Exploratory Browsing in Music Space

Heidi Lam
November 17, 2004

Agenda

- Motivation: Exploratory browsing?
- The ideal infovis solution: what should it be?
- Related work: displaying query-based results
- Prototypes: my proposed solution
- Dataset and implementation
- List of ongoing and future work

Project Idea

- How can computer tools/interfaces better support exploratory browsing?
- What is exploratory browsing?

Query Taxonomy

<table>
<thead>
<tr>
<th>Specified Target</th>
<th>Uncertain Target</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Specified Location</strong></td>
<td></td>
</tr>
<tr>
<td>Navigation: If a map of the space is present</td>
<td>Navigation: If a map of the space is present</td>
</tr>
<tr>
<td>Exploration: If not</td>
<td>Exploration: If not</td>
</tr>
<tr>
<td>Redundant encoding (target and location) to evaluate if the target is found</td>
<td>Single encoding (location) to evaluate if the target is found</td>
</tr>
<tr>
<td><strong>Uncertain Location</strong></td>
<td></td>
</tr>
<tr>
<td>Search/find with static evaluation</td>
<td>Browsing with potentially dynamic evaluation</td>
</tr>
<tr>
<td>(i.e., looking for something defined)</td>
<td>(i.e., target is ill-defined, and its properties may change/be refined along the process)</td>
</tr>
</tbody>
</table>
**Query Taxonomy**

<table>
<thead>
<tr>
<th>Specified Target</th>
<th>Uncertain Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specified Location</td>
<td>Exploration:</td>
</tr>
<tr>
<td>Exploring</td>
<td>Exploration: if not</td>
</tr>
<tr>
<td>Redundant encoding (target and location) to evaluate if the target is found</td>
<td>Single encoding (location) to evaluate if the target is found</td>
</tr>
<tr>
<td>Uncertain Location</td>
<td>Search/find with static evaluation (i.e., looking for something defined)</td>
</tr>
</tbody>
</table>

**Two Scenarios at a Record Store**

1. Looking for Ray Charles’ “Come Rain or Come Shine”
   - **Navigate**: Go to “Jazz” Search under “C” Find “Ray Charles” Search among his albums
   - **Find/Search**: “Do you have Ray Charles’ “Come rain or come shine”?

2. Browsing at the “Classical” section Came across a Jazzified version of Bach Go to the “Jazz” section Ray Charles’ album is on display

**Project Motivation**

- Exploratory browsing is not well-supported by current tools

<table>
<thead>
<tr>
<th>Specified Target</th>
<th>Uncertain Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specified Location</td>
<td>Navigate/Explore File explorer Web browser</td>
</tr>
<tr>
<td>Uncertain Location</td>
<td>Find/Search File searcher Internet search engine</td>
</tr>
</tbody>
</table>

**The Ideal InfoVis Solution**

To better support exploratory browsing, the interface should …

1. Provide context: to allow users to interpret the query results based on their input terms where am I? what am I looking at?
2. Guide navigation: going from the familiar to the unfamiliar where did I come from? where should I go next?
3. Assist refinement of target: based on retrieved results and query terms what am I looking for?
**Related Work: Overview**

- Focus on query-criteria based from a single search mechanism
- 4 approaches:
  1. **Spatial**: retrieved results are clusters into groups based on query terms, and displayed spatially
  2. **List**: retrieved results are displayed as a linear list
  3. **Temporal**: retrieved results in the context of timelines
  4. **Integrated**: multi-view with combinations of the above approaches

**Related Work: Spatial**

1. Show relationship between keywords—**InfoCrystal** (1993)
2. Show clusters only—**Lighthouse** (2000)

**Related Work: List**

- **Google**
- **Stuff I’ve Seen** (2003)
Related Work: Temporal

  provides personal events as landmarks on the time line for the retrieved results

Related Work: Integrated

- InfoSpace (2003): spatial + temporal

Prototype I

- Arranges query results as a **Venn diagram**
  - to put results in context of query terms
  - to relate neighbouring regions by a query term

Prototype I

- Uses a number of visualization techniques to convey these relationships
  - Colour-coding the search word with primary colours, and the cross-area with a mix of those colours
  - Perceptual Layering to indicate the relative importance of each result region

Prototype I

- Difficult to pack squares into non-rectangular containers
  - Limits max display capacity
- Can further cluster music record displays

Prototype II

- Uses rectangular containers
  - Harder to see “Venn” relationships, but still relates neighbours with a single query term
- “Piles” music by composer (or artist, genre, style)
Prototype II

• Semantic zooming
  1. Full display (composer, title, performer)
  2. Partial display (composer, title)
  3. Minimal display (composer)
  4. No display: number of results

Prototype II: New Query

• New query is an “extension” of old, linked by line, colour, and position
• Old queries fade and shrink with time

Dataset

• 8556 mp3 files extracted from 714 albums by 315 different artists
• Rock/pop and electronica
• Labeled with English terms (by Eric Brochu)

Implementation

• Architecture
  – Flat (at the moment): since the amount of data processing required is not extensive

• Platform and language:
  – Java using Eclipse IDE on Windows

• Libraries
  – swt.jar
  – No other graphics library used (yet…)

Current status & Next steps

<table>
<thead>
<tr>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Familiarize with database structure, refine prototype design
- Implement basic layout and individual element selection
- Implement semantic zooming, F+C with animation
- Implement new keyword query (spatial layout)
- Implement new keyword query (animation)
- Preparation of report and presentation