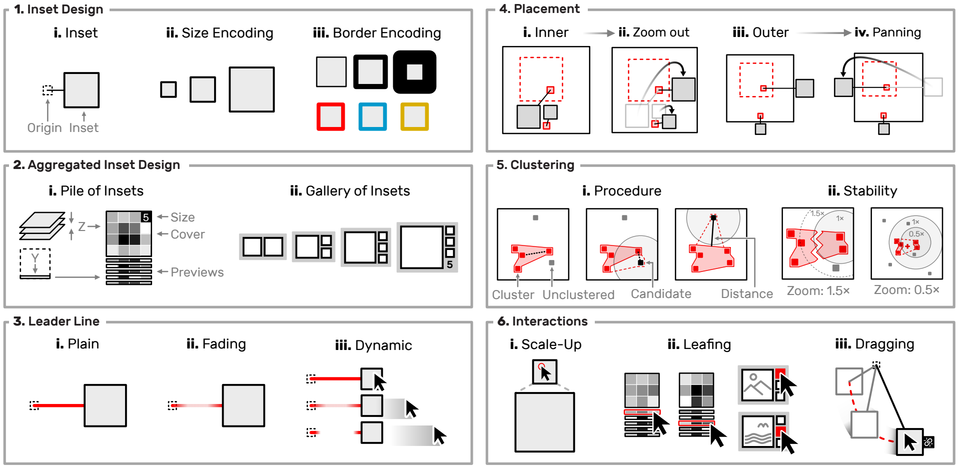

Figure 5: Schematic design principals of Scalable Insets. (1) Inset design and information encoding. (2) Visual representation of aggregated insets. (3) Leader line styles. (4) The inset placement mechanism and stability considerations of Scalable Insets. (5) Aggregation procedure and stability considerations. (6) Interaction between insets applied in Scalable Insets.

contains only small ski areas as well as an outlier, i.e., an annotated region that does not correspond to a ski area. While the second group (Figure 4.5iv) indeed represents several large ski areas, close inspection (Figure 4.6) reveals that the road connecting these ski areas is closed during winter (Figure 4.2Fi). Zooming out again, the user finds a suitable region with several larger ski areas (Figure 4.7) that are conveniently accessible by car through the Interstate 70 nearby Vail (Figure 4.7i).

4 Scalable Insets: Technique

4.1 Inset and Leader Line Design

Insets are small, rectangular thumbnails of annotated patterns at increased magnification. The level of magnification is defined by the display size (in pixels) of the insets and zoom level. The display size of insets can depend on a continuous value, e.g., a confidence score or range between a user-defined minimum and maximum, but is invariant to the zoom level to give more control over DETAIL and CONTEXT. This comes at the cost of reduced awareness of the *depth* of annotated patterns, which we consider less important for Scalable Insets as it does not directly support finding, comparing, and contextualizing patterns. The thickness of the border can be used to encode an additional information and the hue of the border is adjustable to reflect different categories of annotated patterns. Both encodings are illustrated in Figure 5.1ii and Figure 5.1iii.

A leader line is drawn between insets and their origin in order to establish a mental connection as their positions may not coincide (subsection 4.2). We designed three different leader line styles. Plain and fading leader lines are static and only differ in their alpha values along the line (Figure 5.3i and Figure 5.3ii). Dynamic leader lines adjust their opacity depending on the distance of the mouse cursor to an inset or origin. Fading and dynamic leader lines minimize clutter in the event of leader line crossing to preserve context and have been shown to maintain a notion of connectedness [7, 34]. We chose to limit Scalable Insets to straight leader lines for simplicity and performance reasons. While other techniques such as edge bundling [22] or boundary labeling [3] exist, the benefits are expected to be minimal. For inner-placement leader lines are usually very short, since insets are positioned as close as possible to their origin. Barth et al. [1] have shown that straight leader lines for boundary labeling, which is similar to our outer-placement (subsection 4.2), perform comparably to or even better than more sophisticated methods.

4.2 Inset Placement

We have developed an algorithm for positioning insets either within the viewport (inner-placement) or at the boundary of the viewport (outer-placement) based on simulated annealing. We chose simulated annealing for its simplicity, potential to produce high quality layouts [6], and generality [11]. Other methods can produce near-optimal layouts [27] but are not suitable for interactive adjustments in real time.

Our goal for placing insets is to maximize context preservation and locality. For both placement strategies, our algorithm optimizes three primary requirements. First, insets should not overlap with each other. At the same time, insets should be placed as close to their origin as possible. And finally, leader line crossing should be minimized. For the inner-placement, insets inevitably occlude some parts of the context. Therefore, our algorithm additionally tries to avoid occlusion of annotated patterns. Figure 5.4i and Figure 5.4iii illustrate both inner- and outer-placement strategies.

The inner-placement of insets is stable upon panning, i.e., the relative distance between insets and their origin is fixed. However, since the display size of insets is invariant to the zoom level, the algorithm is re-evaluating and potentially re-positioning insets upon zooming. Following all requirements strictly could lead to drastic changes in the positioning even upon subtle navigation. For example, in Figure 5.4ii a subtle zoom out would lead to occlusion of other annotated patterns (indicated as dashed, red boundaries), which would force the insets to *jump* to the next best position. Similarly, in Figure 5.4iv subtle panning to the left would change the closest available position on the boundary and make the inset flip from one side to the other. Since these phenomena would significantly harm the usability of Scalable Insets, we employ three additional requirements to stabilize placement across zoom levels. Position changes of insets are penalized to avoid marginal improvements to the placement, which would otherwise lead to unnecessary movements. Also, the Euclidean distance between an old and a new inset position is limited to avoid large changes. Finally, in the outer-placement approach, insets should not be moved to the opposing side, even if the leader line is getting long.

