Visualizing Poetry: Creating Tools for Critical Analysis

Luis Meneses, Laura Mandell, Richard Furuta

Introduction

Whereas the New Literary History emphasizes contextualizing literary texts, critics such as Stephen Greenblatt always employ close-reading techniques on the texts at hand. Although digital media allegedly privileges "distant reading" (Moretti), we can develop tools and methods to close-read and analyze documents. Moreover, these analyses can be visualized before being discussed by readers. Although too little work has been done in poetry visualization, it can be used to emphasize the structure of the narrative, the organization of the poem, the language elements, and the metaphors employed (Chaturvedi).

We have developed a set of visualization tools that aim to help scholars carry out the critical analysis of poetry. More specifically, the purpose of the visualization tools is to help synthesize and bring forward the key elements found in poems. Our visualization tools were developed using open–source programming languages and we used poems encoded in TEI from the Poetess Archive to create the visualizations.

As results, we found that the tools we developed purposely bring forward the graphical elements of poetry by carefully layering annotations (Tufte) based on literary criticism, thus creating new methods to analyze and create poetry. Additionally, our tools place special emphasis in viewing the poems from different perspectives (Meneses et al. "Computational Approaches to a Catalogue Raisonné of Pablo Picasso's Works") and visualizing the textual representation of the poetic texts in their formal structure (Audenaert et al.). These visualizations will be made available to scholars of all sorts, whether theoretically or historically engaged in the poetic texts that we use. In the following sections we will review the previous research related to poetry visualization, describe the rationale behind the tools we have developed and discuss their characteristics. Finally we will provide conclusions and recommendations for future work.

Previous Work

Little work has been done in creating tools to visualize and analyze poetry. However, most of the work has focused on creating new forms of expression using deformation and transformation techniques. These techniques are focused on creating new art forms using poems as an input source. Examples of these transformations include Visualizing Text by Diana Lange (Lange), Poetry on the Road (Schaffors, Müller and Pfeffer), Text Universe (Rapati) and Ira Greenberg's Syntactic Arthropod (Greenberg). However, we have found that in many cases the transformations applied to a poem renders it unreadable. On the other hand, the visualizations we have developed transform the original poem to create a new representation, but keep the poem visible and available for study and analysis.

The visualizations in Understanding Shakespeare (Thiel) provide an overview of different plays by displaying the most commonly used words grouped for each character. In this visualization, blocks of text represent scenes that are scaled depending on their number of terms. Additionally, the speeches of major characters are highlighted and characters are ordered by appearance. More interestingly, this project includes a section called Shakespeare Summarized where the most representative parts of the text are chosen by a software algorithm that references the frequency of the words it contains. In other words, the summarization algorithm chooses a sentence that contains most of the prominent terms within an entire speech. In the end, this summary provides a way to read an entire play in less than a minute. Although the results of Understanding Shakespeare are displayed on the web, the end result of the project was a printed book version that provides a different way of reading the plays.

Frameworks have been built with the specific purpose of analyzing poetry. Myopia is a framework that enables its users to analyze poetry by visually emphasizing the structure of the narrative, the organization of the poems, the language elements, and the metaphors employed (Chaturvedi). Myopia was built in Python and used open source libraries for displaying multidimensional graphics, which could be manipulated using an intuitive user interface. Additionally, The Mandala Browser is a rich prospect interface that can be used to analyze the different relationships occurring within a document corpus. Because of its flexibility, the Mandala Browser is widely used within the humanities (Gainor et al.; Ruecker, Radzikowska and Sinclair).

Visualizing Poetry

Our visualization tools were developed using Processing ("Processing.Org")[13], an open-source programming language designed specifically for visual artists and designers with the purpose of teaching the fundamentals of programming. Casey Reas and Ben Fry started Processing as a project in 2001. However, Processing has evolved through the years and it is used beyond its original pedagogical scope. Nowadays, Processing is also widely used by visual artists and designers to create new ways of displaying of data, animations and digital artworks.

