

# TouchPat: A Touch Based Zoomable User Interface for Exploring Large Patent Collections

---

*Master Thesis, July 25, 2011*



Michel de Ridder



---

# TouchPat: A Touch Based Zoomable User Interface for Exploring Large Patent Collections

---

THESIS

submitted in partial fulfillment of the  
requirements for the degree of

MASTER OF SCIENCE

in

MEDIA & KNOWLEDGE ENGINEERING

by

Michel de Ridder  
born in Vlaardingen, The Netherlands



Computer Graphics Group  
Department of Mediamatic  
Faculty of EEMCS, Delft University of Technology  
Delft, the Netherlands  
<http://graphics.tudelft.nl>



Research & Development  
European Patent Office  
The Hague, the Netherlands  
<http://www.epo.org>





---

# TouchPat: A Touch Based Zoomable User Interface for Exploring Large Patent Collections

---

*Author:*

Michel de Ridder

*Student id:*

1258931

*Email:*

mridder04@gmail.com

## Abstract

Patent examiners are under increasing pressure to quickly and accurately evaluate incoming patent applications within large databases. For each incoming patent application, they first extract a subset of possibly relevant patents through keyword-based search queries. These results are summarized in long lists, which provide only minimal information on each patent. To immediately determine whether a patent is possibly relevant or not, each patent needs closer examination by opening it and scanning through its content, including drawings. For large result lists, this is a time consuming and strenuous job in the current user interface. In this work, we explore the applicability of novel information visualization techniques and the use of touch-based interactions. Our findings are included in the prototype application *TouchPat*, a touch-based *zoomable user interface* for exploring large patent collections. TouchPat provides an overview of a patent collection up to 1000 patents, and allows examiners to use *multi-touch interactions* to explore and process it. To summarize the information, each patent is visualized with a symbolic thumbnail representation. We use *semantic zooming* to adapt the amount of meta information for each zooming level and make optimal use of the available space. This thesis describes the design and development approach of the TouchPat prototype application, the individual visualization and interaction techniques used and its evaluation with the patent experts. TouchPat was well received with the patent experts, and its user interface experience encourages further improvement in the speed and accuracy of the patent examination process.

Thesis Committee:

Full Professor:

Prof.dr.ir. F.W. Jansen, Faculty of EEMCS, TU Delft

Daily Supervisor:

Dr.ir. G. de Haan, Faculty of EEMCS, TU Delft

External Supervisor:

Dr.ir. B. Diallo, Head of Research, EPO The Hague

Associate Professor:

Ir. F.H. Post, Faculty of EEMCS, TU Delft

Assistant Professor:

Dr.ir. P. Wiggers, Faculty of EEMCS, TU Delft



---

# Preface

I hereby present the results of my Master's thesis project on which I have been working on for 8 months. I was lucky to do an internship at the research and development department of the European Patent Office, The Hague. The European Patent Office (EPO<sup>1</sup>) is a large intergovernmental organization. One of their main tasks is to examine incoming patent applications. The EPO has access to multiple databases containing millions of patents from all over the world. Patent examiners have to create smart search queries in order to get only the most relevant patents from the large database. This project was coordinated partly at Delft University of Technology and partly at EPO, The Hague. EPO was interested in seeing whether the dreams they have about a new user interface that can be controlled with multi-touch interactions, can become reality. The TU Delft is interested in the interactive information visualization aspect of this project.

With the advent of smartphones and tablet computers, people get familiar with using multi-touch devices. The subject of my thesis project really appealed to me and I was pleased to be able to work on a multi-touch application. Prior this project, I did a literature study [Ridder 11], partly for another course and partly for this project, on document and patent visualization techniques and multi-touch interactions.

This project could not have been completed successfully without the guidance, support and feedback of a number of people I would like to thank now. First I would like to thank my supervisors, Gerwin de Haan and Barrou Diallo, for their inspiring and motivating attitude as well as their critical view. Secondly, I would like to thank Bertrand Le Chapelain and Emeline Marttin for their informative workshops which gave much knowledge about patents and patent examination. I would also like to thank the EPO employees that attended my presentation and discussed it with us and Frits Post for his feedback on this project. Finally, I would like to express many thanks to fellow graduate students and lab mates for providing feedback on my application as well as informal lunch meetings.

Last but not least I would like to thank my family and friends for supporting me along this journey.

Michel de Ridder  
Vlaardingen, the Netherlands  
July 25, 2011

---

<sup>1</sup><http://www.epo.org/>



---

# Table of Contents

<b>Abstract</b>	<b>i</b>
<b>Preface</b>	<b>iii</b>
<b>Table of Contents</b>	<b>v</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Background and Motivation . . . . .	1
1.2 Problem Definition and Goal . . . . .	9
1.3 Document structure . . . . .	10
<b>2 Related Work</b>	<b>11</b>
2.1 Document Collection Visualization . . . . .	11
2.2 Patent Visualization . . . . .	13
2.3 Multi-Touch Interactions . . . . .	14
2.4 Approach to Solutions . . . . .	15
<b>3 User Interface Design</b>	<b>17</b>
3.1 Functions . . . . .	17
3.2 Multi-Touch Input . . . . .	19
3.3 Sketches . . . . .	20
3.4 Improvements . . . . .	23
3.5 Document Collection Visualization . . . . .	24
3.6 Thumbnail Visualization . . . . .	27
3.7 Conclusions . . . . .	33
<b>4 Touch-based Interaction</b>	<b>35</b>
4.1 Framework . . . . .	35
4.2 Touch Input . . . . .	36
4.3 Basic Navigation . . . . .	37

---

4.4	Constraint Zooming and Translation . . . . .	38
4.5	Detail Patent Viewer . . . . .	39
4.6	Context Menu . . . . .	42
4.7	Conclusions . . . . .	44
<b>5</b>	<b>The Prototype: TouchPat</b>	<b>45</b>
5.1	Results . . . . .	45
5.2	Expert Reviews . . . . .	47
5.3	Discussions . . . . .	49
5.4	Conclusion . . . . .	51
<b>6</b>	<b>Technical Details</b>	<b>53</b>
6.1	Data Import . . . . .	53
6.2	Touch Input . . . . .	56
6.3	Use of Diagrams . . . . .	58
6.4	Annotations . . . . .	61
6.5	Path Tracking . . . . .	62
6.6	Ranking . . . . .	63
6.7	Performance . . . . .	63
<b>7</b>	<b>Guidelines for Multi-Touch Interfaces</b>	<b>67</b>
7.1	Interaction Guidelines . . . . .	67
7.2	Design Guidelines . . . . .	72
<b>8</b>	<b>Conclusions and Future Work</b>	<b>75</b>
8.1	Conclusions . . . . .	75
8.2	Contribution . . . . .	77
8.3	Future work . . . . .	77
	<b>Bibliography</b>	<b>81</b>
	<b>List of Figures</b>	<b>85</b>
	<b>List of Tables</b>	<b>89</b>

# Chapter 1

---

## Introduction

The number of patents is increasing fast. The patents are stored in large databases that contain millions of patents. Patent examiners must find ways to obtain only patents that contain relevant information to the application. In most of the current digital libraries, keywords are used to create search queries and the results are displayed in list views. These lists do not always provide enough information to decide whether the document is relevant or not. Small result lists can be analyzed easily by studying the documents one by one. Large result lists on the other hand show some problems [Bier 04]: acquiring documents from their citations; managing document collections and citations; finding important sub-collections of documents; choosing documents to read first in a collection; tracking which documents have been read. Current systems deal with these problems separately. A solution that combines the best solutions in one visualization might be better.

In this chapter, we will first give some background information and motivation of this master's thesis project in Section 1.1. In Section 1.2 we describe the problems of the current application and the goal of this project. Finally, an outline of this document is presented.

### 1.1 Background and Motivation

Before we started this project we had to increase our knowledge about patents. The EPO arranged several workshops with patent examiners from different fields. These workshops provided us more information about what patents look like; the applications they use for patent examination with their problems; and what search strategies the examiners use.

#### 1.1.1 Patent Structure

Patents are legal documents. A patent authority grants a set of rights to the inventor to the exclusivity of an invention in a geological area. With these rights, the patent owner can exclude others from making, using and selling the invention during a term up to 20 years. The process in which the exclusivity of a patent application is determined is called *patent examination*. During patent examination, the patent examiner will search for prior art to

prove whether the invention is exclusive or not [Lupu 11].

A patent is structured in four different parts: bibliography, description, claims and drawings. The first page of a patent includes bibliographic data relating to the filing details and ownership of the patent. The bibliographic data is defined in the World International Property Organization (WIPO<sup>1</sup>). ST.9 of the WIPO defines about 60 data entities widely used on the first page of patent documents. Each metadata entity is associated with a unique two-digit INID code (Internationally agreed Numbers for the Identification of bibliographic Data). For example, the number 54 is associated with the title of the invention. Figure 1.1 shows the first page containing bibliographic information of patent number EP 0392853 A2. The databases index all meta data fields which can be used by the examiners to refine their search queries. Some important data entities of the bibliographic information are described here.

- **Title** A patent document is required to have a descriptive title of the invention.
- **Applicant** The applicant is the prospective owner of the patent rights; this can be a company, institution or individual inventor(s).
- **Inventor** Individuals who contributed to the conception of the invention should be named here.
- **Classification** National/International classification codes assigned to the patent at the time of publication (For example: International Patent Classification (IPC) or European Classification (ECLA)).
- **Publication date** The date on which the patent is authority published. For most granted patents this is the date when the exclusive rights start.
- **Application number** The serial number assigned by a patent office when the patent is filed at the patent office.
- **Abstract** In the abstract, the invention is described shortly and the claims of the invention are reflected.


The following pages of the patent include a description and the claims of the patent. The description part gives background information and a description of the invention. The claims part setting out the essential features of the invention to make clear when you will violate the patent. The pages behind the claims section are filled with drawings of the invention or related statistics.

### 1.1.2 Current Applications

The examiners have access to a set of tools which can help them search for relevant patents. The most important tools are: Query Builder, Query Build Assistant (QBA), Viewer and Trimaran. In the Query Builder tool, the examiner creates search queries by making use of Boolean expressions and keywords. Executing a search query results in a list of patents.