4.3 Aggregation

To provide scalability beyond a handful of annotated patterns and preserve CONTEXT, we developed a density-based dynamic clustering algorithm that assigns every annotated pattern in the viewport to a particular group, called a cluster. Each each cluster is represented as a single visual entity, called an aggregated inset. Our clustering approach is based on the spatial distance between annotated patterns in the viewport. Starting with a randomly chosen pattern we find the closest cluster. Only if the distance between the selected pattern and the bounding box of the cluster is closer than a user-defined threshold and the area of the cluster combined with the selected pattern is smaller than a user-defined threshold do we assign the pattern to the given cluster (Figure 5.5i). Otherwise, we create a new cluster composed of the single, selected pattern. Upon adding patterns to clusters, we keep a sorted list of the nearest neighbors for each newly added pattern, which will help us to determine breakpoints upon zoom-out as explained next. Similar to the placement of insets, the clustering of pattern is re-evaluated upon navigation. To improve cluster stability between short, repeated zoom changes, the distance threshold, for deciding whether an inset should be assigned to a particular cluster, is dynamically adjusted as illustrated in Figure 5.5ii. During zoom-in, clusters remain unchanged until the distance between the farthest nearest neighbors is larger than 1.5 times the distance threshold. During zoom-out, clusters and insets will not be merged until their distance is less than half the distance threshold. This limits the changes to cluster composition upon navigation to facilitate the user's mental map of the pattern space. We chose to design a relatively simple clustering approach to ensure high performance in terms of rendering time. Our approach is inspired by DBSCAN [14] but we do not implement recursive scanning of nearby patterns as we strive for a spatially-equal partitioning rather than continuous clusters.

We designed two approaches to visually represent clusters. When exploring large matrices, clusters of annotated patterns are aggregated into piles and feature a 2D cover matrix together with 1D previews. The cover matrix represents a statistical summary of all the patterns on the cluster, e.g., the arithmetic mean or variance. 1D previews are averages across the Y dimension and provide a visual cue into the variability of patterns on the pile. To avoid arbitrary large piles in terms of their display size, the maximum number of previews is limited to a user-defined threshold. We employ KMeans clustering when the number of patterns on a pile is larger than the threshold and only represent 1D previews for the average patterns of each group. Clusters of patterns from photographic images and geographic maps are aggregated into a gallery of cluster representatives. A small number indicates the amount of patterns for clusters larger than four. Drawing on insights from work in exploration of the parameter, design, or ideation space [32, 38, 50, 51], which suggests that diverse examples of groups are more efficient for representation than homogeneous examples, we chose a diverse set of patterns as the representatives. The largest pattern in the gallery is representing the most important pattern, according to a user-defined metric. The next representative is the pattern closest to the clusters centroid in Euclidean space. The last two representative patterns correspond to the two farthermost patterns in terms of their Euclidean distance. The pile-based aggregation is useful for patterns with well-alignable shapes, e.g., rectangles, lines, or points, and provides a concise representation of the average pattern and pattern variance. The gallery aggregation is preferable for patterns of diverse shapes as an average across shape boundaries does not provide meaningful insights.

4.4 Inset Interaction

Scalable Insets add a minimal set of interaction to insets and is otherwise agnostic in terms of the navigation technique. Upon moving the mouse cursor over an inset the hue of its border and leader line are changed and a hull is drawn around the location of the annotated patterns represented by the inset. Upon scale-up (Figure 5.6i), it is possible to leaf through the 1D previews of a pile or the representatives of a gallery (Figure 5.6ii). Insets are draggable (Figure 5.6iii) to allow manual placement and uncover potentially occluded scenery in the overview (CONTEXT). Dragging disconnects insets from the locality criterion to avoid immediate re-positioning to the inset's previous position upon zooming. The disconnect is visualized with a small glyph indicating a broken link (Figure 5.6iii) and can be reversed through a click on this glyph.

5 Implementation

To demonstrate the utility of Scalable Insets, we built a web-based prototype for gigapixel images, geographic maps, and genome interaction matrices. Scalable Insets are implemented as an extension to HiGlass [29], an open-source web-based viewer for large tile-based data sets. The Scalable Insets extension to HiGlass is implemented in JavaScript and integrates into the React [15] application framework. D3 [4] is used primarily for data mapping and matrices are rendered on canvas with PixiJS [36]. Scalable Insets

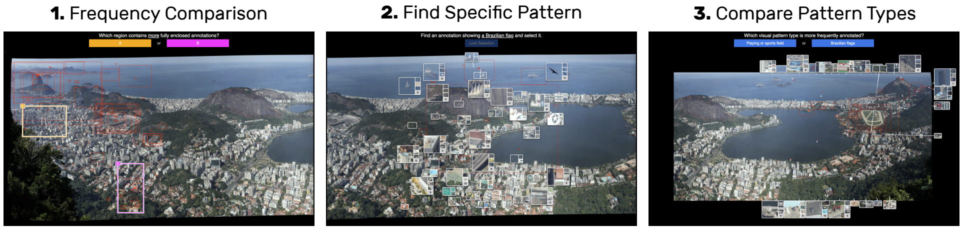
1. Frequency Comparison
Which region contains more fully enclosed annotations?
A
or
B
A
2. Find Specific Pattern
Find an annotation showing a Brazilian flag and select it.
Lock Selection
3. Compare Pattern Types
Which visual pattern type is more frequently annotated?
Playing or sports field
or
Brazilian flags

Figure 6: Interface of the software, showing Rio de Janeiro [52], used in the first user study and example views for each task and technique. (1) Comparing the frequency of annotated patterns in two distinct regions with BBox. (2) Finding a specific pattern that shows a Brazilian flag with Inside. (3) Comparing global frequency of patterns showing either "player or sports field" with "Brazilian flag" with Outside.

utilizes HTML and CSS for positioning and styling insets and leader lines and implements flexible customization of nearly every parameter via a JSON configuration file. The server-site application, which extracts, aggregates, and serves the images for insets, is implemented in Python and built on top of Django [16]. Both, the front-end and back-end applications, are open-source, and freely available at `https://github.com/hms-dbmi/higlass`.