Using Processing has some obvious advantages related with the execution of the source code and its portability. Processing is an extension of the Java Imaging Library, so Processing files can be executed as Java programs. Additionally, the Processing language is also available as a JavaScript port ("Processing.Js")[14], making it possible to run some of our visualizations in a modern web browser. We believe that the main advantage derived from using Processing is the simplification of the development process by encapsulating the complex data structures into simpler objects and methods. However, Processing is not a perfect programming solution. For example, parsing complex XML documents and creating complex user–interactions can be difficult. Because of this, we also relied on external libraries and other programming languages to achieve our end results.

In this paper we will describe 4 different tools we developed to visualize and create poetry: Graphwave, SentimentGraph and SentimentWheel and Ambiances. These visualization tools were created independently, but they all share similar characteristics. In this section we will elaborate on their affordances, advantages and specific uses.

GraphWave

To create this visualization, we used a process that borrows concepts from the term concordance in Picasso's poetry (Meneses et al. *Picasso's Poetry: The Case of a Bilingual Concordance*). This process can be described in four steps. First, we identified the unique terms in a selected poem. Second, each unique term was placed in a node in a Red Black Tree. A Red Black Tree is a binary data structure where operations can be performed O(lg n) time, where n is the number of nodes or terms. Third, each term occurrence along with its metadata was placed in a linked list attached to each unique node in the tree. Finally, the poem transcription and with the binary tree are traversed simultaneously, drawing a square in the canvas for the terms in the poem. The vertical position of each square is calculated based on the frequency of the term it represents, and its color is more saturated if the term is used less frequently. Thus, unique terms exhibit a more saturated color and are placed higher along the canvas while common terms have a more transparent surrogate and have a relative lower position.

In the end, the resulting visualization creates a colorful wave-resembling pattern, where terms are easily identifiable and can be referenced to the different parts of the poem. Additionally, the visualization keeps the poem readable by following the pattern from left to right. Longer poems are split into separate vertical waves. Figure 1 shows a visualization of “When I Have Fears” by John Keats.

