Systemizing Glyph Design for Visualization

Report for Confirmation of Status

Eamonn James Maguire

April 2014

Thesis abstract

In all fields, from the social sciences to biology to physics, data volume, velocity and variety are increasing. Alongside this increase is a drive to understand and make sense of these data, albeit at a much smaller pace. Co-interacting components of this process of understanding are statistical analysis and visualization. As the data to visualize grows, the ability to display all information on a single average sized display is becoming less possible. As such, we can turn to statistics as a guide to indicate which parts of the visualization should be more or less important, using the knowledge within the large collections of psychology literature to give visual emphasis to important attributes of the data.

In this thesis I investigate the link between statistics, visualization and psychology to create visualizations that make information easier to digest. This effort largely relies on glyphs, visual objects composed of one or more visual variables that represent different parts of multi-variate data record. Such glyphs provide a way of visualizing a large amount of information in a compact representation. Their composition however is important in depicting the variables of a data record in a meaningful way. Here we provide a number of systematic mechanisms to compose glyphs that: use appropriate visual channels to depict data variables, meaning the most important information is distinguishable even at low resolutions; are meaningful; and separable with respect to other glyphs in the user display.

This thesis presents a number of novel contributions. Firstly, we present a systematic approach for glyph design taking in to consideration the statistical properties of a qualitative data set and using these statistics to drive glyph creation guided by the psychology literature. Secondly, we present a novel algorithm to compress workflow visualizations using computationally created multi-resolution glyphs. Thirdly, a technique for compression of time series visualization using glyphs. Finally, we present a methodology to create glyphs that are visually separable using a quasi-hamming distance based approach.

Status of Research to Date

At this point, the major body of my research in systemizing glyph design is complete. In the first year, much of my time was spent reading the literature on glyphs in general, and perception literature detailing how the human visual system works. This work was capped by a paper showing the first systematic way of creating glyphs and was presented at IEEE VisWeek in 2012 and published in IEEE TVCG [MRSS+12].

In the second year, I contributed to a number of glyph papers including a survey of the domain [BKC+13] and a piece of work on the use of glyphs for visualizing how poems are read [ARLC+13]. The more extensive piece of work was based on the IEEE VisWeek [MRSS+12] paper that proposed an approach for the compression of common regions of workflows, with the use of automatically generated glyphs to represent those compressed regions. This work was presented at IEEE VIS (renamed from VisWeek) in 2013 and published in IEEE TVCG [MRSS+13].

In the third and current year, I've been working on two major pieces of work. The first is the compression of common regions (motifs) in time series data using glyphs. This is backed by a visual analytics approach to help users in tuning the parameters of the algorithm used to find these common regions. This work has been submitted for IEEE VIS 2014 and is awaiting a response [MSC14]. Additionally, I have contributed significantly to another paper on creating an approach to design glyphs that are visually distinct from each other using a 'quasi hamming distance' metric. This work has also been submitted to IEEE VIS 2014 [LMW+14]. Additionally, two are to appear in the EuroVis 2014 Proceedings as short papers: the first a collaboration with a member of the visualization group at Oxford on poem visualization and the use of animated glyphs to represent turbulence of sound when the poem is read [ARMC14]; and the second presents the use of glyphs to aid interpretation of sequence logos, used to determine conserved regions of protein or DNA sequences [MRSSC14].

Thesis Organization

Following the Introduction (Chapter 1) and Related Work (Chapter 2) based largely on [BKC+13], the thesis is organized in to five main chapters: Chapter 3 on the outline of strategies for systematic glyph creation; Chapter 4 on the creation of glyphs using a taxonomy-based approach (based on [MRSS+12]); Chapter 5 on the visual compression of workflow visualizations using glyphs with an automated approach for the creation of those glyphs (based on [MRSS+13]); Chapter 6 on the compression of time series visualizations with automatically created glyphs (based on [MSC14]); and Chapter 7 on the creation of separable glyphs using a quasi-hamming distance-based approach (based on [LMW+14]). Finally, in Chapter 8 the contributions of the thesis are summarized and future work is discussed.

Research to be done before submission

The major body of research has been completed.

Proposed timetable

April – September 2014: Consolidate paper submissions and recent work into thesis.

October 2014: Thesis submission.

Appendices

Page 6: Proposed table of contents.

Page 10: Copy of [MRSS+12]: Taxonomy-based Glyph Design — with a Case Study on Visualizing Workflows of Biological Experiments.

Page 47: Copy of [ARLC+13]: Rule-based Visual Mappings – with a Case Study on Poetry Visualization.

Page 21: Copy of [BKC+13]: Glyph-based Visualization: Foundations, Design Guidelines, Techniques and Applications.

Page 58: Copy of [MRSS+13]: Visual Compression of Workflow Visualizations with Automated Detection of Macro Motifs.

Page 69: Copy of [MRSSC14]: Redesigning the Sequence Logo with Glyph-based Approaches to Aid Interpretation.

Page 75: Copy of [ARMC14]: Comparing Three Designs of Macro-Glyphs for Poetry Visualization.

Page 81: Copy of [LMW+14]: Fail-safe Glyph Encoding based on Quasi-Hamming Distances: With a Case Study on Visualizing File System Events.

Page 92: Copy of [MSC14]: A Visual Analytical Approach for Model Editing and Testing in Glyph-based Time Series Compression.

References

[ARLC+13] A. Abdul-Rahman, J. Lein, K. Coles, E. Maguire, M. Meyer, M. Wynne, C.R. Johnson, A. Trefethen, and M. Chen, *Rule-based visual mappings – with a case study on poetry visualization*, Computer Graphics Forum **32** (2013), no. 3, 381–390.

[ARMC14] A. Abdul-Rahman, E. Maguire, and M. Chen, *Comparing three designs of macro-glyphs for poetry visualization*, In Proceedings of EuroVis 2014, Short Paper (2014).

[BKC+13] Rita Borgo, Johannes Kehrer, David H.S. Chung, Eamonn Maguire, Robert S. Laramee, Helwig Hauser, Matthew Ward, and Min Chen, *Glyph-based visualiza-*

tion: Foundations, design guidelines, techniques and applications, Eurographics State of the Art Reports, EG STARs, May 2013, pp. 39–63.

[LMW+14] P. Legg, E. Maguire, S. Walton, S. Creese, M. Goldsmith, and Min Chen, *Fail-safe glyph encoding based on quasi-hamming distances: With a case study on visualizing file system events*, IEEE Transactions on Visualization and Computer Graphics *(Submitted)* (2014).

[MRSS+12] Eamonn Maguire, Philippe Rocca-Serra, Susanna-Assunta Sansone, Jim Davies, and Min Chen, *Taxonomy-based glyph design — with a case study on visualizing workflows of biological experiments*, IEEE Transactions on Visualization and Computer Graphics **18** (2012), no. 12.

[MRSS+13] E. Maguire, P. Rocca-Serra, S.-A. Sansone, J. Davies, and Min Chen, *Visual compression of workflow visualizations with automated detection of macro motifs*, IEEE Transactions on Visualization and Computer Graphics **19** (2013), no. 12, 2576–2585.

[MRSSC14] E. Maguire, P. Rocca-Serra, S.-A. Sansone, and M. Chen, *Redesigning the sequence logo with glyph-based approaches to aid interpretation*, In Proceedings of EuroVis 2014, Short Paper (2014).

[MSC14] E. Maguire, S.-A. Sansone, and Min Chen, *A visual analytical approach for model editing and testing in glyph-based time series compression*, IEEE Transactions on Visualization and Computer Graphics *(Submitted)* (2014).

A Proposed Table of Contents

Contents

B Copy of [MRSS+12]:

Taxonomy-based Glyph Design — with a Case Study on Visualizing Workflows of Biological Experiments

Taxonomy-Based Glyph Design — with a Case Study on Visualizing Workflows of Biological Experiments

Eamonn Maguire, Philippe Rocca-Serra, Susanna-Assunta Sansone, Jim Davies, and Min Chen

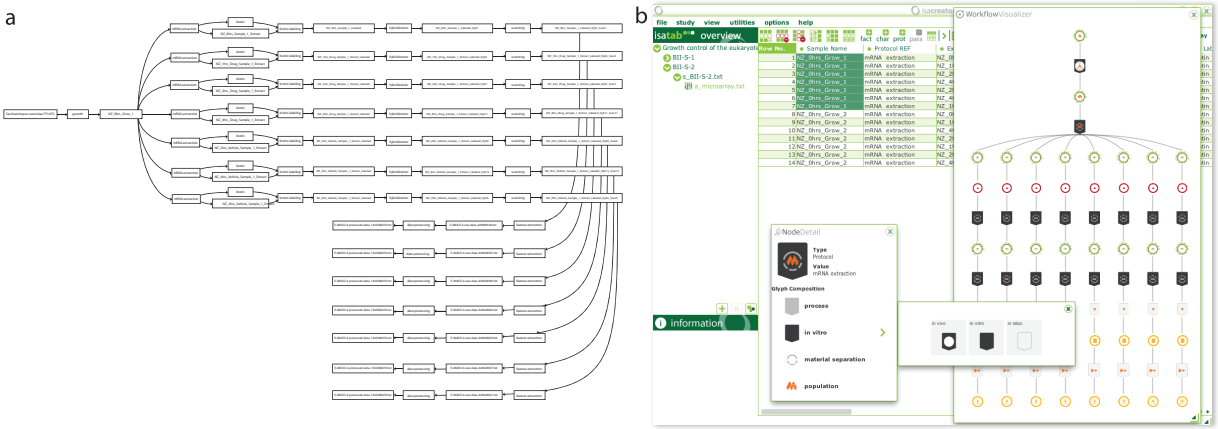


Fig. 1. a) Workflow as rendered currently using toolkits such as GraphViz. b) We propose to replace the textual labels with glyphs, while allowing interactive access to detailed descriptions. This makes it easy to gain an overview, search components and compare workflows. The screenshot shows a prototype developed within ISAcreator, a system for capturing biological experiment metadata.

Abstract—Glyph-based visualization can offer elegant and concise presentation of multivariate information while enhancing speed and ease in visual search experienced by users. As with icon designs, glyphs are usually created based on the designers' experience and intuition, often in a spontaneous manner. Such a process does not scale well with the requirements of applications where a large number of concepts are to be encoded using glyphs. To alleviate such limitations, we propose a new systematic process for glyph design by exploring the parallel between the hierarchy of concept categorization and the ordering of discriminative capacity of visual channels. We examine the feasibility of this approach in an application where there is a pressing need for an efficient and effective means to visualize workflows of biological experiments. By processing thousands of workflow records in a public archive of biological experiments, we demonstrate that a cost-effective glyph design can be obtained by following a process of formulating a taxonomy with the aid of computation, identifying visual channels hierarchically, and defining application-specific abstraction and metaphors.

Index Terms—Glyph-based techniques, taxonomies, design methodologies, bioinformatics visualization.

1 Introduction

Glyph-based visualization is a class of visual representations where a collection of small visual objects (referred to as *glyphs*) are used to encode different attribute dimensions of an input data space. A good glyph design can enable users to conduct visual search more efficiently during interactive visualization, and facilitate effective learning, memorizing and using the visual encoding scheme. A less effective visual design may suffer from various shortcomings such as being perceptually confusing, semantically ambiguous, difficult to learn and remember, or unable to accommodate low-resolution display devices. Most accepted designs have undergone an enduring process of evolution, refinement and standardization. One could not and should not remove the necessity for such processes in glyph design. Meanwhile, the design process for a glyph set usually relies on an assortment of creativity, artistic skill, domain knowledge, intuition about users, and sometimes personal preference. While most of these qualities are absolutely helpful and some are inevitable, it is highly desirable to introduce "systematicism" and objectivity into the design process.

To demonstrate our approach, we elected the field of experimental biology as a test bed for developing a systematic methodology to create a glyph-based visual mapping for visualizing experimental design and experimental processes. While experimental design is at the heart of biological data evidence gathering, representation of such information is confined to either verbose description or ineffective representations. This claim is backed by a survey of public biodata repositories, devised to ensure data perennity, scientific scrutiny and reproducibility. As illustrated in Fig. 1(a), workflows are traditionally drawn as text-based diagrams. Text labels are indices of concepts, and usually they do not encode multivariate information directly. Text-labeled boxes are costly in terms of display space usage as well as time required for visual search since parsing and interpretation of text is a slow post-attentive process [59]. They are particularly ineffective when one needs to compare between different workflows or to identify unusual or missing components in a workflow.

Therefore, exploring an alternative in the form of glyph based rep-

- *Eamonn Maguire is with Oxford e-Research Centre and Department of Computer Science, University of Oxford, UK. E-mail: eamonn.maguire@st-annes.ox.ac.uk.*
- *Philippe Rocca-Serra, Susanna-Assunta Sansone and Min Chen are with Oxford e-Research Centre, University of Oxford, UK. E-mail: {philippe.rocca-serra,susanna-assunta.sansone,min.chen}@oerc.ox.ac.uk.*
- *Jim Davies is with Department of Computer Science, University of Oxford, UK. E-mail: jim.davies@cs.ox.ac.uk.*

Manuscript received 31 March 2012; accepted 1 August 2012; posted online 14 October 2012; mailed on 5 October 2012.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org.

resentations, which are at the core of many successful schematic diagrams in the history of sciences, engineering and business management, offers a potentially more effective means for depicting these workflows. This presents us with an interesting case study where other types of design approaches would not be appropriate, especially in dealing with several hundreds of conceptual names.

It is important to note that domain experts are in general more willing and able to learn and memorize an encoding scheme in order to improve the accuracy and efficiency of routine tasks. At the same time, it is also necessary for an encoding scheme to facilitate effective learning and remembering through appropriate abstraction and metaphors.

Motivated by this case study, we made an observation that when a large number of concepts (or taxa) are organized into a taxonomic tree, the hierarchy typically represents an ordering of different categorization schemes. The schemes that are higher-up in the taxonomy (closer to the root) are usually considered to be more important, which reflect their conceptual coverage, frequency of usage, domain-specific convention, and some other measurable factors. In terms of visual encoding, higher up schemes should ideally be mapped to visual channels that have more discriminative capacity. Hence, we can explore the parallel between the taxonomic hierarchy and the ordering of visual channels based on discriminative capacity to formulate a systematic and relatively objective process for designing a glyph set. This approach addresses one of the most common criticisms of data glyphs: the data to visual attribute mapping bias [58]. Given a large data repository that encompasses many concepts, our design process is composed of four major steps: (i) gathering and processing raw metadata for obtaining a set of taxa (names); (ii) formulating a taxonomy based on a set of categorization schemes (Section 4); (iii) carrying out visual design, which includes the sub-process of determining the ordering of visual channels, proposing optional visual mappings, and identifying metaphoric abstractions and associations (Section 5); and (iv) implementing a glyph-based visualization system, in our case, for depicting workflows of biological experiments (Section 6).

Similar to most design processes, it is helpful to conduct all stages in a progressive and iterative manner. As glyph-based visualization is normally deployed in a specific application domain, it is important to involve domain experts at every stage of the design process. During this work, we met with domain specialists on a weekly basis.

2 Related Work

In this section, we give a brief overview of two most relevant areas in visualization, *glyph-based visualization* and *workflow visualization*. The remainder background information includes the biological data management, perceptual guidance, and categorization algorithms, which will be described in the following sections where the relevant technical details are discussed.

2.1 Glyph-based Visualization

There are many examples of glyph usage in the literature spanning many disciplines, especially in conjunction with many schematic diagrams. Bertin considered a range of simple glyphs in the context of geo-information visualization [7]. Ward surveyed the use of glyphs in visualization, and discussed a number of bias issues, design approaches, and layout options [58]. Ropinksi *et al.* conducted a survey on glyph-based techniques in medical visualizations [48]. In visualization, a number of interesting glyph designs have been presented with noticeable impact on a large range of applications (e.g., medicine [25], software [13], text [47], and scientific computation [62]). Furthermore, Ribarsky *et al.* developed an editing system, Glyphmaker [45], to assist the process of designing glyphs. Post *et al.* proposed a language, Icon Modeling Language, for creating glyphs and glyph-style contours [42].

There have been some recent efforts to use glyph-based visualization in biological sciences. One noticeable attempt is the *Systems Biology Graphical Notation* (SBGN) [28]. The design of SBGN makes use of the notional representations of UML with some modifications to describe the biological entities and their interactions within a biological system. GenoCAD [11] provides a grammar-based language for representing and searching DNA sequences and for building genetic constructs from DNA sequences, facilitating the use of conventional link diagrams. Both systems rely heavily on text labels. In handling a large collection of workflows, they exhibit a few shortcomings, such as inefficiency in using display space and ineffectiveness in supporting some visualization tasks, such as comparisons and novelty or anomalies detection in workflows.

In the literature, several authors encouraged glyph designers to consider visual perception when constructing glyphs [58, 23]. Others examined the design space of icons, which normally encode less information than glyphs (e.g., [21, 29]). There were perceptual studies showing the merits of icons over text labels (e.g., [34, 41]), as well as those showing the contrary (e.g., [60]). Building on such discussions, this work aims to address a methodological need for a systematic approach to glyph design for applications where large collections of concepts need to be visually encoded using glyphs.

2.2 Workflow Visualization

The need for *workflow visualization* is pervasive across many different domains. For example, in business and management the Gantt chart is a form of text based workflow visualization, while BPMN (Business Process Model and Notation) and EPC (Event-driven Process Chain) make use of icons to enrich text labels. In engineering disciplines, various schematic diagrams, such as UML (Unified Modeling Language), Petri-net and circuit diagrams, are used to convey data flow and process interaction.

In this work, we consider workflows used to describe biological experiments. This class of workflow visualization renders the processes enacted on biological materials in an experiment (e.g., removal of blood sample from patient). There has been little effort to develop tools that address the domain-specific needs. Scientists usually use generic text-based graph drawing tools such as GraphViz [3].

One related aspect is *pathway visualization*, which is concerned with the rendering of cellular biological processes. An array of pathway visualization tools have been developed, some of which incorporate glyphs. For example, VANTED [22] overlays glyphs representing gene expression, enzyme and metabolite profile data on top of pathway diagrams from KEGG [37]. GENeVis [9], which also employs glyphs to represent multivariate data, is a comprehensive visualization toolkit for exploration of pathways in conjunction with temporal data. GenoCAD has at its core a workflow generation system and relies on their glyph library for rendering of the different biological processes within a cell. SBGN-ED [14], is an add-on for the VANTED software and performs pathway creation using the aforementioned SBGN [28] visual language.

3 Motivation and Overview

The advent of massively parallel techniques such as DNA microarray, mass-spectrometry based proteomics and metabolomics and next generation sequencing enables molecular biologists to collect, process and manipulate biological signals on a new scale. Big data in biology is a reality. There has been serious effort in biology to ensure quality of experimental records by providing data archiving infrastructure and defining standards for adequate annotation and sufficient details for recapitulation of results. A number of molecular biology signature databases have been or are being established to cover the main molecular dimensions: transcript, protein and metabolites. The captured metadata mostly revolves around a similar configuration where sets of samples corresponding to different conditions are processed, measurements produced and analysis performed, delivering data that needs to be handled and interpreted. While the availability of such metadata facilitates comparison across datasets and enables meta-analysis, providing means to serve an overview of experimental design can assist analysts and data managers alike, from data selection according to relevancy, to error detection such as imbalances, irregularities or inconsistencies in records.

Task Analysis. In the context of meta-databases for experimental records, there are two main groups of users. The majority of users are those who create data for their biological experiments or retrieve data

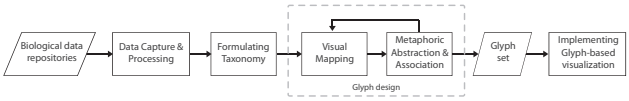


Fig. 2. A systematic process for creating glyph based representations.

relevant to their scientific interests. Their tasks include: (a) entering and editing their own experimental records; (b) transforming workflow records to schematic diagrams for comparison, external memorization, publication and education; (c) uploading and downloading datasets; (d) querying and searching for relevant experiments; (e) understanding workflows of existing experimental designs; (f) identifying similarity and difference between workflows for a common task; (g) understand pooling events and sample relations (derivation).

The second group of users are curators who manage data archives. As biology or bioinformatics scientists themselves, they perform the above-mentioned tasks (d)-(g) frequently, and (b)-(c) occasionally. In addition, they also carry tasks for: (h) checking syntactic correctness of submitted workflows; (i) checking semantic correctness of submitted workflows which usually involves reading the associated publications then comparing the submitted workflows with those described in the publications; (j) interacting with authors of submitted workflows to clarify any inconsistency and misunderstanding; (k) making appropriate corrections in submitted records where necessary; (l) augmenting with appropriate annotation based on additional information found in the associated publications; (m) providing authors with submission feedback. (n) forming an up-to-date overview about the experiments in the data archives, and maintaining an insight about the provenance of the archives; (n) analyzing the grouping, replication, distribution and trend of experiments; (o) analyzing ambiguities, errors and uncertainty in experimental recording, and identifying the need to enrich and refine the relevant standards for ontology, labelling and annotation. (p) providing tools to assist users in creating meta-data from raw data.

It is not difficult to observe that effective workflow visualization can significantly improve users' capability in performing tasks b, e, f, g, h, i, j, l, n, o and p.

While it is necessary to evolve a glyph-based design for workflow visualization over a period, it is also important not to make the first step in an ad hoc manner. Such a process does not scale well with the requirements of the application concerned where a large number of concepts are to be encoded using glyphs. We thus adopt a new systematic process for glyph design by exploring the parallel between the hierarchy of concept categorization and the ordering of discriminative capacity of visual channels. Fig. 2 depicts this process.

Data Capture and Processing. We first retrieved workflow metadata from a biological repository (content of the ArrayExpress archive), through use of a multi-threaded harvesting operation. All experimental workflows were then converted using a MAGE-Tab to ISA-Tab converter, the latter format being more general than the former and can be used to carry metadata payloads for many types of experiments, making the data processing program more "future-proof".

From the 21,000 ISA-Tab files, we extracted all names of protocols (processes) and biological materials, chemical materials, devices and data used in annotation. We also computed the occurrence of each material and process found in the entire set of ISA-Tab files, which have been used as one of the metrics in the next stage (see Section 5 for details). This step results in 61 process names and 3492 names of inputs and outputs (e.g., biological and chemical materials, device measurements and data).

Taxonomy Formulation. Since a taxonomy for such a large collection of terms is absent, we, for starters, established a set of categorization schemes with the help of domain experts. For example, one of the schemes for categorizing processes may be based on different biological inputs to a process (e.g., molecule, cell, organism, etc.). Another scheme may be based on processing methods (e.g., perturbation, combination, etc.). We computed the quality measures of each scheme based a set of generic metrics (see Section 4) then created a taxonomic tree by recursively selecting the best categorization scheme based on the quality measures. We finalized the organization of the taxonomy by allowing domain experts (2 co-authors) to make adjustments according to domain-specific conventions. All leaf nodes of the resulting taxonomy tree are names extracted from the workflow metadata. All non-leaf nodes represent a categorization scheme.

Glyph Design. First, we established an ordering of commonly-used visual channels based on the literature focused on perception and visualization (see Section 5). This allows us to systematically compare the order of categorization schemes in the taxonomy with the order of different visual channels. Ideally, schemes at the upper level of the tree can be mapped to visual channels which are more prominent to our visual system.

We then proceeded to the glyph design process involving two intertwined sub-processes for **visual mapping** and **metaphoric abstraction and association**. We proposed various options of visual channels for each categorization scheme featured in the taxonomic tree. We considered the merits of these visual channels and identify potential conflicts with the visual channels that have been assigned to other schemes. With the direct help from domain experts, we tried to identify a metaphoric abstraction or association for each design option proposed. We evaluated and recorded the intuitiveness and suitability of the abstraction and metaphor association.

Informed by the results the two sub-processes, we finalized a glyph set by selecting a design option for each scheme, while maintaining the order of discriminative capacity of visual channels, minimizing conflicts between different channels, and maximizing the use of metaphoric abstraction and association.

Implementation of Glyph-based Visualization. We then integrated the glyph set with a layout algorithm to form a prototype system for workflow visualization (see Section 6). For demonstration purposes, we tested the workflow visualization in conjunction with ISAcreator, a popular domain agnostic biological experiment metadata capture tool.

4 Taxonomy Generation

Given a large collection of concepts, we should ideally make use of a standard taxonomy, where non-leaf nodes represent different categorization schemes and each scheme provides subclasses that lead to different sub-trees. In many circumstances, however, there is no agreed taxonomic tree as establishing a standard categorization requires non-trivial scientific effort that often spans over a few decades.

The algorithm described below is not intended to establish a semantic-rich taxonomic tree for classifying concepts. It is designed purely for addressing the needs for ordering various categorization schemes (when there is no such an agreed order) to aid glyph design, and for devising a useful tree data structure in implementing the mapping between concepts and glyphs in workflow visualization.

Hence, the criteria used for structuring the tree are based on usage of the concepts and the structural quality of the tree to be constructed.

Taxonomy is a long standing concept that can be traced back to 3000BC [32]. Automatic taxonomy generation has been an active field in computer science and computational biology with existing work largely focusing on clustering algorithms (e.g., [27]). Many such algorithms assume the availability of similarity measures for ordering entities rather than relying on the existence of individual categorization schemes. In these algorithms, there is usually no attempt towards application of a metric for creating meaningful non-leaf nodes in the resulting tree. In this work, it is essential to keep each categorization scheme as a non-leaf node unless it is redundant.

4.1 Ordering Classification Schemes

Let $\mathscr{X} = \{x_1, x_2, \ldots, x_n\}$ be a set of concepts to be classified. In our application, this is the set of all valid names of biological processes and IOs (inputs and outputs) in the database. Each concept x_i is associated with a scalar value, $\mu_i \in [0,1]$, indicating its frequency of usage in relation to the total occurrence of all concepts in the database. There are a number of categorization schemes, $\mathscr{S} = \{S_1, S_2, \ldots, S_m\}$. Each scheme, S_k divides concepts into several classes, $c_1^k, c_2^k, \ldots, c_{l_k}^k$. The

relationship between the concept set $\mathscr{X}$ and different classification schemes can thus be represented by a Boolean matrix, **A**, where each element $\alpha[i,j,k]=1$ if concept x_i belongs to the j^{th} class of scheme S_k; otherwise $\alpha[i,j,k]=0$. In the context of feature-based categorization, one can also view each scheme S_k as a feature, and each class under S_k as a particular type of this feature. Without losing generality, we assume that the classes under the same S_k are disjoint. It is also possible that a concept does not possess the k^{th} feature, and hence does not belong to any class under S_k.

Given the above categorical information about the concept set $\mathscr{X}$, one can choose a categorization scheme $S_k \in \mathscr{S}$, which will divide $\mathscr{X}$ into a number of disjoint subsets corresponding to classes $c_1^k, c_2^k, \ldots, c_{l_k}^k$ and c_0^k. The subset c_0^k contains those concepts which S_k is unable to classify. The partitioning process can be repeated recursively by applying one of the remaining categorization schemes in $\mathscr{S}$ to each subset. This results in a hierarchical categorization tree, which defines a taxonomy for the concepts set $\mathscr{X}$.

The ordering of the schemes in $\mathscr{S}$ thus determines the taxonomic structure of $\mathscr{X}$. It is not difficult to anticipate that many criteria can be used to determine the ordering for a given concept set. Some criteria will no doubt encode application-specific semantics, and some may be subjective or debatable. However, there are also some common-sense criteria that are generic to most applications. These include:

Coverage. The number of concepts that can be classified by scheme S_k is a capacity measure of S_k. The more concepts that S_k can classify (i.e., the fewer in c_0^k), the better. The measure, which is normalized by the set size $|\mathscr{X}|=n$, can be defined as:

$$M_1(S_k) = \frac{\sum_{i=1}^{n} \max_{1\le j\le l_k}(\alpha_{i,j,k})}{n}. \tag{1}$$

Potential Usage. A categorization scheme that is higher up in the taxonomical tree is expected to be used more often in the application concerned. The occurrence frequencies of concepts, μ_i, enable us to estimate the potential usage of a classification scheme as follows:

$$M_2(S_k) = \frac{\sum_{i=1}^{n} \mu_i \max_{1\le j\le l_k}(\alpha[i,j,k])}{\sum_{i=1}^{n} \mu_i}. \tag{2}$$

Subtree Balance. Having a balanced node distribution in a tree is a desirable property of a tree structure. It prevents a tree from having an excessive height, which corresponds to the need for more visual channels. Let l_k denote the number of subclasses in categorization scheme S_k, τ_j be the number of concepts in each subclass $c_j^k, j=1,2,\ldots,l_k$ and σ_τ and $\bar{\tau}$ be the standard deviation and mean respectively of $\tau_1,\ldots,\tau_{l_k}$. We can measure the level of balance as follows:

$$M_3(S_k) = \begin{cases} 0 & \sum_{i=1}^{l_k} \tau_{l_i} = 0 \\ 1 & \sigma_\tau < \varepsilon (\varepsilon > 0) \\ P(\bar{\tau} \pm 1 \mid N_{\bar{\tau},\sigma_\tau}) & \bar{\tau} > 0 \;\&\; \sigma_\tau > \varepsilon \end{cases} \tag{3}$$

where P is the probability that a value within $[\bar{\tau}-1, \bar{\tau}+1]$ falls under the curve given by the normal distribution N with $(\bar{\tau}, \sigma_\tau)$ [40]. There are two special cases. When all values of τ_j are 0, S_k cannot classify any concept; hence the metric returns a zero score. When $\sigma_\tau = 0$, the subtree is totally balanced; hence the metric returns one. As σ_τ approaches zero, the function P becomes numerically unstable, we use a cut-off value ε to prevent this. In this work, we set $\varepsilon = 0.00001$. The normal distribution is preferred over a χ-test, as χ may not be reliable when l_k is a small number.

Number of Subclasses. All visual channels used in glyphs have limited discriminative capacity. A higher number of subclasses in a scheme would require visual encoding to have more codewords (e.g., more colors or more shape types), which will increase users' cognitive load in learning, remembering, and recognizing the codewords. It is thus more desirable to have a smaller number of subclasses, except that a categorization scheme with fewer than 2 sub-classes is useless. Let $\eta^\top$ be an up-limit for the number of codewords, which is set to 10 in this work. We introduce the following metric to measure the discriminative capacity of a scheme:

$$M_4(S_k) = \begin{cases} 0 & \eta_k < 2 \\ \frac{\eta^\top - \eta_k + 2}{\eta^\top} & 2 \le \eta_k \le \eta^\top \\ \frac{1}{\eta^\top} & \eta_k > \eta^\top \end{cases} \tag{4}$$

Although the above four metrics have been normalized to ensure their functional values within the $[0,1]$ domain, the distribution of the values for different schemes can still be rather application-specific and may vary substantially between different metrics. This may lead to inconsistency in combining these metrics.

We thus provide an optional linearization filter for these metrics by mapping the values obtained using a each metric M_i into fractional ranking numbers, which are then normalized into the $[0,1]$ domain with 1 being the best and 0 the worst. For example, consider a set of six schemes with metric values:

$$(S_1,0.5),(S_2,0.3),(S_3,0.54),(S_4,0.8),(S_5,0.85),(S_6,0.6).$$

We first sort the set:

$$(S_2,0.3),(S_1,0.5),(S_3,0.54),(S_6,0.6),(S_4,0.8),(S_5,0.85)$$

We then obtain the normalized ranking values via the *distance for ordinal variables* function $\sigma = \frac{r-1}{R-1}$ where R is the top rank and r is the rank position for each schema.

$$(S_2,0),(S_1,\frac{1}{5}),(S_3,\frac{2}{5}),(S_6,\frac{3}{5}),(S_4,\frac{4}{5}),(S_5,\frac{5}{5}).$$

We denote this filter as a function $R(S_k, M_i, \mathscr{S})$. Using the above set of metrics in conjunction with the filter R, we can derive a combined metric as

$$M(S_k) = \frac{\sum_1^4 \omega_t R(S_k, M_t(S_k), \mathscr{S})}{\sum_1^4 \omega_t}.$$

where $\omega_t, t=1,2,3,4$ are user-adjustable weights for the four individual metrics. Similar to weights in clustering algorithms, these weights need to be used with care as they introduce additional semantics into the ordering algorithm.

Equipped with the combined metric M, the algorithm for establishing an order of different schemes in $\mathscr{S}$ can be described as follows:

```
proc select(𝒮, 𝒳) ≡
  S_BEST := null; m_BEST := 0;
  for S ∈ 𝒮 do
    m := M(S);
    if m > m_BEST
      S_BEST := S; m_BEST := m;
      output(S_BEST);
      remove S_BEST from 𝒮;
      split 𝒳 into subsets 𝒳_1, 𝒳_2, ..., based on S_BEST;
      if | subsets | ≤ 1
        return;
      fi
      for each subset 𝒳_k do
          select(𝒮, 𝒳_i);
      od
    fi
  od.
```

4.2 Application to the Biological Case Study

The biological community has built over the years a comprehensive collection of resources to archive experimental data. As molecular biology became a more data intensive field with the advent of DNA microarrays, came the need to store not only measurements but also ancillary annotation describing experimental conditions and set up, thus ensuring a metadata core always shipped with the data set. The

Table 1. A fragment of the input document passed to the taxonomy generation algorithm. Schemes are grouped by common names preceding the semi-colon, e.g. *S1:On Material* refers to schema 1 and *On Material* is the classification.

Process Name	Occurrences	S1:Material	S1:Data	…	S6:in vitro	S6:in vivo	S6:in silico
labeling	390811	1		…	1		
nucleic acid extr.	350267	1		…	1		
hybridization	345671	1		…	1		
feature extr.	267044		1	…			1
bioassay data trans.	176347		1	…			1
grow	116194	1		…		1	
pool	68004	1		…	1		
…	…	…	…	…	…	…	…

sizes of microarray databases (GEO, AE) constitute prime resources for evaluating and testing our approach [2, 1]. The content of ArrayExpress was therefore accessed obtaining data via parallel calls, converting 21000 experiments and associated experimental metadata to ISA-Tab[46, 50]. This step provided a harmonized format in which to represent not just transcriptomic data, but also genomic, proteomic, metabolomic and other classical experiment types.

Given the data sets, the code analyzes the content of these directories to determine the processes and IOs which exist within the experiments, and the number of times they occur. The analysis revealed 61 processes (a small number resulting from the homogeneity of the database where analysis techniques are targeted primarily towards DNA microarrays) with 1,845,089 occurrences and 8,223 properties of the sample of which 3492 were deemed to be IOs with 486,353 occurrences.

Several distinct, empirical but meaningful, categorizations were devised encompassing a number of facets defining the properties of the experimental process, either in terms of its participants or in terms of key process properties. Classifications, based on features such as the nature of process participants, the granularity scale, the nature of experiments and common types a treatments applied in biological experiments were developed. Some classifications were somewhat informed by the overall assumptions ISA model relies upon, where nodes can be either material or data files and where edges are processes acting upon those nodes. Others resulted from applying a small number of axioms discovered through consultations with domain experts. There were 6 classification schemes in all with a total of 23 sub-classifications, these are detailed in figure 3. Table 4.2 shows a snapshot of the input file used in the classification algorithm.

The schemes are not tied to any existing taxonomic tree, may not be orthogonal and/or may be redundant with respect to others. The development of the classifications was not the result of application of any knowledge engineering methods and there is no ontological commitment, although in theory the classification names used could be derived from an ontological framework. The reason for not relying an ontology in the first place was risk mitigation, where use would almost certainly result in an unbalanced tree as occurrence counts for the terms would not be considered, resulting in creation of many glyphs that would never be used.

The creation of the tree shown in Fig. 3 was a largely iterative quest with continuous involvement of domain experts checking to ensure that the classifications assigned were meaningful. From early versions of the classification matrix creation, classifications were removed, added or merged in consultation with those who knew the data domain. For example, in *S2* there were sub-classifications in *Genetic Modification & Labeling* which are types of *Material Combination*. Therefore, these sub-classifications were removed since it would be technically and semantically incorrect to keep them. These early interactions emphasized the importance of having domain experts in the loop, otherwise classifications and subsequent glyphs would not be as well created as they could be.

The algorithm managed to correctly place the classifications where they were expected (checks were made by hand to ensure that the algorithm performed well). Schema *S6* (*in vitro*, *in silico* and *in vivo*) was the best top level classification due to its overall fitness metric of 3.08 with individual metrics found to be: M1 = 1.0; M2 = 1.0; M3 = 0.9; and M4 = 0.18. *S6* was closely followed by *S1* (on material and on

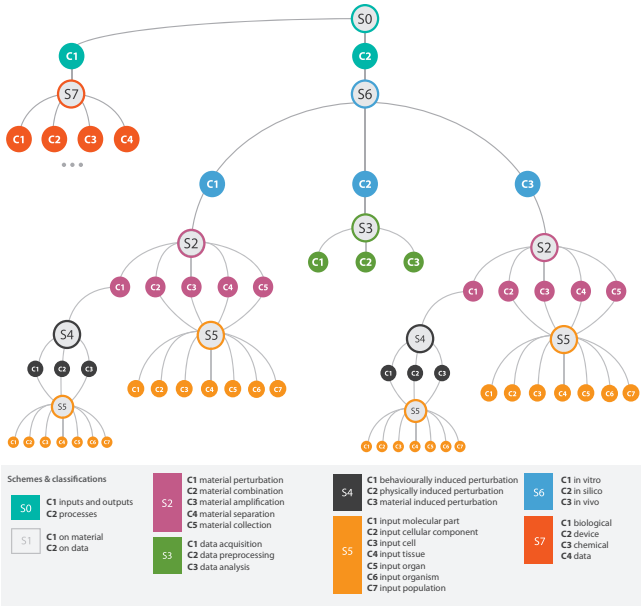


Fig. 3. The classification algorithm arranged the classification schemes in the above order.

data), however after the top level separation, *S1* became redundant as a result of the top level implicitly making a split on data (in silico) and materials (in vitro and in vivo). The algorithm successfully detected this redundancy; hence *S*1 is absent from Fig. 3.

Although the algorithm performed as expected, a possible criticism of the algorithm could be highlighted in the placement of *S5* below *S4*. *S5* could have been placed above *S4* in line with the observation that, for the majority of cases, *S5* was selected over *S4*. The reason *S5* wasn't used to sub-classify C1 of *S4* is due to its number of classifications (7) being greater than that of *S4* (3), therefore *S4* was selected primarily on this metric even though the sub-tree balance of *S5* (0.8) was slightly better than that of *S4* (0.68). To modify this behavior, it was necessary to involve domain experts in refining the tree based on their domain-specific knowledge and case-specific requirements.

5 Visual Encoding

5.1 Perceptual Guidance

A *glyph* is a small visual object composed of a number of *visual channels* which can be used independently as well as constructively to depict attributes of a data record. Glyphs are of a type of visual signs that can make use of visual features of other types of signs, such as icons, indices and symbols.

Although there are several books on sign design (e.g., [6, 4]), they focus on signage in public space, and offer empirical guidance on a large number of issues including standardization, size, location, illumination and so on. Many of these issues are not quite relevant to the need for visualizing the workflows of biological experiments. Here, we draw our design principles mainly from findings in perception, especially in the area of visual search [54, 43].

While the extensive use of signs, icons and pictograms in everyday life reflects their usefulness and effectiveness, several perceptual studies also directly or indirectly confirmed their perceptual and cognitive merits. For example, Franks and Bransford's study on transformation of prototypes [16] suggested that humans can learn to recognize glyphs by rules consciously as well as unconsciously. The presence of iconic memory [53] may facilitate rapid comparison between glyphs in the same display, whereas it is less so for texts.

Guideline on Semantic Relevance. Bertin [7] classified visual channels (which he referred to as retinal variables) into two categories, planar (location) and retinal (size, color, shape, orientation, texture and brightness). Bertin proposed four semantic criteria for determin-

Table 2. Perceptual strength of different visual channels based on levels of organization studied by Bertin[7] and Green[17], 'popout' effect studied extensively in psychology [61, 15, 31, 7, 17, 63, 55, 38, 39], and hierarchy effect studied by Navon [35] and Love [30]. The summary strength of each channel is estimated and it should be considered in conjunction with application-specific conventions and metaphors.

Regions	Visual Channels	Levels of Organization				Popout Effect	Hierarchy Effect	Summary Strength	Convention & Metaphor
		Associative	Selective	Ordered	Quantitative				
a) Main	1. Color	Yes	Yes	Yes		*****	Strong	*****	application specific
	2. Size		Yes	Yes	Yes	***		****	
	3. Shape	Yes	Yes			***		***	
	4. Orientation	Yes	Yes			***		***	
	5.Texture	Yes	Yes	Yes		**		**	
b) Supplementary	6-10 (same as 1-5)						Medium		
	11. Planar	Yes	Yes		Yes	***		***	
c) Interior	Contains (a+b))						Low		

ing the suitability of different channels in representing certain types of information. These semantic criteria are: *associative*, *selective*, *ordered* and *quantitative*. These criteria are important guidelines, though there has been disagreement in the literature as to how individual visual channels are judged. For example, Bertin considered shape as a non-selective variable. Research has shown that shapes such as filled rectangles, circles and triangles do not allow the human visual system to identify one shape from another effectively in a rapid action when they form some global structures [30, 35], (they have poor "pop-out" effect). However, the omission of all shapes as a selective visual channel has been challenged, for example, by [56, 57, 17], who show practice and familiarity can support selectivity with almost any shape.

Guideline on Channel Composition. As a glyph is likely to feature a number of visual channels, the constructive composition may affect how individual channels are perceived. A rich collection of literature on integral and separable dimensions shows that the combined dissimilarity of closely integrated visual channels exhibits Euclidean distance $\sqrt{d_a^2+d_b^2}$ [26, 18], whereas that of separable visual channels exhibits city-block distance d_a+d_b [10, 51]. The latter is more cost-effective than the former in rule-based encoding of multi-faceted concepts, therefore effective glyph design should encompass a non-conflicting set of separable retinal variables.

Guideline on Pop-out Effects. Many classic studies in perception also established the "power" of different visual channels in terms of *pop-out effect* (pre-attentive search), and fixation (during attentive search)[19]. The *pop-out effect* is one which allows identification of a target within a few nanoseconds of initial exposure to the visual search space. A result of several milestone studies focusing on observed response times, the ordering of the four commonly used visual channels follows the consensus: color $\prec$ size $\prec$ shape $\prec$ orientation (e.g.,[61, 44, 48]). The symbol $\prec$ reads as *precedes*. However, the strength of color over the other three channels is generally much more noticeable.

Guideline on Visual Hierarchy. *Visual hierarchy*, with which the environment and objects around us are arranged is a well documented theoretical framework [38, 35, 30, 24, 5]. However, the literature contains a debate over the ways in which the visual system traverses this hierarchy. There are four possible ways: top-down (also called global processing) [35]; bottom-up (also called local processing); middle-out [24]; and salient features (*e.g., edges, points, colors*) [49]. Because glyphs are relatively small in comparison with a whole visualization, we consider at such a "localized level", the top-down and salient features may play more significant roles. The top-down assumption suggests that when consider a glyph in isolation, its global feature will affect visual search more than its local features. Salient features are partly addressed by the pop-out effects.

Based on the above-mentioned perceptual guidance, we considered a generic glyph design template for this work as shown in Fig. 4(a). The template divides a glyph into three regions, namely *main body*, *exterior* and *interior*. In practice, each region can be divided into 2 or more sub-regions (e.g., a twin body glyph or 4 exterior sections), it is convenient to consider the three regions in abstraction. The separation of these regions facilitates the basic separation of visual channels based on the composition guideline, while allowing us to consider them individually according to the hierarchy guideline.

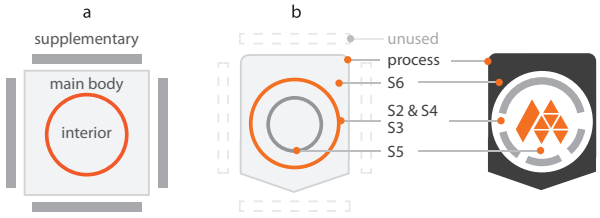


Fig. 4. (a) The glyph template with a main body, an interior region and an exterior region (consisting of 4 sections). (b) Our final design makes use of the main body and interior region. The exterior is reserved for future extension.

In theory, the exterior and interior regions may also be divided into sub-regions in a recursive fashion though in practice this is tightly constrained or discouraged by the very limited display resolution typically available for glyphs. Similar to the design convention for icons, pictograms are normally featured in the interior region. The exterior region may be further divided in four sections (top, bottom, left and right). If glyphs will be connected to form a network or graph (as in this work), the use of these four sections has to take into account the possible incoming and outgoing connections.

Table 2 summarizes relative merits of some of the most commonly-used visual channels in different regions according to Bertin's categorization, pop-out effects and hierarchy effects. We estimated the overall discriminating capacity of each channel by using a summary rating in the penultimate column. We also recognized the importance of the conventions and metaphors in an application. We will discuss visual metaphors further in Section 5.2. We added the last column to highlight the necessity to consider these in a design process.

It is difficult to establish an accurate ranking order of different visual channels by taking all perceptual effects into account in a quantitative manner. The state of the art in perception research is yet to provide all evidence needed for a full and conclusive analysis. Nevertheless, Table 2 can serve a qualitative guidance to glyph designs in this work, providing an ordering of visual channels in parallel with the ordering of categorization schemes discussed in Section 4.

5.2 Mapping Taxonomy to Visual Channels

For each scheme in the taxonomic tree as shown in Fig. 3, we propose a number of design options. Fig. 5 shows some examples of the proposed designs. For example, the first column shows the use of colors to encode the classes of a categorization scheme. The second column shows the use abstract shapes. Some options convey an abstraction from pictorial representations of classes, and in other cases, we try to establish a metaphoric association between a visual channel and a biological categorization.

Metaphoric visual representations enable domain-specific encoding using "natural mapping" [52, 36]. This natural mapping can make it easier for users to infer meaning from the glyph with less effort required to learn and remember them [33]. A recent study showed that visual metaphors can aid memorization of the information depicted in a visualization [8]. However, the same study also showed that visually realistic metaphors (those with a lot of detail) may have a negative

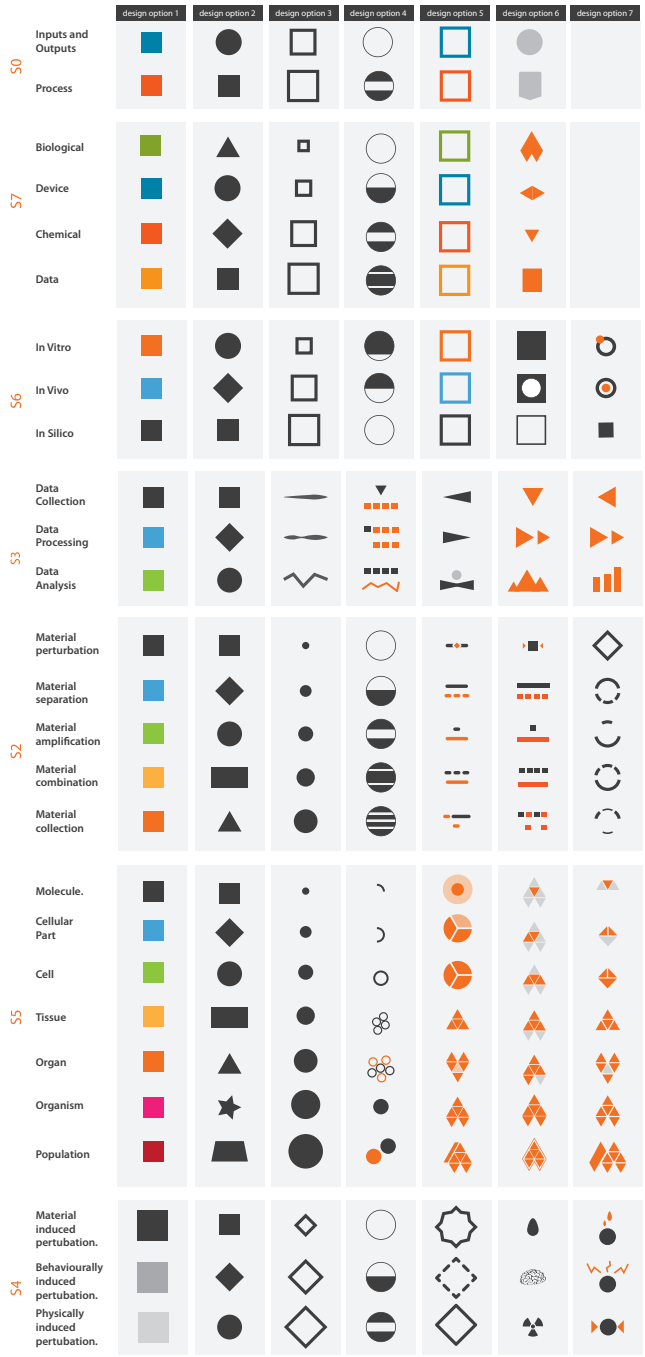


Fig. 5. Experimenting with visual channels: an overview of the various design options available for use in representing the different classifications. Most schemes for IO categorization are not shown due to space restriction.

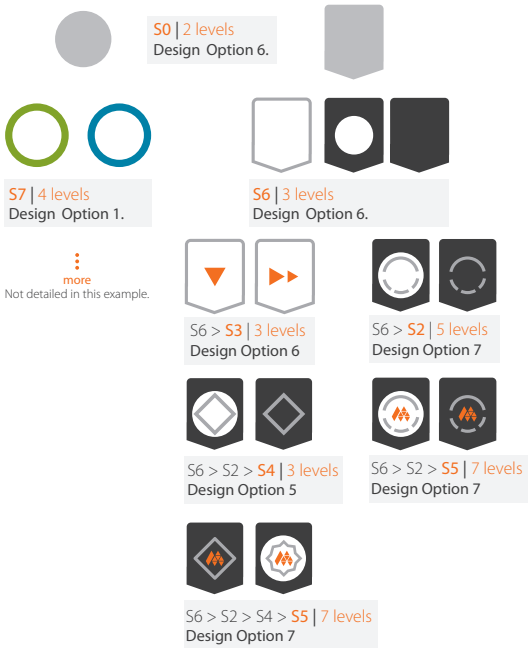


Fig. 6. Formation of the final glyph design. Top level items require the greatest visual power. It is important to be able to distinguish each of the processes based on their parent level in the taxonomy. Distinguishable global shapes provide the difference between IOs (circles) and processes (squares with pointed bottom to indicate directionality).

impact on performance in visual search. Moreover, realistic visual metaphors require a higher pixel resolution, and would lose their discriminating capacity in low resolution conditions.

Based on the design options shown in Fig. 5, we followed the taxonomic tree in Fig. 3 and identified the best option for each scheme in a hierarchical manner. The evaluation criteria include:

- the discriminating capacities of different channels (Table 2);
- metaphoric capacity for aiding learning and remembering;
- potential conflicts, including spatial, perceptual and metaphoric conflicts, with visual channels that have already been assigned to other schemes in the tree;
- encoding costs in terms of requirement for pixel resolution.

This process normally takes a few iterations, during which new design options and new metaphoric abstractions and associations are sometimes proposed.

Based on the hierarchy in Fig. 3, we considered to use color and shape options for S0 (IOs vs. processes) and S7 (four classes of IOs). As introducing 4 main body shapes to encode IOs is not an effective mapping for learning and remembering, we decided to assign outline color of the main body to encode S7, and use two basic shapes, circle and pentagon (a rectangle with a pointer to show workflow direction) to encode S0. Since the main body or the pentagon will only be colored in black or white, S0 also implicitly encoded in using color symbolism, that is, color for IOs and black/white for processes. In effect, S0 is encoded using two visual channels. This redundancy can serve as an error detection in visualization [12].

Fig. 6 shows each of the five schemes for the process subtree, and the design option chosen from Fig. 5. Below we discuss our reasoning for selecting each of the design options for each scheme in the taxonomy.

S6: Process environment. The taxonomic tree suggested a high priority for visual mapping, which is consistent with the domain experts' intuition. We took advantage that black color was not used by S7 for IOs, we assigned a white background to *in silico/in computer* (related to computational processes), and black to *in vivo/in living)* and *in vitro/in glass* (related to materials). Further more, we made use of a shape-based metaphor, fully-filled background for *in vivo* (whole organism), and black background with white cut out for *in vitro* (component of an organism). Together, this was given in Fig. 5 as design option 6. We maintained the overall appearance of the main body of process glyphs in black and white to avoid potential clash with material glyphs.

S2: Types of Material Manipulation. S2 has 5 classes, and we adopted design option 7, which employs visual metaphors that encapsulate strong domain-specific meanings. For example, visual symbol

Fig. 7. Visual representations of the 7 classes in *S5*.

for the *material amplification* class depicts a small segment becoming a large segment.

S4: Type of Experimental Perturbation. S4 defines 3 further subclasses of the *material perturbation* class of S2. Due to the low number of subclasses, we made use of line styles to modify the diamond shape of the *material perturbation*. We made metaphoric association of three line styles as: "solid" line for physically induced perturbation; dash line, a common metaphor for uncertainty and unpredictability, for behaviorally induced perturbation; and wavy line, which is closer to a circle (a visual signature for IOs), for material induced perturbation.

S5: Levels of Material Granularity. This scheme has 7 subclasses (molecule, cellular part, cell, tissue, organ, organism and population), and finding a suitable visual channel was not straightforward. A simplest approach would be to use colors to fill the interior of the glyph, or to use some shape-based encoding. After consulting the domain experts, these two options were ruled out. In fact, domain experts preferred some pictograms to represent the 7 subclasses. After considering a number of more realistic drawings, we found that it was not easy to create realistic representations that can differentiate all subclasses (e.g., cellular part vs. cell; organ vs. organism). We designed a special set of icons as shown in Fig. 7 with three visual channels to aid learning, memorization and recognition. The first visual channel is the overall shape and orientation. The second visual channel is metaphoric abstraction and association. For example, the shape of cell part indicates a portion of a cell. Tissue is associated with a patch, organ with an abstract heart shape, organism with an abstract human, population with two abstract humans. The third visual channel is the number of orange sub-shapes, representing the levels 1-7.

S3: Types of Data Manipulation. S3 defines a 3 further subclasses of the *in silico* class of S6. We use three abstract pictograms to represent data capture, processing and analysis. The encoding makes use of the difference in overall shape, orientation, and number of triangles to aid learning, memorization and recognition. Note that these shapes will not be confused with those for S5 as the white background of the main body provides a distinct context of computing rather than IOs.

Following mapping of all schemes to design options, creation of all glyphs is a straightforward process. Fig. 8 shows all variations of the glyphs associated with the process sub-tree.

We introduced a "crush" test for the designed glyphs by scaling it to different pixel resolutions. Fig. 9 shows some example glyphs display at varying resolutions. One can comfortably see all details at the 40×40 level. At the 10×10 level, one can observe the visual signature of *in vivo* and *in vitro*. Even at the lowest level (5×5), one can still differentiate the two glyphs.

6 Workflow Visualization and ISA Integration

In order to visualize workflows of biological experiments, we had to address the following technical issues: 1) mapping from a name in the metadata to a glyph; 2) creating a workflow visualization with both node placement and connection display; 3) developing a prototype tool for a practical environment such as the ISA tools framework.

The mapping from concept name is achieved by a look-up table which is built from the matrix used in formulating the taxonomic tree and implemented as a tab delimited file. Each concept name is mapped to a text tag encoding the traversal path from the root of the tree to the leaf node corresponding to the name. The tag includes identifiers of the schemes and classes encountered. With the path, a glyph can be constructed dynamically from the pre-defined visual mapping as described in the previous section. The look-up table also enables storage of pre-rendered glyphs in an image format.

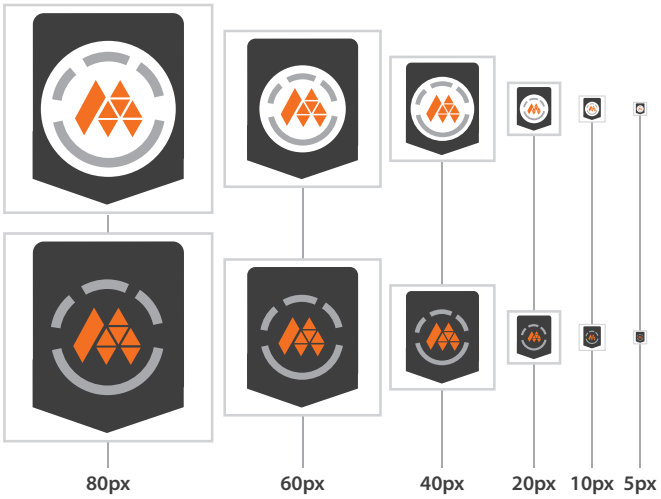


Fig. 9. A "crush" test of glyphs to evaluate the discriminating capacity of various visual channels at different resolutions.

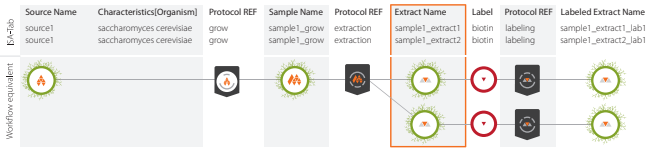


Fig. 10. A workflow is recorded in text form within the ISA-Tab format. Our software translates it to glyph-based visualization. A branching event is automatically detected during the translation.

To generate the workflow, we made use of the layout algorithm available within the Prefuse visualization framework [20]. This framework also brings with it functionality such as panning, zooming and filtering which bring more interactivity to the user and making navigation through large collections of workflows easier. The only requirement for use of Prefuse was a minimal amount of Java code to create the user interface coupled with creation of an XML file format native to Prefuse for representation of the tree structure. This XML format is configurable, we have configured it to contain: *node type* (e.g., process), *node name* (e.g., labeling) and an image to be rendered, which is assigned using a look-up operation through the above mentioned tab delimited mapping file. The order of the XML elements within this file has direct implications on the order these elements are displayed in. Within the ISA-Tab format, there is an implicit time element found in the ordering of the columns in the study sample and assay files. This can be used to construct the XML elements through near direct mappings of processes and IOs, a workflow which is illustrated in Fig. 10. Recognizing branching events is an important part of workflow visualization as illustrated in 10. In software, these branch events can be identified when the preceding nodes of a process have the same names while the succeeding output nodes are different. Fig. 10 highlights one such case, where branching occurs after extraction of 3 materials from one sample.

Our software reads text-based ISA-Tab files as illustrated to create the XML notations required by Prefuse for rendering the experiment workflow illustrated in Fig. 1(b).

