40

Scale Free Information Retrieval:
visually searching and navigating the web

Daniel Ethan Dreilinger

Bachelor of Arts, Mathematics
University of California at San Diego
June 1992

Master of Science, Computer Science
Colorado State University
August 1996

Submitted to the Program in Media Arts and Sciences
School of Architecture and Planning
in partial fulfillment of the requirements for the degree of
Master of Science in Media Arts and Sciences
at the Massachusetts Institute of Technology
June 1998

Author.....
Program in Media Arts and Sciences
May 8, 1998

Certified by.....
Dr. Andrew B. Lippman
Associate Director
MIT Media Laboratory

Accepted by.....
Stephen A. Benton
Chair
Departmental Committee on Graduate Students
Program in Media Arts and Sciences

Scale Free Information Retrieval:
visually searching and navigating the web

Daniel Ethan Dreilinger

Submitted to the Program in Media Arts and Sciences
School of Architecture and Planning
on May 8, 1998
in partial fulfillment of the requirements for the degree of
Master of Science in Media Arts and Sciences
at the Massachusetts Institute of Technology

Abstract

Search has become the dominant means of initiating interaction for users of the Internet and World Wide Web. Current search engine user interfaces present long lists of textual results which often must be laboriously culled through. This thesis introduces an innovative search engine that generates visual and interactive results based upon scale free imaging technology invented at the MIT Media Lab. The information retrieval system developed for this project includes an image-based spider that collects full color high resolution images of web pages, a unique layout engine that produces visual displays of search results, and a searchable index composed of text and image content for over 20,000 web pages. Seven distinct approaches to visual result presentation are described and evaluated in terms of retrieval effectiveness and scalability.

Scale Free Information Retrieval:
visually searching and navigating the web

Daniel Ethan Dreilinger

The following people served as readers for this thesis:

Thesis Reader.....

Henry N. Holtzman
Research Scientist
MIT Media Laboratory

Thesis Reader.....

Staffan Liljegren
Research Manager
Ericsson Media Lab

Contents

1 INTRODUCTION 11

2 BACKGROUND 15

Scale free imaging 15

Information retrieval 18

Visual information retrieval & related work 19

Internet Robots and Spiders 22

Human memory and user interface design 24

3 SPIDER 25

Robot traversal 26

Aliased host names 28

MD5 checksums 29

Robot exclusion standard 30

SQL database 32

Directed graph of hyperlinks 34

Image grabbing 36

Image conversion 37

Updating 38

Administrator interface 38

4 SEARCH ENGINE AND CORPUS 43

Search engine: SWISH-E 43

Modifications 44

Corpus: MIT Media Lab Web 45

5 LAYOUT ENGINE 49

Traditional 49

Thumbnail annotation 50

Thumbnail grid 51

Scale free 54

Scale free grid with dynamic user interface 57

Scale free with emphasis on document similarity 63

Scale free with emphasis on hyperlink relations 66

Proxy browser with directed graph annotations 70

6 EVALUATION 75

Is the information retrieval process improved? 75

Is the visual layout effective? 76

Scale free grid 76

Document similarity clustering 77

Hyperlink similarity clustering 78

Thumbnail annotations 80

Is the proxy browser useful? 81

Is the scale free retrieval system scalable? 84

General comments 85

Future work 86

7 CONCLUSION 89

REFERENCES 91

list of figures

Figure 1.1: Thumbnail images of three web pages. 12
Figure 2.1: Pyramid consists of a series of subsampled images. 16
Figure 2.2: Example of an extremely high resolution scale free image. 18
Figure 2.3: Xerox PARC's MagicLens interface. 20
Figure 2.4: University of New Mexico Pad++ user interface. 21
Figure 2.5: Apple Computer's HotSauce browser. 22
Figure 3.1: Block diagram of system architecture. 26
Figure 3.2: `robots.txt` sample 31
Figure 3.3: Image conversion process. 37
Figure 3.4: Administrator interface view of URL frequency, arranged by hyperlink depth. 39
Figure 3.5: Administrator interface view of URL frequency, arranged by HTTP status code. 40
Figure 3.6 Administrator interface view of URL frequency, arranged by internal status code. 41
Figure 3.7: Administrator Interface view of request for URLs with depth 2. 42
Figure 4.1: Histogram of URL depths from initial page. 48
Figure 5.1: An example of results in traditional layout. 50
Figure 5.2: An example of results in thumbnail annotation formt. 51
Figure 5.3: An example of results in thumbnail grid layout. 53
Figure 5.4: An example of results in the earliest scale free layout. 55
Figure 5.5: A zoomed in view of scale free search results. 56
Figure 5.6: An example of results in the dynamic grid interface. 58
Figure 5.7: Context sensitive window provides additional information. 59
Figure 5.8: Another view of the context sensitive window 59
Figure 5.9: Another view of the context sensitive window 59
Figure 5.10: Panning controls are enabled when user zooms in. 60
Figure 5.11: Another close up of the grid layout. 61
Figure 5.12: An even closer zoom reveals a web page at lifesize. 62
Figure 5.13: An example of results produced by document similarity clustering. 64
Figure 5.14: A cluster of related documents. 65
Figure 5.15: Another cluster of related results. 66
Figure 5.16: An example of results emphasizing hyperlink relations. 68
Figure 5.17: A view of the hyperlink interface after the user has zoomed in. 70
Figure 5.18: Web page viewed without the image annotation proxy 72
Figure 5.19: Web page viewed with the image annotation proxy 73
Figure 6.1: Traditional search results for 'movie map' 79
Figure 6.2: Link-based search results for 'movie map'. 80
Figure 6.3: Proxy view of the Media Lab home page. 83

list of tables

Table 3.1: An example row of the main table of URLs in the SQL database. 34
Table 3.2: An example row of the directed graph database. 35
Table 4.1: Summary statistics for Media Lab collection. 45
Table 4.2: HTTP status codes and frequency of occurrence. 46
Table 4.3: Frequency of HTML tags by type 47

Acknowledgements

My advisor Andrew Lippman for encouraging me to undertake this project and for advising me throughout my two years at the Media Lab. My readers Henry Holtzman and Staffan Liljegren for offering their ideas early on and then reading this thesis and providing thoughtful comments.

The Digital Life Consortium for funding this work, and Deborah Widner, Linda Peterson, and Laurie Ward for keeping everything running smoothly.

The Media Lab professors who made the experience so worthwhile: Pattie Maes, Hiroshi Ishii, Mike Bove, and Ken Haase.

Joey and Alex W., for expanding my musical horizons and elevating my threshold for extremely high temperatures. And Chris Dodge, not just for writing some of the scale free tools, but also for the showshoe adventures.

All the people I worked with at the MIT Media Lab from 1996-1998: Alex M., Arjan, Bill, Brad, Brendan, Brygg, Chris M., Dan G., Dan S., Freedom, Gert-Jan, Giri, Guillaume, Gwelleh, Jennifer, Johathan K., Jon, Jonathan D., Jonathan F., Karrie, Kathi, Maggie, Martin, Mike B., Nelson, Nitin, Nuno, Paul, Rob, Stefan, Thad, Tom, Vadim, Wad, Walter, Woja, and Yuri, to name a few.

Authors of the software I used—Perl and the CPAN, mSQL, SWISH-E, and the scale free imaging tools—without which this project would not have been possible.

My brother Sean for providing a thorough review of this document, and the rest of my family—Seth, Anna, & Chips—for years of encouragement and support. And above all, Allie, for everything—seven years of happiness, the trip to Ireland, Caoilfhionn, and what is yet to come.

1 Introduction

Search has become the dominant means of initiating interaction for users of the Internet and World Wide Web. Other techniques, such as bookmarks and browsing, are still useful but perhaps less so as we are confronted with exponential growth of web resources. The utility of bookmarks diminishes as on-line content becomes more dynamic; benefits of browsing are compromised in the wake of a growing sea of information. The size and dynamic nature of the World Wide Web often necessitate search, even when dealing with a subset of the web.

Most information retrieval systems for content on the World Wide Web—Lycos, Web Crawler, Yahoo, Excite, Alta Vista, HotBot, and numerous enterprise-based tools—offer a predominantly text based user interface. While different in size and subject area, virtually all widely available search tools feature identical interfaces. The user supplies a word or phrase, and the tool replies with a textual list of potentially relevant candidates, sometimes including a brief description or excerpt, and perhaps the date, author, or size. The disadvantage to textual descriptions is that they often fail to capture the essence of the pages they describe. The current searching model requires that extra time be spent requesting, downloading, and making firsthand assessments of relevance. Since it is not uncommon for a web search to result in hundreds, or even thousands, of potentially

relevant documents, the time and effort required to view even a fraction of these documents using current user interface technology is prohibitive. Many of the irrelevant results can be eliminated after a short glance, but only after they have been requested and downloaded into a user's browser from a remote server.

The ubiquity of text-based interfaces is somewhat surprising when you consider the extremely visual nature of the World Wide Web, and the people who use it. We often follow a search result hyperlink and wait for the page to load within our browser, make an initial evaluation based upon how the page looks, and then either reject it or read it more thoroughly. Humans have the ability to recognize and comprehend form very quickly. A person with basic Internet familiarity will quickly comprehend that the three web pages in Figure 1.1 are a personal home page, a commercial news site, and a technical paper, even though these pages are presented in small iconic format.

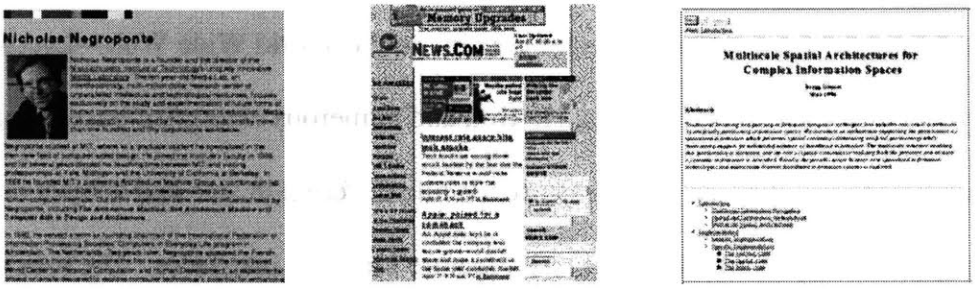

Figure 1.1: Thumbnail images of three web pages.

This thesis sets forth a scale free search engine that generates visual and interactive results and serves as a superior information retrieval interface for the World Wide Web. While the traditional information retrieval model affords one degree of freedom—the order in which the results are listed—scale free information retrieval allows expression of

results in terms of two geometric dimensions (i.e., x and y) and size. Complex relationships among the results are more naturally expressed, even in large quantities. Displaying results in the form of zoomable graphical versions of the corresponding web pages streamlines the search interaction—users can more quickly recognize the type of content represented.

I have explored employing these extra dimensions for more meaningful result presentation in several experiments. For example, size is used to indicate relevance to a query using a traditional information retrieval measure, such as the vector space model. Proximity between two images is used to express the term similarity between that pair of the results. In another experiment, size is used as an indicator of how many times a document is mentioned in other web pages, while proximity suggests distance in terms of number of hypertext links. A theme throughout all the experiments is that users are given a visual summary of multiple documents simultaneously, thereby accelerating the search and retrieval process.

2 *Background*

The software described in this thesis builds on existing research in the following areas: scale free imaging, information retrieval, Internet robots and spiders, and user interface design. Several related projects in data visualization are also described.

Scale free imaging

Andrew Lippman proposed scale-free images in 1995. To produce a scale free image, a picture is transformed into an intermediate form such that the following three criteria hold true:

1. The transformed image requires no significant increase in transmission bandwidth or storage space;

2. The transformation is asymmetric in that it is far easier to decode than encode;

3. The inverse transformation can occur with nearly equal quality at any particular image density or pixel array size.

If these three criteria are met, then the transformation is efficient, independent of viewing size, and quick to be displayed. We then call it "scale-free" since from the recipient's

point of view, it has no inherent raster format. A scale-free image is a natural by-product of predecessor work on scalable video [Bove, Lippman92] and asymmetric subband coding [Adelson84].

Scale free imaging was implemented at the MIT Media Lab in 1995 [Dodge95]. The scale free image format allows images to be rendered very quickly because they are first stored in a pyramid format, and then later re-assembled on the fly. First, a Gaussian pyramid is constructed by repeatedly low-pass filtering the image and then decimating the image by a factor of two. This is continued until the picture converges to a single picture element (pixel) which is the average color value of the full image. Collectively this sequence of images is called a pyramid (see Figure 2.1). Each progressively smaller layer contains one-fourth the number of pixels that comprise its predecessor.

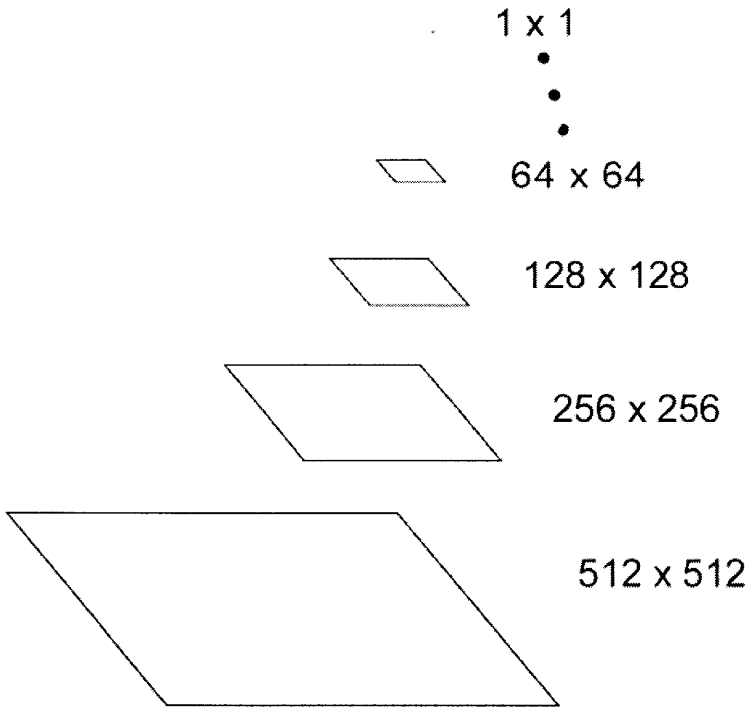

Figure 2.1: Pyramid consists of a series of subsampled images.

At this point sub-band coding can be applied to create a Laplacian pyramid of which each level contains one octave of frequency information. This is a simple data compression scheme that takes advantages of the human visual system's relatively low sensitivity to distortion in the high frequency bands. Once transformed from conventional formats into scale free pyramid formats, images are quickly rendered at any resolution by simply taking the next smaller pyramid level and interpolating as necessary. Because a low pass filter is applied to pyramid layers before subsampling, distortion at display-time is bounded to frequencies that are within one half an octave from the interpolated level.

The next level of abstraction involves organizing scale free images in a hierarchical format with a high-level layout language. The high-level layout environment facilitates arrangement of multiple scale free images at different sizes and locations on a single canvas. For example, Figure 2.2 depicts four successive views of a scale free image of an entire year archive of a Media Lab publication called 'Frames'. The upper left image of contains the entire composite image—a high-level layout description specifies placement of numerous individual scale free images. The upper right image of Figure 2.2 contains a close up of the bottom central region of the first image; the two images below these contain further close ups of the same region. The entire composite image contains 180 million pixels—over 230 times the image information displayable on a standard 1024 x 768 pixel high-resolution monitor.

Chris Dodge and Henry Holtzman developed several tools for creating and viewing scale free images. These scale free tools serve as one of the core software components in the scale free search engine. They also embedded the scale free imaging technology within a

web browser that uses this imaging medium to show the history of the user's browsing session. In this application, users may pull back from their view of a web page and see how it fits into a scale free montage of their browsing session history.

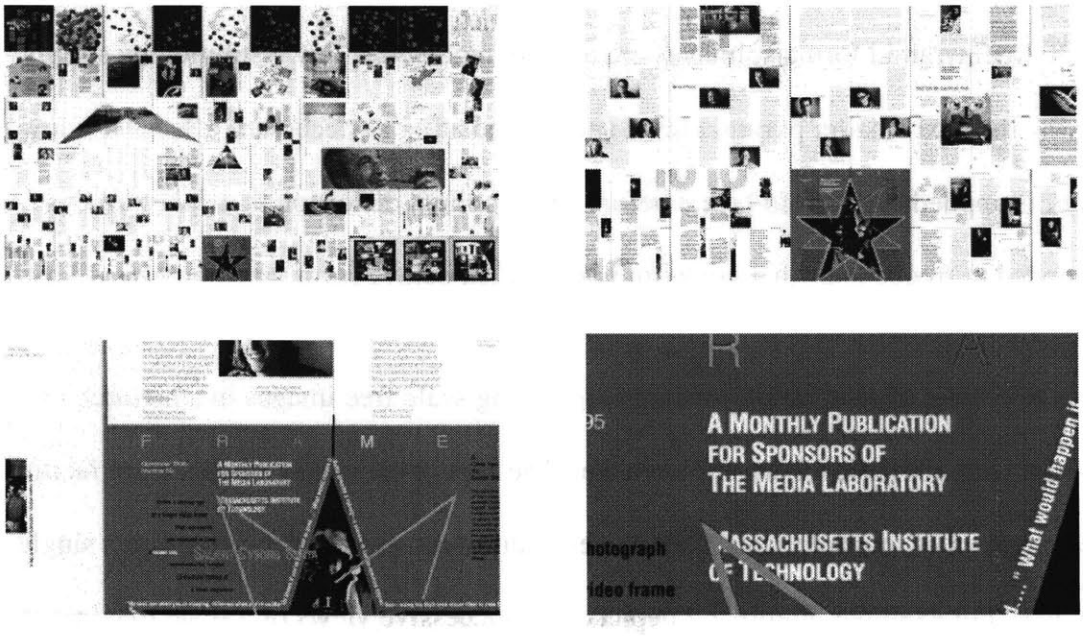

Figure 2.2: Example of an extremely high resolution scale free image.

Information retrieval

Information retrieval systems are one of the earliest applications of computers. Ever since the advent of mass storage devices in the 1950s—magnetic tapes and punched cards—we have relied on information retrieval techniques to efficiently navigate large data repositories.

The vector space model for information retrieval ranking, introduced by Gerard Salton [Salton89] is widely used and well documented. In essence, the vector space model involves using information theoretical observations about a document collection to make

relevance judgements. Documents and queries can be viewed as *n*-dimensional vectors, where *n* is the number of terms used throughout the collection. Individual values in a document (or query) vector correlate to the term frequency (TF) with which the term appears in the document (or query). Documents can be ranked with respect to a query by taking a cross product of the matrix composed of all document vectors and the query and vector. The term frequencies are multiplied by the inverse document frequency (IDF), an inverse measure of the total number of documents in which a specific term appears. Cross products are generally computed quickly because query vectors typically very sparse [Frakes92].

There are a number of commercial tools and several freely available information retrieval system implementations—WAIS, Glimpse, 'Excite for web servers', and SWISH-E. SWISH-E (which uses the vector-space model) was selected as the keyword retrieval engine for this project because of source code availability and ease of modification.

Visual information retrieval & related work

A fair amount of information retrieval research is devoted to image-based systems that specifically retrieve images rather than text-only or multimedia documents. For example, Virage, Inc. and Imagen sell search engines that retrieve images based on evaluation of a similarity measure relative to a query image. Given an initial photograph of a sunset over the ocean, for example, these search engines will find similar photographs within a large photographic database. No keywords are used. Image based techniques are also used in the video domain—Giri Iyengar and Andrew Lippman are pioneering methods for organizing and searching databases of motion pictures. They use Hidden Markov Models

to quickly and accurately classify sequences of video frames; for example TV sequences are classified as news or sports [Iyengar98].

Henry Lieberman investigated interactive browsing of very large display spaces without using the more traditional approach of zooming and panning in a single layer [Lieberman94]. Instead, he invented a software tool called a macroscope that involves zooming and panning in multiple superimposed translucent layers. While the new technique was found to be conducive to browsing maps, the superposition did not fare particularly well with photographs because the technique obscures image recognition. For this reason, the technique would probably not be best suited for web navigation either.

A more general approach to the multi-layered interface is Xerox PARC's MagicLens project investigates the use of 'magic lens filters'—an additional layer of the user interface that can be interactively positioned on the desktop to affect one or more display parameters [Stone94]. For example, Figure 2.3 shows an example of the MagicLens system in a mapping application where the lens serves as a magnifying glass.

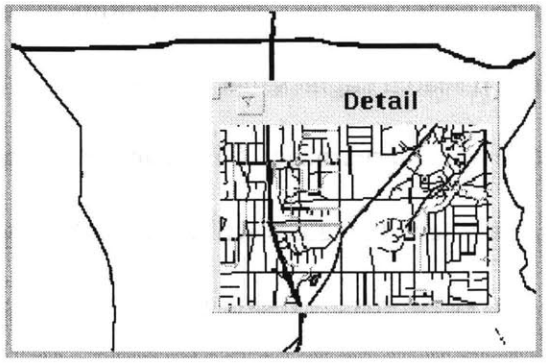

Figure 2.3: Xerox PARC's MagicLens interface.