6 Evaluation

We conducted two user studies (one quantiative, one qualitative) to assess whether people can quickly learn how to navigate and interact with Scalable Insets and if Scalable Insets increases performance (task completion-time, task accuracy) in exploring annotated patterns in large multiscale visualizations.

6.1 Study 1: Quantitative Evaluation

The goal of the first study was to compare the performance of Scalable Insets to traditional highlighting of the annotated patterns using boundary boxes and to assess the effects of context preservation when insets are located within the viewport and locality when insets are placed on the boundary of the viewport.

Techniques—We compared the following three techniques and their configurations, illustrated in Figure 8.

BBox: Annotated patterns are highlighted with boundary boxes.

Inside: Annotated patterns are highlighted with Scalable Insets placed *inside* the image viewport and mildly-translucent boundary boxes.

Outside: Annotated patterns are highlighted with Scalable Insets placed *outside* the image and mildly-translucent boundary boxes.

Data—For the study, we chose seven photographic gigapixel pictures[1] from Gigapan [20] showing different cities (e.g., Figure 6). We used two pictures for practice trials and used the remaining five for the final test trials. The annotated patterns in this study represent user-defined annotations from Gigapan. Image sizes ranged from $100,643 \times 43,935$ pixels to $454,330 \times 149,278$ and the number of patterns ranged from 82 to 924.

Tasks—We defined the following three tasks for which we measured task-completion time (in seconds) and task accuracy (percentage of correct answers).

- **REGION**: *Which region contains more fully enclosed annotations; A or B?* Participants had to chose between the two regions labeled A and B and click on either to confirm their answer. This task aimed to test if Scalable Insets cause clutter and if the aggregation prevents people from identifying dense regions.
- **PATTERN**: *Find an annotation showing* <pattern> *and select it*. The tag *<pattern>* was replaced with a description of a manually chosen pattern type that appears up to a few times in the image, e.g., "a helicopter landing field". The goal of this task is to test our criteria of DETAIL, i.e., whether showing patterns in detail is beneficial.
- **TYPE**: *Which visual pattern type appears more frequently; A or B?* Where *A* and *B* were replaced with two generic pattern types appearing multiple times but not equally often, e.g., "swimming pools" and "construction sites". Again, the participants had to choose between two options, which were collected via a click on either of two buttons. This task aimed at testing our criteria of LOCALITY.

Hypotheses—We formulated one hypothesis per task:

H1 For REGION, there will be no difference in time and accuracy for any of the techniques as the detailed visual structure of annotated patterns is not important for REGION.

H2 For the PATTERN task, INSIDE will be faster than OUTSIDE and OUTSIDE will be faster than BBOX. We expect the inner-placement of insets to be most efficient as the detailed visual structure of annotated patterns is displayed spatially close to their original location, i.e., eye movement is minimized. We expect the outer-placement of insets to be slightly slower compared to INSIDE, due to eye movement, but faster than the baseline technique as it still shows the detailed visual structure of annotated patterns.

H3 For the TYPE task, INSIDE and OUTSIDE will be faster than BBOX. We expect Scalable Insets with both placement mechanisms to perform equally fast and better than BBOX as they both highlight the detailed visual structure of annotated patterns.

[1] Gigapan IDs: 149705, 40280, 48635, 47373, 33411, 72802, and 66020

Participants—We recruited 18 participants (7 female and 11 male) from Harvard University. The majority (13) of participants were aged between 25 and 30. Three participants were aged between 18 and 25 and two between 31 and 35. The participants volunteered, had no visual impairments, and were not required to have any particular computer skills. Each participant received a $15 gift card upon completion of the experiment.

Study Design—We used the following full-factorial within-subject study design with Latin square-randomized order of the techniques. Participants were split evenly between the three technique groups. Task order was kept constant across all participants and conditions. To avoid learning effects between images, the set of annotated patterns were split into three groups that covered an equally-sized region of the image. The order of these regions was kept constant, i.e., the first technique always operated on the first quadrant of the images. To avoid memory effects between REGION and PATTERN we excluded the patterns that we asked for in PATTERN from the trials on REGION. Furthermore, these specific patterns were chosen such that they are not contained in or in close proximity to the two regions that were compared in REGION. Finally, each trial is repeated five times on different images. Since we used real-world annotations the difficulty between images might differ. We ordered the images by size and amount of annotated patterns. The first image contains the fewest annotations and is the smallest in size. The annotation frequency, size, and structural difficulty increases with the last image being the largest and most frequently annotated. The order of the images was kept the same.

Setup—The study was conducted on a MacBook Pro with a 2.7 GHz quad-core processor and 16 GB RAM running MacOS X Sierra. The laptop featured a 15” monitor with 2880 × 1800 pixels at an effective resolution of 1440 × 900 pixels, and was equipped with a standard two-button mouse that featured a scroll wheel.