The software is implemented as a Java application capable of processing ISA-Tab files to create experimental workflows using glyphs. For the purposes of broad dissemination in the near future, we have integrated the workflow visualization directly inside ISAcreator, a Java desktop application for creating and editing ISA-Tab files. Furthermore, the standalone workflow visualization shown in figures 11a and b makes use of the same ISA-Tab parser as available within ISAcre-

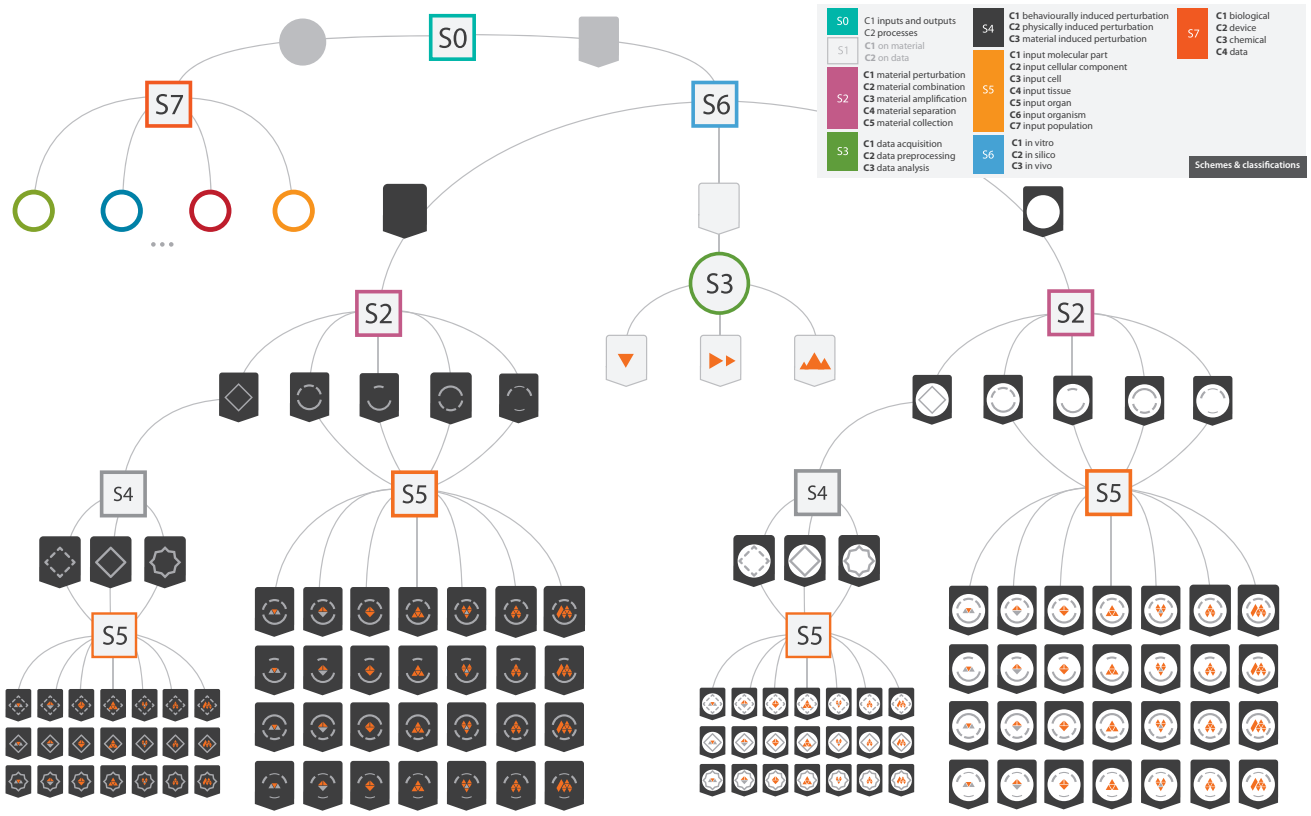


Fig. 8. Overview of the taxonomy based glyph designs in the context of biodomain.

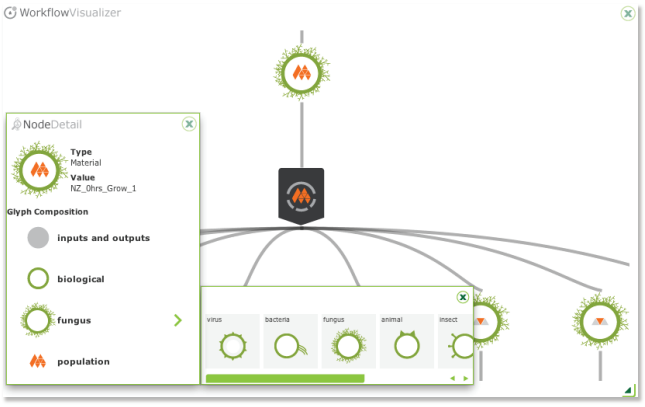


Fig. 11. User can interact with a workflow to view the text descriptions of individual glyphs in pop-up windows, which are also dynamic legends showing how the concept is categorized and how the corresponding glyph is decomposed into different elementary visual representations. Users can also use such a legend to find out other classes in a categorization scheme.

ator. The integration involved the development of user interface elements for ISAcreator software to access the workflow visualization. From the spreadsheet view, users may highlight rows (assays), right click and select an option named "View workflow for assays" to visualize the entire processing pipeline for the corresponding samples as shown in Fig. 1(b).

7 Conclusions and Future Work

Although it was established in psychology that people can learn rules of glyph encoding consciously as well as unconsciously [16], we do not expect users to learn and remember these glyphs without help. We thus allow users to find detailed text descriptions in pop-up windows interactively. As shown in Fig. 11, not only do these windows provide names of processes and IOs, they also serve as dynamic legends, showing how each glyph is decomposed into different elements corresponding to categorization schemes. One can also select a component to view all classes in a scheme.

In this manuscript, we presented a systematic approach for glyph design by using taxonomy as a guide. We demonstrated our approach by analyzing the content of a biological database and showed how our tree-building algorithm could be used to take a large collection of concepts and generate a taxonomic tree, which provides an ordering of a set of categorization schemes (i.e., attribute dimensions). This enabled us to draw a parallel between the ordering of attribute dimensions with the ordering of visual channels compiled from the perception and visualization literature. We involved domain experts in refining the tree as well as in creating metaphoric abstraction and association for glyphs. This work was followed by development of a prototype tool for glyph-based visualization of experimental design workflows as found in biology. The prototype is integrated in the ISA suite of tools to be disseminated to users.

We plan to further the present work to include the provision of "macros" in workflows in order to make graphs more compact and create dynamic web-based rendering of the workflows using vector graphics. We also intend to carry out field-based evaluation through ISAcommons, the user community of ISA-tools framework. Like all potential diagrammatic schemes, we expect that it will take many iterations before it can become a standard. Nevertheless, the development of an online visualization tool will make such a process more efficient.

Acknowledgments

This work was supported by funding from the BBSRC and NERC, grants BB/I000917/1 and BB/I025840/1.

REFERENCES

[1] Arrayexpress. http://www.ebi.ac.uk/arrayexpress/.

[2] Gene expression omnibus (geo). http://www.ncbi.nlm.nih.gov/geo/.

[3] Graphviz. http://graphviz.org/.

[4] R. Abdullah and R. Hümber. *Pictograms, Icons & Signs: A guide to Information Graphics*. Thames & Hudson, 2006.

[5] M. Bar. Visual objects in context. *Nature reviews. Neuroscience*, 5(8):617–629, 2004.

[6] P. Barker and J. Fraser. Sign design guide: a guide to inclusive signage. *JMU and Sign Design Society*, 2000.

[7] J. Bertin. *Semiology of graphics: diagrams, networks, maps*. University of Wisconsin press, 1983.

[8] R. Borgo, A. Abdul-Rahman, F. Mohamed, P. W. Grant, I. Reppa, L. Floridi, and M. Chen. An empirical study on using visual embellishments in visualization. *to appear in IEEE Transactions on Visualization and Computer Graphics*, 2012.

[9] R. Bourqui and M. Westenberg. Visualizing temporal dynamics at the genomic and metabolic level. *Proc. 13th Int. Conf. Information Visualisation*, pages 317–322, 2009.

[10] B. Burns, B. E. Shepp, D. McDonough, and W. K. Wiener-Ehrlich. The relation between stimulus analyzability and perceived dimensional structure. *The Psychology of Learning and Motivation*, pages 77–115, 1978.

[11] Y. Cai, M. L. Wilson, and J. Peccoud. Genocad for igem: a grammatical approach to the design of standard-compliant constructs. *Nucleic Acids Res*, 38(8):2637–44, May 2010.

[12] M. Chen and H. Jänicke. An information-theoretic framework for visualization. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1206–1215, 2010.

[13] M. Chuah and S. Eick. Information rich glyphs for software management data. *IEEE Computer Graphics and Applications*, 18:24–29, 1998.

[14] T. Czauderna, C. Klukas, and F. Schreiber. Editing, validating and translating of sbgn maps. *Bioinformatics*, 26(18):2340–2341, 2010.

[15] J. Duncan and G. Humphreys. Visual search and stimulus similarity. *Psychological review*, 96(3):433–491, 1989.

[16] J. Franks and J. Bransford. Abstraction of visual patterns. *Journal of experimental psychology*, 90(1):65–139, 1971.

[17] M. Green. Toward a perceptual science of multidimensional data visualization: Bertin and beyond. *ERGO/GERO Human Factors Science*, 1998.

[18] S. Handelt and S. Imai. The free classification of analyzable and unanalyzable stimuli. *Attention, Perception, & Psychophysics*, 12(1):108–224, 1972.

[19] C. G. Healey and J. T. Enns. Attention and visual memory attention and visual memory in visualization and computer graphics. *IEEE Transactions on Visualization and Computer Graphics*, 18(7):1170–1188, 2011.

[20] J. Heer, S. K. Card, and J. A. Landay. prefuse: A toolkit for interactive information visualization. In *Proc. ACM CHI*, pages 421–430, 2005.

[21] K. Hemenway. Psychological issues in the use of icons in command menus. In *Proc. ACM CHI*, pages 20–23, 1982.

[22] B. H. Junker, C. Klukas, and F. Schreiber. Vanted: A system for advanced data analysis and visualization in the context of biological networks. *BMC Bioinformatics*, 2006.

[23] A. Karve and M. Gleicher. Glyph-based overviews of large datasets in structural bioinformatics. In *Proc. 11th Int. Conf. Information Visualization, Supplements*, pages 1–6, 2007.

[24] R. Kinchla and J. Wolfe. The order of visual processing: "top-down," "bottom-up", or "middle-out". *Perception & psychophysics*, 25(3), 1979.

[25] G. Kindlmann and C.-F. Westin. Diffusion tensor visualization with glyph packing. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1329–1335, 2006.

[26] D. Krantz and A. Tversky. Similarity of rectangles: An analysis of subjective dimensions. *Journal of Mathematical Psychology*, 12(1), 1975.

[27] R. Krishnapuram and K. Kummamuru. Automatic taxonomy generation: Issues and possibilities. In *Proc. IFSA 2003, LNCS*, pages 184–195, 2003.

[28] N. Le Novère, M. Hucka, H. Mi, S. Moodie, and *et al*. The systems biology graphical notation. *Nat Biotechnol*, 27(8), 2009.

[29] J. P. Lewis and R. Rosenholtz. VisualIDs: Automatic distinctive icons for desktop interfaces. *ACM Transactions on Graphics*, 23(3):416–423, 2004.

[30] B. Love, J. Rouder, and E. Wisniewski. A structural account of global and local processing. *Cognitive psychology*, 38(2):291–607, 1999.

[31] S. Luck and S. Hillyard. Electrophysiological correlates of feature analysis during visual search. *Psychophysiology*, 31(3):291–308, 1994.

[32] E. Maguire. Taxonomy in biology and visualization. http: //isatab.sourceforge.net/docs/publications/Taxonomy.pdf, Jan 2012.

[33] S. McDougall, O. De Bruijn, and M. Curry. Exploring the effects of icon characteristics on user performance: The role of icon concreteness, complexity, and distinctiveness. *Journal of Experimental Psychology: Applied*, 6(4), 2000.

[34] P. Muter and D. Mayson. The role of graphics in item selection from menus. *Behaviour and Information Technology*, 5:89–95, 1986.

[35] D. Navon. Forest before trees: The precedence of global features in visual perception. *Cognitive psychology*, 9(3):353–736, 1977.

[36] D. A. Norman. *The Design of Everyday Things*. Basic Books, 2002.

[37] H. Ogata, S. Goto, K. Sato, W. Fujibuchi, H. Bono, and M. Kanehisa. Kegg: Kyoto encyclopedia of genes and genomes. *NAR*, 27(1), 1999.

[38] S. Palmer. Hierarchical structure in perceptual representation. *Cognitive Psychology*, 9(4):441–915, 1977.

[39] D. Parkhurst, K. Law, and E. Niebur. Modeling the role of salience in the allocation of overt visual attention. *Vision research*, 42(1), 2002.

[40] J. K. Patel and C. B. Read. *Handbook of the Normal Distribution*. Marcel Dekker, 2nd edition, 1996.

[41] J. W. Pellegrino, R. R. Rosinski, H. L. Chiesi, and A. Siegel. Picture-word differences in decision latency: An analysis of single and dual memory models. *Memory and Cognition*, 5:383–396, 1977.

[42] F. Post, T. Walsum, F. Post, and D. Silver. Iconic techniques for feature visualization. *Proc. IEEE Visualization*, pages 288–295, 1995.

[43] P. Quinlan. Visual feature integration theory: past, present, and future. *Psychological bulletin*, 129(5):643–716, 2003.

[44] P. Quinlan and G. Humphreys. Visual search for targets defined by combinations of color, shape, and size: an examination of the task constraints on feature and conjunction searches. *Perception & psychophysics*, 41(5):455–527, 1987.

[45] W. Ribarsky, E. Ayers, and J. Eble. Glyphmaker: creating customized visualizations of complex data. *Computer*, 27:57–64, 1994.

[46] P. Rocca-Serra, E. Maguire, S.-A. Sansone, and *et al*. Isa software suite: supporting standards-compliant experimental annotation and enabling curation at the community level. *Bioinformatics*, 26(18), 2010.

[47] Rohrer and *et al*. The shape of shakespeare: visualizing text using implicit surfaces. In *Proc. IEEE Visualization*, pages 121–129, 1998.

[48] T. Ropinski, S. Oeltze, and B. Preim. Survey of glyph-based visualization techniques for spatial multivariate medical data. *Computers & Graphics*, 35(2):392 – 401, 2011.

[49] D. E. Rumelhart. A multicomponent theory of the perception of briefly exposed visual displays. *Journal of Math. Psych.*, 7(2):191–218, 1970.

[50] S.-A. Sansone, P. Rocca-Serra, D. Field, E. Maguire, and *et al*. Toward interoperable bioscience data. *Nat Genet*, 44(2):121–6, 2012.

[51] R. Shepard. Attention and the metric structure of the stimulus space. *Journal of Mathematical Psychology*, 1(1):54–141, 1964.

[52] H. Siirtola. The effect of data-relatedness in interactive glyphs. *In Proc. 9th Int. Conf. Information Visualization*, 2002.

[53] G. Sperling. The information available in brief visual presentations. *Psychological Monographs: General and Applied*, 74(11):1–29, 1960.

[54] K. T. Spoehr and S. W. Lehmkuhle. *Visual Information Processing*. W. H. Freeman & Company, 1982.

[55] A. Treisman. Focused attention in the perception and retrieval of multidimensional stimuli. *Attention, Perception, & Psychophysics*, 22(1), 1977.

[56] A. Treisman and S. Gormican. Feature analysis in early vision: evidence from search asymmetries. *Psychol Rev*, 95(1):15–48, Jan 1988.

[57] Q. Wang, P. Cavanagh, and M. Green. Familiarity and pop-out in visual search. *Percept Psychophys*, 56(5):495–500, Nov 1994.

[58] M. O. Ward. Multivariate data glyphs: Principles and practice. In *Handbook of Data Visualization*, pages 179–198. 2008.

[59] C. Ware. *Information Visualization: Perception for Design*. Morgan Kaufmann, 2004.

[60] S. Wiedenbeck. The use of icons and labels in an end user application program: an empirical study of learning and retention. *Behavior and Information Technology*, 18(2):68–82, 1999.

[61] L. Williams. The effects of target specification on objects fixated during visual search. *Acta psychologica*, 27:355–415, 1967.

[62] C. Wittenbrink, A. Pang, and S. Lodha. Glyphs for visualizing uncertainty in vector field. *IEEE Transactions on Visualization and Computer Graphics*, 2:266–279, 1996.

[63] J. Wolfe, K. Cave, and S. Franzel. Guided search: an alternative to the feature integration model for visual search. *Journal of experimental psychology. Human perception and performance*, 15(3):419–452, 1989.

C Copy of [BKC+13]:

Glyph-based Visualization: Foundations, Design Guidelines, Techniques and Applications

Glyph-based Visualization: Foundations, Design Guidelines, Techniques and Applications

R. Borgo[1], J. Kehrer[2], D. H. S. Chung[1], E. Maguire[3], R. S. Laramee[1], H. Hauser[4], M. Ward[5] and M. Chen[3]

[1] Swansea University, UK; [2] University of Bergen and Vienna University of Technology, Austria; [3] University of Oxford, UK; [4] University of Bergen, Norway; [5] Worcester Polytechnic Institute, USA

Abstract

This state of the art report focuses on glyph-based visualization, a common form of visual design where a data set is depicted by a collection of visual objects referred to as glyphs. Its major strength is that patterns of multivariate data involving more than two attribute dimensions can often be more readily perceived in the context of a spatial relationship, whereas many techniques for spatial data such as direct volume rendering find difficult to depict with multivariate or multi-field data, and many techniques for non-spatial data such as parallel coordinates are less able to convey spatial relationships encoded in the data. This report fills several major gaps in the literature, drawing the link between the fundamental concepts in semiotics and the broad spectrum of glyph-based visualization, reviewing existing design guidelines and implementation techniques, and surveying the use of glyph-based visualization in many applications.

1. Introduction

Glyph-based visualization is a common form of visual design where a data set is depicted by a collection of visual objects referred to as *glyphs*. In a narrow interpretation,

(a.1) a glyph is a small independent visual object that depicts attributes of a data record;
(a.2) glyphs are discretely placed in a display space; and
(a.3) glyphs are a type of *visual sign* but differ from other types of signs such as *icons*, *indices* and *symbols*.

In a broad interpretation,

(b.1) a glyph is a small visual object that can be used independently and constructively to depict attributes of a data record or the composition of a set of data records;
(b.2) each glyph can be placed independently from others, while in some cases, glyphs can be spatially connected to convey the topological relationships between data records or geometric continuity of the underlying data space; and
(b.3) glyphs are a type of *visual sign* that can make use of visual features of other types of signs such as *icons*, *indices* and *symbols*.

In many applications, the spatial location of each glyph is pre-determined by the underlying spatial structure encoded in the data, such as a map in geo-information visualization, or a volumetric field in diffusion-tensor imaging. In other applications, the spatial location represents the result of a visual mapping from non-spatial information, such as the temporal dimension and semantic grouping of data records.

While glyphs are a form of illustrative graphics and visualization, fundamentally they are dictionary-based encoding schemes. Historically, many of such schemes (e.g., maritime semaphore and signal flags) have made indispensable contributions around the world. Technically, dictionary-based encoding has shown great merits in text compression and image compression. In the era of data deluge, one cannot help to contemplate the cost-effectiveness of using glyph-based visualization in many applications, and the long-term potential of evolving glyph-based encoding schemes into a common visualization language.

The design of glyphs can make use of many different *visual channels* such as shape, colour, texture, size, orientation, aspect ratio or curvature, enabling the depiction of multi-dimensional data attributes. Meanwhile, glyphs are normally recognisable individually, offering a means of visual fusion in multi-field visualization. Similar to most types of visual signs, a specific design of a glyph set is fundamentally a visual coding scheme. Like all coding schemes, a well-designed glyph-based visualization can facilitate efficient and effective information encoding and visual communication. As a type of sign, a glyph is a stimulus pattern that has meanings, which can potentially attract greater attention and stimulate more cognitive activity during visualization than other forms of visual design. In dealing with the ever-increasing problem of data deluge, it is a technique that is not to be overlooked.

In the literature of visualization, there have been a few major surveys related to glyph-based visualization. The survey by Ward [War08] provides a technical framework for glyph-based visualization, covering aspects of visual mapping and layout methods, as well as addressing important issues such as bias in mapping and interpretation. Ropinski et al. [RP08, ROP11] present an in-depth survey on the use of glyph-based visualization for spatial multivariate medical data. Lie et al. [LKH09] describe a general pipeline for the glyph-based visualization of scientific data in 3D along with design guidelines such as the orthogonality of individual attribute mappings. Because glyphs are commonly used in vector field visualization, they have been discussed and compared with other forms of visualization in a collection of surveys on flow visualization [PVH*03, LHD*04, PL09]. However, there is a need to build on these surveys by taking a holistic overview of glyph-based visualization in terms of the fundamental concepts and theories, design guidelines, algorithms and techniques and applications. In particular, this survey is intended to address some noticeable gaps in the literature by:

- systematically examining the extensively rich collection of theories in semiotics, perception and cognition; and identifying their relevance to glyph-based visualization;
- categorizing the technical methods for glyph-based visualization in the scopes of both narrow and broad interpretations, opening up the design space for future technical advances; and
- surveying a large collection of applications where glyph-based visualization has already made an impact.

The survey is organized as follows: Section 2 examines the studies of signs in philosophy, language studies and psychology. We draw fundamental understanding from these studies in order to establish formal definitions of glyphs and ways for classifying them. Section 3 surveys formal design guidelines, mapping techniques and layout algorithms and rendering methods that have been used in practice. Section 4, examines a number of application areas where glyph-based visualization has been deployed. In particular, it describes the benefits brought by glyph-based visualization. Section 5 summarizes the findings that have emerged during the compilation of this survey and proposes new interesting research avenues.

2. History and Related Concepts

The term *glyph* is originated from Greek word, *glyphē*, meaning carving. Since the 16th century, its uses in English have been much associated with etymology, archaeology, topography and graphonomics. Although its contemporary use in the context of multivariate visualization may seem rather different, they share many interesting attributes, such as being "small", being "visual", having "meaning", requiring "learning", and often being "metaphoric". It is thus interesting to study briefly the related history and concepts.

2.1. A brief history of the study of signs

Signs in terms of indices, icons and symbols (Figure 1) are all different aspects of a similar unit of knowledge representation, which has been used as a fundamental concept in trade, commerce and industry from early days to present. Symbolism has played an important part in the development of human culture, especially as a form of communication. The Paleolithic Age, around 18,000 BC, has given us hundreds of examples in the form of cave paintings. The Neolithic Age instead provides the first forms of pre-writing symbols used for communication: the Petroglyphs, images incised in rock *petra* (meaning "stone") + *glyphein* (meaning "to carve") . Tribal societies continue to use this form of symbolic writing even in current times.

An interesting aspect of petroglyphs is their similarity across different continents; the commonality of styles strengthens the hypotheses that human conceptual system is symbolic in nature as investigated by Jungian psychology and early works from Mircea Eliade [EM91]. Psychophysical studies have demonstrated how recurrent geometric patterns (*form constants*) in petroglyphs and cave paintings are "hard-wired" into the human brain. Petroglyphs are ancestors to pictograms (or pictograph) symbolic representations restricted not just to objects but also places, activities and structured concepts. Ideograms (or ideograph) are graphical symbols analogous to pictograms but believed to have appeared later and with the main intent of representing "ideas"; contemporary examples of ideograms can be found in wayfinding signage as well as technical notations such as arabic numerals, mathematical notations or binary systems, which maintain the same meaning despite the difference in language and environment. Pictograms and ideograms are at the base of early written symbols such as cuneiforms and hieroglyphs to sophisticated logographic writing system such as the ones developed in Chinese and Eastern cultures. A logogram (or logograph) is defined as a "grapheme" the fundamental unit of a written language (as opposed to phoneme the fundamental unit of a spoken language). It can represent either a single letter or a morpheme, the smallest meaningful unit in the grammar of a language (e.g. a whole word or concept). The Cuneiform writing system for example, employed signs to represent numbers, things, words, and their phonetics. Egyptian hieroglyphs contained a combination of logographic, alphabetic, and ideographic elements, consisting mainly of three kinds of glyphs: phonetic glyphs, including single-consonant characters that functioned like an alphabet; logographs, representing morphemes, and ideograms, which narrowed down the meaning of a logographic or phonetic word. Chinese characters instead are derived directly from individual pictograms or combinations of pictograms and phonetic signs and represents logograms used in writing Chinese, Japanese and Korean.

Examples of pictograms can be easily found today. Interesting examples are the Pub and Inn signs found in Eng-

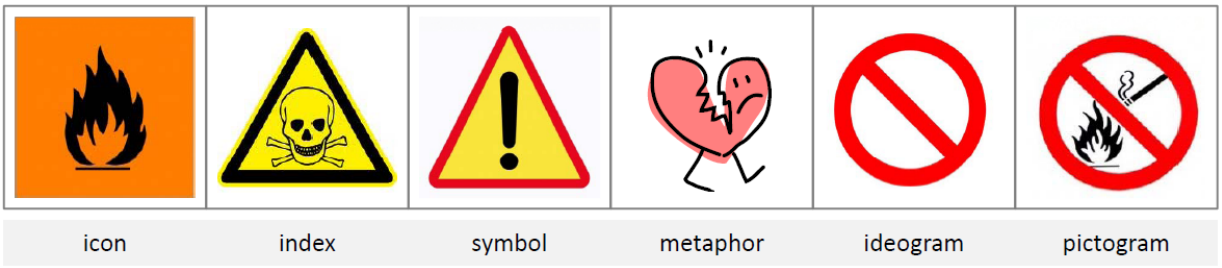


Figure 1: In philosophy, language studies and psychology, signs may take one of the three forms, icon, index and symbol. In many contexts, terms such as visual metaphor, ideogram and pictogram are also used to denote subclasses of signs.

land, Europe and North America. After an edict from King Richard II in 1393 that required all alehouses to post a sign they soon became a method of identifying and promoting themselves to the official ale tasters and the public. These signs still remain a tradition often exposing creative and unusual but always metaphoric. The use of symbols and signs has traversed human history for generations, due to their cross-cultural expressive power. Signs and symbols are fundamental means for communication transcending cultural boundaries. With the advent of the computer era, icons have become one of the most popular means of conveying messages. In the early 1980s the CHI community [BSG89, Bly82, Gay89] investigated the use of sounds in associations with visual display to create a new type of multisensory signs: the "earcons". Today the use of icons, with added sophisticated features such as animations and sounds, is now pervasive throughout most media platforms. As highlighted by Marcus [Mar03] specialised communities such as health and medicine, finance and banking, travel and transportation, and education and training, already possess widespread and sophisticated proprietary visual sign systems. The power of expression inherent to visual sign systems is appealing to media, technology and information visualization alike. The challenge relies on the development of well-designed sign systems.

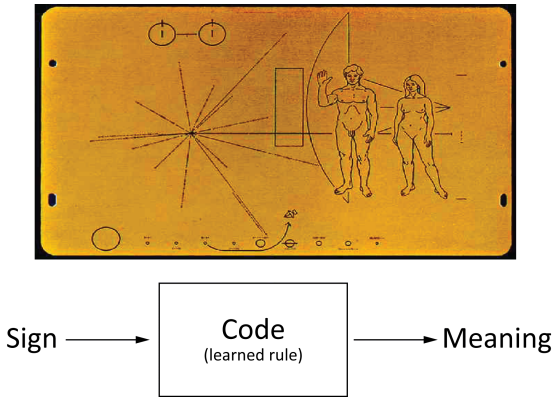


Figure 2: The Pioneer 10 Spacecraft 1972 Plaque.

2.2. Functional Space

According to Peirce [Pei02]'s theory of signs all modes of thinking depend on signs. Signs act as mediators between the external world of objects and the internal world of ideas. A sign in itself is a *stimulus pattern* associated with a *meaning*. Depending on *how* the meaning is associated with the pattern (or object) a sign can be classified as either an icon, an index or a symbol. The icon, index and symbol triad represents the different relationship between the sign and its object. Icons (such as pictures, images, models, or diagrams) represent a sign that itself resembles the qualities of the object it stands for (*physical correlation*). Indexes are defined by some sensory feature (such as a clock, thermometer, fuel gauge, or medical symptom) and therefore represent a sign which demonstrates the influence of its object (*space and time correlation*). Symbols (such as a trophy, medal, receipt, diploma, monument, word, phrase, or sentence) represent a sign which is interpreted as a reference to its object. For this reason, symbols are the only type of sign which do not require any physical, space or time correlation between the sign and its meaning (*metaphysical correlation*).

Codes provide the framework within which signs assume a meaning. A symbol, for example, is a sign where the function is a conventional rule (or coding) and is dependent only on a process of interpretation (Figure 2).

2.2.1. Icons

The functional domain of icons is comprised of: images, metaphors and diagrams. These three items all share topological similarity with the object they are related. Images share sensory qualities, diagrams share relational and structural qualities, while metaphors elicit the representative character of an object by building a parallelism with something else [JL02]. The typology of signs can be described based on the different ways a sign refers to its object [PB55]. Indices require the existence of the object they are a sign of, symbols require an interpreter; while icons require neither object nor interpreter. A Euclidean diagram for example, is made up of streaks of pencil lead that represent a geometric line even though the latter "has no existence" [Pei02].

2.2.2. Indices

The functional domain of indices is comprised of: tracks, symptoms and designations [JL02]. The three types of index represent abstractions that rely on a physical cause/effect relation which is not necessarily simultaneous with the object to which they relate to. Despite simultaneously not being a constraint, an index cannot be a sign without its object (e.g smoke is a symbol/sign of fire).

2.2.3. Symbols

The functional domain of symbols is comprised of all abstractions which rely on a code conventionally used in order to determine meaning. Examples of symbols are languages, mathematical symbols and alphanumeric characters on a computer keyboard. Symbols as signs need an interpreter but do not require any space or time correlation with the object they are a sign of, therefore a symbol represents the only type of sign which: a) can be easily removed from its context; and b) is closely associated with large sets of other words.

2.2.4. Codes

The Pioneer 10 plaque (Figure 2) represents an attempt at communication with alien beings via a "pictorial message" including all three type of signs previously described (e.g. icons, indices and symbols) and it is an exemplar testimony of the importance of what semioticians call codes. Coding is one of the fundamental concepts in semiotics and represents a deterministic functional relation between two sets of entities, namely: a signifier and a signified. Reading an image, like the reception of any other message, is dependent on prior knowledge of possibilities (signifier); we can only recognise what we know (signified). It is this information alone that enables us to separate the code from the message. Related to sign, it is possible to distinguish between three main kind of codes [Cha02]: social codes, textual codes and interpretative codes.

2.2.4.1. Social Codes. All semiotic codes can be broadly classified as social codes, however within our classification we refer to social code in their narrow sense concerning implicit or explicit social agreements and behaviours as in:

- verbal language: phonological, syntactical, lexical, prosodic and paralinguistic subcodes;
- bodily codes: bodily contact, proximity, physical orientation, appearance, facial expression, gaze, head nods, gestures and posture;
- commodity codes: fashion, clothing and cars;
- behavioural codes: protocols, rituals, role-playing and games.

2.2.4.2. Textual Codes. Next to social codes and interpretative codes, textual codes represent one of the major groups of codes. According to Chandler's classification [Cha02], textual codes relate to our knowledge and often act as vehicles to represent reality (representational codes). Examples are:

- scientific codes: including mathematics;
- aesthetic codes: within the various expressive arts (poetry, drama, painting, sculpture, music) and currents (classicism, romanticism, realism);
- genre, rhetorical and stylistic codes: narrative (plot, character, action, dialogue, setting), exposition, argument and so on;
- mass media codes: photography, television, film, radio, newspaper and magazine codes, both technical and conventional (including format).

2.2.4.3. Interpretative Codes. Interpretative codes are perhaps the more interesting as they include:

- ideological codes: individualism, capitalism, liberalism, conservatism, feminism, materialism, consumerism and populism;
- perceptual codes: visual perception.

Perception forms an integral part of the interpretation process. As a semiotic code, perception involves the ability to decode a message presented in a representational form (e.g. a sign) and as such involves a learning process based on the influence of culture and context. In Section 3 we discuss design guidelines that can be taken into consideration to aid the creation of glyphs with attributes making best use of human perception.

A code is a system of syntactic, semantic and behavioural elements which must respond to three basic principles: coherence, homogeneity, and systematicity. In a communicational framework a code is significant if given a message, heterogeneous in nature, it assumes its specificity when transmitted *through* the code. In the context of visual representation the importance of proper coding is therefore self-explicative. Eco [Eco79] distinguishes between "signification" and "communication". Signification is seen as the semiotic event whereby a sign "stands for" something; communication instead is seen as the transmission of information from a source to a destination. In this context codes establish rules for systems of signification and communication is made possible by the existence of a code, or by a system of signification. Without a code or a system of signification, there is no set of rules to determine how the expression of signs is to be correlated with their content.

2.3. Theoretic Frameworks

Whilst semiotics is often encountered in the form of textual analysis, it also involves studying representations and the "reality" always involves representation. Semiosis was first proposed as a term by Charles Sanders Peirce and subsequently expanded by Eco [Eco79] to designate the process by which a culture produces signs and/or attributes specific

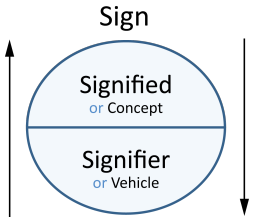


Figure 3: The Dyadic Model of the Sign Notion of Ferdinand de Saussure [SBSR83].

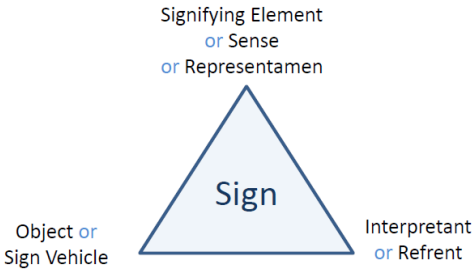


Figure 4: The Structure of the Sign Notion (Triadic Model) of Charles Sander Peirce [PB55].

meanings to signs. In modern semiotics there are two principal models of signs, the dyadic model due to Ferdinand de Saussure [SBSR83], and the triadic model due to Charles Peirce [PB55].

2.3.1. Semiotic Models: Diadic and Triadic

In the Dyadic Model (Figure 3) introduced by Ferdinand de Saussure [SBSR83] a sign is composed of the signifier (the sound pattern of a word, either in mental projection - as when we silently recite lines from a poem to ourselves - or in actual, physical realization as part of a speech act), and the signified (the concept or meaning of the word).

With its Triadic Model (Figure 4), Peirce [PB55] viewed the symbol/index/icon triad as "the most fundamental division of signs", and the majority of semioticians continue to agree [Joh88]. Peirce thus defines "semiosis" as the process by which representations of objects function as signs. Semiosis is a process of cooperation between signs, their objects, and their "interpretants" (i.e. their mental representations). "Semiotic" (i.e. the science of signs) is the study of semiosis and is an inquiry into the conditions which are necessary in order for representations of objects to function as signs.

2.3.2. Semiotic Systems: Algebra

According to Saussure [SBSR83] signs are always part of a formal system with a specific structure and relations. In its Semiotic Algebra Goguen [Gog03] devises a system to

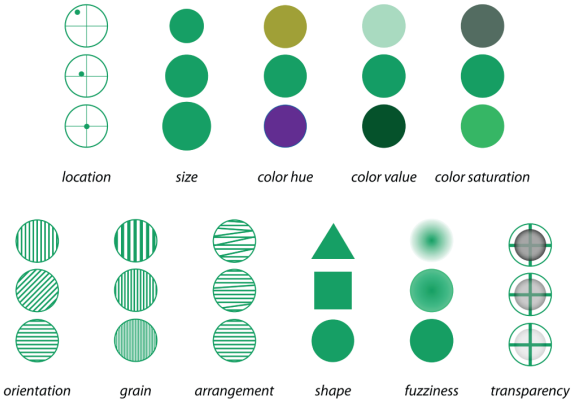


Figure 5: Visual Variables [Mac04].

capture the systematic structure of a sign. In Semiotic Algebra a sign is always divisible into subparts called *sorts* (e.g., colour, location, size). Sorts may have a hierarchical structure with relationship such as inheritance or partial ordering between subsorts. Signs can be composed into more complex signs through constructor rules, functions that build new signs from other signs of given sorts plus additional parameters. Constructors express the whole/part relationship at the base of complex signs. Some sign constructors can be more important than others which gives rise to a priority partial ordering on the constructors of a given sort, for example: the pollutants in a lake may be prioritised by their toxicity, to aid in the design of an appropriate visualization. The complexity of a sign is measured in term of a hierarchy of levels, with atomic signs at the lowest levels and complex sign built from signs at lower or same levels.

2.3.3. Semiotic Systems: Grammar

Bertin [Ber83] proposed the first and probably unique attempt at developing a syntax of visual signs based on formal rules. Bertin identified six visual primitives, or fundamental visual variables, which are at the basis of the construction of any graphics sign: size, colour hue, colour value, grain, orientation, and shape. Bertin rated each visual variable in function of the signified dataset, giving a rating of appropriate or inappropriate to each visual variable for numerical, ordinal, and categorical data. This laid down the grammatical rules of a syntax to guide the choice of appropriate forms of graphical representation. MacEachren [Mac04] proposed adding three extra variables based on advances in graphics technology (Figure 5): clarity (fuzziness) of sign vehicle components, resolution (of boundaries and images), and transparency. He also provides a three-step rating for the full set of visual variables of good, marginal, and poor for use with numerical, ordinal, and categorical data. Mackinlay [Mac86] demonstrated the usefulness of such syntax of visual variables with his early implementation of an expert

system for automating the design of graphical representations.

3. Design Criteria and Guidelines

Glyphs represent different data variables by a set of visual channels including shape, size, colour, orientation, etc. It was a wide-spread opinion in the related research community for a long time that "just" knowing these basic principles of glyph-based visualization would suffice to its successful usage. More recently, however, it has been understood that only well designed glyphs are actually useful. Visual channels such as colour [Chr75] or size [LMvW10] are more dominant and can help to focus the user's attention. Other channels such as position, length, angle or slope can be measured and compared more accurately [CM84a, HBE96]. An effective glyph visualization should, therefore, carefully choose and combine different visual channels. In this section, we discuss critical design aspects and guidelines for glyph-based visualization.

3.1. Design Space

As stated by Pettersson [Pet10] the main goal in information design is clarity of communication; in order to fulfill this goal, all messages must be accurately designed, produced and distributed, and later correctly interpreted and understood by members of the intended audience. Several principles to assist this design process have been proposed in the literature some empirical in nature others more formally defined.

3.1.1. Perceptual Codes

Gestalt psychologists outlined several fundamental and universal principles (or laws) of perceptual organisation which are assumed as a basis of a perceptual code (Figure 6): proximity, similarity, continuity, closure, figure/ground, area, symmetry and prägnanz.

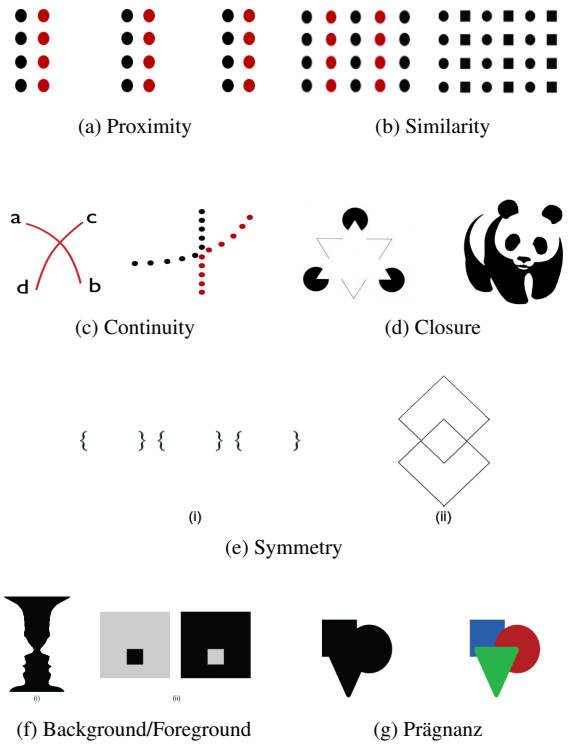
The proximity principle (Figure 6a) states that objects that are closer to one another are perceived to be more related than those that are spaced farther apart. The proximity relation has been proved to be stronger than colour similarity.

The similarity principle (Figure 6b) states that objects that are similar are perceived to be more related than those that are dissimilar.

The continuity principle (Figure 6c) states that elements that are arranged on a line or curve are perceived to be more related than elements not on the line or curve. Continuation is stronger than similarity of colour.

The closure principle (Figure 6d) states that elements in a complex arrangement tend to be grouped into a single, recognisable pattern.

The symmetry principle (Figure 6e) states that objects are



(a) Proximity (b) Similarity

(c) Continuity (d) Closure

(e) Symmetry

(f) Background/Foreground (g) Prägnanz

Figure 6: Gestalt Principles of Perceptual Organisation.

perceived as symmetrical shapes that form around their center. In Figure 6e (i) the perceived picture is usually three sets of opening and closing brackets while in Figure 6e (ii) the dominant picture would be two overlapping diamonds. In the first case symmetrical balance is stronger than proximity while in the second case symmetrical regions tend to be seen as the dominant figures.

The figure/ground principle (Figure 6f) states that elements are perceived as either figure (element of focus) or ground (background or surrounding area). In this principle several factors play an important role: surroundedness, size (or area), symmetry, parallelism, and extremal edges. Each of these five properties can determine which parts of a figure are classified as figure or as background.

The prägnanz principle (Figure 6g) states that confronted with an ambiguous or complex representation the simplest and most stable interpretation is always favoured.

3.1.2. Visual Channels

A *visual channel* is a collection of primitive visual representations that are used to convey different values of a variable. Other terms were introduced in the literature. For example, Bertin called them *retinal variables*, Ware referred to them as *visual encoding variables* as well as *visual channels*. Cleveland and McGill proposed a ranking of several

Geometric Channels	Optical Channels	Topological and Relational Channels	Semantic Channels
• size / length / width / depth / area / volume • orientation / slope • angle • shape • curvature • smoothness	• intensity / brightness • colour / hue / saturation • opacity / transparency • texture (partly geometric) • line styles (partly geometric) • focus / blur / fading • shading and lighting effects • shadow • depth (implicit / explicit cues) • implicit motion / motion blur • explicit motion / animation / flicker	• spatial location • connection • node / internal node / terminator • intersection / overlap • depth ordering / partial occlusion • closure / containment • distance / density	• number • text • symbol / ideogram • sign / icon / logo / glyph / pictogram • isotype

Table 1: Visual Channels [CF12].

visual channels (i.e., position, length, angle, slope, area, volume, colour and density) [CM84b]. Mackinlay extended this exercise to some 13 visual channels [Mac86]. In addition, perceptual studies have been carried out to evaluate the effectiveness of some basic visual channels, resulting in a common consensus about pop-out effects of some of them : colour $\prec$ size $\prec$ shape $\prec$ orientation (e.g., [Wil67, QH87, ROP11]). The symbol $\prec$ reads as *precedes*. However, the strength of colour over the other three channels is generally much more noticeable.

Recently Chen and Floridi organised over 30 visual channels into a simple taxonomy consisting of four categories, namely geometric, optical, topological and semantic channels [CF12].

Combining these into a common table, we have a rich collection of visual channels (Table 1).

Most of these visual channels can be of potential use in glyph design, though only a small number of channels have been used in the literature. This suggests that the design space for glyphs is far from being fully explored.

3.1.3. Design Criteria

According to Eco [Eco79], a general semiotic theory should include not only a theory of how codes may establish rules for systems of signification but a theory of how signs may be produced and interpreted to clarify aspects of communications. In the work of Yousef [You01] five criteria have been proposed and empirically validated in the context of visual metaphors used in interface design. The criteria proposed are referred to with the acronym of CARSE: contextual suitability, applicability of structure, representability/imagery, salience imbalance, prominence and emotional tone.

Context suitability, or relevance, indicates the extent to which the metaphorical sign resembles the source domain with respect to the context of use.

Applicability of structure indicates the extent to which the proposed metaphorical sign is relevant to the new and unfamiliar concept that is being explained. The criteria can be regarded as the correspondence between the source and the target domain, in [TS82] is referred to as "Within-Domain Distance" while Lakoff [Lak95] calls it the "Invariance Principle".

Representability/imagery indicates the ease with which the visual metaphor can be represented.

Salience imbalance refers to Ortony's [Ort93] statement that good metaphors are the ones in which the source (vehicle) domain contains elements or traits, which are highly explicit/prominent; at the same time these traits are very subtle in the target (topic) domain. The visual representation should convey these salient source traits to the receiver.

Emotional tone indicates the importance of emotions triggered by the metaphor as one indicator of the semantic efficacy of the function that is presented metaphorically. In a recent study, Maguire et al. proposed a set of guidelines based

on the literature of psychology and Bertin's categorisation of semantic relevance [MRSS*12]. These guidelines are:

- **Guideline on Semantic Relevance**. Bertin [Ber83] classified visual channels (which he referred to as retinal variables) into two categories, planar (location) and retinal (size, colour, shape, orientation, texture and brightness). Bertin proposed four semantic criteria for determining the suitability of different channels in representing certain types of information. These semantic criteria are: *associative*, *selective*, *ordered* and *quantitative*. Since then, research has also improved Bertin's analysis. For example, it was shown that practice and familiarity can support selectivity with almost any shape [TG88,WCG94,Gre98].
- **Guideline on Channel Composition**. As a glyph is likely to feature a number of visual channels, the constructive composition may affect how individual channels are perceived. A rich collection of literature on integral and separable dimensions shows that the combined dissimilarity of closely integrated visual channels exhibits Euclidean distance $\sqrt{d_a^2 + d_b^2}$ [KT75, HI72], whereas that of separable visual channels exhibits city-block distance $d_a + d_b$ [BSMWE78,She64]. The latter is more cost-effective than the former in rule-based encoding of multi-faceted concepts, therefore effective glyph design should encompass a non-conflicting set of separable retinal variables.
- **Guideline on Pop-out Effects**. Many classic studies in perception also established the "power" of different visual channels in terms of *pop-out effect* (pre-attentive search), and fixation (during attentive search) [HE11]. The *pop-out effect* is one which allows identification of a target within a few nanoseconds of initial exposure to the visual search space. A result of several milestone studies focusing on observed response times it shows the ordering of the four commonly used visual channels to follow the consensus: colour $\prec$ size $\prec$ shape $\prec$ orientation (e.g., [Wil67,QH87,ROP11]). The symbol $\prec$ reads as *precedes*. However, the strength of colour over the other three channels is generally much more noticeable.
- **Guideline on Visual Hierarchy**. *Visual hierarchy*, with which the environment and objects around us are arranged is a well documented theoretical framework [Pal77, Nav77, LRW99, KW79, Bar04]. However, the literature debates over the ways in which the visual system traverses this hierarchy. There are four possible ways: top-down (also called global processing) [Nav77]; bottom-up (also called local processing); middle-out [KW79]; and salient features (*e.g., edges, points, colours*) [Rum70]. Because glyphs are relatively small in comparison with an entire visualization, top-down and salient feature detection play significant roles in selecting a glyph or glyphs of interest. The top-down assumption suggests that when considering a glyph in isolation, its global features will affect visual search more than its local features. Salient features are partly addressed by pop-out effects.

In addition, Maguire et al. also suggested the importance of establishing a metaphoric association between a visual channel and the concept or concepts to be encoded [MRSS*12]. Metaphoric visual representations enable domain-specific encoding using "natural mapping" [Sii02, Nor02]. This natural mapping can make it easier for users to infer meaning from the glyph with less effort required to learn and remember them [MdBC00]. A recent study showed that visual metaphors can aid memorization of the information depicted in a visualization [BARM*12]. However, the same study also showed that visually realistic metaphors (those with a lot of detail) may have a negative impact on performance in visual search. Moreover, realistic visual metaphors require a higher pixel resolution, and would lose their discriminating capacity in low resolution visualizations.

A glyph is composed by a set of visual channels, each of which encodes a variable of a multivariate data record. Naturally the first criterion is that the visual channel should ideally be able to encode many valid values of that variable, or collectively, different visual channels of the glyph could encode many data records with different combinations of data values. However, this is not the only criterion, and in many cases, it may not even be the most important criterion. If the goal is to encode as many values as possible, one may be better off reading these values in text directly. Chung et al. [CLP*13] proposed eight criteria for glyph design in the context of sorting glyphs visually (Figure 7). These are:

a *Typedness* – This criterion refers to whether or not each visual channel in a glyph is appropriately selected to match with the data type of the variable to be encoded. Such data types may include, but not limited to: *nominal*, *ordinal*, *interval*, *ratio*, and *directional*.

b *Visual Orderability* – When a variable to be encoded is orderable, the corresponding visual channel should ideally be orderable visually (e.g., size, greyscale intensity, but not an arbitrary set of shapes).

c *Channel Capacity* – This refers to the number of values that may be encoded by a visual channel. Such a number is often affected by the size of a glyph and many perceptual factors (e.g., just-noticeable-difference, interference from nearby visual objects).

d *Separability* – When two or more visual channels are integrated into a compound channel, such as combining intensity, hue and saturation into a colour channel, the interference between different primitive channels should be minimised.

e *Searchability* – This refers to the levels of ease when one needs to identify a specific visual channel within a glyph for a specific variable.

f *Learnability* – This is often an important criterion in many applications. Ideally, a glyph design should be easy to learn, and easy to remember. There are many factors that may affect a visual design in this context, for instance, whether there are well-defined constructive rules, whether

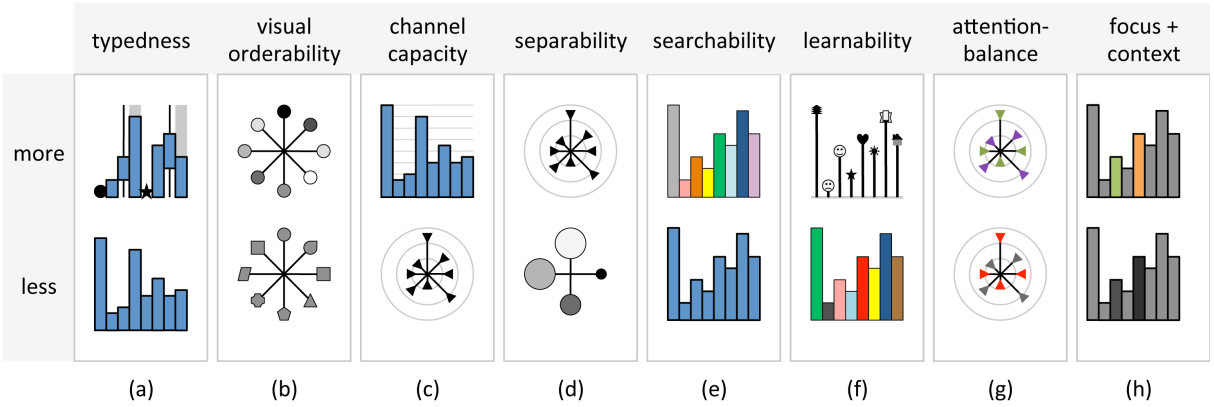


Figure 7: Glyph design criteria [CLP*13].

there are memorable metaphors, whether it is easy to guess, and so on.

g *Attention Balance* – Different visual channels in a glyph will receive different levels of attention. Ideally, the levels of attention should correspond to the levels of importance of the variables. However, this is easier said than done as the relative importance of a variable is often undefined or may vary from tasks to tasks.

h *Focus and Context* – This refers to the need to identify an individual visual channel under a certain interactive operation. For example, when a user select a certain variable as a sort key, it is desirable to highlight the corresponding visual channel so it stands out from other channels.

This is not an exhaustive list, and there are other design criteria, such as aesthetic appearance that also play an important role.

3.1.4. Design Processes

Petterson [Pet10] introduces four categories of principles supporting the visual representation design process:

- Functional Principles: focus, structure, clarity, simplicity, emphasis and unity;
- Administrative Principles: accessibility, cost, ethics and quality;
- Aesthetic Principles: harmony and aesthetic proportion;
- Cognitive Principles: facilitating attention, facilitating perception, facilitating mental processing and facilitating memory.

For each category Petterson provides detailed guidelines on how to achieve the target result with the appropriate use of text, picture, layout and colour.

Given the abundance of multivariate data, perceptual and cognitive efficiency is at the core of glyph-based visualization. Karve and Gleicher [KG07] identify three considerations for the design of complex and compound glyphs: integral-separable dimension pairs, natural mappings and perceptual efficient encoding. Integral-separable dimension pairs focus on the readability of multi-attribute glyphs and multi-glyphs displays, Karve and Gleicher [KG07] argue that individual glyphs should combine as many separable visual attributes as possible and multi-glyph displays should be dense, juxtaposing related items, and employing repetitive design motifs that support inter-glyph comparison. Natural mappings (e.g. use of metaphoric representations) focuses on the natural relationship between data and glyph features; a clear relationship between visual and data attributes enhances glyph usability. Perceptual efficiency of the encoding focuses on the encoding of a continuous variable; horizontal bars on a shared positional scale are found to be the most accurate method followed in decreasing order of accuracy by interval length, slope, area, volume, and colour.

3.1.4.1. Measurements and Norms If symbol design is to progress, we need to know more about why some symbols are easier to use than others. A major obstacle facing researchers attempting to answer this question has been the difficulties in quantifying symbol characteristics so that they can be experimentally controlled. A good way of controlling symbol characteristics experimentally is to obtain subjective ratings of each characteristic.

Although there has been a long tradition in psycholinguistic research of using normative ratings to control item characteristics for words and pictures, no normative ratings for symbols have yet been produced. McDougall et al. [MCdB99, MdBC00] address the problem by providing normative ratings for five symbol characteristics considered determinant in the development of easy to use and understand symbols: concreteness, visual complexity, meaningfulness, familiarity, semantic distance.

McDougall et al. highlights and investigates several interesting correlations between these five criteria. Concreteness, for example, (as opposed to abstraction) is somehow in op-

position to visual complexity; concrete symbols tend to be more visually obvious because they depict objects, places, and people that are already familiar. In contrast, abstract symbols represent information using graphic features such as shapes, arrows and so on. One of the reasons why concrete symbols are more visually obvious may simply be because the extra detail provided in concrete symbols makes them easier to comprehend. In contrast, however, design guidelines typically suggest that the design of symbols or icons should be kept as simple as possible. Other researchers have focused on the fact that concrete symbols are more meaningful than abstract symbols.

Semantic, or articulatory, distance is a measure of the closeness of the relationship between the symbol and what it is intended to represent. A number of classification systems have been developed in order to attempt to characterise the different relationships between symbols and their functions [Pei02].

Familiarity reflects the frequency with which symbols are encountered. This property is thought to be an important determinant of usability. It is evident that user performance improves dramatically as a result of learning symbols and signs. The effects of some symbol characteristics on performance, such as colour and concreteness, diminish as symbols become more familiar but others, such as complexity, do not.

In [MdBC00, NC08] the relationship between: concreteness/visual complexity, concreteness/meaningfulness and meaningfulness/familiarity/semantic distance were examined in detail using subjective rating methods. For each characteristic subjects had to choose bipolar adjectives based on a five-point scale to indicate their perception of an icon. Ng et al. [NC08] propose a review of the relationships among the same five characteristics together with a description of three types of measures used in literature to quantify such relationships:

- subjective rating (as in [MdBC00]);
- icon-based metric: the measure is obtained summing up the components of an icon, such as letters, lines, arrows and so on;
- automated visual measurement: the measure is a function of icon features extracted via image analysis techniques such as edge-detection, perimeter determination, decomposition and so on.

Other symbol characteristics present in literature are discriminability, distinctiveness and configurality, however to provide a normative rating is a much harder task since such characteristics can only be defined (and quantified) in relation to the other symbols included in the display as a whole [MdBC00].

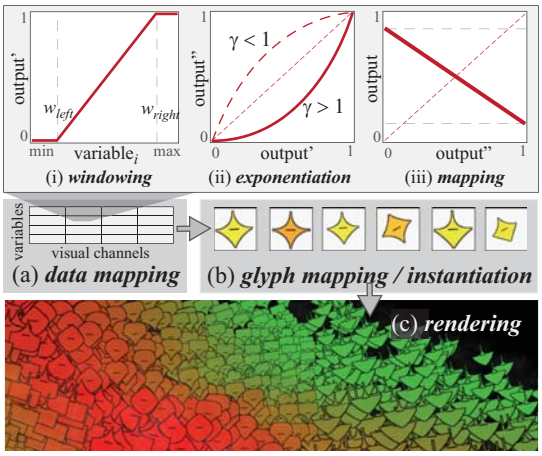


Figure 8: A pipeline for creating glyphs [LKH09]: (a) Each data variable is subject to three stages of data mapping: windowing, exponentiation and mapping. (b) The data variables are mapped to the different visual channels of a glyph (e.g., upper/lower shape, size, and rotation) and used to instantiate the individual glyphs. (c) Finally, the glyphs are rendered in their spatial context.

3.2. General Design Considerations and Guidelines

3.3. Design and Usage Guidelines for Glyphs

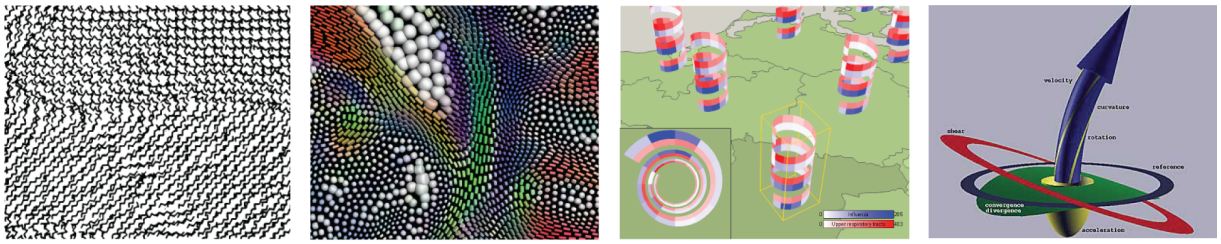
A number of design guidelines (marked with DGx in the following) for glyph-based visualization have been proposed [War02, War08, RP08, LKH09, ROP11, MRSS*12], and we review them in the following. Ward [War02] surveys glyph-based representations for information visualization and presents a taxonomy for glyph placement. Ropinski et al. [RP08, ROP11] propose a perception-based glyph taxonomy for medical visualization. Glyph-based visualizations are categorised according to:

- *pre-attentive* visual stimuli such as glyph shape, colour and placement, and
- *attentive* visual processing, which is mainly related to the interactive exploration phase (e.g., changing the position or parameter mapping of a glyph).

In the context of medical visualization, the authors propose usage guidelines for glyphs, which are addressed later on.

Inspired by the work of Ropinski and Preim [RP08], Lie et al. [LKH09] propose further guidelines for glyph-based 3D data visualization. Aligned with the visualization pipeline [HS09], the task of creating a glyph-based 3D visualization is divided into three stages as shown in Figure 8:

- during *data mapping*, the data attributes of a record are remapped (to achieve, for example, some contrast enhancement) and mapped to the different visual channels of a glyph;



(a) (b) (c) (d)

Figure 9: Small simple glyphs vs. large and complex glyphs: (a) Stick figures form textural patterns [PG88]. (b) Dense glyph packing for diffusion tensor data [KW06]. (c) Helix glyphs on maps for analyzing cyclic temporal patterns for two diseases [TSWS05]. (d) The local flow probe can simultaneously depict a multitude of different variables [dLvW93].