<sup>1</sup>[http://www.wipo.int/standards/en/part\\_03\\_standards.html](http://www.wipo.int/standards/en/part_03_standards.html)




**Europäisches Patentamt**  
**European Patent Office**  
**Office européen des brevets**

(11) Publication number: **0 392 853 A2**

(12) **EUROPEAN PATENT APPLICATION**

(21) Application number: 90303991.5  
 (51) Int. Cl.<sup>5</sup>: **G11B 7/00, G11B 7/007, G11B 7/013**  
 (22) Date of filing: 12.04.90

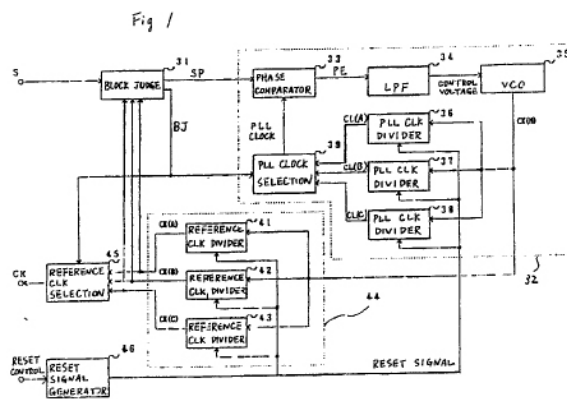
<p>(30) Priority: 13.04.89 JP 95648/89</p> <p>(43) Date of publication of application: 17.10.90 Bulletin 90/42</p> <p>(94) Designated Contracting States: DE FR GB NL</p> <p>(71) Applicant: <b>SHARP KABUSHIKI KAISHA</b> 22-22 Nagaïke-cho Abeno-ku Osaka 545(JP)</p> <p>(72) Inventor: <b>Deguchi, Toshihisa</b> C-722 Daiya Heights, 2-200-5,</p>	<p><b>Gakuendaiwa-cho</b> <b>Nara-shi, Nara-ken(JP)</b> Inventor: <b>Fuji, Hiroshi</b> <b>3-1-30-206 Kataokadai, Kanmaki-cho</b> <b>Kitakatsuragi-gun, Nara-ken(JP)</b> Inventor: <b>Terashima, Shigeo</b> <b>95-86 Kaminoshō-cho, Nikaido</b> <b>Tenri-shi, Nara-ken(JP)</b></p> <p>(73) Representative: <b>Huntingford, David Ian et al</b> <b>W.P. THOMPSON &amp; CO. Coopers Building</b> <b>Church Street</b> <b>Liverpool L1 3AB(GB)</b></p>
---	--

(54) **An apparatus for recording and reproducing information on and from an optical disk.**

(57) An apparatus for recording and reproducing information on and from an optical disk in which at least one optical beam and a reference clock signal are used. The optical disk comprises recording tracks which are divided into a plurality of blocks and concentrically arranged along the radial direction. The apparatus comprises: a block judging unit for judging which one of the blocks is the one which

is currently impinged by the optical beam; a clock signal generator for generating a plurality of clock signals which are different in frequency from each other; and a clock signal selecting unit for selecting one of the clock signals as the reference clock signal, on the basis of the judgment of the block judging unit.

EP 0 392 853 A2



Xerox Copy Centre

Figure 1.1: First page of patent with number EP 0392853 A2, published 17 October 1990.

For large resulting lists, a collection that contains over 500 patents, the examiner will refine the search query by adding keywords. For small resulting lists, the examiner will export the result to the Viewer tool to study the patents in more detail.

The QBA tool helps refining a search query, by creating synonyms for the defined keywords. Synonyms help examiners in finding new keywords to refine their search query or as starting point for a new search query. Some inventors try to use synonyms for commonly used words in order to get the rights of an invention. An examiner cannot afford to miss any information, the QBA tool help preventing missing any information.

Once the examiner has a resulting list smaller than 500 patents, the examiner starts to examine the patents in detail. The list itself does not provide enough information to directly say something about the relevance of the patents. Therefore, the results are exported to the Viewer tool. The Viewer is their main tool to visualize a single patent document. Figure 1.2 shows the interface of this application. The viewer application contains three different windows. On the left side it has a text viewer where the full-text information will be displayed. In the middle it contains an annotation bar. This bar gives an overview of where predefined keywords are occurring in the text and their frequency. And on the right side it has an image browser, which can be used to watch the drawings of the patent.

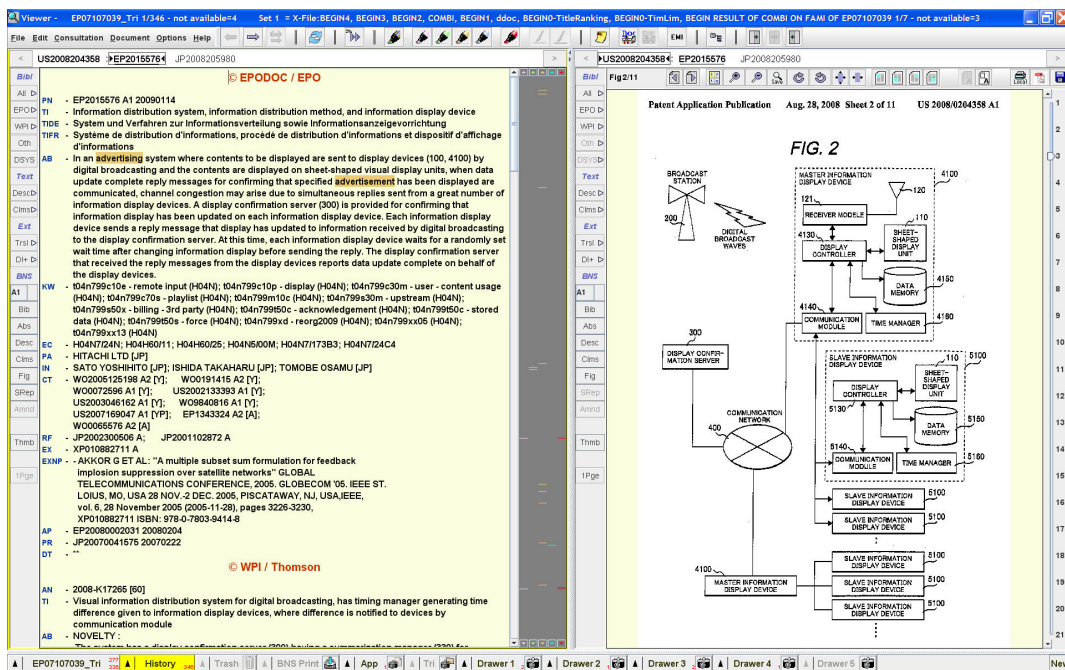


Figure 1.2: The user interface of the Viewer tool contains three sections: a text viewer, an annotation bar and an image browser.

The final tool we describe is Trimaran. The Trimaran tool is used by the examiner to write their final report for a patent application. The examiner will go through all claims of the application and check the exclusiveness for each. With Trimaran the examiner can write a

report containing references that should be cited in the course of the examination.

### 1.1.3 Search Strategies

How examiners search for relevant patents depends on the field of the application and the examiner. In some fields text is the most important while in other fields images are more important in searching for relevant information. For example, in the electronics field images are very important. The field of the application also decides whether they have to search for related patents only or whether there need to be searched for non-patent literature like articles and books as well. In general the examiner always starts with analyzing the patent application. Some examiners create a summary in their head while others write them out on a paper. We describe two examples of search strategies used in different fields.

#### Field of Audio, Video and Media

A patent examiner in the field of Audio, Video and Media has showed us the steps he normally takes to examine a new patent application, during a workshop. He starts his examination process with reading the patent in detail and taking a look at the drawings. Based on what is written in the application he starts to build search queries and see how many results he gets. If the resulting patent collection contains too many patents, he will refine the query or apply statistics on them. The statistic command creates an overview of how many patents belong to each classification. The result of these statistics can be used to refine the search query. A different way to refine the search query is by looking at patent families or by looking for earlier patents of the applicant or inventor. A patent family is a set of either patent applications or publications taken in multiple countries to protect a single invention by a common inventor(s) and then patented in more than one country. Once the resulting patent collection is small enough (<500), he will export the collection to the Viewer tool to study the patents one by one. The examiner has to analyze the images (images are very important in the field of electronics) or add annotations in order to quickly get insight about the relevance of the patent. The examiner can add annotations by giving specific keywords a highlight color. These annotations are visualized both in the text and in the annotation bar and help indicating what part of the text should be read first. If a patent is relevant, he will add the patent to one of the drawers (stacks). He creates different stacks of patents based on their relevance. One stack contains all patents that have a high relevance and another stack contain all patents with less relevance. If a patent contains relevant information, he will also add a note to it. Notes can be useful in retrieving the patents later on. He will walk through all the claims of the application, to check whether they are exclusive or not.

#### Field of Pure and Applied Organic Chemistry

A patent examiner in the field of Pure and Applied Organic Chemistry has showed us a different way to examine a patent application. She starts her patent examination process with reading the application in detail and creating a summary of it on a paper. This summary contains three different sections: *visualization*, *table of concept* and a *claim tree*. The visualization should describe the main concept of the patent application. An example of this

visualization in the field of chemistry is visible in Figure 1.3. The examiners are experts in their field. If they are not able to make such a drawing, it indicates that there is information missing. This visualization is very useful in the field of biochemistry but for other fields, like telecommunications, it is less useful.

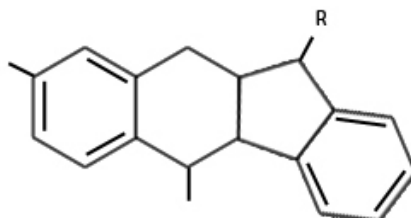


Figure 1.3: A patent application is summarized with a visualization that describes the main concept of the application.

The second part is a *table of concept*. This table gives an overview of the most important concepts defined in the claims section of the application. This table contains for each concept: the title, classification and keywords. Table 1.1 shows the structure used for the table. This table is very useful for defining search queries, by making different combinations of keywords and classifications. For example, the combination of the keywords of concept 1 with the classification of concept 2 and the keywords of concept 3 covers all concepts. The result of this query contains patents that have information from all concepts and therefore have a high relevance.

Concept	Concept 1	Concept 2	Concept 3
Classification	A61K31/10	A61K9/16	A61K38/16, A61K48/00
Keywords	Keywords	Keywords	Keywords

Table 1.1: This table shows the structure of the table of concepts. This table contains a title, classification and keywords for the three most important concepts of the patent application.

The last part is a *claim tree*. The claims described in the claim section of the application are linked to each other. A claim tree visualizes the relations between claims, by showing the connection between two claims. Figure 1.4 shows an example of a claim tree. The deeper in the tree, the more specific the information is written. It is also possible that claims are not linked to other claims. To check whether an application is unique the examiner has to check whether the claims made are exclusive or not. Marking all the claims covered so far or which might be straight forward, helps indicating how much work is still need to be done.

The summary sheet created gives a perfect overview of the application. The first step an examiner should take in the patent examination process is to look for previous patents by the applicants or inventors. Previous applications about the same topic may contain cita-

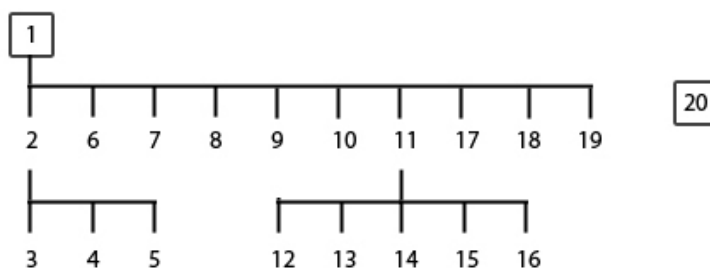


Figure 1.4: A claim tree shows the relations between the claims in a patent application.

tions to related patents and documents. It is also important to check the references noted in the patent application and to patents or documents which cite these. Basically any kind of documents (i.e. scientific papers and books but also to websites) can be referred. The final step is to build search queries based on combinations of keywords and classifications defined in the table of concepts. The best way is to start with very specific information and broaden the search area if little or no results are obtained.

#### 1.1.4 Patent examination process

The patent examiner has limited time to check whether the claims of an incoming patent application are new, inventive and has an industrial application. The examiner starts the patent examination process by analyzing the application in detail. He creates search queries with keywords from the topic of the application. The results of the queries are returned in a list view. These lists can contain up to a few thousands patents and do not provide enough information to directly decide over the relevance of the patent. The number of resulting patents can be decreased by refining the search query. An examiner cannot afford to miss any relevant information, so at some point they have to study the patents in more detail. The examiner has to open the patents one by one in order to get more information. Once the examiner has a result list of <500 he will study the collection in more detail. One problem with patent documents is that they are too long and complex to study them in full depth while everything from a patent might be relevant. The examiner has to scan the document in order to quickly go through the entire collection. The patent examination process is summarized in Figure 1.5.