Procedure—The study was conducted in individual sessions. Upon arrival, participants were informed about the general procedure of the experiment and consent was collected. We first introduced the data used in the study and briefly (2 minutes) demonstrated the functionality of the application interface. The participants then started a study software that guided them through each task. At the beginning, gender and age group was collected and participants had to solve a 12-image Ishihara color blindness test [24]. Prior to the actual test trials of each task, participants were shown detailed instructions and they had to complete two practice trials to familiarize themselves with the user interface and the respective task. The first out of five timed trials was started upon clicking on the start button located in the center of the screen. Once the user selected an answer the trial timer stopped and a click on the next button was awaited before starting the next trial. Participants were instructed to finish the trials *as fast as possible and as accurately as possible* but to also *rely on their intuition* when estimating frequencies (REGION and TYPE). At the end, participants anonymously filled out a questionnaire on general improvements and rated insets approach in regard to its general impression, usefulness, and usability on a 5-point Likert scale.

Results For each participant we collected measures from 810 timed trials, resulting in a total number of 14,580 trials. We found that completion time was not normally distributed after testing goodness-of-fit of a fitted normal distribution using a Shapiro-Wilk test. To remove outliers we visually inspected dot plots with individual data points and decided to remove trials that are more than 4 standard deviations away from the arithmetic mean time. These trials are most likely related to severe distraction. This resulted in the removal of 4 trials from REGION, 1 trial from PATTERN, and 1 trial from TYPE. Given the non-normal distribution of completion time and the unequal number of trials due to outlier removal, we report on non-parametric Kruskal-Wallis and Mann-Whitney U tests for assessing significance. For errors, we used a Chi-square test of independence. All p-values are reported for $\alpha = 0.05$. In the following, we report on time (in seconds) and accuracy (in per) by task, summarized in Figure 7.

Results for REGION—A pairwise post-hoc analysis revealed significant difference for completion time between BBOX-INSIDE ($p = .0114$) and BBOX-OUTSIDE ($p = .0015$) but not for INSIDE-OUTSIDE. The respective mean times are BBOX=13.4s (SD=14.2), INSIDE=14.1s (SD=10.4), OUTSIDE=14.2s (SD=8.25). This constitutes an approximate speedup of 5.0% for BBOX over INSIDE and 5.6% for BBOX over OUTSIDE. These results let us reject H1 as BBOX was fastest. Given that in absolute numbers INSIDE and OUTSIDE are only 0.7s and 0.8s slower than BBOX suggests that overhead imposed by insets is fairly small and likely diminishes upon performance improvements to the current implementation of Scalable Insets as discussed later.

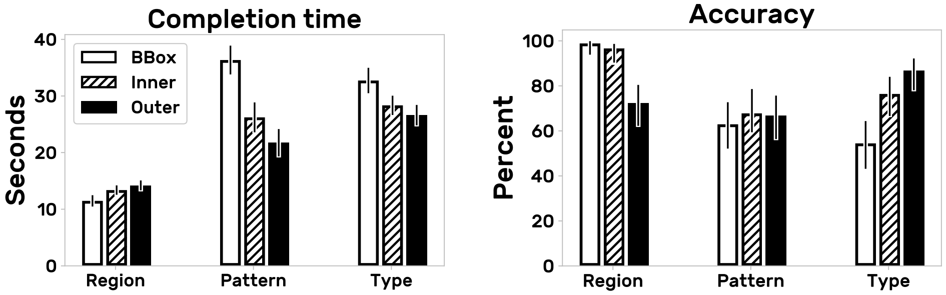

Figure 7: Mean completion time in seconds (lower is better) and mean accuracy in percent (higher is better) across tasks and techniques. Error bars indicate the 95% confidence intervals. Note, due to non-normal distribution of completion time we report significance on the median using Kruskal-Wallis and Mann-Whitney U tests.

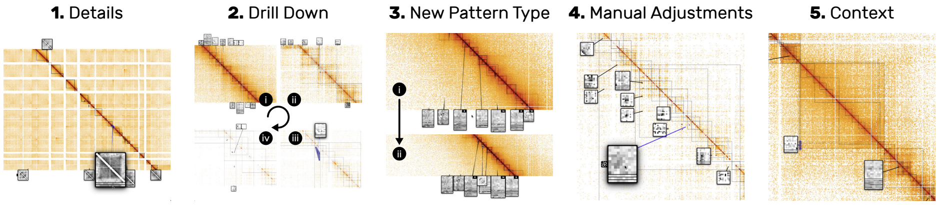

Figure 8: Exploratory notes from the second user study. (1) Detailed inspection of an unexpected pattern through scale-up and leafing. (2) Example drill-down into the original location of a clustered pile of insets until they disperse. (3) Upon zoom-out a new pattern type was displayed as an inset (see little arrow) and immediately recognized. (4) Manual inspection of the context around the pile's original location (end of blue line). (5) Focus on a pile of two insets due to their location (corner of the larger rectangle).

For accuracy, we found significant increase for BBOX (27 %-points) and INSIDE (24 %-points) over OUTSIDE ($\chi^2(1, N = 177) = 30.4, p < .0001$ and $\chi^2(1, N = 179) = 22.7, p < .0001$). We believe that this difference might be caused by misinterpretation or confusion of leader line stubs as they might have appeared like or occluded the bounding box around annotated patterns.

Results for PATTERN—We found significant differences for completion time between BBOX-INSIDE ($p = .0453$) and BBOX-OUTSIDE ($p = .0006$). The mean times were BBOX=39.4 (SD=42.5), INSIDE=26.2 (SD=25.1), OUTSIDE=21.8 (SD=22.9). This amounts for an approximate 44.9% speed up for OUTSIDE over BBOX and 33% speedup for INSIDE over BBOX and shows high potential for search task when the detailed visual structure of patterns and their location is important. While we could not confirm a superiority of INSIDE over OUTSIDE, we find a clear improvement of Scalable Insets over the baseline technique BBOX. We can thus partly accept H2. The speedup is very strong indicator that the core principal of Scalable Insets is efficient for pattern search. While the difference between INSIDE and OUTSIDE is not significant we are still surprised by the much stronger speedup for OUTSIDE. A possible explanation is that the alignment of insets in the outer-placement mechanism (OUTSIDE) is potentially beneficial for fast sequential scanning. This advantage might alleviate if contextual properties are included in the search task as well, e.g., find an annotated car located on an intersection, which we did not explicitly test for. For accuracy, we did not find any significant difference between the techniques.