- *glyph mapping* (or glyph instantiation) creates the individual glyphs, properly arranged across the domain; and
- during *rendering*, the glyphs are placed in the resulting image, where one has to cope with issues such as visual cluttering or occlusion.

For each of these steps, the following sections discuss critical design aspects and guidelines for glyph-based visualization.

Table 2 illustrates different papers which are consistent with the design guidelines presented here. The papers are also categorised according to the utilised visual channels, dimensionality of the visualization space, and density of glyph placement.

[DG1] Task-based choice of visualization space. Glyph-based visualization approaches vary with respect to whether they are constructed in a 2D or 3D visualization space. In case of abstract data such as census or financial data, this decision is often dependent on the task at hand. However, certain scenarios with 3D volumetric or flow data inherently require a 3D visualization. We think that it also makes sense to consider glyph-based visualizations, which are based on the placement of glyphs on 3D surfaces [RSMS*07] (called 2.5D in the following).

[DG2] Task-based compromise between complexity and density. Glyph-based visualization approaches span a certain spectrum from dense arrangements of relatively simple shapes such as stick figures [PG88] (Figure 9a) to individual instances of complex glyphs that reveal a lot of information (but only for few, selected places, Figures 9c and d). Additionally, we can differentiate visualization solutions according to which visual channels are varied according to the data, and how many different values a glyph eventually represents. Usually this number is not too large, often 2 to 4, but then also examples exist where dozens of values are represented (e.g., the local flow probe [dLvW93] in Figure 9d).

[DG3] Hybrid visualizations. Ropinski et al. [RP08] suggest combining glyphs with other visualization techniques such as isosurfaces or volume rendering, which provide spatial context [RSMS*07, CM93]. When glyphs are not placed in a dense way, the space between them can be used for additional information. Treinish [Tre99], for example, visualizes multivariate weather data using colour contouring on vertical slices and isosurfaces that represent cloud boundaries. At user-defined locations (vertical profiles), the wind velocity and direction are represented by a set of arrow glyphs. Streamlines following the wind direction are seeded at each arrow. Kirby et al. [KML99] use concepts from painting for visualizing 2D flow. They combine different image layers with glyphs, elongated ellipses, and colour.

3.4. Data Mapping

Each dimension or variable of a data set will map to a specific graphical attribute. By modifying the order of dimensions while preserving the type of mapping, as many as N! alternate "views" of the data can be generated. An important issue in using glyphs is to ascertain which ordering(s) will be most supportive of the task at hand. Several possibilities exist, beyond random ordering or the order in which the variables were originally stored [War08]:

- **Correlation-driven.** Many researchers have proposed using correlation and other similarity measures to order dimensions for improved visualization [Ber83, ABK98, FK03, BS92]. These orderings help reveal clusters of similar variables, outlier records, and gradual shifts in relationships between variables.
- **Complexity and Symmetry-driven.** Gestalt principles indicate we have a preference for simple shapes, and we are good at seeing and remembering symmetry. In [PWR04] the shapes of star glyphs resulting from using different dimension orders were evaluated for two attributes: monotonicity (the direction of change is constant) and symmetry (similar ray lengths on opposite sides of the glyph). The ordering that maximised the number of simple and symmetric shapes was chosen as the best. User studies showed improved performance with complexity and symmetry optimised orderings.
- **Data-driven.** Another option is to base the order of the dimensions on the values of a single record (base), using

Figure 10: The figure shows monetary exchange rates over 3 years using random ordering.

Figure 11: In this figure, the dimensions are sorted based on the first record. Gradual changes and anomalies are much easier to perceive.

an ascending or descending sort of the values to specify the global dimension order. This can allow users to see similarities and differences between the base record and all other records. For example, sorting the exchange rates of 10 countries with the U.S. by their relative values in the first year of the time series exposes a number of interesting trends, anomalies, and periods of relative stability and instability (Figures 10 and 11).

- **User-driven.** As a final strategy, we can allow users to apply knowledge of the data set to order and group dimensions by many aspects, including derivative relations, semantic similarity, and importance. Derivative relations mean that the user is aware that one or more dimensions may simply be derived through combinations of other dimensions. Semantic similarity indicates dimensions that have related meanings within the domain.

Finally, some dimensions are likely to have more importance than others for a given task, and thus ordering or assigning such dimensions to more visually prominent features of the glyph will likely have a positive impact on task performance. In order to optimally represent a data variable using a visual channel of the glyph, the corresponding data range should be normalised, for instance, to the unit interval [ROP11, War02, LKH09]. The remapped data attributes parameterize the visual appearance of a glyph. Ropinski et al. [RSMS*07], for example, use an interface similar to a transfer function editor for mapping a data attribute to a visual channel of the glyph.

Lie et al. [LKH09] propose three consecutive steps for the data mapping stage. First, the data values within a user-selected data range $[w_{left}, w_{right}]$ are mapped to the unit interval (Figure 8a (i)). Values outside this range are clamped to 0 or 1, respectively. Consequently, the contrast of the visualization can be enhanced with respect to a range of interest (sometimes called *windowing*). A linear mapping would be a natural choice for this step, but also other forms of mapping could be considered, such as a discontinuous mapping. Another option would be a ranking-based mapping where the data is sorted first and each discrete value (or bin) is then shown differently, for example, using different shapes such as a triangle, circle, or star [STH02]. After the windowing, the contrast of a data variable can be further enhanced using an optional exponential mapping $e(x) = x^{\gamma}$. Using a value $\gamma \in]0,1[$, smaller values are represented more prominently (see the dashed red curve in Figure 8a (ii)). In contrast, larger values are emphasised with $\gamma > 1$. Since an exponential mapping can be hard to interpret, it should not be used as a default mapping. It can rather be applied when the user is interactively exploring the visualization, for instance, by modifying the parameter mappings to focus of different portions of the data. Finally, a third mapping step enables the user to restrict or transform the output range that should be depicted by a visual channel. Using a reverted mapping, for instance, smaller values, which are possibly more important to the user, are depicted in an enhanced style while larger values are de-emphasised. Consequently, also semantics of the data variables can be considered, which is an important guideline when mapping a data variable to a visual channel of a glyph [War02, ROP11].

3.5. Glyph Mapping / Instantiation

During glyph mapping the individual glyphs are created by representing the data variables with different visual channels of a glyph. During this step, the glyphs are also properly arranged across the domain. In the following, we discuss general design guidelines during this mapping stage as well as guidelines related to glyph shape and appearance.

[DG4] Perceptually uniform glyph properties. When mapping a data variable to a glyph property, equal distances in data space should be perceived equally as well. This is an important guideline for glyph design, and it was originally developed for colour maps [RTB96]. The box plot [MTL78], for example, uses position and height of the box / whiskers

Authors / Technique	design guideline													visual channel					
	[DG1] visualization space	[DG2] complexity vs. density	[DG3] hybrid visualizations	[DG4] perceptually uniform properties	[DG5] redundant mapping	[DG6] importance-based mapping	[DG7] view point independence	[DG8] simplicity and symmetry	[DG9] orthogonality and normalization	[DG10] intuitive / semantical mapping	[DG11] balanced glyph placement	[DG12] facilitate 3D depth perception	[DG13] interactive occlusion control	color	shape	size / height / length	orientation	texture	opacity
Brewer [Bre99]: Color use guidelines				■		■				■				■					
Cleveland & McGill [CM84]: Graphical perception	2D/3D			■					■		■			■	■	■	■	■	
Crawfis & Max [CM93]: Vector field visualization	3D	2				■	■	■		■	■	■		■			■		■
de Leeuw & van Wijk [dLvW93]: Local flow probe	3D	-3				■			■	■				■	■	■	■		
Healey & Enns [HE99]: Combining textures and colors	2.5D	1		■				■	■	■	■			■	■	■	■	■	
Healey et al. [HBE96]: Preattentive processing	2D			■	■			■	■		■			■	■	■	■	■	
Kindlmann & Westin [KW06]: Glyph packing	3D	2		■			■	■	■	■	■	■		■	■		■	■	
Kindlmann [Kin04]: Superquadric tensor glyphs	2.5D	1.5		■				■	■	■	■				■		■	■	
Kirby et al. [KML99]: Concepts from painting	2D	1		■			■		■	■	■			■	■	■	■	■	■
Laidlaw et al. [LAK*98]: Stochastic glyph placement	2D	2		■			■		■	■	■	■		■	■	■	■	■	■
Li et al. [LMvW10]: Symbol size discrimination	2D				■				■						■	■			
Lie et al. [LKH09]: Design aspects of glyph-based 3D visualization	3D	2			■	■		■	■	■	■		■	■	■	■	■		■
McGill et al. [MTL78]: Variations of box plots	2D	-3			■	■	■		■	■	■				■	■			
Meyer-Spradow et al. [MSSD*08]: Surface glyphs	2.5D	0	■			■	■	■	■	■	■		■	■	■	■			■
Peng et al. [PWR04]: Clutter reduction using dimension reordering	2D	1		■	■			■	■		■				■	■			
Pickett & Grinstein [PG88]: Stick figures	2D	3		■	■			■	■								■	■	
Piringer et al. [PKH04]: Depth perception in 3D scatterplots	3D								■	■		■	■	■		■			■
Rogowitz et al. [RTB96]: How not to lie with visualization	3D		■	■		■			■	■				■					■
Tominski et al. [TSWS05]: Helix glyphs on geographic maps	2.5D	-2	■		■					■	■	■		■		■			
Treinish [Tre99]: Task-specific visualization design	2.5D	-2	■	■		■			■	■	■			■	■		■	■	
Ward & Guo [WG11]: Shape space projections	2D	3		■	■			■						■	■	■			

Table 2: Categorisation of glyph-based approaches according to design guidelines, visualization space and visual channels. In DG2, the approaches span a spectrum from individual instances of complex glyphs (-3) to dense arrangements of relatively simple shapes (+3).

to encode minimum and maximum value, median, and other quartile information of a data distribution. A negative example in this context would be mapping a data variable to the radius of a circle. The circle's area then increases quadratically with respect to the radius (instead of linearly). Li et al. [LMvW10] study the perception of symbol size, which is assumed to be the second dominant visual channel (after colour [Chr75]). Their experiments suggest that the perception of size can be best represented by a power law transformation. Another negative example would be the usage of a rainbow colour map, which is not perceptually uniform and does not have a perceptual ordering [BT07].

[DG5] Redundant mapping of variables. According to Ward [War08], there are three different mappings:

- a one-to-one mapping assigns each data variable to a different visual channel of the glyph;
- a one-to-many mapping makes use of redundancies by mapping a data variable to multiple glyph channels. Such a mapping can reduce the risk of information loss by encoding important variables multiple times, which is also an important guideline for glyph design [LKH09,ROP11].
- a many-to-one mapping represents multiple data variables by the same kind of visual channel, for example, the height of bars in a histogram or profile glyph. Such a mapping is useful when comparing the different data variables for a data element [War08].

[DG6] Importance-based mapping. According to Ropinski et al. [ROP11], important variables should be enhanced in the visualization, for instance, by using a redundant mapping (compare to the previous guideline). Moreover, the mapping should guide the user's *focus of attention*, e.g., using more prominent visual stimuli such as colour, size or opacity to encode relevance. Ropinski et al. [RSMS*07], for example, use surface glyphs to show data from positron

emission tomography (PET). An inverse mapping is used, which maps low PET activity to thick and high PET activity to thin glyphs. Consequently, interesting regions with reduced activity are shown in an enhanced style. Maguire et al propose an algorithmic approach to importance-based mappings [MRSS*12]. Their algorithm builds a taxonomy (a hierarchical classification) from a list of qualitative terms grouped into classification schemes. The higher up some classification scheme is in the taxonomy (determined algorithmically and based on term use for instance), the stronger the visual channel to represent that scheme will be.

3.5.1. Shape Design

One of the most prominent visual channels of a glyph is its shape. Ropinski et al. [ROP11] distinguish between *basic glyph shapes* such as variants of superquadrics [Bar81] (sphere, torus) and *composite shapes* that combine multiple basic shapes. Since basic shapes can be perceived pre-attentively the authors argue that they should be used to convey the most important information. Composite glyphs, on the other hand, are interpreted in the exploration phase and are usually not pre-attentive, i.e., they are analysed sequentially. Chernoff faces [Che73], for instance, represent data variables by different features of a cartoon face (e.g., shape of the face; size and position of eyes, nose, and mouth; curvature of the mouth). The Glyphmaker [RAEM94] provides a user interface that enables non-programmers to map data variables to the different properties of a glyph such as position, colour, shape, overall size and transparency. Kraus and Ertl [KE01] propose a similar tool for scientific data.

[DG7] View point independence: Glyph shapes should be unambiguously perceivable independent of the viewing directions [ROP11]. When using 3D glyph shapes, one has to account for possible distortions introduced when viewing the glyph from a different point of view. Lie et al. [LKH09], therefore, suggest to use 2D billboard glyphs in order to avoid this problem. In certain scenarios, however, it makes sense to use 3D glyphs, for example, when they have a semantic meaning. Such an example would be arrow glyphs that depict a flow field [CM93]. Kindlmann [Kin04] use superquadric glyph shapes that fulfill DG7. For composite shapes, Ropinski et al. [ROP11] distinguish between directional and non-directional glyphs.

[DG8] Simplicity and Symmetry: According to Gestalt laws [War04], simple and symmetric shapes facilitate the perception of visual patterns. Also, simple glyph shapes enhance the detection of minor shape changes as well as outliers [War08]. Peng et al. [PWR04], for instance, automatically reorder the data-to-property mapping for generating more symmetric and simple star glyphs. Lie et al. [LKH09] propose horizontally symmetric glyphs that are based on superellipses, which should facilitate the mental reconstruct of glyph parts that are occluded.

In the following, additional guidelines for shape design are discussed in relation to other visual properties.

3.5.2. Other Visual Properties / Glyph Appearance

Pre-attentive visual stimuli such as position, width, size, orientation, curvature, colour (hue), or intensity are a powerful way to represent data [CM84a, HBE96]. These visual channels are rapidly processed by our low-level visual system and can thus be used for the effective visualization of large data. Special care is required, however, if several such stimuli are combined—the result may not be pre-attentive any more. Healey and Enns [HE99] propose simple texture patterns and colour to visualize multivariate data. Different data variables are encoded in the individual elements of a perceptual texture using equally distinguishable colours and texture dimensions such as element density, regularity and height. Groups of neighboring elements form texture patterns that can be analysed visually.

Ward [War08] identifies different biases that are introduced when mapping a data variable to a glyph property. The first kind of biases are related to human *perception*. Different properties of a glyph can be perceived and related with varying accuracy. Cleveland and McGill [CM84a] identify different visual channels and perform perceptual experiments. The visual channels are ordered based on how accurately they can be perceived: 1) position along a common scale; 2) position along non-aligned scale; 3) length, angle or slope; 4) area; 5) volume or curvature; 6) shading or colour saturation. Moreover, adjacent properties of a glyph are easier to relate and compare than nonadjacent (Ward calls these *proximity*-based biases [War08]). Finally, data variables mapped to semantically or perceptually *grouped* glyph properties (e.g., the ears or eyes in Chernoff faces [Che73]) are easier to distinguish than variables mapped to non-related features.

[DG9] Orthogonality and Normalization: When designing glyphs, it is especially important to consider how different glyph properties interact with each other and thereby possibly distort the interpretation (compared to channel composition [MRSS*12]). A challenge in this context is the *orthogonality* [LKH09] of the different glyph components, meaning that it should be possible to perceive each visual cue independently (or to mentally reconstruct the depicted data variables as suggested by Ropinski et al. [ROP11]). Moreover, one has to account for distortions introduced by the different glyph properties. When using, for example, glyph shape to represent a data variable this affects the area (size) of the glyph as well. Accordingly, such effects should be *normalised* against each other [LKH09]. In the previous example, the overall glyph size could thus be altered in order to compensate for the changes introduced by variations in shape. However, it is not always easy to design a glyph-based visualization such that the different data-to-property mappings are independent and do not influence each other (e.g., the interpretation of shape details is usually influenced by the size of the glyph).

[DG10] Intuitive mapping based on semantics. Semantics of the data should be incorporated in the glyph mapping [War08, LKH09, ROP11, MRSS*12]. Crawfis and Max [CM93], for instance, combine small coloured vector glyphs depicting wind velocity with contour surfaces representing cloudiness. Another example would be to represent temperature with a diverging colour map [Bre99], where white is used to indicated 0°C, blue indicates minus and red plus degrees.

3.5.3. Glyph Placement

The placement of glyphs is a prominent visual stimuli and can be used to convey information about the data. In the context of information visualization, Ward [War02] categorizes placement strategies into *data-* and *structure-driven* placement. The former is directly based on individual variables or spatial dimensions of the data, or on derived information such as principal components. Examples of data-driven strategies are placing the glyphs in a 2D scatterplot or locating them aligned with the underlying data grid (in case of spatial data). Structure-driven placement, on the other hand, is based on the ordering, hierarchical or other relationships of the data variables. According to Ropinski et al. [ROP11] such strategies, however, are not directly applicable to medical data. Therefore, they suggest *feature-driven* placement as an additional category, where glyphs are placed on local data features such as iso-surfaces [RSMS*07, MSSD*08]. We consider it useful to also consider *user-driven* placement, where glyphs are manually placed to investigate the data at a certain location [dLvW93, Tre99].

In the context of data-driven placement [War02], glyphs can be placed according to *derived information* as well. Dimensionality reduction approaches, for instance, aim at reducing the data dimensionality while maintaining the higher-dimensional characteristics. Such placement strategies can facilitate the perception of similar glyph shapes, which should be located close to each other. Principal component analysis [WG11] (PCA) is such an example, which transforms multivariate data into an orthogonal coordinate system that is aligned with the greatest variance in the data. Wong and Bergeron [WB97] apply multi-dimensional scaling (MDS) for mapping higher-dimensional data items into a lower-dimensional space while preserving the dissimilarities between the items. Since MDS also maintains the higher-dimensional structure of the data, it is well suitable for subsequent clustering. With such methods, however, the semantic meaning of the glyph location is usually lost, in contrast to techniques that are based on the raw data [War02].

[DG11] Balanced glyph placement. Glyphs may overlap and form unwanted aggregations in image space, for instance, resulting from a regular data grid. Such aggregations should be avoided, since they may be erroneously identified as features [ROP11, War02]. Laidlaw et al. [LAK*98], for instance, apply random jittering when placing brush strokes to represent DTI data. Kindlmann and Westin [KW06] use a particle system for densely packing superquadric glyphs (Figure 9b). Meyer-Spradow et al. [MSSD*08] evenly distribute surface glyphs by combining a random placement with relaxation criteria.

In the context of glyph placement, the number of depicted data variables must be seen in relation to the available screen resolution (compare to DG2). Large and complex glyphs such as the local probe [dLvW93] can be used when only a few data points need to be visualised (or during individual exploration). If many glyphs should be displayed in a dense manner, however, a more simple glyph may be desirable [PG88, KW06, LKH09].

3.6. Rendering

In the final stage of the visualization pipeline (Figure 8), glyphs are transferred from visualization space to the resulting image, where one has to cope with issues such as visual cluttering, depth perception, and occlusion [LKH09]. In the following, we discuss approaches such as halos, interactive slicing, or brushing.

[DG12] Facilitate depth perception for 3D visualizations. In cases where many glyphs overlap, *halos* can help to enhance the depth perception and to distinguish individual glyphs from each other [LKH09]. Piringer et al. [PKH04] and Interrante et al. [IG98] use halos to emphasize discontinuity in depth and to draw the users attention towards objects. For improving the depth perception for non-overlapping glyphs, a special colour map (called *chroma depth* [Tou97]) can be used to represent depth. Since colour is a dominant visual channel, however, it is questionable whether to use it for depicting depth instead of depicting a data variable.

[DG13] Avoid occlusion by interactive slicing or brushing: Occlusion is a major problem when reading glyphs. Therefore, it can be advantageous to employ interactive slicing or brushing. Using a view dependent slice-based visualization, for example, glyphs that are located in front of a user-controlled plane are not displayed [LKH09]. Using linking and brushing in coordinated multiple views, glyphs can be filtered out based on user-defined criteria [KMDH11].

[DG14] Avoid perspective projections when using glyph size to encode a data variable [ROP11]. In such cases, an orthographic projection is preferable, which supports the comparison of glyph size at different locations.

3.7. Glyph Interaction

Interaction in glyph-based visualizations forms an important aspect in modern visual analytics. Legg et al. [LCP*12] introduce such an example in sport notational analysis, by developing the MatchPad: an interactive visualization software that incorporates a series of intuitive user-interactions and a

scale-adaptive layout to support data navigation. One essential requirement in notational analysis in sport is the ability to review key video event footage. Since glyphs have a limited encoding capacity, it would be impractical to map such data (e.g., a video clip, tracking data) entirely to a glyph. Thus, the authors interactively link the playback of videos through glyphs to support rapid information retrieval.

The work of Chung et al. [CLP*13] extends this further by integrating focus+context techniques into glyph-based visual analytics to emphasise the perceptual orderability of attributes on glyphs. They propose a system that incorporates a focus+context glyph-based interface to control and understand high-dimensional sorting of multivariate data. Selected components on the glyph are rendered in focus which adjusts and populates various sorting parameters within a linked *Interactive, Multidimensional Glyph* (IMG) plot. The IMG plot arranges the glyphs along two primary sorting axes. Various interactive tools are described to support user exploration which include: brushing tools for selecting glyphs, pan-and-zoom, and optional display preferences (e.g., connectivity lines) for conveying additional data.

4. Application

Glyph-based visualization is an excellent tool for representing single or multiple data attributes. Whilst generic glyphs are desirable and have been well-studied (e.g., Star glyphs [SFGF72] and Chernoff faces [Che73]), the effectiveness of such designs for conveying information are limited when presented with challenging, complex data forms such as vector and tensor data. In addition to various data type constraints, other factors must be considered. For example, the sampling resolution greatly affects how small or how large the glyph can be in order to avoid visual clutter (compare to DG2). Thus, we find that many glyphs are attribute-dependant and that their specific application context is an integral aspect to the design process. In this section, we report a selection of important papers that focus on novel glyph-based visual techniques that have been explored and utilised in various scientific domains.

4.1. Medical Visualization

The recent survey by Ropinski and Preim [RP08, ROP11] provides an overview of existing glyph-based visualization techniques used in the medical domain and propose guidelines for developing more valuable glyph representations. A glyph taxonomy based on the way information is processed when interpreting glyph visualizations is used to classify such techniques. Within semiotic theory, this consists of a two-phase information process: 1) *pre-attentive* processing, that is mainly stimulated by glyph attributes such as size, colour and shape along with glyph placement, texture mapping and glyph filtering and 2) *attentive stimuli* processes which are based on glyph-interaction paradigms. Examples include a colour legend which users can use to formulate more quantitative glyphs and repositioning glyphs where the glyph properties adapt depending on the location. Based on this classification, the authors describe eight usage guidelines which they evaluate against modern diffusion tensor imaging and cardiac visualization.

Westin et al. [WMM*02] introduce a novel analytical solution to the Stejskal-Tanner diffusion equation system from which a set of derived diffusion tensor metrics describes the geometric properties of a diffusion ellipsoid. Using three tensor eigenvalues, the quantitative shape measures, c_l, c_p, and c_s indicate the linear, planar and spherical properties of a tensor. In addition, the authors present a visualization technique using a composite tensor glyph built from a sphere, disc and rod that are mapped to the three eigenvalues which aims to reduce the ambiguity caused by traditional ellipsoid representation. The composite glyphs are colour-coded according to shape such that blue is mapped to linear case, yellow to planarity and red for spherical case.

Oeltze et. al [OHG*08] incorporate 3D glyphs for visualizing perfusion parameters in conjunction with their ventricular anatomical context. They propose two glyph designs: (a) 3D Bull's Eye Plot (BEP) Segment and (b) 3D Time-Intensity Curve (TIC) Miniatures for depicting four perfusion parameters: *Peak Enhancement (PE)*, *Time To Peak (TTP)*, *Integral* and *Up-slope* which describe the myocardial contractility and viability. The 3D BEP segments are ring-shaped glyphs which extend the previous work [CWD*02] from 2D to 3D space. An improved glyph (TIC miniatures) enables intuitive mapping of all important parameters in cardiac diagnosis as a result of encoding TIC semantic metaphors (glyph shape) that is familiar to domain experts. They apply their technique on three datasets from a clinical study.

The work by Meyer-Spradow et al. [MSSD*08] present an interactive 3D glyph-based approach for the visualization of SPECT-based myocardial perfusion data. They utilise a supertorus prototype glyph which characterises SPECT data based on its colour, opacity, size and roundness. The glyphs are positioned along a 3D surface (i.e., the myocardium) using a random distribution with relaxation for depicting information of the underlying tissue. One motivation of such a placement strategy is to provide a more even-distribution of glyphs. This addresses the problem of unbalanced placement that can occur from regular grid sampling in complex and non-uniform meshes (compare to DG11).

4.2. Event Visualization

Event and activity visualization is a rapidly growing research topic. The work by Botchen et al. [BBS*08] describe the VideoPerpetuoGram (VPG), a dynamic technique for visualizing activity recognition found in video streams. This involves stacking temporally spaced intervals of key video

frames and using colour filled glyphs to represent geometric information (e.g., object identifier, position, size), semantic information (e.g., action type and inter-object relation) and statistical information (the certainty and error margins of the analytical results). They demonstrate their technique on surveillance video footage for summarizing the motion of people and actions.

Pearlman and Rheingans [PR07] introduce a glyph-based approach for visualizing computer network security using compound glyphs. The compound glyph representation is a pie chart in which the size and colour of each segment is mapped to the amount of activity and the type of service. One of the motivations of using a simple pie chart design, is its ability to extend to the temporal domain by slicing the glyph as concentric layers for depicting information at different time instances. Each glyph indicates a node on the network in which connectivity lines in the visualization represent the communication between nodes. They successfully demonstrate their method on a simulated network consisting of a small set client users.

The work by Parry et al. [PLC*11] introduce a novel event selection concept for summarising video storyboards. Video storyboard is a form of video visualization, used to summarise the major events in a video using illustrative visualization. There are three main technical challenges in creating a video storyboard, (a) event classification, (b) event selection and (c) event illustration. Among these challenges, (a) is highly application-dependent and requires a significant amount of application- specific semantics to be encoded in this system or manually specified by the users. This paper focuses on challenges (b) and (c) which they demonstrate using a case study on Snooker video visualization. For event illustration, the authors explore a collection of iconic glyphs which convey some metaphors in addition to data values for event labelling. These include ball objects that vary in size, opacity and colour for representing ball trajectory and semantic information (e.g., ball type, event importance), textured circle glyphs and numbered icons for depicting the sequences of shots, and a pie chart icon to represent scoring and video timing information.

A more thorough investigation of incorporating visual semantics into glyph designs is explored by Legg et al. [LCP*12]. They describe the MatchPad: an interactive glyph-based visualization for mapping events and actions in sports notational analysis. Sports event analysis provides an example where a large number of event types need to be depicted in a manner to facilitate rapid information retrieval. A comprehensive review of mapping such data is discussed using different levels of abstractions. These include the evaluation of abstract icons and colour for encoding each event type. Whilst the approach may be suitable for data attributes with a small number of enumerative values, the range of categorical attributes in sports results to many different shapes or colours making it cognitively challenging to learn, remember or guess. Instead, the authors explore the use of metaphoric pictograms which are commonly used in many domain-specific visualization (e.g., electronic circuit diagrams) and visualization for the masses (e.g., road signs). Metaphoric glyphs can come in different forms, ranging from abstract representation to photographic icons (Figure 12), where the use of appropriate visual channels can provide semantic cues that are easy to learn, remember or guess. The MatchPad adopts a scale-adaptive layout to position glyphs along a timeline interactively based on the viewpoint zoom factor. This minimises glyph occlusion which they demonstrate successfully using a case study on Rugby.

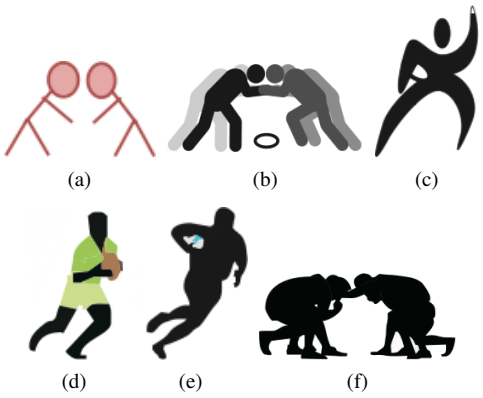


Figure 12: Some designs of metaphoric pictograms for visualizing event data in Rugby by Legg et al. [LCP*12]. In (a), initial stickmen designs were produced to prompt an artist. The artist produced several different designs: (b) a refined stickman design, (c) contemporary design, (d) a posterised colour design and (e) a silhouette design. In (f) the scrum is depicted using the silhouette design (cf. (a) and (b))

4.3. Multi-field Visualization

Due to its multivariate characteristics, geometric shapes are commonly used to represent multiple data attributes. Superquadrics and Angle-Preserving Transformations by Barr [Bar81] presents such an approach by introducing geometric shapes (superquadrics) used for creating and simulating three-dimensional scenes. The author defines a mathematical framework used to explicitly define a family of geometric primitives from which their position, size, and surface curvature can be altered by modifying a family of different parameters. Example glyphs include: a torus, star-shape, ellipsoid, hyperboloid, toroid. Furthermore, the author describes angle-preserving shape transformations that can be applied to primitives to create geometric effects such as bending or twisting.

Crawfis and Allison [CA91] introduce a novel approach for visualizing multiple scientific data sets using texture

mapping and raster operations. The interactive programming framework enables users to overlay different data sets by defining raster functions/operations. Such a function may include glyph textures for mapping data attributes (e.g., vector data). Using a generated synthetic dataset, they present a method for reducing the visual clutter by mapping colour to a height field and using a bump map to represent the vector and contour plots. The final texture is mapped onto a 3D surface.

Using the set of superquadrics defined by Barr [Bar81], Shaw et. al [SEK*98] describe an interactive glyph-based framework for visualizing multi-dimensional data. As opposed to the analytical focus in the previous work, the authors describe a method for mapping data attributes appropriately to shape properties such that visual cues effectively convey data dimensionality without depreciating the cognition of global data patterns. They map in decreasing order of data importance, values to location, size, colour and shape (of which two dimensions are encoded by shape). Using superellipsoids, they apply their framework to the "thematic" document similarities [SEK*98] and magnetohydrodynamics simulation of the solar wind in the distant heliosphere [E*00], [ES01].

The report by Taylor [Tay02] provides an overview of successful and unsuccessful techniques for visualizing multiple scalar fields on a 2D manifold. The author first hypothesises that the largest number of data sets can be displayed by mapping each field to the following: a unique surface characteristic, applying a different visualization technique to each scalar field or by using textures/glyphs whose features depend on the data sets. Such a framework revealed limitations of up to four scalar fields. This led to the research of two new techniques that prove effective for visualizing multiple scalar fields, (1) *data-driven spots (DDS)* [Bok03] - using different spots of various intensities and heights to visualize each data set, and (2) *oriented slivers* [WEL*00] - using sliver-like texture glyphs of different orientations for visualizing multiple scalar fields in which luminance is mapped to the relative scalar values.

One successful technique is developed by Kirby and Laidlaw [KML99] who stochastically arrange multiple visualization layers to minimize overlap. This paper extends the work by Laidlaw et. al [LAK*98] by applying visualization concepts from oil painting, art and design, to the problems in fluid mechanics. Given a permutation of layers, a user-specified importance value is attached to each visualization of increasing weights in order to provide greater emphasis to higher layers. Visual cues such as colour and opacity indicate regions and layers of importance (e.g., Rate of strain tensor example emphasise the velocity more by using black arrows). This method enables the simultaneous depiction of 6-9 data attributes, in which the authors apply to a simulated 2D flow field past a cylinder at different reynolds number. The example shows the visualization of velocity, vorticity, rate of strain tensor, turbulent charge and turbulent current using a series of visualization techniques such as tensor ellipses, vector arrows and colour mapping.

4.4. Geo-spatial Visualization

We often find that geo-spatial visualization may incorporate inter-disciplinary techniques from other domains, and thus can be classified under more than one category. MacEachren et al. [MBP98] is such an example where the authors present a novel approach to visualize reliability in mortality maps using a bi-variate mapping. Given a base geographical map (United States), the technique involves using colour filled regions to represent the data and texture overlay to represent the reliability.

Healey and Enns introduce a different approach [HE99] using multi-coloured perceptual texture elements known as *pexels* for visualizing multivariate scientific datasets across a height field. The pexels appearance is determined by encoding attribute values into three texture dimensions: height, density and regularity. Pexels incorporate pre-attentive features (e.g., height) to improve the accuracy of visual search-based tasks. To assess its effectiveness, the authors apply their technique on a typhoon data set where wind speed, pressure and precipitation is mapped to the pexel properties.

Pang [Pan01] provides an overview of various geo-spatial uncertainty metrics and identifies two methods for integrating this data into a geo-spatial representation: (a) mapping uncertainty information to graphic attributes (e.g., hue, opacity) or by using (b) animation to convey uncertainty. By treating uncertainty fields as an additional layer of information in cartography, techniques such as uncertainty glyphs can be visualised independently and overlaid on top of an existing geo-spatial visualization.

The work of Sanyal et al. [SZD*10] introduce glyphs, ribbons and spaghetti plots for interactively visualizing ensemble uncertainty in numerical weather models. They demonstrate their work on the 1933 Superstorm simulation, where the visual mappings illustrate the statistical errors (e.g., mean, standard deviation, interquartile range and 95% confidence intervals) in the data.

4.5. Flow Visualization

In the flow visualization community, De Leeuw and Van Wijk [dLvW93] present an interactive probe-glyph for visualizing multiple flow characteristics in a small region. One focus is the visualization of six components: velocity, curvature, shear, acceleration, torsion and convergence. In order to facilitate such a mapping, the authors incorporate a larger glyph design. The core components of the glyph consists of the following: 1) a curved vector arrow where the length and direction represents the velocity and the curvedness is mapped to the curvature, 2) a membrane perpendicular to the

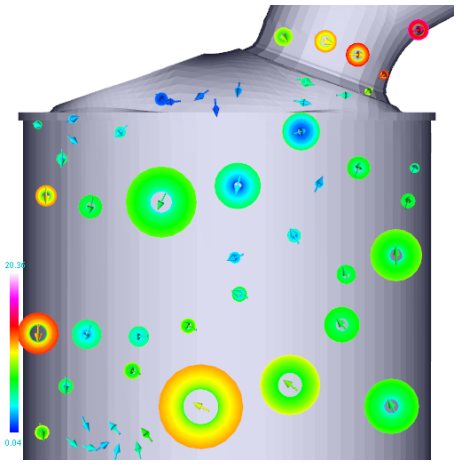


Figure 13: The visualization of vector field clustering of flow around an engine. A combination of $|v|$-range and θ-range glyph is used for depicting the range of vector magnitude and direction in each vector cluster [PGL*12]

flow where its displacement to the centre is mapped to acceleration, 3) candy stripes on the surface of the velocity arrow illustrates the amount of torsion, 4) a ring describes the plane perpendicular to the flow over time (shear-plane), and finally 5) the convergence and divergence of the flow is mapped to a *lens* or osculating paraboloid. Placement of such probes are interactively placed by the users along a streamline to show local features in more detail.

Vector Plots for Irregular Grids Dovey [Dov95] extends Crawfis and Max's method [CM92] from regular to curvilinear and unstructured grids. In order to visualize vector fields on unstructured grids, physical space and parameter space resampling methods are employed. During the physical space resampling, the vector field is linearly interpolated at each sample point, then the physical coordinates of the point are calculated, and lastly related oriented glyphs (plots) are projected from back to front. Although this ensures that sample points are uniformly distributed, physical space resampling is computationally expensive. To address this problem, it may be preferable to resample to parameter space instead. At first, random points are directly generated in parametric space with an area-weighted distribution. Then a relatively accurate and dense resampling can be approximated by mapping the parameterised coordinate to physical coordinate grid points. Vector field visualization on arbitrary 3D surfaces can be efficiently achieved with parameter space resampling.

Martin et al. [MII*08] present a study to validate the effectiveness of traditional 2D hurricane visualizations by observing the users ability to mentally integrate the magnitude and direction of flow in a vector field. In particular, the authors focus on evaluating 2D glyphs (or wind barb) - a technique commonly used for depicting wind magnitude and direction in weather visualizations. For both magnitude and direction, users had to estimate the value at a given point and estimate the average value over a rectangular region. The authors use a real hurricane simulation data set in their study.

Hlawatsch et al. [HLNW11] introduce a glyph for visualizing unsteady flow with static images. The flow-radar (glyph) is constructed by transforming time-dependant vector attributes into polar co-ordinates, whereby vector direction is mapped to angle, and the time to radius. In addition, the velocity magnitude is encoded using colour. The radar glyphs provides a visual summary of the flow over multiple time steps. A method for visualizing flow uncertainty is described using a single arc that represents the angular variation at given seed point. The authors demonstrate their work on two CFD simulation data set.

Peng et al. [PGL*12] describe an automatic vector field clustering algorithm and presents visualization techniques that incorporate statistical-based multivariate glyphs. The authors clustering algorithm is given by: 1) derive a mesh resolution value for each vertex, 2) encode vector and mesh resolution values into R, G, B and α in image space. Clusters naturally form in this space based on pixel intensity. 3) the clusters are merged depending on a similarity value derived using euclidean distance, mesh resolution, average velocity magnitude and velocity direction. A collection of clustering glyph-based visualizations are introduced, such as $|v|$-range glyph or "disc" (Figure 13 for example) that depicts the local minimum and maximum vector. The inner and outer radius of the disc is mapped to the vector magnitudes. The θ-range glyph combines a vector glyph that illustrates the average velocity direction and magnitude, and a semi-transparent cone that shows the variance of vector field direction. Other visualizations include streamlets that are traced from the cluster centre, and colour coding with mean velocity. The authors demonstrate their clustering results on a series of synthetic and real-world CFD meshes.

4.6. Tensor Visualization

The work of Laidlaw et al. [LAK*98] presents two novel methods for visualizing Diffusion Tensor Imaging (DTI). The first method uses normalised ellipsoids, where the principal axes and radii are mapped to the tensor eigen vectors and eigen values respectively. Glyph normalisation reduces the visual clutter and enables full depiction of the data set. The second method incorporates concepts from oil painting to represent seven tensor data attributes as multiple layers of varying brush strokes which is composited into a single visualization. The authors demonstrate their technique on DTIs of healthy and diseased mouse spinal cords.

Building upon previous research by Barr [Bar81] and Westin et al. [WMM*02], Kindlmann [Kin04] introduces a novel approach of visualizing tensor fields using superquadric glyphs. The motivation of superquadric tensor

glyphs addresses the problems of asymmetry and ambiguity prone in previous techniques (e.g. cuboids and ellipsoids). An explicit and implicit parameterisation of superquadric primitives is presented, along with geometric anisotropy metrics c_l, c_p, c_s [WMM*02] and user-controlled edge sharpness parameter γ, to create a barycentric triangular domain of shapes that change in shape, flatness and orientation under different parameter values. A subset of the family of superquadrics is chosen and applied towards visualizing a DT-MRI tensor field which is then compared against an equivalent ellipsoid visualization.

Kriz et al. [KYHR05] provides a review of visualization techniques on second-order tensors which include: Lame's stress ellipsoids, Haber glyphs [Hab90], Reynolds tensor glyph [HYW03], and hyper streamtubes [DH93]. Furthermore, the authors introduce a Principal, Normal and Shear (PNS) glyph for visualizing stress tensors and their gradients. The method extends the stress ellipsoids by mapping the shearing stress component to the surface colour of the ellipsoid.

Kindlmann extends his previous work [Kin04] to glyph-packing [KW06], a novel glyph placement strategy. The goal of this work is to improve upon the discrete nature of glyph-based visualization through the use of regular grid sampling, to a more continuous character such as texture-based methods by *packing* the glyphs into the field. A tensor-based potential energy is defined to derive the placement of a system of particles whose finals positions will be used to place glyphs. Hlawitschka et al. [HSH07] presents an alternative glyph packing using Delaunay triangulation which successfully reduces the computation cost.

More recently, Schultz and Kindlmann [SK10] introduce superquadric glyphs that can be used to visualize the general symmetric second order tensors that could be non-positive-definite. The work extends previous glyph-based methods (e.g., [Kin04], [WMM*02]) which concerntrate on tensors with strictly positive eigenvalues such as diffusion tensors, to the general case by mapping the glyph shape to show eigenvalue sign differences. The shape between two eignvectors is convex if the corresponding eigenvalues have the same sign, and concave if they are different.

Chen et al. [CPL*11] present a novel asymmetric tensor field visualization method to provide important insight into fluid flows and solid deformations. Existing techniques for asymmetric tensor fields focus on the analysis, and simply use evenly-spaced hyperstreamlines on surfaces following eigenvectors and dual-eigenvectors in the tensor field. They describe a hybrid visualization technique in which hyperstreamlines and elliptical glyphs are used in real and complex domains, respectively. This enables a more faithful representation of flow behaviours inside complex domains. In addition, tensor magnitude, which is an important quantity in tensor field analysis is mapped to the density of hyperstreamlines and sizes of glyphs. This allows colours to be used to encode other important tensor quantities. To facilitate quick visual exploration of the data from different viewpoints and at different resolutions, the authors employ an efficient image-space approach in which hyperstreamlines and glyphs are generated quickly in the image plane. The combination of these techniques leads to an efficient tensor field visualization system for domain scientists. They demonstrate the effectiveness of their visualization technique through applications of complex simulated engine fluid flow and earthquake deformation data.

4.7. Uncertainty Visualization

A number of approaches have been used to quantify and visualize uncertainty. In particular, glyphs are well suited for illustrating uncertainty, detailed by the early work of Wittenbrink et al. [WPL96] who evaluates the effective use of glyphs for visualizing uncertainty in vector fields simulated from winds and ocean currents. Several uncertainty metrics are depicted simultaneously such as direction, magnitude as well as mean direction and length using a variety of glyph attributes that are commonly mapped (e.g., length, area, colour and/or angles). Lodha et al. [LPSW96] presents a system (UFLOW) for visualizing uncertainties in fluid flow. The system analyses the changes that occur from different integrators and step-sizes used for computing streamlines. The authors visualize the differences between each streamlines using several approaches such as glyphs that encode the uncertainty through their shape, size and colour.

Pang et al. [PWL96] and Verma and Pang [VP04] present comparative visualization tools to analyse differences between two datasets. Streamlines and stream ribbons are generated on two datasets, one being a sub-sampled version of the other. To compare streamlines, the euclidean distance between them is used. Glyphs are added to aid the user in seeing how a pair of streamlines differ. Brown [Bro04] demonstrates the use of vibrations to visualize data uncertainty. Experiments using oscillations in vertex displacement, and changes in luminance and hue are investigated.

MacEachren et al. [MRO*12] is another instance of empirical research that evaluates the effects of visualizing different categories of uncertainty using discrete symbols. Building upon the theoretical framework by Bertin [Ber83] on visual semiotics, they provide insight on the effects of using abstract symbols that vary only a single visual variable (e.g., shape, hue, orientation) in comparison to iconic symbols that are of more pictorial form. Both sets of symbols underwent two distinct experiments which focus on assessing their *intuitiveness* for representing different categories of uncertainty and *effectiveness* for a typical map use task: assessing and comparing the aggregate uncertainty in two map regions.

Ribicic et al. [RWG*12] describe an interactive sketch-based visualization system for investigating simulation mod-

els and assess the uncertainty associated with changing different numeric parameters. In particular, the authors demonstrate their approach on flood management simulation as a means of risk assessment. Such an approach provides an intuitive mechanism for transforming sketches into boundary conditions of a simulation and to deliver visual feedback to end-users. A set of glyphs and icons are used to depict various simulation attributes. These include vector glyphs for illustrating the force field on a water flow and ensemble handle glyphs for representing uncertainty values.

5. Summary and Conclusions

In this state of the art report, we have presented a comprehensive survey of the subject of glyph-based visualization. In particular, we have made connection between glyphs and the history of signs and perceptual studies on visual channels and the use of icons. We have brought together a substantial collection of design criteria and guidelines. We have examined a variety of methods and algorithms for visual mapping, computing spatial layout, rendering glyphs and supporting glyph-based interaction. Noticeably, we have gathered an indisputable set of evidence in different applications, suggesting that glyph-based visualization is useful and can bring about cost-effective benefits in many data-intensive tasks.

While this survey has confirmed that glyph-based visualization is an important technique in the field of visualization, we have also observed some doubts in the community about the encoding capability of glyphs primarily due to its size, limited capacity of individual visual channels and cognitive demand for learning and memorization. Although such reservations are very reasonable and cannot be overlooked in any practical applications, they do not undermine the relative merits of glyph-based visualization, which have already been demonstrated in everyday life as well as many applications. These merits include:

- rapid semantic interpretation (e.g., traffic signs [LCP*12]),
- more scalability in dimensions for multivariate data visualization (e.g., [Kin04, SK10]),
- suitable for both dense and sparse layout (e.g., [KW06, LKH09, LHD*04, LCP*12]),
- no significant disadvantage in learning and memorization (e.g., [MdBC00]),
- can be evolved into a standard form (e.g., many schematic diagrams),
- can be evolved into a language (e.g., grapheme-based languages).

This survey has also helped us identify major gaps in the current research on glyph-based visualization. While existing perception studies on visual channels and icons have provided a concrete foundation for glyph-based visualization, most findings are directly applicable only to glyph representations in three or fewer dimensions. We hence would like to encourage more empirical studies on high dimensional glyph representations. As we mentioned in Section 1, glyph-based visualization essentially offers a form of dictionary-based compression. Naturally, this allows us to draw inspiration, theories and techniques from established disciplines, such as data communication and historical linguistics, and research subjects such as information theory, data compression, and lexicography. Perhaps more ambitiously, the field of visualization may channel more energy and innovation into the development of a common framework, which may one day become the basis of a common visualization language.

6. Acknowledgmenets

The authors wish to thank all participants of the Workshop on Glyph-based Visualization, which was held on 7-8 May 2012, and in particular the keynote speakers of the workshop, Professor Colin Ware (University of New Hampshire, USA) and Professor Sine McDougall (Bournemouth University, UK). Parts of this work have also been funded by the Austrian Science Fund (FWF) through the ViMaL project (No. P21695).

References

[ABK98] ANKERST M., BERCHTOLD S., KEIM D.: Similarity clustering of dimensions for an enhanced visualization of multidimensional data. In *Information Visualization, 1998. Proceedings. IEEE Symposium on* (oct 1998), pp. 52 –60, 153. 11

[Bar81] BARR A. H.: Superquadrics and angle-preserving transformations. *IEEE Computer Graphics and Applications 1*, 1 (Jan. 1981), 11–23. 14, 17, 18, 19

[Bar04] BAR M.: Visual objects in context. *Nature reviews. Neuroscience 5*, 8 (2004), 617–629. 8

[BARM*12] BORGO R., ABDUL-RAHMAN A., MOHAMED F., GRANT P. W., REPPA I., FLORIDI L., CHEN M.: An empirical study on using visual embellishments in visualization. *to appear in IEEE Transactions on Visualization and Computer Graphics* (2012). 8

[BBS*08] BOTCHEN R. P., BACHTHALER S., SCHICK F., CHEN M., MORI G., WEISKOPF D., ERTL T.: Action-based multifield video visualization. *IEEE Trans. Visualization and Computer Graphics 14*, 4 (2008), 885–899. 16

[Ber83] BERTIN J.: *Semiology of graphics*. Univ. of Wisconsin Press, 1983. 5, 8, 11, 20

[Bly82] BLY S.: Presenting information in sound. In *Proceedings of the 1982 conference on Human factors in computing systems* (1982), CHI '82, ACM, pp. 371–375. 3

[Bok03] BOKINSKY A. A.: Multivariate data visualization with data-driven spots. 18

[Bre99] BREWER C.: Color use guidelines for data representation. In *Proc. Section on Statistical Graphics* (1999), pp. 55–60. 15

[Bro04] BROWN R.: Animated visual vibrations as an uncertainty visualisation technique. In *Proceedings of the 2nd international conference on Computer graphics and interactive techniques in Australasia and South East Asia* (2004), GRAPHITE '04, ACM, pp. 84–89. 20

[BS92] BORG I., STAUFENBIEL T.: Performance of snow flakes, suns, and factorial suns in the graphical representation of multivariate data. *Multivariate Behavioral Research 27*, 1 (1992), 43–55. 11

[BSG89] BLATTNER M. M., SUMIKAWA D. A., GREENBERG R. M.: Earcons and icons: Their structure and common design principles. *Human-Computer Interaction 4*, 1 (1989), 11–44. 3

[BSMWE78] BURNS B., SHEPP B. E., MCDONOUGH D., WIENER-EHRLICH W. K.: The relation between stimulus analyzability and perceived dimensional structure. *The Psychology of Learning and Motivation* (1978), 77–115. 8

[BT07] BORLAND D., TAYLOR R.: Rainbow color map (still) considered harmful. *IEEE Computer Graphics and Applications 27*, 2 (2007), 14–17. 13

[CA91] CRAWFIS R., ALLISON M. J.: A scientific visualization synthesizer. In *IEEE Visualization* (1991), pp. 262–267. 17

[CF12] CHEN M., FLORIDI L.: An analysis of information in visualisation. *to appear in Synthese* (2012). 7

[Cha02] CHANDLER D.: *Semiotics: The Basics*. Routledge, 2002. 4

[Che73] CHERNOFF H.: The use of faces to represent points in k-dimensional space graphically. *J. American Statistical Association 68*, 342 (1973), 361–368. 14, 16

[Chr75] CHRIST R.: Review and analysis of color coding research for visual displays. *Human Factors 17*, 6 (1975), 542–570. 6, 13

[CLP*13] CHUNG D. H. S., LEGG P. A., PARRY M. L., BOWN R., GRIFFITHS I. W., LARAMEE R. S., CHEN M.: Glyph sorting: Interactive visualization for multi-dimensional data (under review). 8, 9, 16

[CM84a] CLEVELAND W., MCGILL R.: Graphical perception: Theory, experimentation, and application to the development of graphical methods. *J. American Statistical Association 79*, 387 (1984), 531–554. 6, 14

[CM84b] CLEVELAND W., MCGILL R.: Graphical perception: Theory, experimentation, and application to the development of graphical methods. *J. American Statistical Association 79*, 387 (1984), 531–554. 7

[CM92] CRAWFIS R., MAX N.: Direct volume visualization of three-dimensional vector fields. In *Proceedings of the Workshop on Volume Visualization* (New York, Oct. 19–20 1992), Acm Press, pp. 55–60. 19

[CM93] CRAWFIS R., MAX N.: Texture splats for 3D scalar and vector field visualization. In *Proc. IEEE Visualization* (1993), pp. 261–266. 11, 14, 15

[CPL*11] CHEN G., PALKE D., LIN Z., YEH H., VINCENT P., LARAMEE R. S., ZHANG E.: Asymmetric tensor field visualization for surfaces. *IEEE Transactions on Visualization and Computer Graphics 17*, 6 (2011). 20

[CWD*02] CERQUEIRA M. D., WEISSMAN N. J., DILSIZIAN V., ET AL.: Standardized myocardial segmentation and nomenclature for tomographic imaging of the heart. *Circulation 105*, 4 (2002), 539–542. 16

[DH93] DELMARCELLE T., HESSELINK L.: Visualization of second order tensor fields with hyperstreamlines. *IEEE Computer Graphics and Applications 13*, 4 (1993), 25–33. 20

[dLvW93] DE LEEUW W. C., VAN WIJK J. J.: A probe for local flow field visualization. In *Proc. IEEE Visualization* (1993), pp. 39–45. 11, 15, 18

[Dov95] DOVEY D.: Vector plots for irregular grids. In *IEEE Visualization* (1995), pp. 248–253. 19

[E*00] EBERT D. S., ET AL.: Procedural shape generation for multi-dimensional data visualization. *Computers and Graphics 24*, 3 (June 2000), 375–384. 18

[Eco79] ECO U.: *A theory of semiotics*. Advances in semiotics. Indiana University Press, 1979. 4, 7

[EM91] ELIADE M., MAIRET P.: *Images and symbols: studies in religious symbolism*. Mythos: The Princepto/Bollingen Series in World Mythology. Princeton University Press, 1991. 2

[ES01] EBERT D. S., SHAW C. D.: Minimally immersive flow visualization. In *IEEE Transactions on Visualization and Computer Graphics* (2001), vol. 7(4), IEEE Computer Society, pp. 343–350. 18

[FK03] FRIENDLY M., KWAN E.: Effect ordering for data displays. *Comput. Stat. Data Anal. 43*, 4 (Aug. 2003), 509–539. 11

[Gay89] GAYER W. W.: The sonicfinder: An interface that uses auditory icons (abstract only). *SIGCHI Bull. 21* (August 1989). 3

[Gog03] GOGUEN J.: Semiotic morphisms, representations, and blending for user interface design. In *AMAST Proceedings* (2003), pp. 1–15. 5

[Gre98] GREEN M.: Toward a perceptual science of multidimensional data visualization: Bertin and beyond. *ERGO/GERO Human Factors Science* (1998), 1–30. 8

[Hab90] HABER R. B.: Visualization techniques for engineering mechanics. *Computing Systems in Engineering 1*, 1 (1990), 37–50. 20

[HBE96] HEALEY C., BOOTH K., ENNS J.: High-speed visual estimation using preattentive processing. *ACM Trans. Computer-Human Interaction 3* (1996), 107–135. 6, 14

[HE99] HEALEY C., ENNS J.: Large datasets at a glance: Combining textures and colors in scientific visualization. *IEEE Trans. Visualization and Computer Graphics 5*, 2 (1999), 145–167. 14, 18

[HE11] HEALEY C. G., ENNS J. T.: Attention and visual memory in visualization and computer graphics. *IEEE Transactions on Visualization and Computer Graphics 18*, 7 (2011), 1170–1188. 8

[HI72] HANDELT S., IMAI S.: The free classification of analyzable and unanalyzable stimuli. *Attention, Perception, & Psychophysics 12*, 1 (1972), 108–224. 8

[HLNW11] HLAWATSCH M., LEUBE P., NOWAK W., WEISKOPF D.: Flow radar glyphs - static visualization of unsteady flow with uncertainty. In *IEEE Transactions on Visualization and Computer Graphics* (2011), vol. 17, pp. 1949–1958. 19

[HS09] HAUSER H., SCHUMANN H.: Visualization pipeline. In *Encyclopedia of Database Systems*, Liu, Özsu, (Eds.). Springer, 2009, pp. 3414–3416. 10

[HSH07] HLAWITSCHKA M., SCHEUERMANN G., HAMANN B.: Interactive glyph placement for tensor fields. In *Advances in Visual Computing* (2007), pp. I: 331–340. 20

[HYW03] HASHASH Y. M. A., YAO J. I.-C., WOTRING D. C.: Glyph and hyperstreamline representation of stress and strain tensors and material constitutive response. *International Journal for Numerical and Analytical Methods in Geomechanics 27*, 7 (2003), 603–626. 20

[IG98] INTERRANTE V., GROSCH C.: Visualizing 3D flow. *IEEE Computer Graphics and Applications 18*, 4 (1998), 49–53. 15

[JL02] JOHANSEN J., LARSEN S.: *Signs in use: an introduction to semiotics*. Routledge, 2002. 3, 4

[Joh88] JOHANSEN J.: *The Distinction between Icon, Index, and Symbol in the Study of Literature*. Berlin: Walter de Gruyter, 1988. 5

[KE01] KRAUS M., ERTL T.: Interactive data exploration with customized glyphs. In *Proc. WSCG* (2001), pp. 20–23. 14

[KG07] KARVE A., GLEICHER M.: Glyph-based overviews of large datasets in structural bioinformatics. In *Proceedings of the 11th International Conference on Information Visualization (IV 2007)* (jul 2007), pp. 1–6. 9

[Kin04] KINDLMANN G.: Superquadric tensor glyphs. In *Proc. Symp. Data Visualization (VisSym)* (2004), pp. 147–154. 14, 19, 20, 21

[KMDH11] KEHRER J., MUIGG P., DOLEISCH H., HAUSER H.: Interactive visual analysis of heterogeneous scientific data across an interface. *IEEE Trans. Visualization and Computer Graphics 17*, 7 (2011), 934–946. 15

[KML99] KIRBY R., MARMANIS H., LAIDLAW D.: Visualizing multivalued data from 2D incompressible flows using concepts from painting. In *Proc. IEEE Visualization* (1999), pp. 333–340. 11, 18

[KT75] KRANTZ D., TVERSKY A.: Similarity of rectangles: An analysis of subjective dimensions. *Journal of Mathematical Psychology 12*, 1 (1975), 4–34. 8

[KW79] KINCHLA R., WOLFE J.: The order of visual processing: "top-down," "bottom-up", or "middle-out". *Perception & psychophysics 25*, 3 (1979). 8

[KW06] KINDLMANN G., WESTIN C.-F.: Diffusion tensor visualization with glyph packing. *IEEE Trans. Visualization and Computer Graphics 12*, 5 (2006), 1329–1336. 11, 15, 20, 21

[KYHR05] KRIZ R. D., YAMAN M., HARTING M., RAY A. A.: Visualization of Zeroth, Second, Fourth, Higher Order Tensors, and Invariance of Tensor Equations. *Computers and Graphics 21*, 6 (2005), 1–13. 20

[Lak95] LAKOFF G.: The Contemporary Theory of Metaphor. *Metaphor and Thought, A. Ortony (2nd Ed.)* (1995), 202,252. 7

[LAK*98] LAIDLAW D. H., AHRENS E. T., KREMERS D., AVALOS M. J., JACOBS R. E., READHEAD C.: Visualizing diffusion tensor images of the mouse spinal cord. In *IEEE Visualization* (1998), pp. 127–134. 15, 18, 19

[LCP*12] LEGG P. A., CHUNG D. H. S., PARRY M. L., JONES M. W., LONG R., GRIFFITHS I. W., CHEN M.: Matchpad: Interactive glyph-based visualization for real-time sports performance analysis. *Computer Graphics Forum 31*, 3pt4 (2012), 1255–1264. 15, 17, 21

[LHD*04] LARAMEE R., HAUSER H., DOLEISCH H., VROLIJK B., POST F., WEISKOPF D.: The state of the art in flow visualization: Dense and texture-based techniques. *Computer Graphics Forum 23*, 2 (2004), 203–221. 2, 21

[LKH09] LIE A. E., KEHRER J., HAUSER H.: Critical design and realization aspects of glyph-based 3D data visualization. In *Proc. Spring Conference on Computer Graphics (SCCG 2009)* (2009), pp. 27–34. 2, 10, 12, 13, 14, 15, 21

[LMvW10] LI J., MARTENS J.-B., VAN WIJK J. J.: A model of symbol size discrimination in scatterplots. In *Proc. Human Factors in Computing Systems (CHI'10)* (2010), pp. 2553–2562. 6, 13

[LPSW96] LODHA S. K., PANG A., SHEEHAN R. E., WITTENBRINK C. M.: UFLOW: Visualizing uncertainty in fluid flow. In *IEEE Visualization* (1996), pp. 249–254. 20