The examiner will create search queries and analyze these on a high level base. If the result list contains too much patents, the examiner will refine the query else he will analyzed the result list in more detail. Finally, when the examiner found enough evidence to provide the exclusiveness of the patent application, he will right his final report.

Before the patents were digitalized, patents were stored and ordered in large archives. The examiners had to search in this archive for patents that contain relevant information to a patent application. Once the examiner collected a couple of patents, he spread them out on the floor and created stacks to create a reclassification of the patents. Examiners that work for a longer time at the office begin to remember where specific patents can be found and

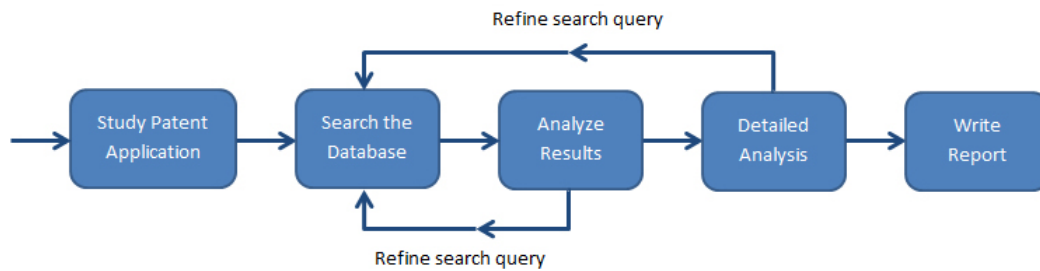


Figure 1.5: The patent examination process is summarized in five steps.

identified them by their appearance. Creating a visualization of the patent collection can help recognizing patents by their place in space and how they are visualized.

The development of multi-touch displays has opened a new world of possibilities in the field of controlling an application in a more natural way. With the arrival of smartphones and tablet computers, multi-touch is becoming more common. The EPO is interested in whether these new possibilities are useful for their application and to see whether it is possible to get back the so called "hands on" experience. In Figure 1.6 several examples of multi-touch devices are visible which can be used as multi-touch input device.



Figure 1.6: An overview of different multi-touch devices that can be used as platform for our prototype.

## 1.2 Problem Definition and Goal

We have described the structure of a patent document, the applications examiners use and two different search strategies in different fields. This knowledge allowed us to define a couple of problems with the current application.

- **Integration** Examiners currently use many different tools to help with their patent examination. One problem with these tools is that there is no direct connection between the different applications. To use the same dataset, the results need to be exported and imported from one tool to the other tool. Another disadvantage of having separate tools is that they appear separately in the taskbar. As a result, to find the desired tool the examiner needs to switch between the different windows. A better solution is one tool that contains different views and does not require importing and exporting the data.
- **Annotation** Annotation gives users quick insight about the relevance of the patent and what parts of the patent should be read first. With annotations, keywords can be highlighted by giving them a specific background color. In the current application, annotations are only used in the Viewer tool in the text viewer and in the annotation bar. These annotations can be useful in a patent overview as well, by creating an iconic representation for each patent based on the annotations. The icon helps filtering patents from the collection that are not relevant.
- **Usability** The current interface is optimized to operate with a mouse device, and contain many elements such as buttons, lists and tabs. When examiners study a single patent or patent collection, they must make a large number clicks. This affects the learning curve of the application. A long training period is needed before the user gets familiar with all the possible options.
- **Overview** None of the current tools the examiners use is able to provide an overview of a patent collection. The only way to analyze a patent collection is by going through the patents one by one. A global overview of the patent collection gives better insight in which patents can be skipped and which should be read in detail. It also may lead to different search strategies and starting positions.

The goal of this project is to design and implement a prototype with which the examiner can browse and navigate through a large patent collection in a different way as they are used to. The starting point of our prototype is a collection of 1000 documents, obtained by a search query for a patent application, see Figure 1.7. We will include the most important functionality of the current application, try to improve these and add new visualization techniques.

The following main research goal was formulated:

**Research goal:** Investigate and construct a prototype that gives the user a visual overview of a patent collection of up to 1000 documents; allow the user to browse and navigate through this collection by using a multi-touch input device; and be able to create a subcollection of two to five patents that are most relevant based on the patent application.

Based on my literature study in the field of document and patent visualization [Ridder 11] we have defined three hypotheses that we think will help in creating a good prototype:

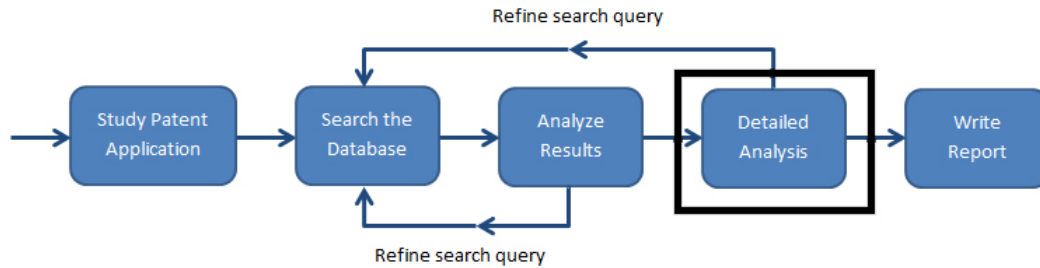


Figure 1.7: The focus of this project lays on analyzing a subcollection up to 1000 patents in more detail.

- Zoomable User Interface creates a workable overview.
- Patents are recognized by their visual representation and spatial location.
- Multi-touch input devices creates natural interactions.

### 1.3 Document structure

The structure of this thesis is as follows: Chapter 2 provides an overview of previous research done in the field of document collection visualization, patent visualization and multi-touch interactions. Based on our findings, we have defined the functions and tasks of the application and created our first sketches for the application. Chapter 3 describes these and explains how these have led to the current user interface designs. Chapter 4 goes in more detail about the interaction between the user and the application. It starts with an overview of the advantages and disadvantages of touch input and principles for user interfaces and ends with the results of using touch input to browse and navigate through the patent collection. Chapter 5 combines the results of previous made choices in a prototype and describes how we have evaluated the prototype. Chapter 6 provides an overview of the technical issues and choices made during the implementation phase. Chapter 7 describes guidelines which can be used building a multi-touch application. Finally Chapter 8 concludes this thesis, discuss the contribution of this project and propose some future work.



## Chapter 2

---

# Related Work

Related work for this project can be found in several research areas: document collection visualization, interaction techniques, thumbnail representations, patent visualization, multi-touch ergonomics and multi-touch interactions. Prior this project, I did a literature study [Ridder 11] which mainly focussed on document visualization and interaction techniques. We extended this study with research from the other fields. In this chapter, we will give a short overview of these developments, the most relevant results and what our approach to solutions is.

### 2.1 Document Collection Visualization

In the past, several frameworks were developed to visualize large document collections. These frameworks can be categorized in two groups: document thumbnails and icons (stimulates visual memory) and multiple document layouts (stimulates spatial memory).

The first visualization technique we want to discuss is space filling thumbnails [Cockburn 06] and belongs to the first group. For each document in the collection a small unique thumbnail is created and visualized space filling by placing them next to each other. The thumbnails can be generated in different ways, for example by using an image of the first page, an image of a random page or an image from the document. It depends on the type of the document whether the thumbnail representations are useful. For example, patent documents are black and white and look very similar, while scientific papers contain color images and are easier to distinguish. Space filling thumbnail make very efficient use of the available space by filling the whole screen, see Figure 2.1. Once the document collection becomes very large, the thumbnails become so small that the information in the thumbnails is not visible. For very large collections we have to look for alternatives.

Recently, strong support was found that the figures of search results should be visualized next to the meta information [Divoli 10]. Bier [Bier 04] proposes to create thumbnail representation that combines both text and an image, see Figure 2.2. This thumbnail representation contains more information about the document to make decision about the relevance. The next visualization technique from the first group we want to discuss is tilebars [Hearst 95]. The tilebar visualization is a symbolic representation of the document based on keywords.

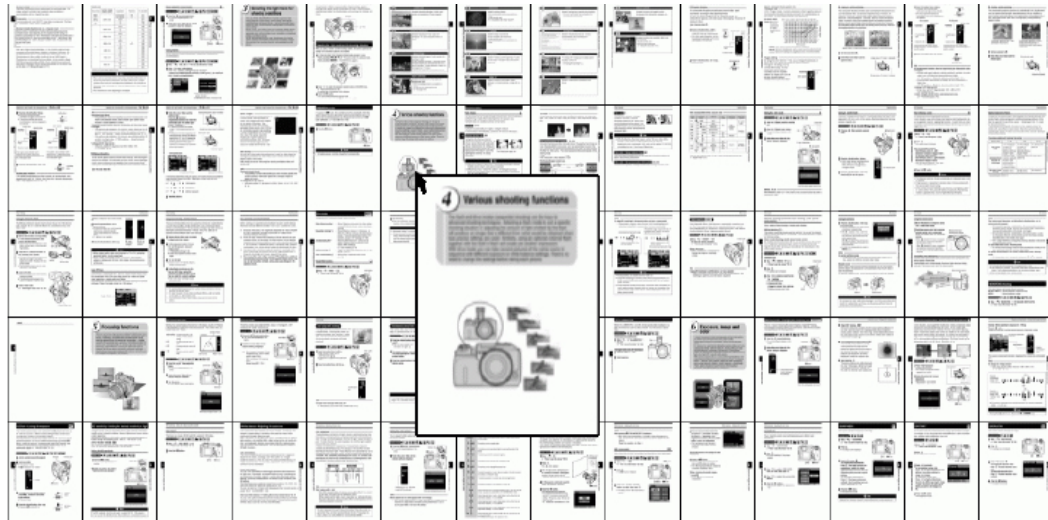


Figure 2.1: Space filling thumbnails fills the screen with a thumbnail for each document in the collection. The mouse pointer indicates which document should be previewed larger [Cockburn 06].



Figure 2.2: Combining thumbnails with meta information of the document (title, author and year of publication) [Bier 04].

The user has to define a couple annotation groups, which each contains one or more keywords and a highlight color. Next, the document is divided in equal parts and in each part the frequency of the keywords is counted and visualized by a tile in the tilebar. The more frequent a keyword occur in one of the parts the darker the color will be. This visualization gives quick insight about the relevance of a document and what part of the document should be read first.

In the second group, multiple document layouts, one of the first proposed visualization techniques is the graph visualization [Keim 02]. Graph visualizations are useful to visualize relations between documents or clusters of documents. Graph visualization on large document collections bring down some difficult problems. A large number of elements in a graph have bad influence on the performance, view-ability and usability of the graph [Herman 00]. Suggestions to solve this problem are a combination of new graph visualization techniques like: balloon view, radial view and hyperbolic view and interaction techniques like fisheye.

Another solution is to cluster the graph [Rauber 01]. This decreases the number of patents visible on the screen but keeps the graph small and understandable for the user. One example of a clustering algorithm is a self-organized map (SOM) [Yoon 02]. The SOM method showed meaningful results on actual document data sets [Morris 01] and is quite easy to implement. There is still room for improvement on better visualizing relations between different documents by color and space. Clusters are useful when the collection of the documents become too large to visualize them with thumbnails.