Results for TYPE— We found only marginal significant ($p = .0892$) differences for completion time between BBOX-OUTSIDE. The mean times were BBOX=32.7 (SD=21.3), INSIDE=28.3 (SD=16.1), OUTSIDE=26.6 (SD=17.2). Although only marginally significant, we recognize an approximate 18.7% speedup for OUTSIDE over BBOX on completion time. These results let us accept hypothesis H3. For accuracy, our results show pairwise significant differences between BBOX-INSIDE ($\chi^2(1, N = 179) = 9.66, p = .0019$) and BBOX-OUTSIDE ($\chi^2(1, N = 180) = 23.5, p < .0001$). This describes an approximate improvement of 22 percentage points for INSIDE over BBOX and an approximate improvement of 32 percentage points for OUTSIDE over BBOX. While the completion time alone is not conclusive, the results for accuracy indicate that OUTSIDE and INSIDE provide a much better understanding of the distribution of pattern types. Participants with BBOX were only marginally significantly slower but made a lot more mistakes.

Qualitative feedback—In the closing questionnaire, participants were asked to rank general impression (Q1), usefulness of Scalable Insets (Q2), and simplicity to learn operating Scalable Insets (Q3) on a 5-point Likert scale ranging from 1: strongly disagree or negative to 5: strongly agree or positive. The results are presented in Table 1. Overall, the participants perceived the Scalable Insets approach as a promising (Q1) and useful (Q2) for exploration. The exceptionally high ratings for the usability (Q3) indicate that the participants had no problem learning how to operate Scalable Insets. The complete questionnaire can be found in the Supplementary Table S1. Two aspects were mentioned in the closing questionnaire multiple times: sudden disappearance of insets once the size of the original location is larger than a pre-defined threshold and the relatively low resolution of insets until scale-up. The first aspects highlight the need to adjust the dynamic rendering and placement of insets based on the user's region of interest, which could be inferred from the location of the mouse cursor. The relatively low resolution was due to performance reasons and can easily be mitigated through precomputation of image thumbnails.

6.2 Study 2: Qualitative Evaluation

The goal of our second study was to evaluate the usability and usefulness of Scalable Insets in the context of a scientific application. To that end we conducted six open-ended exploratory sessions with computational biologists, exploring large-scale genome interaction matrices in structural genomics (Figure 8) using both INSIDE and OUTSIDE. The apparatus was the same as in the first study.

Dataset—We obtained large-scale matrices at the order of 3×3 million rows and columns. The matrices were generated by biologists to study the probability of physical interaction between any two segments in the three-dimensional structure of a genome over time or across patients [9]. The matrix is visualized as a heatmap where darker cells indicate higher probability of phyical interaction [28]. Various algorithms are able to identify visual patterns inside the matrices that have potential biological implications for regulation of gene expression, replication, DNA repair, and other biological functions. The frequency of these patterns ranges from a few hundred to hundreds of thousands per matrix and analysts are interested in finding, comparing, and contextualizing patterns across many zoom levels (more details are described elsewhere [33]).

Participants—We recruited 6 computational biologists (2 female and 4 male) working in the field of structural genomics. 3 of the experts are PhD candidates and 3 are postdoctoral researchers. Each expert was already familiar with HiGlass but did not see its Scalable Insets extension before. All participants volunteered, had no visual impairments, and received $15 upon completion of the experiment.

Tasks & Study Design—The study consisted of individual open-ended sessions lasting between 20 and 30 minutes. Domain experts were asked to verbalize their thoughts and actions. Most participants started with INSIDE and some with OUTSIDE. In both cases the participants had to switch after half the time.

Procedure—Again, we first explained the study procedure and collected consent. Then, each participant was briefly introduced to Scalable Insets and the data that was to be explored (<2 minutes). Next, we asked the participants to freely explore the data while verbalizing their thoughts. The screen content and audio was recorded for later analysis. At the end, each participant anonymously filled out a questionnaire on the usability, usefulness, and missing features.

Results The results suggest that Scalable Insets is easy-to-learn. Participants immediately picked up the core concept of Scalable Insets and started exploring the data set as usual. Some participants started their exploration with slow pan and zoom, but all ended up using the interface in a fast manner. Having magnified and aggregated views of annotated patterns right inside the visual space was perceived useful for exploring genome interaction matrices. The domain experts were able to find and evaluate the detailed visual structure of the patterns early on without having to navigate to every location of an annotated pattern.

We notice that the inner- and outer-placement led to different behavior. For

	Study 1			**Study 2**							
	Q1	**Q2**	**Q3**	**Q1**	**Q2**	**Q3**	**Q4**	**Q5**	**Q6**	**Q7**	**Q8**
Mean	**3.9**	**3.8**	**4.5**	**4.2**	**4.7**	**3.7**	3.5	1.8	2.7	**4.7**	**4.8**
1, 2, and 3	5	5	3	1	0	1	3	6	2	0	0
4 and 5	13	13	14	5	6	5	3	0	1	6	6

Table 1: Results of the closing questionnaire in study 1 and 2. Note that Q3 of study 1 was only answered by 17 out of 18 participants and Q6 of study 2 was only answered by 3 out of 6 participants. Mean values are shaded by their values.

the outer-placement, people zoomed less often into the original location of an inset and instead compared the visual details of the patterns more frequently. P3 noticed *"It's easier to compare [the patterns] since they are all lined up nicely."*. Some participants preferred one placement approach over the other but everyone noted that it would be useful to have the ability to switch between placement modes fluently. The complete protocol of actions can be found in the Supplementary Table S2.