In their work, Bederson and Hollan investigate use of zooming and panning interfaces in the Pad++ project [Bederson94]. Their TCL/Tk system renders text and graphics content at varying levels of magnification. The research focuses on use of this interface metaphor as applied to navigation of large dataspaces, but does not specifically address keyword searching. An example screen image is shown in Figure 2.4. Their workgroup has also created a zooming web browser that addresses inter-document hyperlinks [Bederson96].

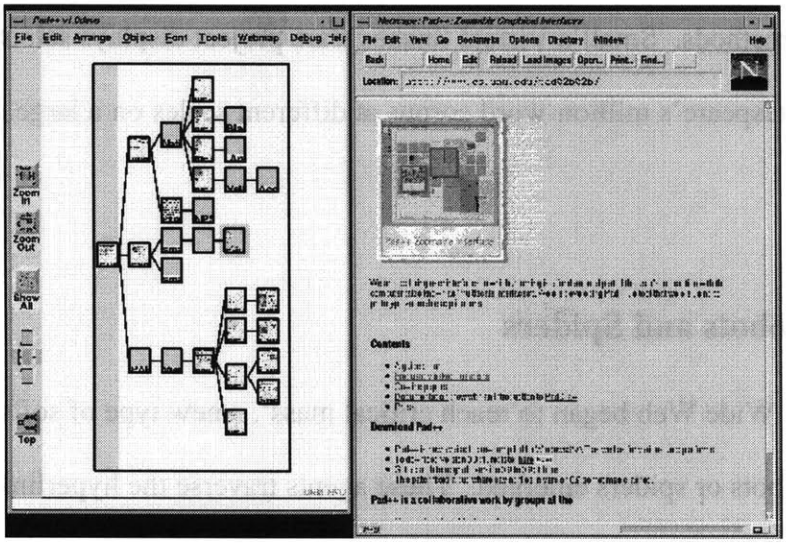

Figure 2.4: University of New Mexico Pad++ user interface.

Apple Computer's HotSauce project (shown in Figure 2.5) allows three dimensional navigation of textual objects representing the structure of a hypertext collection [Apple96]. Authors use Meta Content Format (MCF) files to express database information. HotSauce differs from the approach presented in this thesis insofar as the user is presented with text rather than images.

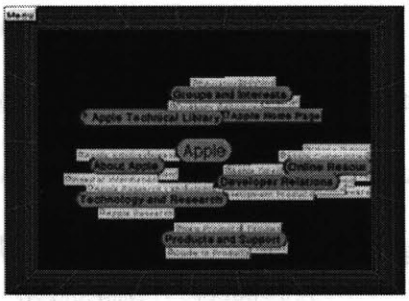

Figure 2.5: Apple Computer's HotSauce browser.

David Small, of the Aesthetics and Computation group at the MIT Media Lab, has researched techniques of navigating large bodies of text using typographical and three-dimensional methods. Small's Virtual Shakespeare project displays selections from William Shakespeare's million-word corpus at different scales on a large display [Small96].

Internet Robots and Spiders

As the World Wide Web began to reach critical mass[1], a new type of software agent called web robots or spiders emerged. These agents traverse the hyperlinked World Wide Web and gather full text content for inclusion in search engine indexes, by following hyperlinks from page to page. I describe the issues involved in building these tools in "Internet Search Engines, Spiders, and Meta-Search Engines" [Dreilinger96].

Commercial enterprises such as Lycos and Infoseek offer web-based search services that address the problem of searching the Internet. These centralized services consist of a spider, an index created during the spider's traversal, and a search engine. Millions of

[1] In the Spring of 1995, just as the National Science Foundation backbone was shut down and Internet backbone infrastructure shifted into the commercial arena, there were approximately 19,000 web sites [Rickard96, Netcraft98]. At this writing, in the Sping of 1998 there are more than 2.2 million web sites.

users consult the search engine with short textual queries and receive lists of hyperlinks to potentially relevant documents.

Meta-search engines such as SavvySearch [Dreilinger95] provide an additional layer of automation to address the growing number of search services by querying multiple conventional search engines simultaneously and displaying the results in an integrated format. Virtually all of these tools—both the conventional search engines and the meta-search engines—utilize the same user interface model: a keyword query leads to a textual listing of potentially relevant results along with overly brief descriptions that often do not capture the essence of the document. Descriptions typically consist of an automatically generated summary of the document, or simply the first few lines of text. Web authors can control these descriptions using meta-tags, and many take advantage by supplying misleading information that happens to make their page appear more prominently in the result listing.

In the future, metadata[2] standards like Resource Description Framework (RDF), eXtensible Markup Language (XML), and Platform for Internet Content Selection (PICS) will become the behind-the-scenes language of the web, allowing web spiders to *understand* web pages, rather than just read them. The World Wide Web Consortium's RDF and PICS standards, when widely deployed, will enable search engines to better catalog web pages and offer increased searching accuracy by providing search engines high level metadata pertaining to web page content [Lassila98].

[2] Metadata is information about information.

Human memory and user interface design

The memory research field provides ample evidence that people are far more adept at recognizing information than they are at recalling it from memory [Preece94]. This observation was a motivating factor in construction of the scale free search engine. In cases were the user is familiar with the domain being searched, the traditional textual titles and descriptions fail to take full advantage of the recognition versus recall phenomenon. The textual snippets reported to the user represent a small snapshot of a real web page—a small window of words that might not be easily recognized. The visual search results introduced in this paper exercise a user's recognition in important ways beyond text, such as color, form, and context.

There are also compelling reasons to use images when searching unfamiliar domains. Even if users will not recognize specific documents, they can still recognize familiar forms if they have any prior experience with the web. For example, there are distinct visual similarities among personal home pages, commercial home pages, technical reports, bullet lists, and slide presentations—regardless of author. Once users have learned the fundamental visual nature of the common forms of web pages, the recognition versus recall phenomenon applies when they see new pages with familiar form. Finally, small images arguably use screen real estate more efficiently than textual page titles and summaries.

3 Spider

Before the search engine can begin to field queries, an indexed collection of documents is necessary. The purpose of the scale free web spider is threefold. First, it must identify all documents to be included in the collection by following the branching network of hyperlinks, beginning with an initial web page. Second, the spider needs to retrieve the textual contents of all pages and store them locally, so that a searchable index can be created.

Third, the spider retrieves a visual snapshot—including inlined images, font colors, sizes, and styles, and text formatting—of each page as it would be seen in a web browser. The visual snapshot is certainly the most unique characteristic of the spider, which differentiates it from all other web spiders.[3] This chapter relates the issues and challenges encountered in construction of the spider.

A system diagram is presented in Figure 3.1. The spider collects information from the web and fills a local database with both images and HTML text. The Search engine, described in Chapter 4, creates an inverted index, and specially designed tools convert

[3] The Internet Archive Project, and the commercial spin-off called Alexa (www.alexa.com), incorporates a spider that collects multiple forms of content including images. This is done so complete web pages can later be reproduced during transient failures and for historic reasons.

images into their ultimate GIF and Scale Free formats. The last three subsystems: the user interface, the layout engine, and the search engine work together to field users' queries.

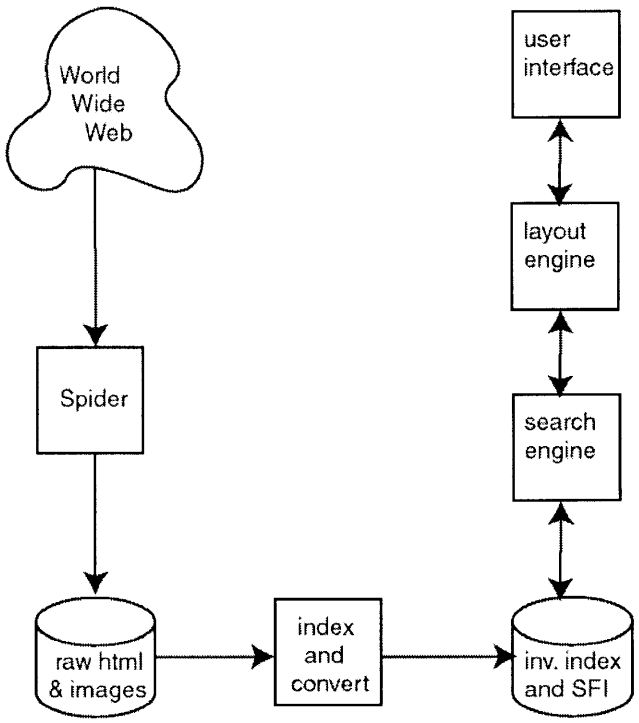

Figure 3.1: Block diagram of system architecture.

Robot traversal

The traversal algorithm is similar to standard breadth first search, with a few subtle differences. Breadth first search (BFS) is applied to web searching by starting with an initial page—in the case of the MIT Media Lab domain, the Media Lab home page is used. The spider extracts all hyperlinks and adds them to the end of a list of pages yet to be explored. The process continues recursively with the second URL on the list—the second page is grabbed and as before, its URLs are extracted and added to the end of the

list. After each page is explored, the HTML content is stored locally and the URL is marked on the list as finished. The process repeats until the list has no more unmarked URLs.

The subtle refinements of the BFS algorithm relate to search depth, inclusion criteria, and response time reordering. Depth is limited to 21 hyperlinks away from any web servers' home page. This prevents traversal of infinite document spaces described later in this chapter. Some URLs are automatically skipped, such as those that contain a question mark. Question marks are good indicators that a page executes a CGI program on the server. Those with excessively long URLs[4] are also ignored because they are indicators of automatically generated content. Occasionally, access of a web page will be postponed temporarily in order to minimize the rate at which requests are sent to any specific remote web server. This prevents the robot from getting tied up waiting for an overloaded, unresponsive server.

Breadth first search is arguably the most appropriate algorithm for traversing web content, especially if one accepts the heuristic that more important content is located within a few hyperlinks of a server's root page. Traversal algorithm choice becomes a more important factor when it is impossible to index all content for all servers, as would be the case if we were indexing the entire Internet [Pinkerton94]. In this case you want to ensure that more important pages are collected first because it is not possible to collect all pages.

[4] URLs greater than 255 characters in length were ignored. So far the robot has not had to reject any URLs on this criteria.

Aliased host names

A challenge of indexing the Media Lab domain (and large academic and corporate networks in general) is the great frequency of hostname aliases. The mapping of valid hostnames to web servers is many-to-one. If special consideration is not taken with the development of the spider, numerous identical pages would be encountered but remain undetected—indeed entire identical web servers would be processed and added to the index. This problem was solved by first finding the IP address for the hostname of each URL as it was discovered, then immediately performing a reverse IP to hostname lookup and regarding the first name on this list as the definitive name of the machine. All of the various aliases pointing to the same machine are grouped together and considered the same. The original URLs are also stored so that they can be quoted at query time for increased result utility.

For example, the alias `daniel.www.media.mit.edu` points to the machine `www.media.mit.edu`. While the intention of this alias is to serve URLs such as `http://daniel.www.media.mit.edu/daniel/index.html`, it also enables access to any other URL on that machine, such as `http://daniel.www.media.mit.edu/joey/index.html`, and in fact these URLs are encountered when web page authors use relative hyperlinks. In this case, the double lookup step finds that daniel.www.media.mit.edu is hosted by the machine with IP address `18.85.13.110`, which in turn returns the hostname `www.media.mit.edu` when a lookup is performed[5]. All the aliases are ultimately condensed to one[6].

[5] These hostnames and IP addresses are current as of March 1998 but are subject to change.

MD5 checksums

Even with special attention paid to multiply hosted web pages, not all duplicates can be found. Identical web pages are often served via multiple URLs, which cannot be detected with the URL alone. For example, the following URLs all point to the same physical page, but cannot be detected by URL alone:

```
http://www.media.mit.edu/~daniel
http://www.media.mit/~daniel/index.html
http://www.media.mit.edu/~daniel/home.html
```

To address this issue, an MD5 checksum is computed on the HTML text of every retrieved web page and incorporated in the index. MD5 is a message digest algorithm developed by Rivest in 1992 [RSA98, Rivest92]. The algorithm's purpose is to create unique 128-bit message digests or signatures of text messages using a one-way hash function. In cryptographic applications the digests serve as an aid to signing a document with a private key. The scale free web spider uses digests to detect identical web pages by generating a unique MD5 digest for each web page encountered and storing it in the SQL database. These digests are then used at query time to eliminate identical documents that originated from different URLs. While it is theoretically possible for MD5 checksums for different web pages to be the same, this is unlikely to occur as there are 2^{128} checksum possibilities.

---

[6] This is not the same as virtual hosting, where multiple machine names point to the same web server, but the server returns different content depending on the machine name.

Robot exclusion standard

Several interoperability issues must be taken into consideration when designing a web robot, specifically *how often a server should be accessed* and *which content is appropriate for collection.* It is extremely important to avoid interfering with the routine operation of the web servers that are indexed. Spiders are capable of requesting URLs thousands of times faster than human beings. Their rapid-fire requests can be harmful if care is not exercised by the spider author to insure a proper delay between accesses. Some areas of a web site's file space, such as a temporary directory, might be inappropriate for inclusion in an automatically created index. There are numerous situations in which a large volume of content is automatically created by a computer program, and is simply not appropriate for indexing, due to its transient or exceedingly verbose nature.

The robot exclusion standard, set forth in 1994 by participants of the Web Robots e-mail discussion list[7] as an informal set of guidelines to help guide web spiders through a web site, addresses the latter question of which content is appropriate. To implement the standard, the web administrator simply creates a file named `robots.txt`, which lists specific user agents (spiders, robots, etc.) by name, and any content to which they are forbidden access. For example, a typical `robots.txt` file might look like the one shown in Figure 3.2[8]:

[7] At the time the list was hosted by Nexor. Subsequently it was hosted by WebCrawler; presently it is hosted by mcmedia.com.
[8] This text is from http://www.yahoo.com/robots.txt, as sampled in March, 1998.

```
User-agent: *
Disallow: /gnn
Disallow: /msn
Disallow: /pacbell
# Rover is a bad dog <http://www.roverbot.com>
User-agent: Roverbot
Disallow: /
```

Figure 3.2: `robots.txt` sample.

In this case, all user agents (the asterisk is a wildcard, standing for anything) are forbidden access to URLs beginning with `/gnn`, `/msn`, `/pacbell`, and one specific robot, named Roverbot, is excluded from all available content. Presumably it was excluded because it did not interoperate with the server in an acceptable manner.

While the robot exclusion standard is effective when used by both parties—the content provider and user agent author—only about 5–10 percent of web administrators choose to provide such guidance. Not only do relatively few web administrators opt to provide a `robots.txt` file, but many user agent authors choose to ignore `robots.txt` entirely. There is no enforcement. For this reason, in addition to abiding by the robot exclusion standard, the scale free spider makes use of additional heuristic information about the Media Lab corpus.

Several large repositories of non-Media Lab e-mail discussions were identified and marked to be ignored because they would have substantially increased the size of the index. In addition, an infinite document space composed of vendor-supplied technical help and a large non-Media Lab bibliographic database were encountered and manually

marked to be ignored. If this action were not taken, the index would have been too large for the SWISH-E engine.

The maximum depth limit of 21 hyperlinks from the root of any web server works as a general measure to avoid infinite traversal. Investigation showed that hyperlinks around this depth and beyond were primarily limited to slide presentations.

The robot was invoked from within the Media Lab domain and was free to access all content, including pages that are limited to internal users only. No special consideration was made within the robot to distinguish between public and private content. Everything is indexed; then at query time, internal results are eliminated with a filter if the query originated from an external site.

Finally, the robot exclusion standard does not address the rate at which documents should be requested. The scale free spider accommodates this goal by alternating requests among different servers whenever possible, and waiting for an interval between consecutive requests to the same server. Rather than use a static interval of 5–10 seconds between requests, a dynamic interval of twice the time of the last response received from that particular server is applied. If a server is overloaded with requests, spider access slows; when a server is lightly loaded, spider access is much faster.

SQL database

A disk-based persistent storage solution is mandated by the fact that the robot must sometimes be stopped and restarted without loss of data. Another requirement is that

multiple programs must simultaneously access the database. SQL was an attractive back-end choice to provide atomic reads and writes, and disk based persistent storage.

The core data structures for the spider are maintained in a Structured Query Language (SQL) database. Of the several freely available SQL databases, Hughes' mSQL was selected based on its simplicity of installation and operation, and overall appropriateness for the task at hand. Hughes' mSQL (or mini-SQL) is an evolved, robust implementation of a small subset of SQL89 commands and data types [Hughes98].

The primary relational database table includes fields for normalized URL, original URL, the date first found, the date last checked, HTTP status when last checked, local storage location for HTML text in images, URL discovery depth, and the MD5 message digest checksum, and an internal status flag. Images are not stored inside the database—instead, a pointer to a collection of files on disk is used. The internal status flag tracks the state to which the page has progressed along the multi-step process from page discovery to image acquisition and conversion.

Table 3.1 depicts one row of data from this relation. In this example, the URL points to the Autonomous Agents Group home page, which was originally discovered with the machine alias '`agents`', and subsequently standardized to '`belladonna`' by the robot. The UNIX timestamps for the date first found and last checked appear next, followed by the last HTTP status, then the base name of any local files containing related content. The internal status flag is set to 1, meaning that the URL has been discovered by the robot but not yet indexed. The depth of 3 indicates that the robot found it three steps

away from the starting page, and finally, the unique MD5 checksum for the HTML contents of the page appears at the bottom.

URL	http://belladonna.media.mit.edu/groups/agents/
original URL	http://agents.www.media.mit.edu/groups/agents/
first found	886393663
last checked	892236946
status	200
file location	97/365905
internal flag	1
depth	3
MD5 checksum	9ef2bce043c553b3232573008c5e8bc5

Table 3.1: An example row of the main table of URLs in the SQL database.

Since image acquisition and conversion is a multi-step process, dedicated programs simply need to query the database and retrieve URLs within particular flag value. The acquisition and conversion programs then perform their tasks asynchronously and update the status flag appropriately. The SQL database allows concurrent access to several such dedicated programs.

Two other SQL tables maintained include a cache of `robots.txt` files and a directed graph of hyperlinks between documents in the collection. The `robots.txt` files are cached so that they are only retrieved from the network once every ten days—typically these files are quite small. The directed graph is described in the next section.

Directed graph of hyperlinks

As the robot traverses the document space, it keeps track of outbound hyperlinks from every page (an *outbound* hyperlink is defined as a hyperlink from the present page which refers to any other document in the collection; an *inbound* hyperlink is the opposite). At

the end of the traversal, this information is aggregated, and a table of both inbound and outbound hyperlinks is created for every URL. This data structure, called a directed graph, is collected and maintained by the robot for use in some of the layout schemes described in Chapter 5. The directed graph also enabled an interesting spin-off project involving a proxy web browser, which presents the user with web pages that are modified to include thumbnail images of inbound hyperlinks in one margin and outbound hyperlinks in the other margin.

URL id	4da3291c3f98658335b0dda282be06bf
outbound links	ff31d44b3e0636a23465eed902b2837a:44/706726
	6bd97f6c483a451d32370c8a7a34e0c2:53/882781
inbound links	e903a7a860d2bdeb15a8f633c481d52a:64/921844
	3d296bd2328be070f883024896e56844:97/463409
	0b49323a10e2aacf29b199129bea6c47:69/469055
	f31723284ba31d282c8b3d61b7dd640c:80/933197

Table 3.2: An example row of the directed graph database.

Table 3.2 shows a directed graph entry for a single web page. MD5 checksums are used rather than URLs because they provide a unique way of describing a page; URLs are non-unique.[9] The outbound field contains a list of web page checksums and local storage locations that are pointed to by the selected page (two, in this case); the inbound field contains a list of web pages that point in to the selected page (four, in this case).

[9] The reason we use URLs to identify web pages rather than MD5 checksums is that URLs are easy to remember and remain constant when the HTML content of a web page changes; MD5 involves cryptic 32 byte hexadecimal strings that change every time a web page is updated.

Image grabbing

Graphic versions of web pages were captured using a Netscape version 4.0 web browser on a Microsoft Windows NT platform. This combination of browser and platform was selected because it could accurately render most of the visual aspects of the majority of web pages. In particular, only this version of Netscape supports printing of background colors and images—two visual elements that add distinction to a web page.

Because the image grabbing takes place on a Windows NT platform while the rest of the process is performed under UNIX, it is impossible to do all processing within a single program. Instead, multiple programs must work in concert. An image work server was created to delegate the work to separate image grabbing and image conversion clients. SQL could have been used for this task but was not because the mSQL implementation does not permit requests for a single record, and therefore would have been considerably less efficient.