[LRW99] LOVE B., ROUDER J., WISNIEWSKI E.: A structural account of global and local processing. *Cognitive psychology 38*, 2 (1999), 291–607. 8

[Mac86] MACKINLAY J.: Automating the design of graphical presentations of relational information. *ACM Trans. Graph. 5*, 2 (Apr. 1986), 110–141. 5, 7

[Mac04] MACEACHREN A. M.: *How Maps Work - Representation, Visualization, and Design*. Guilford Press, 2004. 5

[Mar03] MARCUS A.: Icons, symbols, and signs: visible languages to facilitate communication. *interactions 10* (May 2003), 37–43. 3

[MBP98] MACEACHREN A. M., BREWER C. A., PICKLE L. W.: Visualizing georeferenced data: Representing reliability of health statistics. *Evironment and Planning A* (1998), 1547–1561. 18

[MCdB99] MCDOUGALL S., CURRY M., DE BRUIJN O.: Measuring symbol and icon characteristics: Norms for concreteness, complexity, meaningfulness, familiarity, and semantic distance for 239 symbols. *Behavior Research Methods 31* (1999), 487–519. 9

[MdBC00] MCDOUGALL S., DE BRUIJN O., CURRY M.: Exploring the effects of icon characteristics on user performance: The role of icon concreteness, complexity, and distinctiveness. *Journal of Experimental Psychology: Applied 6* (2000), 291–306. 8, 9, 10, 21

[MII*08] MARTIN J. P., II J. E. S., II R. J. M., LIU Z., CAI S.: Results of a user study on 2D hurricane visualization. *Computer Graphics Forum 27*, 3 (2008), 991–998. 19

[MRO*12] MACEACHREN A. M., ROTH R. E., O'BRIEN J., LI B., SWINGLEY D., GAHEGAN M.: Visualizing semiotics & uncertainty visualization. *IEEE Trans. Visualization and Computer Graphics 18*, 12 (2012), 2496–2505. 20

[MRSS*12] MAGUIRE E., ROCCA-SERRA P., SANSONE S.-A., DAVIES J., CHEN M.: Taxonomy-based Glyph Design – with a Case Study on Visualizing Workflows of Biological Experiments. *to appear in IEEE Transactions on Visualization and Computer Graphics 18* (2012), –. 8, 10, 14, 15

[MSSD*08] MEYER-SPRADOW J., STEGGER L., DÖRING C., ROPINSKI T., HINRICHS K.: Glyph-based SPECT visualization for the diagnosis of coronary artery disease. *IEEE Transactions on Visualization and Computer Graphics 14*, 6 (2008), 1499–1506. 15, 16

[MTL78] MCGILL R., TUKEY J., LARSEN W.: Variations of box plots. *The American Statistician 32* (1978), 12–16. 12

[Nav77] NAVON D.: Forest before trees: The precedence of global features in visual perception. *Cognitive psychology 9*, 3 (1977), 353–736. 8

[NC08] NG A. W., CHAN A. H.: Visual and cognitive features on icon effectiveness. *Lecture Notes in Engineering and Computer Science 2169* (2008), 1856Ú1859. 10

[Nor02] NORMAN D. A.: *The Design of Everyday Things*. Basic Books, 2002. 8

[OHG*08] OELTZE S., HENNEMUTH A., GLASSER S., KÜHNEL C., PREIM B.: Glyph-based visualization of myocardial perfusion data and enhancement with contractility and viability information. In *Proceedings of the First Eurographics conference on Visual Computing for Biomedicine* (2008), EG VCBM'08, pp. 11–20. 16

[Ort93] ORTONY A.: *Metaphor and thought*. Cambridge University Press, 1993. 7

[Pal77] PALMER S.: Hierarchical structure in perceptual representation. *Cognitive Psychology 9*, 4 (1977), 441–915. 8

[Pan01] PANG A.: Visualizing uncertainty in geo-spatial data. In *In Proceedings of the Workshop on the Intersections between Geospatial Information and Information Technology* (2001). 18

[PB55] PEIRCE C., BUCHLER J. E.: *The Philosophical Writings of Peirce*. New York: Dover, 1955. 3, 5

[Pei02] PEIRCE C.: *The New Elements of Mathematics*. 1902. 3, 10

[Pet10] PETTERSSON R.: Information design−principles and guidelines. *Journal of Visual Literacy 29*, 2 (2010), 167–182. 6, 9

[PG88] PICKETT R., GRINSTEIN G.: Iconographic displays for visualizing multidimensional data. In *Systems, Man, and Cybernetics, 1988. Proceedings of the 1988 IEEE International Conference on* (1988), pp. 514–519. 11, 15

[PGL*12] PENG Z., GRUNDY E., LARAMEE R., CHEN G., CROFT N.: Mesh-driven vector field clustering and visualization: An image-based approach. *IEEE Transactions on Visualization and Computer Graphics (TVCG) 18*, 5 (2012), 283–298. 19

[PKH04] PIRINGER H., KOSARA R., HAUSER H.: Interactive focus+context visualization with linked 2D/3D scatterplots. In *Proc. Coordinated & Multiple Views in Exploratory Visualization (CMV)* (2004), pp. 49–60. 15

[PL09] PENG Z., LARAMEE R. S.: Higher dimensional vector field visualization: A survey. In *TPCG* (2009), pp. 149–163. 2

[PLC*11] PARRY M. L., LEGG P. A., CHUNG D. H. S., GRIFFITHS I. W., CHEN M.: Hierarchical event selection for video storyboards with a case study on snooker video visualization. *IEEE Transactions on Visualization and Computer Graphics 17* (2011), 1747–1756. 17

[PR07] PEARLMAN J., RHEINGANS P.: Visualizing network security events using compound glyphs from a service-oriented perspective. In *Proceedings of the Workshop on Visualization for Computer Security* (2007), pp. 131–146. 17

[PVH*03] POST F., VROLIJK B., HAUSER H., LARAMEE R., DOLEISCH H.: The state of the art in flow visualisation: Feature extraction and tracking. *Computer Graphics Forum 22*, 4 (2003), 775–792. 2

[PWL96] PANG A., WITTENBRINK C., LODHA S.: *APPROACHES TO UNCERTAINTY VISUALIZATION*. Technical Report UCSC-CRL-96-21, University of California, Santa Cruz, Jack Baskin School of Engineering, Sept. 1996. 20

[PWR04] PENG W., WARD M., RUNDENSTEINER E.: Clutter reduction in multi-dimensional data visualization using dimension reordering. In *Proc. IEEE Symp. Information Visualization* (2004), pp. 89–96. 11, 14

[QH87] QUINLAN P., HUMPHREYS G.: Visual search for targets defined by combinations of color, shape, and size: an examination of the task constraints on feature and conjunction searches. *Perception & psychophysics 41*, 5 (1987), 455–527. 7, 8

[RAEM94] RIBARSKY W., AYERS E., EBLE J., MUKHERJEA S.: Glyphmaker: Creating customized visualizations of complex data. *IEEE Computer*, 7 (1994), 57–64. 14

[ROP11] ROPINSKI T., OELTZE S., PREIM B.: Survey of glyph-based visualization techniques for spatial multivariate medical data. *Computers & Graphics 35*, 2 (2011), 392–401. 2, 7, 8, 10, 12, 13, 14, 15, 16

[RP08] ROPINSKI T., PREIM B.: Taxonomy and usage guidelines for glyph-based medical visualization. In *Proc. Simulation and Visualization (SimVis)* (2008), pp. 121–138. 2, 10, 11, 16

[RSMS*07] ROPINSKI T., SPECHT M., MEYER-SPRADOW J., HINRICHS K., PREIM B.: Surface glyphs for visualizing multimodal volume data. In *Proc. Vision, Modeling, and Visualization (VMV)* (2007), pp. 3–12. 11, 12, 13, 15

[RTB96] ROGOWITZ B. E., TREINISH L. A., BRYSON S.: How not to lie with visualization. *Comput. Phys. 10* (1996), 268–273. 12

[Rum70] RUMELHART D. E.: A multicomponent theory of the perception of briefly exposed visual displays. *Journal of Math. Psych. 7*, 2 (1970), 191–218. 8

[RWG*12] RIBICIC H., WASER J., GURBAT R., SADRANSKY B., GROLLER M. E.: Sketching uncertainty into simulations. *IEEE Trans. Visualization and Computer Graphics 18*, 12 (2012), 2255–2264. 20

[SBSR83] SAUSSURE F., BALLY C., SECHEHAYE A., RIEDLINGER A.: *Course in general linguistics*. Open Court Classics. Open Court, 1983. 5

[SEK*98] SHAW C. D., EBERT D. S., KUKLA J. M., ZWA A., SOBOROFF I., ROBERTS D. A.: Data visualization using automatic, perceptually-motivated shapes. In *In Proceedings of SPIE 3298, Visual Data Exploration and Analysis V* (apr 1998), pp. 208–213. 18

[SFGF72] SIEGEL J., FARRELL E., GOLDWYN R., FRIEDMAN H.: The surgical implication of physiologic patterns in myocardial infarction shock. *Surgery 72* (1972), 27–35. 16

[She64] SHEPARD R.: Attention and the metric structure of the stimulus space. *Journal of Mathematical Psychology 1*, 1 (1964), 54–141. 8

[Sii02] SIIRTOLA H.: The effect of data-relatedness in interactive glyphs. *In Proc. 9th Int. Conf. Information Visualization* (2002), 869–876. 8

[SK10] SCHULTZ T., KINDLMANN G.: Superquadric glyphs for symmetric second-order tensors. *IEEE Trans. Visualization and Computer Graphics 16*, 6 (2010), 1595–1604. 20, 21

[STH02] STOLTE C., TANG D., HANRAHAN P.: Polaris: A system for query, analysis, and visualization of multidimensional relational databases. *IEEE Trans. Visualization and Computer Graphics 8*, 1 (2002), 52–65. 12

[SZD*10] SANYAL J., ZHANG S., DYER J., MERCER A., AMBURN P., MOORHEAD R.: Noodles: A tool for visualization of numerical weather model ensemble uncertainty. *IEEE Transactions on Visualization and Computer Graphics 16*, 6 (Nov./Dec. 2010), 1421–1430. 18

[Tay02] TAYLOR R.: Visualizing multiple fields on the same surface. *IEEE Computer Graphics and Applications 22*, 3 (2002), 6–10. 18

[TG88] TREISMAN A., GORMICAN S.: Feature analysis in early vision: evidence from search asymmetries. *Psychol Rev 95*, 1 (Jan 1988), 15–48. 8

[Tou97] TOUTIN T.: Qualitative aspects of chromo-stereoscopy for depth-perception. *Photogrammetric Engineering and Remote Sensing 63*, 2 (1997), 193–203. 15

[Tre99] TREINISH L.: Task-specific visualization design. *IEEE Computer Graphics and Applications 19*, 5 (1999), 72–77. 11, 15

[TS82] TOURANGEAU R., STERNBERG R. J.: Understanding and appreciating metaphors. *Cognition* (1982), 203–244. 7

[TSWS05] TOMINSKI C., SCHULZE-WOLLGAST P., SCHUMANN H.: 3D information visualization for time-dependent data on maps. In *Proc. Int'l. Conf. Information Visualization (IV '05)* (2005), pp. 175–181. 11

[VP04] VERMA V., PANG A.: Comparative flow visualization. *IEEE Transactions on Visualization and Computer Graphics 10*, 6 (Nov./Dec. 2004), 609–624. 20

[War02] WARD M.: A taxonomy of glyph placement strategies for multidimensional data visualization. *Information Visualization 1*, 3/4 (2002), 194–210. 10, 12, 15

[War04] WARE C.: *Information Visualization: Perception for Design*, second edition ed. Morgan Kaufmann, 2004. 14

[War08] WARD M.: Multivariate data glyphs: Principles and practice. In *Handbook of Data Visualization*, Chen C.-H., Härdle W., Unwin A., (Eds.), Springer Handbooks Comp. Statistics. Springer, 2008, pp. 179–198. 2, 10, 11, 13, 14, 15

[WB97] WONG P. C., BERGERON R. D.: Multivariate visualization using metric scaling. In *Vis1997* (1997), pp. 111–118. 15

[WCG94] WANG Q., CAVANAGH P., GREEN M.: Familiarity and pop-out in visual search. *Perception & psychophysics 56*, 5 (Nov 1994), 495–500. 8

[WEL*00] WEIGLE C., EMIGH W. G., LIU G., TAYLOR R. M., ENNS J. T., HEALEY C. G.: Oriented sliver textures: A technique for local value estimation of multiple scalar fields. In *Graphics Interface* (2000), Canadian Human-Computer Communications Society, pp. 163–170. 18

[WG11] WARD M., GUO Z.: Visual exploration of time-series data with shape space projections. *Computer Graphics Forum 30*, 3 (2011), 701–710. 15

[Wil67] WILLIAMS L.: The effects of target specification on objects fixated during visual search. *Acta psychologica 27* (1967), 355–415. 7, 8

[WMM*02] WESTIN C. F., MAIER S. E., MAMATA H., NABAVI A., JOLESZ F. A., KIKINIS R.: Processing and visualization for diffusion tensor MRI. *Medical Image Analysis 6*, 2 (2002), 93–108. 16, 19, 20

[WPL96] WITTENBRINK C. M., PANG A., LODHA S. K.: Glyphs for visualizing uncertainty in vector fields. *IEEE Trans. Vis. Comput. Graph 2*, 3 (1996), 266–279. 20

[You01] YOUSEF M. K.: Assessment of metaphor efficacy in user interfaces for the elderly: a tentative model for enhancing accessibility. In *Proceedings of the 2001 EC/NSF workshop on Universal accessibility of ubiquitous computing: providing for the elderly* (2001), WUAUC'01, ACM, pp. 120–124. 7

D Copy of [ARLC+13]:

Rule-based Visual Mappings – with a Case Study on Poetry Visualization

Eurographics Conference on Visualization (EuroVis) 2013
B. Preim, P. Rheingans, and H. Theisel
(Guest Editors)

Volume 32 (2013), Number 3

Rule-based Visual Mappings – with a Case Study on Poetry Visualization

A. Abdul-Rahman[1], J. Lein[2], K. Coles[2], E. Maguire[1], M. Meyer[3], M. Wynne[1,4], C. R. Johnson[3], A. Trefethen[4], and M. Chen[1]

[1]Oxford e-Research Centre, and [4]IT Services, University of Oxford
[2]Department of English, and [3]Scientific Computing and Imaging Institute, University of Utah

Abstract

In this paper, we present a user-centered design study on poetry visualization. We develop a rule-based solution to address the conflicting needs for maintaining the flexibility of visualizing a large set of poetic variables and for reducing the tedium and cognitive load in interacting with the visual mapping control panel. We adopt Munzner's nested design model to maintain high-level interactions with the end users in a closed loop. In addition, we examine three design options for alleviating the difficulty in visualizing poems latitudinally. We present several example uses of poetry visualization in scholarly research on poetry.

1. Introduction

A poem is a complex dynamic system. It features a variety of structural and relational information, including *formal information* (e.g., *lines*, *stanzas*), *phonetic information* (e.g., *meter*, *intonation*, *timing*), and *semantic information* (e.g., *genres*, *words*, *repetition*, *sentiment*). In addition, poems are studied from many different angles. Specific lenses or contexts for analysis might include the body of a poet's work, a historical period, a nation or geographic location, a group or movement, and so on. Although a poem is not a big data set, poets frequently devote hours to a close reading of a poem. In many ways, a *close reading* is a literary form of "data exploration", in which scholars pay close attention simultaneously to various individual linguistic, literary and sociological features, as enumerated above, as well as to the interplay and relationships among these features.

This paper is concerned with a user-centered design study on the visualization of poems. It could be said that one of the largest "professional gulfs" possible exists between computer scientists and poets. To a computer scientist, a poem appears to be an ordered sequence of letters – and may be considered as a multivariate data set. From this point of view, it would seem that one might define a set of variables within the complex information space of a poem and then employ a multivariate visualization technique such as parallel coordinate plots to discover various structures. Poets, on the other hand, insist that finding and reasoning through structural and relational information in a poem is at the heart of their close reading practice. This process is complicated, time consuming, and inherently experiential; the result is that interpretations vary from one person (or one context) to another. As one of the poets in this project explained, "Close reading involves moment-by-moment choices, where every choice activates some possibilities and deactivates others".

Such a "professional gulf" presented us with an interesting challenge and motivated us to demonstrate the potentially universal power of visualization by developing a suitable tool to help poets to explore the information spaces of poems. Following the nested design process [Mun09], the visualization researchers and the literary scholars formed a closed loop and engaged in a variety of collaborative activities, including joint workshops, brainstorming meetings, joint literature studies, close reading observations and recordings, questionnaires, requirements analyses, design evolution, and discussion of visualization results (Sections 3-6). In addition, one of the visualization researchers and one of the humanities scholars shared an office throughout the project.

We realized our technical solution would need to accommodate two potentially conflicting requirements: it would need to address a large number of variables (poetic attributes) while providing readers with the ability to make "moment-by-moment choices" as they explored visualizations. Inspired by the concept of the "artificial assistant" presented in [MHS07], we formulated a number of rules for reducing the options of visual mappings in an automatic and

context-sensitive manner (Section 4). We designed and developed a rule-based interface for facilitating poem visualization (Section 5). We also developed and experimented with three different techniques for alleviating the difficulty in visualizing poems latitudinally (Section 6). This study showed that a rule-based approach shows promise in addressing the dual requirements mentioned above; it helped us make significant progress in developing visualization tools useful to close readers of poetry.

2. Related Work

In this section, we give a brief overview of the related work in *visual literature analysis* and other aspects of visualization, such as design process and visual encoding.

Visual Literature Analysis. There are many tools and techniques for exploring texts and documents. Some support the examination of the relationship between words using graphs (e.g., [Mil95, WV08, vHWV09, Pal]), while others use a radial, space-filling representation [CCP09]. A number of techniques (e.g., pixel-based visualization [KO07, OBK*08], word cloud [VCPK09], and inkblots [AC07]) focus on viewing categorical information and statistical information about a document, revealing various signature patterns for comparative analysis. These techniques are designed primarily for *distant reading* [Mor05]. They are effective in many text analysis applications (e.g., [KO07, DZG*07]), enabling examination of a large amount of text as a whole. For literary *close reading*, however, such approaches appear to offer limited scope and flexibility in supporting specific users' unique explorations. They neglect or ignore textual characteristics that poets and close readers value highly namely the complex and often subtle dynamic interplay among poetic features as they exist and develop in time and within particular contexts.

Our work addresses this gap by providing a visual and information-rich medium to literary scholars for exploring a large number of variables in a poem. While other programs [Pla06,CAT*12] identify sonic structures and patterns to analyze, they do not capture or convey individual texts' dynamic complexity. The closest existing tool to ours is Myopia [CGM*12], a poetry visualization tool. The main difference between our work and Myopia is our support for visualizing many more poetic variables, such as dynamic temporality and relational data, and our use of rule-based mapping for visual encoding.

Other Aspects of Visualization. Our work builds on the research of others, including particularly relevant aspects of the following projects:

Mackinlay *et al.* [MHS07] presents a system that incorporates a default set of user interface commands for generating an effective graphical visualization. They also explore the issue of user experience in automatic presentation.

Munzner [Mun09] presents a four-level model for creating and evaluating a visualization system, where the output of each level is the input for the next one. Munzner examines a few visualization systems, such as visualization of genealogical graphs [MB05] and *MatrixExplorer* system [HF06], that utilize a nested approach in their development. Other frameworks proposed for designing and evaluating visualization systems include [AS04], [LPP*06] and [VPF06]. Our tool adopts the nested design model [Mun09] as it allows us to engage on a high-level with the end users, while including their feedback in every level of the design process, thereby minimizing some of the drawbacks of evaluation.

There is a rich collection of work on visual encoding. For example, Bertin [Ber83] discusses visual encoding from a cartographer's point of view. Ware [War12] provides guidelines layout and design practices in data visualization and glyph design. Cleveland and McGill [CM84], and Heer and Bostock [HB10] explore the rankings of visual channels according to different types of data.

3. Domain Problem Characterization

In this section and the following three sections, we detail a user-centered design study by Munzner's [Mun09] nested model. The first level of this model is *domain problem characterization*.

Collaboration. This is a collaborative project between the University of Oxford and the University of Utah. The project team consists of computer scientists (specializing in visualization and related topics) and humanities scholars (specializing in poetry and linguistics). Our human-centered design process continues to involve literary scholars in various stages of the software development and implementation.

Initial Engagement. We arranged for visualization researchers to observe how literary scholars examine a small corpus by formulating and verifying hypotheses without the aid of visualization. In comparison with many other fields in the arts, humanities and social sciences, the discipline of poetry may typically involve smaller corpora and shorter text documents. During a Dagstuhl seminar, we established a list of typical scholarly questions that a literary scholar may ask during close reading. For example:

- What are the usages and temporal developments of different poetic elements, such as prosody, forms, syntax, etc.?
- Can visualization be used in analyzing at any given moment whether a transition or moment of innovation may be taking place in poetry as a whole and therefore how energized a current poetic moment may be?
- How do sound and other poetic features work in individual poems? Since all poems live in "time", even on the page, many new and interesting observations might be made with visualization for different poetic elements.

These questions provided our first attempt to characterize the domain problem and user tasks. After a period of studying existing scholarship on text visualization, rhyme and meter, the full team met. In addition to project meetings, a special workshop was organized where many humanities scholars brought up a variety of interesting problems that might be subject to visualization solutions. The visualization researchers also attended a poetry reading and engaged in in-depth discussions with humanities scholars on sound and timing in poems, existing text visualization techniques, the relative merits of animation and static visualization, the use of colors in visualization, and so on. This first meeting, then, involved an intense period of mutual education.

Characterization. These few days of intensive collaborative activities helped the visualization researchers appreciate that poetic information is rich and highly compressed: that a word or even a syllable in a poem does not operate in a singular way. While poets are interested in statistical aggregation, their work is actually centered on the relationships between words, through which meaning, message, idea, impression, affect and impact arise and develop. Not content to settle on a single interpretation of a poem, they constantly suggest alternate *interpretations* of a specific component or an attribute, discussing and debating the relative "strength" or value of a given interpretation. Perhaps the most important observation made is that these interpretations seem much like *hypotheses* in scientific studies, except that there are usually many such hypotheses about just one poem. What became exciting was the idea that visualization could help literary scholars to make observations more effectively, to stimulate different interpretations, and to visually evaluate interpretations.

4. Data and Operation Abstraction

Data and Operations. In order to help visualization researchers gain a better understanding of the data, literary scholars in the project proposed a pilot study on the short poem "Night" by *Louise Bogan* [Bog95] with a specific emphasis on how sound develops in the poem through time. The initial, simple hypothesis to be evaluated was that the timing of reading this poem aloud might be sensitive to different accents. Two literary scholars from two different regions recorded their readings, and a recording of a computer reading was also added into the data set. These recordings were visualized with the words of the poem. Perhaps because there were not enough samples, we did not manage to discover sufficient evidence to support the hypothesis. The one thing we did discover is that the human interpretations were more similar to each other than they were to the computer interpretation, which appeared to weigh each syllable uniformly. In other words, the humans seemed to be *interpreting* the poem as they read it. Although this small exercise was inconclusive, the literary scholars were made aware of the importance and capability of multivariate visualization. This led to a focused team effort in which we worked to identify various poetic attributes that might be subject to computation and that might be visualized to stimulate hypotheses and evaluate them visually.

We began with a questionnaire about sound, designed to help us compile a list of observation tasks (i.e., *operations*) and the associated sonic attributes (i.e., *variables*) that one would wish to visualize for these tasks. We were able to capture 52 tasks, including, for example, the following:

- *Which of the words in the poem rhyme with each other?*
- *How many of the vowels are unrounded vowels?*
- *How many groupings of repeated words are in the poem?*
- *Where are the caesuras in the poem?*
- *How many of the caesuras are masculine caesuras?*

Abstraction. The team identified 33 measurable variables associated with 52 tasks. We then followed the first questionnaire with a second, which asked the poets to rate the importance of each variable from their point of view. The scholars eliminated 7 variables, because some are global statistics that do not vary within a single poem (e.g., line count), some are difficult to determine literally (e.g., semantic connection between two words), and some are less relevant to poems in English (e.g., pulmonic or non-pulmonic consonants). This leaves 26 variables, which are categorized primarily into two groups, namely the *semantic group* and the *phonetic group*. A table that details the 26 variables is given as part of the supplementary materials.

The variables in the semantic group are associated with the written words in Latin alphabet, while those in the phonetic group are associated with the sound and pronunciation of the words, for which we use the International Phonetic Alphabet (IPA) [Int99], a standardized representation of sounds for oral language. By using these two alphabets, we can compute most of the other variables – such as phonetic repetitions (e.g., end rhyme, internal rhyme, assonance, consonance, alliteration), word repetitions, and their frequency and connectivity – directly. In addition, we utilize external resources, such as sentiment ontology [Nie11], to attach semantic information to words, and UCREL CLAWS5 [UCR] to classify words grammatically. We employ a look up table to obtain articulatory features of vowels and constants [Pho].

We did explore the option of reducing the total number of poetic variables to a level that is more manageable in developing a visualization system. However, we found that removing some variables would significantly reduce the variety of observations that one could make in close reading. Our literary scholars were very reluctant to lose any variables. We were also aware that this list was not definitive or exhaustive, and would likely expand in the future. Thus, we decided to take on the challenge of handling 26 poetic variables in our goal to design a visualization system that literary scholars would find easy, inviting, and useful to engage as part of their close reading practice.

5. Encoding and Interaction Design

For the purposes of visualization, a *poem* might be described as a sequence of *words*, usually divided into an arrangement of *lines* where each line may end with a rhyming sound depending on its form. In a poem, lines may also be grouped into *stanzas*. In the case of poems that are not divided into lines ("prose poems"), the decision not to use lines is an important one and must be taken into account in the design process. The ordering of words, lines (or lack thereof), and stanzas or "verse paragraphs" is thus an important attribute in the visualization, as it encodes temporal relationships between phonetic and semantic components. This attribute is thus encoded spatially as the basic contextual frame of a poem. Beyond the previously identified sonic requirements, therefore, we felt that it might be premature to introduce into our model those multivariate visualization techniques that do not explicitly encode spatial ordering of words, such as with parallel coordinate plots, multi-dimensional scaling and so on. We therefore focused on visual designs that depict poetic variables in conjunction with a display of the poem, either explicitly in textual form or implicitly in spatial location.

Let $V = \{v_1, v_2, \ldots, v_n\}$ be a set of all *poetic variables*, and $C = \{c_1, c_2, \ldots, c_m\}$ be a set of all *visual channels* available for visual mapping. One approach is to allow $v_i, i = 1, 2, \ldots, n$ to map to any $c_j, j = 1, 2, \ldots, m$. Although encoding 26 poetic variables using 26 different visual channels in visualization is possible, in this scenario our target users would have to determine 26 different mappings through a naïvely designed control panel. Still, a control panel remains necessary as it is unlikely that literary scholars would want to visualize all poetic variables simultaneously. Moreover, the inclusion of all variables would induce clutter and interference between visual channels, impairing the user's experience of the visualization and ability to engage in visual exploration.

Alternatively, we could have a fixed mapping from V to C. This would facilitate easy implementation. However, this would mean that some poetic variables would always map to more powerful visual channels [MRSS*12], while the visual analysis of other "less important" variables would be more difficult. This becomes problematic when some research tasks demand intensive observation of these "lesser" variables. Additionally, this approach would restrict the introduction of new poetic variables.

Inspired by the concept of the "artificial assistant" presented in [MHS07], we formulated four rules for reducing options for visual mappings in an automatic and context-sensitive manner. The goal is to provide a balanced solution to the visual and interaction designs. By intentionally avoiding fixed visual mapping, we can provide literary scholars with the freedom both to 1) choose different research foci, channeling their efforts towards specific regions in the hypothesis space, and 2) introduce new variables. Through a set of rules offering guidance on visual mapping, we can transfer some existing knowledge about best practices in visualization to literary scholars, while making interaction more scalable and less tedious.

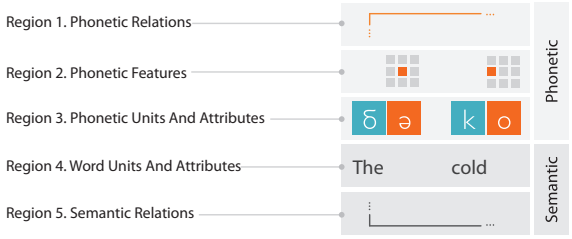


Figure 1: *The division of the visual representation of a poem into 2 main domains (phonetic and semantic) and 5 regions.*

Rule 1: Spatial Separation. Psychology research demonstrates that spatial memory and reasoning is an important factor that affect visual search (e.g., [Kos80, LS87, Kri00]). Meanwhile, spatial separation provides an effective means for alleviating the interference among visual channels and avoiding integrated visual channels [She64, MRSS*12]. Given numerous poetic variables, it is natural to group them based on some application-specific conceptual proximity, such as semantic meaning, task relevance, or a combination of several factors. One can then map such groups spatially to different regions, facilitating effective visual search and memorization. In our study, domain experts provided a grouping of the variables. This grouping resulted in our division of the visual representation of a poem into two main domains and five regions. As illustrated in Figure 1, the upper domain is designated for phonetic variables and consists of three regions, namely, 1) *phonetic relations*, 2) *phonetic features* and 3) *phonetic units and attributes*. The lower domain is designated for semantic variables, and it consists of two regions: 4) *word units and attributes* and 5) *semantic relations*.

The spatial separation based on conceptual grouping facilitates easy memorization of and visual search for a specific variable. It is not necessary for these five regions to be organized precisely as shown in Figure 1. In Section 6, we will discuss different layouts of the poem visualization. In this section, we focus the discussions on encoding and interaction design using the basic layout.

Regions 3 and 4 form the central axis of the basic spatial layout. To distinguish these two regions visually, region 3 has a boxed display for each phonetic unit, whereas region 4 has an appearance of free text. Each region can feature its own set of visual channels. For example, region 3 may offer the use of *symbol*, *symbol color* and/or *intensity*, *symbol size*, *filled box color*, *intensity* and/or *texture*, *box outline color, style* and *thickness*. Region 4 may offer *symbol*, *symbol color* and/or *intensity*, *symbol size*, *background color* and/or *intensity*, *underline color*, *style, texture* and/or *thickness*.

Regions 1 and 5 have similar data types and thus share

Figure 2: *(a) A representation of the position of the tongue for a close front rounded vowel. (b) The glyph design in our system for a close rounded vowel. (c) A representation of the tongue moving from a close front to a close back. (d) The glyph shows the transition of the vowel positions where a light shade represents the previous position and a darker shade indicates the current position.*

similar visual appearance. Nevertheless, the spatial separation helps minimize the confusion between information in these two regions. We use connection lines as the main visual objects in both regions. The visual channels available for these lines may include *color*, *intensity*, *opacity*, *thickness*, *height*, *curvature*, *style* and/or *terminal shape*. Region 2 is a glyph-based region for multivariate representations of phonetic variables. Currently, we provide glyphs designed for showing low-level features, such as characteristics and positions of each phonetic articulation in relation to the human vocal system.

Given a poetic variable v, we can use a look-up table to find its group categorization of $\Lambda_{group}(v)$. Similarly we can obtain the region categorization of c as $\Theta_{region}(c)$. This allows us to define the first rule as:

$$S_{R1}(v,c) = \begin{cases} 1 & \text{if } \Lambda_{group}(v) \text{ is permitted in } \Theta_{region}(c) \\ 0 & \text{otherwise} \end{cases}$$

The glyphs in region 2 are designed to supplement the phonetic symbols in region 3 as they help associate specific vowels or consonants to the "positions" and "geometries" of their sounds. We considered several different designs, starting from a mouth-shaped representation and finally settling on an abstract representation of $n_{row} \times n_{col}$ positions. Due to the size of the glyphs, we found that it was more difficult to recognize the positions in the designs that feature more realistic geometry.

The 3×3 positions of vowels encode (from left to right) *front*, *central* and *back* (which part of the tongue is raised); and (from top to bottom) *close*, *mid* and *open* (how far the tongue is raised). We use circles to represent a rounded vowel and squares to represent an unrounded vowel. To make the design more distinguishable, we also map the rounded and unrounded vowel positions to different colors orange and purple, respectively. Figure 2 shows an example glyph of a rounded vowel.

The 3×2 positions of consonants encode (from left to right) the two major classes of *places of articulation*: *front articulation* (labial, coronal), and *back articulation* (dorsal, radical, laryngeal); and (from top to bottom) the three major classes of *manners of articulation*: *complete obstruent* (stop), *partial obstruent* (fricative, affricate), and *sonorant*. We use hollow and filled square in blue to represent voiceless and voiced sound respectively.

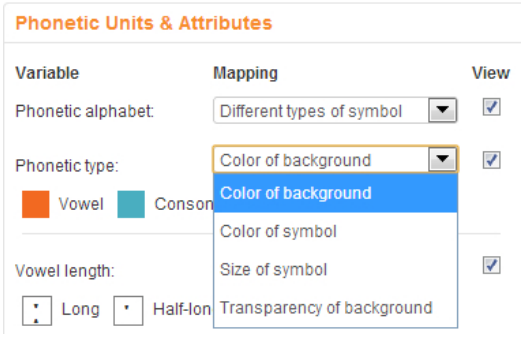


Figure 3: *A snapshot of the final interface showing some of the variables with their visual mappings and legend.*

The colors used in the glyph region are consistent with the default color scheme in the phonetic unit region, where orange and blue represent the vowels and consonants. In addition, we use two different color intensities to encode the temporal transition of sound (Figure 2).

Rule 2: Type Compatibility. The poetic variables have four literal types: *units* (i.e., alphabets), *attributes* (i.e., nominal or ordinal), *relations* (i.e., sets or pointers), and *attributes of relations* (i.e., nominal or ordinal). Bertin [Ber83] proposes four criteria for determining the *associative*, *selective*, *ordered* and *quantitative* capabilities of a number of visual channels. However, Bertin does not provide a *relational* criterion that defines whether or not a visual channel is suitable for depicting the relation between 2 or more entities.

Based on these five criteria, we can define two vector functions $\Lambda_{type}(v)$ and $\Theta_{type}(c)$. $\Lambda_{type}(v)$ returns five *requirement scores*, $[\lambda_a, \lambda_s, \lambda_o, \lambda_q, \lambda_r]$, indicating the requirements for v to be associative, selective, ordered, quantitative, and relational respectively. Similarly, $\Theta_{type}(c)$ returns five *capability scores*, $[\theta_a, \theta_s, \theta_o, \theta_q, \theta_r]$. All these scores are within the range of [0, 1]. With these two vector functions, we can define the rule for assessing type compatibility as:

$$S_{R2}(v,c) = \frac{\Lambda(v) \bullet \Theta(c)}{5(\lambda_a + \lambda_s + \lambda_o + \lambda_q + \lambda_r)}$$

For example, the Latin alphabet type is a nominal variable, and we score its requirements as $[1,1,0,0,0]$. End rhyme relation is a relational variable, and we score its requirements as $[1,1,0,0,1]$. We score each visual channel according to its capacity, for instance assigning $[0.7,1,0.7,0,0]$ to the filled box color in region 3, and $[0.3,1,0,0,1]$ to the connection line channel in region 1.

Rule 3: Channel Capacity. The *capacity* of a channel c is the maximum number of distinguishable values it can represent. Such a maximum capability value depends not only on the representable range within the computer, such as the

Figure 4: *A sonnet showing its phonetic and rhyming connections, such as its internal rhymes, end rhymes and frequency of the IPA characters.*

maximum length of line in pixels, or 256^3 for colors. More importantly, it depends on many perceptual factors, such as *just noticeable difference* [BF73] and interference from nearby or integrated visual channels [She64].

For example, although the box outline in region 3 can be associated with visual channels such as color, thickness, and style, the capacity of the thickness channel is very limited as it has ≤ 3 pixels of room for variation. Although the display may be capable of showing 256^3 colors, the capacity of the color channel is in fact much lower, typically at about 5-12 colors [Hea96, War12], due to unreliable color perception and interference from the filled box colors. In general, it is desirable to use a visual channel of a higher capacity, though this is often in conflict with other requirements.

We assign each poetic variable with a minimal capacity requirement, v_{min} and a maximum requirement, v_{max}. Meanwhile, each visual channel c is given a capacity range, c_{lower} and c_{upper}, where c_{lower} indicates the ideal capacity limit, and c_{upper} the strict maximum capacity. We define rule 3 for ranking the suitability of channel capacity as:

$$S_{R3}(v,c) = \begin{cases} 1 & \text{if } v_{max} \leq c_{lower} \\ 0 & \text{if } v_{min} \geq c_{upper} \\ 0.5 & \text{otherwise} \end{cases}$$

In general, the type compatibility rule (S_{R2}) attempts to determine how well a specific visual channel is suited for a given poetic variable. The channel capacity rule (S_{R3}) tries to ensure that there is enough "bandwidth" in the visual channel to display a desirable number of values for a poetic variable in a visually distinctive manner.

Rule 4: Already in Use. If a particular visual channel in a region is already assigned to a poetic variable, it is of course not appropriate to overload the same channel. Hence the 4th rule is defined as:

$$S_{R4}(v,c) = \begin{cases} 0 & \text{if c is in use} \\ 1 & \text{otherwise} \end{cases}$$

Since each of $S_{R1}, S_{R2}, S_{R3}, S_{R4}$ is already normalized in the range of $[0,1]$, we can treat all of the rules as *desirability functions* [DS80]. The overall desirability S for each pair of v and c is thereby:

$$S(v,c) = \left(\prod_{i=1}^{4} S_{Ri}(v,c)\right)^{1/4}$$

We can define a threshold $0 \leq \tau < 1$ such that when $S(v,c) > \tau$, the visual channel c is available to the poetic variable v as an optional visual mapping. Figure 3 shows a pull-down menu with a significantly reduced number of visual mapping options. When users choose an option, a legend automatically appears, allowing users to inspect the suitability of particular visual mappings (e.g., type compatibility and channel capacity). The legends of the default mappings are given as part of the supplementary materials.

Evaluation. Evaluation processes were (and continue to be carried) out at different stages of the software design and implementation. Our team has conducted two main forms of evaluation so far: explicit evaluation, conducted through demonstration and trials of prototypes; and implicit evaluation, conducted through discussions about observable features and patterns depicted in visualizations. On several occasions, the evaluations led us back to the higher levels in the nested model. For example, after the evaluation of the rule-based visual mapping, the literary scholars organized a recorded close reading session so that visualization researchers could observe and appreciate the "levels of detail and complexity" that a poem visualization should ideally reach. Below, we give three examples of evaluations we conducted.

Figure 4 shows three sections of a sonnet. Each connection line represents a rhyme relation linking specific words to other rhyming words to reveal rhyming patterns across a poem. The height of the line is used to distinguish different rhymes, with internal rhymes shown in gray and end rhymes in purple. Thicker lines indicate those rhymes that occur more frequently. Such connection lines, have also been used to show musical refrains [Wat02] and allow for an easy overview of the distribution of rhyming patterns throughout the poem. During the demonstration and evaluation of this prototype, we discovered that the basic latitudinal layout does not support overview very well, though it works well in providing a detailed visualization. This led to brainstorming sessions on different layout algorithms.

Figure 5 shows a visualization for comparing the vowel positions of a free-verse poem, a sonnet and a nursery rhyme with those in a novel, a political manifesto and a paper in *Nature*. The visualization researchers were initially puzzled by

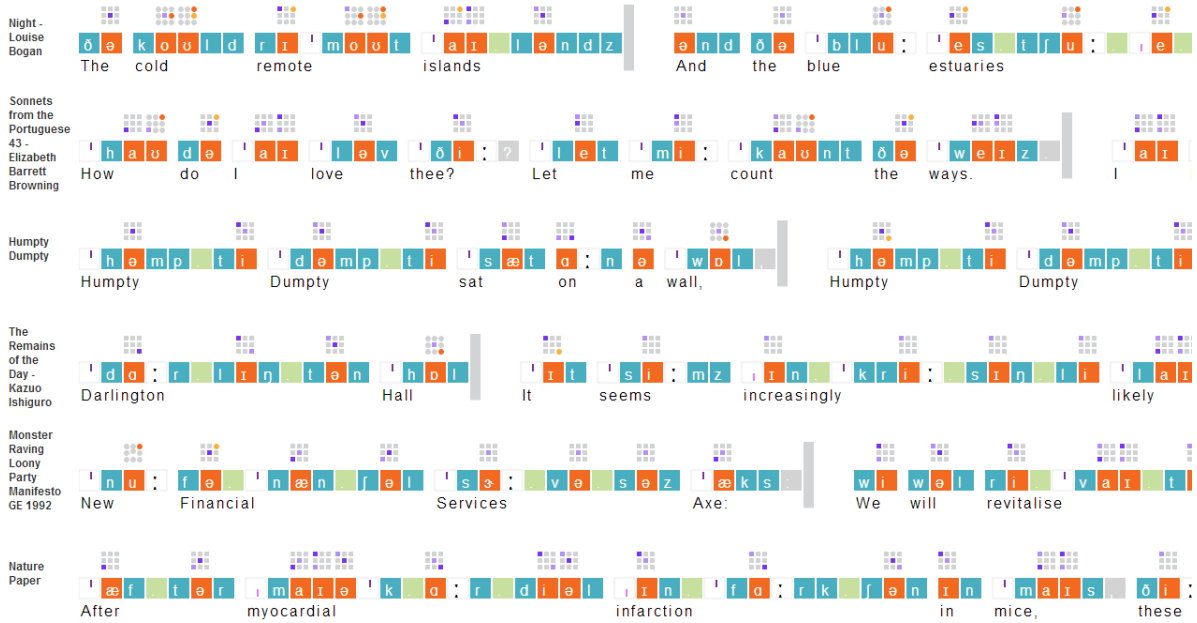


Figure 5: *Visualization of poems and texts with a focus on vowel positions. The top three are poems: a free-verse poem, a sonnet and a nursery rhyme. The last three are prose texts: a novel [BNC07], a political manifesto [BNC07] and a Nature paper. The gray vertical bars indicate line breaks in poems or sentence breaks in other texts.*

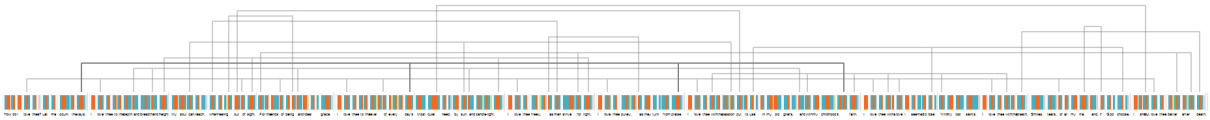
the visible difference in the glyph-based representations of phonetic features for poems and other texts. A Skype meeting was held to discuss this visualization result as an implicit evaluation. This visualization stimulated a few new hypotheses. For example, the two poems at the top clearly exhibit more dynamic changes in vowel positions than either of the prose pieces. This provides evidence to support the understanding of literary scholars that poets strive to enhance the density of their poems by creating dynamic sound through sonic recurrence and change in every word throughout the poem; it also leads to a hypothesis that language in prose texts or everyday use may have evolved for practical reasons to aid audience comprehension speed and accuracy by avoiding such sonic turbulence.

The above two examples of evaluation were carried out during the initial software prototyping stages. After the evaluation confirmed the merits of the rule-based visual mapping and interaction design, the desktop-based prototype was transformed to a web-based system called "Poem Viewer". The increased accessibility provided by this format enables further and more wide-spread ongoing evaluation. Thus far, the two poetry scholars have visualized 30 poems and poetic fragments. They used the tool mostly alone, but also once in the presence of a visualization scientist, who recorded their responses and – as is typical in ethnographic studies on visualization – asked what insight they had obtained. The poetry scholars responded that they would not likely look for insight from the tool itself. Rather, they would look for enhanced poetic engagement, facilitated by visualization. It is this enhanced engagement – not with the tool but with the poem – which might subsequently prompt new insights. At the same time, the poetry scholars observed that the most promising use of Poem Viewer was with the visualization of the sound placement in the mouth. They explained the merits of visualization: "*While it takes at least as long to get information about mouth positions on a given word from the visualization as from saying the words and thinking about where they happen in the mouth, it is much faster to compare two different visualizations by eyeballing them for turbulence than to make such a comparison manually (or orally, if you will).*"

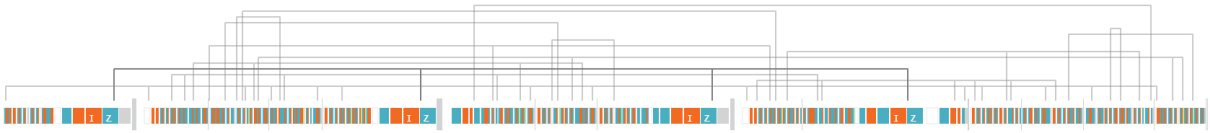
As the second example (Figure 5) shows, if sound placement is used as the measure, poems are more musically various or disrupted than prose. Using the vowel positions, poetry scholars were able to observe sound developing across the texts and linking lines and stanzas through patterns of recurrence and change, which are of intense interest to poetry scholars. For example, they can see the repetition of the long "o" sound linking "cold" and "remote" evolving into the repeated long "u" sound of "blue" and "estuaries" in the free-verse poem. Such a transformation cannot be seen easily either in simple alphabet spellings or IPA character representations of syllables. Something that is normally heard and has to be related using a mental image can now be visualized with the help of glyphs. The scholar was also able to use the visualization to watch the evolution of the vowel from "I" to "love" to "thee" in the sonnet.

Still another interesting example of this visualization's

(a) Basic visualization of a sonnet (only 15% of the poem is being displayed).



(b) A sonnet that has been downsampled with uniform scaling.



(c) Focus+context of a sonnet using a non-uniform scaling.

Figure 6: *Three different latitudinal layouts of a poem.*

usefulness occurs in the nursery rhyme, "Humpty Dumpty", in particular in looking at the phrase "sat on a wall". Using the IPA character, we can see that "on" and "wall" are repeating the same vowel sound even though they are not using the same alphabetic letters. Without the vowel position glyphs, poetry scholars may not consciously think about the fact that "sat on the wall" resides in the lower register, moving only briefly to the neutral mid-point. This is interesting since the positioning links those words even though they do not all rhyme. The link is further strengthened by the fact that all the words are single-syllable words. The combination of the lower register and the single syllable makes the phrase more emphatic than, for example, the phrase "Humpty Dumpty", which is both multi-syllabic and resides in the mid-to-upper vowel register. This is an observation that a poetry scholar may come to much more slowly – if at all – without the tool.

6. Algorithm Design

The major algorithmic development in this project is the layout of a poem visualization. On one hand, the basic latitudinal layout enables the phonetic and semantic connections across a poem. On the other hand, as identified in a formal evaluation session mentioned in the previous section, it does not support overview very well. Figure 6(a) shows a section of such a basic visualization of a poem. For each character/box that is to be displayed, we assign a set of display geometries such as x, y, *width* and *height*. For the basic visualization, the rendering moves horizontally by the width, $x_{new} = x_{previous} + width$.

Figure 6(b) shows a basic visualization that has been downsampled in the x direction with uniform scaling. The poem is deformed horizontally, where $x_{new} = x_{previous} + width * (\eta)$. η is a scaling factor < 1. An advantage of using this approach is that the user is provided with an overall view of the poem with its connecting arcs. A disadvantage is that the user is not able to see any phonetic or Latin characters.

Poem Viewer demonstrates an alternative approach to the uniform downsampling method. As shown in Figure 6(c) when a particular rhyme is selected, the latitudinal visualization ensures that the corresponding regions along the x-axis are still in focus, while the rest of the visualization is downsampled to provide the context. Algorithmically, we achieve this by first establishing a set of focal points, $fx_1, fx_2, \ldots, fx_k$ according to a selected connection line or a rhyming pattern. We then create a curve with a Gaussian function at each focal point. We finally render the overview for using this curve as a transfer function for determining the rate of downsampling. Using this approach, the user not only has an overview of the rhyming connections across the poem but is also able to identify the connecting rhyming characters.

The conventional layout of a poem is two-dimensional. We thus investigated the feasibility of visualizing poems in their "organic" layout. The main difficulty is that connection lines running among other visual channels, such as symbols and filled boxes, causes extensive visual cluttering. Poem Viewer thus implements a layout that allows connecting lines to run on either the left- or right- hand sides of the poem, as shown in Figure 7. The semantic connections, such as repeated words, are displayed on the left hand side of the poem, where the height of the arc indicates the different groupings of repeated words, while the right hand side is used for displaying the phonetic connections in the poem. For internal rhyme relations and word-based semantic relations, the arc height alone is not sufficient to separate the different groups. We thus introduce an offset mechanism to relax the connection terminators in y direction. For each line in a poem, we store its grouping of repeated words (or internal rhymes), moving from the most to least frequent, (top

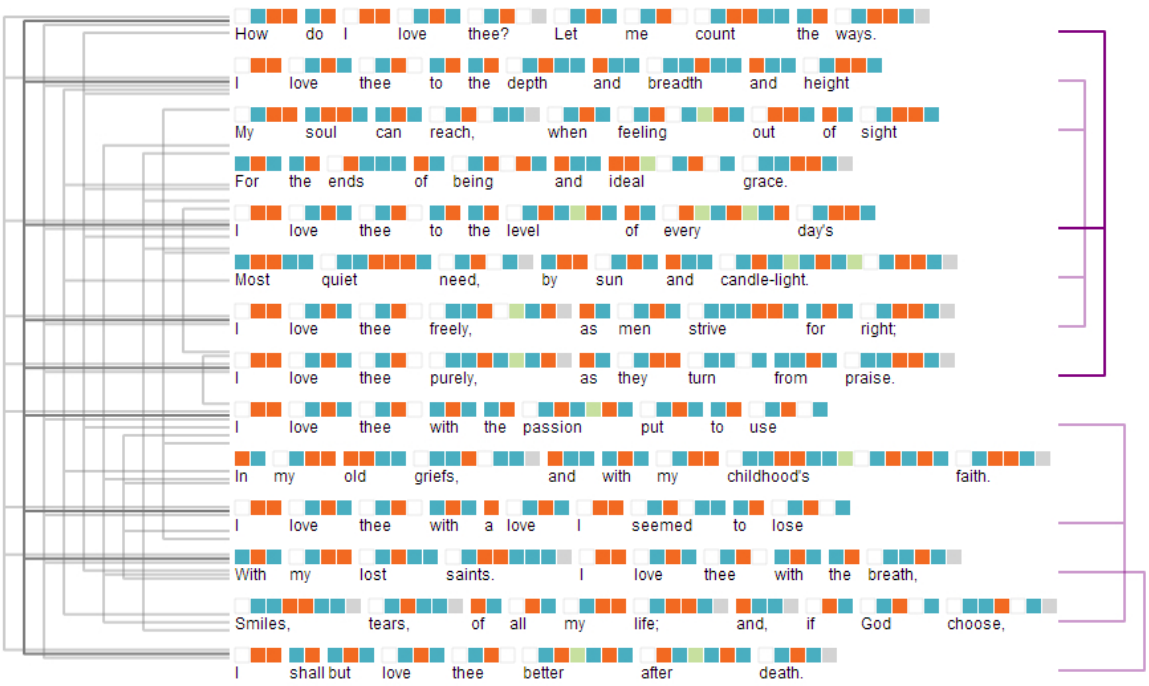


Figure 7: *A structurally laid out poem with its phonetic and semantic connections.*

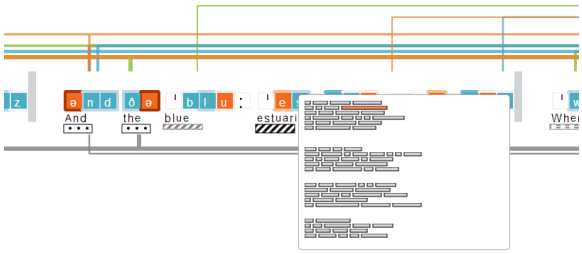


Figure 8: *A user can interact with the visualization to view the location of a word in the structured layout. The texture under each word represents a grammatical "part of speech".*

to bottom). As we proceed through the list, we offset its y position by a constant δ. The main disadvantage is that one cannot associate individual connection lines with any particular word or internal rhyme within a line of text.

We have observed that poetry scholars tend to know the poem under study extremely well. Hence, the basic latitudinal visualization shown in Figure 6(a) remains useful in close reading. We can augment the basic visualization with a context view interactively. As shown in Figure 8, the user can click on a word to see its position in the context of the whole poem. Other forms of interaction include selecting and highlighting a specific word or relation throughout the poem. All four layout algorithms can perform in real-time on an Intel laptop (2.8GHz, 8GB of RAM). This makes the algorithms suitable for interactive visualization.

7. Conclusions

In this paper, we reported on our experience of a fascinating collaboration between computer scientists and literary scholars and on our design study on poetry visualization. We found that the nested model by Munzner [Mun09] offers an effective framework for organizing different types of user engagements. Through this intensive collaboration, we were able to formulate a rule-based solution to address the need for *high-dimensional multivariate visualization of poems* — a notion that was difficult for both computer scientists and literary scholars to grasp at the beginning of our work. In addition, we developed a user interface to support the tasks of visual mapping along with several layout algorithms. We are in the process of transforming the prototype system into a browser-based tool for easy installation and use. We will also conduct a number of focused studies on some literary hypotheses generated during this project.

Acknowledgments

This work was funded jointly by NEH (USA) and JISC (UK) under the Digging Into Data Challenge program.

References

[AC07] ABBASI A., CHEN H.: Categorization and analysis of text in computer mediated communication archives using visualization. In *Proc. 7th ACM/IEEE-CS Joint Conf. Digital Libraries* (2007), pp. 11–18. 2

[AS04] AMAR R., STASKO J.: A knowledge task-based framework for design and evaluation of information visualizations. In *Proc. IEEE Symp. Information Visualization* (2004), pp. 143–150. 2

[Ber83] BERTIN J.: *Semiology of Graphics: Diagrams, Networks, Maps*. University of Wisconsin Press, 1983. 2, 5

[BF73] BOOTH D., FREEMAN R. J.: Discriminative measurement of feature integration. *Acta Psychologica* (1973). 6

[BNC07] BNC: The British National Corpus, version 3 (BNC XML Edition). Distributed by Oxford University Computing Services on behalf of the BNC Consortium, 2007. URL: http://www.natcorp.ox.ac.uk/. 7

[Bog95] BOGAN L.: Night. The Blue Estuaries: Poems 1923-1968, 1995. 3

[CAT*12] CLEMENT T., AUVIL L., TCHENG D., CAPITANU B., MONROE M., GOEL A.: Sounding for Meaning: Analyzing Aural Patterns Across Large Digital Collections. In *Digital Humanities* (2012). 2

[CCP09] COLLINS C., CARPENDALE M. S. T., PENN G.: Docuburst: Visualizing document content using language structure. *Computer Graphic Forum 28*, 3 (2009), 1039–1046. 2

[CGM*12] CHATURVEDI M., GANNOD G., MANDELL L., ARMSTRONG H., HODGSON E.: Myopia: A Visualization Tool in Support of Close Reading. In *Digital Humanities* (2012). 2

[CM84] CLEVELAND W. S., MCGILL R.: Graphical perception: Theory, experimentation, and application to the development of graphical methods. *Journal of the American Statistical Association 79*, 387 (1984), 531–554. 2

[DS80] DERRINGER G., SUICH R.: Simultaneous optimization of several response variables. *Journal of Quality Technology 12*, 4 (1980), 214 – 219. 6

[DZG*07] DON A., ZHELEVA E., GREGORY M., TARKAN S., AUVIL L., CLEMENT T., SHNEIDERMAN B., PLAISANT C.: Discovering interesting usage patterns in text collections: Integrating text mining with visualization. In *Proc. 16th ACM Conf. Info. and Knowledge Management* (2007), pp. 213–222. 2

[HB10] HEER J., BOSTOCK M.: Crowdsourcing graphical perception: Using mechanical turk to assess visualization design. In *Proc. of the SIGCHI* (New York, 2010), pp. 203–212. 2

[Hea96] HEALEY C. G.: Choosing effective colours for data visualization. In *Proc. 7th Conf. Visualization* (1996). 6

[HF06] HENRY N., FEKETE J.-D.: MatrixExplorer: A dual-presentation system to explore social networks. *IEEE Trans. Visualization and Comp. Graphics 12*, 5 (2006), 677–684. 2

[Int99] INTERNATIONAL PHONETIC ASSOCIATION: *Handbook of the International Phonetic Association*. Cambridge University Press, 1999. 3

[KO07] KEIM D. A., OELKE D.: Literature fingerprinting: A new method for visual literary analysis. In *IEEE VAST* (2007), pp. 115–122. 2

[Kos80] KOSSLYN S. M.: *Image and Mind*. Cambridge: Harvard University Press, 1980. 4

[Kri00] KRISTJÁNSSON A.: In search of remembrance: Evidence for memory in visual search. *Psychological Science 11*, 4 (2000), 328 – 332. 4

[LPP*06] LEE B., PLAISANT C., PARR C. S., FEKETE J.-D., HENRY N.: Task taxonomy for graph visualization. In *Proc. AVI BELIV Workshop* (2006), pp. 1–5. 2

[LS87] LARKIN J. H., SIMON H. A.: Why a Diagram is (Sometimes) Worth Ten Thousand Words. In *Cognitive Science* (1987), vol. 11, pp. 65 – 99. 4

[MB05] MCGUFFIN M., BALAKRISHNAN R.: Interactive visualization of genealogical graphs. In *IEEE Symp. Information Visualization* (2005), pp. 16 – 23. 2

[MHS07] MACKINLAY J., HANRAHAN P., STOLTE C.: Show Me: Automatic presentation for visual analysis. *IEEE Trans. Visualization & Comp. Graphics 13*, 6 (Nov. 2007), 1137–1144. 1, 2, 4

[Mil95] MILLER G. A.: WordNet: A lexical database for English. *Communications of the ACM 38*, 11 (Nov. 1995), 39–41. 2

[Mor05] MORETTI F.: *Graphs, Maps, Trees: Abstract Models For A Literary History*. Verso, 2005. 2

[MRSS*12] MAGUIRE E., ROCCA-SERRA P., SANSONE S.-A., DAVIES J., CHEN M.: Taxonomy-Based Glyph Design - with a Case Study on Visualizing Workflows of Biological Experiments. *IEEE Trans. Visualization & Comp. Graphics 18*, 12 (2012), 2603 – 2612. 4

[Mun09] MUNZNER T.: A nested model for visualization design and validation. *IEEE Trans. Visualization & Comp. Graphics 15*, 6 (Nov. 2009), 921–928. 1, 2, 9

[Nie11] NIELSEN F. Å.: A new ANEW: Evaluation of a word list for sentiment analysis in microblogs. In *Proc. ESWC Workshop on 'Making Sense of Microposts'* (May 2011), pp. 93–98. 3

[OBK*08] OELKE D., BAK P., KEIM D., LAST M., DANON G.: Visual evaluation of text features for document summarization and analysis. In *IEEE VAST* (Oct. 2008), pp. 75 –82. 2

[Pal] PALEY W. B.: TextArc. URL: http://www.textarc.org/. 2

[Pho] PHOTRANSEDIT: Text to phonetics. URL: http://www.photransedit.com/Online/Text2Phonetics.aspx. 3

[Pla06] PLAMONDON M. R.: Virtual verse analysis: Analysing patterns in poetry. *Literary and Linguistic Computing 21*, suppl 1 (2006), 127–141. 2

[She64] SHEPARD R.: Attention and the metric structure of the stimulus space. *Journal of Mathematical Psychology 1*, 1 (1964), 54 – 141. 4, 6

[UCR] UCREL: Claws part-of-speech tagger for English. URL: http://ucrel.lancs.ac.uk/claws/. 3

[VCPK09] VUILLEMOT R., CLEMENT T., PLAISANT C., KUMAR A.: What's being said near "Martha"? Exploring name entities in literary text collections. In *IEEE VAST* (2009), IEEE, pp. 107–114. 2

[vHWV09] VAN HAM F., WATTENBERG M., VIEGAS F. B.: Mapping text with phrase nets. *IEEE Trans. Visualization & Comp. Graphics 15*, 6 (Nov. 2009), 1169–1176. 2

[VPF06] VALIATI E. R. A., PIMENTA M. S., FREITAS C. M. D. S.: A taxonomy of tasks for guiding the evaluation of multidimensional visualizations. In *Proc. AVI BELIV Workshop* (2006), pp. 1–6. 2

[War12] WARE C.: *Information Visualization: Perception for Design*, 3rd ed. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2012. 2, 6

[Wat02] WATTENBERG M.: Arc diagrams: Visualizing structure in strings. In *Proc. IEEE Symp. Information Visualization* (2002), pp. 110–116. 6

[WV08] WATTENBERG M., VIÉGAS F. B.: The Word Tree, an interactive visual concordance. *IEEE Trans. Visualization & Comp. Graphics 14*, 6 (Nov. 2008), 1221–1228. 2

E Copy of [MRSS+13]:

Visual Compression of Workflow Visualizations with Automated Detection of Macro Motifs

Visual Compression of Workflow Visualizations with Automated Detection of Macro Motifs

Eamonn Maguire, *Student Member, IEEE*, Philippe Rocca-Serra, Susanna-Assunta Sansone, Jim Davies, and Min Chen, *Member, IEEE*

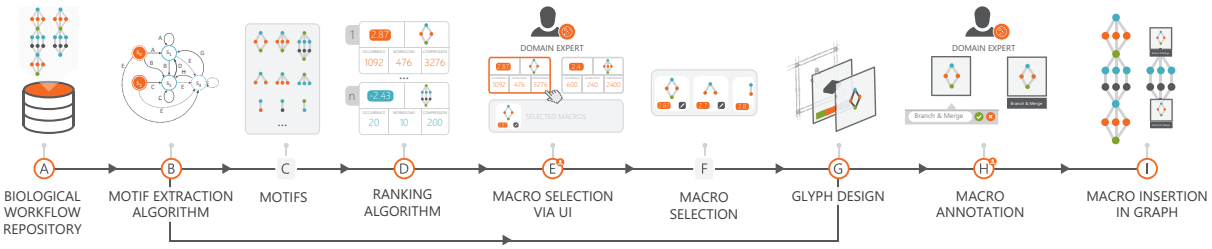


Fig. 1. An overview of the processes implemented in our system, AutoMacron, for compression of workflow visualizations. A) Biological workflows are imported from public repositories. B) The database of workflows is analyzed by our novel motif extraction algorithm that is sensitive to node/edge types and is built using a state-transition approach. C) The algorithm generates a large list of motifs. D) The motifs are ranked using metrics measuring their overall occurrence, scope of influence and compression potential. E) Based on the ordered recommendations by the system as well as their own knowledge, domain experts explore the space of motifs to identify suitable 'macros'. F) A subset of motifs are selected for macro generation. G) A glyph is assigned to each macro automatically using a design pattern that utilizes the state output from the motif extraction algorithm. H) Domain experts are able to annotate macros with additional text labels. I) Macros can be inserted into the relevant workflows as a method for graph compression.