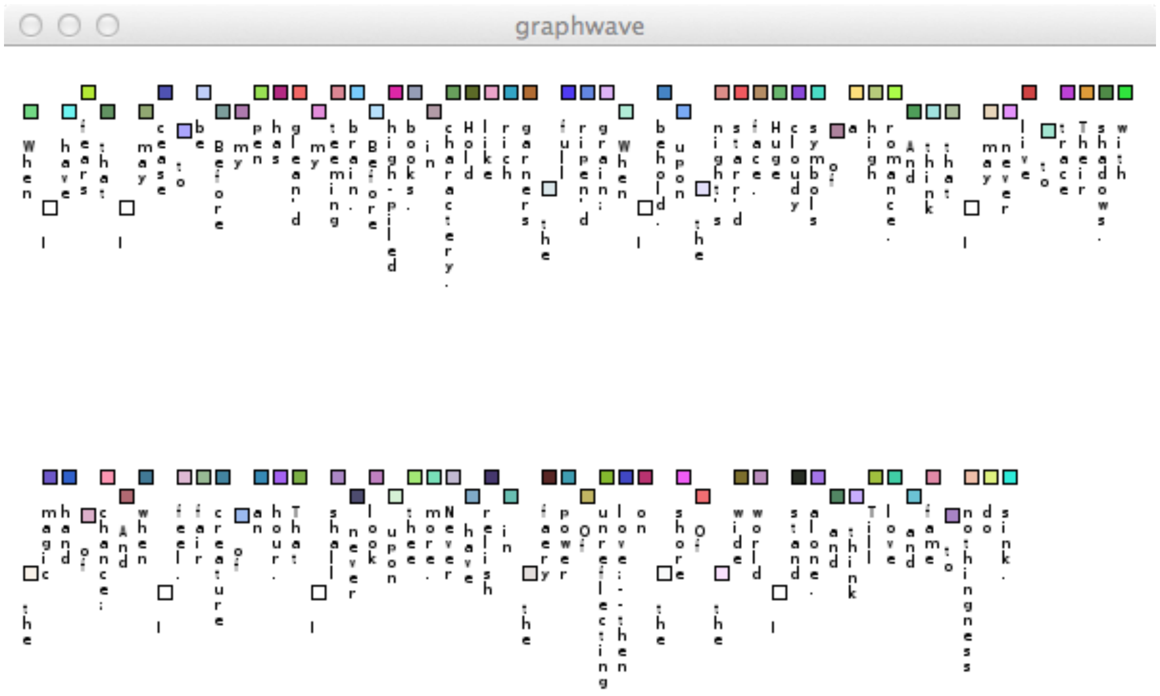
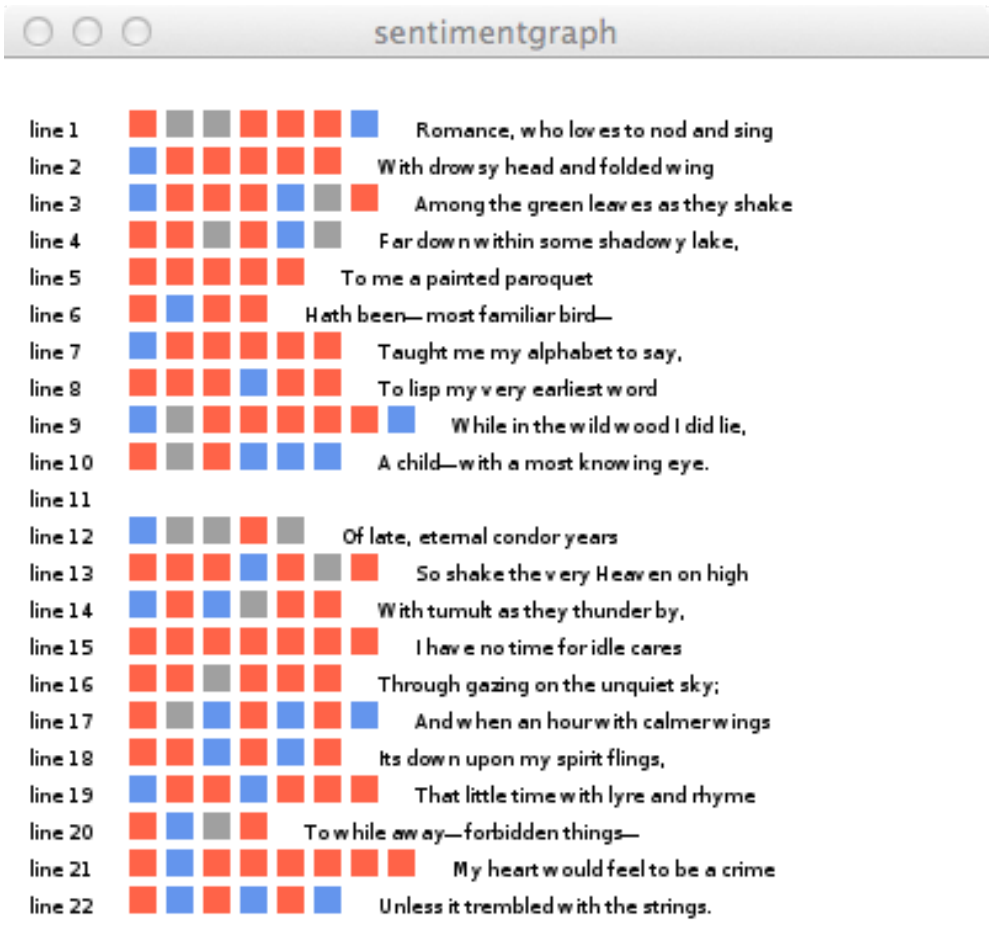


Figure 1: Viewing “When I Have Fears” through SquareWave.

SentimentGraph

In the SentimentGraph, we analyzed the sentiments behind each term in the text. We created this visualization using a two-step process. First, we used a Python script to query the Sentiment Analysis API ("Sentiment Analysis - Text-Processing.Com Api V1.0 Documentation"), which labeled each term as positive, neutral or negative. Second, we used the sentiments for each term and we correlated them against the poem transcription. Terms with a negative connotation are represented as red squares, whereas neutral terms are gray and positive are blue. Additionally, we included information regarding line numbers and a transcription of the poem. However, one drawback of this visualization is that it ignores the surrounding context of each term and the sentiments extracted from the web API reflect only the most common use of each word. Figure 3 shows the visual result after feeding “Romance” by Edgar Allan Poe to the visualizing algorithm.