Some people needed more time to get used to the outer-placement approach with fading leader lines but identified the benefit for context preservation by themselves quickly. A drawback of the current implementation, mentioned by many participants, is the sudden disappearance of insets once their original location is larger than a pre-configured threshold (e.g., 24×24 pixels). Although it is necessary to remove insets, once the region of the annotated pattern becomes large enough, to release visual space for other patterns too small to be seen, it could be beneficial to adjust the threshold based on the region the user is zooming in or out. Insets in close proximity to the mouse cursor could remain visible until the user changes its focus through navigating elsewhere.

The closing questionnaire asked the participants to rate the usability and usefulness of the Scalable Insets approach for exploring genome interaction matrices. We asked participants to rate Scalable Insets on a 5-point Likert scale from 1: strongly disagree or negative to 5: strongly agree or positive on (Q1) their overall impression, (Q2) perceived usefulness of insets, (Q3) intuitiveness of the interface, (Q4) usefulness in its current form, (Q5) similar support by other tools, (Q6) performance comparison to other tools, (Q7) usefulness upon implementation of domain specific features, and (Q8) perceived usefulness outside the domain. The results are presented in Table 1 and the complete questionnaire can be found in the Supplementary Table S3.

The domain experts strongly indicated that they would use Scalable Insets to explore their annotated patterns, under the condition that additional application-specific and required features are added. For example, P4 would like to "pin" certain insets to compare and aggregate them with other annotated patterns. While this is easy to implement and already possible to some extent, it ultimately leads to different tasks which are already supported by complementary tools HiPiler [33], also available with HiGlass. Other participants requested to feature dynamically changing the color map, resolution, or size of an inset as well as adding and removing annotated patterns for highlighting.

7 Discussion

We designed Scalable Insets as a guidance technique to improve exploration and navigation of high numbers of annotated patterns for tasks that involve awareness of the patterns' detailed visual structure, context, and location. As the first study indicates, there is strong evidence that Scalable Insets support pattern search and comparison of pattern types. The second study found that the choice of placement depends on the importance of overview and context; inner-placement was preferred for contextualizing annotated patterns, while outer-placement was preferred for pattern comparison and gaining an overview.

Scalability and Limitations—Scalable Insets has been designed for dense multiscale visual spaces such as gigapixel images, maps, and large matrices, where every pixel is associated to some data point. The current prototypical implementation can simultaneously visualize and place about a hundred insets. The number of insets can be improved but it might require additional features such as filtering to cope with the cognitive challenge of too many patterns.

Tradeoff Between Details, Context, and Locality— Scalable Insets set out to provide a tradeoff between DETAIL, CONTEXT, and LOCALITY to manage exploration of high numbers of annotated patterns but to this end the tradeoff is configured upfront by employing sensible defaults for the three use cases presented in this paper. An unsolved question beyond the Scalable Insets technique is what defines a "good" tradeoff and how could this tradeoff be adjusted interactively during navigation and exploration.

Inset Design and Cluster Representation— The design of the insets content highly depends on the data type and specific tasks. We provide two generic approaches: piling for pattern types of homogeneous shape, such as dots or blocks in matrices, and a gallery view of cluster representatives for pattern types of high variance and diverging shape, such as patterns found in images and geographic maps. As participants in both studies noted, there are further possibilities for application specific cluster representations.

Generalizability—Scalable Insets is not limited to the three multiscale data types we have presented throughout this paper. Our technique can be applied to any kind of multiscale visualization that exhibits a large number of sparsely-distributed patterns. Even monoscale visual spaces that entail a third dimension, such as magnetic resonance imaging, could be enhanced with Scalable Insets. The effectiveness of Scalable Insets depends on how important the detailed visual structure, context, and location of the annotated patterns is. Scalable Insets is designed as a guidance technique with a minimal interaction space to be compatible with a wide range of navigation techniques. Therefore, while our prototypical implementation of Scalable Insets in HiGlass exclusively relies on pan and zoom for navigation, other navigation techniques like PolyZoom [25] could easily be used as well.

8 Conclusion and Future Work

Scalable Insets enable guided exploration and navigation of large numbers of sparsely-distributed annotated patterns 2D multiscale visual spaces. Scalable

Insets visualizes annotated patterns as magnified thumbnails and dynamically places and aggregates these insets based on location and pattern type. While currently, Scalable Insets supports images, maps, and matrices, we plan for other data types and scenarios, investigate techniques to cope with dense regions of patterns, and support more free-form exploration, e.g., through pinning and manually grouping of insets.

Acknowledgement

We wish to thank all participants from our two user studies who helped us evaluate Scalable Insets and provided useful feedback. This work was supported in part by the National Institutes of Health (U01 CA200059 and R00 HG007583).

References

[1] L. Barth, A. Gemsa, B. Niedermann, and M. Nöllenburg. On the readability of boundary labeling. In *International Symposium on Graph Drawing and Network Visualization*, pp. 515–527. Springer, 2015.

[2] B. B. Bederson and J. D. Hollan. Pad++: a zooming graphical interface for exploring alternate interface physics. In *Proceedings of the 7th annual ACM symposium on User interface software and technology*, pp. 17–26. ACM, 1994.

[3] M. A. Bekos, M. Kaufmann, A. Symvonis, and A. Wolff. Boundary labeling: Models and efficient algorithms for rectangular maps. *Computational Geometry*, 36(3):215–236, 2007.

[4] M. Bostock, V. Ogievetsky, and J. Heer. D^3 data-driven documents. *IEEE transactions on visualization and computer graphics*, 17(12):2301–2309, 2011.