The image work server's protocol is simple: clients connect and announce their specialty; the server responds with a particular URL to grab or image convert. This scheme facilitates parallelism both in image grabbing—a network intensive task, and image conversion—a computationally intensive task. In the initial version, it was impossible to include these tasks within the web spider because of the platform requirements. It is probably best to keep them asynchronous because the grabbing and conversion is a much slower process than discovery and the HTML fetching.

In the immediate future it should be possible to combine all tasks into one process, and execute them on a single machine, as Netscape has recently announced plans to release their browser source code to general public. This would allow total automation on a single architecture.

Image conversion

After web pages are grabbed and saved in PostScript format, an image conversion client converts the PostScript to the MIT Media Lab scale free image format. Details regarding this format are described in detail in Chapter 2. At the same time the images are also converted and saved both as 96 x 96 pixel GIF format thumbnails and as 48 x 48 pixel thumbnails. These thumbnails are particularly useful for rapid prototyping layout algorithms and experimenting with other visual ideas. The entire process for a single image, shown in Figure 3.3, takes about 30 seconds with the current software.

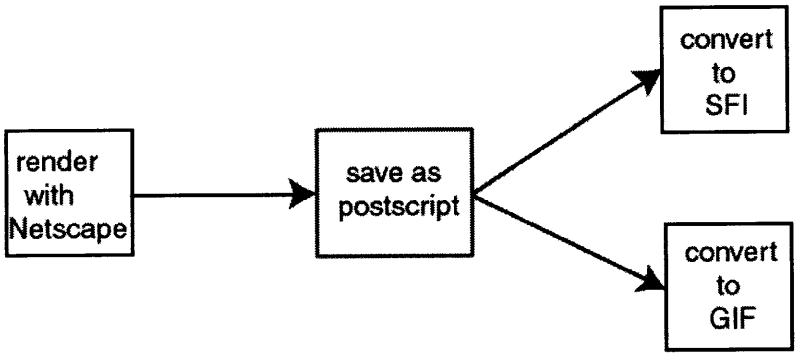

Figure 3.3: Image conversion process.

Updating

The spider is written in Perl and is executed from a UNIX command line. An *update* option allows the tool to the run in update mode after an initial traversal has been completed. In this mode, the robot only asks the remote server for a time stamp of each page. It only requests full pages that have changed since they were last accessed, and only changed pages are marked for image grabbing and conversion. All other pages are ignored, making subsequent updating traversals are much faster.

Administrator interface

An administrator interface to the robot is accessible via the World Wide Web. It provides simple visualization information of the traversal progress. A depth-oriented view, shown in Figure 3.4, breaks down the URLs in the database by depth from the root of their respective servers.

A status-oriented view displays the number of URLs for each HTTP status code encountered. This view is shown in Figure 3.5, and a complete list of these codes is included in Table 4.2.

Finally, as illustrated in Figure 3.6, the database is broken down by the internal status flag value. This illustrates how much of the collection has reached (and how many errors have been encountered in) each stage of the discovery, image capture, and conversion process. In each report, the quantities are hyperlinked to lists of database entries that satisfy a particular criterion. For example, the URLs discovered at any requested depth

can be listed by selecting the appropriate hyperlink. Figure 3.7 shows an example of such a report for a depth of two.

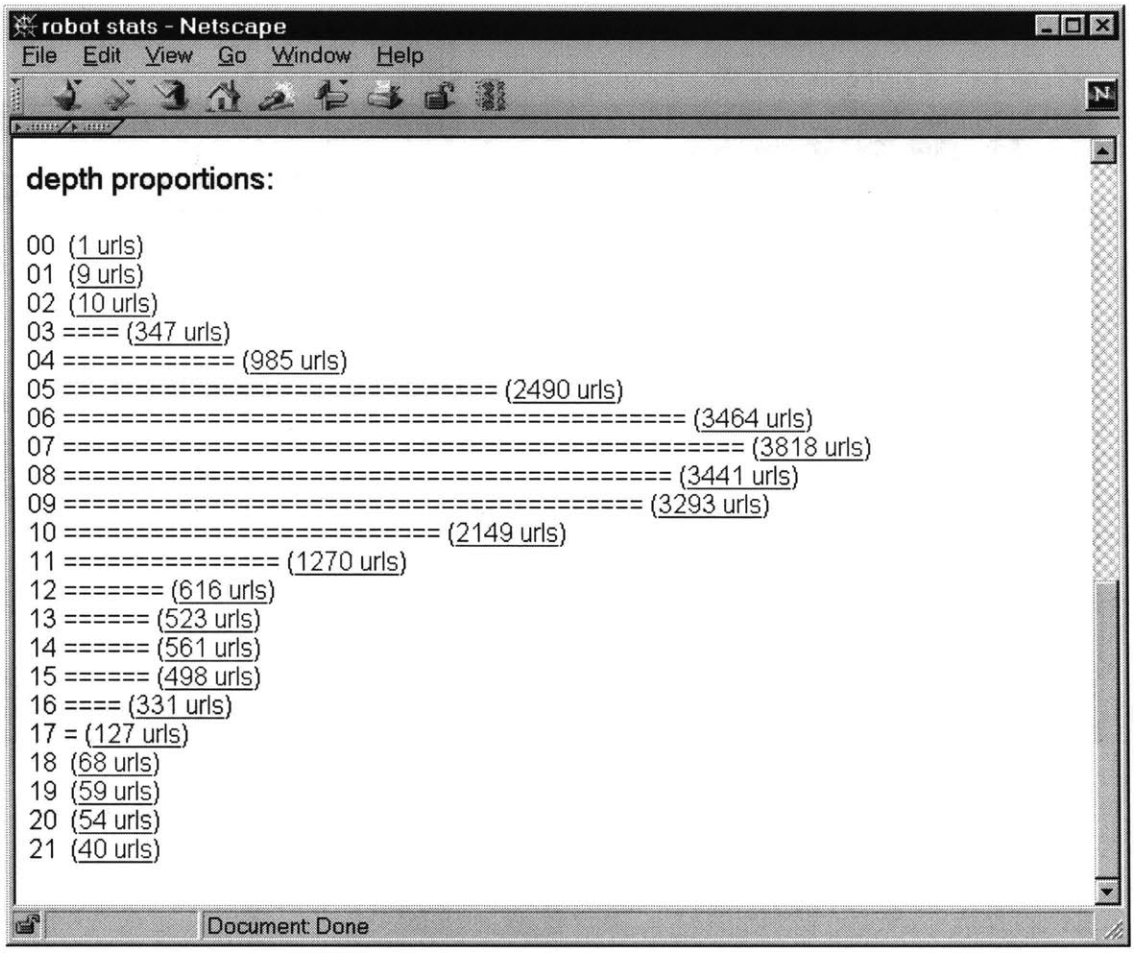

Figure 3.4: Administrator interface view of URL frequency, arranged by hyperlink depth.

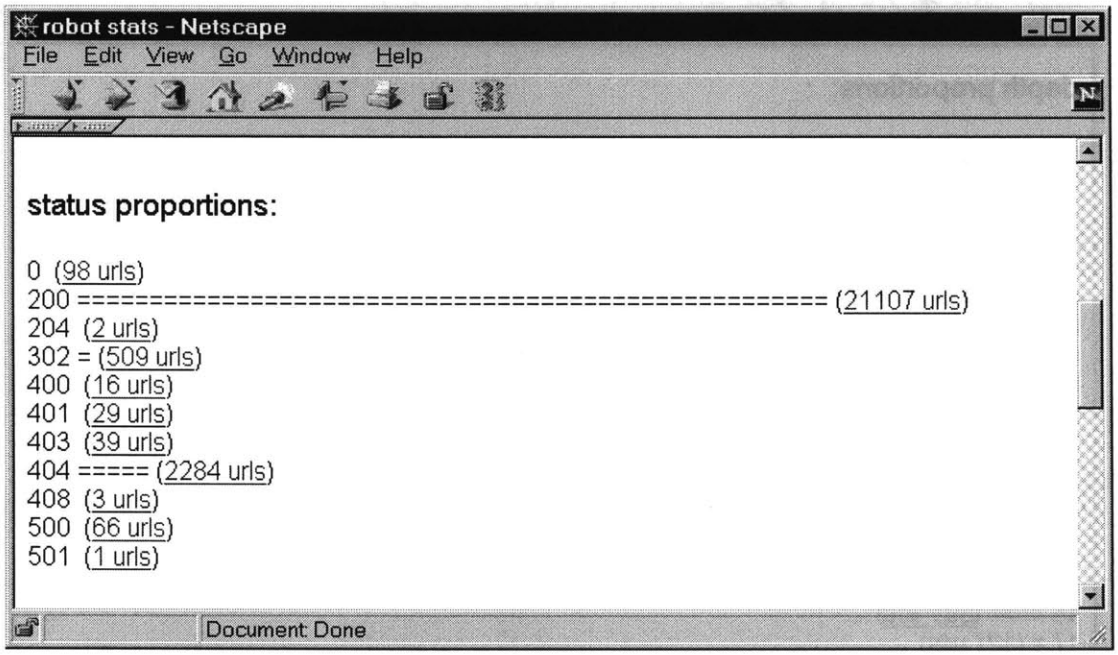

Figure 3.5: Administrator interface view of URL frequency, arranged by HTTP status code.

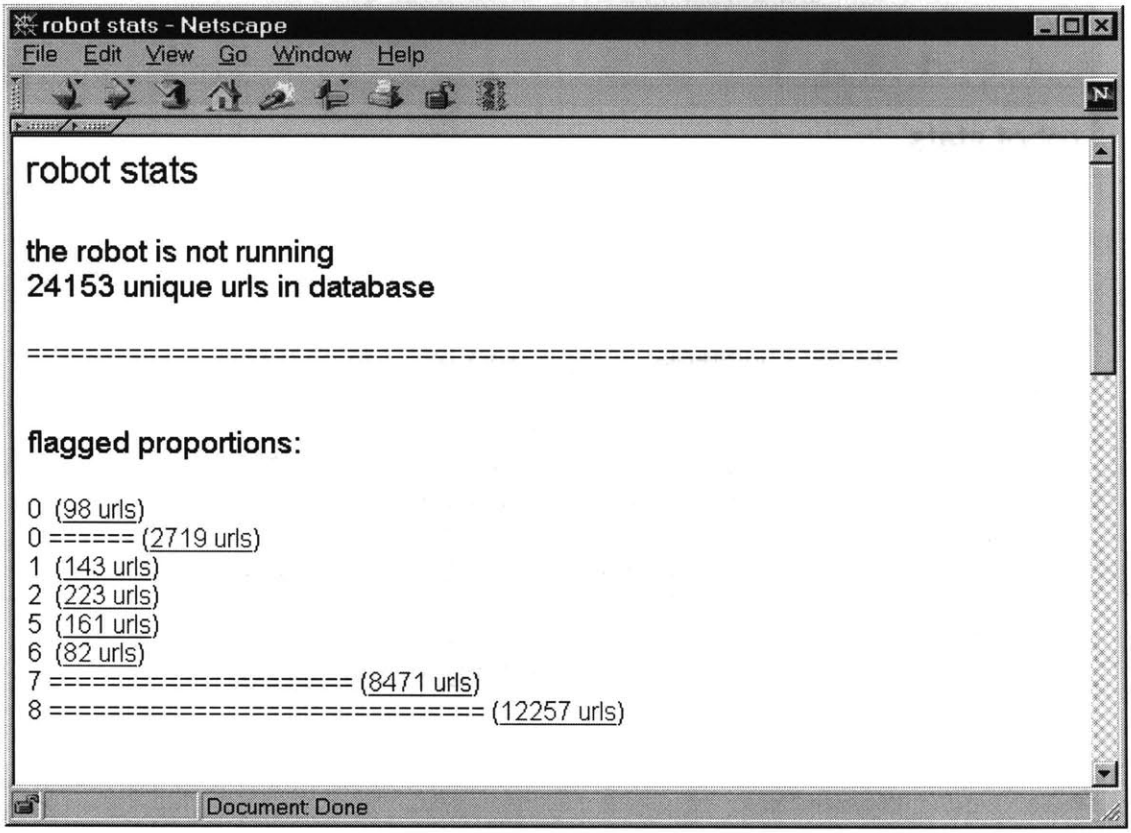

Figure 3.6 Administrator interface view of URL frequency, arranged by internal status code.

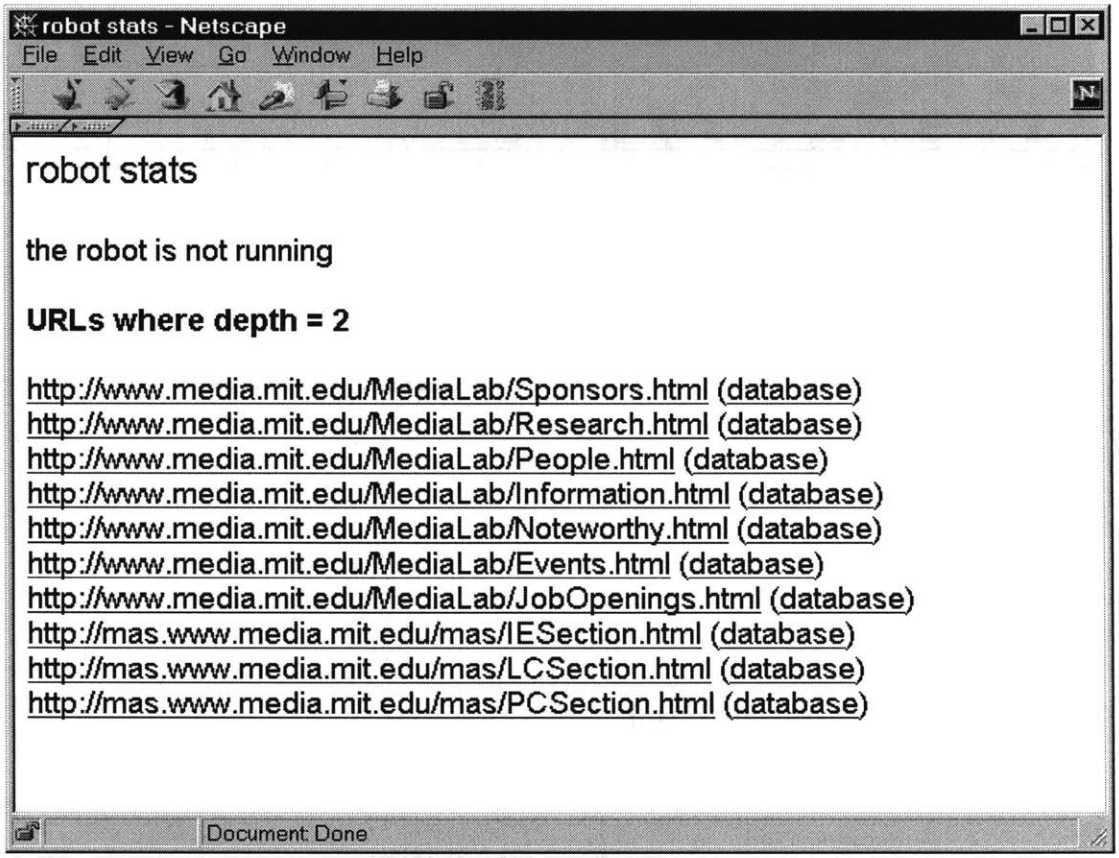

Figure 3.7: Administrator Interface view of request for URLs with depth 2.

4 Search Engine and Corpus

Once the spider has gathered an archive of web content, an effective means of searching is required. This chapter outlines the features and modifications made to the search engine, and provides descriptive information about the MIT Media Lab corpus used in the scale free information retrieval project. The Media Lab corpus is a dynamic collection that is constantly evolving. For this reason, the spider is continually updating the index, and the numerical figures reported in this chapter represent a snapshot taken in March, 1998.

Search engine: SWISH-E

SWISH-E was selected as the keyword search engine from among the several free search engines available [Swish98]. The most attractive features of SWISH-E are the availability of source code and its relative performance compared to other options. The freely available software supports basic keyword queries and also allows the use of the Boolean operators: 'and', 'or', and 'not'. The SWISH-E system also includes features specifically targeted for HTML content. For example, queries may potentially specify fields such as 'title' or 'meta'.

SWISH is an acronym for Simple WAIS Indexing System for Humans. SWISH-E stands for SWISH-Enhanced, an improved version of SWISH made available by the Berkeley Digital Library project.[10] Search result rankings in the SWISH system are computed using the vector space model, as described in Chapter 2. Keyword searching within the scale free search engine is based upon SWISH-E with additional modifications.

Modifications

Several modifications were made to the SWISH software in order to enable features such as phrase searching, duplicate elimination, recording of document titles, image locations, original URLs and discovery dates.

Phrase searching was implemented by adding a second automated stage to the search process. After the search engine processes an initial Boolean 'and' query, the resulting documents are scanned with Perl's Boyer-Moore string processing algorithm to check for phrases. While not the most efficient solution, it is adequate because phrase searching is not the default option and the expected number of users who will take advantage of this phrase feature is relatively low.

Additional fields were added to the search engine so that search results could be displayed in a format similar to that used by the familiar major search services. These fields include, a document title, date last indexed, MD5 checksum, and URL. The only

[10] The 'E' in SWISH-e refers to enhancements made by the SWISH authors before it was further modified for this thesis project.

exception is that there are no document excerpts, summaries, or abstracts in the index, although these could be easily added.

One of the additional fields—the MD5 checksum—allows elimination of duplicate results at query time. This is often a problem in search engines dealing with World Wide Web documents, as it is impossible to tell that documents are identical by URL alone. The MD5 checksum provides a convenient way to verify this.

Corpus: MIT Media Lab Web

unique URLs	23,765
valid URLs (status 200)	20,855
Hostnames used	425
Distinct web servers	85
unique words in corpus	166,817
average document length	4,142 characters
	670 words
Average URL length	65 characters
total hyperlinks	87,024
Average hyperlinks per page	4.3

Table 4.1: Summary statistics for Media Lab collection.

The initial corpus used for this project is the MIT Media Lab's network of web pages. The Media Lab collection is distributed among 85 web servers with a total of 23,765 unique URLs after the initial normalization takes place (Table 4.1 provides a summary of the major statistics)[11]. Of these about 2,242 were not found, 493 were relocated, and 175 others resulted in other errors. Table 4.2 reports the number of URLs resulting in each status message countered.

[11] All of the figures here are current as of March 1998, but are constantly changing as the project is dynamic. The robot continually discovers new pages and adds them to the collection.

Status Code	Message	Number of URLs
200	OK	20,855
204	No Content	2
302	Moved Temporarily	493
400	Bad Request	16
401	Unauthorized	29
403	Forbidden	39
404	Not Found	2,242
408	Request Timeout	3
500	Internal Server Error	66
501	Not Implemented	1

Table 4.2: HTTP status codes and frequency of occurrence.

The average document length was 4,142 characters and had an average of 670 words (this figure includes stopwords, but not HTML markup); the total number of unique words in the corpus is 166,817. No stemming or suffix removal was used.[12] The average URL is about 65 characters. There were 87,024 hyperlinks from one Media Lab document to another—an average of about 4.2 per page.

Because of the multimedia nature of the work done at the Media Lab, this collection was especially appropriate for a visual search engine. Of the 20,855 web pages indexed, the vast majority of them contain one or more substantial visual elements.

As Table 4.3 illustrates, 66 percent contained one or more image tags (<IMG>). 47 percent specify a background color (BGCOLOR attribute of <BODY>), seven percent include background images (BACKGROUND attribute of <BODY>), and 31 percent employ a table (<TABLE>). 84 percent of web pages in the collection use one or more these elements.

Hypertext markup language (HTML) includes a cornucopia of other tags that can affect visual appearance. Two other very common tags are heading (<H>), which appears in 71 percent of the collection, and font (<FONT>), used to indicate a special typeface or type size. Font tags appear in 26 percent of the collection. When all of the previously mentioned tags are considered, 96 percent of the corpus contains important information affecting visual layout. Even the remaining handful of relatively unformatted web pages convey important information—the very fact that the author used no special formatting.

HTML Tag Type	Frequency
Image	66%
Background color	47
Background image	7
Table	31
Heading	71
Font	26
Any of above	96

Table 4.3: Frequency of HTML tags by type.

These measures have not been applied to any other collections yet, but it is not unreasonable to expect comparable frequencies of HTML tags, since many of the same layout idioms are widely used. Even if a collection has far fewer visual tags, a visual retrieval approach could still provide assistance in identifying titles, paragraphs, and other textual structures.

Finally, when we look at the distribution of URLs by hyperlink depth from the initial URL, we see the graph depicted in

[12] Stemming and suffix removal are morphological transformations sometimes used in information retrieval to reduce the total number of terms under consideration. For example, 'layering', 'layered', 'layers,' and 'layer' are all considered the same after being processed by an English language stemmer.

Figure 4.1. These depths represent an upper bound but not necessarily the shortest path; they merely reflect the depth at which the spider happened to discover a URL.