We also found an application that fits in both groups. Past research has shown that spatial memory helps us finding back things in virtual spaces. DataMountain [Robertson 98] is an example of an application that takes advantages of human spatial cognition. This application allows you to informally arrange your space in a very personal way and continue with your last arrangement once you come back to the application. Some conclusions followed from tests with DataMountain are: users find it easier to organize their favorites with DataMountain than with the standard favorite manager of a web browser; with DataMountain less attempts are needed to find related websites; users were more likely to find a website after one month of not using the favorites with DataMountain. We want to make use of this kind of properties by stimulating both the visual and spatial memory.

## 2.2 Patent Visualization

In the field of patent visualization we looked for visualization techniques used in the same area of the examination process as the focus of our solution. Most of the research is done in visualizing relations between patents, visualizing the queries and to do patent analysis. An example of a patent visualizations technique proposed, is a SOM. A SOM can be useful in identifying the technology vacuum, grasping patent conflicts and monitoring technology portfolios. Other ways to visualize relations between patents proposed are: timeline visualization [Sun 08] (see Figure 2.4), crossmap visualization [Giereth 07], citation maps [Gress 10].

Recently, PatViz [Giereth 08a] [Koch 10] proposes an interactive and visual way of building queries, see Figure 2.4. The PatViz project contains several visualization views to help analyze patents [Giereth 08b]: patent graph, world map, treemap, aggregation tree, text view, term cloud, geo-timeline, bar charts and tables. Tests with PatViz have shown that there was a high understanding of the queries while most users had never worked with a system providing interlinked and interactive visual interfaces. PATExpert<sup>1</sup> have developed a similar visualization tool [Wanner 08]. This tool shows four different views: a treemap, a list view, a patent graph and a console which contains information like the classification of the selected patent. The most important visualization technique to analyze a patent collection is probably text annotation. Text annotation can be applied in the text or to create abstract representations. Agatonovic [Agatonovic 08] proposes a method to automatically annotate sections, references and measurements in a patent.

---

<sup>1</sup><http://www.patexpert.org/>

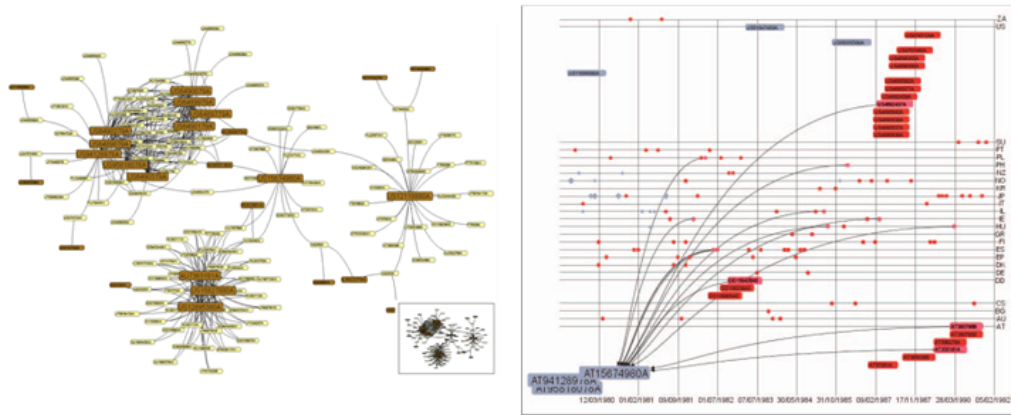


Figure 2.3: Force-directed layout of a patent family on the left and 2D matrix visualization on the right [Giereth 07].

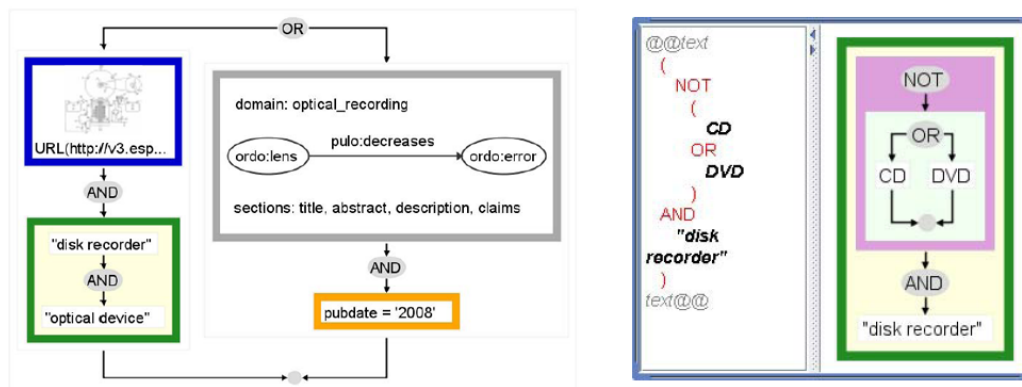


Figure 2.4: Visual representation of combined search expressions on the left and an example of text query with Boolean operators on the right [Koch 10].

## 2.3 Multi-Touch Interactions

Previous research in the field of multi-touch interactions is mostly done in ergonomics [Wigdor 07]. Several papers [Kewaley 08] [Moscovich 07] [Microsoft 09] [Wigdor 10] propose general rules that can be used in developing a multi-touch application. Freeman has proposed work in gesture based interactions [Freeman 09]. Research on multi-touch interactions showed that gesture based interaction is more efficient than button based interaction in multi-touch applications. OctoPocus [Bau 08] is an example of a dynamic guide that combines on-screen feedforward and feedback to help users learn, execute and remember gesture

sets. Still there need to be done a lot more research in ergonomics on the long term and multi touch user interface design. Isenberg [Isenberg 10] introduces Cambiera, a multi-touch application that is related to the goal of our project, see Figure 2.5. This application contains many aspects we want to include in our application. However, this work focuses on collaborative work and small document collections. For large patent collections we need to make improvements or choose other methods.

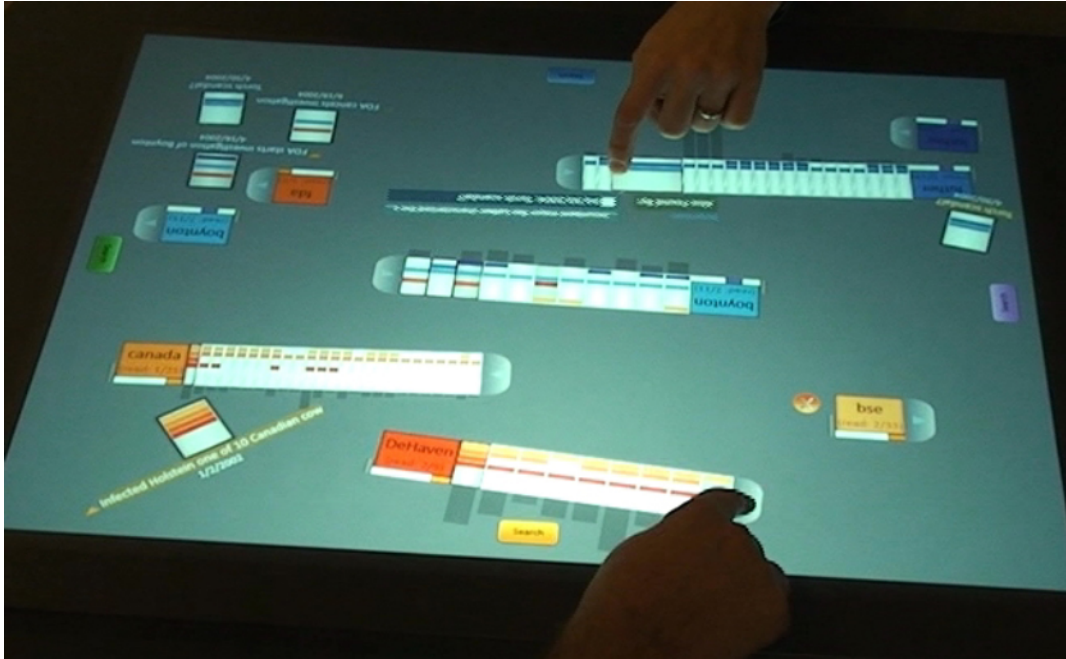


Figure 2.5: Overview of the workspace of Cambiera during an analysis session. Both analysts have arranged several search boxes and documents in the space related to their current hypotheses. [Isenberg 09].

## 2.4 Approach to Solutions

Analyzing a large document collection by going through the documents one by one is inefficient. We think there is need for a different approach to search through a large document collection. We want to provide an overview of the whole document collection with a zoomable user interface [Bederson 94]. A zoomable user interface provides an overview of the collection in which the user can change the scale of the viewed area in order to see more detail or less, and browse through the collection. Space filling thumbnails could be used to visualize the collection in a structured and efficient way, the whole screen will be used to visualize the collection. Instead of using an image as thumbnail representation we want to create semantic thumbnails [Dunsmuir 09] that can change the amount of information to display, see Figure 2.6. We will use text information, images and a tilebar of the document to create a representation that creates the best possible summary in the available space and to allow

smooth transitions between the different zooming levels. Multi-touch interactions will be used to create a more natural way of interacting with the application than used to in regular desktop applications.

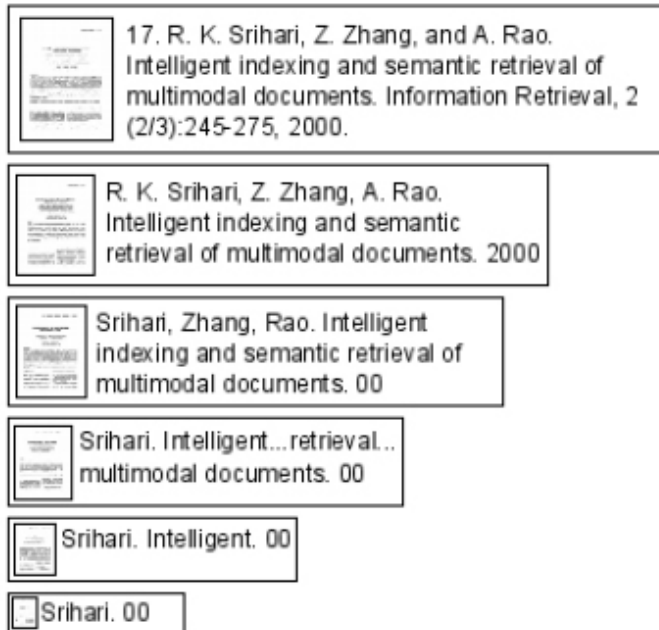


Figure 2.6: With semantic zoom levels you can specify for each zoom level how much information should be visualized [Dunsmuir 09].

## Chapter 3

---

# User Interface Design

The literature study [Ridder 11] and domain analysis provide us many ideas about how to visualize large document collections. During an experimental phase we wanted to get familiar with the Microsoft Surface Touch Beta API and in the meantime create several small applications to solve problems related to visualization, navigation and interactions. Before we started, we first defined the functions and tasks of the application. We have used these to create our first user interface sketches for the application. In this chapter we focus on the visual experiments and in the next chapter we will focus on the interaction experiments.

This chapter starts with an overview of the functions and tasks of the application in Section 3.1. Section 3.2 describes some multi-touch principles that limit and change the way we design a user interface. Section 3.3 presents the results of our sketches. Section 3.4 describes how we have evaluated the sketches and improved them. Section 3.5 explains different visualization techniques for large document collections. Section 3.6 describes how an efficient thumbnail representation for each document for different zooming levels can be created. This chapter ends with the conclusions of our user interface design.

### 3.1 Functions

The main task of our prototype is to create a collection of the two most relevant patents to a patent application, which are used as basis for the final written report. These patents are selected from a collection up to 1000 patents obtained by a search query. To achieve this goal, the user must perform different tasks and functions. The tasks and functions are categorized in two groups: patent collection browsing and single patent browsing. The first group contains tasks and functions applied on a patent collection and the second group contains tasks and functions applied on a single document. The tasks and functions of the single patent have many similarities with the current application. But instead of controlling the applications with mouse input, they will be driven with touch input.