sentimentgraph

line 1 🟥⬜⬜🟥🟥🟥🟦 Romance, who loves to nod and sing
line 2 🟦🟥🟥🟥🟥🟥 With drowsy head and folded wing
line 3 🟦🟥🟥🟥🟦⬜🟥 Among the green leaves as they shake
line 4 🟥🟥⬜🟥🟦⬜ Far down within some shadowy lake,
line 5 🟥🟥🟥🟥🟥 To me a painted paroquet
line 6 🟥🟦🟥🟥 Hath been—most familiar bird—
line 7 🟦🟥🟥🟥🟥🟥 Taught me my alphabet to say,
line 8 🟥🟥🟥🟦🟥🟥 To lisp my very earliest word
line 9 🟦⬜🟥🟥🟥🟥🟥🟦 While in the wild wood I did lie,
line 10 🟥⬜🟥🟦🟦🟦 A child—with a most knowing eye.
line 11
line 12 🟦⬜⬜🟥⬜ Of late, eternal condor years
line 13 🟥🟥🟥🟦🟥⬜🟥 So shake the very Heaven on high
line 14 🟦🟥🟦⬜🟥🟥 With tumult as they thunder by,
line 15 🟥🟥🟥🟥🟥🟥🟥 I have no time for idle cares
line 16 🟥🟥⬜🟥🟥🟥 Through gazing on the unquiet sky;
line 17 🟥⬜🟦🟥🟦🟥🟦 And when an hour with calmer wings
line 18 🟥🟥🟦🟥🟦🟥 Its down upon my spirit flings,
line 19 🟦🟥🟥🟦🟥🟥🟥 That little time with lyre and rhyme
line 20 🟥🟦⬜🟥 To while away—forbidden things—
line 21 🟥🟦🟥🟥🟥🟥🟥🟥 My heart would feel to be a crime
line 22 🟥🟦🟥🟦🟥🟦 Unless it trembled with the strings.

Figure 2: Viewing "Romance" through SentimentGraph.

SentimentWheel

The visualization behind SentimentWheel borrows some concepts from SentimentGraph. Creating this visualization involves a two-step process. Frist, we used a Python script to analyze the sentiments expressed. This time, we analyzed the sentiments expressed by each the each line of the poem. As before, the sentiments for each line were extracted using the Sentiment Analisis API. Second, we ran the sentiments we obtained with the API along the lines and individual terms form each poem. In this case, we decided to visualize the poem as a wheel. The line sentiments are expressed as consecutive circles: red for negative, gray for neutral and blue for positive feelings. One of the main advantages of this visualization is that the poem is readable by reading the terms clockwise. Figure 3 shows the resulting visualization of "When I Have Fears" by John Keats.