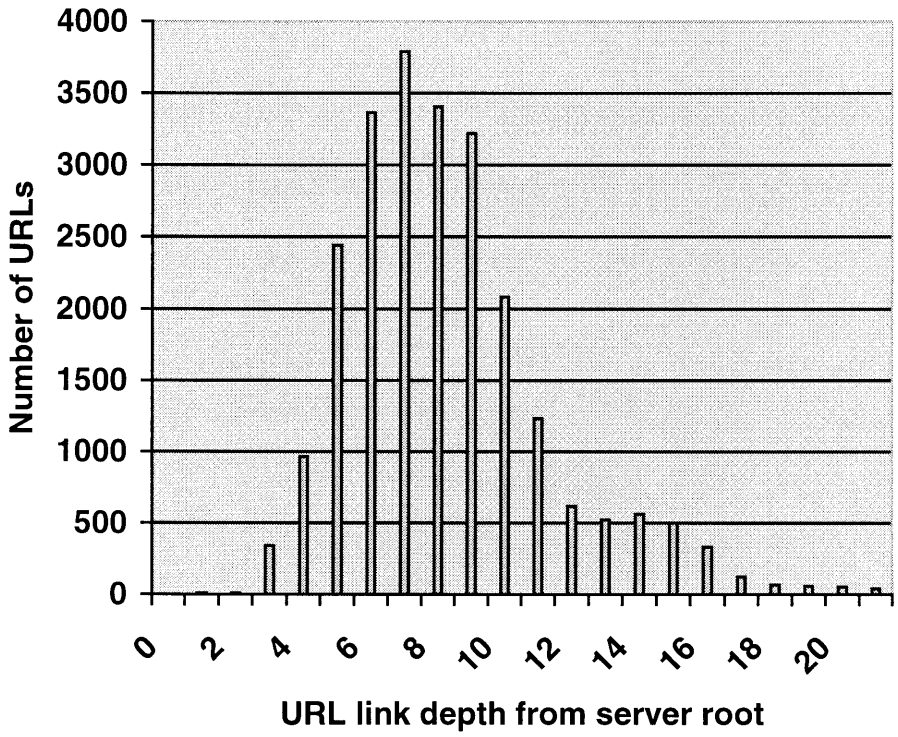

Figure 4.1: Histogram of URL depths from initial page.

Most of the content can be accessed in a dozen or so hyperlinks. The distribution drops off significantly after 17 links, and thereafter has a long tail. Virtually all of the documents in this part of the curve are slide presentations. The tail is truncated because the robot was instructed to ignore depths greater than 21. A high concentration of depths from 5 to 10 suggests that the majority of web content is available within 10 links from a server's home page.

5 *Layout Engine*

The modified SWISH-E search engine described in the previous chapter maps a user's keyword query with a list of potentially relevant web pages. Traditional search engine user interfaces would simply present this list to the user in a textual format. The scale free layout engine enhances the searching process by providing a variety of more efficient and meaningful visual interfaces that incorporate images of the web pages of the search results. This chapter presents the seven experimental visual search result layout approaches that are currently available for users to select from. Chapter 6 offers analysis and evaluation of their effectiveness.

Traditional

The **traditional layout** is a simple textual list of results, similar to what one might experience with most of today's commercial systems. This layout is included as an option, so users have a familiar baseline reference for making comparisons with the alternative visual layouts. An example output of the traditional layout for the query 'digital life' is presented in Figure 5.1.

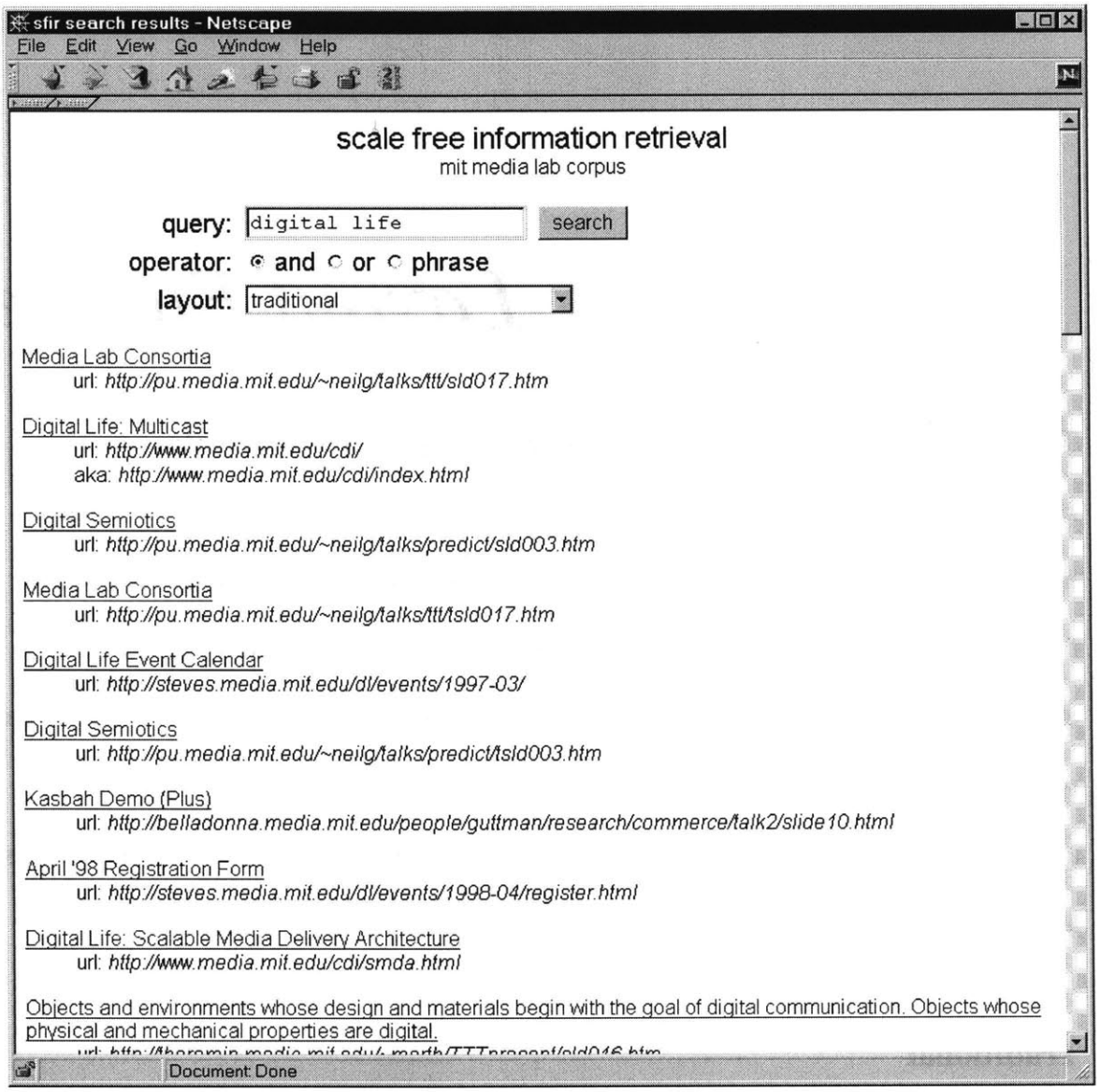

Figure 5.1: An example of results in traditional layout.

Thumbnail annotation

The **thumbnail annotation layout** is similar to the traditional, but with inclusion of a thumbnail image alongside each result in the list. Figure 5.2 illustrates the typical results display for the traditional format. An image size of 96 x 96 pixels is used in this setting. At this resolution, many important document features are easily recognized.

For example, background, font and hyperlink colors, inlined images, paragraphs of text, bullet lists, and tables are often quickly recognized by the user.

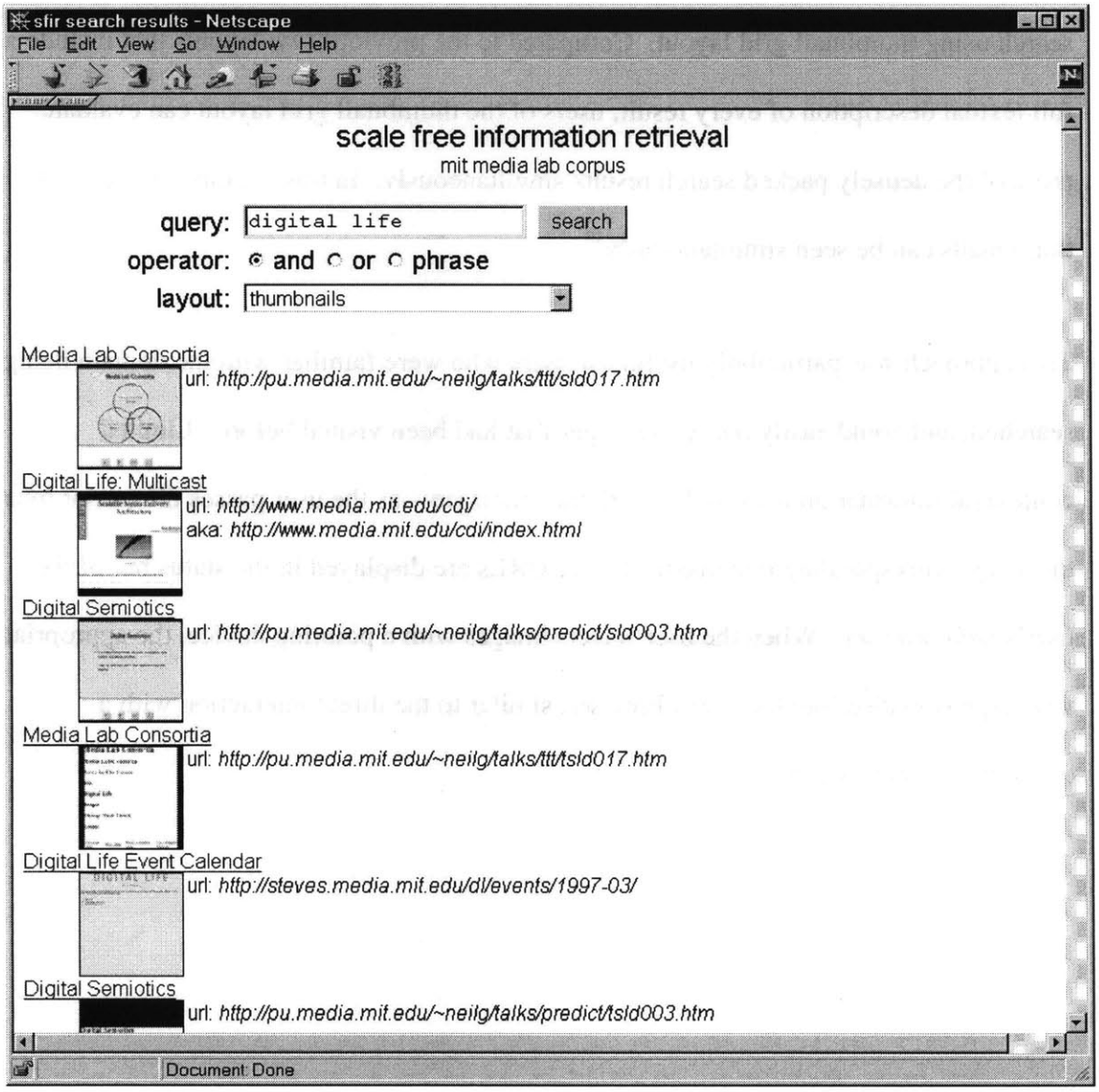

Figure 5.2: An example of results in thumbnail annotation formt.

Thumbnail grid

The thumbnail annotation layout provides the useful addition of images but still requires more or less linear browsing of the results, as no more than five to ten can fit on the

screen at one time, depending on window size. The next layout experiment, **thumbnail grid**, dispenses with textual descriptions and lists entirely, and instead formats the images used in the previous layout as a two-dimensional array. Figure 5.3 depicts the results of a search using thumbnail grid layout. Compared to the previous two layouts that include a full textual description of every result, users of the thumbnail grid layout can evaluate more of the densely packed search results simultaneously. In this example a total of 42 thumbnails can be seen simultaneously[13].

This approach was particularly useful for users who were familiar with the domain being searched, and could easily recognize pages that had been visited before. Limited contextual information is available with this interface—as the user passes the cursor over the array, corresponding document titles or URLs are displayed in the status bar of the user's web browser. When the user selects images with a pointing device, the appropriate web page is loaded into the user's browser, similar to the direct interaction with a traditional search engine.

[13] An additional couple of rows of thumbnails are also accessible to the user via the scrollbar on the right hand side.

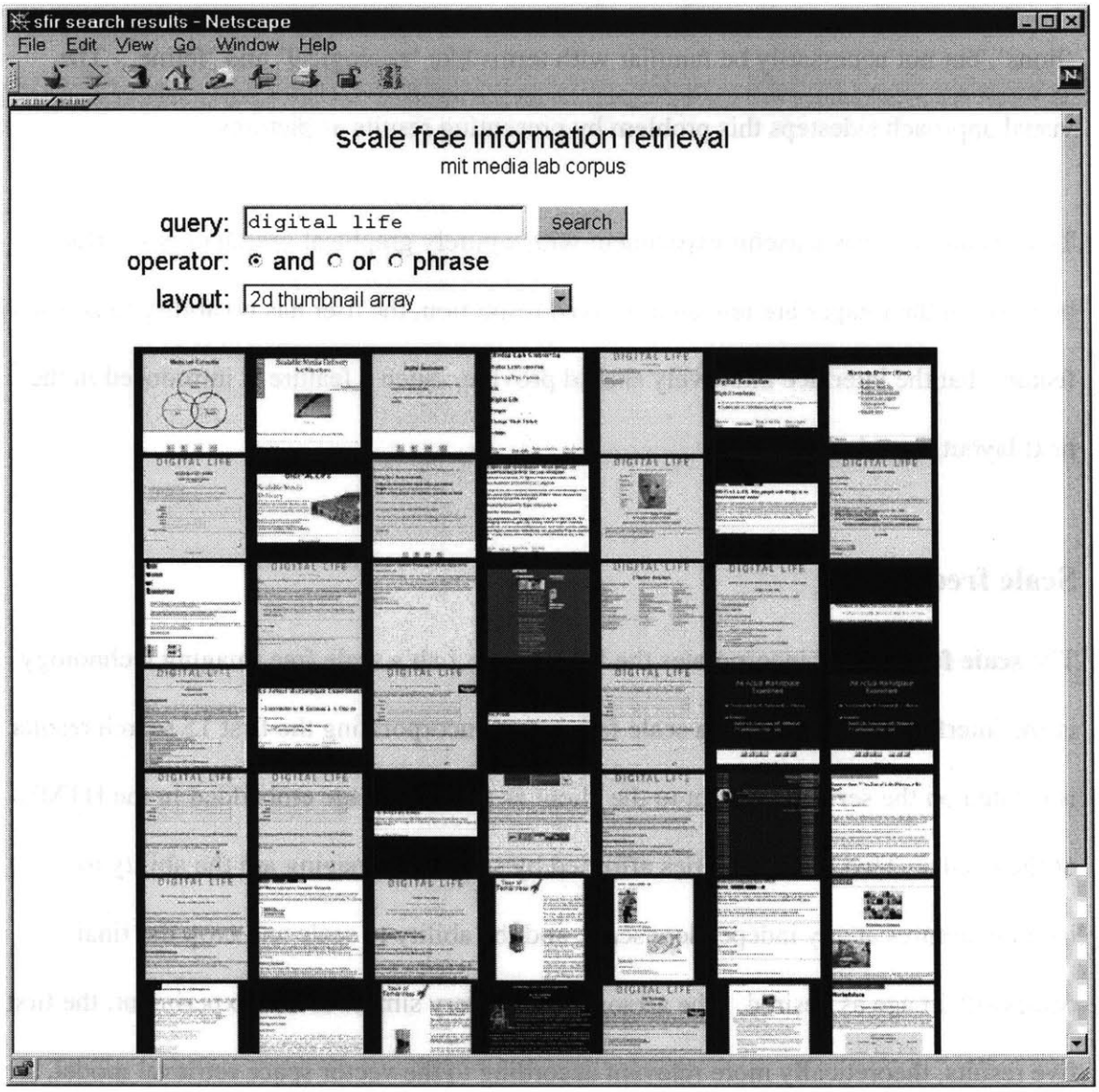

Figure 5.3: An example of results in thumbnail grid layout.

The visual interface is no more difficult to operate than a traditional interface. Some users, such as children, might even find the visual interface substantially easier to use because less linguistic knowledge is required. For example, a user might search for 'lions', but not necessarily be familiar with terms like 'zoological' and 'feline'. The visual approach sidesteps this problem by presenting results as pictures.

Thumbnail grid was a useful experiment with a purely graphical search engine. But because all the images are rendered at fixed resolution, the user has no ability to zoom—a feature that the interface intuitively should provide. Such a feature is introduced in the next layout experiment.

Scale free

The **scale free layout** incorporates the MIT Media Lab's scale free imaging technology in the interface. In this mode, a scale free image incorporating the first 15 search results is created on the server, and sent to the client as a single image embedded in the HTML of the results. Two novel qualities afforded by scale free imaging are the ability to include pictures at any independent scale, and the ability to scale and crop the final composite image as desired. The arrangement is very simple in this experiment: the first five results, theoretically more relevant according to the vector space retrieval model, are presented in larger form in the first row; the second row contains the next 10 results at a somewhat smaller scale. Figure 5.4 shows an example of results displayed in this manner.

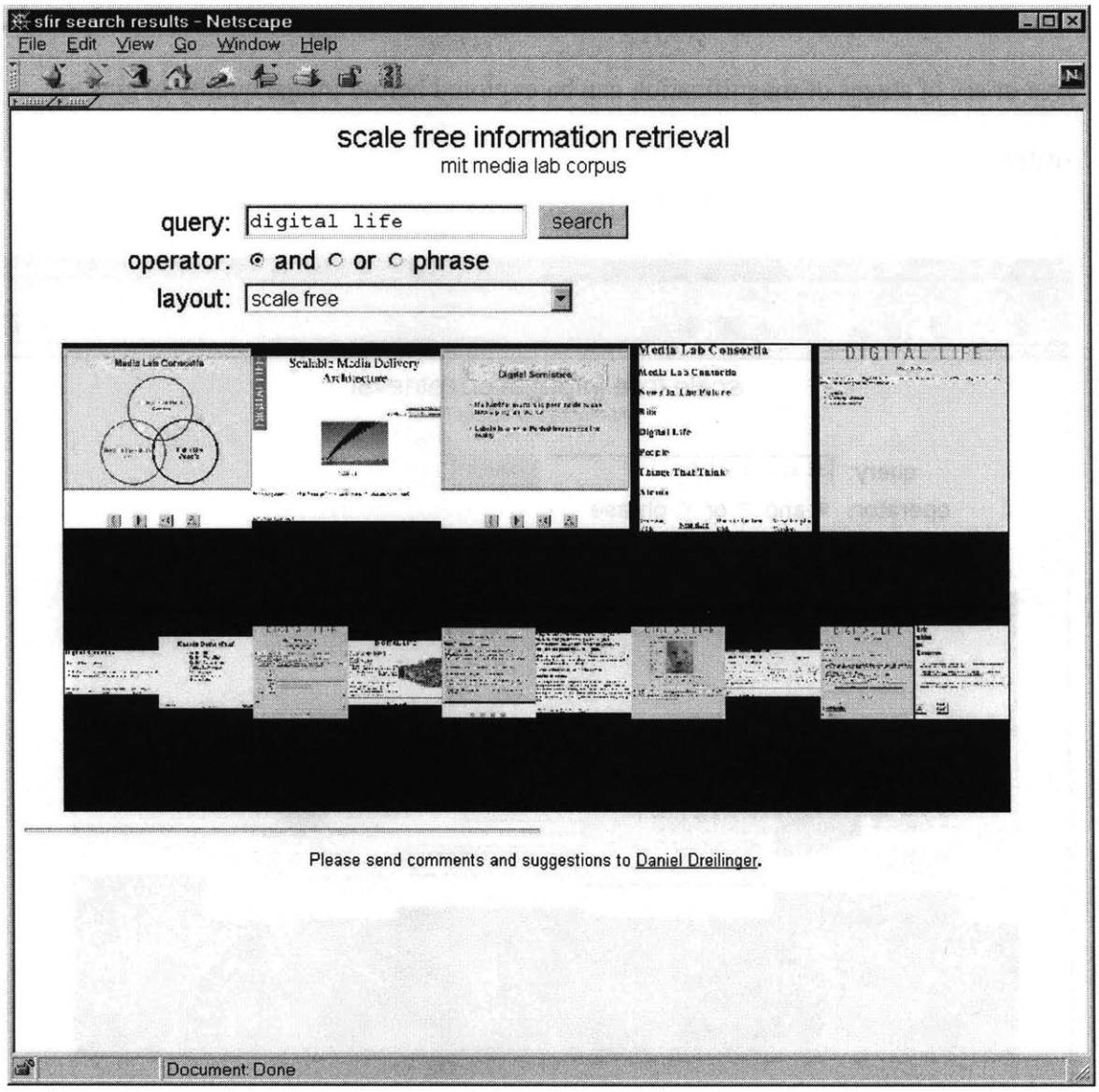

Figure 5.4: An example of results in the earliest scale free layout.