[5] M. S. T. Carpendale and C. Montagnese. A framework for unifying presentation space. In *Proceedings of the 14th annual ACM symposium on User interface software and technology*, pp. 61–70. ACM, 2001.

[6] J. Christensen, J. Marks, and S. Shieber. An empirical study of algorithms for point-feature label placement. *ACM Transactions on Graphics (TOG)*, 14(3):203–232, 1995.

[7] C. Collins, F. B. Viegas, and M. Wattenberg. Parallel tag clouds to explore and analyze faceted text corpora. In *Visual Analytics Science and Technology, 2009. VAST 2009. IEEE Symposium on*, pp. 91–98. IEEE, 2009.

[8] T. R. de Janeiro Hong Kong Connection. Corcovado, 2018. [Online]. Available: http://www.gigapan.com/gigapans/66020 (Accessed: 31-March-2018).

[9] J. Dekker, A. S. Belmont, M. Guttman, V. O. Leshyk, J. T. Lis, S. Lomvardas, L. A. Mirny, C. C. O'shea, P. J. Park, B. Ren, and the 4D Nucleome Network. The 4d nucleome project. *Nature*, 549(7671):219, 2017.

[10] C. Dunne and B. Shneiderman. Motif simplification: improving network visualization readability with fan, connector, and clique glyphs. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 3247–3256. ACM, 2013.

[11] S. Edmondson, J. Christensen, J. Marks, and S. Shieber. A general cartographic labelling algorithm. *Cartographica: The International Journal for Geographic Information and Geovisualization*, 33(4):13–24, 1996.

[12] N. Elmqvist, T.-N. Do, H. Goodell, N. Henry, and J.-D. Fekete. ZAME: Interactive large-scale graph visualization. In *Visualization Symposium, 2008. PacificVIS'08. IEEE Pacific*, pp. 215–222. IEEE, 2008.

[13] N. Elmqvist, N. Henry, Y. Riche, and J.-D. Fekete. Melange: space folding for multi-focus interaction. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 1333–1342. ACM, 2008.

[14] M. Ester, H.-P. Kriegel, J. Sander, X. Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Kdd*, vol. 96, pp. 226–231, 1996.

[15] I. Facebook. React - a javascript library for building user interfaces, 2018. [Online]. Available: https://github.com/facebook/react/ (Accessed: 31-March-2018).

[16] D. S. Foundation et al. Django - the web framework for perfectionists with deadlines, 2018. [Online]. Available: https://github.com/django/django (Accessed: 31-March-2018).

[17] G. W. Furnas. *Generalized fisheye views*, vol. 17. ACM, 1986.

[18] G. W. Furnas and B. B. Bederson. Space-scale diagrams: Understanding multiscale interfaces. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 234–241. ACM Press/Addison-Wesley Publishing Co., 1995.

[19] S. Ghani, N. H. Riche, and N. Elmqvist. Dynamic insets for context-aware graph navigation. In *Computer Graphics Forum*, vol. 30, pp. 861–870. Wiley Online Library, 2011.

[20] GigaPan Systems. GigaPan, 2018. [Online]. Available: http://www.gigapan.com (Accessed: 31-March-2018).

[21] K. W. Hall, C. Perin, P. G. Kusalik, C. Gutwin, and S. Carpendale. Formalizing emphasis in information visualization. In *Computer Graphics Forum*, vol. 35, pp. 717–737. Wiley Online Library, 2016.

[22] D. Holten and J. J. Van Wijk. Force-directed edge bundling for graph visualization. In *Computer graphics forum*, vol. 28, pp. 983–990. Wiley Online Library, 2009.

[23] C. Y. Ip and A. Varshney. Saliency-assisted navigation of very large landscape images. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):1737–1746, 2011.

[24] S. Ishihara. *Tests for colour-blindness*. Kanehara Shuppan Company, 1960.

[25] W. Javed, S. Ghani, and N. Elmqvist. Polyzoom: multiscale and multifocus exploration in 2d visual spaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 287–296. ACM, 2012.

[26] S. Jul and G. W. Furnas. Critical zones in desert fog: aids to multiscale navigation. In *Proceedings of the 11th annual ACM symposium on User interface software and technology*, pp. 97–106. ACM, 1998.

[27] P. Karnick, D. Cline, S. Jeschke, A. Razdan, and P. Wonka. Route visualization using detail lenses. *IEEE transactions on visualization and computer graphics*, 16(2):235–247, 2010.

[28] P. Kerpedjiev. Fast contact matrix visualization for the web, 2017. [Online]. Available: https://github.com/hms-dbmi/higlass (Accessed: 31-March-2018).

[29] P. Kerpedjiev, N. Abdennur, F. Lekschas, C. McCallum, K. Dinkla, H. Strobelt, J. M. Luber, S. B. Ouellette, A. Ahzir, N. Kumar, J. Hwang, B. H. Alver, H. Pfister, L. A. Mirny, P. J. Park, and N. Gehlenborg. HiGlass: Web-based visual comparison and exploration of genome interaction maps. *bioRxiv*, 2017. doi: 10.1101/121889

[30] P. Kerpedjiev et al. Largest ski areas on each continent, 2015. [Online]. Available: http://emptypipes.org/largest-ski-areas (Accessed: 31-March-2018).

[31] H.-K. Kong, Z. Liu, and K. Karahalios. Internal and external visual cue preferences for visualizations in presentations. In *Computer Graphics Forum*, vol. 36, pp. 515–525. Wiley Online Library, 2017.

[32] B. Lee, S. Srivastava, R. Kumar, R. Brafman, and S. R. Klemmer. Designing with interactive example galleries. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 2257–2266. ACM, 2010.