### 3.1.1 Patent Collection Browsing

The following functions and tasks belong to the patent collection browsing group.

- **Browse and navigate** From the literature study, it was clear that a zoomable user interface was the best solution to visualize a large patent collection. A zoomable user interface provides an overview of the patent collection but also allow zooming in on patents. The user will use zooming and translation gestures to browse and navigate through the patent collection.
- **Add annotations** The end-users are already familiar with annotations. In the current application, they only use annotations in the Viewer application. Annotations help to give quick insight about the relevance of a patent and the important parts of the patent. The user should be able to add new, change and remove annotations. New annotations are added by defining a specific highlight color for a keyword.
- **Create subcollections** By providing an overview of up to 1000 patents, the user might not know what patents should be read first. The number of patents can be decreased by creating a subcollection based on a property. Analyzing a subcollection helps to focus on one topic at the time. Examples to create a subcollection are: get all patents of one classification, get all patents of one inventor, get all patents of one applicant, get all patents between two dates and finally by creating a patent collection manually.
- **Sort collections** Some subcollections may contain a large number of patents. The number of patents can be further decreased by refining the search query or by ordering the patents. Ordering the patents helps indicating which patents should be studied first. The user can use different properties to sort a patent collection. Some examples to order a patent collection are: order by a ranking, classification, time showed, applicant and year. This will display the most relevant patents, based on the ordering, on one side of the screen.
- **Visually compare patents** In the current application it is not possible to directly compare two patents with each other. The examiner has to open them one by one and remember or write down what they are about. With a zoomable user interface it is possible to visually compare the meta information of two or more patents with each other by zooming in on the patents or by creating a subcollection. The patent examiner can zoom in on the collection until four patents are displayed on the screen. Each patent is displayed in one corner and the most important meta information of the patent will be displayed.

### 3.1.2 Single Patent Browsing

The following functions and tasks belong to the single patent browsing group.

- **Scan patents** The user should be able to visually scan the patent in order to get a first impression about the relevance of the patent. An annotation bar gives quick insight about the frequency of keywords of interest and where they are located in the document. The annotation bar allows direct jumping to the location of interest and reading that section in more detail. In some fields, images are also very important for document scanning. An image browser is needed to be able to browse through the drawings.



- **Analyze patent in detail** If a patent is relevant to the application, the user should be able to analyze the patent in detail. We will need a full-text information reader and an image browser in order to analyze the patent more thoroughly. In the full-text information reader text can be read; keyword(s) can be highlighted and it should be possible to add notes to the patent. The image browser is needed to browse through the drawings and open a single drawing in order to watch it in more detail.
- **Highlight words or sentences** Highlighting of words and sentences is very important in order to analyze a patent document. The user should be able to select keyword(s) in the text viewer and use these to highlight or use them as notes. While writing the final report for a patent application, the highlights and notes can be used to remember what a patent was about or to directly go to the important parts of the patent.
- **Go to previous or next patent** In the patent overview, the user can use flick gestures to go to the previous or next patent in the collection. We also want to include patent navigation in the single patent viewer. The user should be able to analyze a collection of patents in more detail without going back to the overview.

## 3.2 Multi-Touch Input

Previous applications are mostly designed to be controlled with mouse inputs. These interfaces contain many instrumental GUI elements or widgets, such as scrollbars, buttons, tabs and bars. This results in a large number of clicks to be made. Designing a multi-touch application needs a different strategy. This section describes different aspects of designing a multi-touch user interface.

### 3.2.1 Mouse vs. Touch

There are several differences between mouse and touch inputs. These differences have effect on the design of a user interface for a multi-touch application.

- **Precision** The mouse pointer is a very small icon that can be used to precisely click on the different elements in the user interface. Touch input on the other hand will be controlled by your fingertips. The position of fingertip will be based on the center of your fingertip, which differs each time and for each user. It is hard to decide where your fingertip is exactly positioned on the screen, the so-called "fat finger" problem [Siek 05] which makes touch input less precise than mouse input.
- **Blocking view** A finger does not float transparently in space like a mouse pointer. The size of the mouse pointer is very small so the user is able to see almost everything on the screen. However, with touch input the user will not be able to see the screen covered by your finger, hand and arm.
- **Focus points** Mouse input only contains one focus point. The focus point can be controlled by moving the mouse and using the two mouse buttons. The number of click combinations with a mouse device is limited. A multi-touch input device makes use of multiple focus points. This allows making more advanced combinations of finger inputs.

### 3.2.2 Layout

Designing a user interface for a multi-touch application needs a different approach as we are used to. The differences between mouse and touch input have impact on designing a user interface for a multi-touch application. In current desktop applications, buttons may be smaller than a fingertip which makes them hard to click. In a multi-touch application you want to get rid of most buttons or increase the size of the buttons to make them better clickable. Using touch input blocks the view behind your finger, hand and arm. To block the view as less as possible we have to rethink the user interface and change the position of certain elements. The location on which the elements should be placed depends on whether the user is left handed or right handed. For a right handed user the elements should be displayed on the right and bottom side of the screen. This will prevent blocking the view of the rest of the screen.

### 3.2.3 Ergonomics

The default computer set-up uses a mouse and keyboard as input device and a monitor as output device. The ideal location of the monitor is placing it in front of the user and the ideal angle of the keyboard (input device) is 15 degrees. In a multi-touch set-up both the input and the output will be controlled by the multi-touch screen. Placing this screen in the standing position requires stretching your arms, while placing the screen flat requires watching down. Both ways can cause pain in your arms or neck. One possible solution is to combine two screens, one touch screen is placed flat as the input device and one screen is placed standing as the output device. Another solution would be an adjustable angle of the screen, this might also ease transition between single-person and collaborative use. For short sessions the position of the multi-touch device may have little impact, but especially for long session more research on ergonomics is needed.

## 3.3 Sketches

Before we started implementing our ideas, we first created some sketches. These sketches provide us improvements early in the development process, before we have spent time in the implementation. Because of the lack of completeness of the sketches, it is easy to deal with additions and changes [Arnowitz 08]. Sketches are created quickly which do not have to be perfectly drawn. It can be understood by people from different fields and since the sketches are far from perfect and easy to change, the audience will quicker give real feedback about the prototype. You can even subtract parts from the prototype or draw new parts for the prototype. Sketches even create the first possibility to test some interactions with the prototype. One of your fingers can be used as mouse pointer and by clicking on specific parts of the prototype other parts can be positioned over the prototype.

Our first sketches are created based on our findings during the workshop, the functions and tasks of the application and the multi-touch principles. The sketches are grouped into three different groups: the user interface of the patent collection viewer, the user interface of the single patent viewer and different thumbnail representations.

Figure 3.1 provides an overview of different ways to create a thumbnail representation of a patent document. For one patent document we created different thumbnail layouts in which we combined text and images. We changed the position of the elements but also the size of the thumbnails. We highlighted the thumbnail representation used as starting point of the application. In our literature study we found proofs that images should be combined with text to create a good thumbnail representation. During the project the thumbnail representation is improved by adding more images and by changing the annotation icon. Figure 3.2 provides different user interfaces for the patent collection viewer. The focus of these user interfaces was on displaying the patents and at the edges there is space for buttons and a legend. In a later phase of the project we decided to focus on a minimalistic user interface with only a few buttons and the focus on the content itself. The chosen visualization is highlighted in the Figure. Figure 3.3 provides different user interfaces for the single patent viewer. All these user interfaces include the three most important aspects of the viewer: a text browser, an image browser and an annotation bar, but display them in different ways. Again we choose the most minimalistic user interface as starting point. During the project the interface is improved where needed.

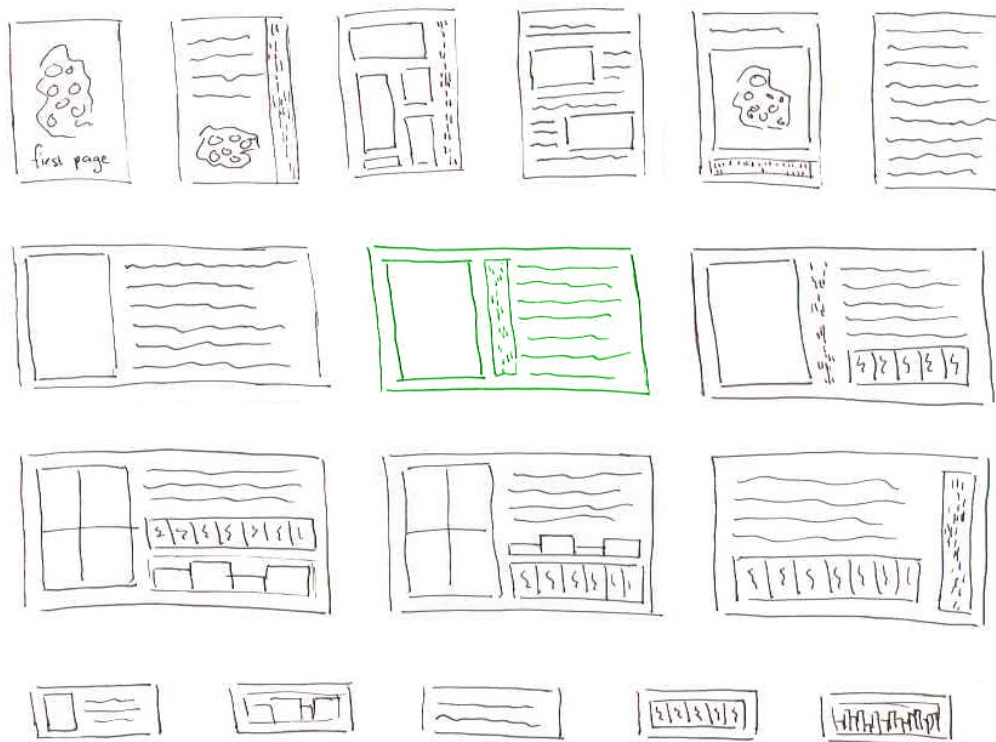


Figure 3.1: For one patent document we created different thumbnail layouts by changing the size and combining text and images in different ways. The highlighted thumbnail representation was used as starting point.