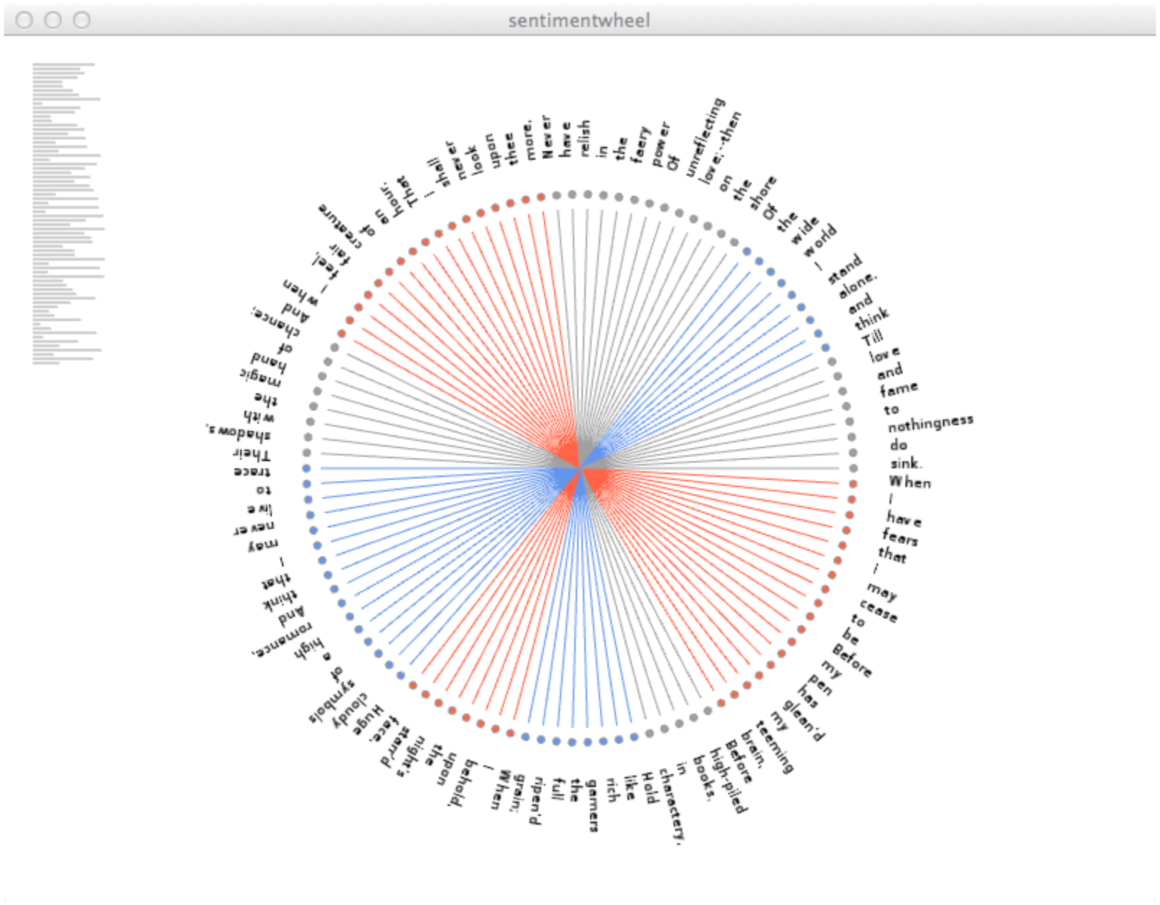


Figure 3: Viewing "When I Have Fears" through SentimentWheel.

Ambiances

For Ambiances we took a different approach. Our approach was based on the premise that the tools used nowadays to visualize poetry have three important characteristics. First, the visualizations are created after the poem has been published. Second, the scholars who create and use the visualization usually do not have any relationship to the author who created the original literary work. Third, the visualizations are a direct consequence of the transformations applied to the original text. When we put together all three characteristics we are left with a visualization tool that serves its purpose when highlighting certain passages in a poem, but does not have any effect or significant influence on the author as the poem was written beforehand.

To summarize, we attempted to challenge the three notions that in our opinion have defined how poetry is visualized. For this purpose, we built an interactive framework to write and visualize poetry. We have named this framework "Ambiances" (Meneses, Furuta and Mandell). The main goal of our framework is to create a tool that affords a symbiotic relationship between writing and visualizing a poem. In our framework, the process of writing a new poem influences its resulting visualization and the visualization also affects the process of writing.

The first prototype of Ambiances was composed of three different areas or "environments": a text editor where an author composes the poem, a minimalist-programming environment optimized for writing Processing code, and an environment where the resulting visualization was displayed.

However, we soon realized that using programming code in one of the environments created a steep learning curve and overhead for the authors. To solve this problem we eliminated the programming environment and used a Microsoft Kinect sensor in its place. A Microsoft Kinect is a motion sensing input device originally created as a peripheral for the Xbox 360 game console. When used in conjunction with open-source libraries, a Kinect sensor enables users to interact with computers through a natural user interface using hand gestures and body postures.

In the current prototype of Ambiances, the layout of the environments allows users to collaborate synchronously: authoring the poem and the visualization at the same time. Additionally, the interface encourages the authors to receive instant feedback through the visualization. Given that the authors cannot critique each other directly and can communicate through the visualization, Ambiances provides an unobtrusive way of writing poetry collaboratively that encourages unexpected interactions. For example: in the specific case where the visual elements are developing in syncopated opposition, we believe that the visualization and the interactions will provide hints that will allow the author of the poem to modify certain figures of speech accordingly. Figure 4 shows the resulting visualization obtained during the creation of a poem through Ambiances.