[33] F. Lekschas, B. Bach, P. Kerpedjiev, N. Gehlenborg, and H. Pfister. HiPiler: Visual exploration of large genome interaction matrices with interactive small multiples. *IEEE transactions on visualization and computer graphics*, 24(1):522–531, 2018.

[34] A. Lex, C. Partl, D. Kalkofen, M. Streit, S. Gratzl, A. M. Wassermann, D. Schmalstieg, and H. Pfister. Entourage: Visualizing relationships between biological pathways using contextual subsets. *IEEE transactions on visualization and computer graphics*, 19(12):2536–2545, 2013.

[35] J. Liang and M. L. Huang. Highlighting in information visualization: A survey. In *Information Visualisation (IV), 2010 14th International Conference*, pp. 79–85. IEEE, 2010.

[36] G. D. Ltd. The html5 creation engine: Create beautiful digital content with the fastest, most flexible 2d webgl renderer, 2018. [Online]. Available: https://github.com/pixijs/pixi.js (Accessed: 31-March-2018).

[37] Mapbox, Inc. Mapbox, 2018. [Online]. Available: https://www.mapbox.com (Accessed: 31-March-2018).

[38] J. Marks, B. Andalman, P. A. Beardsley, W. Freeman, S. Gibson, J. Hodgins, T. Kang, B. Mirtich, H. Pfister, W. Ruml, et al. Design galleries: A general approach to setting parameters for computer graphics and animation. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pp. 389–400. ACM Press/Addison-Wesley Publishing Co., 1997.

[39] R. E. Mayer. *The Cambridge handbook of multimedia learning*. Cambridge university press, 2005.

[40] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon. Network motifs: simple building blocks of complex networks. *Science*, 298(5594):824–827, 2002.

[41] T. Moscovich, F. Chevalier, N. Henry, E. Pietriga, and J.-D. Fekete. Topology-aware navigation in large networks. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 2319–2328. ACM, 2009.

[42] OpenStreetMap contributors. Planet dump retrieved from https://planet.osm.org . `https://www.openstreetmap.org`, 2017.

[43] K. Perlin and D. Fox. Pad: an alternative approach to the computer interface. In *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, pp. 57–64. ACM, 1993.

[44] E. Pietriga, F. Del Campo, A. Ibsen, R. Primet, C. Appert, O. Chapuis, M. Hempel, R. Muñoz, S. Eyheramendy, A. Jordan, et al. Exploratory visualization of astronomical data on ultra-high-resolution wall displays. In *Software and Cyberinfrastructure for Astronomy IV*, vol. 9913, p. 99130W. International Society for Optics and Photonics, 2016.

[45] C. Pindat, E. Pietriga, O. Chapuis, and C. Puech. Jellylens: content-aware adaptive lenses. In *Proceedings of the 25th annual ACM symposium on User interface software and technology*, pp. 261–270. ACM, 2012.

[46] R. Rao and S. K. Card. The table lens: merging graphical and symbolic representations in an interactive focus+ context visualization for tabular information. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 318–322. ACM, 1994.

[47] S. S. Rao, M. H. Huntley, N. C. Durand, E. K. Stamenova, I. D. Bochkov, J. T. Robinson, A. L. Sanborn, I. Machol, A. D. Omer, E. S. Lander, et al. A 3D map of the human genome at kilobase resolution reveals principles of chromatin looping. *Cell*, 159(7):1665–1680, 2014.

[48] A. C. Robinson. Highlighting in geovisualization. *Cartography and Geographic Information Science*, 38(4):373–383, 2011.

[49] M. Sarkar, S. S. Snibbe, O. J. Tversky, and S. P. Reiss. Stretching the rubber sheet: a metaphor for viewing large layouts on small screens. In *Proceedings of the 6th annual ACM symposium on User interface software and technology*, pp. 81–91. ACM, 1993.

[50] P. Siangliulue, J. Chan, S. P. Dow, and K. Z. Gajos. Ideahound: improving large-scale collaborative ideation with crowd-powered real-time semantic modeling. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, pp. 609–624. ACM, 2016.

[51] J. O. Talton, D. Gibson, L. Yang, P. Hanrahan, and V. Koltun. Exploratory modeling with collaborative design spaces. *ACM Transactions on Graphics-TOG*, 28(5):167, 2009.

[52] The Rio de Janeiro - Hong Kong Connection. The Rio de Janeiro - Hong Kong Connection, 2018. [Online]. Available: https://www.visgraf.impa.br/riohk/ (Accessed: 31-March-2018).

[53] C. Tominski, S. Gladisch, U. Kister, R. Dachselt, and H. Schumann. Interactive lenses for visualization: An extended survey. In *Computer Graphics Forum*, vol. 36, pp. 173–200. Wiley Online Library, 2017.

[54] M. Trapp, C. Beesk, S. Pasewaldt, and J. Döllner. Interactive rendering techniques for highlighting in 3d geovirtual environments. In *Advances in 3D Geo-Information Sciences*, pp. 197–210. Springer, 2011.

[55] S. Van den Elzen and J. J. Van Wijk. Multivariate network exploration and presentation: From detail to overview via selections and aggregations. *IEEE Transactions on Visualization and Computer Graphics*, 20(12):2310–2319, 2014.

[56] C. Ware and M. Lewis. The dragmag image magnifier. In *Conference companion on Human factors in computing systems*, pp. 407–408. ACM, 1995.

[57] W. Willett, J. Heer, and M. Agrawala. Scented widgets: Improving navigation cues with embedded visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1129–1136, 2007.

[58] J. Zhao, D. Wigdor, and R. Balakrishnan. Trailmap: facilitating information seeking in a multi-scale digital map via implicit bookmarking. In

Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 3009–3018. ACM, 2013.