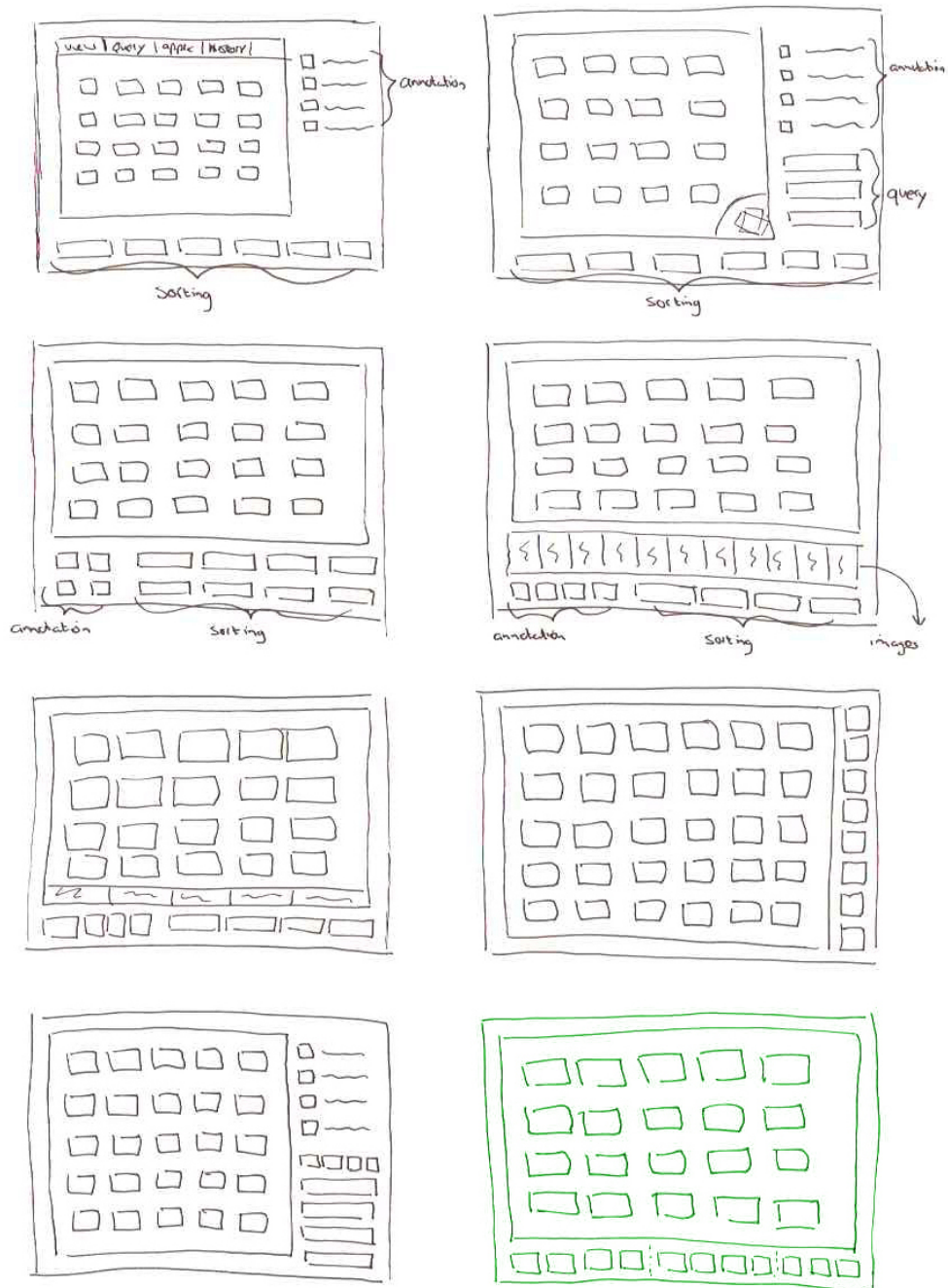


Figure 3.2: Different sketches for the user interface of the patent collection viewer based on the functions and tasks of the application. The focus was on displaying the patents and at the edges there is space for buttons and a legend. The highlighted user interface was used as starting point.

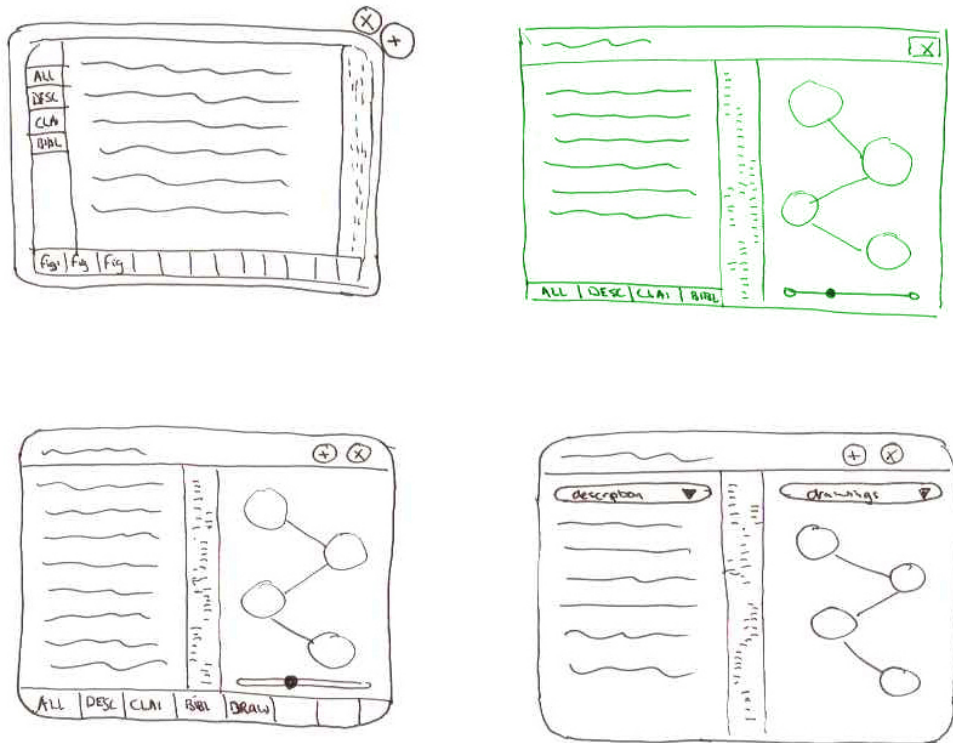


Figure 3.3: Different sketches for the user interface of the single patent viewer. They all contain the three most important aspects of the viewer: a text browser, an image browser and an annotation bar, but display these in different ways. The highlighted user interface was used as starting point.

### 3.4 Improvements

The user should be able to do many different tasks in the patent overview. In user interface designs for desktop applications we are used to adding controls or elements for each action. Therefore, our first sketches automatically lead to user interfaces that contain many controls and elements. In order to click these controls and elements properly with touch input we need to increase the size of the elements. Instead of increasing the size of the controls, we removed the elements and created more space for the actual content of the application. We have added the controls to a *context menu* in order to keep the functionality of the removed elements. The context menu will only be visible when your finger is for a few seconds on the screen. We did not remove all actions from the interface, since some need to be accessible from all viewers.

We have used the interface of the current Viewer application as a basis for our single patent viewer. We have optimized the user interface to work with touch input, by increasing the size of buttons and removing unnecessary controls and elements. The first sketches still

contained more elements as desired. The sketches of the single patent viewer are visible in Figure 3.2. To further decrease the number of controls and elements we have made some additional changes to the user interface. We have combined the text from all parts into one document, instead of having three buttons to display the bibliographic, description and claims section separately. For each section a label is added to the annotation bar. These labels help indicating which part of the document is visible and to easily switch to a different part of the document. We have also removed all image manipulation buttons and used touch input to browse through the drawings or manipulate a single drawing. It feels more natural when visual feedback on the drawing is provided, but it also leaves more space available to visualize the drawings.

The thumbnail sketches show different thumbnail representations for a patent document by combining text, images, annotations and using different layouts. The sketches are visible in Figure 3.3. A good representation of the patent requires displaying the most important meta information first and the least important information as last. We have defined a set of rules which determines the order of displaying all the meta information. These rules are based on the results of questions with examiners, the available space and the display order used in the current application. For each zooming level a different thumbnail representation is created and each representation tries to create the best summary possible in the available space.

## 3.5 Document Collection Visualization

Our application should give an overview up to 1000 documents. Research in the field of document visualization has shown different solutions to visualize document collections. We have chosen two visualization techniques that fit the best for our problems and worked these out into two small applications. Our first choice is space filling thumbnails, since this method makes optimal use of the available space. Our second choice is the list view representation, since this method has similarities with the current way of searching for relevant patents. Finally, we combined the good elements of both methods into a single visualization. The small applications created for testing purpose directly provide us feedback about how these methods work with touch input.

### 3.5.1 Space Filling Thumbnails

Space filling thumbnails provide an overview of the document collection, by visualize each document with a unique thumbnail [Cockburn 08]. The thumbnails are placed screen filling, by placing them next to each other. In the most extreme case the screen is divided into thumbnails of one by one pixel. On an average screen resolution of 1280x1024 pixels there is space for 1310720 pixels/documents. However, these thumbnails are so small that the user won't be able to recognize the document or read information from it. Increasing the resolution of the thumbnails improves the recognition of the document based on their colors and images. The larger a document collection is, the smaller the thumbnails will be. Interaction techniques like hovering provide the user additional information about the document. Hovering the thumbnail will display a larger window with more details of the document. We experimented with space filling thumbnail in combination with hovering by creating an

overview of our scientific paper collection. The result of this experiment is visible in Figure 3.4. This technique is very effective for papers, since they can easily be recognized by their colors and images, even when the thumbnail is small. In patent documents the images are black and white, which make it much harder to recognize them by their images. Small black and white images look very similar and are hard to distinguish. For patent documents we need to show additional or other information to be able to recognize it. The hovering effect can be activated both by touch input and mouse input. With mouse input the popup will be displayed by entering the image with the mouse cursor, while with touch input you have to hold your finger on the screen. Holding your finger on the screen will limit the number of other actions you can do with touch input.



Figure 3.4: Space filling thumbnails are used to create an overview of the patent collection by making optimal use of the whole screen. This figure provides the result of an experiment with space filling thumbnail. By hovering the thumbnails, a larger window with details of the document will appear.

### 3.5.2 List View

The list view is a visualization technique which can be used to visualize much information. In our case, for each document in the collection an item is added to the list view. The documents are visualized with meta information (i.e.: title, author, year and thumbnail) and images. We have experimented with list views by creating a list view of our scientific paper collection. The result of this experiment is visible in Figure 3.5. Each item in the list provides a short summary of one of the documents from the collection. This summary helps recognizing a document, but also gives insight about the relevance of the document. It is important to choose wisely what information should be displayed. A title says more about the document than a document number does. Small document collections can be analyzed quickly, by scrolling through the list and scan the meta information. For large document collections a list view is less useful. Going through the whole list is a time consuming job and information might be skipped when scrolling too fast.



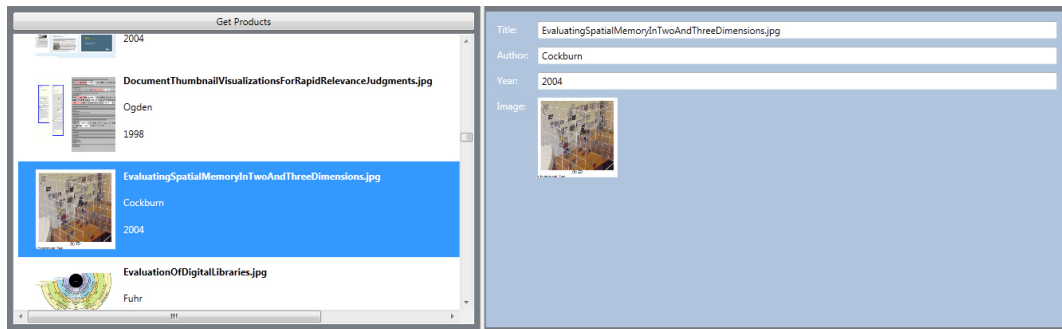


Figure 3.5: In a list view representation of a document collection, each document is visualized by its most important meta information. This figure provides the result of an experiment with a list view.

### 3.5.3 Results

Both visualization techniques have shown useful properties to visualize large document collections. Spacing filling thumbnails makes optimal use of available space by filling the screen efficiently with thumbnails. For articles, these thumbnails give the user much information about the document, but for patents additional information is needed. The list view has shown that meta information combined with images provides a good summary of the document. Recently strong support is found that figures from articles should be shown alongside the meta information [Divoli 10]. We have combined the good points of both visualization techniques in one visualization. We have visualized the document collection space filling by visualizing each document in the collection with a symbolic representation of the document. See Figure 3.6. For small thumbnails, only limited amount of space is available to create a good summary of the documents. By zooming in on the overview, more space will be created, which can be used to add additional information of the documents. Section 3.6 describes how semantic thumbnails can be created.