The most striking characteristic of this interface is that the user can point to any part of the image and zoom in for a close-up with a simple click of the pointing device. If a group of adjacent results strikes the user as interesting, it is easy to request a close-up and obtain additional resolution information without visiting web pages individually. Figure 5.5 portrays a close up of the search results shown in Figure 5.4. Since the highest

resolution of web page images stored in the scale free format is approximately life size[14], five or six of stages of magnification can be explored before image quality begins to suffer.

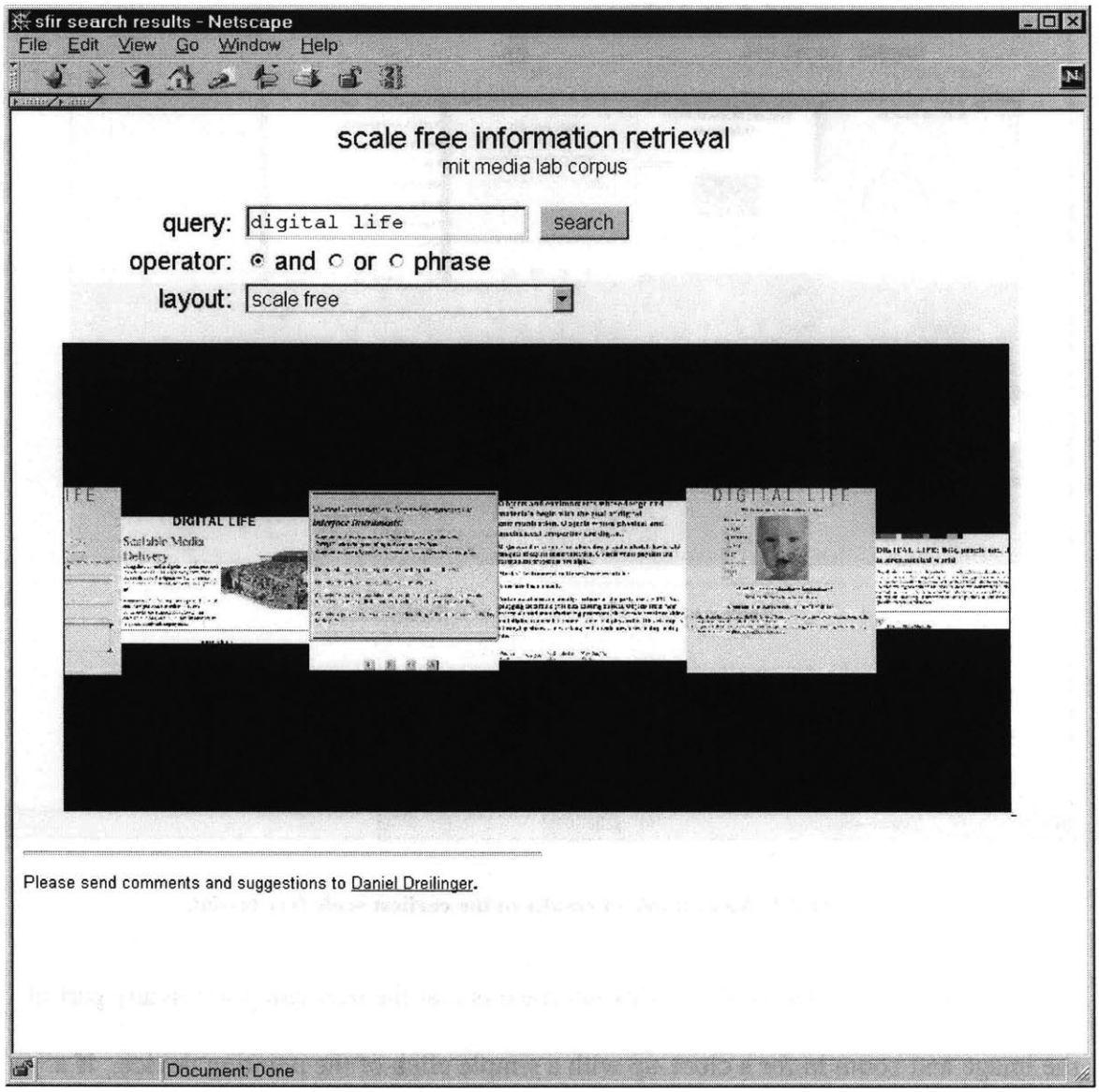

Figure 5.5: A zoomed in view of scale free search results.

[14] Meaning at full zoom, all text is legible and images appear at comparable resolutions to how they would be seen if viewed directly with a web browser. Specifically they are 540 x 540 pixels.

The initial scale free approach worked fairly well as a prototype, but was missing several essential user interface features, such as display of page titles and URLs—and even the ability to visit a URL. The next approach, **scale free grid with dynamic user interface,** introduces these and other features.

Scale free grid with dynamic user interface

Several additional features are included in this interface, making navigation simple and intuitive. First, with the introduction of a small additional popup window, ancillary contextual information is dynamically displayed as the user's cursor passes over parts of the scale free image. This additional window shows a thumbnail image, a page title, and a URL, and potentially could include other information pertaining to specific results.

Figure 5.6 shows the dynamic scale free grid user interface in which results are displayed. Depending on screen geometry, this interface is typically seen with the small window superimposed on the larger one. Figure 5.7 depicts the small window alone. The user is free to move the small window about the screen to any desired location. As the user slides the cursor over the embedded tiles in the scale free image mosaic, the small window automatically displays information pertaining to the tile under the cursor. Figure 5.8 and Figure 5.9 demonstrate two more views of the small window loaded with different web page information.

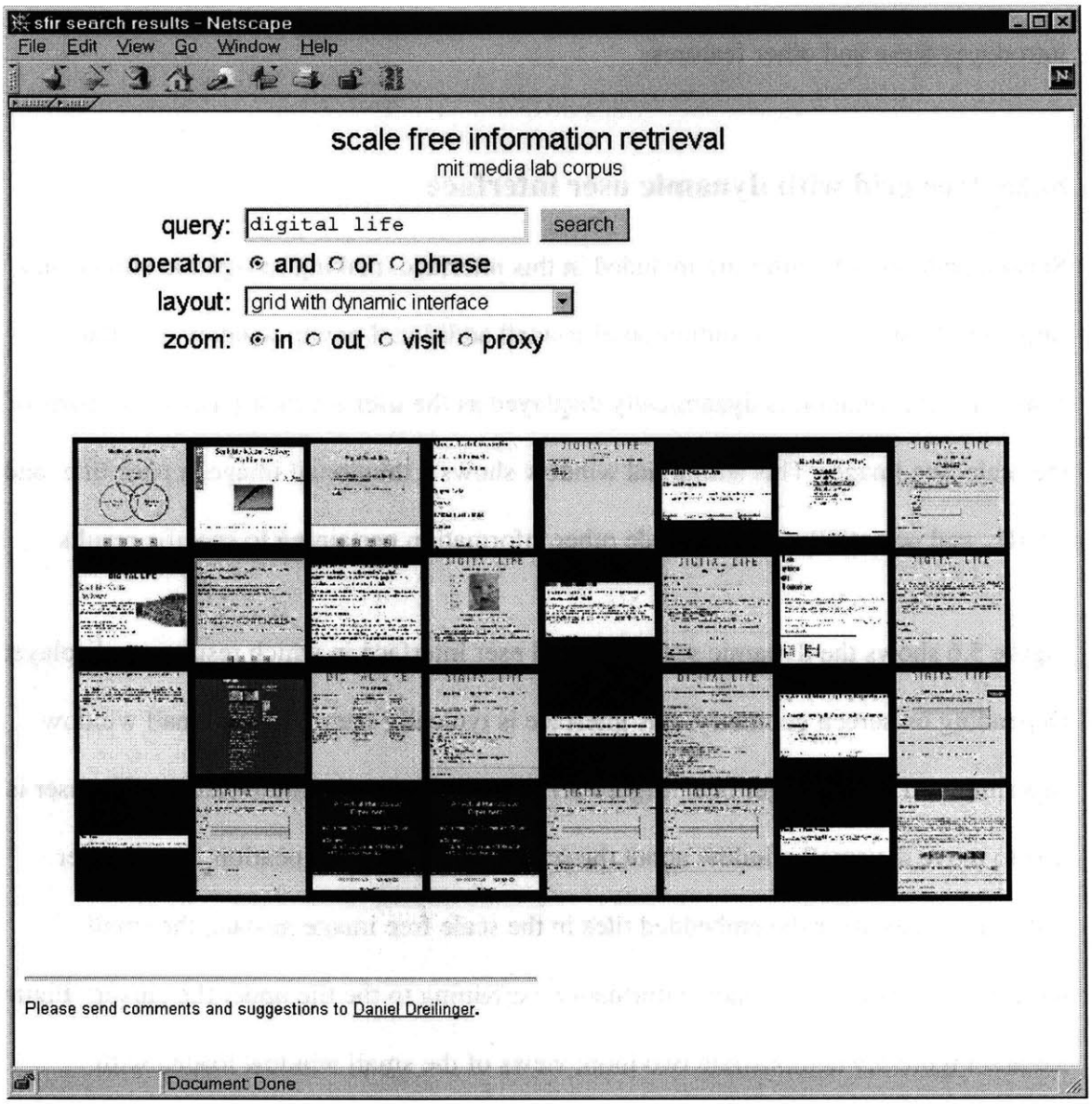

Figure 5.6: An example of results in the dynamic grid interface.

Figure 5.7: Context sensitive window provides additional information.

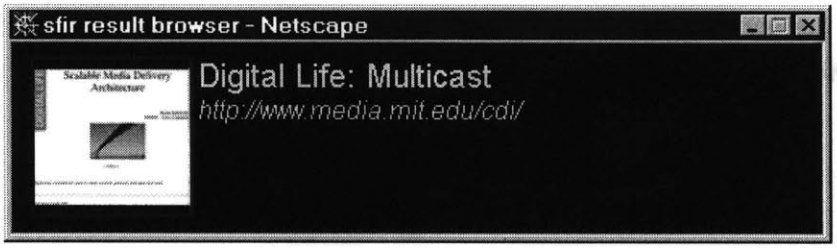

Figure 5.8: Another view of the context sensitive window.

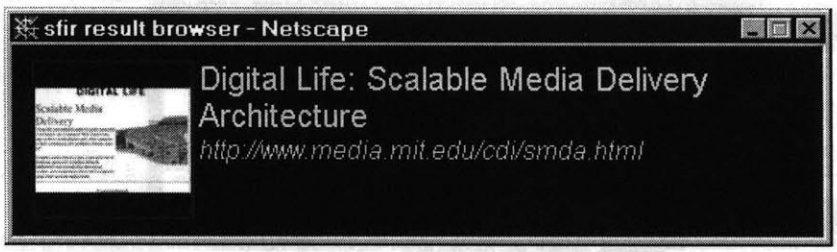

Figure 5.9: Another view of the context sensitive window.

A set of radio buttons, labeled 'in', 'out', 'visit', and 'proxy', affect interpretation of subsequent mouse clicks within the scale free image part of the user interface. When the interface mode is set to 'in', clicking the mouse on the scale free image zooms in and enlarges the part of the image under the cursor. Similarly, clicking on the image while in 'out' mode causes the view to zoom out. When the interface mode is set to 'visit', a mouse click will load the corresponding web page for the underlying tile into the user's browser. The 'proxy' mode, described in detail shortly, is an enhanced version of 'visit'

mode. In a typical search session, the user zooms in to one or more areas of the scale free image before selecting a page to visit. The default mode of the interface is 'in'.

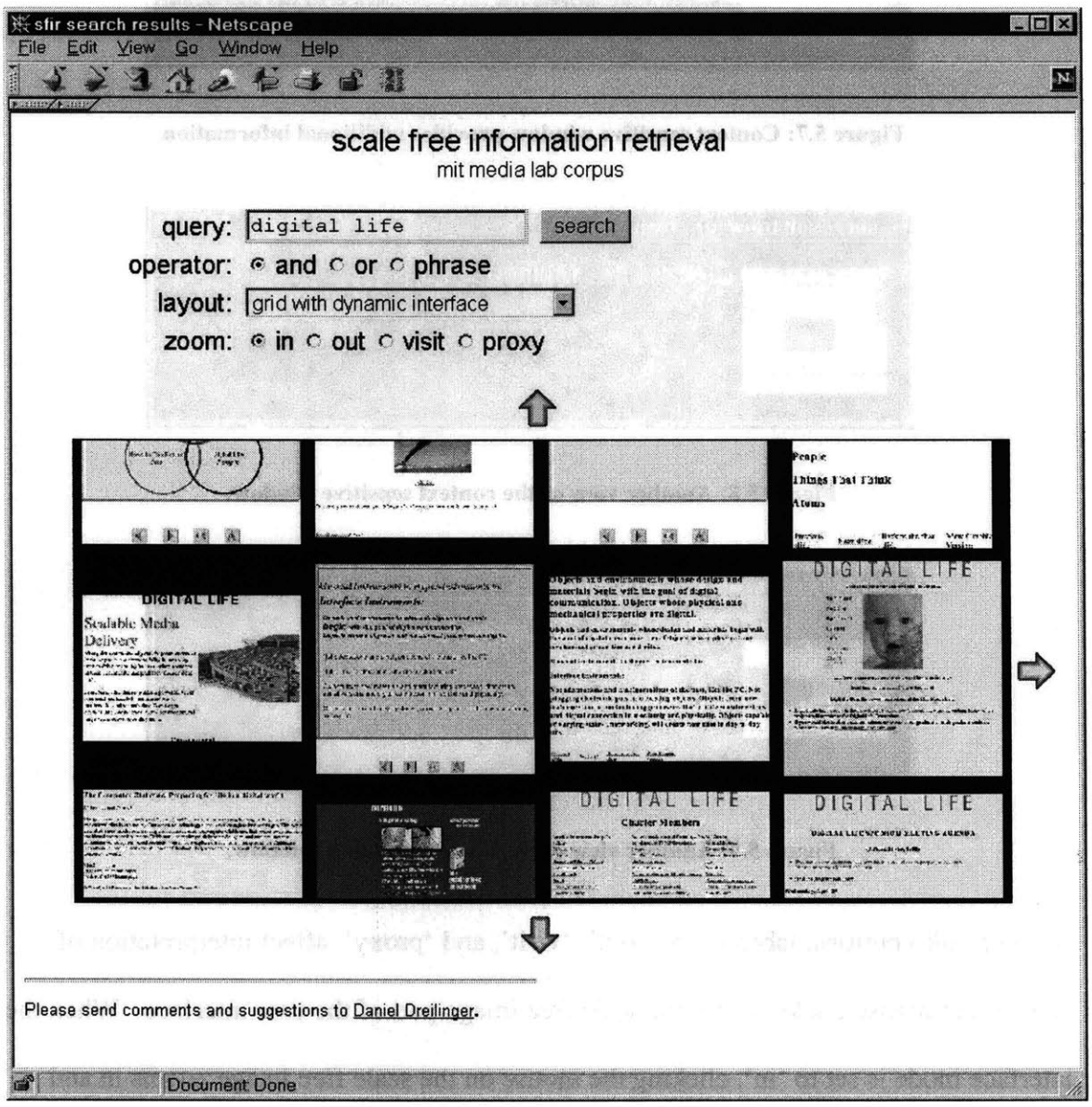

Figure 5.10: Panning controls are enabled when user zooms in.

After the user has zoomed into a region of the results image, a set of panning navigation controls appear. These are the arrows visible in Figure 5.10, which perform as expected when activated—a new image is displayed with the visible portion of the image shifted in

the direction of the selected arrow. When the absolute edge of the main image is reached, the arrow pointing toward that direction is no longer displayed. Because of the scale free nature of the results, a high level of resolution is provided in subsequent close ups, as can be seen in Figure 5.11 and Figure 5.12.

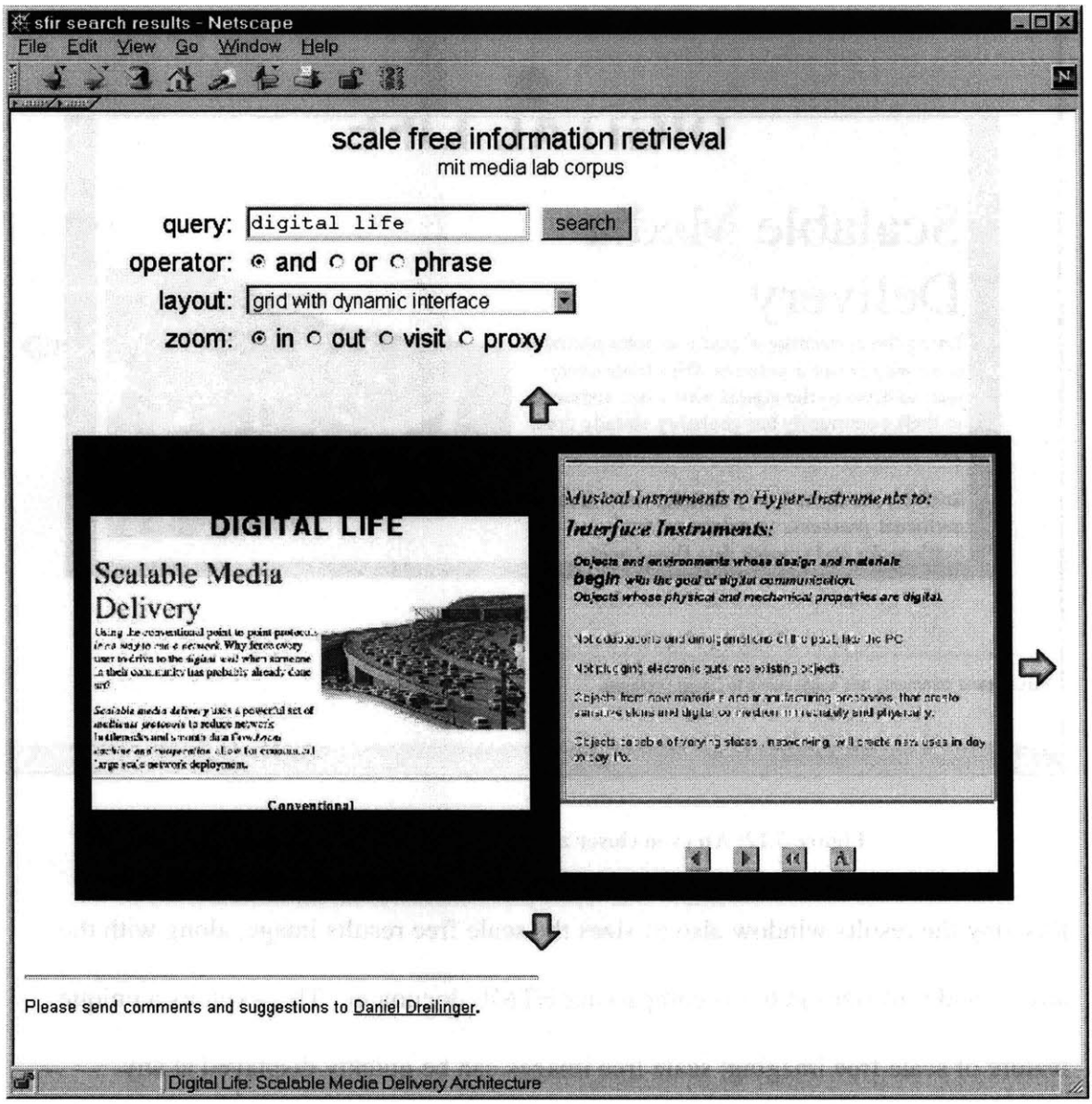

Figure 5.11: Another close up of the grid layout.