Abstract—This paper is concerned with the creation of 'macros' in workflow visualization as a support tool to increase the efficiency of data curation tasks. We propose computation of candidate macros based on their usage in large collections of workflows in data repositories. We describe an efficient algorithm for extracting macro motifs from workflow graphs. We discovered that the state transition information, used to identify macro candidates, characterizes the structural pattern of the macro and can be harnessed as part of the visual design of the corresponding macro glyph. This facilitates partial automation and consistency in glyph design applicable to a large set of macro glyphs. We tested this approach against a repository of biological data holding some 9,670 workflows and found that the algorithmically generated candidate macros are in keeping with domain expert expectations.

Index Terms—Workflow visualization, motif detection, glyph-based visualization, glyph generation, state-transition-based algorithm

1 Introduction

The term 'macro' is derived from the Greek word *makro* meaning *big* or *far*. In computer science, the term is defined as "a single instruction that expands automatically into a set of instructions" [4]. It is commonly used as a noun (*e.g.*, a macro), or an adjective (a macro command). In schematic diagrams, such as electronic circuit diagrams, data flow diagrams, and control engineering block diagrams, *macros* are commonly used to provide hierarchical concept abstraction as well as visual compression. Not only can macros facilitate "overview first, details on demand" in visualization [35], but they can also speed up visual search and reduce cognitive load for experienced users, who are knowledgeable about or have become accustomed to the specification of individual macros. In effect, their functionality bears some resemblance to *acronyms*.

This work is motivated by the need to reduce the visual complexity of biological experiment workflows in an extension to work conducted by Maguire *et al.* [24]. Such workflows describe the sequence of processes (arranged with respect to a temporal dimension) enacted on biological materials and signals obtained through experimental observations. Large repositories of experimental data offer a data corpus that can be tapped into for workflow analysis and, more specifically, detection of commonly-used subgraphs (also referred to as *motifs*). Success in "compressing" such recurring motifs using macros would significantly reduce the time required for creating, and perhaps more importantly, viewing and comparing workflows.

When sketching out workflows on paper, individual scientists often abstract a commonly performed sequence of steps into a macro procedure or represent a set of parallel steps by using a macro step. Such a macro encapsulates the sense of a bigger block, a higher-level abstraction, and multiple steps, just like in programming and other schematic diagrams. Despite the potential benefits of using macros, one major stumbling block that hinders the availability and use of macros in workflow visualization is the lack of standards. Nevertheless, steps towards standardization can be taken. With the availability of large collections of workflows in biological experiment repositories, computational approaches may be applied to detect commonly-used motifs (*i.e.*, topological patterns in workflows). It provides domain experts

- *Eamonn Maguire is with the Oxford e-Research Centre and Department of Computer Science, University of Oxford, UK. E-mail: eamonn.maguire@st-annes.ox.ac.uk.*
- *Philippe Rocca-Serra is with the Oxford e-Research Centre, University of Oxford, UK. E-mail: philippe.rocca-serra@oerc.ox.ac.uk.*
- *Susanna-Assunta Sansone and Min Chen are with the Oxford e-Research Centre, University of Oxford, UK. E-mail: susanna-assunta.sansone@oerc.ox.ac.uk.*
- *Jim Davies is with Department of Computer Science, University of Oxford, UK. E-mail: jim.davies@cs.ox.ac.uk.*
- *Min Chen is with the Oxford e-Research Centre, University of Oxford, UK. E-mail: min.chen@oerc.ox.ac.uk.*

Manuscript received 31 March 2013; accepted 1 August 2013; posted online 13 October 2013; mailed on 4 October 2013.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org.

with an objective means for establishing a list of candidate macros based on their usage. The final selection of macros can be determined semantically in traditional ways (*e.g.*, community consensus, popularity ranking or recommendations by standards bodies).

The main contributions of this work are as follows:

- Our overall approach for using a computational methodology to identify candidate macros in relation to a workflow repository is new. This enables exhaustive search and objective selection of candidates while still allowing domain experts to make the final decision on macro creation.
- We propose an efficient algorithm for extracting motif patterns in workflows by taking into account node and edge types. This enables motif grouping through inclusion of semantic context. Our tests show that this type-sensitive algorithm performs faster than existing generic algorithms in the literature; and
- Our motif extraction algorithm is based on a finite-state machine. As workflows are directional and acyclic, the state transitions for identifying a motif encode the structure of the motif. We make use of such information to characterize the structural pattern of selected macros visually. This facilitates partial automation when designing an individual glyph for each macro.

2 Related Work

Schematic representations of workflows are commonplace in a range of disciplines. A workflow typically describes a sequence of steps, followed from initiation to completion when conducting a piece of work. Perhaps the most widely used workflow visualization is the *Gantt chart*, which depicts tasks, resources and their dependencies in a temporal manner. Efforts such as VisTrails [10], VTK [34] and SmartLink [37] make use of workflows to depict the processes followed to create a visualization. In Taverna [16] and Kepler [7], workflow visualization allows users to build reproducible pipelines for data analysis. Other workflow visualizations include the Business Process Model and Notation (BPMN), Petri-net, and programming flowchart, all of which convey work flow, data flow and process interactions within often very complex systems.

In this work, we consider workflows used to describe biological experiments. This class of workflow visualization renders the processes enacted on biological materials in experimental setups, from sample collection through experimental perturbation to signal acquisition and interpretation. While most scientists have been using generic text-based graph drawing tools such as GraphViz [1], new workflow visualization tools are emerging (*e.g.*, [24]).

Graph reduction is a family of algorithms and techniques for reducing visual complexity by using graph filtering and graph aggregation [39]. *Graph filtering* involves removal of certain nodes and edges from the graph, either deterministically or stochastically [22, 39]. *Graph aggregation* selectively merges two or more nodes into one, hence preserving some information about the nodes and edges to be removed. Many selection algorithms exist, such as methods for building hierarchical levels of detail, clustering based on node/edge attributes and edge bundling for clutter reduction [39, 15].

One subset of graph reduction techniques is motif based. Motifs, in the context of graphs, are "patterns of interconnections occurring in complex networks" [25, 27]. A considerable amount of effort has been dedicated to the automatic identification and characterization of meaningful motifs in individual graphs or sets of graphs (*e.g.*, [18, 23, 41, 33, 38, 21, 12]). Replacing recurring motifs with macros can provide hierarchical concept abstraction, visual compression, improved readability and cost-effective task performance. Macros feature extensively in various graph representations, such as schematic diagrams, communication network diagrams and workflow diagrams. For example, VisMashup [31] utilizes macros to simplify the large body of steps required to create a visualization. Dunne and Shneiderman propose to simplify graphs by using fan and arced glyphs to represent common topological structures [12]. Shneiderman and Aris propose the use of user-defined semantic substrates for compressing network visualization [36]. However, determining suitable macros

Table 1. Some commonly used motif finding algorithms.

Algorithm	Sampling	Node Limit	Focus
mfinder [19]	exact & estimated	6	discovery
FPF (mavisto) [33]	exact	4	search
FANMOD [41]	exact & estimated	8	discovery
NeMoFinder [11]	exact	12	search
MODA [26]	exact	not defined	search
G-Tries [28]	exact	> 9	discovery
Grochow-Kellis [13]	exact	9	search
Kavosh [17]	exact	8	discovery
Color-coding [5]	estimated	10	discovery

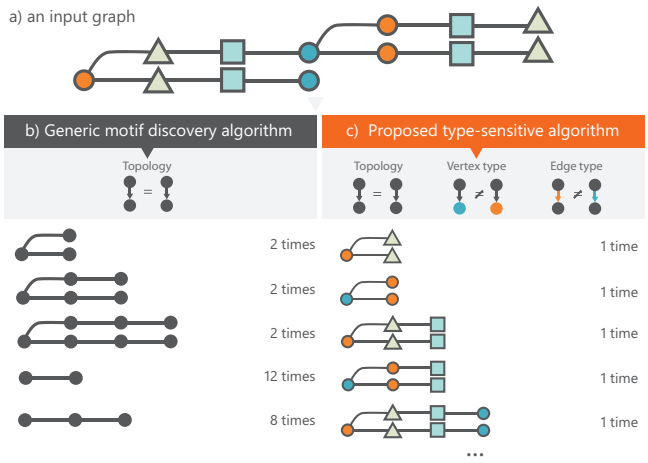


Fig. 2. (a) A graph, with typed nodes and edges, where different node types are mapped to different shapes and colours. (b) Generic motif discovery (or search) algorithms focus on topological differences. (c) Our motif extracting algorithm takes varying node/edge types into account, yielding semantically aware motif grouping.

usually depends on both the semantic content of the corresponding motif and its potential use in graph reduction. Hence, relying solely on topological information may not lead to meaningful visualization, while relying solely on user input does not scale up to a large collection of graphs.

As shown in Table 1, many motif search/discovery algorithms exist. Ribeiro *et al.* [28], Kashani *et al.* [17] and Wong *et al.* [42] published benchmarks detailing processing time for a subset of the algorithms in Table 1. As subgraph matching is fundamentally an *NP*-complete problem, these algorithms are computationally intensive. Wong *et al.* report *maVISTO* taking 14,000 seconds to find size three motifs in an *E. Coli* network. Comparing that to one second for FANMOD to analyze the same network, one can see the huge variability in algorithm performance. As the size of the target motifs grows, performance decreases. Using the same *E. Coli* network but searching for size 8 motifs, FANMOD will need 9000 seconds to finish the operation [42].

In Table 1 the second column indicates whether the sampling in a search space is exact (enumerating all possible candidates) or estimated. The third column indicates the maximum number of nodes in a motif the algorithm can handle. The final column indicates whether the algorithm is focused on *motif discovery* (finding repetitive patterns in a set of graphs), or *motif search* (finding a given motif in a set of graphs). All these algorithms focus on topological patterns in graphs only and do not consider the types of nodes and edges as a search constraint. As illustrated in Figure 2, these generic algorithms typically focus on topological patterns in a graph. However, the types, or semantic categories of nodes and edges are an interesting property in defining a macro. There is a more semantically aware motif search algorithm implemented in VisComplete [32, 20] that compares node labels and node order to predict the next step in a VisTrails pipeline analysis over a larger corpus of pipelines. However this algorithm is topologically less sensitive and provides no way to identify branch/merge events in a motif, or edge types. A potentially useful algorithm for identifying suitable candidate macros should be type as well as topology

sensitive. Our work focuses on such an algorithm, offering additional advantages in computational performance and providing visual mappings with meaningful structural information.

In our work, when considering whether a subset of steps in a given workflow is the same as another subset in the same or a different workflow, we must consider the semantics attached to each step (see Figure 2). Consequently, the task of motif discovery and search is highly constrained by the types of nodes and edges. Therefore, it is necessary, as well as advantageous, to develop specific motif extraction algorithms for workflow visualizations. In addition, we propose a novel concept of partially automating the design of macro glyphs. We consider the use of multi-resolution glyphs to depict macros at different levels of detail when a user interacts with the visualization, a technique referred to as semantic zooming [9, 40].

3 Motivation and System Overview

Over the past two decades, Biology has benefited from entering the digital era, becoming a data intensive field. Ancillary to this development, important efforts have been undertaken to preserve and curate digital artifacts in biology, including experimental workflows. A workflow is a form of directed graph (digraph). Scientists mainly rely on two types of workflow visualization: digraphs with text labeled boxes or glyph nodes. The latter enables more compact visual representation than the former [24] allowing domain experts to perform their tasks such as error checking and comparison more quickly. However, workflows can still be quite large and complex, containing many repeated subgraphs, which demand some effort for identification in 'flat' representations. Hence, it is highly desirable to introduce macro representations in workflow visualizations, as both text-based and glyph-based workflow visualizations can benefit from macro based visual compression. Although experimental workflows in biology exhibit specific properties that are different from workflows in other disciplines, they often share some characteristics. Almost all exhibit temporal ordering and most feature only acyclic digraph topologies. We are therefore hopeful that our algorithm and experience can be transferred to workflow visualizations in other disciplines.

The domain experts involved in this work (also co-authors of this paper) identified the following requirements:

1. Given a number of workflows, can we obtain the most common patterns/motifs within those workflows?
2. For each motif, can we obtain the occurrence frequencies for that motif and information about which workflows they occur in?
3. Can we automatically create macro representations of those motifs, with the possibility of adding extra annotation to them?
4. Can motif patterns in a workflow be substituted by common macros to make the workflow more compact?

Depending on discipline and requirements, macros may be created by individuals based on their own knowledge about needs and usage. However, such an individualized approach would be impractical if applied to the curation of large collections of experimental workflows as found in data repositories. Yet, the availability of such data provides an opportunity for the computational identification of commonly occurring motifs in workflows. The statistics of such motifs offer useful guidance to motif selection when defining macros.

To carry out this work, information was extracted from a curated resource currently holding over 22,000 experimental design workflows (ArrayExpress) [2]. We used a subset of this collection, comprising 9,670 workflows considered to be well-formed with respect to correct connectivity and semantic annotation. This subset of workflows, available in the ISA-Tab format [29, 30], offers a good representation of experiment typology. These ISA-Tab files were processed and transferred to a graph database, which provides an optimized environment for storing graph structures and a query language optimized for graph traversal [8]. For this purpose, we selected *Neo4j* [3] – a freely available graph database providing a query language called Cypher, fast traversal algorithms and high scalability (up to several billions of nodes and edges on a single computer).

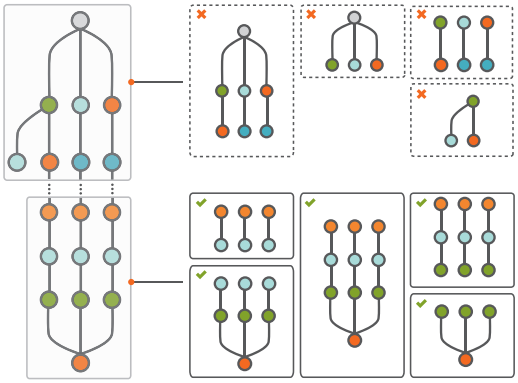


Fig. 3. In a workflow graph, some subgraphs can be considered as "legitimate motifs" while others cannot, depending on node type. Invalid motifs do not have the same output node type, as indicated by the differing colors. Conversely, valid motifs are those that have the same input and output node type.

Our system consists of three main functional steps, as depicted in Figure 1. (B, C, D) We algorithmically scan all workflows in the collection and extract all valid subgraphs as candidate motifs. The patterns and occurrence statistics of these motifs are recorded. The definition of validity and the extraction algorithm will be detailed in Sections 4.1 and 4.2. (E, F) We provide a user interface for domain experts to define macros by selecting motifs from an ordered list of candidates, based on the statistics computed as well as their domain knowledge about individual motifs and the context of their usage. This will be discussed in detail in Section 4.3. (G, H) We provide an automatic means for defining the pictogram of a macro by making use of the state transition information captured by the algorithm during the motif discovery phase. In addition, each macro will normally be displayed with a text label, which is defined by the domain expert through the user interface. The mechanism for automatic creation of a pictogram will be detailed in Section 5.

We created a tool, integrating the above requirements. The tool also allows for substitution of frequent motifs with their macro counterparts in experimental workflows. Furthermore, the tool can be deployed as a standalone software package or exposed through an API.

4 From Motifs to Macros

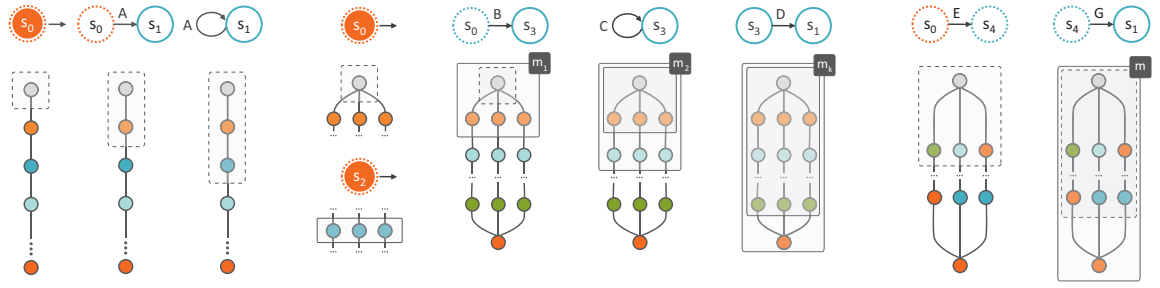
Motif discovery and substitution in graphs typically consists of four processes [25, 6]:

1. *Subgraph Generation*: Scan each graph for all possible n-node subgraphs.
2. *Motif Amalgamation*: Group together subgraphs that are topologically equivalent and generate a representative subgraph of each group (a motif).
3. *Macro Selection*: Assign a significance value to each motif with respect to other motifs in the collection (for instance, by computing frequency of occurrence) and select the most significant motifs as macros.
4. *Macro Substitution*: Search graphs for subgraphs that match with macros and replace them with that macro.

In applications such as biological network rendering, it can be safely assumed that subgraphs should be connected. By contrast, in applications such as very-large-scale integration (VLSI) design, such a condition is sometimes relaxed, as macros are often used to group elements for a cost-effective spatial placement. In general, the number of subgraphs can be rather large. That is why many algorithms only deal with small subgraphs, such as the 4-node subgraphs studied in [25].

4.1 Candidate Macros in Workflows

The workflows and macros considered in this work exhibit the following characteristics:



(a) singular branching (b) homogeneous branching (c) heterogeneous branching

Fig. 5. Examples of state transitions with different rules. Homogeneous edges are depicted by using the same color.

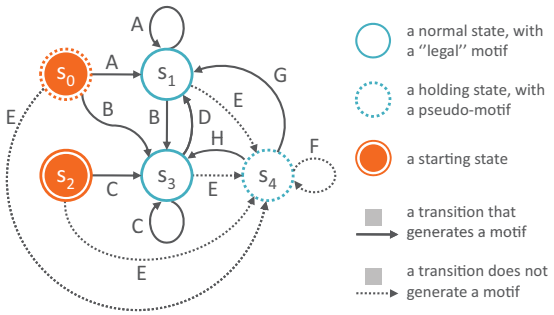


Fig. 4. The state transitions in motif generation.

1. A workflow is an acyclic digraph.
2. A macro must consist of at least two nodes/nodes.
3. Macros are not only topologically sensitive, but also semantically sensitive. In other words, two subgraphs are said to be in the same motif group only if they are isomorphic and every pair of corresponding nodes (and edges) are of the same type.
4. It is not necessary to have a path between every pair of nodes in a motif.
5. For each node in a macro, there must exist a path from the macro's input node, and a path to the macro's output node.
6. Each macro must have a single input type and a single output type (including *bundled* input and output).
7. A bundled input or output may contain any number of connection edges, but they must be of the same material type.
8. A macro may receive a bundled input converging from different preceding nodes and may deliver a bundled output branching off to different successive nodes.

By relying on these characteristics we can significantly reduce the number of subgraphs and motifs to be extracted in the *motif generation* stage of the algorithm. Figure 3 illustrates those subgraphs that are 'legitimate' candidates and others that are 'illegitimate'. In addition, owing to characteristic three described above, the *Motif Amalgamation* and *Macro Substitution* stages are much less onerous in comparison to subgraph matching based on topology only.

4.2 Motif Generation

Owing to the aforementioned conditions that are specific to workflows, we could not make use of existing motif generation software and algorithms such as those surveyed in [42]. A specialized motif generation algorithm was needed, and for that we adapted the widely accepted 'pattern-growth' approach [42]. We describe the algorithm by evolving a state transition diagram as shown in Figure 4.

Singular Branching (Rule A). As shown in Figure 5(a), the simplest workflow is perhaps a single path composed of n different steps (*e.g.* experimental steps). The algorithm can be activated from any node and will only move forward, following the flow of the work. Since a single node cannot be a macro, the search will park at state S_0 as illustrated in Figure 5(a), where the orange background indicates a starting state and the dotted outline implies that it is only a holding state and does not output a "legitimate motif".

When the algorithm encounters the next node n_1, it obtains the first 'legitimate motif', $m_1 = \langle n_0 \rightarrow n_1 \rangle$, and moves to the state S_0. The edge is labeled with A indicating that this follows rule **A**. From n_1, the algorithm then encounters n_2, it outputs another motif $m_2 = \langle n_0 \rightarrow n_1 \rightarrow n_2 \rangle$, and remains at the same state. For the workflow in Figure 5(a), this self-loop continues to generate motifs in growing sizes until the end of the path or the number of nodes in the motif reaches a predefined maximum.

Homogeneous Branching (Rules B, C, D). One extension of the singular branching case is multiple runs of the same sequence of steps, in parallel. When the work flows forward, the same type of edges connect to the next set of nodes. These edges can consist of bundled together as an input or output of a macro motif (see condition 7 in Section 4.1).

When an edge first branches to multiple nodes, as illustrated in Figure 4, the algorithm moves from state S_0 or S_1 to S_3, following a transition of singular branching to bundled homogeneous branching. This is referred to as rule **B**. S_3 indicates more than one edge is being bundled at this state and the number of edges may vary.

The algorithm will self-loop as long as the motif grows with such bundled edges. Rule **C** indicates a transition within the state of homogeneous branching. The number of edges in an edge bundle can change as long as there is more than one edge and they are of the same type.

When all bundled branches converge to a single node, the algorithm returns to state S_1. This transition is referred to as rule **D**. Figure 5(b) shows three examples of applying rules **B**, **C** and **D** respectively. Applying any of the three rules will result in a bigger motif than the previous one.

An additional state S_2 is included for a scenario when the algorithm starts with a row of parallel nodes with bundled input and output. We will discuss this scenario later in the context of reactivating the algorithm for bundled edges.

Heterogeneous Branching (Rules E, F, G, H). Recall conditions 6 and 7 in Section 4.1: a macro must have a single input and single output and each can be bundled edges of the same material type. When these two conditions are met, we can have heterogeneous flow within a macro. This subset of rules is designed to 'grow' this particular type of motif.

Rule **E** is applied when the algorithm first encounters an heterogeneous pattern. This transition leads to a holding state S_4 and it does not generate any motif. In this state, all nodes scanned so far are grouped as an interim pseudo-motif and output edges are placed in an interim pseudo-bundle.

The interim state is maintained by a self-loop transition; *i.e.*, rule **F** as long as the bundled edges remain heterogeneous. When the edges in an interim pseudo-bundle finally converge to a single node or a set of nodes with homogeneous output edges, the algorithm can leave the holding state S_4. Rule **G** defines the transition to singular branching state S_1, while rule **H** defines the homogeneous branching state S_3. Both rules will generate a new motif, which includes all nodes in the interim pseudo-motif and the newly encountered node(s).

Termination and Reactivation. The algorithm strictly follows the

Table 2. Performance of our motif-finding algorithm on graphs of varying size and at a range of search depths. Averages (last column) are taken across three graphs G1, G2 and G3 in each category. For each graph at each depth, the recorded time was the average time over five runs. A more detailed table is available in the supplementary materials.

Graph Size	Motif Depth	G1	G2	G3	Seconds
Small	3	0.018	0.017	0.024	0.02
	4	0.034	0.026	0.025	0.03
	5	0.048	0.038	0.042	0.04
	6	0.059	0.041	0.046	0.05
	7	0.075	0.049	0.049	0.06
	8	0.086	0.063	0.06	0.07
	9	0.095	0.064	0.062	0.07
	10	0.106	0.069	0.072	0.08
Medium	3	0.153	0.065	0.056	0.09
	4	0.212	0.097	0.085	0.13
	5	0.24	0.13	0.115	0.16
	6	0.293	0.161	0.138	0.2
	7	0.343	0.189	0.159	0.23
	8	0.389	0.219	0.178	0.26
	9	0.429	0.241	0.183	0.28
	10	0.477	0.287	0.192	0.32
Large	3	0.296	0.756	0.354	0.47
	4	0.393	1.062	0.45	0.64
	5	0.652	1.309	0.414	0.79
	6	0.632	1.528	0.42	0.86
	7	0.75	1.829	0.341	0.97
	8	0.809	2.028	0.396	1.08
	9	0.889	2.287	0.528	1.23
	10	1.047	2.489	0.604	1.38

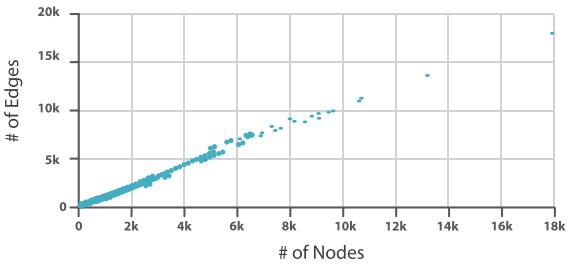


Fig. 6. Scatter plot showing the distribution of workflows (each depicted as a point) as node count versus edge count.

breadth-first search strategy. It may terminate in two situations: (a) when the predefined maximum depth that a macro may be at is reached; (b) when it encounters a node without an output edge (*i.e.*, the termination point of a workflow). The termination condition is tested in all states in Figure 4.

It is necessary to reactivate the algorithm by choosing each of the different nodes in a workflow as a starting node at state S_0. In addition, each bundle of edges of the same material type can also be a starting point at S_2. Although theoretically appropriate, invoking the algorithm recursively from a starting node of a workflow proved not to be feasible in practice. We therefore made use of a queue, which initially contains all nodes in a workflow. We fetch nodes from the queue one at time to invoke the algorithm from S_0. Every time the algorithm reaches state S_3, we have a set of homogeneous nodes that were just encountered (*e.g.*, $[n_{1a}, n_{1b}, n_{1c}]$ in Figure 5(b)). We store all k-node subsets of such a set ($k = 2, 3, \ldots$) in a list. When we finish with all individual nodes in the queue, we sort the list and remove redundant subsets. We then invoke the algorithm with the sorted and cleaned list from S_2. Note that each subset in the list is a 'legitimate motif, hence S_2 is not a holding state.

Performance. To evaluate the performance of the algorithm, we tested it against nine graphs representing biological workflows of varying sizes on a MacBook Pro with a 2.53GHz Intel Core i5 CPU and 8GB RAM. The nine graphs were divided into three groups, three small, three medium and three large, based on their node and edge counts. Figure 6 shows the distribution of workflows in our collection in terms of nodes (x-axis) and edges (y-axis). 93% of all workflows have a node count below 1000 and the average number of nodes per workflow is approximately 322. Given these statistics, we define *small* as 200 nodes or below (covering 58% of workflows); *medium* between 201 and 600 nodes (covering a further 28% of workflows); and *large* over 601 nodes (covering the remaining 14%).

For each of the nine selected graphs, motifs between a depth of three and ten were searched for, with each search repeated 5 times to account for any variability. Table 2 gives the runtime (in seconds) of applying our algorithm to the nine test graphs. The runtime is scalable to our collection of some 10,000 workflows, as the algorithm runs as a batch process. It also shows that the speed of our motif finding algorithm compares favorably the current best general purpose motif finding algorithms.

Our algorithm search space differs from the general purpose algorithms listed in Table 1, since we have introduced specific constraints on motif structure, taking into account the notion of node/edge types. Nevertheless, we could have theoretically used a two-stage method, by first using a general-purpose algorithm to identity an initial list of motifs, then filtering out those motifs that do not meet our requirements. Since our algorithm is generally faster than these general purpose algorithms, plus the computation to infer back the node/edge type is potentially large, there is no advantage to using this two-stage approach.

4.3 Selecting Macros from Candidate Macros

Given a list of motifs extracted using the algorithm in Section 4.2, we can categorize them into individual groups. Motifs in each group share the same subgraph topology, and the same set of nodes and edges in terms of their numbers and types. Each motif group thus becomes a candidate macro. It is important to emphasize that selecting macros from macro candidates requires a fair amount of knowledge about biology, biological experiments and the uses of workflows. In some cases, other information, such as the time when and context in which certain motifs appear frequently, may also feature in the selection process. Therefore, it is not sensible to make this selection process fully automatic. In this work, we provide a user interface to assist domain experts in selecting macros from a large list of candidates.

Following the advice of domain experts specialized in biological data curation, we understood the essential requirement for such a user interface is to provide key indicators for each macro candidate g_i, including the depth of its subgraph, $A_d(g_i)$, the total number of its occurrences in the data repository, $A_t(g_i)$, the number of workflows that contain it, $A_w(g_i)$, and its compression power $A_c(g_i)$. For each indicator, normally the higher value the indicator has, the more selectable the candidate is. However, when the comparison is not clear cut across different indicators, the domain experts will have to make an informed decision based on all the indicators as well as their tacit knowledge about the macro candidate (*e.g.*, biological semantics, importance in science, expected future usage). As shown in Figure 7B, the user interface displays each candidate with these indicators. The candidates are organized into columns, each representing a specific depth of the subgraphs in all macro candidates. Each candidate is shown with the basic pattern along with three indicators, A_t, A_w, A_c. The detailed structure of each macro candidate can be viewed on demand by using mouse interaction.

In order to help domain experts examine a large number of candidates speedily, we sort macros candidates in each column by a ranking score, allowing domain experts to inspect the most promising candidate first. The ranking score is based on indicator A_t, A_w, and A_c.

Indicator 1: Occurrences in the data repository. Indicator A_t returns the total number of times a motif has occurred across the entire database of workflows. It emphasizes the importance in selecting motifs that are highly used, inferring their functional importance.

Indicator 2: Workflow presence. Indicator A_w returns the total number of workflows in which a motif has appeared. This provides a measure of how widely a motif is used across different biological experiments, counterbalancing the possible distortion in situations where a motif is heavily used in a relatively small number of workflows.

Indicator 3: Compression Potential. Indicator A_c is calculated by first subtracting the number of nodes in the motif (A_n) by 1, since these

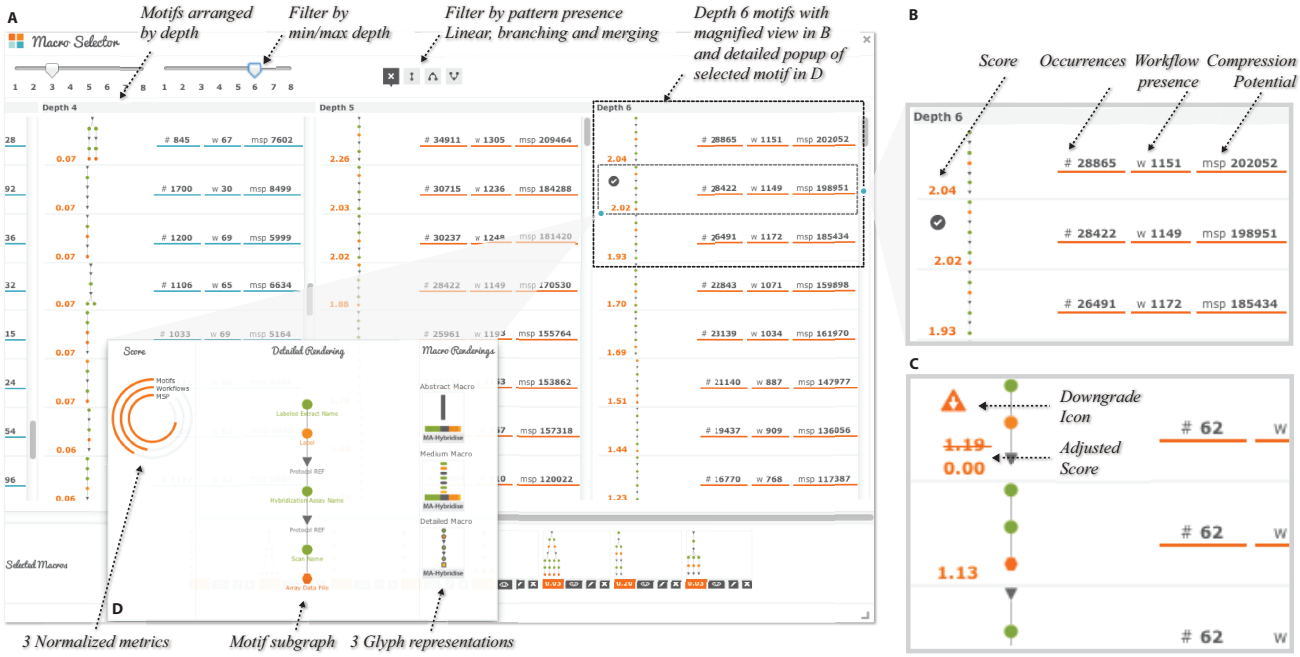


Fig. 7. AutoMacron provides a user interface (A) for domain experts to select macros from a list of computed motifs. As shown in the detailed view in (B), the overall score determines the order of motifs in the list. The three indicators are shown in unnormalized form in order to be semantically meaningful. The detail view (C) shows an example where a score is adjusted dynamically when a motif encompassing other motifs is selected. (D) shows a pop-up window for a specific macro, detailing its subgraph and three automatically generated pictograms for different levels of details in visualization.

nodes will be replaced by a single macro node, and then multiplying by its occurrence (A_t). It is written as $A_c(g_i) = (A_n(g_i) - 1) * A_t(g_i)$.

For each of these three indicators, we map it to a fixed range $[-1, 1]$ using a linear mapping based on the min-max range of each indicator, yielding three normalized metrics M_1, M_2 and M_3. These are combined into a single ranking score using a weighted average as:

$$S(g_i) = \sum_{k=1}^{3} \omega_k M_k(g_i)$$

where ω_k are three weights defined by users. Our system makes no assumptions about the merits of one indicator over another, so the default weights are set to one. We chose to have the score $S(g_i)$ in the range of $[-3, 3]$, as it helps domain experts to connect the score back to the three indicators.

Figure 7 shows a screenshot of the user interface on the left (A), and two detailed views with annotation on the right (B, C). A domain expert normally examines macro candidates with the largest depth value (the rightmost column) first. Once a motif, $\mathscr{M}$, is selected, it may affect the current results returned by the indicators. As such a motif, $\mathscr{M}$, may contain many other candidate motifs as subgraphs. It is important for the domain expert to be aware of the impact of this decision on those candidate motifs yet to be examined. The system thus updates the indicators of all those candidate motifs included in $\mathscr{M}$. Instead of modifying A_t and A_w directly, the system shows a corrected score calculated by considering the new values for A_t, A_w and A_c, implying the difference if all $\mathscr{M}$ were to be removed from the repository.

5 Macro Design

Having obtained a collection of motifs suitable for use as macros within our corpus of workflows, we now have the task of designing their visual representation. It is important to keep the users in mind and ensure that the design reflects what a typical user, in our case a biologist, would expect to find in a macro. We consulted domain experts as to the visual elements that users considered the most important to view. A number of attributes were identified and are listed below in order of importance:

1. an impression of topology/structure within a macro (*e.g.*, it may be an entirely linear path, a set of parallel paths, or it may contain branch/merge events);
2. an impression of types of nodes in a macro (*e.g.*, the overarching theme of the macro);
3. textual description, (*i.e.*, additional annotation to provide concrete semantic meaning and help in understanding);
4. an impression of density within a macro, (*i.e.*, the size of the corresponding subgraph).

Given the attributes listed above, we devised three design options, as illustrated in Figure 8, for creating pictograms. These pictograms are created automatically based on the states encountered when the motif is found by the motif generation algorithm in Section 4.2. When the algorithm moves from one state to another, the pictogram grows by adding a new visual component reflecting the subgraph pattern just encountered. We provide three alternative designs for each macro. The first option is pixel-based, the second is shape-based and the third is a miniature version of the subgraph. All three design options are stored as vector graphics, so they are suitable for multi-scale display, for example, through zooming operations. Textual descriptions of the macros are always provided by experts.

Figure 7(D) shows a pop-up window with a detailed view of a macro, and the three design options. The users can choose to have a fixed design for the macro, or have a multi-scale variation according to the level of detail at which a user is viewing the workflow.

Overview/Low detail. At the overview level, the fine-grained details of the workflow (*e.g.,* lines) utilize visual channels occupying a high spatial frequency. It is more effective for nodes in a graph to occupy low spatial frequencies, which will be distinguishable by a user. The pixel-based design option enables users to use visual channels that are visible and roughly distinguishable in low spatial frequencies. Although individual nodes and edges are not visible, the user can still gain a rough impression about the topology and node types.

Medium detail. At medium detail, users should be able to see more information through the shape-based design. The major steps from input to output (*i.e.*, following the state transitions) become more distinguishable. Each horizontal segment of the pictogram is colored by

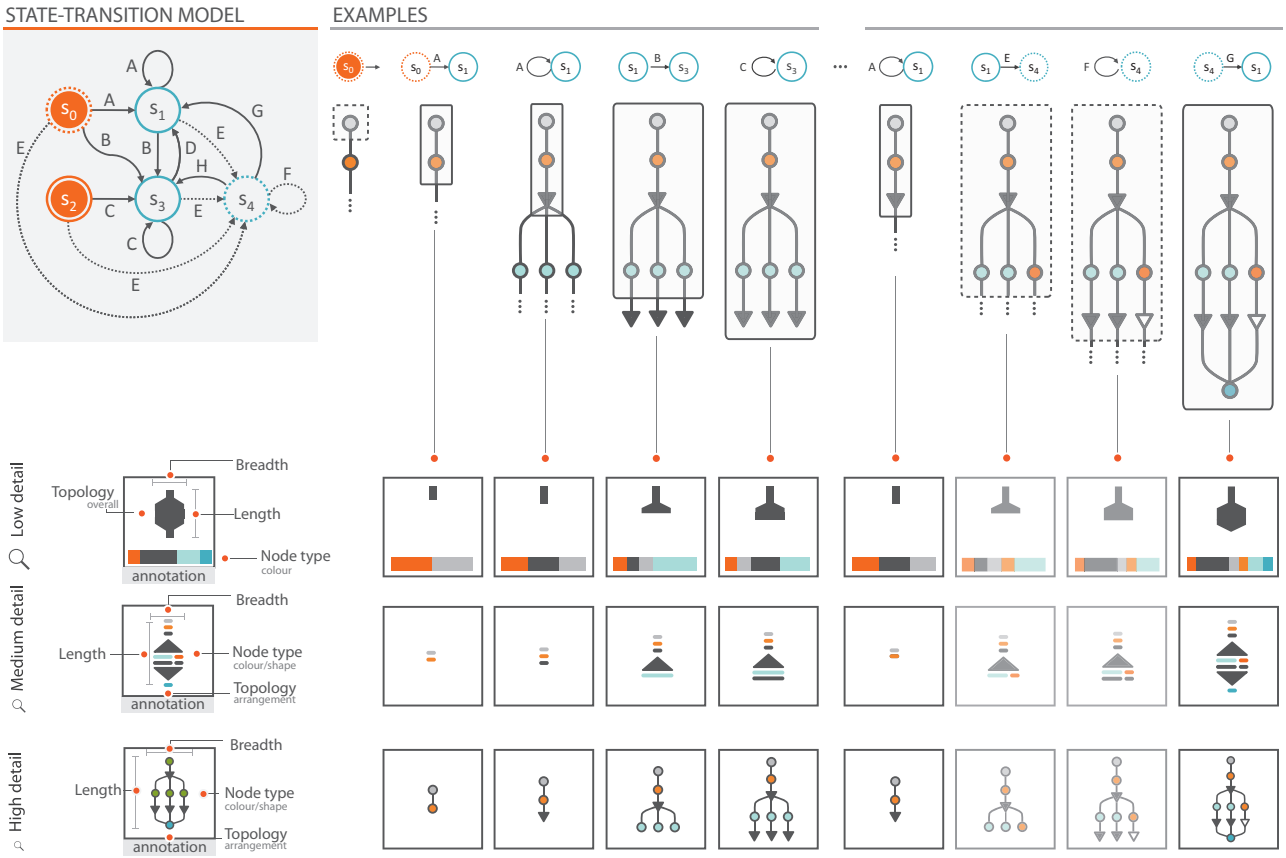


Fig. 8. From left to right: automatic generation of a micro pictogram based on the state transitions encountered. The lower three rows: Three design options for representing macros.

node type, branching and merging is shown with triangles of mirrored directionality, and heterogeneity of nodes is depicted in separated tracks of differing colors.

High detail. When a user zooms into a small region of the visualization, a miniature version of the subgraph becomes available, showing details of the topology and node types. This representation has a lot of high spatial frequency information and is only suitable for detailed examination in close up views.

6 Software Implementation and Use

To demonstrate our approach, we developed an open-source Java tool named *AutoMacron* that may be used in two modes: 1) standalone for those wishing to discover common motifs in a database of graphs; and 2) as an API for those wishing to integrate the utilities for motif discovery into their own software. We are also in the process of adding the capability to import formats other than ISA-Tab. The software provides the following functionalities:

1. Load files that have a handler (*e.g.,* ISA-Tab in our use case) into a graph database;
2. Analysis of all graphs in the database instance to determine the dominant motifs;
3. Allow for selection of macros from the pool of over-represented motifs found by the algorithm;
4. Visualize graphs both in uncompressed/compressed representations, and/or export those graphs in GraphML format for visualization in other environments;
5. Render differing images depending on the zoom level (semantic zooming [9, 40]). For this, we have extended the Prefuse visualization library [14].
6. Allow for search of a graph database for a user-defined semantically-annotated motif;
7. Export macros for use in other software.

Using AutoMacron, over 12,000 valid motifs were found in a collection of 9,670 existing workflows from ArrayExpress. From that set, those motifs scoring below zero with the aforementioned aggregated score $S(g_i)$ were removed from consideration leaving just over 400 candidates.

Further examination of these candidates was conducted by domain experts using the macro selection utility shown in Figure 7. Examination is aided through both presentation of the metrics and 'live' highlighting of motif representations as they occur in the original graph representation. Following the manual selection of 30 of these macros, the domain experts labeled each macro with a textual description, making the corresponding glyphs more meaningful and identifiable. These macros could then be used to substitute the more complex representations in the original graph in an effort to compress the representation.

Aside from the dedicated *AutoMacron* tool, the motif finding functionalities have been incorporated into ISAcreator, which is a tool used by domain scientists to annotate and curate biological experiment workflows and other necessary documentation. As shown in Figure 9, the user is given the option to compress an experiment workflow using the macros selected by domain experts for biological data curation.

7 Evaluation

Aided by direct collaboration and regular interaction with domain experts, development of the software followed an evolutionary prototyping process whereby users evaluated the prototype at every major stage of the development. For each prototype users contributed their feedback to the software and algorithm outputs.

In this section, we summarize the feedback given from the last iteration where we performed the following: 1) analysis of the performance

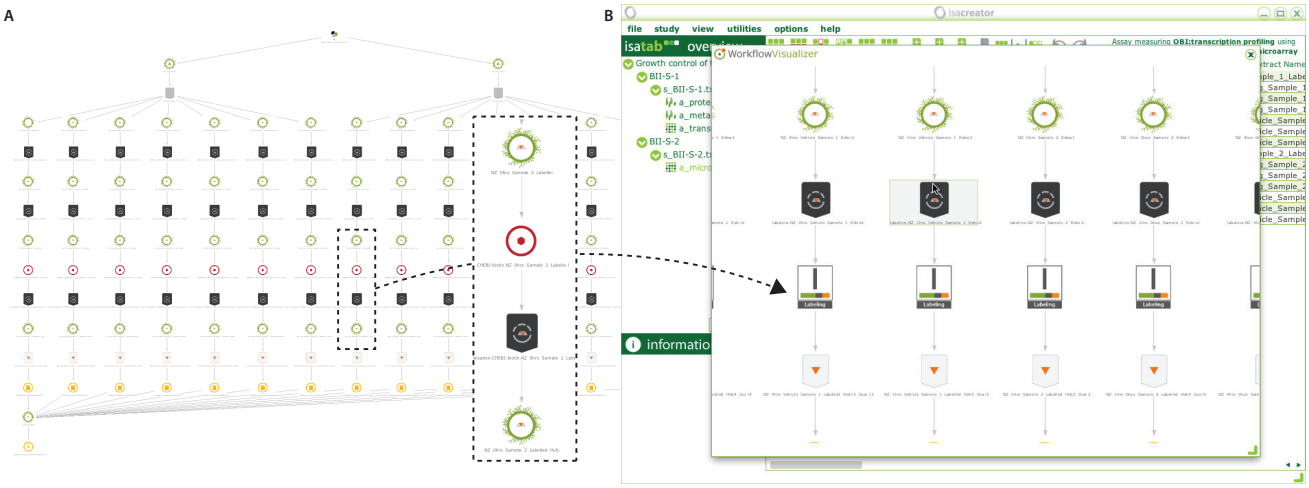


Fig. 9. A) A typical workflow, where a 4-node motif was selected as a macro using AutoMacron, which provides ISAcreator with an API. (B) A screenshot of ISAcreator, where each occurrence of the original 4-node motif has been replaced with a macro glyph.

of the algorithm presented in this paper in comparison with that of the best existing (and available) algorithm; and 2) observation of how the software met the initial requirements as identified in Section 3 by interviewing expert users.

7.1 Evaluation Against Existing Algorithms

We wished to test the performance of the best in class in existing algorithms with that of the algorithm presented in this paper, not necessarily just in speed, but also in what was found and how that compared with domain experts' expectations. We compared the performance of FANMOD [41] and the algorithm presented in this paper in an attempt to discover which motifs could be found and how they related to the expectations of domain experts.

Firstly, we ran a simple test graph through FANMOD, to detect three, four, five, six, seven and eight node graphs. In FANMOD, there is a requirement to run each analysis separately, searching for size 8 node subgraphs does not yield all three to eight node subgraphs. Figure 10B shows the output of a four node motif analysis and highlights FANMODs motif discovery match for a pattern highlighted in Figure 10A. As aforementioned, there is no notion of node or edge type, therefore all four node graphs with the same serial topology will be the same, even though they represent entirely different concepts. Additionally, FANMOD, typical of the existing class of algorithms, returns invalid results with respect to our definition of a motif as defined in Section 4.1.

Following this, we ran the same graph through AutoMacron to generate motifs up to depth 8. Note that FANMODs restriction is on node number (up to eight), whereas our algorithm can have potentially hundreds of nodes at depth eight. Figure 10C shows the HTML output from AutoMacron's analysis. The highlighted motif corresponds to those highlighted in Figures. 10A and B, we magnify the motif to show how our algorithm identifies the exact pattern with topology, node and edge type preserved.

Finally, we had the domain expert who inspected the graph manually and extract the motifs they would expect the algorithm to find. We compared what FANMOD and AutoMacron found with what the user expected. Our summary results are shown in table 3 with all analysis outputs available at https://bitbucket.org/eamonnmag/automacron-evaluation.

The results show that AutoMacron identifies less macros that FANMOD, however if one was to consider all the invalid motifs AutoMacron filters out due to incompatibility with our rule set (see Section 4.1), AutoMacron would have reported many more. When we consider just the 'valid' motifs, AutoMacron has many more than FANMOD (fifty compared with nineteen for FANMOD). This is as a direct result of the semantics added by AutoMacron. To illustrate,

Table 3. Results of motif identification by the domain expert, FANMOD and AutoMacron. Analysis included: **MIdent** - the ability of the domain expert to identify motif pictograms generated by the algorithms and match them to the original graph; and **UIdent** - the percentage of motifs found by the algorithm with respect to the number identified manually by the domain expert.

Source	Motif Id.	Valid Motifs	Acc	MIdent	UIdent	Time (ms)
Domain Expert	6	6	n/a	n/a	n/a	n/a
FANMOD	73	19	26%	4/19 (21%)	5/6 (83%)	1030*
AutoMacron	50	50	100%	49/50 (98%)	6/6 100%	119

consider a simple two node directed subgraph ($v \rightarrow v$) with 3 possible node types (n), AutoMacron could theoretically identify $n(n-1)$ different motifs whereas FANMOD and other algorithms already listed here could only identify one. Figure 10B and C show for instance how one 4-node motif found in FANMOD maps to five potential, but only one correct motif in AutoMacron. Overall both algorithms identified the majority of motifs expected by the user, not unexpected considering both mechanisms are exhaustive, however FANMOD struggled when it came to identifying the larger motifs, due to the node limit of eight, hence its UIdent value of 83%.

The user was also asked to identify motifs based solely on the pictograms representing topology (macros) output from each program. In 98% of occasions, the user could identify AutoMacron motifs, aided by both color coding and better topological arrangement. Conversely, with FANMOD the user was able to decode 21% of the outputs.

7.2 Evaluation Against User Requirements

The algorithm and tool were tested by two domain experts to determine how the software has met the four requirements identified in Section 3. For each requirement, we summarize their feedback below.

1. Given a number of workflows, can we obtain the most common patterns/motifs within those workflows?

   We tested AutoMacron on nearly 10,000 workflows, the software returned an abundance of motifs that were sorted by a score. The filtering tools helped us in finding motifs with specific topological events.'
2. For each motif, can we obtain the occurrence frequencies for that motif and information about which workflows they occur in?

   For each motif reported by the software, we were able to recover statistics about how often the pattern occurred as well, how many workflows the pattern appeared in and the names of these individual workflows.'
3. Can we automatically create macro representations of motifs with the added ability to add extra annotation?

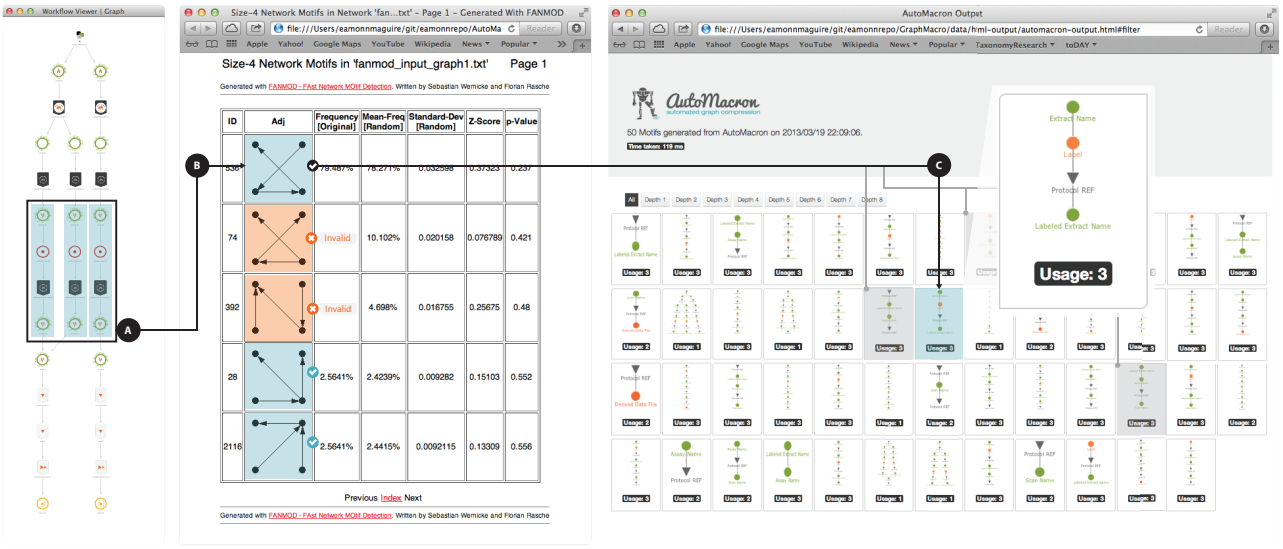


Fig. 10. A) A simple experimental graph visualized using the software from Maguire *et al.* [24]. In this example, we are showing a specific pattern and how that pattern is represented in both FANMOD and AutoMacron. B) HTML Output for 4-node motifs from FANMOD's analysis identifies 5 motifs with 4 nodes. 2 are incorrect (highlighted in orange) according to our rule set in Section 4.1. C) HTML output from AutoMacron's analysis which identifies all 50 motifs. We have highlighted the specific motif being searched for (in blue) matching the highlight pattern in A and B, and show the other serial 4 node motifs (highlighted in grey) that would erroneously be considered the same had it not been for preserved semantics of nodes and edges.

Each motif had three variations of 'macro' representations created automatically that could be used depending on the resolution available. We were able to add small pieces of text to these to help identification of the function of these macros, e.g. *labeling, extraction, or scanning.'*

4. Can motif patterns in a workflow be substituted by common macros to make the workflow more compact?

   The software provides a function to show compressed views of a workflow which automatically substitutes motif patterns with their macro representation. The function was also integrated to ISAcreator allowing us to serve out compressed representations of workflows to our users.

Further to this, users added that the software allowed them to identify erroneous annotations very quickly. For example, through inspecting the motifs, one domain expert discovered a prominent motif with nodes in the wrong position. This was only detectable via the algorithm's ability to maintain node type. On this matter, the domain expert commented "*Being able to detect such errors in annotation so quickly has enabled us to build scripts to fix those records and improve annotation quality. We can use this tool to find and fix inaccuracies in biological workflows much more quickly than we could before.*"

In many ways, the domain experts regard AutoMacron as a time-saving tool. Macro glyphs have a similar function to traffic signs. Domain experts normally expect certain macro glyphs in certain workflows. Although glyphs are small, they provide more assurance than observing detailed subgraphs directly because macros are computationally determined. It has helped them to construct the motif space computationally, which would otherwise takes years of effort. It has enabled them to explore the space of motifs efficiently with the aid of the ordered recommendations by the system. It has allowed them to create macros quickly without the need to design a pictogram for each macro. In the medium term, their everyday tasks, such as error checking, comparison, and identifying best practice, can be performed more speedily. In the long run, they also hope that this will bring benefits to the wider community; for instance, in experimental documentation and scientific publication.

While the discipline of biology has led the way in collecting workflows as part of data curation and sharing, we anticipate that some other disciplines will follow this trend soon. Our approach to workflow visualization in general and macro generation in particular can be adapted for other types of workflows if they have been curated.

8 Conclusions and Future Work

In this work, we introduced a new approach to macro creation aimed at reducing visual complexity in workflow visualizations. We developed a novel algorithm to discover motifs, with discrimination of node and edge type in a large collection of graphs. The algorithm was specifically designed for motifs in workflows and performed more efficiently than general-purpose motif finding algorithms.

We used a statistically-informed approach and an intuitive user interface to help domain experts in selecting macros from motif candidates. We devised a novel design method for automated creation of pictograms for macros by making use of the state transition information obtained by the motif discovery algorithm. We implemented our methods in a software system, available either through a dedicate graphical user interface (GUI) or through an API. Domain experts were able to use the system both to generate macros for compression of workflows and to find errors in a large corpus of existing workflows. Additionally, the selected macros and graph substitution algorithms are integrated into the ISA tools suite used by a large body of experimental biologists [30].

Our future work will cover two directions: the use of motif occurrence information to compare graphs based on 'motif fingerprints'; and the use of macro-based standardized constructs in an experiment builder to simplify and improve the construction of workflows.

Acknowledgments

This work was supported by funding from the BBSRC and NERC, grants BB/I000917/1 and BB/I025840/1.

References

[1] Graphviz. http://graphviz.org/, 2012.
[2] Arrayexpress. http://www.ebi.ac.uk/arrayexpress/, Last Accessed: June 2013.
[3] Neo4j. http://www.neo4j.org, Last Accessed: June 2013.
[4] Oxford dictionaries, *Macro* definition. http://oxforddictionaries.com/definition/english/macro, Last Accessed: June 2013.