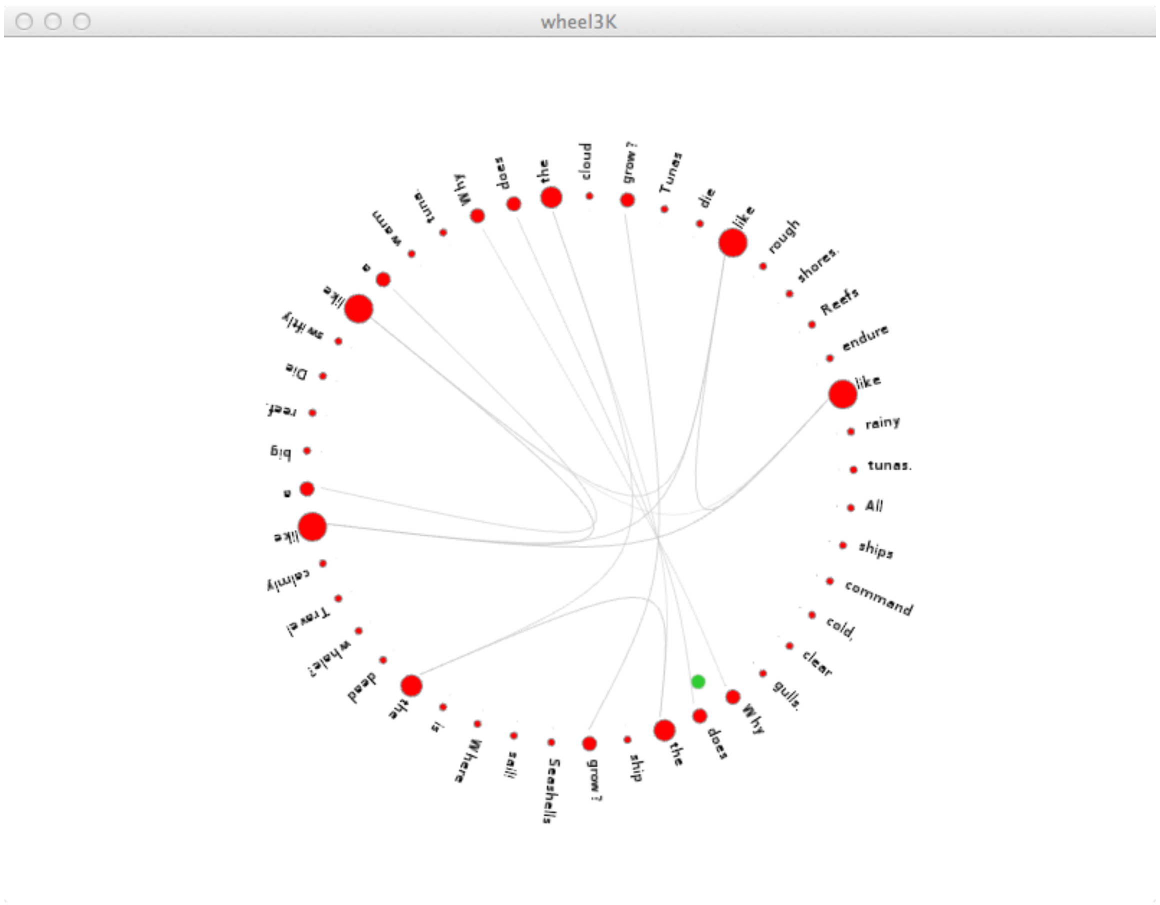


Figure 4: Visualizing the creation of a poem through Ambiances.

Discussion

We believe that the visualization tools we have developed constitute a different approach and method to close-read documents and open up new possibilities for analysis. In the specific cases when sentiments are used, they can create a baseline to facilitate the comparison and exploration of features from documents from different time periods, languages and themes. Because of the characteristics of the SentimentGraph and SentimentWheel, we can view documents from different periods as if they were concurrent. Thus, we have developed an interface that can be used to identify, discovering, and analyzing hidden relationships between the documents in a collection.