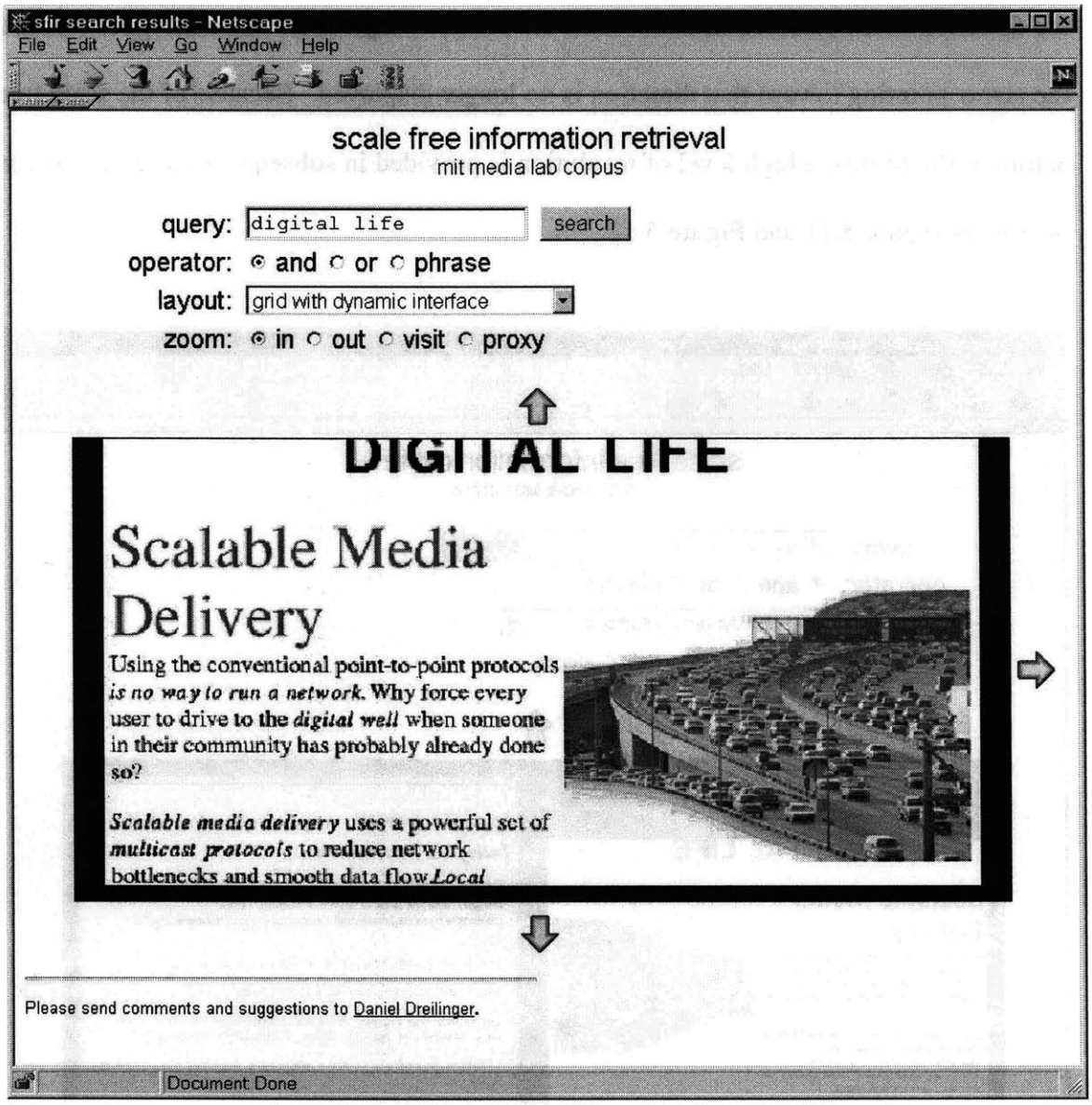

Figure 5.12: An even closer zoom reveals a web page at lifesize.

Resizing the results window also re-sizes the scale free results image, along with the arrows and font sizes in the encompassing HTML document. This exploits a unique feature of scale free imaging: scale free images can be quickly displayed at any resolution. Thus, the scale free concept is extended in the user interface by also scaling the HTML fonts and graphics.

Scale free with emphasis on document similarity

Multiple documents are relevant to a single query but in entirely different contexts. It can therefore be useful to group a set of results into two or more clusters in such a way that intra-cluster document similarity is greater than inter-cluster document similarity. The term 'clustering' has strong visual connotations, and this technique would seem a natural extension of the image based search engine.

While efficient clustering algorithms exist in the theoretical information retrieval literature [Charikar97, Anick97], an ad hoc technique was implemented in the initial prototype. This was done to speed development and try to more quickly evaluate the overall efficacy of visual clustering.

The ad hoc clustering algorithm used in the **document similarity layout** works as follows: evaluate the initial user query against the inverted index using the search engine. Take the highest ranked, unmarked result from head of the list, extract the first N keywords, and evaluate those keywords as a query with the search engine. Create a new cluster containing the document just used, and mark that document in the original result list. Add to the current cluster and mark each of the results of the second query with a score greater than the threshold T that also appeared in the original list. Repeat until there are C clusters or no more unmarked results in the original list.

Various parameter tunings were considered for N, T, and C. In the end, the first 100 keywords were used to create a maximum of six clusters. The threshold T, specific to the SWISH-E search engine, was set to 200 on a scale of 0 to 1000.

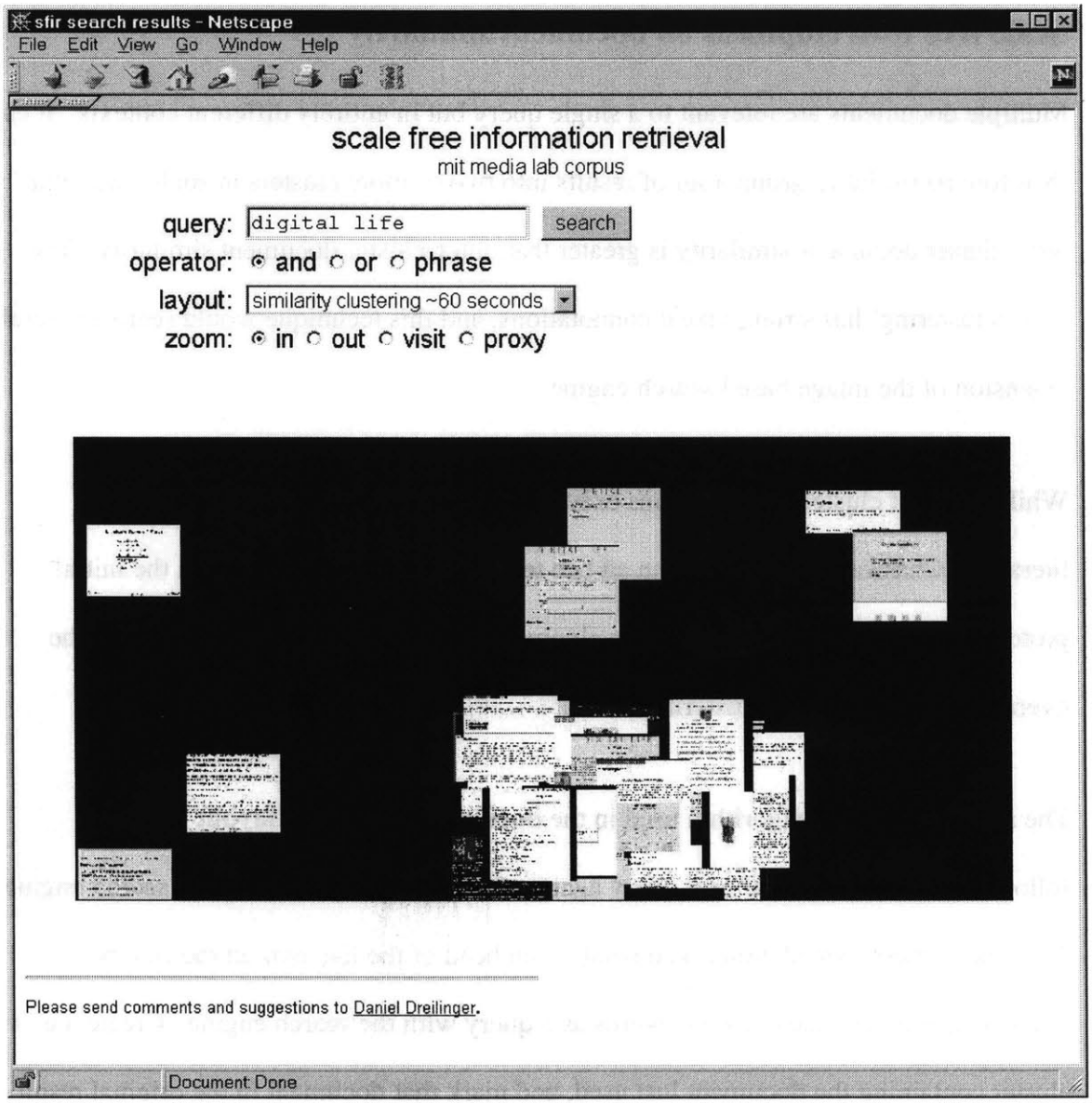

Figure 5.13: An example of results produced by document similarity clustering.

Figure 5.13 demonstrates the results with the Media Lab corpus and the query 'digital life'. In this display, the four small groupings of results are clusters of related documents. The two in the upper right are related to the topic "digital semiotics" and can be seen closer in Figure 5.14. The two in the upper middle relate to a Digital Life conference; a

zoomed in view is seen in Figure 5.15. The large cluster in the bottom center consists of unclustered results.

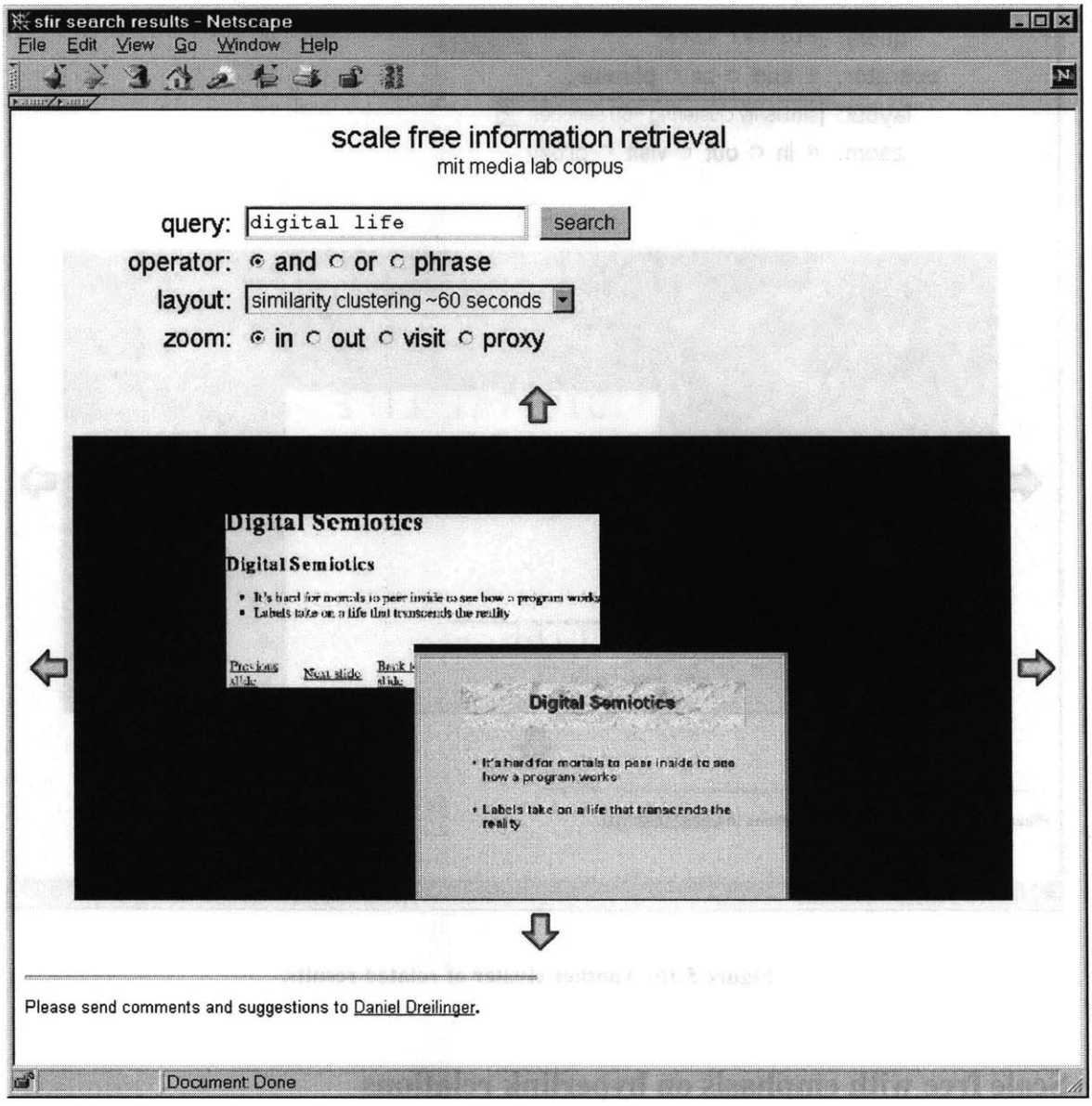

Figure 5.14: A cluster of related documents.

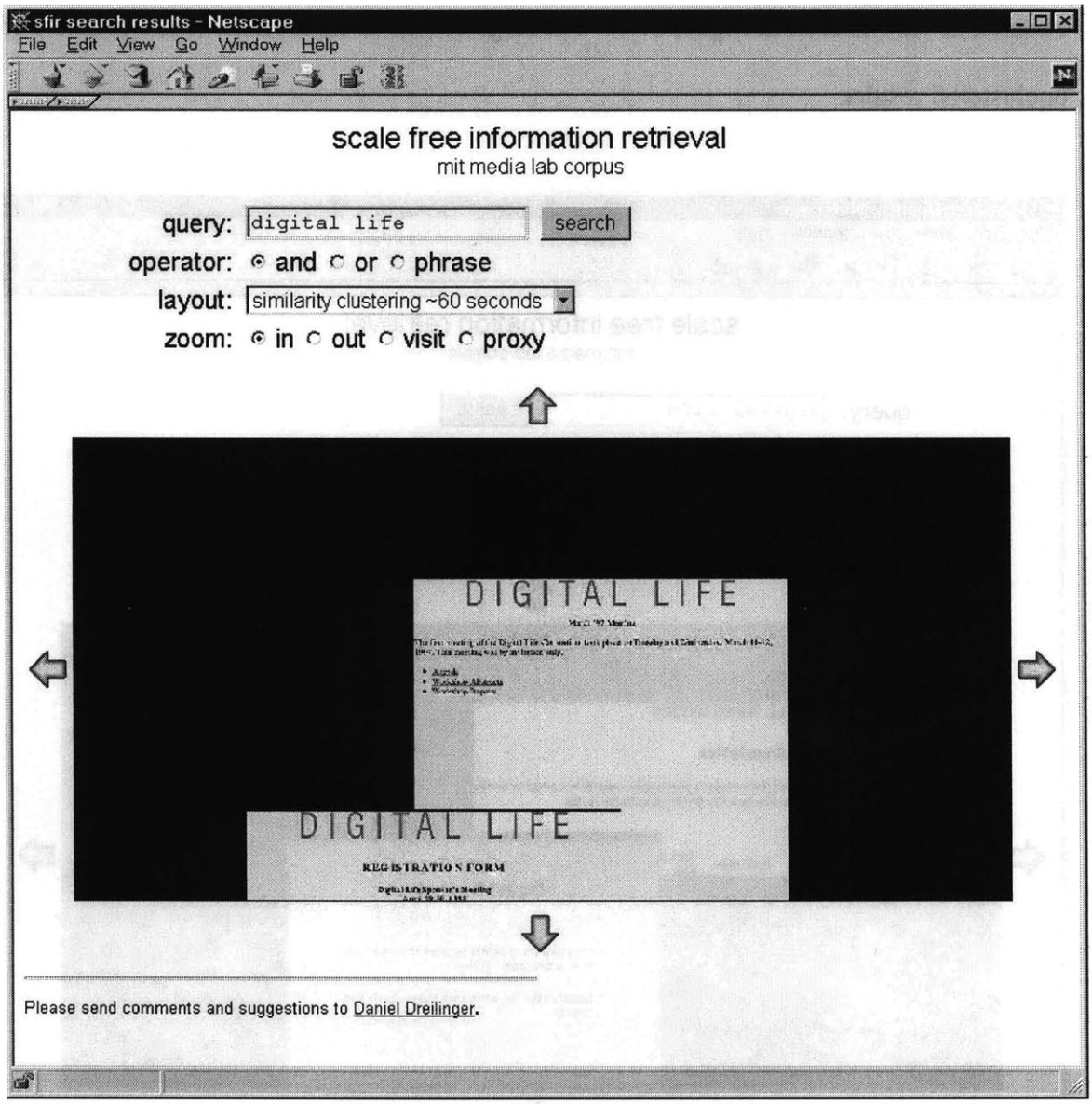

Figure 5.15: Another cluster of related results.

Scale free with emphasis on hyperlink relations

As discussed in Chapter 3, the scale free robot's role is to discover new content and direct its search based on embedded inter-document hyperlinks. Each time a page is gathered, and its outbound hyperlinks extracted, this information is recorded. At the end of a traversal, all of the hyperlink information is converted into a directed graph—a data

structure representing all hyperlinks, both inbound and outbound. This structure provides an inexpensive and fast mechanism for finding all hyperlinks into and out of an arbitrary document.

The **hyperlink clustering layout** accentuates link information in the user interface. There are numerous ways to present these relationships; the one described below was devised, somewhat arbitrarily, to satisfy a number of criteria.

First, the keyword query is submitted to the information retrieval engine, which returns a list of relevant documents. Each returned document is then looked up in the directed graph to determine inbound and outbound links. A new score is assigned to each document, which is simply the total number of hyperlinks—both in and out—connecting it with other documents within the set of results. The highest ranking document under the new scoring is placed in the center of the scale free results image. All hyperlinked documents are placed around it in a roughly circular form. Inbound linking documents are given a slightly smaller radius, while outbound linking documents are given a larger radius. Neighbors with both types of hyperlinks have an intermediate radius. The placement continues recursively—linked documents are placed around those in the initial circle, and so on, until all documents have been placed.

The area of embedded thumbnail images decreases exponentially with each recursive iteration—the image in the center is the largest, while those circling it are about one fourth the size, and so on. Embedded image area is also function of the number of tiles in any given circle, with smaller sizes used when there are more tiles so as to better pack

them without excessive overlapping. Finally, up to four more totally disjoint link clusters of documents are similarly displayed near each of the four corners of the display.

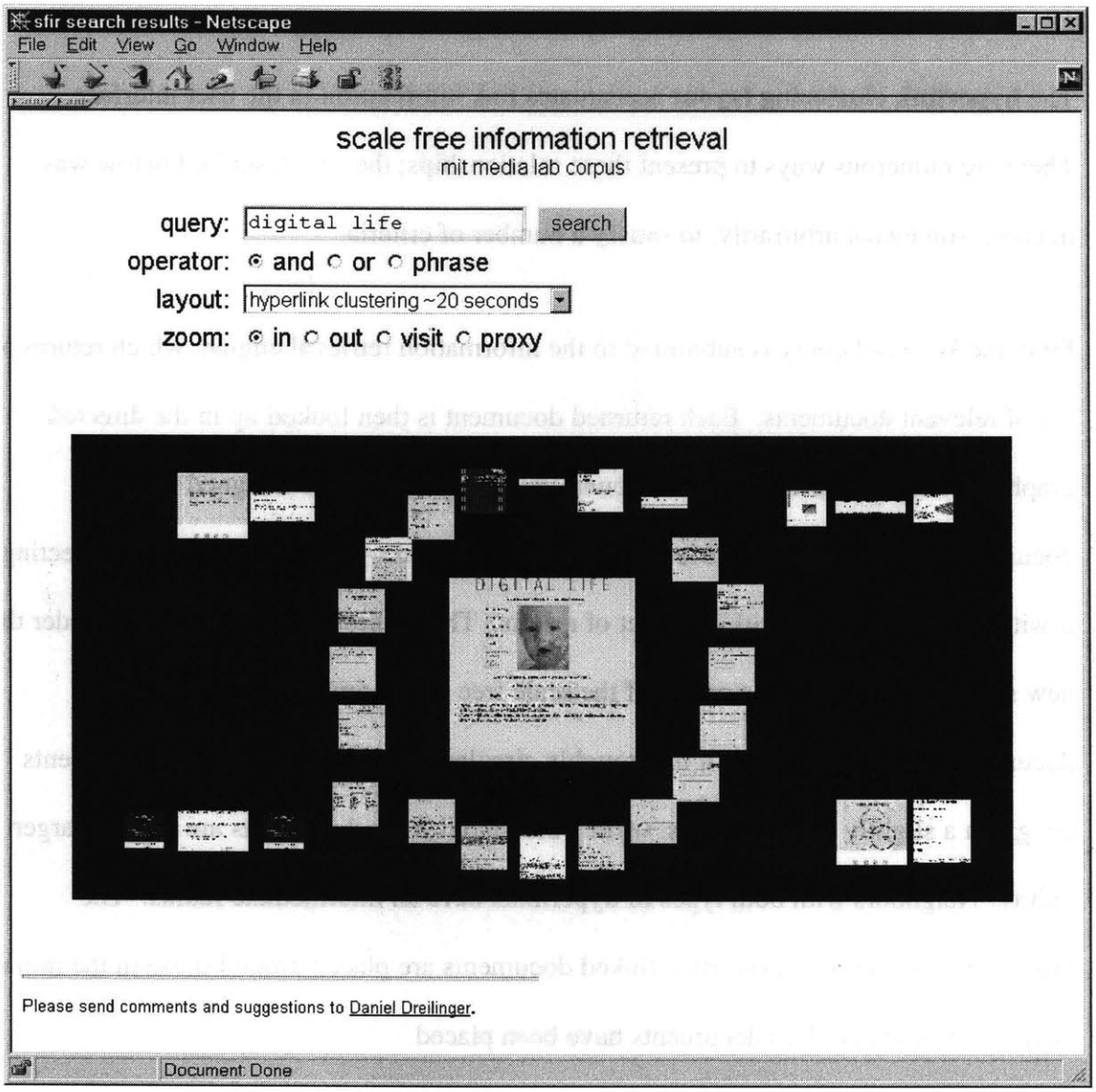

Figure 5.16: An example of results emphasizing hyperlink relations.