[5] N. Alon, P. Dao, I. Hajirasouliha, F. Hormozdiari, and S. Sahinalp. Biomolecular network motif counting and discovery by color coding. *Bioinformatics (Oxford, England)*, 24(13):241–249, 2008.

[6] U. Alon. Network motifs: theory and experimental approaches. *Nature reviews. Genetics*, 8(6):450–461, 2007.

[7] I. Altintas, C. Berkley, E. Jaeger, M. Jones, B. Ludascher, and S. Mock. Kepler: an extensible system for design and execution of scientific workflows. *16th International Conference on Scientific and Statistical Database Management.*, pages 423–424, 2004.

[8] S. Batra and C. Tyagi. Comparative analysis of relational and graph databases. *Internation Journal of Soft Computing and Engineering (IJSCE)*, 2, 2012.

[9] B. B. Bederson and J. D. Hollan. Pad++: a zooming graphical interface for exploring alternate interface physics. *UIST '94 Proceedings of the 7th annual ACM symposium on User interface software and technology*, pages 17–26, 1994.

[10] S. P. Callahan, J. Freire, E. Santos, C. E. Scheidegger, C. T. Silva, and H. T. Vo. VisTrails: visualization meets data management. *Proceedings of the 2006 ACM SIGMOD international conference on Management of data*, pages 745–747, 2006.

[11] J. Chen, W. Hsu, M. Lee, and S. Ng. NeMoFinder: dissecting genome-wide protein-protein interactions with meso-scale network motifs. *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 106–115, 2006.

[12] C. Dunne and B. Shneiderman. Motif simplification: improving network visualization readability with fan, connector, and clique glyphs. *CHI '13: Proc. SIGCHI Conference on Human Factors in Computing Systems*, 2013.

[13] J. Grochow and M. Kellis. Network motif discovery using subgraph enumeration and symmetry-breaking. *Proceedings of the 11th annual international conference on Research in computational molecular biology*, pages 92–9106, 2007.

[14] J. Heer, S. K. Card, and J. A. Landay. prefuse: A toolkit for interactive information visualization. In *Proceedings of ACM CHI*, pages 421–430, 2005.

[15] D. Holten. Hierarchical edge bundles: visualization of adjacency relations in hierarchical data. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):741–748, 2006.

[16] D. Hull, K. Wolstencroft, R. Stevens, C. Goble, M. Pocock, P. Li, and T. Oinn. Taverna: a tool for building and running workflows of services. *Nucleic acids research*, 34(Web Server issue):W729–W732, 2006.

[17] Z. Kashani, H. Ahrabian, E. Elahi, N. Abbas, E. Ansari, S. Asadi, S. Mohammadi, F. Schreiber, and M. Ali. Kavosh: a new algorithm for finding network motifs. *BMC Bioinformatics*, 10, 2009.

[18] N. Kashtan, S. Itzkovitz, R. Milo, and U. Alon. Efficient sampling algorithm for estimating subgraph concentrations and detecting network motifs. *Bioinformatics*, 20(11):1746–1758, 2004.

[19] N. Kashtan, S. Itzkovitz, R. Milo, and U. Alon. Network motif detection tool mfinder tool guide. *Weizmann Institute of Science: Depts of Mol Cell Bio and Comp Sci & Applied Math, Rehovot, Israel (2002-2005)*, 2005.

[20] D. Koop, C. Scheidegger, S. Callahan, J. Freire, and C. Silva. VisComplete: automating suggestions for visualization pipelines. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1691–1698, 2008.

[21] M. Kuramochi and G. Karypis. Finding frequent patterns in a large sparse graph*. *Data mining and knowledge discovery*, 11(3):243–271, 2005.

[22] J. Leskovec and C. Faloutsos. Sampling from large graphs. *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 631–636, 2006.

[23] A. Ma'ayan, S. Jenkins, R. Webb, S. Berger, S. Purushothaman, N. Abul-Husn, J. Posner, T. Flores, and R. Iyengar. Snavi: Desktop application for analysis and visualization of large-scale signaling networks. *BMC Systems Biology*, 3, 2009.

[24] E. Maguire, P. Rocca-Serra, S.-A. Sansone, J. Davies, and M. Chen. Taxonomy-based glyph design — with a case study on visualizing workflows of biological experiments. *IEEE Transactions on Visualization and Computer Graphics*, 18(12), 2012.

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

[26] S. Omidi, F. Schreiber, and M. Ali. MODA: an efficient algorithm for network motif discovery in biological networks. *Genes & genetic systems*, 84(5):385–395, 2009.

[27] G. Pavlopoulos, M. Secrier, C. Moschopoulos, T. Soldatos, S. Kossida, J. Aerts, R. Schneider, and P. Bagos. Using graph theory to analyze biological networks. *BioData mining*, 4, 2011.

[28] P. Ribeiro and F. Silva. G-tries: an efficient data structure for discovering network motifs. *Proceedings of the 2010 ACM Symposium on Applied Computing*, pages 1559–1566, 2010.

[29] P. Rocca-Serra, E. Maguire, S.-A. Sansone, and *et al*. ISA software suite: supporting standards-compliant experimental annotation and enabling curation at the community level. *Bioinformatics*, 26(18), 2010.

[30] S.-A. Sansone, P. Rocca-Serra, D. Field, E. Maguire, and *et al*. Toward interoperable bioscience data. *Nature genetics*, 44(2):121–6, 2012.

[31] E. Santos, L. Lins, J. Ahrens, J. Freire, and C. Silva. VisMashup: streamlining the creation of custom visualization applications. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1539–1546, 2009.

[32] C. Scheidegger, H. Vo, D. Koop, J. Freire, and C. Silva. Querying and creating visualizations by analogy. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1560–1567, 2007.

[33] F. Schreiber and H. Schwöbbermeyer. Mavisto: a tool for the exploration of network motifs. *Bioinformatics*, 21(17):3572–3574, 2005.

[34] W. J. Schroeder, K. M. Martin, and W. E. Lorensen. The design and implementation of an object-oriented toolkit for 3D graphics and visualization. *VIS '96 Proceedings of the 7th conference on Visualization '96*, pages 93–ff., 1996.

[35] B. Shneiderman. The eyes have it: A task by data type taxonomy for information visualization. *Proceedings IEEE Workshop Visual Languages*, pages 336–343, 1996.

[36] B. Shneiderman and A. Aris. Network visualization by semantic substrates. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):733–740, 2006.

[37] A. Telea and J. J. van Wijk. SMARTLINK: an agent for supporting dataflow application construction. *Springer*, pages 189–198, 2000.

[38] T. von Landesberger and M. Görner. A system for interactive visual analysis of large graphs using motifs in graph editing and aggregation. *Proceedings of Vision Modeling Visualization Workshop*, 2009.

[39] T. von Landesberger and A. Kuijper. Visual analysis of large graphs: State-of-the-art and future research challenges. *Computer Graphics Forum*, 30(6):1719–1749, 2011.

[40] C. Weaver. Building highly-coordinated visualizations in improvise. *IEEE Symposium on Information Visualization*, pages 159–166, 2004.

[41] S. Wernicke and F. Rasche. Fanmod: a tool for fast network motif detection. *Bioinformatics*, 22(9):1152–1153, 2006.

[42] E. Wong, B. Baur, S. Quader, and C. Huang. Biological network motif detection: principles and practice. *Briefings in Bioinformatics*, 13(2):202–215, 2012.

F Copy of [MRSSC14]:

Redesigning the Sequence Logo with Glyph-based Approaches to Aid Interpretation

Eurographics Conference on Visualization (EuroVis) (2014)
N. Elmqvist, M. Hlawitschka, and J. Kennedy (Editors)

Redesigning the Sequence Logo with Glyph-based Approaches to Aid Interpretation

Eamonn Maguire, Philippe Rocca-Serra, Susanna-Assunta Sansone, and Min Chen

Oxford e-Research Centre, University of Oxford, UK

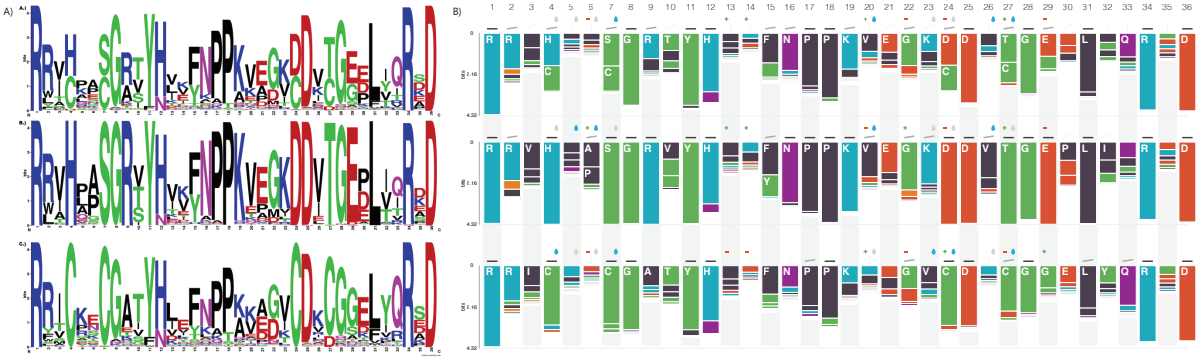


Figure 1: *A) A 'traditional' sequence logo from [Bio13] showing: top - the consensus across 1809 protein sequences; middle - gram negative bacteria; and bottom - gram positive bacteria. B) Our sequence logo visualizing the same data as in A. This approach keeps in place the idea of the original sequence logo but removes the dominance of letters for space filling and adds glyphs to visualize overall properties of the dominating amino acids at a sequence location (e.g. charge and hydropathy).*

Abstract

Sequence logos have been a prominent visualization tool in biological research since their inception two decades ago. Their primary use is in communicating conservation of biological sequences (protein, DNA or RNA), to indicate largest conservation at particular positions - namely places where only ever one or two possible residues (nucleotides or amino acids) are observed. Conservation is indicative of functional importance, as changes, being selected against, reveal a loss of fitness for living organism or cells. Criticism of the sequence logo has long existed, largely directed towards perception problems caused through use of letter height to indicate frequency. Here, we present a solution for use as a static image in publications or interactively on the web to address the reported flaws of the sequence logo. In addition to our improvements, we propose glyph based enhancements, to highlight qualitatively relevant chemical insights resulting from residue substitution between sequences.

1 Introduction

Advances in sequencing technologies have turned a time consuming and expensive process into a quasi main stream commodity. Next generation sequencing allows exploration of the blueprints of life in fascinating new ways, making the genome of thousands of species available to computational analysis. Of particular interest to scientists is the detection of regions of genomes under selective pressure and to ascertain which regions of DNA, RNA and proteins are functionally and/or structurally important across species (e.g., the conserved regions of a protein in mice, human, rat and horse).

Although our ability to process this data has increased, the methods used to visualize and report this conservation in scientific publications has not moved on in over 20 years. This technique, named the named 'sequence logo' [SS90, Sch02]

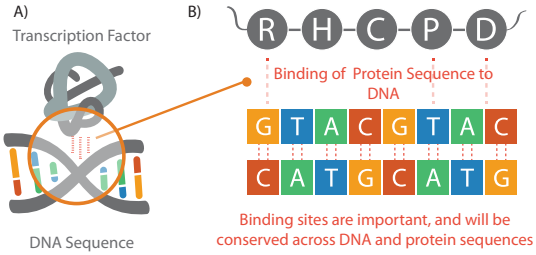


Figure 2: *A) An abstraction of a transcription binding site where a protein binds to a region of DNA. B) Certain residues, amino acids for the protein and DNA nucleotides for DNA are important in creating a bond between the DNA and protein to enable some function to occur.*

is comprised of a series of letters stacked on top of each other. It is these letters and their arrangement which are cause for concern due to their perception problems [Bio13] when too stretched or squashed to interpret. We now introduce a new sequence logo design aimed at addressing their perception issues. This has been implemented in a JavaScript library that can be easily customised and incorporated in to online resources for interactive exploration, or exported as a scalable vector graphic (SVG) for inclusion in traditional journal publications.

2 Background

Transcription Factors are proteins with a central function: they can, with high specificity recognize and bind to DNA regions and trigger transcription of DNA into messenger RNA. Those genomic regions are known as transcription factor binding sites (TFBS) and their identification is essential to the reconstruction of transcriptional regulatory networks of cells. Sequence comparison across organisms can be used to highlight areas of conservation to pinpoint unknown TFBS. Low variability at some position i generally points to a level of functional importance in that region of DNA or protein.

In order to visualize these binding sites, sequence logos were devised as a visualization technique to view a position weight matrix (PWM), which specifies for each position in a sequence the likelihood of observing a particular residue (DNA/RNA nucleotide or amino acid). From this matrix, a simple frequency-based sequence logo can be drawn by iterating through each position i in the sequence. For each position, the height of each letter a can be calculated as $p(a) \times R$ where $p(a)$ is the measured probability of a occurring at position i, and R is the maximum height at position i.

To determine the maximum height at position i, the principal mechanism is to compute its 'information content', a term derived from Claude Shannon's information theory [Sha48]. Information content represents the level of certainty at a specific position i by measuring how much the probability distribution of the detected letters in that position deviates from a uniform distribution, where all valid letters have an equiprobable chance of occurring. Given s valid letters, $a_j, j = 1, 2, \ldots, s$, in an alphabet $\mathbb{Z}$ and their probability distribution function, $p(a_j)$, the level of uncertainty can be defined by the following entropy measure:

$$H(\mathbb{Z}) = -\sum_{j=1}^{s} p(a_j) log_2 p(a_j)$$

where we use the base 2 logarithm, and the unit for the uncertainty H is 'bit'. For an equiprobable distribution, $p(a_j) = 1/s$ for all letters in the alphabet $\mathbb{Z}$. The above entropy measure yields the maximum uncertainty (commonly referred to as the maximum entropy), $H_{max}(s) = log_2(s)$.

For example, for 20 types of amino acids, we have $H_{max}(20) = 4.32$ bits, and for 4 types of nucleic acids, $H_{max}(4) = 2$ bits.

Since it is unlikely that there is an equiprobable distribution of letters at each position i, the actual uncertainty is usually less than H_{max}. Hence the difference between the maximum uncertainty and the measured uncertainty can be defined as the certainty, which is visually encoded as the total height R_i for all letters at position i.

A high stack indicates a high level of certainty with a strong preference for one or at most two nucleotides. A low stack implies a low level of certainty with many possibilities. The idea behind this visual design is that the highly conserved positions 'pop out' more due to their greater height.

3 Related Work

There have been a few attempts to redesign the sequence logo in the past, with most having focused on modifying how such letters are positioned on the y-axis with Kullback Leibler [KL51], position-specific scoring matrix (PSSM) [FZL*04], and berry logos [BHLB06, Ber13]. These in part can alleviate some of the perception issues with the sequence logo, though all continue to use letters to represent size apart from LogoBar by Perez *et al* [PBKB06]. We extend on the work by Perez through support for DNA/RNA sequences, glyphs, an evaluation and a web-based implementation.

4 Motivation

The overall goal of this work is to improve the sequence logo to address their perception issues [Bio13]. Our approach is a three stage process with a domain-expert always in the loop. Those stages are:

1. improving the current *sequence logo* design to address perceptual issues caused by: a) use of letter size to represent value; and b) the arrangement of letters on the y-axis which can make residues look more conserved than others based on the number of letters it is stacked upon;
2. adding *glyphs*, which are visual objects used to depict attributes of a data record [BKC*12] to annotate positions

Figure 3: *The amount of ink used in a letter can influence the perceived 'weight' of the letter. Within a normalized plotting, 'W' and 'M' take up 3 times as much ink as 'I', 'J' and 'L', and twice as much as 'S'.*

when comparing within sets of sequence logos, for instance to highlight qualitative changes of residue; and

3. an Ã§ of the redesigned sequence logo with both advanced and 'naive' sequence logo users to ascertain the benefits of the new design.

5 Design

5.1 Sequence Logo

The frailties of the original sequence logo reside in the following problems:

1. *Using letter size to show value* — letters have an undesirable property in that they are of differing densities. We have quantified this effect (see Figure 3) and found that and 'R', 'H' and 'W' take up to about three times as much ink as 'I' ,'J' or 'L' for example. When used for space filling, this ultimately leads to perception issues as denser letters will be more visible. Take Figure 1 - at position 31, 'L' (Leucine) is the dominant amino acid, however it is harder to see compared with 'R' or 'D' at positions 34 and 36 respectively. Use of letters also means that when there are many positions to display, letters become 'squashed' so it is often impossible to read them;
2. *Placement of the most dominant letter on top of the less dominant letters, leading to possible misinterpretation of the size of a letter depending on where it is placed on the y axis* — this is particularly evident in Figure 1 at positions 2 and 4 of the top sequence logo where we may compare heights between 'R' (Arginine) and 'H' (Histidine). In plot A, 'H' is positioned higher in the chart due to the letters below it. This gives the impression that the conservation of 'H' at position 4 is greater than the conservation of 'R' at position 2. In reality, 'R' is more conserved at position 2, which is evident in plot B.

We propose a design that retains the ideas of the original sequence logo, so as to ease uptake of a new representation, whilst subtly overcoming its perception issues. Our design, shown in Figure 4 uses filled bars to represent size in place of the letters similar. The top 'residues' (amino acids or nucleotides) are positioned at the top or bottom of the plot so that the conserved sequence can be read more easily than in the original logo. The colours of the bars are a function of the type of amino acid, however these are not fixed and can be changed depending on how the user wishes to visually group residues.

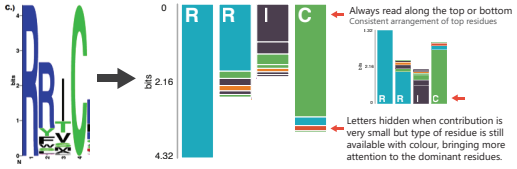


Figure 4: *Redesign of the sequence logo layout from the original (left) to the new version.*

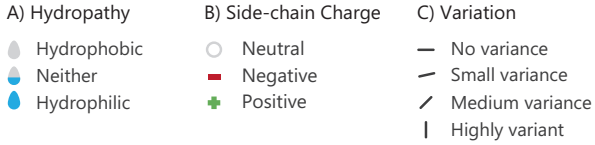


Figure 5: *Glyphs indicating dominating properties of residues at each position. A) Hydropathy (relevant for amino acids). B) Side-chain charge (relevant for amino acids only). C) Variation: indicated using 'GestaltLines'.*

5.2 Glyphs

Amino acid residue side-chain charge and hydrophobicity strongly affect protein folding and its final 3D conformation, so any significant change in those qualities may result in a significant functional change. Therefore, providing a visual aid to remind users of those dimensions presents opportunities to enhance interpretation.

Additionally, we experiment with another glyph-based approach, named 'Gestalt Lines' [BNRS13] to provide an additional indicator of variance.

The glyphs will be positioned above each position to give an overall impression of whether or not there are any dominating characteristics of the amino acids present at a position. These glyphs are shown in Figure 5. Glyphs for hydropathy and side-chain charge are only present in amino acid sequence logos.

6 Implementation

We have implemented an open source JavaScript library (https://github.com/ISA-tools/SequenceLogoVis), built using RaphaelJS [Rap14]. This library renders protein, DNA or RNA sequences as seen in Figures 1B and 6 and a set of parameters allows customization of the visualization, as seen in Figure 6. The web version supports interactivity, providing detail on demand [Shn96] for each position to show the distribution of letters at each position in the sequence alongside contextual information about the amino acids or nucleotides. Additionally, the sequence logo can be saved as a scalable vector graphic (SVG) for inclusion in publications.

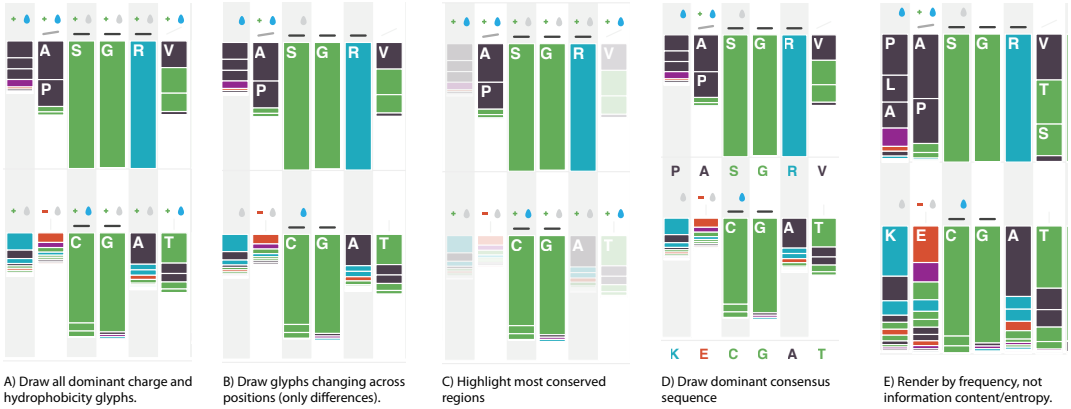


Figure 6: *The rendering library contains many options allowing users to configure the sequence logo.*

7 Evaluation

In order to assess if our changes helped biologists read sequence logos, we devised a survey focused on evaluating the ability to compare the size of letters in the old version of the sequence logo (see Figure 1A) and the new version (Figure 1B). We used the same data to create the two versions of the sequence logo and asked participants to determine which letter between two was the largest. Our hypothesis, as stated earlier, is that it is harder to compare the letters than it is to compare blocks. 41 scientists (15 bioinformaticians, 23 biologists and 3 computer scientists) with varying levels of familiarity with sequence logos, took part in the survey.

Figure 7 shows survey results (upper part) and test images (lower part). In three out of the four questions given to users, a higher number of correct answers were recorded with the new sequence logo. The gain was significant for questions 2 and 3, with up to twice as many correct answers with the new representation. The exception was in question four, where there was a small (8%) advantage gained by using the original version. The greater density of 'G' over 'L' may have lead to more people choosing correctly by chance, explaining the observation. More thorough experimentation would be needed to validate this hypothesis however.

The feedback from users regarding the glyphs was also largely positive with: 80% of respondents agreeing that showing hydropathy was useful; 83% agreeing that showing side-chain charge was useful; and 59% indicating that the variance glyph using GestaltLines was useful. The feedback has led to the removal of the GestaltLines since the level of variance can be adequately determined using bar height.

The approval of our redesign was also measured with users asked to give their preference between Figure 1A and B. 95% of respondents said that they preferred the new representation over the original, citing amongst others, 'cleanliness' and 'clarity' as the major factor in their decision.

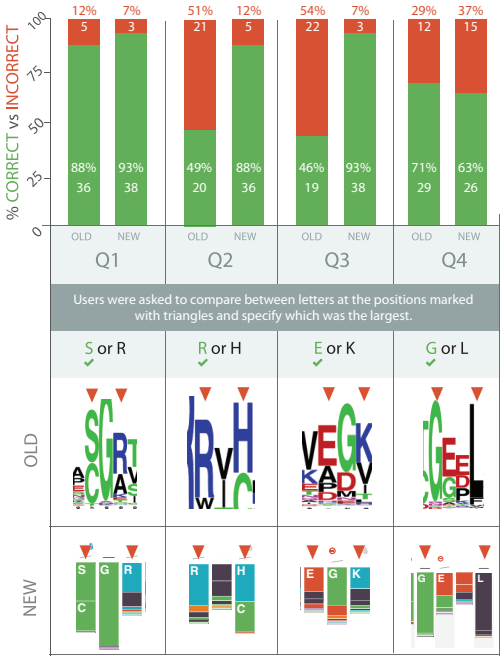


Figure 7: *Evaluation responses showing the ability to discriminate between conservation levels in the two designs of the sequence logo, old/original and new.*

8 Conclusion

We have presented a new design for the sequence logo that incorporates glyph-based techniques to aid interpretation. We have also provided an implementation of this new logo that can be immediately incorporated in to the workflows of scientists for interactive use or inclusion in publications. Our usability tests showed that users generally found consensus sequence reading tasks easier with the new sequence logo. Users, on the whole, agreed that the new representation did a better job of improving the display of salient information.

References

[Ber13] BERRYLOGO: https://github.com/leipzig/berrylogo, 2013. 2

[BHLB06] BERRY C., HANNENHALLI S., LEIPZIG J., BUSHMAN F. D.: Selection of target sites for mobile dna integration in the human genome. *PLoS computational biology 2*, 11 (2006), e157. 2

[Bio13] BIOVIS: http://www.biovis.net/year/2013/info/redesign-contest, 2013. 1, 2

[BKC*12] BORGO R., KEHRER J., CHUNG D. H., MAGUIRE E., LARAMEE R. S., HAUSER H., WARD M., CHEN M.: Glyph-based visualization: Foundations, design guidelines, techniques and applications. In *Eurographics 2013-State of the Art Reports* (2012), The Eurographics Association, pp. 39–63. 2

[BNRS13] BRANDES U., NICK B., ROCKSTROH B., STEFFEN A.: Gestaltlines. In *Computer Graphics Forum* (2013), vol. 32, Wiley Online Library, pp. 171–180. 3

[FZL*04] FUJII K., ZHU G., LIU Y., HALLAM J., CHEN L., HERRERO J., SHAW S.: Kinase peptide specificity: improved determination and relevance to protein phosphorylation. *Proceedings of the National Academy of Sciences of the United States of America 101*, 38 (2004), 13744–13749. 2

[KL51] KULLBACK S., LEIBLER R. A.: On information and sufficiency. *The Annals of Mathematical Statistics* (1951), 79–86. 2

[PBKB06] PÉREZ-BERCOFF Å., KOCH J., BÜRGLIN T. R.: Logobar: bar graph visualization of protein logos with gaps. *Bioinformatics 22*, 1 (2006), 112–114. 2

[Rap14] RAPHAELJS: http://raphaeljs.com/, 2014. 3

[Sch02] SCHNEIDER T. D.: Consensus sequence zen. *Appl Bioinformatics. 1* (2002), 111–119. 1

[Sha48] SHANNON C.: A mathematical theory of communication. *Bell Systems Technical Journal. 27* (1948), 379–423. 2

[Shn96] SHNEIDERMAN B.: The eyes have it: A task by data type taxonomy for information visualizations. In *Visual Languages, 1996. Proceedings., IEEE Symposium on* (1996), IEEE, pp. 336–343. 3

[SS90] SCHNEIDER T. D., STEPHENS R. M.: Sequence logos: A new way to display consensus sequences. *Nucleic Acids Res. 18* (1990), 6097–6100. 1

G Copy of [ARMC14]:

Comparing Three Designs of Macro-Glyphs for Poetry Visualization

Eurographics Conference on Visualization (EuroVis) (2014)
N. Elmqvist, M. Hlawitschka, and J. Kennedy (Editors)

Comparing Three Designs of Macro-Glyphs for Poetry Visualization

Alfie Abdul-Rahman, Eamonn Maguire, and Min Chen

Oxford e-Research Centre, University of Oxford, United Kingdom

Abstract

Glyphs have been successfully used in poetry visualization for depicting the characteristics and positions of each phonetic articulation in relation to the human vocal system. While existing glyph designs provide visual representations for detailed observation and external memorization of the dynamics throughout a poem, they are less effective for observing the relationship and variance between different lines in a poem and in comparing different poems. In this short paper, we present three designs of macro-glyphs for summarizing the spatio-temporal dynamics at the level of poetic lines. In particular, we use statistics of a collection of poems to guide and optimize the designs. We report our comparative study on the effectiveness of these three designs.

1 Introduction

In poetry, close reading is a literary form of "data exploration", in which scholars pay concerted attention collaboratively to various individual linguistic, literary and sociological features, as enumerated above, as well as to the interplay and relationships among these features [ARLC*13]. Poetry visualization can support close reading by enabling more effective observation and external memorization of a collection of features, allowing scholars to devote more cognitive capacity to explore different interpretations and hypotheses. This work focuses on the visualization of phonemic features, the details of which are difficult to see and remember because such features are by nature auditory and temporal, but not visual; and in English, they are indirectly encoded in the original text. Some poetry scholars refer the process of study phonemic features in close reading as "chewing the sound".

In [ARLC*13], mini-glyphs were used to represent three phonemic variables of vowels and consonants, and the transition between two consecutive vowels or consonants. When visualizing a poem, these mini-glyphs annotate all phonetic symbols translated from the original text. While they are particularly useful for detailed observation and external memorization of the phonetic dynamics of a poem at the level of phonemes and words, they are less effective in observing the relationships between the different lines in a poem, e.g., the level of dynamics of a poem and individual lines, similarity, diversity and changes of phonetic structure at the level of lines. Hence it is desirable to provide visual representations at a level higher than the mini-glyphs in [ARLC*13].

In this paper, we propose to introduce *macro-glyphs* for encoding multivariate time series associated individual lines in a poem. We considered three main glyph designs, namely *static radial glyph*, *animated transitions*, and *static transitions with temporal highlight*. We used various statistical indicators of 35 poems (range from sonnets, nursery rhymes and free-verse) to guide our designs, and we evaluated the three designs by consulting humanities scholars.

2 Related Work

There is a large collection of previous work on text and document visualization. The dominant techniques in this area have been statistics graphics (e.g., [VCPK09, AC07]), network visualization (e.g., [Mil95, WV08, vHWV09, Pal02, CCP09]),) and pixel-based visualization (e.g., [KO07, OBK*08]).

There have been several pieces of existing work on poetry visualization. For example, in [CGM*12] and [CAT*12], pixel-based visualization was used to display the poetic forms and variables, such as meter and rhyming patterns. In [MFM13], radial-based visualization was used to guide poem composition in a collaborative visualization. In [ARLC*13], a web-based interactive system was developed allowing scholars to display some 26 variables using a combination of visual channels (including glyph, network, color and texture). In terms of visualizing phonemic vari-

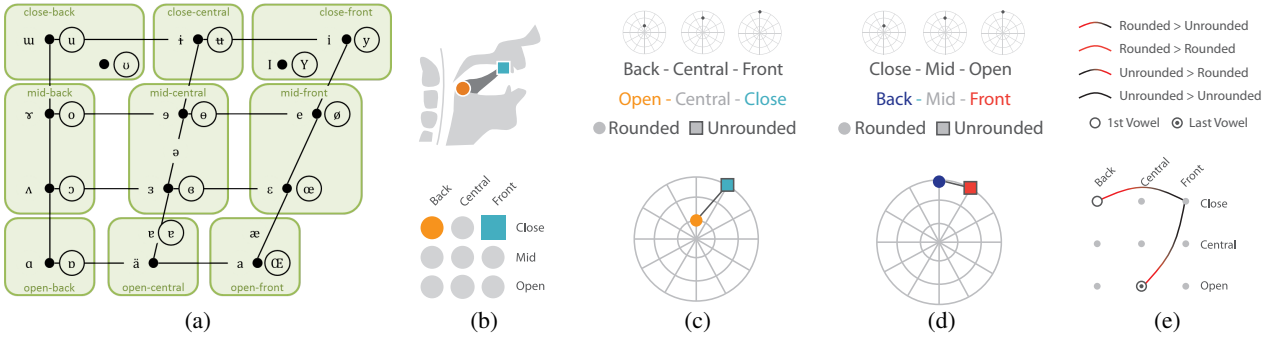


Figure 1: *(a) The vowel chart (from left to right)* back, central *and* front *(which part of the tongue is raised) and (from top to bottom)* close, mid *and* open *(how far the tongue is raised). (b) A representation of a vowel position transition from a close back rounded to a close front unrounded. (c)-(e) The mapping of a vowel transition to the macro-glyph designs.*

ables, apart from the mini-glyph used in [ARLC*13], line graph, density plot, and animated trajectories were used to display two phonemic variables of vowels for visualizing sung vowels in music [FAW11]. As discussed early, mini-glyphs are ineffective in conveying phonetic dynamics at the level of poetic lines. Meanwhile, with line graphs, each poetic line would be displayed as a few time-series corresponding to different variables. Comparing different poetic lines would mean comparing different sets of multivariate time series. Density plots were effective for observing statistical pattern of a large number of vowels, for instance, in a poem or a piece of reasonably long text. Hence they are not suitable for visualizing phonetic dynamics at the level of poetic lines. In [FAW11], animated trajectories, which were referred to "vowel worm", were considered to be promising. We adapted this approach in one of our three designs.

The existing literature on visual encoding has offered us ample guidance in narrowing down our design options. Due to the constraints of the space, here we highlighted those which have brought about most significant influence. Bertin [Ber83] examined a set of basic glyph designs from a cartographer's point of view. Ware [War12] discussed many perception considerations in visualization designs. Ward [War08] provided a technical framework for glyph-based visualization, covering aspects of visual mapping and layout methods, as well as addressing important issues such as bias in mapping and interpretation. Much of existing work in the area of glyph-based visualization can be found in the recent survey by Borgo *et al.* [BKC*13]. Radial-based projection has been used to depict temporal data [DBS*11, WAM01, DH02] and the visualization of human movements [ZFAQ13]. Curved flow symbols [WSD11] have been used to visualize traffic movements. Time-series visualization, such as ThemeRiver [HWN02], has been deployed to depict temporal patters in a large collections of messages or documents. Maguire *et al.* used statistics in the source data to guide the visual design [MRSS*12, MRSS*13]. Several authors, e.g., [RFF*08, Fis10], have discussed the uses of animation in visualization.

3 Visualization & Design Approach

In this paper, we consider only poems in English, and use the International Phonetic Alphabet (IPA) [Int99] to represent the sound and pronunciation of words in poems. We focus on vowels sounds, though the proposed designs can easily be adapted for consonants. Figure 1(a) illustrates three phonemic features of different vowels annotated using the IPA. The x-direction encodes the variable *frontness* or *backness*. The vowels displayed along the line on the left are the *back vowels*, in the middle the *central vowels* and on the right the *front vowels*. The positions correspond to the back, central, and front parts of the tongue respectively. Different vowels were produced when different parts of the tongue move up or down [Hou98, LD12]. The y-direction encodes the variable *vowel height* that is defined by both the mandible and the tongue. The vowels displayed along the top line are the *close vowels*, in the middle are the *close-mid vowels* and *open-mid vowels*, and at the bottom the *open vowels*. A vowel is said to be closed if the tongue is raised high and the mandibular is closed. It is said to be open if the tongue is rested at the floor of the mouth, and mandibular is open [Hou98, LD12]. The third variable is *lip rounding*, which refers to the shape of the lips during the production of a vowel [Hou98, LD12]. In Figure 1(a), all rounded vowels are shown in a circle.

Consider a line in a poem. Let L denote a multivariate time series, which is an ordered set of phonemic features of all vowels in the line, i.e., $L = \{(f_i, h_i, r_i) \,|\, i = 1, 2, \ldots\}$, where f_i defines the frontness (front, central, back), h_i defines vowel height (open, mid, close), and r_i defines lip rounding (rounded, unrounded). The common goal of the three designs to be described in the following sections is to encode L using a macro-glyph.

Static Radial Glyph. In this design, we use clockwise angular positions to depict temporal ordering of vowels in a poetic line. In addition, we utilized three other visual channels, namely radial position, color and shape. Our first design decision is how many radial lines in each glyph. With a circle, we may consider 4, 8, 12, 24, 30, and 60. Using our collection of poems, we compiled the statistics about the

number of vowel phonemes per line. Among the 465 lines in the collection, 84.3% contain ≤ 12 vowel phonemes, while the maximum number of vowel phonemes is 22. Based on the statistics, we decided that 12 radial lines per circle is the most suitable option, because (i) it is consistent with the clock metaphor, (ii) it offers an appropriate density of radial lines, and (iii) most poetic lines need 84.3% of lines need only 1 macro-glyph and 15.7% need 2. If 8 were chosen, 47.1% lines need 2 macro-glyphs and 2.4% need 3. If 24 were chosen, the radial lines would be too densely packed, while 84.3% would use only half of the circle.

Our second design decision is to decide the pairing of phonemic variables and visual channels. A few poetry scholars advised us during the design stage that vowel height and frontness may be more interesting than lip rounding. To help us decide the mapping, we calculated the average difference and standard deviation for vowel height (avg diff: 0.8704, std dev: 0.6699) and frontness (avg diff: 0.7709, std dev: 0.7073). According to the average difference, vowel height may have a slightly higher priority, while according to the standard deviation, frontness may be more favourable. As the statistics is inconclusive, we created two variations of the static radial glyph as illustrated in Figure 1(c,d). As rounding is considered least important by the poetry scholars, we assigned the color channel to frontness in (c) and vowel height in (d), while mapping lip rounding to the shape channel in both variation. We created a logological link between phonemic features and colors to help memorization, i.e., O for Open/Orange, C for Close/Cyan, F for Front/Fire and B for Back/Blue. Naturally, we used circle for rounded and square for unrounded vowels.

Animated Transitions. We explored two variations of animated transitions: turbulence and beat. We used the abstract representation of 3×3 positions in Figure 1(a) as the 9 main reference points. When a vowel is moved from one position to another, a transition line is drawn. For the turbulence variation, we trace out the movements using trailing circles as it moves across the line (similar to [FAW11]). We used two different colors to encode lip rounding, i.e., black (unrounded vowels) and red (rounded vowels). The color and gradient of a trace line encodes the relationship between the two consecutive vowels, e.g., a gradient line from red to black indicates the change from a rounded vowel to an unrounded vowel. For the beat variation, the transition lines are represented by a worm-like shape. In addition, we used the line opacity to encode the frequency of a transition, i.e., the more transitions take place between two positions, the darker the residual line becomes.

The advantage of this design is that only one macro-glyph is needed for any line. We anticipated some disadvantages of using animation. For example, it is hard to detect if there is no change in vowel positions. Nevertheless, as some scholars were enthusiastic about animation, we decided that it is better to include both variations in our evaluation.

Static Transitions with Temporal Highlight. This design uses the same 3×3 layout as the Animated Transitions, but place more emphasis on the transition lines and less on animation. All transition lines are static while a small marker moves along the transition lines to convey the temporal ordering. As shown in Figure 1(e), three different circles indicate the first, last and intermediate vowel positions. A self-loop is used when there is a repetition of the same vowel position. Similar to the design of animated transitions, we used color and gradient of the transition lines to indicate four types of transitions. We also created a variation that encodes lip rounding by placing rounded or unrounded markers at the beginning and end of transition lines.

To help us determine the optimal paths to draw transition arcs and self-loops, we computed the statistics about the frequencies of transitions between different vowel positions. In our collection of poems, we found that there are no edges leaving or entering open-central and close-central vowel positions. This is likely because these vowel positions are more commonly used in non-English languages, such as Norwegian *butt* [bʉt] 'blunt' [Int99], or in accented English dialect. Using the statistics, we created a lookup table for all 81 valid transitions, specifying the curve parameters for each transition.

4 Evaluation

Observing sound dynamics in poems requires a fair amount of general knowledge about linguistics and literature, specialized knowledge about poetry and experience of close reading. A controlled user study with arbitrary participants may not reflect the presence of such knowledge and experience. We hence evaluated the three designs of macro-glyphs by consulting four domain experts, including a Professor in Italian Literature (with research interest in educational and poetic theory), and three postgraduate students (in English and French literatures) who studied poetry in the past. The evaluation started with our explanation and live demonstration of the three designs and their variations. This was followed by a multiple-choice questionnaire with 43 questions. Among these, 6 were common questions that were repeated for each of the three designs and their variations, 9 questions were asked individually for specific designs, and 4 general questions were asked at the end, featuring comparison across three designs. The structured survey was followed by free-form discussions. Participants were offered the opportunities to provide alternative answers to the questionnaire after the discussion. A qualitative summary of the experts' answers to the questionnaire, and their comments and suggestions made during the discussions is given below.

Static Radial Glyphs

- It is helpful to the tasks of identifying similar phonetic patterns among poetic lines, and differentiating dynamics and sound structures in poems.
- The two different visual mappings are both important as

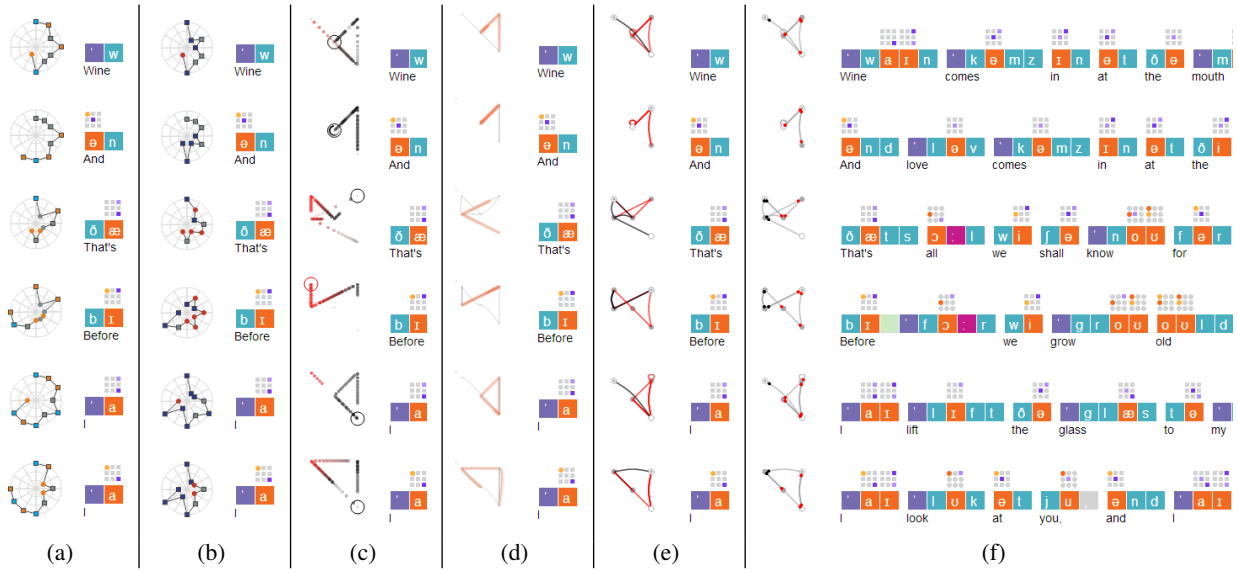


Figure 2: *Visualization of a short poem "A Drinking Song" by* W. B. Yeats *with macro-glyphs. (a) Static radial macro-glyph with a frontness layout while (b) static radial macro-glyph with a vowel height layout. (c)-(d) Animated transitions macro-glyphs with its variations (e)-(f) Static transitions with temporal highlight macro-glyphs with its variations.*

different types of poems may exploit different part of the vowel system in a language. Having both variations enables scholars to explore different hypotheses.
- Making a logological link between phonemic features and colors helps in remembering the color coding.
- The differentiation between circles and squares are sufficient for scholars to differentiate rounded and unrounded vowels.

Animated Transitions

- It is fun to watch.
- It is intuitive for observing the transitions in the flow of sound, but after a while, one tends to lose the tracking of the temporal ordering.
- It is not easy to follow the movements. The residual line patterns do not convey temporal order or direction.
- Using the line opacities to encode the frequencies of the movement is helpful but can be ambiguous.

Static Transitions with Temporal Highlight

- It is helpful to the tasks of observing some kind of similar phonetic patterns among poetic lines, such as the frequencies of certain phonemic features and transitions.
- It is intuitive for observing transitions in the flow of sound, but after a while, one tends to lose the tracking of the temporal ordering.
- The residual line patterns convey adequate directional information and frequency, but do not show temporal ordering.
- The moving highlight is helpful but not sufficient for remembering the temporal ordering.

In general, macro-glyphs with animation are fun to work with. Nevertheless, for the task of analyzing the phonetic dynamics in poems, macro-glyphs that convey most information statically are more useful. The domain experts considered that no single phonemic feature is important by itself. An individual feature may become interesting and significant in relation to other phonemic features, or other aspects of poems. The macro-glyphs allow scholars to observe and make connections between these features.

5 Conclusions and Future Work

This design study has shown that it is feasible to use macro-glyphs to encode a small amount of temporal information while depicting multivariate features. In visual analysis of phonetic dynamics of poems, it is important for a visual representation to support the observation and external memorization of temporal ordering, and directions of the movement. The domain experts prefer to have different visual representations for observing different types of dynamics (e.g., emphasizing vowel height or frontness), while having an essential requirement for observing different multivariate features at individual phonemes in a temporally connected manner. The study also confirms the disadvantages of animation at least from the perspective of analyzing phonetic dynamics in poems, while echoing the previous findings in [RFF*08, Fis10]. Our future work will focus on the visualization of temporal patterns of poetic lines while preserving the spatial context of phonetic features.

References

[AC07] ABBASI A., CHEN H.: Categorization and analysis of text in computer mediated communication archives using visual-

ization. In *Proc. 7th ACM/IEEE-CS Joint Conf. Digital Libraries* (2007), pp. 11–18. 1

[ARLC*13] ABDUL-RAHMAN A., LEIN J., COLES K., MAGUIRE E., MEYER M., WYNNE M., JOHNSON C., TREFETHEN A., CHEN M.: Rule-based visual mappings – with a case study on poetry visualization. *Computer Graphics Forum 32*, 3 (2013), 381–390. 1, 2

[Ber83] BERTIN J.: *Semiology of Graphics: Diagrams, Networks, Maps*. University of Wisconsin Press, 1983. 2

[BKC*13] BORGO R., KEHRER J., CHUNG D. H., MAGUIRE E., LARAMEE R. S., HAUSER H., WARD M., CHEN M.: Glyph-based visualization: Foundations, design guidelines, techniques and applications. In *Eurographics State of the Art Reports* (May 2013), EG STARs, pp. 39–63. 2

[CAT*12] CLEMENT T., AUVIL L., TCHENG D., CAPITANU B., MONROE M., GOEL A.: Sounding for Meaning: Analyzing Aural Patterns Across Large Digital Collections. In *Digital Humanities* (2012). 1

[CCP09] COLLINS C., CARPENDALE M. S. T., PENN G.: Docuburst: Visualizing document content using language structure. *Computer Graphic Forum 28*, 3 (2009), 1039–1046. 1

[CGM*12] CHATURVEDI M., GANNOD G., MANDELL L., ARMSTRONG H., HODGSON E.: Myopia: A Visualization Tool in Support of Close Reading. In *Digital Humanities* (2012). 1

[DBS*11] DROCOURT Y., BORGO R., SCHARRER K., MURRAY T., BEVAN S. I., CHEN M.: Temporal visualization of boundary-based geo-information using radial projection. *Computer Graphics Forum 30*, 3 (2011), 981–990. 2

[DH02] DRAGICEVIC P., HUOT S.: SpiraClock: A Continuous and Non-Intrusive Display for Upcoming Events. In *Extended Abstracts of CHI* (2002), pp. 604–605. 2

[FAW11] FROSTEL H., ARZT A., WIDMER G.: The vowel worm: Real-time mapping and visualisation of sung vowels in music. In *Proc. 8th Sound & Music Computing* (Jul 2011), pp. 214 – 219. 2, 3

[Fis10] FISHER D.: *Animation for Visualization: Opportunities and Drawbacks*. Beautiful Visualization. O'Reilly Media, 2010, ch. 19, pp. 329 – 352. 2, 4

[Hou98] HOUSE L. I.: *Introductory Phonetics and Phonology: A Workbook Approach*. Psychology Press, 1998. 2

[HWN02] HAVRE S., WHITNEY P., NOWELL L.: ThemeRiver: Visualizing thematic changes in large document collections. *IEEE Trans. Visualization & Comp. Graphics 8* (2002), 9–20. 2

[Int99] INTERNATIONAL PHONETIC ASSOCIATION: *Handbook of the International Phonetic Association*. Cambridge University Press, 1999. 2, 3

[KO07] KEIM D. A., OELKE D.: Literature fingerprinting: A new method for visual literary analysis. In *IEEE VAST* (2007), pp. 115–122. 1

[LD12] LADEFOGED P., DISNER S. F.: *Vowels and Consonants*, 3rd ed. Wiley-Blackwell, 2012. 2

[MFM13] MENESES L., FURUTA R., MANDELL L.: Ambiances: A Framework to Write and Visualize Poetry. In *Digital Humanities* (2013). 1

[Mil95] MILLER G. A.: WordNet: A lexical database for English. *Communications of the ACM 38*, 11 (Nov. 1995), 39–41. 1

[MRSS*12] MAGUIRE E., ROCCA-SERRA P., SANSONE S.-A., DAVIES J., CHEN M.: Taxonomy-Based Glyph Design - with a Case Study on Visualizing Workflows of Biological Experiments. *IEEE Trans. Visualization & Comp. Graphics 18*, 12 (2012), 2603 – 2612. 2

[MRSS*13] MAGUIRE E., ROCCA-SERRA P., SANSONE S.-A., DAVIES J., CHEN M.: Visual Compression of Workflow Visualizations with Automated Detection of Macro Motifs. *IEEE Trans. Visualization & Comp. Graphics 19*, 12 (2013), 2576–2585. 2

[OBK*08] OELKE D., BAK P., KEIM D., LAST M., DANON G.: Visual evaluation of text features for document summarization and analysis. In *IEEE VAST* (Oct. 2008), pp. 75 –82. 1

[Pal02] PALEY W. B.: TextArc, 2002. URL: http://www.textarc.org/. 1

[RFF*08] ROBERTSON G., FERNANDEZ R., FISHER D., LEE B., STASKO J.: Effectiveness of animation in trend visualization. *IEEE Trans. Visualization & Comp. Graphics 14*, 6 (Nov 2008), 1325–1332. 2, 4

[VCPK09] VUILLEMOT R., CLEMENT T., PLAISANT C., KUMAR A.: What's being said near "Martha"? Exploring name entities in literary text collections. In *IEEE VAST* (2009), pp. 107–114. 1

[vHWV09] VAN HAM F., WATTENBERG M., VIEGAS F. B.: Mapping text with phrase nets. *IEEE Trans. Visualization & Comp. Graphics 15*, 6 (Nov. 2009), 1169–1176. 1

[WAM01] WEBER M., ALEXA M., MULLER W.: Visualizing time-series on spirals. In *IEEE Symp. Information Visualization* (2001), pp. 7–13. 2

[War08] WARD M. O.: Multivariate data glyphs: Principles and practice. In *Handbook of Data Visualization*. Springer, 2008, pp. 179 – 198. 2

[War12] WARE C.: *Information Visualization: Perception for Design*, 3rd ed. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2012. 2

[WSD11] WOOD J., SLINGSBY A., DYKES J.: Visualizing the dynamics of london's bicycle-hire scheme. *Cartographica 46*, 4 (2011), 239–251. 2

[WV08] WATTENBERG M., VIÉGAS F. B.: The Word Tree, an interactive visual concordance. *IEEE Trans. Visualization & Comp. Graphics 14*, 6 (Nov. 2008), 1221–1228. 1

[ZFAQ13] ZENG W., FU C.-W., ARISONA S. M., QU H.: Visualizing interchange patterns in massive movement data. *Computer Graphics Forum 32* (2013), 271–280. 2

H Copy of [LMW+14]:

Fail-safe Glyph Encoding based on Quasi-Hamming Distances: With a Case Study on Visualizing File System Events

Fail-safe Glyph Encoding based on Quasi-Hamming Distances: With a Case Study on Visualizing File System Events

Philip A. Legg, Eamonn Maguire, Simon Walton, Michael Goldsmith,
Sadie Creese, and Min Chen, *Member, IEEE*

Abstract—In many applications of spatial or temporal visualization, glyphs provide an effective means for encoding multivariate data objects. However, because glyphs are typically small, they are vulnerable to various perceptual errors. In information theory and communication, the concept of *Hamming distance* underpins the study of codes that support error detection and correction by the receiver without the need for corroboration from the sender. In this work, we propose a novel concept of *quasi-Hamming distance* in the context of glyph design. We examine the feasibility of estimating quasi-Hamming distance between a pair of glyphs, and the minimal Hamming distance for a glyph set. This measurement enables glyph designers to determine the differentiability between glyphs, facilitating design optimization by maximizing distances between glyphs under various constraints (e.g., the available number of visual channels and their encoding bandwidth). We demonstrate this concept through a case study on visualizing file system events in Dropbox and Git. Our evaluation shows that the concept of quasi-Hamming distance allows us to design fail-safe glyphs, significantly reducing the vulnerability of glyph-based visualization by empowering users to detect and correct communication error.

1 Introduction

Glyph-based visualization [40, 2] is a common form of visual design where some data records are depicted by pre-defined visual objects, which are called *glyphs*. Glyph-based visualizations are ubiquitous in modern life since they make excellent use of the human ability to learn abstract and metaphoric representations in order to facilitate instantaneous recognition and understanding. Glyphs can be used to encode variables of different data types, categorical (e.g., [24, 29]) as well as numerical (e.g., [22, 10]). However, glyphs are typically small, and are often designed with a high-degree of similarity in order to facilitate mapping consistency, semantic interpretation, learning and memorization. In many applications of spatial or temporal visualization, there are quite often a large number of small glyphs required. Hence we are particularly concerned about the *differentiability* of glyphs and potential perceptual errors in observation and exploration.

Fig. 1 shows example cases that may render some glyphs indistinguishable. Zooming-out actions in data exploration can reduce glyph size significant. For example, they could make some shapes (e.g., circle and hexagon) and textures appear similarly, while confusing the categorization of sizes (e.g., big, medium, small). Meanwhile, environmental lighting conditions and printing or photocopying facilities can cause color and greyscale degeneration. Not only would such changes make some glyphs indistinguishable, but would also confuse the association between different colors or grayscales. Whilst a dynamic legend may help alleviate the confusion about various mappings, it demands users to view the legend on a regular basis, incurring additional cognitive load in terms of the effort for the bothersome visual search and memorization of the unstable mapping keys. Other issues could also include color- or change-blindness, short- or long-sightedness, clustering, occlusion, distortion, and so on.

In the visualization literature, there are many useful guidelines that could be adopted for effective visualization [2]. Bertin [1] advised that size is not associative, hence unsuitable for encoding categorical attributes. Tools such as ColorBrewer [17] can be used to generate qualitative colormaps for effective separability of attributes. Many glyph designers apply their creative intuition to ensure the diversity and legibility of different glyphs. This poses some challenging research questions for effective glyph design, including, '*Is there a theoretical framework to encompass various design guidelines?*' and '*Is there a systematic approach to design a fail-safe glyph set?*'

In this work we propose a conceptual framework for glyph-design based on the Hamming distance (Section 3). Because of the perceptual nature of many design aspects, we introduce the notion of *quasi-Hamming distance* (QHD) (Section 4). Using this notion, we are able to translate qualitative assessment of perceptual distances in a design to Hamming distances. When the minimal Hamming distance for a glyph set is 1, the glyph set is vulnerable to the 'noise' during observation and exploration. When the minimal Hamming distance is 2, the glyph set facilitates some error detection, with which the viewer can use interaction (e.g., zooming-in, or looking at the legend) to investigate the error. When the minimal Hamming distance is 3 or more, the glyph set facilitates some error correction at the receiving end. This enables us to adjust the design to ensure a minimal Hamming distance among a set of glyphs. It is a systematic approach to optimize the design of a set of glyphs, providing fail-safe glyph encoding.

To support this novel concept for glyph-based visualization, this work includes the following additional contributions:

- We outline several methods for estimating QHD, and present two proof-of-concept of experiments for estimating QHD based on the grading by human participants and using image-comparison metrics respectively (Section 4).
- We present a case study on visualizing file system events, where glyphs were designed to facilitate error detection and correction, and were evaluated by human participants and image-comparison metrics (Section 5).
- We demonstrate the uses of the set of fail-safe glyphs to visualize event log data captured from two popular real-world file systems, namely, Dropbox and Git (Section 6). The former provides users around the world with file sharing facilities, while the latter is a distributed version control system that supports collaborative software development.

- *Philip A. Legg is with the Cyber Security Centre, University of Oxford. E-mail: phil.legg@cs.ox.ac.uk.*
- *Eamonn Maguire is with the Oxford e-Research Centre, University of Oxford. E-mail: eamonn.maguire@oerc.ox.ac.uk.*
- *Simon Walton is with the Oxford e-Research Centre, University of Oxford. E-mail: simon.walton@oerc.ox.ac.uk.*
- *Michael Goldsmith is with the Cyber Security Centre, University of Oxford. E-mail: michael.goldsmith@cs.ox.ac.uk.*
- *Sadie Creese is with the Cyber Security Centre, University of Oxford. E-mail: sadie.creese@cs.ox.ac.uk.*
- *Min Chen is with the Oxford e-Research Centre, University of Oxford. E-mail: min.chen@oerc.ox.ac.uk.*

Manuscript received 31 March 2014; accepted 1 August 2014; posted online 13 October 2014; mailed on 4 October 2014.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org.

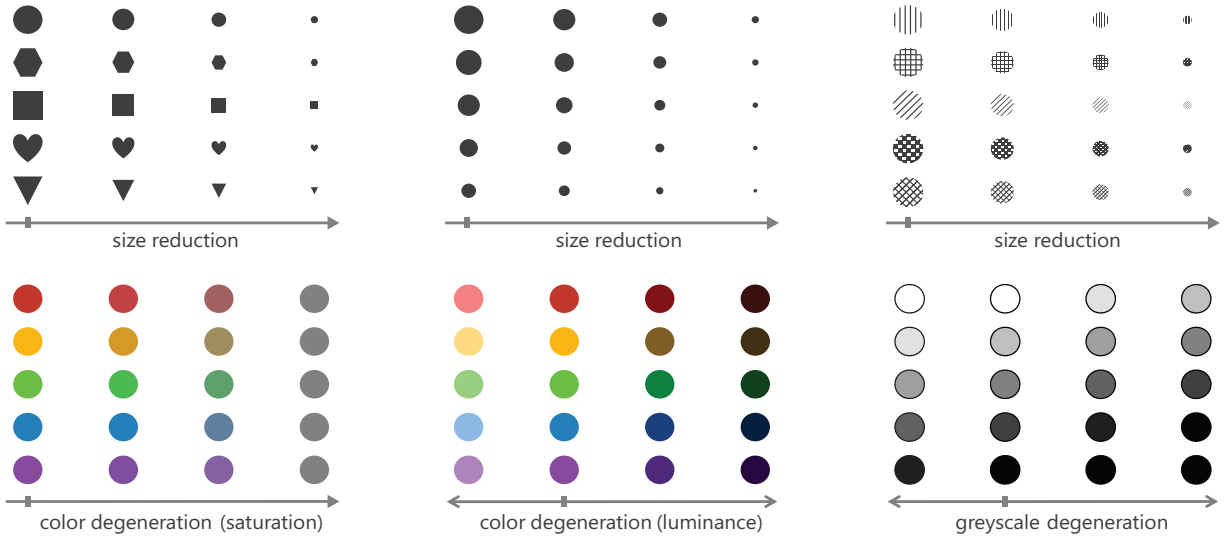


Fig. 1. Three different types of quality generation are applied to several glyphs, each of which is encoded using a single visual channel. The original quality is indicated by a marker on the x-axis. When size, saturation and luminance are changed, they become more difficult to differentiate.

2 Related Works

As the related works, we consider three topics of interest: representing hierarchical and temporal information, glyph-based and event-based visualization, and visualization separability and cognition.