In a scholarly environment, comparing documents in a collection is a common undertaking. However, this comparison is limited to two documents. In our case, we can compare different documents by using multiple instances of the interface. In addition, poems can be analyzed from different perspectives by using different visualizations with different poems. We believe that the use of this use of the visualizations enables scholars to carry out new forms of analysis, explore new theories, and discover hidden relationships among different poems and authors.

Additionally, developing Ambiances left us with questions that we still need to answer. One of them is if Ambiances will work better with authors in specific genres, writing styles, languages and cultural backgrounds. It is not unconventional to assume that it will, but we still need proof. For this purpose, we are gathering a diverse group of authors to participate in this study. The feedback that we will collect from the authors and their interactions will be included in the new iterations of the system.

We must also address questions regarding capturing, storing and replicating the end result in the prototype for Ambiances. These become complicated issues, since we consider the end result to include the poem and its revisions, the interactions between the authors and the multiple resulting visualizations. Most text editors can store the multiple revisions made to a text. However, we must devise and implement mechanisms to store the interactions with the sensors and synchronize them with the visualizations. Our aim is to capture and recreate the performance that the authors were involved when creating a poem.

Conclusions and Future Work

We have described our approaches using the visualization algorithms and techniques to help scholars carry out the critical analysis of poetry. We believe that we achieved our goal by synthesizing and highlighting specific elements from selected poems. However, there are research possibilities open in this area. We summarize future work in three points. First, we will investigate if integrating specific annotations from TEI markup into our visualizations helps scholars during their research. Second, we will strive to create a system that will enable scholars and researchers to annotate and share their resulting visualizations. Finally, we will attempt to create methods that will allow scholars store their visualizations and retrieve them for later use.

References

"Processing.Js". 8/22/2012. http://www.processingjs.org/.
"Processing.org". 4/2/2012 2012. http://www.processing.org/.
"Sentiment Analysis - Text-Processing.Com Api V1.0 Documentation". 8/22/2012. http://text-processing.com/docs/sentiment.html.

Viewing Texts: An Art-Centered Representation of Picasso's Writings. Digital Humanities 2007. 2007. Print.

Chaturvedi, Manish "Visualization of Tei Encoded Texts in Support of Close Reading." Miami University, 2011. Print.

Gainor, Rhiannon, et al. "A Mandala Browser User Study: Visualizing Xml Versions of Shakespeare's Plays." *Visible Language* 43.1 (2009): 60-85. Print.

Greenberg, Ira. "Syntactic_Arthropod: Built with Processing". 8/22/2012. http://iragreenberg.com/poetess/viz02/.

Lange, Diana. "Visualizing Text - Openprocessing". http://openprocessing.org/sketch/44133.

Meneses, Luis, Richard Furuta, and Laura Mandell. "Ambiances: A Framework to Write and Visualize Poetry." *Digital Humanities 2013*. 2013. Print.

Meneses, Luis, et al. "Computational Approaches to a Catalogue Raisonné of Pablo Picasso's Works." *Interdisciplinary Journal for Germanic Linguistics and Semiotic Analysis* (2011). Print.

Picasso's Poetry: The Case of a Bilingual Concordance. Digital Humanities 2008. 2008. Print.

Moretti, Franco. *Graphs, Maps, Trees: Abstract Models for a Literary History*. London, New York: Verso, 2005. Print.

Rapati, Tiemen "Text Universe". 8/22/2012. http://openprocessing.org/sketch/10383.

Ruecker, Stan, Milena Radzikowska, and Stéfan Sinclair. *Visual Interface Design for Digital Cultural Heritage*. Surrey, United Kingdom: Ashgate Publishing, 2011. Print.

Schaffors, Andrea , Boris Müller, and Florian Pfeffer. "Esono.Com - Poetry on the Road 2006". 8/22/2012. http://www.esono.com/boris/projects/poetry06/.

Thiel, Stephan. "Understanding Shakespeare". 8/22/2012. http://www.understanding-shakespeare.com/index.html.

Tufte, Edward. *Envisioning Information*. Cheshire, CT: Graphics Press, 1990. Print.