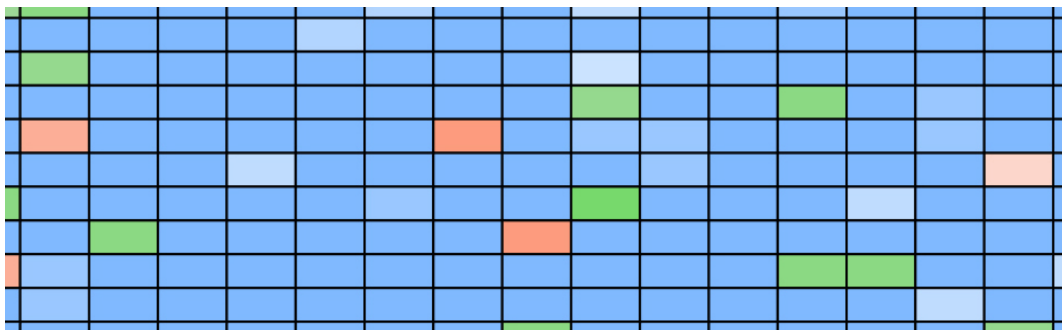


Figure 3.6: A space filling symbolic representation of the patent collection in which each document is visualized by a rectangle and a background that indicates the classification of the patent.



## 3.6 Thumbnail Visualization

For large document collections, only limited space is available to add information to the thumbnails. Each document is represented by a rectangular object. Instead of using only text and images we can also use the properties of the object to add relevant information to the thumbnail representation. Figure 3.7 provides an overview of the design space (the different properties that can be used to add information to the representation). In (a) the default representation is visualized. In (b) we changed the border color (For example, to indicate whether a patent is selected). In (c) we changed the border thickness (For example, to indicate the number of pages of a patent). In (d) we changed the background color (For example, to indicate the classification of a patent). In (e) we added an icon to the thumbnail (For example, to indicate whether the patent has been read). To recognize a patent or say something about the relevance of the patents we can add an image of the first page of patent (f), add one of the drawings (g) or add text to the thumbnail representation (h). The text element itself also contains different properties that can be used to add more information: the font size, font color, font weight and background color. In (i) we created an abstract representation in the thumbnail.

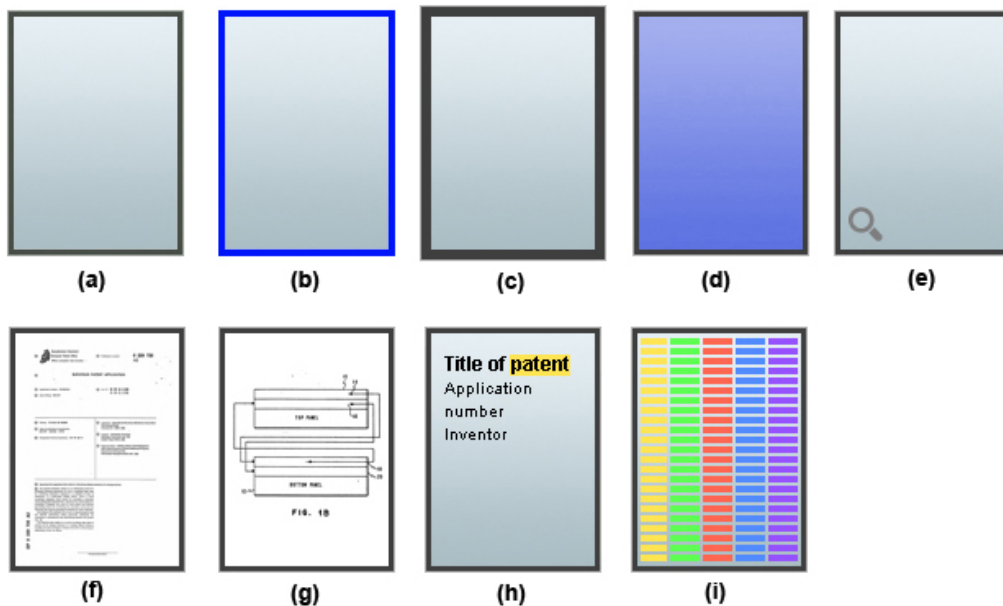


Figure 3.7: The design space of the symbolic representation contains different elements that can be used to add information to the thumbnail.

### 3.6.1 Annotation

A very important property that can be used in the thumbnail representation is annotations. Keywords are highlighted with a specific color. This is a strong visualization technique

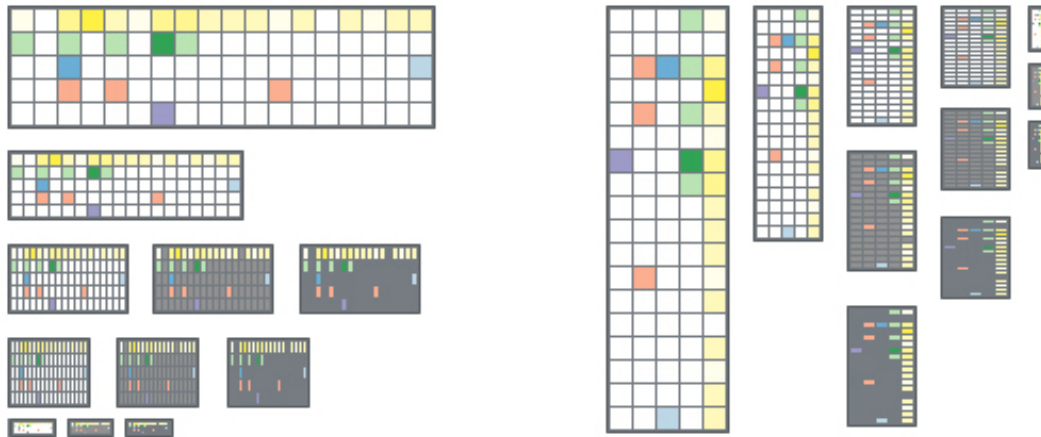


Figure 3.8: Different types of tilebars are created by changing the layout, size and background color. The tilebars are generated based on five predefined annotation groups, each group contains one or more words and a highlight color.

which gives quick insight about the relevance of the document. Annotation is something the examiner is used working with. In the current Viewer tool annotations are only used in the text viewer and the annotatiobar. We think that these annotations could be useful in the thumbnail representation as well. We want to use the annotations to create a tilebar [Hearst 95] of the documents. Tilebars are an iconic representation of the document based on the frequency and location of specified keywords. To generate a tilebar for a document, we need to split up a document into several parts. In each part of the document, the frequency of the keywords is counted. Based on the number of occurrences, a colored rectangle is added to the icon. The more frequent a keyword occurs, the darker the color will be.

In an experiment we have generated different tilebars based on five different annotation groups for one document. Each annotation group contains a highlight color and one or more keywords. We have generated different types of tilebars by changing the layout, size and background color. The result of this experiment is visible in Figure 3.8. We directly noticed that the smallest tilebars are too small to clearly see the colors. The larger tilebars are clearer and directly show important information of the patent document. For example, some keywords do not occur in the patent or what the most important parts are.

### 3.6.2 Semantic Zoom Levels

Section 3.5.3 proposes a zoomable user interface where each thumbnail is represented with a symbolic representation. By zooming in on the overview the size of the thumbnails increases. The extra space created can be filled with extra information about the document. We used semantic zoom levels [Dunsmuir 09] [Sengupta 04] to generate different thumbnail representations for each zooming level. For example, the smallest thumbnails only contain text, while a large thumbnail contains both text and images. This allows us to create smooth transitions between the different zoom levels.

Semantic zoom levels can also be applied to the text itself. Bier [Bier 04] has proposed Text and Thumbnail (TAT) to semantically change the amount of meta information to visualize, based on the zooming level. The first step is to define different text variants for each type of meta information. For example, for the year of publication two different text variants can be defined:  $Y1$  is a two digit number and  $Y2$  a four digit number. For the author five different variants can be defined:  $A1$  is the last name of the first author,  $A2$  contains all last names,  $A3$  contains all last names + the first letter of the first names,  $A4$  contains all last names + the first names and  $A5$  contains all author information. For the title four different text variants can be defined:  $T1$  is the first word of the title,  $T2$  contains the three longest words,  $T3$  is the title without stop words (i.e.: the, a, an, etc.) and  $T4$  is the full title. Next, rules that decide what information should be displayed at what zooming level are defined. Table 3.1 shows an example of these rules and the results of these. This TAT representation can be used to efficiently fill the new space created by zooming in on the collection.

Rule	Result
A1+Y1	Strobelt. 10.
A1+T1+Y1	Strobelt. Document. 10.
A1+T2+Y1	Strobelt. Document ...Visualization Documents. 10.
A1+T3+Y1	Strobelt. Document Cards: Top Trumps Visualization Documents. 10.
A2+T3+Y1	Strobelt, Oelke, Rohrdantz, Stoffel, Keim, Deussen. Document Cards: Top Trumps Visualization Documents. 10.
A3+T3+Y1	H. Strobelt, D. Oelke, C. Rohrdantz, A. Stoffel, D. Keim, O. Deussen. Document Cards: Top Trumps Visualization Documents. 10.
A3+T4+Y1	H. Strobelt, D. Oelke, C. Rohrdantz, A. Stoffel, D. Keim, O. Deussen. Document Cards: A Top Trumps Visualization for Documents. 10.
A4+T4+Y1	Hendrik Strobelt, Daniela Oelke, Christian Rohrdantz, Andreas Stoffel, Daniel Keim, Oliver Deussen. Document Cards: A Top Trumps Visualization for Documents. 10.
A4+T4+Y2	Hendrik Strobelt, Daniela Oelke, Christian Rohrdantz, Andreas Stoffel, Daniel Keim, Oliver Deussen. Document Cards: A Top Trumps Visualization for Documents. 2010.
A5+T4+Y2	Hendrik Strobelt, Daniela Oelke, Christian Rohrdantz, Andreas Stoffel, Daniel A. Keim, Oliver Deussen. Document Cards: A Top Trumps Visualization for Documents. 2010.

Table 3.1: Text and Thumbnail (TAT) is used to semantically change the amount of meta information to visualize for each zooming level [Bier 04].

### 3.6.3 Choice of Colors

The background color in our thumbnail representation indicates the classification of the patent. At the EPO they make use of two classification systems: the International Patent Classification (IPC) and the European Classification System (ECLA). ECLA is an extension of the IPC and more frequently used at the EPO. The classification code consists of five different parts. For example, the classification code G11B7/00 is classified in Section G (Physics); classified in group G11 (Information Storage); classified in subgroup G11B (Information storage based on relative movement between record carrier and transducer); classified in sub subgroup G11B7 (Recording or reproducing by optical means, e.g. recording using a thermal beam of optical radiation); and finally classified in G11B7/00. There are eight different sections defined in the EPC. For each Section we used a color that highly contrasts the other colors. The high contrast is needed to create a better distinction between the different classifications. Colorbrewer<sup>1</sup> [Harrower 03] proposes a color map containing eight colors with high contrast. All colors are defined in RGB-values (Red, Green and Blue) and HSV-values (Hue, Saturation and Value). We also created a color map to indicate the classified group of the patent. For this, we used the HSV-values and kept the hue value the same and changed the saturation and value. By keeping the hue value the same, different colors tints for each color can be created. In Figure 3.9 the eight different colors of the classified Sections and eight different colors of the classified group are visible.