Originally proposed by Johnson and Shneiderman [20], Treemaps have proven a popular technique for visualizing hierarchical data, such as file systems, using a nested rectangle approach. In the literature, there is much work that aims to extend upon this by increasing the capacity of data that can be visualized, and the addition of temporal information. Tu and Shen propose using contrast Treemaps to depict change between two or more snapshots of hierarchical data [38]. Card *et al.* introduce *TimeTree*, which allows a user to interactively browse temporal change in hierarchical data [4]. Lamping and Rao propose the use of a hyperbolic browser for visualising large hierarchical data that incorporates a fish-eye lens to provide focus and context [23]. Holten extends this approach to show hierarchical edge bundles that depict relationships between data point [18]. Burch and Diehl introduce *TimeRadarTress*, that use a radial tree layout to depict hierarchy, with associated circle sectors to also show temporal changes [3]. Therón also uses a radial layout, based on a tree-ring metaphor for depicting hierarchical data whilst also incorporating the temporal element [37]. More recently, Guerra-Gómez *et al.* introduced *TreeVersity2* for visualizing temporal changes in dynamic hierarchical data [15].

Ward [40, 41] provides a technical framework for glyph-based visualization that covers aspects of visual mapping and layout methods, as well as addressing important issues such as bias in mapping and interpretation. Borgo *et al.* [2] provide a state of the art report on glyph-based visualization that address many of the design guidelines and techniques that have been utilized in the field. Lie *et al.* [25] discuss a variety of design considerations for glyph-based visualization including data mapping, glyph instantiation, and rendering, for three-dimensional data. Event visualization aims to allow the user to understand not only that a change in time has occurred, but more specifically, to understand the semantic attributes surrounding the change event. Glyph-based visualization is popular for event-based visualization due to the capability of intuitive encoding of the event semantics, and the capacity of multivariate glyph representation. Legg *et al.* propose *MatchPad* for analyzing sports event data using glyph-based visualization [24]. Parry *et al.* also use event-based visualization for mapping temporal events and context in Snooker [30]. Kapler and Wright developed a prototype system *GeoTime* that displays military events in a combined temporal and geo-spatial visualization [21]. Gatalsky *et al.* use the similar concept of the 'space-time cube' to visualize spatio-temporal information relating to earthquake events [14]. Pearlman and Rheingans use glyphs for visualizing network security events [31]. Suntinger *et al.* [35] also use glyph-based event visualization to create an *Event Tunnel* for business analysis and incident exploration. Jänicke *et al.* [19] developed *SoundRiver* that mapped movie audio/video content to glyph visualizations on a timeline. Ware and Plumlee [42] investigate the use of glyph-based visualization for encoding multi-variate weather data such as temperate, pressure, wind direction and wind speed. Duffy *et al.* [10] use glyph-based video visualization to encode 20 different variables in a single glyph design for semen analysis. Wongsuphasawat *et al.* introduce LifeFlow as a scalable interactive overview of temporal event sequences [43]. Ferreira *et al.* visualise spatio-temporal events for assessing New York taxi trips [13]. Luo *et al.* visualise events in automated text analysis from large text collections, using their proposed system *Event River* [28]. Recently, storyboard visualization has become popular for depicting key events in temporal data such as movie content [36, 26].

Chen and Jänicke [6] proposed an information-theoretic approach that compares the process of developing visualizations with the traditional communication model. Chen [5] discusses an information-threotic viewpoint for the development of visual analytics. Duke *et al.* [11] discuss the importance surrounding interpretation, examining how viewers could potentially perceive different meaning from the same visualization. Liu *et al.* [27] distributed cognition as a theoretical frame for visualization. Van Wijk [39] discusses the value of visualization, and how it is a combination of art, science, and the real world. Chung *et al.* [8] discuss eight design principles for sortable glyphs. Particular principles that are important to our work are separability, searchability, attention-balance, and learnability. As part of our interest in separability, there are numerous image comparison metrics that have been proposed in image processing literature (e.g., [33, 12]). Such metrics have also been adopted in visualization, where Zhou *et al.* [44] studied 11 metrics for comparative visualization. Daniel and Chen [9] also studied image comparison metrics for video visualization. Recently, Schmidt *et al.* [34] developed an interactive web application for supporting image comparison.

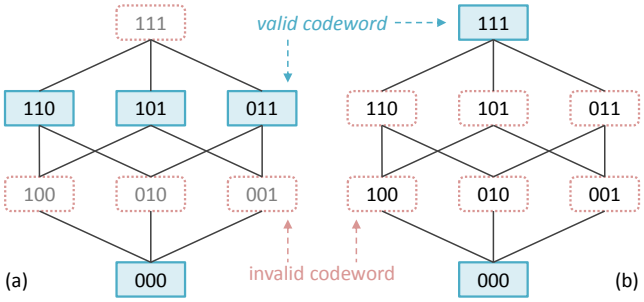


Fig. 2. Two 3-bit codes. (a) A code can detect 1-bit errors. (b) A code can detect 2-bit errors and correct 1-bit errors.

Fig. 3. Two examples that illustrate the phenomena of error detection and error correction in glyph-based visualization. (Above) A viewer may sense that the glyph on the left may not be correct in a distorted visualization, and consult the legend to correct the error. (Below) A viewer may unconsciously perceive the glyph on the left as a star shape due to gestalt effects and a priori knowledge about the glyph set.

3 Hamming Distance

In information theory and data communication, a *code* consists of a finite set of *codewords*, each of which is a digital representation of a letter in an alphabet. In the context of binary encoding, *Hamming distance*, proposed by Richard Hamming in 1950 [16], is a measure of the number of bit positions in which two codewords differ. Considering all pairs of codewords in a code, the minimal distance is referred to as the minimal Hamming distance of the code. (In the literature, the word minimal is often confusingly omitted.) In communication, there are two main strategies for handling errors that occur during transmission. *Automated error detection* allows the receiver to discover that any error has occurred and to request a retransmission accordingly. *Automated error correction* enables the receiver to detect an error and deduce what the intended transmission must have been. Hamming defined the following principle:

Theorem. A code of $d+1$ minimal Hamming distance can be used to detect d bits of errors during transmission. A code of $2d+1$ minimal Hamming distance can be used to correct d bits of errors during transmission [16].

For example, given a 3-bit code as illustrated in Fig. 2, there are 8 possible codewords. One may select a subset of these codewords to construct a code with its minimal Hamming distance equal to 2 bits or 3 bits. Fig. 2(a) shows one of such codes, which has 4 codewords and is of 2 bits Hamming distance. This code can detect 1-bit errors since any change of a valid codeword by 1 bit would result in an invalid codeword, which would lead the receiver to discover the error. Fig. 2(b) shown another code, with 2 codewords and is of 3 bits Hamming distance. It can detect 2-bit errors and correct 1-bit errors. When a valid codeword (e.g., 111) is changed by 1 bit during transmission (e.g., 110), the receiver can detect such an error and recover the intended codeword based on the nearest neighbor principle. Of course, if a 2-bit error occurred during transmission, the receiver would be able to detect the error but could not make a correct 'correction'. Nevertheless, if it is known that 2-bit errors are likely to occur then this should either be used as only an error detection code, or a code with a longer Hamming distance should be used instead.

4 Quasi-Hamming Distance for Glyph Design

A set of glyphs is a code, and each glyph in the set is a valid codeword. During visualization, there can be errors in displaying or perceiving a glyph. If a viewer can detect that a perceived glyph is not quite 'right', conscious or unconscious effort can be made to correct such an error. Conscious effort, which is an analogy of error detection and repeated transmission, may typically include zooming in to have a close look, or consulting the legend. Unconscious effort, which is an analogy of error correction, may include some gestalt effects [7], and inference from other visual information [32]. Fig. 3 shows two example glyph sets, each with 8 codewords. Given the two display errors depicted on the left, i.e., an arrow glyph is skewed in a distorted printout and a shape glyph is occluded by another shape, one can detect both errors easily. The error with the arrow glyph may need some conscious effort, and that with the shape error can usually be corrected unconsciously. This suggests that it is possible to establish a conceptual framework, similar to Hamming distance, for error detection and error correction in glyph-based visualization.

However, understandably, measuring the distances and errors in visual perception is not as simple as measuring those represented by binary codewords. We thereby propose an approximated conceptual framework based on the principle of Hamming distance, and we call it *Quasi-Hamming Distance* (QHD). The term 'quasi' implies that the distance measure is approximated, and the quantitative measure of perceptual errors is also approximated. The main research questions are thereby (i) whether we can establish a measurement unit common to both measures, and (i) how we can obtain such measurements.

The answer to the first research question is that we can utilize 'bit' as the common unit for both distance and error measurement. Let us first consider an ordered visual channel, such as brightness or length, as a code C. Theoretically C can have a set of codewords $\{c_1, c_2, \ldots c_n\}$ such that the difference between two consecutive codewords is the just-noticeable difference (JND) of this visual channel. We can define the QHD between each pair of codewords c_i and c_j as $|i-j|$ bits. During display and visualization, if c_i is mistaken for c_j, we can call this a d-bit error where $d = |i-j|$. Now let us extend this concept to a less ordered visual channel (e.g., hue) or an integrated channel (e.g., color). Theoretically, we can construct a code C by uniformly sampling the space of the visual channel (e.g., the CIE L*a*b* color space) while ensuring that every pair of samples differ by at least the JND of this channel. These codewords, i.e., samples, can be organized into a network, where the distance between any two codewords can be approximated proportionally according to the JND (i.e., JND = 1 bit). Note that the possible perception error rate with a code that maximizes the number of codewords based on JND is likely to be very high. In practice, one designs a glyph set based only on a small subset of samples in a visual channel or more commonly in the multivariate space of several visual channels. Hence a QHD measure based on JND would be too fine to use in practice, though in a longer term, JND can provide an *absolute reference measure* once we have obtained such measures for most visual channels in visualization.

This leads to the second research question, i.e., given a glyph set, how can we measure the distance between glyphs? One may consider using the following methods:

1. **Estimation by expert designers.** This practice has always existed in designing exercises such as for traffic signs and icons in user interfaces. To formalize this practice, designers can explicitly estimate and label the distance between each pair of glyphs in a glyph set. While this approach may be most convenient to the designers, its effectiveness depends very much on the experience of the designers concerned and it is rather easy to overlook certain types of display and perception errors.

2. **Crush tests.** One can simulate different causes of errors, such as those illustrated in Fig. 1, and determine at which level of degeneration glyphs may become indistinguishable. The corresponding level of degeneration can be defined as QHD. While this approach would yield more consistent estimation of QHD, more research would be required to compile a list of different

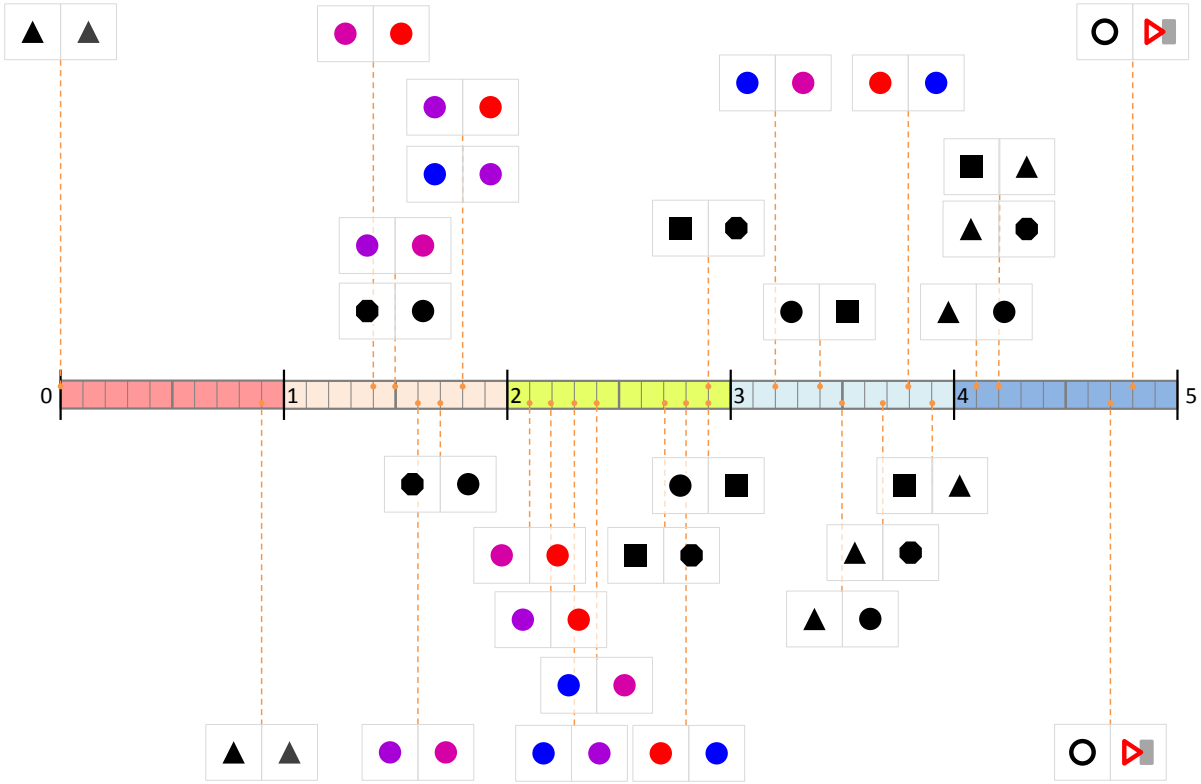


Fig. 4. Sample of results for the primitive glyph pairs used in both the user-centric estimation (top) and by computer-based similarity measure (bottom). Low values indicate difficult to differentiate and high values indicate easy to differentiate.

causes of errors and define coherent levels of degeneration across different causal relationships.

3. **Task-based evaluation.** Similar to (2), one can simulate different visualization conditions, enlist users to perform their tasks, measure users' performance, and transform performance measures to QHD. On the one hand, this approach is perhaps most semantically meaningful for a particular glyph set in a specific application context. On the other hands, the performance measures collected may feature many confounding effects, while there may only be a small number of users available for such an evaluation.
4. **User-centric estimation.** One may conduct a survey among human participants about how easy or difficult to differentiate different glyphs. By removing task-dependency in (3), more participants can be involved in such a survey, yielding more reliable estimation of QHD.
5. **Computer-based similarity measures.** There are a large collection of image similarity measures in the literature [33, 12]. In a longer term, it is likely that we will be able to find measures that are statistically close to user-centric estimation, though there is not yet a conclusive confirmation about optimal image similarity measures, and there are hardly any metrics specially designed for measuring similarity of glyphs.

To demonstrate the feasibility of estimating QHD, we conducted two proof-of-concept experiments based on methods (4) and (5). We conducted a survey among 20 participants, all of whom are either employees or students at University of Oxford. About 50% encountered glyph-based visualization in their course or work. The results of one participant were considered as an outlier and were not included in the statistics. After a brief introduction by one co-author of this paper, the participants were asked to rate how difficult or easy to differentiate 104 pairs of glyphs in an integer scale from 0 to 10. The survey was conducted during an informal lunch, where pizzas were served. No cash payment was involved.

The 104 stimuli pairs were divided into three main categories, 8 *reference pairs*, 48 *primitive pairs*, and 48 *application-specific pairs*. We will discuss the last category in Section 5 in detail. The 96 primitive and application-specific pairs were mixed together in a randomized order. The 8 reference pairs were placed at positions 1, 2, 35, 36, 69, 70, 103 and 104 for helping participants to regularize their scores and for enabling us to check temporal consistency. The details about the questionnaire, the stimuli grouping and the survey results can be found in the supplementary materials. Here we briefly describe the survey results in relation to the reference and primitive pairs.

The category of reference pairs are divided into two groups. Group A consists of 4 pairs of very similar glyphs, and Group B consists of 4 pairs of very different glyphs. We expected that participants will assign very low scores (difficult to differentiate) to those in A and high scores (easy to differentiate) to those in B respectively. As mentioned early, we place one pair from A and one from B at regular intervals. The average scores for the 4 pairs in group A are (0.0, 0.4, 1.4 and 2.8) respectively and those for group B are (9.3, 9.0, 8.9, 9.7) respectively, indicating that they have statistically served as references for the minimal and maximum QHD in this survey.

The category of primitive pairs consists of 8 groups for estimating QHD in relation to 8 visual channels, namely hue, shape, components, connection lines, luminance, size, texture, and orientation. Each group has 6 pairs of stimuli, facilitating pairwise comparison of 4 different codewords of each channel. For hue and luminance channels, after choosing the 1st and 4th codewords we used a perceptually uniform color model (Hunter's Lab) to determine the 2nd and 3rd codewords at 50% and 75% distance from the 1st. The upper part of Fig. 4 shows a small selection of survey results, where we converted the $[0, 10]$ score

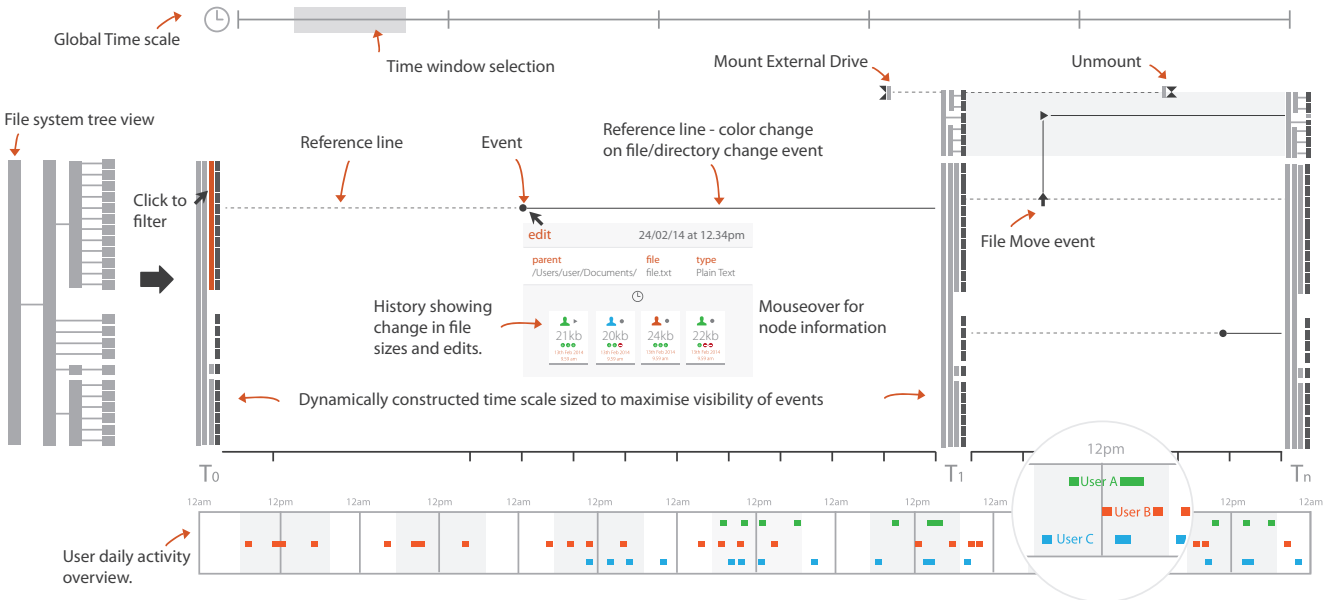


Fig. 5. Visualization overview. We use a timeline approach with a condensed file system hierarchy at the left of the visualization. File system activities are depicted by glyphs on the timeline, with connected lines to show correspondence to the file system. Keyframes are displayed to depict significant changes to the file system hierarchy, or at user-specified time intervals. Interation with mouse cursor displays context box with detailed information on the selected file, directory, or activity.

range to a $[0,5]$ QHD range, and we considered that any QHD < 2 is potentially risky for error detection, and any QHD < 3 is potentially risky for error correction.

In our second experiment, we measured the similarity between each pair of glyphs using a computer-based metric. The metric combines difference of pixel colors and that of spatial occupancy. The former captures a variety of feature differences such as colour, luminance, size, and orientation, etc. and is defined as the mean Euclidean distance between all corresponding pixels in the two images representing the pair of glyphs. The latter captures some location-invariant features such as spatial occupancy and additional components, and is defined as the difference between the numbers of pixels with $\leq 80\%$ luminance. Both difference measures are first normalized to the $[0,1]$ range, and are then scaled to the same QHD range as the survey (with the same min, mean, max), before being combined into a single metric. The lower part of Fig. 4 shows the computed similarity measures for the same selection of stimuli pairs.

5 Case Study: Visualizing File System Events

The problem of visualizing file systems plays a significant role in the short history of computer-assisted visualization. In 1991, Johnson and Shneiderman, who were motived by the need to visualizing the structure of a file system, published their seminal paper on treemaps [20]. Today, not only are file systems much larger and contain many more files, they are also shared by many more users and have many more events. One important aspect of a file system is to support collaborative activities, such as sharing files within multi-partner projects and developing software by a team of programmers. While there are text-based mechanisms for recording events in relation to a file system or a specific folder, the amount of data contained in typical log files can easily escalate to the point where it becomes too overwhelming for anyone to read on a regular basis until perhaps some disastrous events take place. To our knowledge, there is no effective visualization technique for allowing users of such collaborative environments to observe events in a cost-effective manner.

In this case study, we designed and developed a novel glyph-based visualization tool for observing events in a file system. There are several technical challenges. Firstly, the hierarchical nature of the file system needs to be depicted so that the spatial context of where a particular event has occurred can be identified. Secondly, the temporal information about events needs to be conveyed so that the activity ordering can also be observed and reasoned. Thirdly, there are a wide range of activities (e.g., copying a file, modifying a file) that are typically performed, which would need to be distinguishable in a visualization. Finally, the visualization should be able to support collaborative environments by depicting activities from different users.

Fig. 5 presents an overview of the visualization and the design process that was adopted for the layout. The visualization layout consists of a number of different components that we shall discuss. The central area of the visualization is the main activity window where the events performed on the file system are displayed on a timeline using glyph-based visualization. The glyphs that have been designed for this application are discussed in detail in the next section. To the left of the main activity window is the file system tree view, which represents the file system using a traditional tree representation where directories are shown as light gray nodes and files shown as dark gray nodes. Since most directories are likely to contain files, leaf nodes are typically file objects. This tree is shown as a condensed display to the left of the main timeline and serves as a reference to spatial context of the file location. File system events in the main window are positioned in correspondence to the file system hierarchy. Connected lines are displayed to relate the file activities back to the particular file in the hierarchy. For events that involve a change in the file system hierarchy, such as copying or moving files, these connecting lines also indicate the new position of the file. The condensed file hierarchy also serves as a 'keyframe', where the current state of the file system hierarchy is shown, either after a significant change to the hierarchy (e.g., copying files, or mounting an external drive), or at a particular time interval specified by the user. Finally, the file hierarchy can also be used to select the directory of interest that should be shown on the visualization. This provides a mechanism for 'zooming in' to a particular directory, or 'zooming out' to view the root directory. At the top of the display is a time window selection bar, that allows the user to specify the time period that should be shown in the main activity window. At the bottom of the display is a daily activity overview that provides a summary of which users have perform some event and at what time. The interface

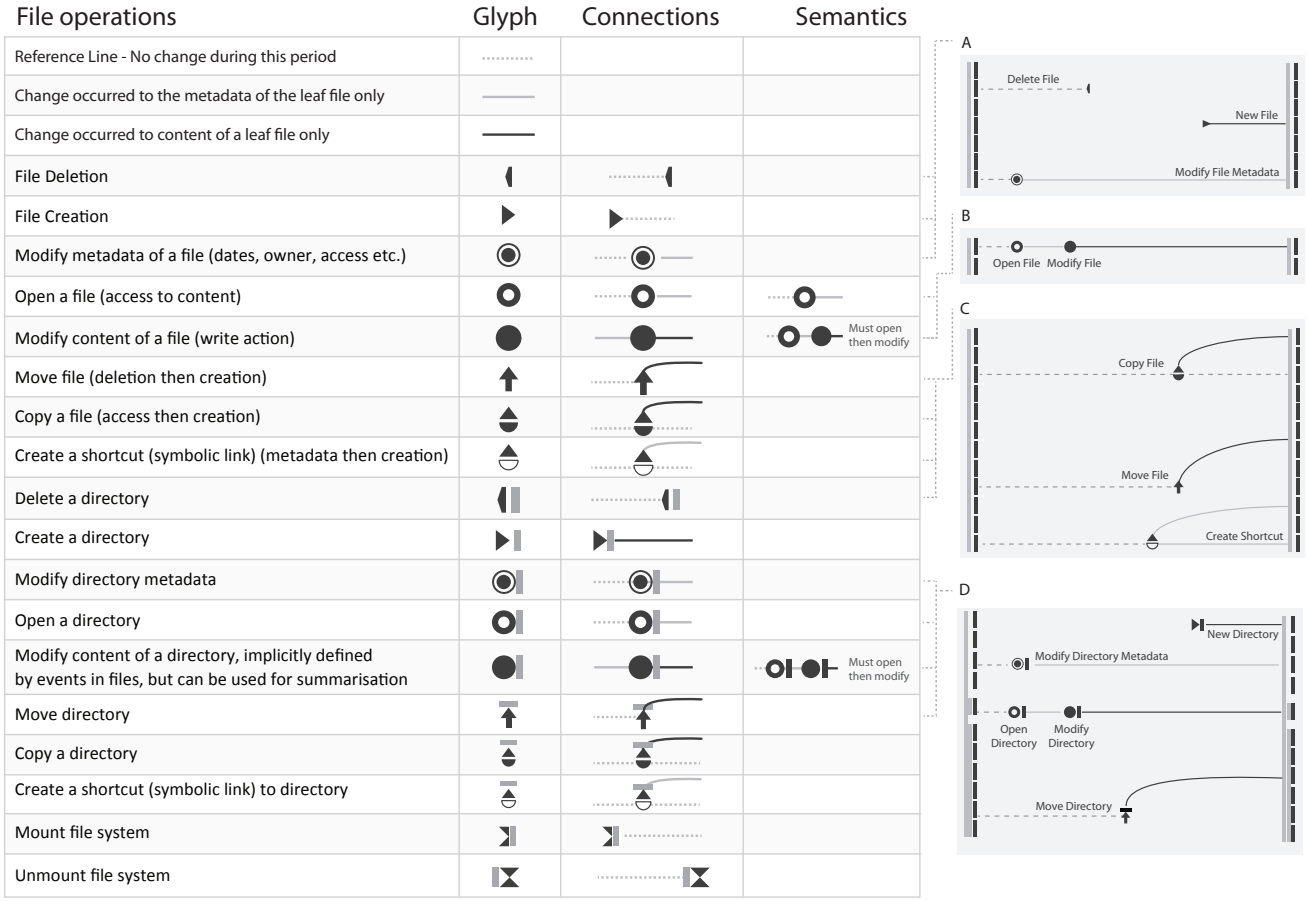


Fig. 6. The 18 glyphs designed to represent different events in file systems. Each event is associated with its primary glyph representation in the second column. In addition, an event may be associated with special signatures in terms of connection (in the third column) and semantic ordering (in the fourth column).

also supports a pop-up window that is displayed as the mouse hovers over a particular event. The window provides more information that is related to both the current event, and the file.

5.0.1 Designing Event Glyphs

The concept of QHD has been considered throughout the design, development, evaluation and application of the aforementioned glyph-based visualization tool. Our understanding and appreciation of the concept have improved along with this process. Fig. 6 shows 18 glyphs for most common events in file systems. These events include creation, modification, deletion, copying, moving and renaming. The action may be applied to a file, a directory, a device, a shortcut (symbolic link), or meta-data. The designs of these event glyphs were evolved in several stages.

Initial Design. We first designed a set of glyphs in conjunction with the overall visual design of the visualization tool as shown in Fig. 5. This allowed us to appreciate how these glyphs may be used, and what are the typical display conditions such as glyph sizes, density, and available visual channels. It was at this stage when we decided that the basic glyph designs should not feature the hue channel, and reserve this intuitive and powerful visual channel to depict user-specific or data-specific variables.

Expert Estimation. Four visualization researchers took part in this research, and all had publications in areas of glyph-based visualization. We used our knowledge about different visual channels and our experience in glyph designs to improve the original designs. This is similar to method (1) in Section 4. We noticed that although we could reach agreement as to how easy or difficult it can be to differentiate pairs of glyphs, we could not easily agree on the reasons why. When we explicitly tried to determine the QHD between a pair of glyphs, we were often influenced by many different features, component shapes, convexity, aspect ratio, curvature, and so on. This experience partly led us to appreciate more the multi-faceted nature and the complexity in estimating QHD. For most of the glyphs in Fig. 6, their designs became stabilized at this stage.

Crush Tests. We applied crush tests to all glyphs designed during the case study. In several cases, we carried out systematic testing by applying consistent zooming factors to all glyphs. More often, when we were considering individual glyphs, we carried out ad hoc crush tests by using facilities in our drawing software, such as zooming, and overlaying a translucent shape on top of glyphs. At this stage, we realized that simulating different conditions that would cause glyph quality to degenerate was not a trivial undertaking. In many ways, this also echoed the multi-faceted nature and the complexity in estimating QHD as mentioned above.

Human-centric Estimation. We conducted a survey, partly to gain a better understanding about QHD in the context of individual visual channels (see Section 4), and partly to evaluate our event glyphs in Fig. 6. We considered 20 different glyph designs, for which there would be 190 pairwise comparisons. We selected 48 pairs that were considered potentially more risky than other pairs. We found that only 1 pair scored below 2 bits in terms of QHD in the survey. The final designs of the glyphs did not include this pair. The details of this evaluation will be given in Section 5.0.2.

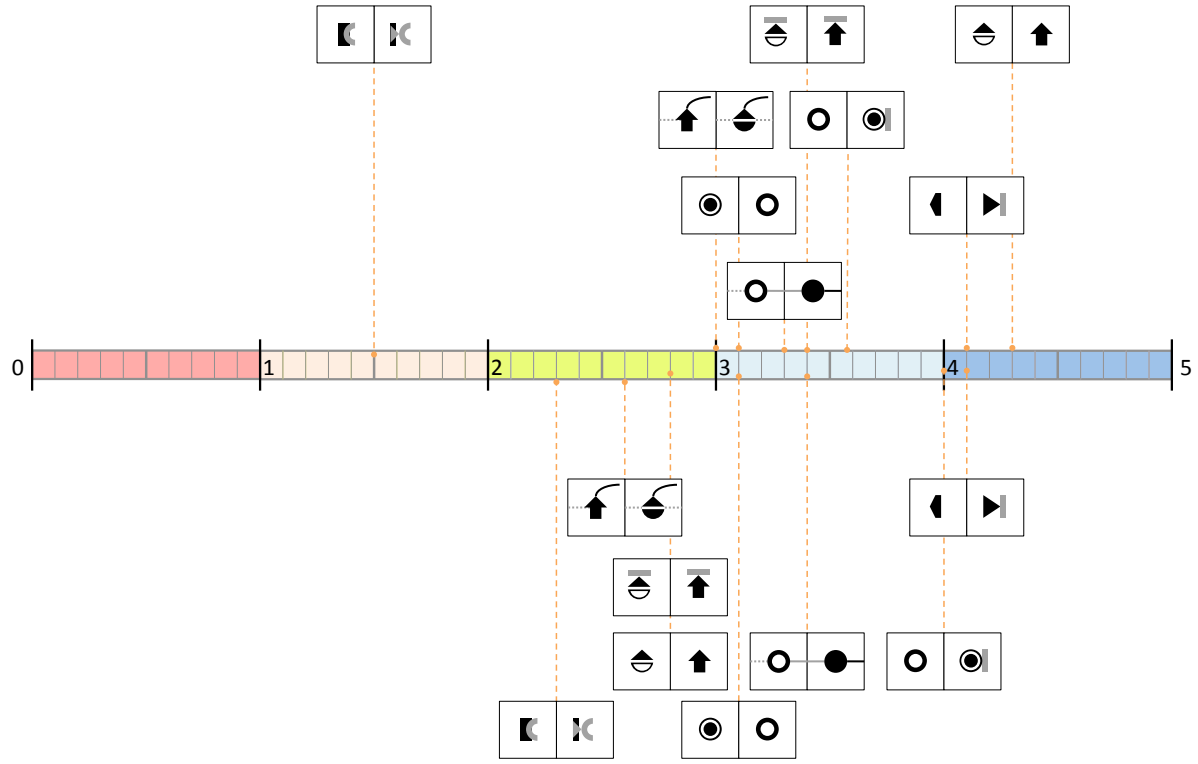


Fig. 7. Sample of results for the file visualization glyph pairs used in both the user-centric estimation (top) and by computer-based similarity measure (bottom). Low values indicate difficult to differentiate and high values indicate easy to differentiate.

Computer-based Similarity Measures. We used the same metric as mentioned in Section 4 to measure the QHD of the 48 pairs that might be potentially risky. We found that they all passed this QHD test. The details of this evaluation will be also given in Section 5.0.2.

Deployment in Software. In addition to the above design effort based on the concept of QHD, we incorporated the glyph set into the visualization tool and used the tool to visualize events in a Dropbox folder and a Git repository. This allowed us to gain direct experience about how these glyphs might be viewed and interpreted in practical applications. The details of this deployment will be discussed in Section 6.

Differentiability is only one aspect of glyph design. We have to consider other aspects such as how easy it is to learn and to remember glyphs, how glyphs may be connected, and how they may be ordered if the corresponding events happened to the same file or directory. As shown in Fig. 6, we utilized some similar designs for files and directories to assist in learning and memorization. Meanwhile, we also consider how they may be connected. The three types of connection lines as shown on the top of the figure and the different orientations as shown in the third column may potentially add additional features for differentiating glyphs. For example, all lines connecting to a *deletion* glyph will always come from left, and all connecting to a *creation* glyph will always extend towards right. All lines connecting to a *copy* or *move* glyph will suggest a spatial shift vertically. In addition, semantic ordering, such as *open* before *read*, may also add addition QHD to the glyph designs as illustrated on the right side of Fig. 6.

5.0.2 Evaluating Event Glyphs

We evaluated those glyphs in Fig. 6 based on the QHD obtained from a human-centric survey and by using computer-based similarity measures. We utilized the same survey and metrics as described in Section 4 because this allowed us to compare the QHD for our glyph set against that for the reference pairs and primitives discussed in Section 4. Note that for our glyphs only the potentially risky pairs were evaluated.

The human-centric estimation provided us with most meaningful insight about the quality of the glyphs. The 104 pairs of glyphs evaluated by participants have an average QHD of 2.9 bits. The average QHD for the reference pairs (Groups A and B) is 2.7 bits. The average for the primitive pairs (Groups C, D, E, F, G, H, I, J) is 2.6 bits. The average for the potentially risky pairs in our glyph set (Groups O, P, Q, R, S, T, U, V, W, X, Y, Z) is 3.2 bits. Almost all of our glyph pairs have their QHD above 2 bits, except one pair (QHD = 1.5 bits) which was not used in the final design. The upper part of Fig. 4 shows a selection of the survey results.

Meanwhile, the computer-based metric also measured our glyph pairs favourably. As mentioned in Section 4, the average QHD estimated by the metric is normalized to have the same min, mean and max as the human-centric estimation. The complete set of 104 glyph pairs have an average QHD of 2.9 bits The average QHD for the reference pairs is 2.6 bits. The average for the primitive pairs is 2.7 bits. The average for the potentially risky pairs in our glyph set is 3.1 bits. The lowest QHD for our potentially risks glyph pairs is 2.1 bits.

The evaluation also showed some interesting phenomena. The additional features added to the directory glyphs have reduced QHD among directory glyphs. For example, when comparing Group O and Group Q, where the glyphs for *creation*, *modify metadata* and *modify content* were compared within the context of files and directories respectively, human-centric estimation shows a noticeable difference. The average QHD for Group O (files) is 3.4 bits and that for Group Q (directories) is 2.6 bits. Similarly, for Group T and Group V where *move*, *copy* and *short cut* glyphs were compared, the average QHD for Group T (files) is 3.3 bits, and that for Group V (directories) is 2.8 bits. Meanwhile, the computer-based similarity measures suggest little difference between O and Q and between T and V. This suggests that further research is necessary to enrich the existing findings in percep-

tion about the distance functions for integrated and separable visual channels [29].

6 Example Applications: Dropbox and Git

To demonstrate the applicability of the above glyph encoding scheme, we developed an interactive visualization tool for visualizing event log data captured from two real world systems, namely Dropbox[1] and Git[2], both of which are popular for collaborative file usage.

Our glyph-based visualization software is comprised of two parts: (a) a back end for processing commit logs of Dropbox and Git, and (b) a front-end serving as a web-based user interface as well as a software library. The back end was written in Python. It generates two types of JSON (JavaScript Object Notation) files for the front-end, one describing the directory structure and the other describing the events that occurred. The front-end was created with a combination of HTML5, CSS and JavaScript (utilising Raphaël.js[3] for the visualization element and jQuery for control of popup events). The glyphs are stored in a font file created using the IcoMoon[4] service.

In addition to glyph-based visualization, the system supports a variety of interactions including:

- Filtering different types of file system events;
- Filtering different users;
- Selecting a specific directory as a subtree;
- Selecting different time period;
- Zooming, and scrolling along both the directory axis and the time line;
- Showing a detailed pop up window when the mouse hovers over an event glyph.

6.1 Visualizing Dropbox Activity Log

Dropbox is a popular cloud service that allows users to synchronise their files across multiple devices. It allows users to create shared directories that other users can be invited to access. This makes it especially useful for collaborative activities between institutions and colleagues, and for sharing personal media with family and friends. Since the Dropbox service was founded in 2007, its user base has grown to 200 million in 2013. Because of the file sharing capability, it is desirable for users to visualize events in a shared folder, for instance, to see which file has been created or modified recently and by whom. Although the service does provide a text-based activity log that users can access, it is time-consuming, and to some extent, tedious to read a long list of events.

As shown in Fig. 8, glyph-based visualization allows users to gain an overview of the events in a shared folder at ease. As discussed previously, we purposely avoided using colors in our glyph design. This provides the visualization system with the flexibility to encode context-sensitive information, such as different users, or different types of files. In Fig. 8, colors are used to depict different users. We can observe that three different users have who have accessed the system during this time (shown by the blue, orange, and green glyphs).

The three vertical bars are simplified views of the directory tree concerned. It is a fairly large directory with three further levels of subdirectories. The period displayed spans over 39 hours, and a variety of events took place. Some indicate close collaboration, when a line of activities linked different users to a single file. For example, the top line in the left half section shows that the orange user created a file, and then opened and modified it. After several hours, the blue user modified the metadata of the file (possibly renaming it or changing its access date). Several hours later, the green user opened and modified it. If a viewer wishes inspect an event in detail, or simply forget the meaning of a glyph, he/she can simply hovers the mouse over the glyph and a pop-up window will displays details about the event and the file or directory concerned. The detailed information includes the filename, the parent directory, and the file type. It also shows file-specific history, such as the access that each user performed and the file size at each access. With this, the collaborative activities become more transparent. Collaborative colleagues can become aware of ongoing actions, reducing the needs for informing each other of every file access actions using emails.

[1] http://www.dropbox.com
[2] http://git-scm.com
[3] http://raphaeljs.com
[4] http://icomoon.io

6.2 Visualizing Git Repository History

Git is a distributed version control system, allowing programmers to develop software in a collaborative manner. It was first released in 2005, and became popular among programmers after the launch of web-based hosting services such as GitHub. The main file storage is known as the repository, and users can pull from, or push to, the repository from their local version of the Git file storage. Although Git maintains a comprehensive history of file access and revision for a repository, it is time-consuming for Git users to read the history data of a Git repository. Glyph-based visualization can offer an efficient means to gain a quick overview of such history data.

Fig. 9 shows an example visualization of a shared repository between two of the co-authors of this work. As both users in this collaboration had a good understanding of the joint software development project, they only need a quick glance at the visualization to sense of the ongoing programming effort by each other. In comparison with reading the repository history, the visualization provides a much more efficient and effective means for their 'silent communication' and 'autonomous coordination'. It is also easy for any one of the co-developers to know major actions, such as deleting a file. From Fig. 9, one can quickly identify two file deletion actions, one around 17:00 on Monday and one around 17:00 on Tuesday.

7 Conclusions

In this paper, we have presented a novel conceptual framework, called *quasi-Hamming distance* (QHD), which facilitates a systematic approach to designing fail-safe glyph encoding schemes. As the conceptual framework is built on the well-proven theories and practice in communication, it offers the potential to stimulate further research on this topic with a depth and a breadth comparable to the topic of error detection and error correction in communication. To demonstrate the feasibility of this conceptual framework, we presented two proof-of-concept experiments, where we obtained QHD measures from a human-centric survey and from computer-based similarity measures. To demonstrate its practical applicability, we conducted a case study where event logs from Dropbox and Git were visualized using a set of fail-safe glyphs. From the very beginning, the glyph set was designed to ensure a high-level of differentiability, while accommodating the necessary requirements such as minimal uses of colors and metaphoric consistency. The glyphs were evaluated using QHD measures obtained from a human-centric survey and computer-based similarity measures.

We very much consider this as the first step towards the establishment of a collection of mathematical and cognitive theories, experimental findings and statistics, design techniques and computational metrics for guiding and aiding glyph designs. This work highlights a number of gaps, where further research is needed. For example, it is highly desirable for us to understand the relationship between the JND measures of various visual channels and differentiability of glyphs encoded using such visual channels. It is also highly desirable to research into computer-based similarity measures that are statistically closer to human-centric estimation, or even better to collected task performance measures.

References

[1] J. Bertin. *Semiology of Graphics*. University of Wisconsin Press, 1983.

[2] R. Borgo, J. Kehrer, D. H. Chung, E. Maguire, R. S. Laramee, H. Hauser, M. Ward, and M. Chen. Glyph-based visualization: Foundations, design guidelines, techniques and applications. In *Eurographics State of the Art Reports*, EG STARs, pages 39–63, May 2013.

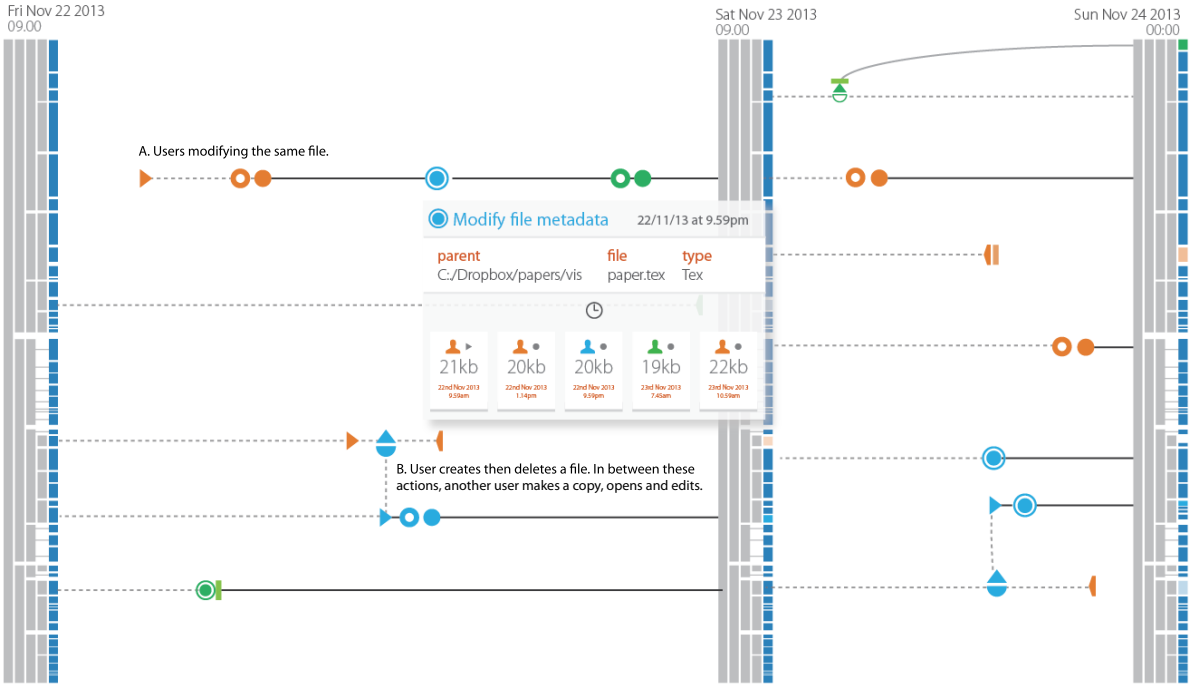


Fig. 8. Glyph-based visualization is used to display events in a Dropbox activity log. The vertical bars are abstract representation a directory tree and the timeline flows from left to right. At (A) we can see the case where a number of users have been modifying the same file. The popup gives more details about the modifications, who did them and when to allow for provenance tracking. At (B) we have another interesting case where *user X* (orange) creates a file, then *user Y* creates a copy of this file, opens it and modifies its contents. *User X* then deletes the original file, however a modified copy exists elsewhere.

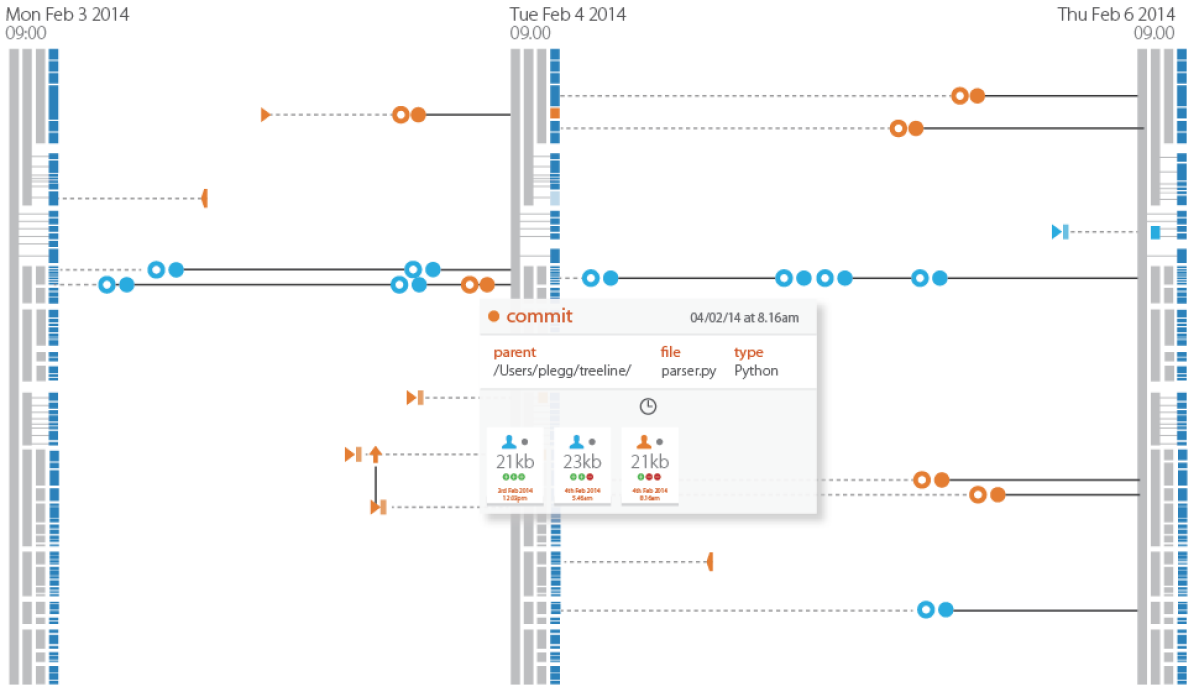


Fig. 9. Glyph-based visualization is used to display events in the history data of a Git repository, where two co-developers were actively working on different parts of the software. They distributed their effort in a coordinated manner, so as to avoid conflicts and problems when merging code. Only on one occasion, *user X* (orange) has modified a file in an area of the repository normally occupied by *user Y* (blue). We can also see that *user X* has been most active when it comes to creating and deleting directories/files compared to *user Y* who has been more active in terms of editing and committing files.

[3] M. Burch and S. Diehl. TimeRadarTrees: Visualizing dynamic compound digraphs. *Computer Graphics Forum*, 27(3):823–830, 2008.

[4] S. Card, B. Suh, B. Pendleton, J. Heer, and J. Bodnar. Time Tree: Exploring time changing hierarchies. In *Proc. IEEE VAST*, pages 3–10, 2006.

[5] C. Chen. An information-theoretic view of visual analytics. *IEEE Computer Graphics and Applications*, 28(1):18–23, 2008.

[6] M. Chen and H. Jaenicke. An information-theoretic framework for visualization. *IEEE Transactions on Visualization and Computer Graphics*, 16(6):1206–1215, 2010.

[7] M. Chen, S. Walton, K. Berger, J. Thiyagalingam, B. Duffy, H. Fang, C. Holloway, and A. E. Trefethen. Visual multiplexing. *Computer Graphics Forum*, 33(3), to appear, 2014.

[8] D. H. Chung, P. A. Legg, M. L. Parry, R. Bown, I. W. Griffiths, R. S. Laramee, and M. Chen. Glyph sorting: Interactive visualization for multi-dimensional data. *Information Visualization*, to appear, 2014.

[9] G. Daniel and M. Chen. Video visualization. In *Proc. IEEE Visualization*, page 54, 2003.

[10] B. Duffy, J. Thiyagalingam, S. Walton, D. J. Smith, A. Trefethen, J. C. Kirkman-Brown, E. A. Gaffney, and M. Chen. Glyph-based video visualization for semen analysis. *IEEE Transactions on Visualization and Computer Graphics*, to appear, 2014.

[11] D. J. Duke, K. W. Brodlie, D. A. Duce, and I. Herman. Do you see what I mean? *IEEE Computer Graphics and Applications*, 25(3):6–9, May 2005.

[12] D. Eler, M. Nakazaki, F. Paulovich, D. Santos, M. Oliveira, J. Neto, and R. Minghim. Multidimensional visualization to support analysis of image collections. In *Proc. SIBGRAPI*, pages 289–296, 2008.

[13] N. Ferreira, J. Poco, H. T. Vo, J. Freire, and C. T. Silva. Visual exploration of big spatio-temporal urban data: A study of New York city taxi trips. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2149–2158, 2013.

[14] P. Gatalsky, N. Andrienko, and G. Andrienko. Interative analysis of event data using space-time cube. In *Proc. IEEE Information Visualisation*, pages 145–152, 2004.

[15] J. Guerra-Gomez, M. L. Pack, C. Plaisant, and B. Shneiderman. Visualizing change over time using dynamic hierarchies: TreeVersity2 and the StemView. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2566–2575, 2013.

[16] R. W. Hamming. Error detecting and error correcting codes. *Bell System Technical Journal*, 29(2):147–160, 1950.

[17] M. Harrower and C. A. Brewer. *ColorBrewer.org: An Online Tool for Selecting Colour Schemes for Maps*, pages 261–268. John Wiley and Sons, Ltd, 2011.

[18] D. Holten. Hierarchical edge bundles: Visualization of adjacency relations in hierarchical data. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):741–748, 2006.

[19] H. Jänicke, R. Borgo, J. S. D. Mason, and M. Chen. SoundRiver: Semantically-rich sound illustration. *Computer Graphics Forum*, 29(2):357–366, 2010.

[20] B. Johnson and B. Shneiderman. Tree-maps: a space-filling approach to the visualization of hierarchical information structures. In *Proc. IEEE Visualization*, pages 284–291, 1991.

[21] T. Kapler and W. Wright. Geotime information visualization. In *Proc. IEEE Information Visualization*, pages 25–32, 2004.

[22] G. Kindlmann and C.-F. Westin. Diffusion tensor visualization with glyph packing. *IEEE Transactions on Visualization and Computer Graphics*, 12(5):1329–1336, 2006.

[23] J. Lamping and R. Rao. The hyperbolic browser: A focus+context technique for visualizing large hierarchies. *Journal of Visual Languages and Computing*, 7(1):33–55, 1996.

[24] P. A. Legg, D. H. S. Chung, M. L. Parry, M. W. Jones, R. Long, I. W. Griffiths, and M. Chen. MatchPad: Interactive glyph-based visualization for real-time sports performance analysis. *Computer Graphics Forum*, 31(3):1255–1264, 2012.

[25] A. E. Lie, J. Kehrer, and H. Hauser. Critical design and realization aspects of glyph-based 3D data visualization. In *Proc. SCCG*, pages 27–34, 2009.

[26] S. Liu, Y. Wu, E. Wei, M. Liu, and Y. Liu. Storyflow: Tracking the evolution of stories. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2566–2575, 2013.

[27] Z. Liu, N. Nersessian, and J. Stasko. Distributed cognition as a theoretical framework for information visualization. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1173–1180, 2008.

[28] D. Luo, J. Yang, M. Krstajic, W. Ribarsky, and D. Keim. EventRiver: Visually exploring text collections with temporal references. *IEEE Transactions on Visualization and Computer Graphics*, 18(1):93–105, 2012.

[29] E. Maguire, P. Rocca-Serra, S.-A. Sansone, J. Davies, and M. Chen. Taxonomy-based glyph design with a case study on visualizing workflows of biological experiments. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2603–2612, 2012.

[30] M. L. Parry, P. A. Legg, D. H. S. Chung, I. W. Griffiths, and M. Chen. Hierarchical event selection for video storyboards with a case study on snooker video visualization. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):1747–1756, 2011.

[31] J. Pearlman and P. Rheingans. Visualizing network security events using compound glyphs from a service-oriented perspective. In *Proc. VizSEC 2007*, Springer Mathematics and Visualization, pages 131–146. 2008.

[32] P. Rheingans and C. Landreth. Perceptual principles for effective visualizations. In *Perceptual Issues in Visualization*, pages 59–74. Springer-Verlag, 1995.

[33] N. Sahasrabudhe, J. West, R. Machiraju, and M. Janus. Structured spatial domain image and data comparison metrics. In *Proc. IEEE Visualization*, pages 97–515, 1999.

[34] J. Schmidt, M. E. Gröller, and S. Bruckner. Vaico: Visual analysis for image comparison. *IEEE Transactions on Visualization and Computer Graphics*, 19(12):2090–2099, Dec. 2013.

[35] M. Suntinger, J. Schiefer, H. Obweger, and M. E. Groller. The event tunnel: Interactive visualization of complex event streams for business process pattern analysis. In *Proc. IEEE Pacific Visualization*, pages 111–118, 2008.

[36] Y. Tanahashi and K.-L. Ma. Design considerations for optimizing storyline visualizations. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2679–2688, 2012.

[37] R. Therón. Hierarchical-temporal data visualization using a tree-ring metaphor. In *Smart Graphics*, Springer LNCS, volume 4073, pages 70–81, 2006.

[38] Y. Tu and H.-W. Shen. Visualizing changes of hierarchical data using treemaps. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1286–1293, 2007.

[39] J. van Wijk. The value of visualization. In *Proc. IEEE Visualization*, pages 79–86, 2005.

[40] M. O. Ward. A taxonomy of glyph placement strategies for multidimensional data visualization. *Information Visualization*, 1(3/4):194–210, 2002.

[41] M. O. Ward. *Multivariate Data Glyphs: Principles and Practice*. Springer Handbooks Comp. Statistics. 2008.

[42] C. Ware and M. D. Plumlee. Designing a better weather display. *Information Visualization*, 12(3-4):221–239, 2013.

[43] K. Wongsuphasawat, J. A. Guerra Gómez, C. Plaisant, T. D. Wang, M. Taieb-Maimon, and B. Shneiderman. LifeFlow: visualizing an overview of event sequences. In *Proc. ACM/SIGCHI CHI*, pages 1747–1756, 2011.

[44] H. Zhou, M. Chen, and M. Webster. Comparative evaluation of visualization and experimental results using image comparison metrics. In *Proc. IEEE Visualization*, pages 315–322, 2002.

I Copy of [MSC14]:

A Visual Analytical Approach for Model Editing and Testing in Glyph-based Time Series Compression

A Visual Analytical Approach for Model Editing and Testing in Glyph-based Time Series Compression

Eamonn Maguire, Student Member, IEEE, Susanna-Assunta Sansone, and Min Chen, Member, IEEE

Abstract— Time series data is ubiquitous in the present world. In many applications, e.g., finance, meteorology, biology, chemistry, physics and engineering, such data is constantly being recorded. It is highly desirable to speed up the process of viewing time series data by exploring new visual representations to complement the conventional line graph that was invented over a millennium ago. One existing methodology is to detect Frequent Long Patterns (FLPs) and replace them with short and abstract notations, such as glyphs. In many applications, the deployment of this methodology has encountered the familiar problem of separating noise from anomalous features. On the one hand, the detection algorithm needs to tolerate a fair amount of noise to facilitate a high compression ratio. On the other hand, the detection algorithm needs to avoid any false positive at all cost. In this paper, we propose a visual analytics approach to address this paradoxical problem. We first use the conventional FLP detection algorithm to create a rough model that places its emphasis on noise tolerance. This initial model is then used to identify a set of FLPs in a corpus of time series, which serve as the testing results of the model. We transform this set of test FLPs to a parameter space, allowing for in detail analysis using parallel coordinates and network plots. Interactive visualization enables users to identify as many false positive results as possible, and more importantly, to identify features that can be used to filter them out. The brushing interaction with the parallel coordinates is then used to refine the initial model by activating feature-based filtering instructions in a model editing window. This approach can be iteratively applied to a model being developed. We have implemented a prototype system to demonstrate this novel approach.

1 Introduction

In many applications, such as finance, meteorology, biology, chemistry, physics and engineering, users have collected a large volume of time series data. A *time series corpus* $\mathscr{T}$ is a collection of time series that record phenomena of a similar nature. On the one hand, users often find that it is time consuming and cognitively demanding to browse many time series in a time series corpus. On the other hand, the corpus provides an opportunity to identify frequently occurring patterns. When such patterns are of a reasonable length, it is advantageous to replace them with a short abstract representation, such as a text label, a signature pattern, a glyph or a combination of these (e.g., [18]). This is a form of *visual compression*, which is commonly seen in real life. For example, in signage management, frequently encountered restrictions and events are encoded as signs and icons, while those of an occasional or exceptional nature are written in text. From an information-theoretic perspective, such a strategy is fundamentally the same as that used in entropy encoding and dictionary encoding [10, 31].

In time series processing and visualization, the algorithm for identifying *Frequent Long Patterns* (FLPs) plays a critical role. It has to be sufficiently tolerant to noise that causes variations to the patterns in the corpus. Without a high-degree of noise tolerance, the criteria of 'frequent' and 'long' could not be met, and the objective of visual compression could not be fulfilled. However, paradoxically the FLP algorithm is also required to exclude any anomalous patterns, which would easily be mistaken as frequently occurring patterns once they are visually compressed. In many situations, an anomalous pattern differs from frequent occurring patterns in a subtle way, and a conventional FLP algorithm may not handle the features that characterize such differences.

Fig. 1(A) shows six segments of a time series. They all appear to be fairly similar, and a typical FLP algorithm (Fig. 1(B)) would consider them to be the same. However, the 4th segment is an anomaly. An ideal visual compression should not replace this segment with a glyph, while compressing as many others as possible, as illustrated in Fig. 1(C).