Figure 5.16 shows results of the query 'Digital Life' against the MIT Media Lab corpus. The main page for the Digital Life Consortium is placed in the center, because it has the most hyperlinks. All linked documents are placed in a circle around the first document,

with a varying radius used depending on hyperlink direction. Documents pointed to by hyperlinks within the center document have a larger radius (as can be seen at around 8 o'clock in Figure 5.16) while those containing hyperlinks that point into the center have a smaller radius (as can be seen at around 2 o'clock in Figure 5.16). Reciprocal hyperlinks have an intermediate radius. Similarly linked clusters appear in the four corners.

Result re-ranking based on link frequencies has led to improved retrieval in many cases. In the present example, the most relevant or main document about the query topic—the Digital Life home page—is nowhere near the top of the list of results originally returned by the search engine. In fact, this phenomenon often occurred in other queries.

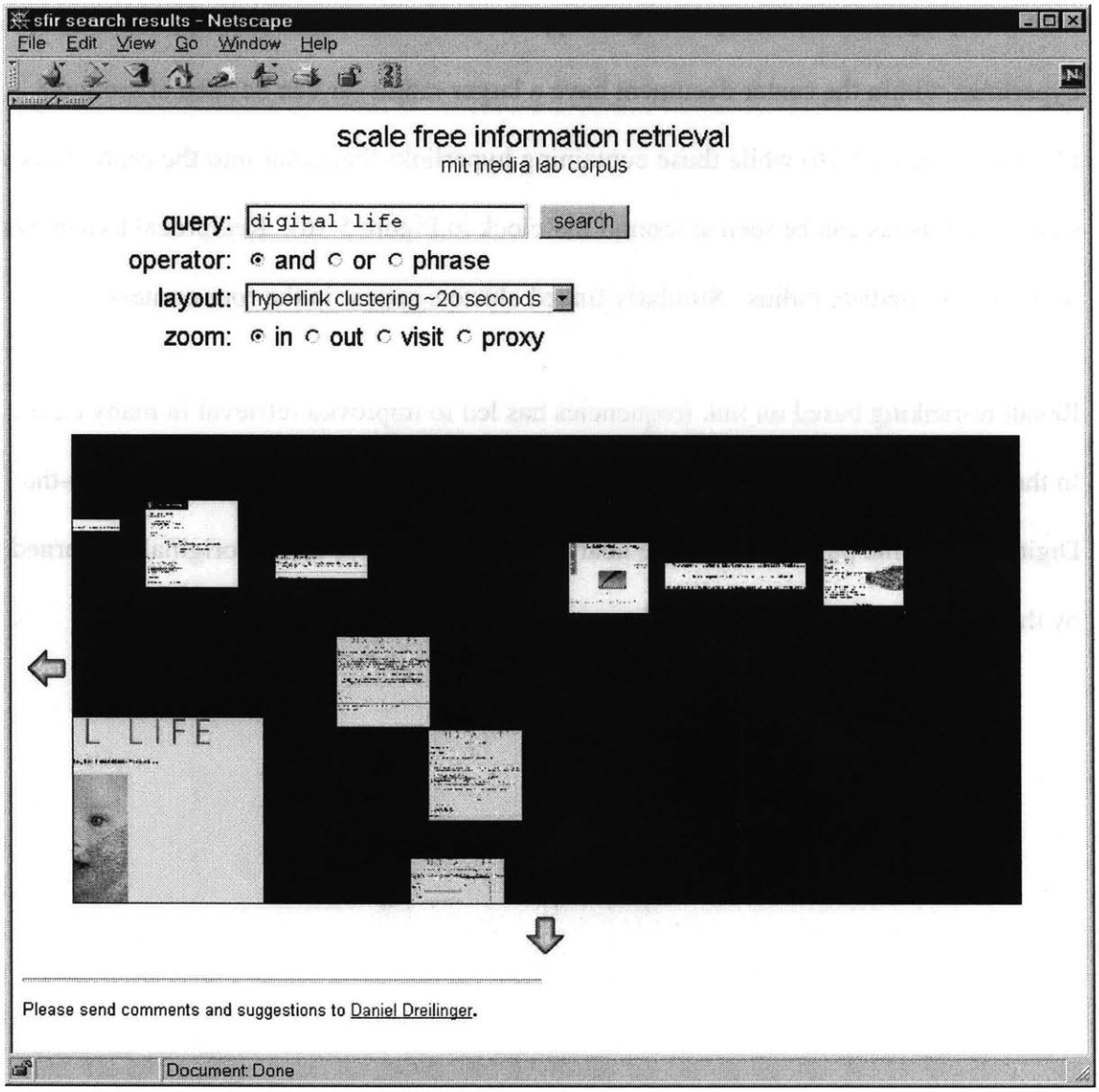

Figure 5.17: A view of the hyperlink interface after the user has zoomed in.

Proxy browser with directed graph annotations

The previous mechanisms addressed the problem of 'searching'—a situation in which the user has a specific information need in mind. The term 'browsing' is loosely applied to situations in which the user does not have specific information need, but rather is exploring a collection with a lesser degree of focus or intent. The **proxy browser with**

directed graph annotations applies the previously introduced visual search paradigm and robot generated imagebase to the activity of browsing.

The proxy browser adds hyperlink information in the form of thumbnail images to the user's browser window in real-time as the user explores the web. Figure 5.18 and Figure 5.19 depict two views of the same web page. The first is a typical view of the original web page; the second has been annotated by the proxy with additional visual hyperlink information. On the left side of the window is a column of thumbnail images of the web pages containing hyperlinks pointing into the current page, with the total count identified above the column. On the right side is a column showing images of pages that are the targets of hyperlinks on the present page. A single line at the top of the window lists the original URL along with options for turning the proxy off and for returning to the scale free search engine.

The two columns of link thumbnails provide the user with a quick summary of the types of hyperlinked pages. The inbound hyperlink identification feature is particularly novel—this information is not readily ascertained from the HTML content of a web page[15]. The proxy quickly produces this information because a directed graph of intra-corpus hyperlinks has been pre-computed. As each image is hyperlinked to the proxy's rendition of the page that it represents, the user continues to get annotated pages as long as they navigate with the thumbnails. A future version will also modify inlined hyperlinks to point back to the proxy, allowing captive annotation.

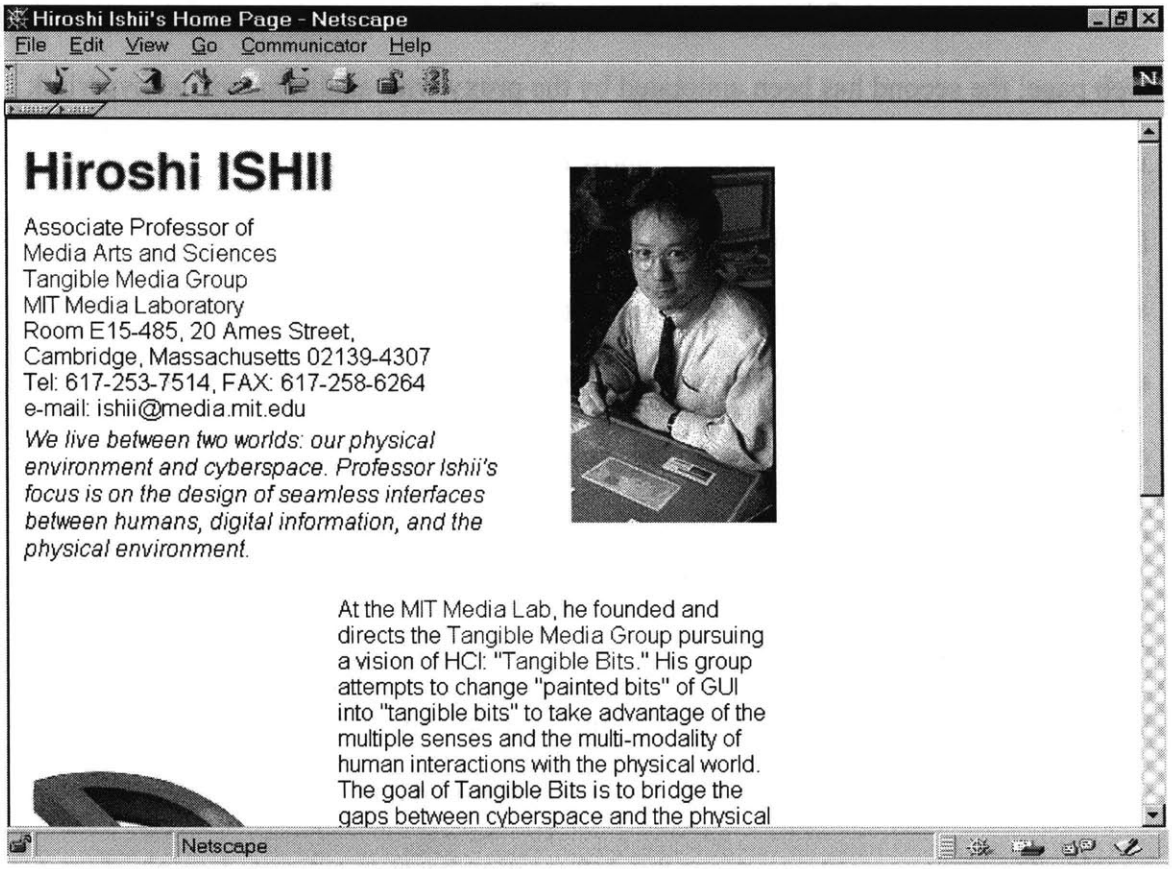

Figure 5.18: Web page viewed without the image annotation proxy.

[15] The World Wide Web Consortium is, however, working on standards for true, bi-directional hyperlinks.

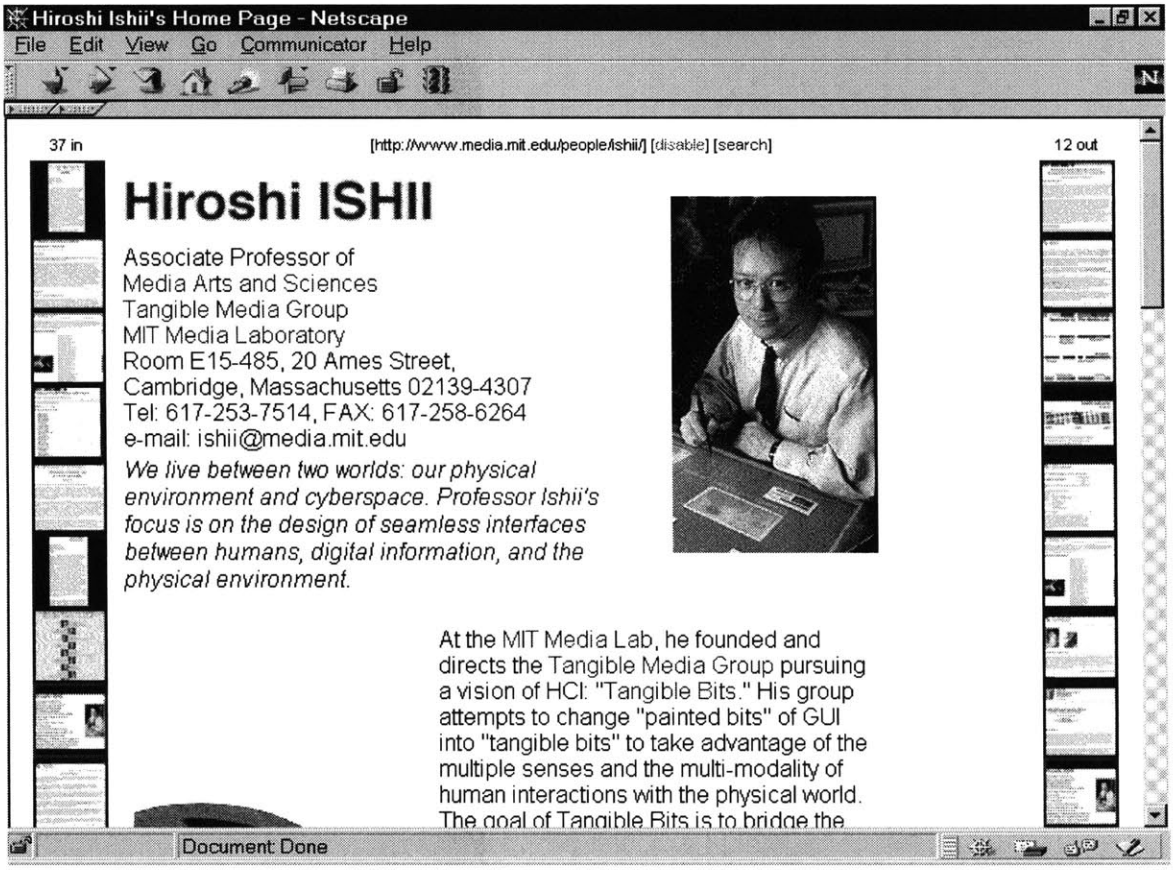

Figure 5.19: Web page viewed with the image annotation proxy.

6 *Evaluation*

This chapter addresses questions about the scale free information retrieval system's effectiveness, usability, and scalability. From an implementation standpoint, the system was successful: all design goals were accommodated. The scale free information retrieval system also showed promise from a usability standpoint—constructive comments and criticisms were received and are reported below.

Is the information retrieval process improved?

The two standard measures employed in traditional information retrieval evaluation, precision and recall, remain effectively unchanged. Recall, the proportion of all relevant documents that the search engine retrieves, is unchanged because the underlying search engine still uses the same ranking criteria. Similarly, precision, the proportion of retrieved documents that also happened to be relevant, does not change.

Empirical evidence suggests that the scale free search engine reduces the overall time required to complete a search. This is due to the fact that a greater number of results are displayed simultaneously and less effort must be spent assimilating textual document tiles and descriptions as must be done with a traditional search engine. Benefits of the visual retrieval system are mitigated slightly when a user's network bandwidth is constrained.

In some cases users can control the bandwidth by requesting smaller images, but then there is a resolution-bandwidth tradeoff.

Is the visual layout effective?

Layout utility depends on a number factors including the user's information need, the size and diversity of the results, and the particular layout scheme selected. A large-scale user study was not conducted, thus it is not possible to offer a definitive answer regarding the effectiveness of the visual layout. Anecdotal evidence from the few dozen users who have tried the system indicates that visual layout of search results could provide a substantial improvement over existing user interfaces. The following section concentrates on the advantages and disadvantages of each of the three advanced layouts and one of the earlier prototypes.

Scale free grid

The scale free grid layout has number of appealing qualities. Users found it easy to understand without instruction, The layout rule is simple and readily apparent: place up to 32 images of equal size in a 4x8 array. This layout has a very high ratio of foreground matter to background space which means images are displayed relatively larger compared to other layouts, and less zooming is necessary. A few simple improvements could be made to enhance this layout. As it is, no attention is paid to adjacency relationships between tiles, so an opportunity for increased information density is foregone. If the tiles were rearranged according to document hyperlinks or document similarity relationships, the overall layout could have been much more meaningful. Zooming would also be more

productive because neighboring tiles are related to one another, and a single zooming action could enlarge multiple tiles of interest.

Document similarity clustering

The relatively poor performance of document similarity clustering can be attributed to the oversimplified algorithm used, and the excessively slow response time. Generally, users were interested in the idea and expected to see clusters of related documents. Because of the visual nature of the system, there was an expectation that *visually* similar items would be clustered together. However, since the clustering approach deals specifically with keywords and clustering centered around statistical keyword similarity, the desired effect was not realized.

In addition to further experimentation with more advanced keyword clustering techniques, it would be interesting to experiment with totally visual clustering approaches. For example, computer vision methods, such as edge detection and color analysis, could be applied to the thumbnails, and similarity made to depend on these measures. The ISODATA clustering algorithm [Jain88] could be used to pre-cluster the collection based on visual features, making query-time clustering based on visual characteristics a quick table lookup.

Alternatively, visual heuristics included in the HTML might be examined. Histograms of the relative frequency of various HTML tags, such as images, hyperlinks, and lists, might be compared instead of keywords. For example, documents with the highest image-to-

text ratio would be grouped in one cluster, while text-only documents are placed in another.

Hyperlink similarity clustering

Hyperlink similarity was well received by the initial users. While some commented that the increase in their zooming lead to overall slower information retrieval, users appreciated being able to see the relevant documents in a recursive display based on hyperlinks. Because relevance was determined based on hyperlink frequencies, there were occasions when retrieval effectiveness was substantially improved. It was often the case that the most relevant query result—the home page for a project or individual—was buried and ranked 20 or below in the results produced by the traditional search engine. When relevance is recomputed as a function of hyperlink frequency, home pages tend to move up in position. Thus, there is strong heuristic evidence that hyperlink relationships are an indicator of relevancy, at least in some queries. Recall the 'digital life' example from Chapter 5. In the traditional results, the Digital Life Home page was far from first, while the high hyperlink factor caused it to be first in the hyperlink based layout.

Figure 6.1 and Figure 6.2 provide another example of this phenomenon. In this case the query is 'movie map'. Within the Media Lab it is generally agreed that this phrase refers to the World Wide Movie Map project. However, due to the relative generality of the two search terms, a multitude of unrelated web pages outrank the Movie Map home page when ranked traditionally (Figure 6.1). In contrast, when results are rearranged according to in and out hyperlink frequency, the Movie Map home page becomes the center of attention (Figure 6.2).

The hyperlink-based layout takes both inbound and outbound hyperlinks into consideration. It may also be useful to visualize the results based solely on one type or the other.

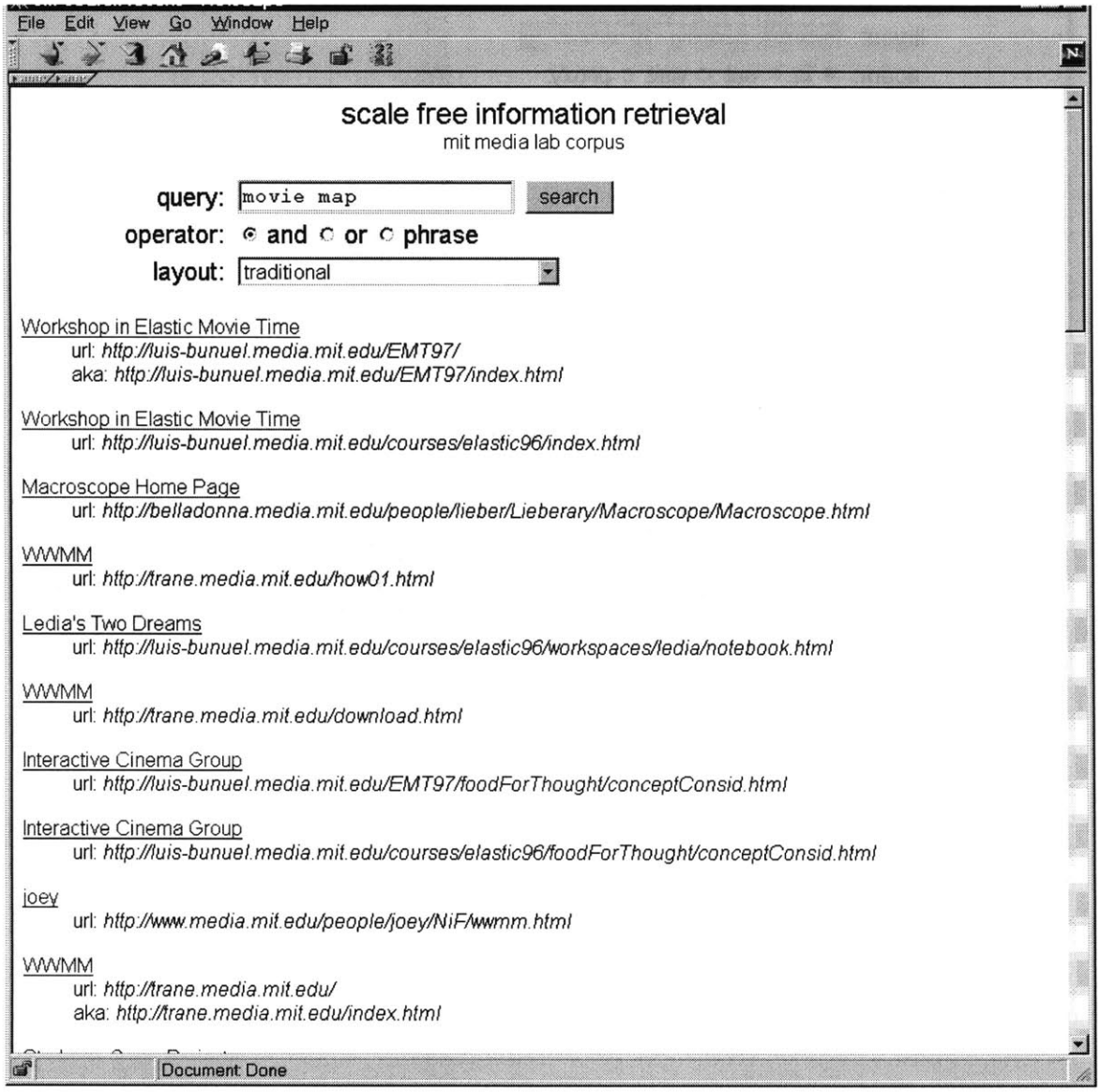

Figure 6.1: Traditional search results for 'movie map'.