In this paper, we present a visual analytics approach to address this paradoxical conflict of requirements upon an FLP algorithm. Consider that an FLP identified by an FLP algorithm is a model. The basic idea is to add an additional filtering capability to this model. As the filtering does not take place within the parameter space of the original FLP algorithm, it can handle features that the FLP algorithm cannot. The function of visual analytics is to support the following in an iterative manner:

- discover any anomalous patterns that may have been included by an FLP model;
- compute a bags of features that may be used to characterize different time series segments identified by an FLP model, and to visualize them in the feature space;
- assist users in a number of analytical tasks for identifying the appropriate filters, e.g., results clustering and tagging, and observing the effects of feature selection on filtering results; and
- facilitate model editing by transforming interaction in visualization to text-based instructions in the FLP model.

In the remainder of the paper, we first give a brief overview of time series analysis and visualization in Section 2. In Section 3 we outline a visual analytics loop for analyzing, visualizing, testing, and editing an FLP model, and we describe a prototype system that supports such a visual analytics loop. In Section 4, we describe an FLP algorithm that serves as the core component of an FLP model. In Section 5, we describe a collection of features that have been implemented to support model analysis and editing. In Section 6 we describe the model editing system and introduce the logical operators that can be used to combine models as well as build up rules for parameter refinement. Finally, in Section 7 we present a visualization to visualize the results of the time series compression.

2 Related Work

The related work can be divided in to 'time series analysis', focusing specifically on 'normalization', 'approximation' and 'similarity', and 'time series visualizations'.

- *Eamonn Maguire is with the Oxford e-Research Centre and Department of Computer Science, University of Oxford. E-mail: eamonn.maguire@st-annes.ox.ac.uk.*
- *Susanna-Assunta Sansone is with the Oxford e-Research Centre, University of Oxford. E-mail: sa.sansone@gmail.com.*
- *Min Chen is with the Oxford e-Research Centre, University of Oxford. E-mail: min.chen@oerc.ox.ac.uk.*

Manuscript received 31 March 2014; accepted 1 August 2014; posted online 13 October 2014; mailed on 4 October 2014.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org.

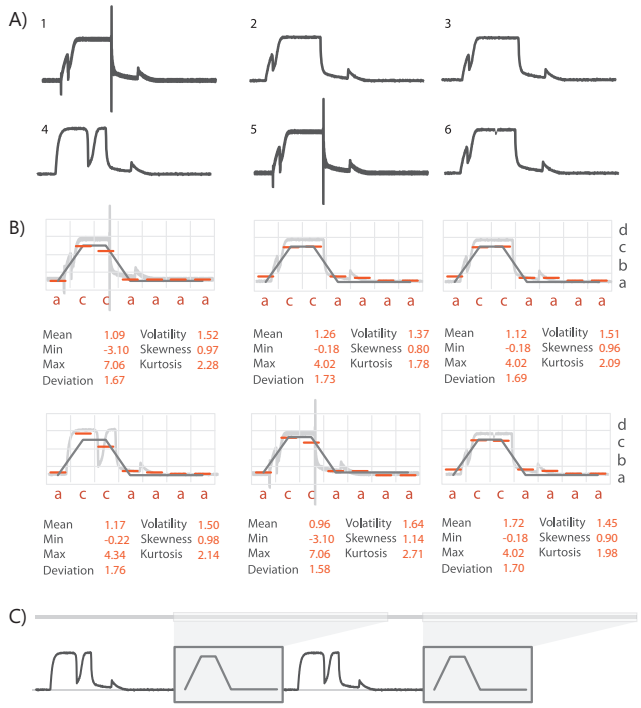


Fig. 1. A) Six time series subsections identified in a time series corpus of space shuttle valve time function taken from [27] B) Symbolic representation of each time results in all motifs being seen as the same, however their more detailed metrics/features differ C) Leaving out the anomalous pattern (A4) from the compression as requested by a user means that the final time series compresses the actual common motifs and leaves the anomalous pattern in plain view.

2.1 Time Series Analysis

Time series analysis encompasses five main sub-categories of operations: indexing, clustering, classification, summarization and anomaly detection [34].

Indexing: This is to provide support to storing, searching and retrieving time series in relation to a data repository.

Clustering: This is to determine how a time series compares with other time series in a database.

Classification: Given a predefined categorization, this is to determine if a new time series belongs to a specific class.

Summarization: Given a large time series dataset, this is to summarize the time series and/or its main attributes using a more compact representation (e.g., a feature vector or a visualization).

Anomaly detection: Given some time series, highlight anomalous incidents based on the profiles of a set of normal events (e.g., [28]).

For all of the above tasks, computing the similarity of between time series data is key. This area of research has a large amount of literature dedicated to it. Here we focus on some of the most relevant and commonly used algorithms that lend themselves to the task of time series comparison, both for whole and subsequence comparison.

Normalization. Before comparing two or more time series, an adjustment may need to be made to the data sets so that the data itself is comparable. For example, if we are looking for commonality in trends, the similarity score should be comparing the overall topology between points, rise and falls rather than being too concerned with where on the y-axis those trends reside. This step is called normalization. One of the most common approaches used for this step is Z-normalization, where the time series are manipulated to have a mean of zero and a standard deviation of one.

Approximation. Following normalization, we may wish to reduce the dimensionality of the data. This means that instead of comparing a time series in terms of real data points, we compare approximations that are smaller in size but largely retain the important features of the series. There are numerous techniques available for this process, all with their own faults and merits. Here we focus on the most relevant and commonly used techniques for approximation of a time series, those being: Discrete Fourier Transformation (DFT) [12] is an algorithm capable of representing the overall 'features' of a time series through spectral decomposition. This means that the signal represented by the time series is decomposed into a series of sine (and/or cosine) waves each represented by a Fourier coefficient [24] - this proves to be a very efficient way of compressing data; Discrete Wavelet Transformation (DWT) [8] is an algorithm similar in principle to DFT, however it has one key difference in that some of the wavelet coefficients represent small local regions of the series meaning that wavelets lend themselves to providing a multi-resolution view of a time series. The results of DWT do not lend themselves to being indexed (for comparison purposes). However an extension called the Haar Wavelet can be calculated efficiently and it's outputs can be indexed for efficient time series matching [7]. The key drawback with DWT is that the length of the time series must be an integral power of two [24]; Piecewise Linear Appromixation (PLA) [6] or Segmented Regression is an algorithm that breaks a time series up in to 'windows', then represents that window with a line segment that best fits the values in that window; Piecewise Aggregate Approximation (PAA) [24] works by splitting a time series up in to 'windows' where each window contains one or more time points. For each window, the average of the time points within is recorded and stored in a vector. The approach is simple yet is shown to rival DFT and DWT [34, 26]; Adaptive Piecewise Constant Approximation (APCA) [25] is algorithmically similar to PAA with extensions that further compress the representation using a technique common to data compression known as run-length encoding (RLE). This extension is the addition of a second number beside the mean value of the window indicating the length of the segment; and Symbolic Aggregate Approximation (SAX) [34, 35] which builds on PAA to take the average values calculated for each window then assigns a letter to that window based on where the mean value falls under a Gaussian curve. The result is a symbolic representation of a time series that lends itself to many computational manipulations such as hashing or storage in data structures such as suffix trees for fast pattern searching. Suffix trees have been used by Lin *et al* [34], to build a motif discovery tool for time series data, capable of highlighting anomalous data. A weakness of the symbolic approach however is the need to define a window and alphabet size ahead of time - these will differ depending on the domain due to differences in periodicity, variance for example.

Similarity. Following these steps, comparison can be performed on the outputs of the approximation which will give an idea of how close a time series is to another. Starting with the most simple of methods, there is Euclidean distance, where for two time series (t_1, t_2) the Euclidean distance would be calculated using $\sqrt{\sum_{k=1}^{n}(t_{1_k} - t_{2_k})}$ where t_{n_k} represents a point in time series n. This metric is fast to compute due to its simplicity, however it carries the caveat that the two sequences must be of the same length. When sequences are not of the same length, there are more complicated algorithms largely based on dynamic programming methodologies such as: Dynamic Time Warping [4] that allow for comparison between sequences of varying lengths and also support shifts in the series; Edit distance with Real Penalty (ERP) allows for gaps in a time series that may be penalised using some configurable value; Longest Common Subsequence (LCSS) [43] introduces a threshold value allowing a degree of mismatch between sequences; and Edit Distance on Real sequences (EDR) [9] merges the concept of gaps and mismatch thresholds presented in ERP and LCSS respectively. Due to the use of dynamic programming techniques, DTW, ERP, LCSS and EDR have limitations in that there are many comparisons made between sequences that have no relevance whatsoever. To address these limitations, there is the Fast Time Series Evaluation [37] algorithm that introduced a more efficient way of building up the comparison matrix meaning that non-related sequences were never compared.

2.2 Time Series Visualization

Time series visualizations, given their use across every possible field have been published about heavily in the past two decades. More complete surveys of time series visualization, can be found in [2, 13]. Here we highlight some of the work in the visualization domain that aims to ease the task of time series analysis and provide a more effective means for users to navigate their data. Such work includes: StackZooming [21] which provides a hierarchical zooming interface for time series data; a spectral visualisation system for visualizing overviews of financial data [23]; the use of 'lenses' to focus on particular areas of a time-series [30, 48]; LiveRAC [36] for computer/information system management; Horizon charts[19] that attempt to aid comparison of many time series in one height fixed display; TimeSearcher [20] and VizTree [33] that allow for exploration of time series via a visual query interface; spiral visualizations to allow for trend discovery in datasets [45, 11]; importance-driven layouts for time series data [16]; multi-resolution techniques for exploration of time-series data [17]; and the visual exploration of frequent patterns in time series data [18].

Glyphs have been used for visualization in time series for a number of tasks. Many such uses have been in order to summarise a series [29, 15, 44, 47]. They have also been used to represent uncertainty in time series data [3].

3 A Visual Analytics Approach

Fig. 2 shows an overall pipeline for detecting FLPs in a time corpus, and for compressing a time series with glyphs. Steps 1, 2, 5 and 6 represent the pipelines traditionally used in the literature. Steps 3 and 4 represent the newly added visual analytics steps for model testing, visualization, analysis and editing.

Similar to the visual analytics system by Legg *et al* [32] for sports video search refinement, this system will allow users to refine FLP models resulting from a conventional FLP algorithm so that users may filter the FLPs most appropriate for their use case.

This pipeline is detailed as follows:

1. a **Time Series Corpus** as input to the system - this is usually a very large collection of time series.
2. a **Frequent Long Pattern (FLP) detection subsystem** - this detects FLPs in the time series, the output of which is a list of different FLP candidates. A user can choose a specific FLP as an individualized FLP model. The model will detect only patterns that match this FLP;
3. a **Visual Analytics Platform** consisting of a number of components to aid model output view, edit and test and refinement operations. The interface that realizes this platform is shown in Fig. 2 A,B,C and D. It is divided in to four linked panels serving a number of complementary functionalities. These are:
   - **A) Model Debugging Panel**: results from the FLP model are presented in this overview area where results of the model can be approved or rejected. The network view presents each motif as a glyph representing the approximation amongst its feature space. These glyphs can be arranged in a number of different topologies:
     (a) by distance between the symbolic representation of each motif - this is calculated on motifs with equal lengths through their euclidean distance. This is defined by the equation below from Lin *et al* [35]

$$MINDIST(S_X, S_Y) = \sqrt{\frac{n}{w}} \sqrt{\sum_{i=1}^{w} (dist)(S_{X_i}, S_{Y_i})^2} \quad (1)$$

     where the *dist* function indicates the use of a lookup table to determine the distance between two letters, say A and D in the symbolic approximation. Since A and D are further away from each other than A and B for each, $dist(A,D)$ would return a larger value than $dist(A,B)$
     (b) by whether motifs have been accepted or not; or
     (c) by a hierarchy representing parent-child relationships where parent motifs (e.g., abbaca) have a sequence that contains N child motifs (e.g., abb, bb, bba, or baca)

     This view also provides users with a way to select a glyph and view more information about the motif it represents. A popup shown in Fig. 3A allows users to mark motifs as accepted or rejected, view their feature space in more detail, obtain their context in the collection of time series being analysed (see Fig. 2 B) and view the original time series.
   - **B) Time Series Overview Panel**: provides a summary view of the time series being used by the motif finding algorithm. This is also used to highlight where motifs occur within the time series to provide further contextual information to users.
   - **C) Model and Parameter Editing Panel**: split in to two sections: the *model* area - this area allows users to create new models, changing the window ω and alphabet size α and combine the results of these models; and the *refinement*, providing an interface for users to edit the feature ranges required for acceptance or rejection of a motif. Users can recalculate at any point and see the results update in Fig. 2 A. See Section 6.
   - **D) Feature Space Panel**: to visualize the feature space of the motifs, we use parallel coordinate plots. Users can brush a combination of axes to find the 'best' parameters that yield the motifs they really want. This is aided by a refinement window shown in 3 B and a link with the network view which filters out nodes that have parameters outside the brushed region(s) (see 2A and D). See Section 5.
4. a **Refinement Loop** which allows re-runs of the FLP detection model(s) given user-defined refinements to the conditions required for motif acceptance;
5. a collection of **Accepted Motifs**; and

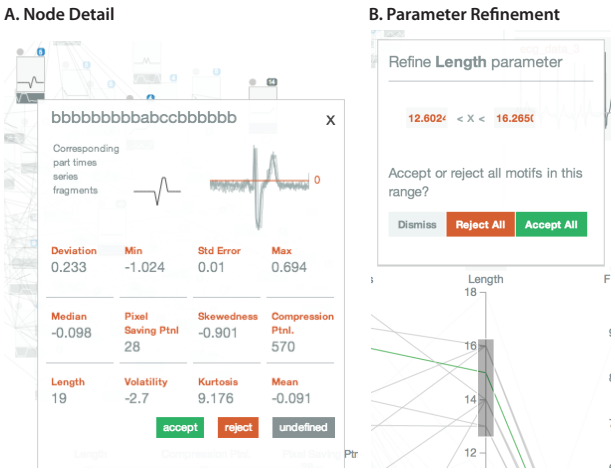


Fig. 3. A) Node detail view showing the feature space of the nodes, the approximation and the corresponding parts of the time series that this motif is associated with. Additionally, the context is shown for the motif in the time series panel on hovering over a node. B) On brushing an axis in the parallel coordinate plot, a parameter refinement window appears to allow the user to either remove results based on particular parameters or to accept them

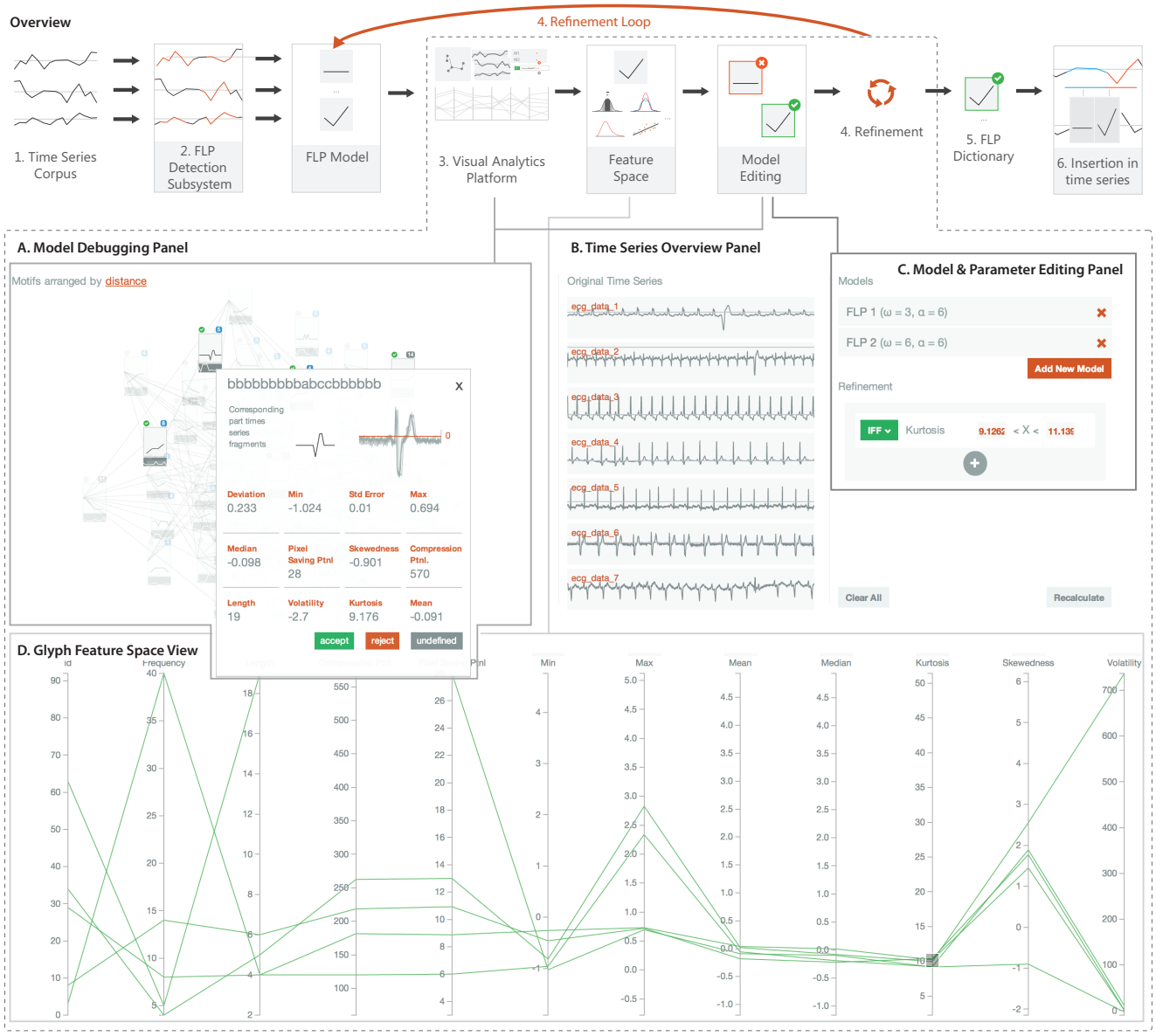


Fig. 2. The overview of the process followed in this paper and a screenshot of the visual analytics platform built to facilitate this process. The visual analytics platform interface has four interlinked panels to support: A) visual model debugging; B) overview of the larger time series to support context views for each motif (where it occurs in the actual time series); C) model and parameter editing; and D) navigation and refinement of the feature space for each motif.

6. a **Time Series Compression** step which inserts the motifs as glyphs in to time series for compression and/or anomaly detection (see Section 7).

4 FLP Detection Subsystem

In data compression, *dictionary encoding* is a family of methods that search a text for strings from a dictionary, and replace the matching strings with the corresponding codewords defined in the dictionary. The same dictionary is maintained by the encoder and the decoder, facilitating faithful translation from the original text to the compressed representation, and back to the original text. The primary goal of such a compression strategy is to reduce storage requirements. In a less algorithmic form, we apply a similar approach in writing by using abbreviations (e.g., UN for United Nations and symbols (e.g., § and ©)). Such a 'compression' method can be based on a general 'dictionary' applicable in a wide context (e.g., commonly-used currency signs), a 'dictionary' that is meaningful only to a specific group of people or in a specific context (e.g., mathematical symbols, abbreviations for measurement units), or an ad hoc 'dictionary' to address a very specific requirement (e.g., we introduced FLP as an abbreviation for *Frequent Long Pattern*). The primary goal is to reduce the time and cognitive load required for reading long and frequently-occurring text strings, while it also facilitates space saving in many cases.

The condition of 'long and frequently occurring' reflects the principle of *entropy encoding* in information theory [10]. The most well-known entropy encoding technique is Huffman coding. In everyday life, we can observe many intuitive entropy encoding schemes. For example, traffic signs are designed to speed up readings of restrictions and instructions, which otherwise would require a few words or sentences. For less commonly used restrictions and instructions, we still have text-based traffic sign boards. The balance between icon- or glyph-based and text-based visual representations in traffic signage features considerations of frequency of usage, length and complexity of the original text, importance, and various human factors (e.g., learnability, recognition, and memorization).

This has inspired us to apply the concepts of entropy encoding and

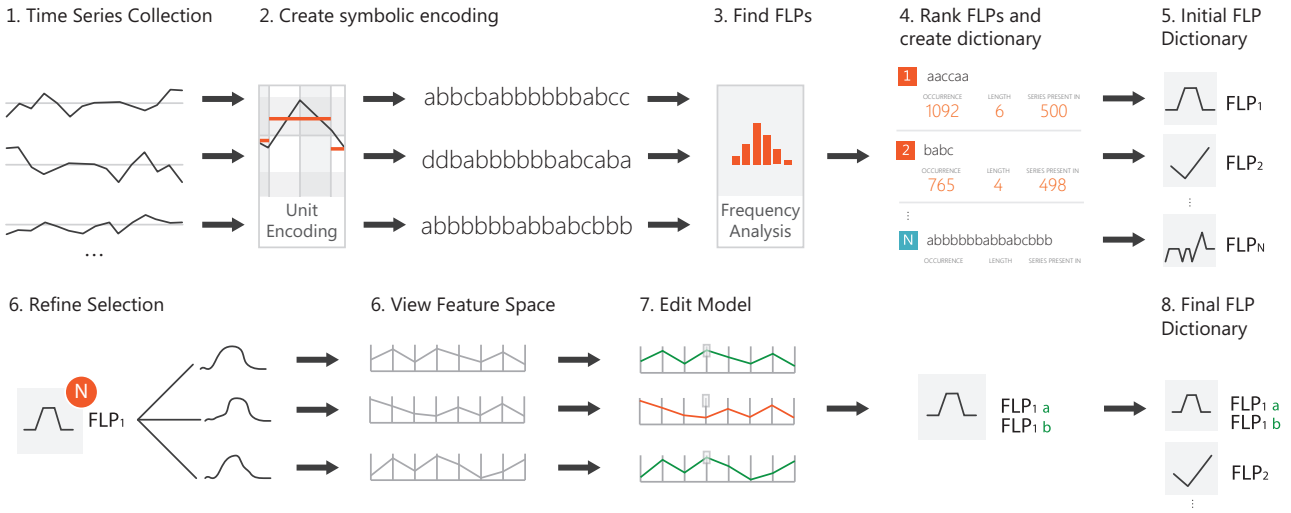


Fig. 4. Dictionary compilation steps – given a number of time series datasets $(T_1, T_2, \ldots, T_n)$ (1), each is approximated by a symbolic representation (2). A list of Frequent Long Patterns (FLPs) are found (3) from analysis of the symbolic encodings and these FLPs are ranked by their 'compression potential' (4). Highly-ranked FLPs are added to a dictionary. Finally, glyph representations are created for each FLP.

dictionary encoding to visualization, since they are everyday phenomena and underpinned by well-established mathematical theories. We can draw a parallel between a time series and a piece of text, and between a frequent long pattern in a collection of historical time series and a common long string in a text corpus. This analogy suggests that we can adopt a computational process to compile a dictionary, since there have been many previous works on symbolic processing of time series data in the literature (e.g., [33]). As glyphs have been used for time series visualization (e.g., [38]), we thus focus on a visual representation that supports dictionary-based compression while maintaining overview and selective detail views.

Fig. 4 illustrates the algorithmic steps for creating the FLP dictionary for a corpus of times series data. This dictionary creation algorithm is designed to process a collection of time series in order to establish a dictionary consisting of a set of FLPs in a specific context and can consists of the following steps:

1. **Time Series Collection.** It is necessary to collect a set of time series in order to generate good statistical indications about common patterns and outliers.
2. **Unit Encoding.** Given a collection of time series, the next step in *Dictionary Compilation* is to translate them to symbolic representations. These symbolic representations collectively form a 'text corpus'.
3. **Find FLPs.** In this step, an efficient algorithm based on a suffix-tree structure is deployed to detect a list of FLPs.
4. **Ranking FLPs.** For each detected FLP, we compute its potential compression capacity alongside other parameters in the feature space (see Section 5). We then choose the high capacity patterns and/or those with features matching any user-defined requirements defined in the visual analytics platform.
5. **Creating Glyphs.** The glyphs can be created automatically based on various attributes of the time series they represent.

Consider that we have N_p FLPs in a time series, and each FLP is of a length of $M_i (i = 1, 2, \ldots, N_p)$ samples. If each glyph requires a display space with a fixed width ω_g (in pixels), the compression ratio in terms of horizontal visual space requirement is:

$$C_{FLPs} = \frac{\omega_s \sum_{i=1}^{k} M_i}{\omega_g N_p} \qquad (2)$$

where ω_s is the number pixels per sample, including interval space, along the t-axis in a conventional line graph. The overall compression ratio for the time series is

$$C_{timeseries} = \frac{\omega_s N_t}{\omega_s (n - \sum_{i=1}^{k} M_i) + \omega_g N_p} \qquad (3)$$

Spatial compression is merely an indicator of the potential benefits of a frequency-based encoding strategy. In general, representing parts of a time-series as appropriate glyphs can benefit some common visualization tasks, such as identifying familiar FLPs, focusing attention on potential anomalies, and performing visual searches. The main cost is the loss of accuracy of those glyph-encoded parts of the time series. The trade-off is a universal mechanism at the heart of visualization. Many visualization techniques, such as zooming, glyph-based techniques and metro-maps, feature such a trade-off. In fact, the conventional line graphs for time-series visualization usually incur a significant loss of accuracy in comparison with the raw data, since the screen resolution is usually much lower than the data resolution. With interactive visualization, the loss of accuracy in overview can be recovered from details on demand [40], for example, using zoom or pop-up windows.

4.1 Unit Encoding

Let T be one of such a time series with N_t samples, we first divide it into N_{unit} *time units*, each of which consists of a fixed number of samples. In other words, there are N_{spu} samples per unit, such that $N_{unit} = \lceil N_t / N_{spu} \rceil$. In the literature, various terms have been used to denote such a time unit, e.g., 'time primitive' and 'chronon' in [5] and 'window' in [33].

An encoding scheme is then selected to map each unit to a symbol in an alphabet A. In this paper, we denote such symbols with lower case letters, such as a, b, and so on. Such an encoding scheme usually facilities an initial lossy compression in numerical representations. The potential compression ratio is influenced primarily by the size of the alphabet $|A|$, the number of samples per unit, N_{spu}, and the number of bits per sample value. However, some parts of the time series will not be encoded as glyphs and will be displayed as conventional time-series plots instead since their points do not fall within the realms of the frequently occurring patternss.

Being able to index and compare time series is of importance in this work due to the necessity to determine whether or not particular patterns in a time series are different from what has already been

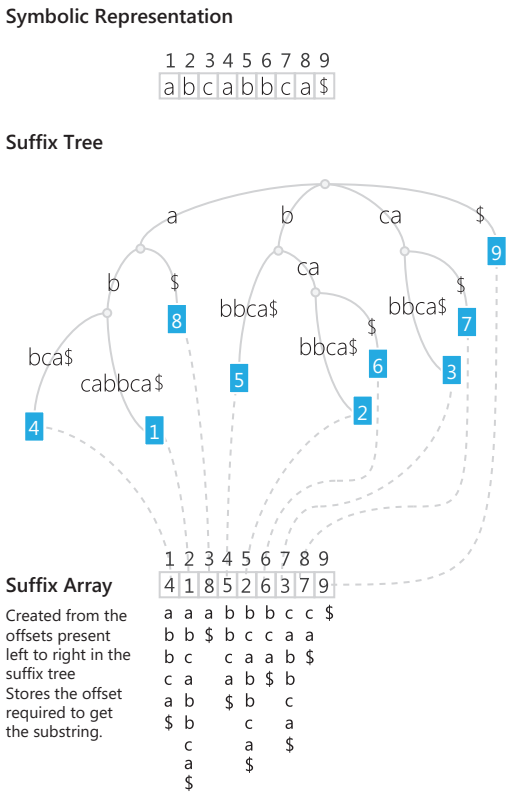


Fig. 5. Suffix tree and array representations, adapted from [22].

seen. In this work, we chose to base our algorithm on the symbolic approximation approach because it: 1) is widely used by the time series community; 2) is performant with O(n) complexity; and 3) has output ideally formed for indexing, hashing and storage in data structures such as suffix trees as performed in [33].

4.2 Frequent Long Pattern (FLP) Detection Algorithm

From the symbolic representation calculated in Section 4.1, we need to extract the most common, long substrings from available set of sequences with the expectation that selection of the most common strings will lead to greater compression.

The FLP detection model has three principle components: an 'enhanced' suffix array (with longest common (LCP) prefix array) to provide a fast, space efficient platform for fast string searching; an algorithm to compute the common elements in the string and their frequency; and a filter to fulfil any user-defined refinements on the features that motifs should have - by default, the model satisfies the requirement of finding those patterns matching the FLP properties of both 'long and frequent'.

4.2.1 Finding the FLP Candidates

A suffix tree [46, 42], or position tree is a data structure used to represent all suffixes of a string [42]. Fig. 5 is an example of such a tree and its corresponding suffix array adapted from Kasai *et al* [22]. Suffix trees thrive in their ability to speed up previously computationally intensive string operations such as: finding the longest common substring; finding repeats; finding maximal palindromes; and inexact string matching (the Weeder algorithm).

Although suffix trees are fast to query and build, they are costly to store. An alternative to the suffix tree is the suffix array, a sorted array of all suffixes that are able to perform all tasks that a suffix tree can perform in the same time when 'enhanced' with a Longest Common Prefix (LCP) array [1]. The LCP array gives for each adjacent pair of suffixes, the longest common prefix shared between them. For example, in Fig. 5, the longest common prefix between positions 6 and 7 in the suffix array, or '*bca*' and '*cabbca*' is 2, referring to the length of '*ca*' common to both suffixes. From this LCP array in combination with the suffix array, [1] showed that all suffix tree traversal types could be supported.

The algorithm, defined below takes a collection of time series approximations S_{approx}, finds all the common patterns, then returns a filtered list of FLPs matching the criteria defined in Section 4.2.2.

procedure DETECT_FLPS(S_{approx})
  $Set_{FLPs} \leftarrow \{\}$
  for all $S_i \in S_{approx}$ **do**
    $Array_{suffix} \leftarrow$ **sort**(**createSuffixArray**(S_i))
    $Array_{lcp} \leftarrow$ **createLCPArray**($Array_{suffix}$)
    $pos_1 \leftarrow 0$
    $index \leftarrow 0$
    while $index < Array_{lcp}.length$ **do**
      $pos_2 \leftarrow Array_{suffix}[index+1]$
      $length \leftarrow Array_{lcp}[index]$
      $end_{index} \leftarrow$ **max**$(pos1, pos2) + length$
      $FLP_{candidate} \leftarrow$**getSuffix**$(S_i, index, end_{index})$
      if $FLP_{candidate} \exists Set_{FLPs}$ **then**
        get$(Set_{FLPs}, FLP_{candidate}).occurrence$++
      else
        $FLP_{candidate}.occurrence \leftarrow 1$
        $Set_{FLPs} \leftarrow FLP_{candidate}$
      end if
      $index \leftarrow index + 1$
      $pos1 \leftarrow pos2$
    end while
  end for
  return Filter_FLPS(Set_{FLPs}) ▷ see 4.2.2
end procedure

4.2.2 Selecting the 'Best' FLPs

Let Ξ be an FLP found using the detection algorithm in Section 4.2.1, which can be written as a series of letters $\Xi = \alpha_1\alpha_2\ldots\alpha_m$. L_Ξ denotes its length (i.e., the number of letters) and F_Ξ denotes its frequency of occurrence obtained by the detection algorithm. F_Ξ can be the number of occurrence of Ξ in the data repository, or be moderated by a predefined maximum number. The compression ratio for Ξ is

$$C(\Xi) = \frac{\omega_s \times N_{spu} \times L_\Xi}{\omega_g} \tag{4}$$

In Eq. (4), ω_s is the minimal width along the t-axis required to adequately distinguish a data point in a time series (e.g., 2 pixels), ω_g be the screen width of a glyph, and N_{spu} is the window size for the symbolic approximation (i.e., the number of samples per letter). As the actual realization of this compression ratio depends on the probability of its occurrence in an input string, we thus moderate $C(\Xi)$ with F_Ξ as

$$P_C(\Xi) = F_\Xi \times C(\Xi) \tag{5}$$

We refer to $P_C(\Xi)$ as the *potential compression power* of Ξ. For example, consider a 20-letter string and Ξ, $N_{spu} = 10$. This FLP has 200 samples. If ω_s = 2 pixels, and ω_g = 40 pixels, we have $C(\Xi) = \frac{2\times10\times20}{40} = 10$. This implies that the glyph replacement will take up 10% of the original space requirement for the 200 samples. Assuming an unmoderated $F_\Xi = 50$, we have $P_C(\Xi) = 500$.

The key balance to be found in the approach is finding the optimal number for N_{spu} for any given dataset so that the system is able to compress optimally and intelligently. A larger number for N_{spu} will lead to a coarser representation of the time series, however it will probabilistically lead to further rates of compression. Conversely, a small number for N_{spu} will result in a more fine grained approximation for all time series but lead to less compression due to a generally higher amount of entropy in the approximation.

The selection of N_{glyphs} 'best' FLPs relies mainly on the P_C scores. This may depend on a predefined limit of N_{glyphs}, as too many glyphs would incur extra difficulties in learning, recognition and memorization. This may alternatively depend on a preset threshold for P_C. In the simplest case, the process can stop here and the results can be compiled into a 'dictionary' of FLPs $\mathscr{D}$. The resulting dictionary can then be used to compress a time series (see Section 7).

In some cases however, an FLP may classify what can be deemed an anomaly along with normal patterns due to subtle differences in the series even though the overall shape of the distribution is similar. This is where the visual analytics platform comes in to play whereby users can view the feature space of an FLP (mean, standard deviation, kurtosis, etc.) can sub classify an FLP and its corresponding time series with more fine-grained control.

5 Feature Space

As mentioned earlier, The FLP detection subsystem in Section 4 focuses on noise tolerance. Each of the selected FLPs determines an individualized model, Ξ, which is defined in the resolution of the unit encoding. This represents a significant approximation in terms of both temporal range and attribute value range. A selected FLP can easily encompass many similar patterns of the time series. To be able to distinguish between those time series segments that are 'valid' versus those that are not, we need the visual analytics platform to refine the model. First, we apply the individualized model, Ξ, to all or a subset of time series in the corpus. This would result in a collection of results, $\mathscr{R} = \{R_1, R_2, \ldots, R_m\}$, detected by using Ξ. We then construct a feature space for $\mathscr{R}$.

There are a variety of computable attributes for time series data used in the literature (e.g., 23 considered by Tam *et al* [41]). We consider a number of these variables to characterize each motif and the times series data they represent.

Alongside more trivial metrics such as *maximum* and *minimum* values, *average*, *standard deviation* and *variance*, the software also calculates the following:

1. *Correlation/error*: calculated as the Pearson product-moment correlation coefficient between the approximation values and the original time series values as a way of identifying how close the approximation is to the reality. This coefficient is defined in Equation 6 where X and Y are two arrays representing the approximated series and the time series respectively, and $\bar{X}, \bar{Y}$ represent the mean of both arrays.

$$\rho = \frac{\sum_{i=1}^{n}(X_i - \bar{X})(Y_i - \bar{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \bar{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i - \bar{Y})^2}} \tag{6}$$

2. *Kurtosis*: kurtosis, as illustrated in Fig. 6 gives a measure of the 'peakedness' of a distribution. This is calculated using the equation defined in Equation 7 where X is the original time series T.

$$\kappa = \frac{1}{n\sigma_X^4}\sum_{i=1}^{n}(X_i - \bar{X})^4 \tag{7}$$

3. *Skewness*: the skewness, illustrated in Fig. 6 gives a measure of 'curve asymmetry' [41] and is defined in Equation 8.

$$\gamma = \frac{1}{n\sigma_X^3}\sum_{i=1}^{n}(X_i - \bar{X})^3 \tag{8}$$

4. *Burstiness*: also called local variance, indicates how quick adjacent values rise or fall. The equation below has been adapted from [39].

$$L_V = \frac{1}{n-1}\sum_{i=1}^{n-1}\frac{3(X_i - X_{i+1})^2}{(X_i + X_{i+1})^2} \tag{9}$$

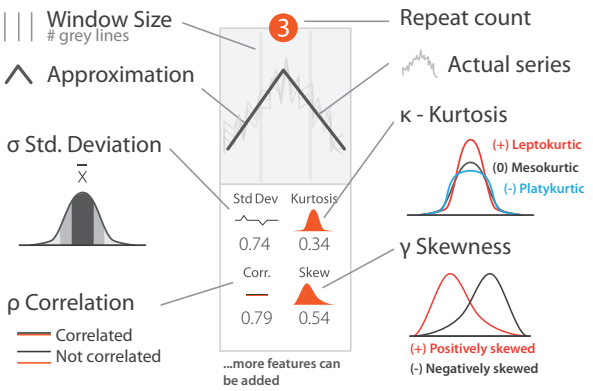


Fig. 6. A) The compression power of a particular FLP and the part of the time series compressed by a glyph is given by an overview bar. B) Details are available on demand when users click on a glyph in an implementation inspired by StackZooming [21].

Additionally, each motif has a compression potential calculated as part of the FLP algorithm discussed in Section 4.2.

Similar to Tam *et al* [41], we visualize these parameters using parallel coordinate plots (see Fig. 2D) and provide interaction to allow users to edit the model. Additionally, these features can be incorporated in to glyphs as illustrated in Fig. 6 and visualized for the user.

6 Model Editing

The process of model editing is enabled through many components within the visual analytics platform shown in Fig. 7 A and B. Users can click on nodes in the network view and accept or reject individual motifs whilst seeing the effect of such refinements immediately. The model editing window, shown in more detail in Fig. 7 A2 and B2 can be split in to two sections: 1) the model panel; and 2) the refinement panel.

The model panel is there to support the generation of motifs with differing window and alphabet sizes for the SAX algorithm used for FLP detection. These models are combined together and the combination of model outputs provides the total number of motifs available for selection.

The refinement panel supports the acceptance or rejection of motifs based on one or more of the parameters in the feature space. Users can combine these operations with a number of logical operations such as union (OR), intersection (AND) and subtract (SUB). Users can also specify whether values should lay within a range or be less than, less than or equal to, greater than or greater than or equal to some limit value.

The combination of simple logic statements provides a way for users to visually program the model so that it outputs only the desired results.

Fig. 7 shows an overview of the visual analytics workflow applied for the engineering example depicting the solenoid current measurements on a Marotta valve used to control fuel on a space shuttle. In Fig. 7 A, a network view displays the motifs found by the FLP detection algorithm as glyphs. Hovering over a glyph highlights the area that glyph represents in the time series corpus. This shows that one FLP represents all of patterns in the time series corpus, however an expert (derived from annotations in the original data set) marked the FLP highlighted in the black outline as anomalous. The next step will involve filtering this pattern out by looking in more detail at the FLPs and the time series patterns they represent.

Fig. 7 B shows a view of all 16 occurrences of the FLP of interest from the first step. Each occurrence has its own feature space representing the actual series represented. We have clicked on the motif of interest and can see in the popup that the profile matches the anomaly we wish to exclude.

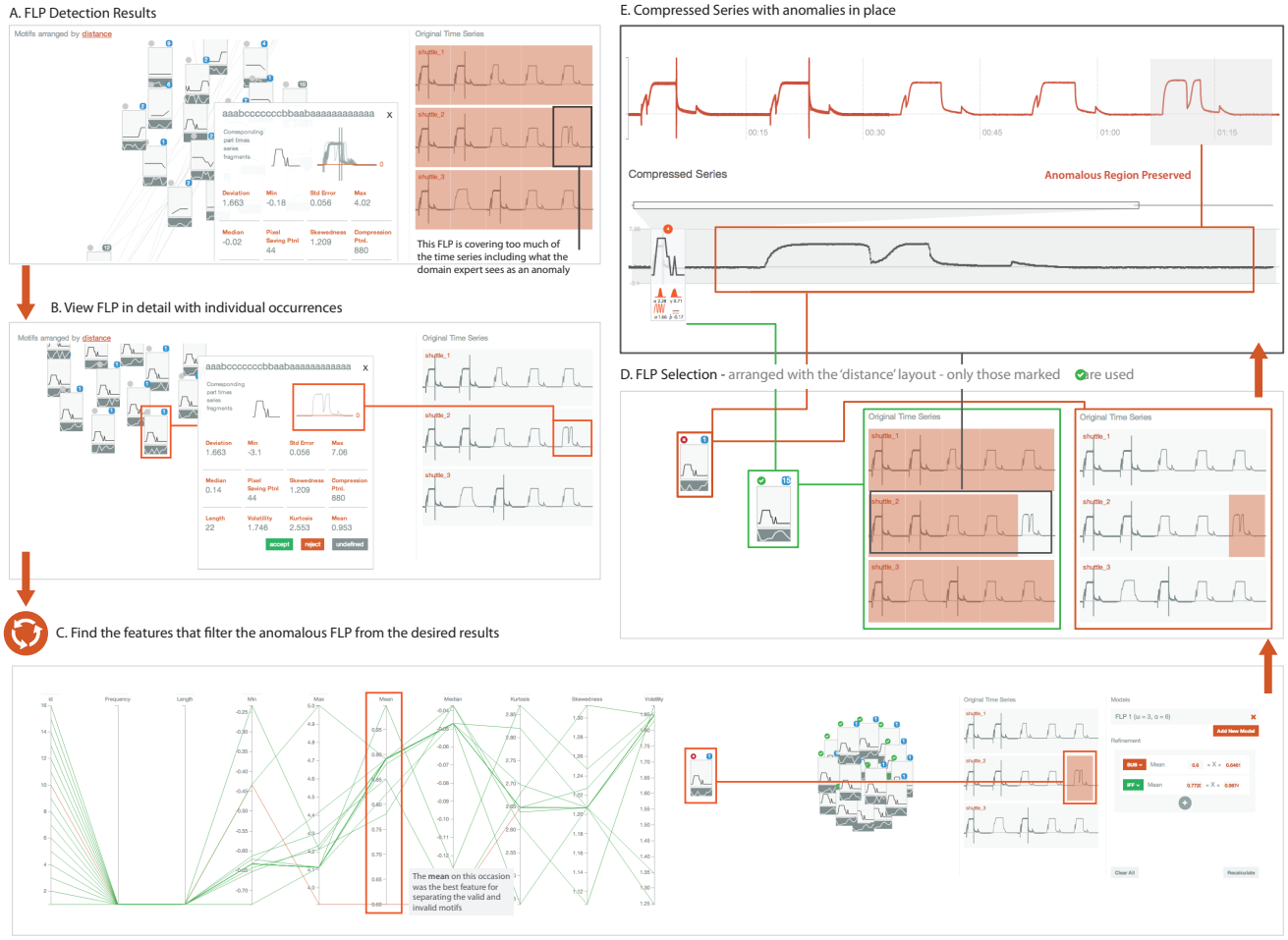


Fig. 7. A visual analytics approach to model refinement on an engineering time series depicting the solenoid current measurements on a Marotta valve used to control fuel on a space shuttle [14].

Having identified the anomalous motif, Fig. 7 C shows how, through use of the parallel coordinates, we can find the features that are able to classify between valid and invalid FLPs. In this case, the mean was a good choice since the anomalous FLP had a mean much lower than the others. The network layout automatically updates to visually separate the rejected FLPs from those that were accepted. In D, the set of FLPs are then exported as a dictionary where feature refinements are also preserved for later compression. Finally, a time series can be compressed using the FLPs in the dictionary and some time series T to create a compressed representation of the time series with the anomalous region preserved. This visual compression step is described more in the next section.

7 Visualizing the Compressed Time Series

Given a dictionary of FLPs $\mathscr{D}$ obtained from the FLP Detection Subsystem (Section 4.2), the process of glyph-based time series compression, illustrated in Fig. 8, can be applied repeatedly to many time series. Using traffic signage as an analog, once functional and visual representations of different signs have been decided, we can place them in any applicable location.

Glyph-based time series compression involves two steps: 1) given a dictionary of FLPs $\mathscr{D}$ and a time series T, create a compressed representation of a time series; and 2) using this representation of the time series, render it.

For the first step, we create the enhanced suffix array representation as performed when creating $\mathscr{D}$. From this data structure, a binary search is performed for each of the FLPs defined in $\mathscr{D}$ to identify all positions (defined by the suffix array) the particular FLP appears in. From the constructed list of matching positions for each FLP, some additional processing is made for the rendering layer. The first is to merge directly adjacent matches, that is, say one of our FLPs is composed of $abcd$, then for $abcdabcdddd, \ldots$, we have a record of $\Xi_{abcd} \uplus [1,5]$ including a list of matching indexes. Here the operator $\uplus$ denotes annotation, meaning that a FLP Ξ of symbols $abcd$ is annotated with index values 1 and 5. Since the distances in the start indexes for the occurrences of Ξ_{abcd} are separated exactly by its length L_Ξ, $1 + L_\Xi = 5$, the second occurrence can be merged with the first, and a 'repeat count' for Ξ_{abcd} is increased, so $\Xi_{abcd} \uplus [1,R2]$. Finally, a further check is performed to ensure that the indexes for the matched FLPs do not overlap, that is if $\Xi_{bcd} \uplus [2,6]$ has start indexes overlapping with $\Xi_{abcd} \uplus [1,R2]$. In this case, the FLP with the lowest compression ratio (C) power will be removed from the results. At the end of this process, the software produces a map from each start index of a FLP to its symbolic representation and repeat count.

The second step, rendering the time series, involves the incorporation of the glyphs representing compressed regions in to a time series visualization, the design of which is shown in Fig. 9. This representation maintains an overview bar showing the overall length of the time series whilst showing how much of a region has been compressed by a glyph. The glyphs provide information about the approximation, the run length (how many times the motif was repeated) and a number of metrics from the feature space. Glyphs can be interacted with and expanded to provide details of the underlying compressed time series represented by the glyph. This can be seen in Fig. 7 A and B. Addi-

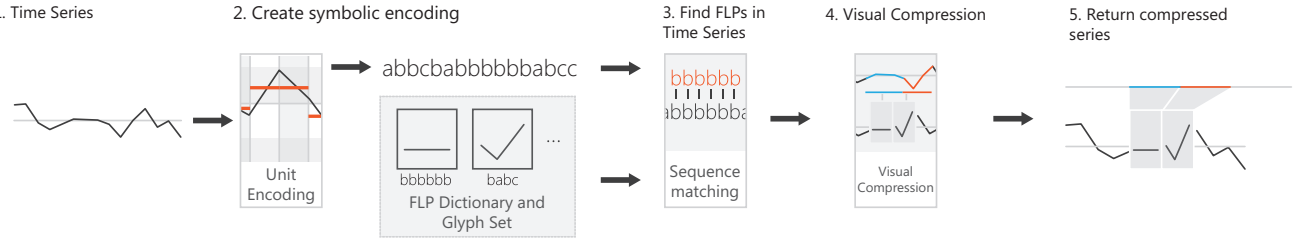


Fig. 8. Dictionary-based compression steps – Given a time series T_1, we encode it symbolically (2). Then, with the FLP dictionary created in (A), an algorithm finds the occurring FLPs in the incoming sequence (3). These found FLPs are replaced with glyphs (4) and the compressed time series is visualized.

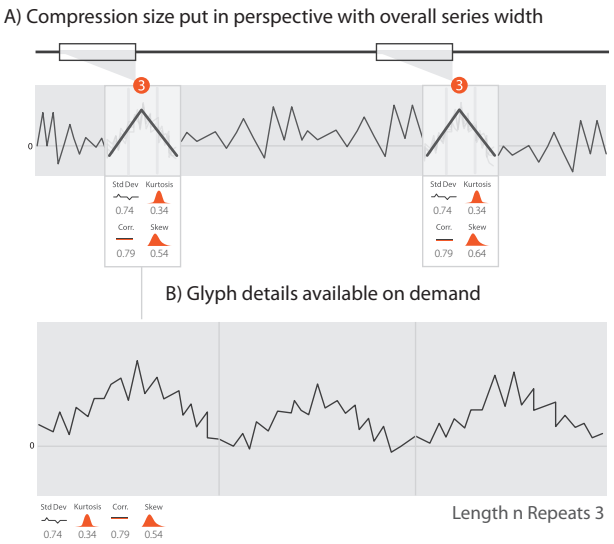


Fig. 9. A) The compression power of a particular FLP and the part of the time series compressed by a glyph is given by an overview bar. B) Details are available on demand when users click on a glyph in an implementation inspired by StackZooming [21].

tionally, when large areas are compressed, this changes the rendering of a time series somewhat since the series can spread over a much larger range. In many cases this is acceptable, but in some cases, the spacing may introduce interpretation problems. To navigate this issue, users can select to maintain the aspect ratio of the original series.

8 Software Availability

The software will be open source (on publication) and has been developed in a modular way to enable reuse at any level of the software stack. Each of the three components, the: FLP detection subsystem, implemented in Python; visual analytics platform, implemented as a thick client built with HTML5, CSS and JavaScript (with D3 and jQuery); and library to visualize the results, also built with HTML5, CSS and JavaScript (with D3 and jQuery) are available separately and can be run independently. The visual analytics platform and the compressed series output code rely on a simple JSON (JavaScript Object Notation) serialized format while the FLP detection subsystem outputs its results in the format required by the front end components.

Due to the modularity in the code base, and in particular the visual analytics platform, we envisage the software being of use in model refinement for similar problems requiring a visual analytics approach.

9 Conclusions and Future Work

In this work, we utilized the capabilities of visual analytics to address a challenging problem in time series analysis and visualization. In order to refine a model derived from the traditional FLP algorithm, visual analytics acted as a 'software development tool' by enabling users to test the model, analyze and visualize its results in a feature space, identify false positive results, identify related parameter ranges, and edit the model accordingly.

Because the feature space is computed directly with the original time series data, it can capture finer feature differences between an anomaly and its closely related FLP. Because the computation is only for a limited set of test results, it is also cost-effective. Most importantly, this allows human users to have the direct control over the tasks of debugging the results from a model and selecting features for refining a model.

In our future work, we would like to develop new visual representations of model space, which can support model visualization and editing in a semantically more meaningful manner.

References

[1] M. I. Abouelhoda, S. Kurtz, and E. Ohlebusch. Replacing suffix trees with enhanced suffix arrays. *Journal of Discrete Algorithms*, 2(1):53–86, 2004.

[2] W. Aigner, S. Miksch, W. Mller, H. Schumann, and C. Tominski. Visualizing time-oriented data — A systematic view. *Computers & Graphics*, 31(3):401 – 409, 2007.

[3] W. Aigner, S. Miksch, B. Thurnher, and S. Biffl. Planninglines: novel glyphs for representing temporal uncertainties and their evaluation. pages 457–463, 2005.

[4] D. J. Berndt and J. Clifford. Using dynamic time warping to find patterns in time series. In *KDD workshop*, volume 10, pages 359–370. Seattle, WA, 1994.

[5] M. Bögl, W. Aigner, P. Filzmoser, T. Lammarsch, S. Miksch, and A. Rind. Visual analytics for model selection in time series analysis. *IEEE Transactions on Visualization and Computer Graphics, Special Issue "VIS 2013"*, 19, 12/2013 2013.

[6] S. H. Cameron. Piece-wise linear approximations. Technical report, DTIC Document, 1966.

[7] K. Chan and A. W. Fu. Efficient time series matching by wavelets. pages 126–133, 1999.

[8] K.-P. Chan and A. W.-C. Fu. Efficient time series matching by wavelets. In *Data Engineering, 1999. Proceedings., 15th International Conference on*, pages 126–133. IEEE, 1999.

[9] L. Chen, M. T. Özsu, and V. Oria. Robust and fast similarity search for moving object trajectories. In *Proceedings of the 2005 ACM SIGMOD international conference on Management of data*, pages 491–502. ACM, 2005.

[10] M. Chen and H. Jaenicke. An information-theoretic framework for visualization. *Visualization and Computer Graphics, IEEE Transactions on*, 16(6):1206–1215, 2010.

[11] Y. Drocourt, R. Borgo, K. Scharrer, T. Murray, S. I. Bevan, and M. Chen. Temporal visualization of boundary-based geo-information using radial projection. *Computer Graphics Forum*, (3):981990, 2011.

[12] C. Faloutsos, M. Ranganathan, and Y. Manolopoulos. Fast subsequence matching in time-series databases. volume 23. ACM, 1994.

[13] S. Fernandes Silva and T. Catarci. Visualization of linear time-oriented data: a survey. In *Web Information Systems Engineering, 2000. Proceedings of the First International Conference on*, volume 1, pages 310–319 vol.1, 2000.

[14] B. Ferrell and S. Santuro. Nasa shuttle valve data. http://www.cs.fit.edu/ pkc/nasa/data/ ., 2005.

[15] J. Fuchs, F. Fischer, F. Mansmann, E. Bertini, and P. Isenberg. Evaluation of alternative glyph designs for time series data in a small multiple setting. 2013.

[16] M. C. Hao, U. Dayal, D. A. Keim, and T. Schreck. Importance-driven visualization layouts for large time series data. pages 203–210, 2005.

[17] M. C. Hao, U. Dayal, D. A. Keim, and T. Schreck. Multi-resolution techniques for visual exploration of large time-series data. pages 27–34, 2007.

[18] M. C. Hao, M. Marwah, H. Janetzko, U. Dayal, D. A. Keim, D. Patnaik, N. Ramakrishnan, and R. K. Sharma. Visual exploration of frequent patterns in multivariate time series. *Information Visualization*, 11(1):71–83, 2012.

[19] J. Heer, N. Kong, and M. Agrawala. Sizing the horizon: The effects of chart size and layering on the graphical perception of time series visualizations. In *In Proc. ACM Human Factors in Computing Systems (CHI*, pages 1303–1312, 2009.

[20] H. Hochheiser and B. Shneiderman. Interactive exploration of time series data. In *Discovery Science*, pages 441–446. Springer, 2001.

[21] W. Javed and N. Elmqvist. Stack zooming for multi-focus interaction in time-series data visualization. pages 33–40, 2010.

[22] T. Kasai, G. Lee, H. Arimura, S. Arikawa, and K. Park. Linear-time longest-common-prefix computation in suffix arrays and its applications. In *Combinatorial Pattern Matching*, pages 181–192. Springer, 2001.

[23] D. A. Keim, T. Nietzschmann, N. Schelwies, J. Schneidewind, T. Schreck, and H. Ziegler. A spectral visualization system for analyzing financial time series data. In *Proceedings of the Eighth Joint Eurographics / IEEE VGTC conference on Visualization*, EUROVIS'06, pages 195–202, 2006.

[24] E. Keogh, K. Chakrabarti, M. Pazzani, and S. Mehrotra. Dimensionality reduction for fast similarity search in large time series databases. *Knowledge and information Systems*, 3(3):263–286, 2001.

[25] E. Keogh, K. Chakrabarti, M. Pazzani, and S. Mehrotra. Locally adaptive dimensionality reduction for indexing large time series databases. In *ACM SIGMOD Record*, volume 30, pages 151–162. ACM, 2001.

[26] E. Keogh and S. Kasetty. On the need for time series data mining benchmarks: a survey and empirical demonstration. *Data Mining and Knowledge Discovery*, 7(4):349–371, 2003.

[27] E. Keogh, J. Lin, and A. Fu. Time series datasets, http://www.cs.ucr.edu/ eamonn/discords/.

[28] E. Keogh, J. Lin, and A. Fu. Hot sax: Efficiently finding the most unusual time series subsequence. In *Data mining, fifth IEEE international conference on*, pages 8–pp. IEEE, 2005.

[29] E. Keogh, L. Wei, X. Xi, S. Lonardi, J. Shieh, and S. Sirowy. Intelligent icons: Integrating lite-weight data mining and visualization into gui operating systems. In *Data Mining, 2006. ICDM '06. Sixth International Conference on*, pages 912–916, 2006.

[30] R. Kincaid. SignalLens: Focus+Context applied to electronic time series. *IEEE transactions on visualization and computer graphics*, 16(6):900–907, 2010.

[31] W. Lang, M. Morse, and J. M. Patel. Dictionary-based compression for long time-series similarity. *Knowledge and Data Engineering, IEEE Transactions on*, 22(11):1609–1622, 2010.

[32] P. Legg, D. Chung, M. Parry, R. Bown, M. Jones, I. Griffiths, and M. Chen. Transformation of an uncertain video search pipeline to a sketch-based visual analytics loop. *IEEE transactions on visualization and computer graphics*, 19(12):2109–2118, 2013.

[33] J. Lin, E. Keogh, and S. Lonardi. Visualizing and discovering non-trivial patterns in large time series databases. *Information visualization*, 4(2):61–82, 2005.

[34] J. Lin, E. Keogh, S. Lonardi, and B. Chiu. A symbolic representation of time series, with implications for streaming algorithms. *DMKD*, 2003.

[35] J. Lin, E. Keogh, L. Wei, and S. Lonardi. Experiencing SAX: a novel symbolic representation of time series. *Data Mining and Knowledge Discovery*, 15, 2007.

[36] P. McLachlan, T. Munzner, E. Koutsofios, and S. North. LiveRAC: interactive visual exploration of system management time-series data. In *Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, page 14831492, 2008.

[37] M. D. Morse and J. M. Patel. An efficient and accurate method for evaluating time series similarity. In *Proceedings of the 2007 ACM SIGMOD international conference on Management of data*, pages 569–580. ACM, 2007.

[38] P. Saraiya, P. Lee, and C. North. Visualization of graphs with associated timeseries data. In *Proc. IEEE Information Visualization*, pages 225–232, 2005.

[39] S. Shinomoto, K. Shima, and J. Tanji. Differences in spiking patterns among cortical neurons. *Neural Computation*, 15(12):2823–2842, 2003.

[40] B. Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *Visual Languages, 1996. Proceedings., IEEE Symposium on*, pages 336–343. IEEE, 1996.

[41] G. K. Tam, H. Fang, A. J. Aubrey, P. W. Grant, P. L. Rosin, D. Marshall, and M. Chen. Visualization of time-series data in parameter space for understanding facial dynamics. In *Computer Graphics Forum*, volume 30, pages 901–910. Wiley Online Library, 2011.

[42] E. Ukkonen. On-line construction of suffix trees. *Algorithmica*, 14(3):249–260, 1995.

[43] M. Vlachos, M. Hadjieleftheriou, D. Gunopulos, and E. Keogh. Indexing multi-dimensional time-series with support for multiple distance measures. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 216–225. ACM, 2003.

[44] M. O. Ward and Z. Guo. Visual exploration of time-series data with shape space projections. In *Computer Graphics Forum*, volume 30, pages 701–710. Wiley Online Library, 2011.

[45] M. Weber, M. Alexa, and W. Müller. Visualizing time-series on spirals. In *Infovis*, volume 1, pages 7–14, 2001.

[46] P. Weiner. Linear pattern matching algorithms. In *Switching and Automata Theory, 1973. SWAT'08. IEEE Conference Record of 14th Annual Symposium on*, pages 1–11. IEEE, 1973.

[47] H. Wickham, H. Hofmann, C. Wickham, and D. Cook. Glyph-maps for visually exploring temporal patterns in climate data and models. *Environmetrics*, 23(5):382–393, 2012.

[48] J. Zhao, F. Chevalier, E. Pietriga, and R. Balakrishnan. Exploratory analysis of time-series with ChronoLenses. *IEEE transactions on visualization and computer graphics*, 17(12):2422–2431, 2011.