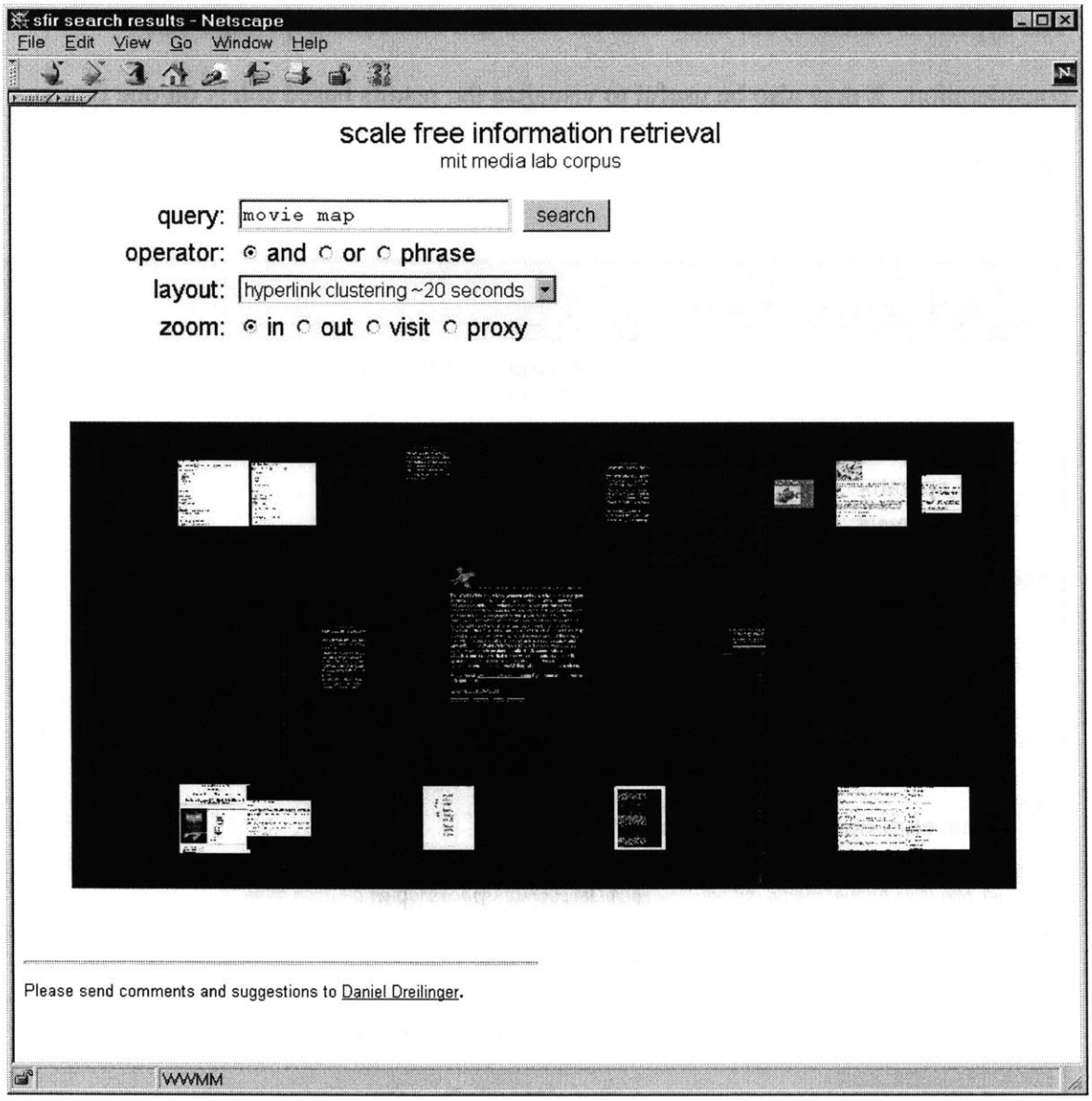

Figure 6.2: Link-based search results for 'movie map'.

Thumbnail annotations

Even though it was one of the most primitive prototypes, users expressed an appreciation for the thumbnail annotations layout. This format consists of the standard textual result list annotated with a small thumbnail and image of each result. The surprising level of interest may be due to familiarity and speed—most users of the Internet and electronic

information retrieval systems are familiar with textual result lists, and are accustomed to receiving them quickly. The thumbnail annotations are a natural interface to many users; no learning or training is necessary because the interface is understood *a priori.*

Thumbnails enjoyed many of the visual benefits described in the preceding sections, but are also associated with some of the drawbacks inherent in long result listings. Thumbnails quickly convey contextual and structural information about the documents that they represent. Limited parallelism is afforded in results interpretation, but only to the number of items that fit a single screen.

Is the proxy browser useful?

The web proxy was very well received among initial users. One of the primary benefits users reported was the navigation assistance provided when browsing a smaller, localized group of inter-link web pages. Regardless of the interface design selected by the original web page author, the proxy adds an intuitive and consistent layer of visual aids.

Many individuals are curious about which pages have links pointing to the page they are currently viewing. The proxy provides a convenient interface to this novel type of information, and in fact, the first thing users did with the proxy was navigate to their own personal home page, anxious to see which pages had links to themselves.

User feedback and general intuition suggest a scalability issue: What happens when you apply this interface to the entire web, not just to a small subset? Won't the displays of in-lined images grow without bound? For extremely popular web pages, yes, there will be

long lists of inbound links. The benefit of the tool in these cases, however, is in indicating the overall *number* (and perhaps suggesting the overall visual nature) of linked pages. Figure 6.3 illustrates this phenomenon with the Media Lab home page—with 1007 inbound links, it is the single most popular page in the collection. In this case, there is no expectation that users will closely examine the left column. The column is truncated at 200 results and not a single user has commented on this fact thus far.

The proxy will continue to offer its previously observed benefit when used with the hundreds of millions of less popular web pages. If necessary, control could be added for toggling display of off-site links.

Another idea suggested by a user was to also capture the fringe of the Media Lab collection—the set of off-site pages that lie just one or two links away from a Media Lab page. This would provide the benefit of being able to show users both internal and off-site page images.

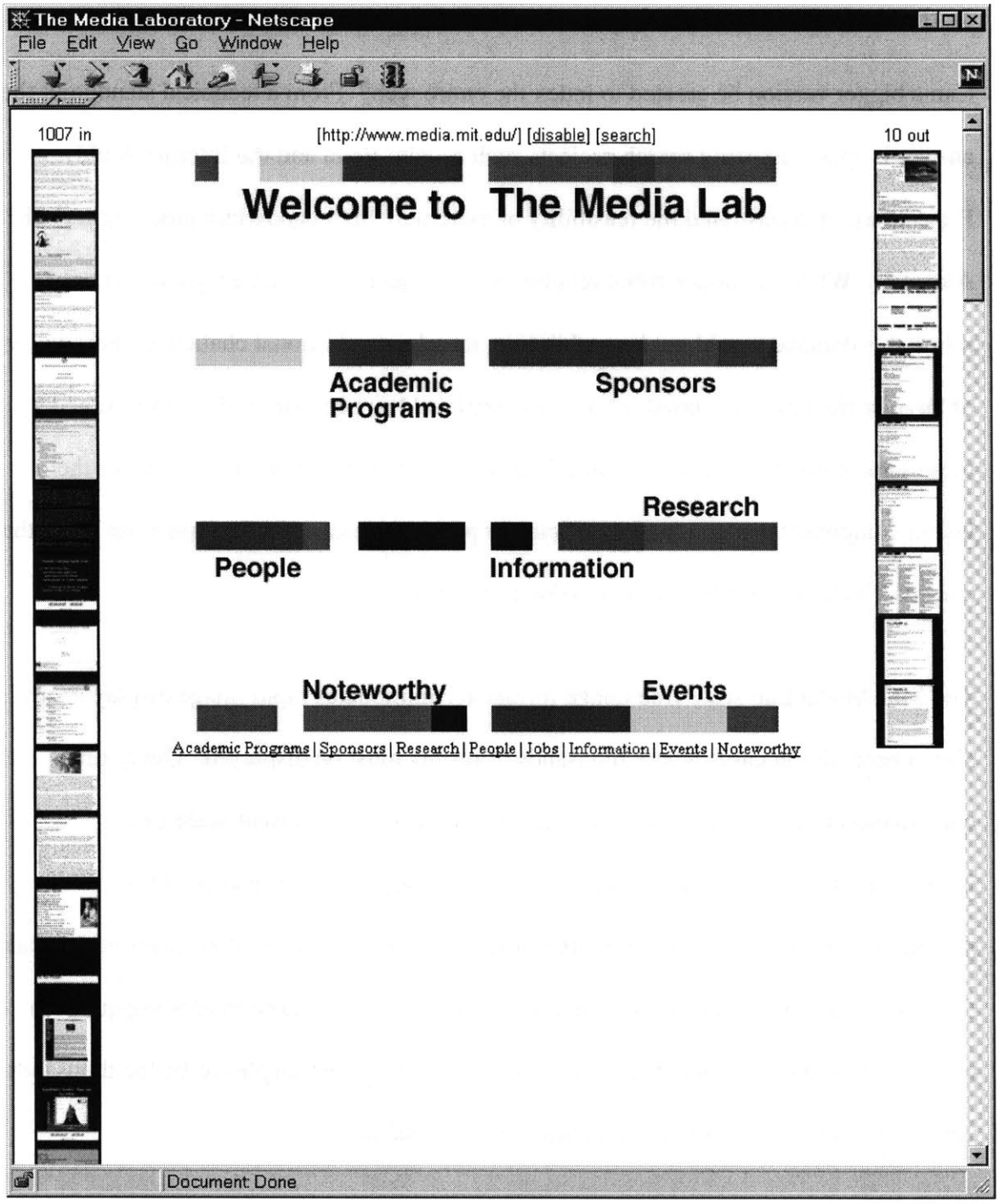

Figure 6.3: Proxy view of the Media Lab home page.

Is the scale free retrieval system scalable?

Can a bigger version be created to index the entire web? From a technical standpoint the answer is 'yes'. Existing search projects such as Alta Vista and the Internet Archive Project have demonstrated the feasibility of extremely high bandwidth indexing [AV98, Alexa98]. While the image robot requires greater bandwidth than a typical text only robot, this demand would not be prohibitive, merely an additional challenge. Feasibility of large-scale, keyword-based information retrieval has been similarly demonstrated by these same projects. The remainder of the scalability question involves whether the layout component can continue to efficiently provide meaningful displays even when the corpus is huge and result sets grow to be quite large.

The MIT Media Lab scale free image format accommodates rapid image display at any size, a necessity in cases where thousands of results must be displayed. Query time computational resources are at least somewhat higher in the current scale free implementation because each image file requires a separate disk access. This performance hit could be substantially improved by storing the smallest layer of the scale free images directly in the information retrieval database. A more interesting question remains open: do the results continue to be meaningful when displayed by the thousands versus the 50 to one hundred maximum explored thus far?

In order to provide rapid access to at least some of the 1000 results, it is necessary to display them in a variety of sizes so at least a few will be large enough to recognize and select. A uniform grid based layout is probably not appropriate for a massive array. Document similarity clustering could be ideally suited to displaying huge results sets if it

is possible to partition the results into a reasonable number of clusters and select a representative document from each to be displayed in a larger size. The user would then zoom in to the most appropriate looking cluster. It would also be possible to dynamically re-cluster the results into sub clusters once the display has been cropped.

The hyperlink based layout would continue to function as it does presently, even when number of results grows dramatically. Since document scale varies with hyperlink depth, links that are three steps away are small, and those that are four or more links away are not always clearly visible. Depending on the fan-out characteristics, the hyperlink layout would continue to show only the four or so layers of links at any given time.

General comments

Users provided other helpful comments regarding the system as a whole. The most universal issue raised was that of *spam* and how it might adversely affect the system in the future. Spamming is the practice of using inaccurate or outright deceptive information to attract undeserved attention to a web page. Spammers manipulate conventional search engines by modulating the keyword frequencies in their web pages to make them appear relevant to specific search terms. Another technique used—one which might be called bait and switch—involves making a special keyword frequency adjusted web page available to web spiders (which are easily detectable by the user agent field they use and domain from which they originate), while making a totally different web offering to human visitors. Spammers could theoretically manipulate the scale free image grabbing spider with similar methods. While a few heuristic avoidance methods come to mind, there is no easy answer or solution to this problem.

Another design choice questioned was the arbitrary selection of black as background color for the scale free images. In some cases the striking contrast with white web pages was distracting. For this reason, a shade of gray may have been more appropriate. It is impossible to change the background color in the current implementation, but might be worth considering modifying the scale free code to support an alpha or transparent channel, so that any background color can be used in the future.

Finally, a square aspect ratio was arbitrarily selected to simplify layout algorithms. Because index content is HTML, there is no right or wrong aspect ratio. The biggest drawback is that only the first page or so is captured, and no indication of the total length is available.

Future work

The implementation of this project is complete in this sense that design goals were satisfied by a fully functioning prototype, and all of the preconceived layout experiments were conducted. The design and implementation process raised many new questions and ideas for further exploration, as well as numerous opportunities for improvements. In no particular order, the more major future work includes:

1. Experimentation with hyperlinks based layouts that deal exclusively with inbound or outbound hyperlinks but not both. Doing so (at least in the outbound hyperlink case) will place a greater emphasis on the structural design created by web authors.

2. Total automation the spider subsystem. The current implementation requires a fair amount of manual effort, but after the recently publicly released Netscape source code is modified, spider automation should be totally automated. Once automated, high level controls for starting and stopping the spider should be added to the administrator interface.

3. Expansion of the user interface such that it is very fast, robust, totally self explanatory, and can be used by a diverse community of users. Doing this will make it possible to collect usability data. Since the prototype was put together quickly, there are numerous performance and reliability improvements that could be made.

4. Experimentation with different underlying search technologies such as latent semantic indexing. This should be feasible since the search engine subsystem is totally interchangeable.

5. The scale free imaging subsystem was made to handle movies in addition to stills. We can extend the system along this dimension and evaluate its performance at retrieving motion pictures. In addition to movies, it would also be pertinent to index other types of media, such as postscript, PDF, sounds and images. Metadata could be very helpful for some of these.

6. Application of the technology to the task of indexing the entire web. Many of the ideas stated here would continue to hold, but this massive task would also necessitate changes to accommodate the new order of magnitude-a multithreaded spider and large-scale search engine, for instance.

7 Conclusion

Information retrieval systems for the World Wide Web have evolved to an impressive state. The technical hurdles surrounding large-scale spidering, updating, and indexing, have been skillfully surpassed. The typical user interface, however, has seen little change since the earliest days of information retrieval.

This thesis introduces a new visual interface—scale free information retrieval—and describes details of a prototype implementation including seven experimental layout approaches. The fully working prototype includes a specially designed web spider, a keyword search engine for handling user queries, several databases of web page information, and a unique graphical layout engine for displaying the results. An index of the 85 MIT Media Lab web servers, containing a total of more than 20,000 web pages, was created as the initial test bed.

The new system distinguishes itself by using pictures as a powerful supplement, and even replacement, for the traditional textual descriptions used to communicate search results. Even the smallest thumbnails contain a wealth of visual information regarding document structure and nature. Furthermore, the small images can reflect this information with a high degree of parallelism.

Research in cognitive science supported the initial idea; real user feedback validated the efficacy of the working prototype. There was also an unexpected but important discovery: hyperlink heuristics can significantly improve search result rankings.

Not only is the technology clearly appropriate for indexing small intranet collections, as was done with the Media Lab corpus, but there is also strong evidence that it could be used for indexing the entire web. Existing systems demonstrate the feasibility of extremely large-scale spider based indexing of the web, and scale free imaging and the nature of several of the visual layout designs presented are inherently scalable as well.

References

Adelson, E. H., Anderson, C. H., Bergen, J. R., Burt, P. J., and Ogden, J. M., "Pyramid Methods in Image Processing", RCA Engineer, 29(6), pp. 33-41, 1984.

Anick, P. G. and Vaithyanathan, S., "Exploiting clustering and phrases for context-based information retrieval" in Research and development in information retrieval, Page 314, 1997.

Apple Computer, HotSauce Project, http://hotsauce.apple.com/, 1996.

Bederson, B. and Hollan, J., Stewart, J., Rogers, D., Druin, A., and Vick, D., "A Zooming Web Browser" in Proceedings of SPIE Multimedia Computing and Networking, Volume 2667, pp. 260-271, 1996.

Bederson, B. and Hollan, J., "Pad++: A Zooming Graphical Interface for Exploring Alternate Interface Physics" in Proceedings of UIST, 1994.

Berkeley Digital Library Project, SWISH-E Home Page, http://sunsite.berkeley.edu/SWISH-E/, 1998.

Bove, V. M., and Lippman, A. B., "Scalable Open Architecture Television", SMPTE Journal, January 1992.

Charikar, M., Chekuri, C., Feder, T., and Motwani, R., "Incremental clustering and dynamic information retrieval" in ACM Theory of computing, 1997, pp. 626–635.

Digital Equipment Corporation, Alta Vista Search Service, http://www.altavista.digital.com/, 1996.

Dodge, C., "An Adaptive Imaging Environment", MIT Media Laboratory Technical Report, http://tvot.www.media.mit.edu/projects/TVOT/Agenda95-2/Scalefree.html, 1996.

Dreilinger, D., "Internet Search Engines, Spiders, and Meta-Search Engines" in *Bots and other Internet Beasties* (ed. Joseph Williams), Sams Net, Indianapolis, Indiana, 1996, Chapter 12, pp. 237–256.

Dreilinger, D., SavvySearch Search Service, http://savvy.cs.colostate.edu:2000/, 1995.

Frakes, W. B and Baeza-Yates, R. (eds.), *Information Retrieval: Data Structures & Algorithms,* Prentice Hall, pp. 363-392, 1992.

Hughes Technologies, Mini SQL 2.0 Introduction, http://www.hughes.com.au/library/msql2/info.htm, 1997.

Iyengar, G., and Lippman, A. B., "Models for automatic classification of video sequences" in Storage and Retrieval from Image and Video Databases VI, SPIE/IS&T conf. 3312, San Jose, January 1998.

Jain, A. K. and Dubes, R. C., *Algorithms for Clustering Data*, Prentice Hall, 1988.

Kahle, B., Alexa Internet Home Page, http://www.alexa.com/, 1998.

Koster, M., The Standard for Robot Exclusion, http://info.webcrawler.com/mak/projects/robots/norobots.html, 1994.

Lassila, O. and Swick, R. R., "Resource Description Framework (RDF) Model and Syntax", http://www.w3.org/TR/WD-rdf-syntax/, 1998.

Lieberman, H., "Powers of Ten Thousand", MIT Media Laboratory Technical Report, http://lieber.www.media.mit.edu/people/lieber/Lieberary/Macroscope/Macroscope.html, 1994.

Murtaugh, M., NiF Elastic Catalog, http://nif.www.media.mit.edu/, 1996.

Netcraft, "The Netcraft Web Server Survey", http://www.netcraft.com/Survey/, 1998.

Pinkerton, B., "Finding what people want: experiences with the WebCrawler", 1995.

Preece, J., Rogers, Y., Sharp, H., Benyon, D., Holland, S., and Carey T., *Human-Computer Interaction,* Addison-Wesley, p. 118, 1994.

Rickard, J., "Boardwatch Magazine Internet Service Providers Directory", http://www.boardwatch.com/isp/winter98/intarch.html, 1996.

Rivest, R., RFC 1321: The MD5 Message-Digest Algorithm. Internet Activities Board, April 1992.

RSA Cryptography Labs, Frequently Asked Questions v3.0, http://www.rsa.com/rsalabs/newfaq/, 1998.

Salton, G., *Automatic Information Organization and Retrieval*, McGraw-Hill Book Company, N.Y., pp. 21–65, 1968.

Small, D., "Navigating Large Bodies of Text" in IBM Systems Journal, Vol. 35, No. 3&4, 1996.

Stone, M. C., Fishkin, K., and Bier, E. A., "The Movable Filter as a User Interface Tool", SIGCHI, 1994.