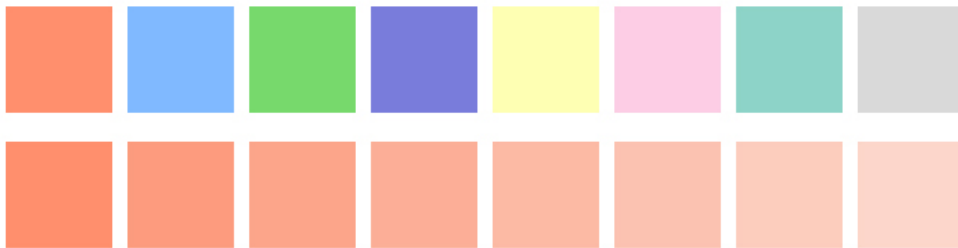


Figure 3.9: Overview of the selection of representative colors. The top row show eight different colors used to indicate the classified Section of a patent and the bottom row shows eight different colors used to indicate the classified group of the a patent.

### 3.6.4 Results

We have combined previous described information about the object space, annotations and semantic zoom levels, to create different types of thumbnail representations for one document. The result is five different thumbnail representations for the different zooming levels. These thumbnail representations provide us information about what information should be visualized and the effect of scaling the thumbnails.

---

<sup>1</sup><http://colorbrewer2.org/>

## Page image

This thumbnail is created from a picture of one of the pages. The first page of a document contains important information like the title, author and abstract of the document. A combination of this information gives a good summary of the document. An image of the first page allows directly reading of the information from the image. For small thumbnails the text and images become too small to be able to read it. For patent documents, which contain black and white images, these small thumbnails look the same and are therefore hard to distinguish. Besides the first page, a random page of the document can also be used as thumbnail. A random page gives worse results as using the first page. It is possible to get an image of the references page which is not the most relevant page from the document. The last option we want to discuss is to use the last opened page of the document. For documents that have not been opened yet, a random page can be used. Again, this representation works better for large thumbnails than for small thumbnails. See Figure 3.10.

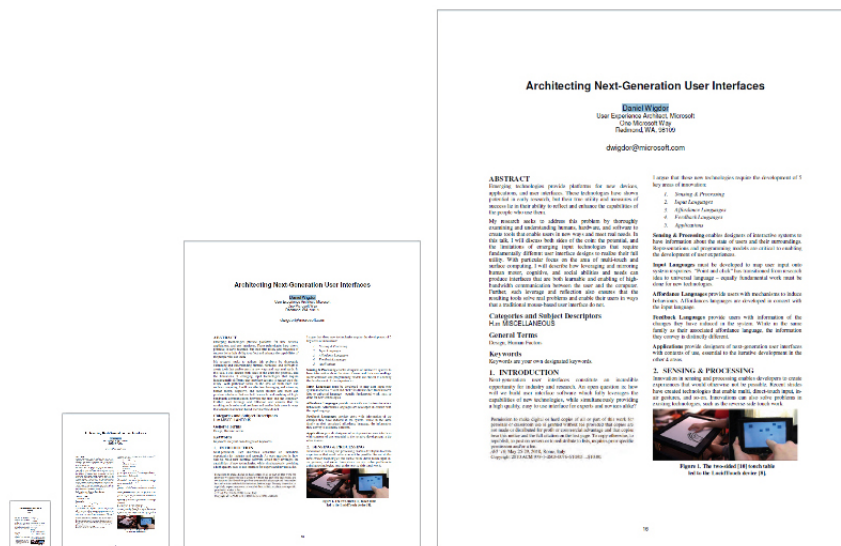


Figure 3.10: Thumbnail representation of an article in four different scales. We used the first page of the article as image.

## Images

This thumbnail is created from one of the images of the document. Using a random image from the document does not always provide a good representation of the document. It can happen that randomly a graph image is chosen. A graph image is not always very representative for the document, which makes it hard to indicate whether the document is relevant or not. For a patent document, the first image is more representative. The first image in a patent document is often chosen by the applicant to be the most representative for the patent. The examiner can also decide for himself what image should be used. For large document collections, it takes much time selecting an image for each document. Again, this representation works better for large thumbnails than for small thumbnails.

### Text only

This thumbnail is created from meta information like the title, author and abstract of the document. For different sizes of thumbnails, different representation needed to be generated to fill up the space efficiently. By increasing the size of the thumbnail, we will also increase the font size and the amount of information that is displayed. Large thumbnails will have much space available, that can be filled with much information about the document and creates a good summary of the document. Small thumbnails have little space available. This space is filled with text with a small font size, which makes the text hard to read. Increasing the font size does not solve this problem, because the available space is too small. See Figure 3.11.

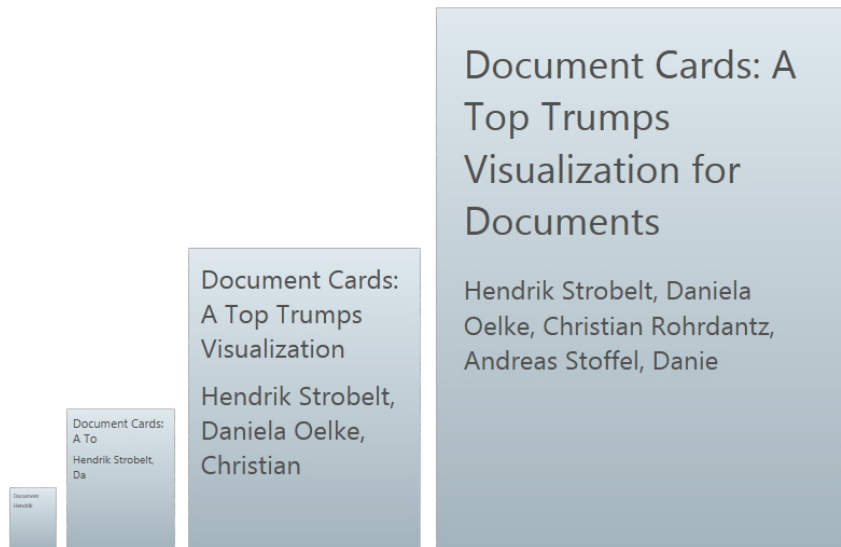


Figure 3.11: Thumbnail representation of an article in four different scales. The amount of meta information and font size will gradually increase when the thumbnails become larger.

### Text and image

This thumbnail is created from meta information and an image of the document. This representation is inspired by Bier [Bier 04]. The smallest thumbnail contains only meta information. By increasing the size of the thumbnails, the font size will be increased; the amount of information to be displayed is increased; and an image will be added. We have changed the layout of this thumbnail representation to make more efficient use of the available space. In the smallest thumbnails we have left out the images, since they will be hardly visible. The combination of text and an image gives a better representation of the document than the previous thumbnails. See Figure 3.12.



Figure 3.12: Thumbnail representation of an article in five different scales. The thumbnail representation combines meta information with images. The larger the thumbnail is the more information can be displayed.

### Text, images and icon

This thumbnail is created from meta information, an image and an iconic representation of the document. The difference with previous thumbnail representations is that it contains an annotation icon. The annotation icon gives quick insight about the occurrences of predefined keywords. The small thumbnails will only display the most important meta information like the title. Increasing the size of the thumbnails will increase the amount of information to be displayed; the size of the text; the size of the image; and it will add an annotation icon. The combination of text, an image and an icon provides a good summary of the document. This thumbnail can be helpful in deciding whether the document is relevant or to recognize it, see Figure 3.13.

## 3.7 Conclusions

Before we started creating and implementing a user interface, we first created a list of functions and tasks. We used these together with the multi-touch user interface principles as basis for our user interface sketches. These sketches are evaluated and improved based on our own experience and feedback from others. During an experimental phase we have experimented with the Microsoft Surface Touch Beta API and in the meantime solved little visualization problems related to the project.

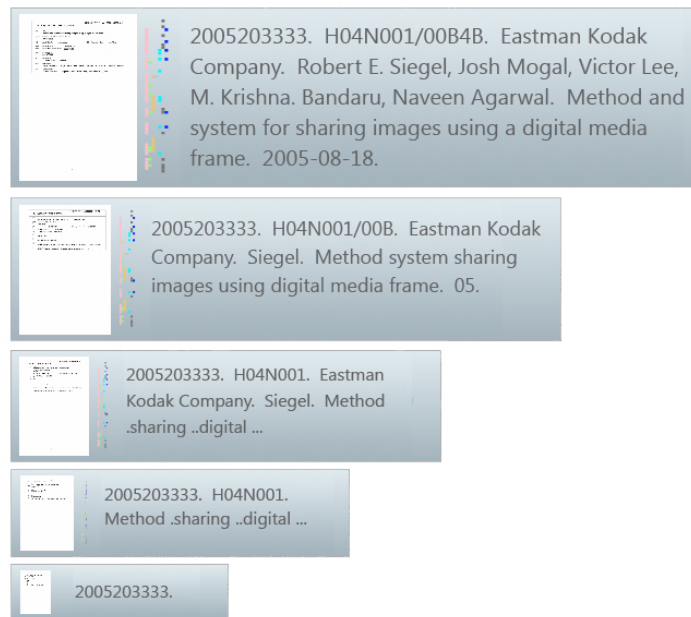


Figure 3.13: Thumbnail representation of an article in five different scales. The thumbnail representation combines meta information, images and an annotation icon. The larger the thumbnail, the more information can be displayed.

Designing a user interface on a multi-touch device requires an unusual approach. The most important differences between mouse and touch input are: touch input is less precise than mouse input, the view is blocked by your finger, hand and arm and multi-touch devices have multiple focus points. This affects the size of elements and the location of the elements on screen. We basically have to rethink the whole user interface and start from scratch. We should think wisely about the position of the elements, how we should visualize the elements and what gestures we use to activate actions. More research is still needed on the ergonomics of multi-touch devices and applications.

We chose 2D space filling thumbnails as visualization techniques for large document collections. Space filling thumbnails makes optimally use of the available space. Each document is visualized by a thumbnail representation that combines text, images and an iconic representation of the document. With Semantic zoom levels we can generate a different thumbnail representation for each zooming level. For each document, the best possible summary in the available space will be generated. In Chapter 4 the experimental phase continues, but we will focus on touch-based interactions there. We will combine the results of the touch-based interaction experiments with these results in Chapter 5 in a prototype.



## Chapter 4

---

# Touch-based Interaction

The development of multi-touch displays opens a whole new world of possibilities in the field of controlling an application in a more natural way. Multi-touch sensing enables a user to interact with a system with more than one finger at the same time. This creates the possibility to work with multiple users on a system at the same time. The idea of using a multi-touch device was already proposed in 1960 [Buxton 07]. Still a lot of research needs to be done on the user interface design and interactions. Previous work is mostly done on visual and physical ergonomics. This chapter focused on how to make use of natural interactions to browse and navigate through a patent collection with touch input.

This chapter starts with a description of the environment and tools required in Section 4.1. Section 4.2 explains the different multi-touch actions that we use in our application. Section 4.3 describes the main user interface and the basic navigation that can be used to control it. Section 4.4 describes how and why we have added constraints to the basic navigation. Section 4.5 describes how we experimented with the different functionalities of the detail patent viewer. Section 4.6 describes how we have created the context menu optimized for touch input. This chapter ends with some conclusions about multi-touch interactions in our application.

### 4.1 Framework

To build and test a multi-touch application we need to have a multi-touch input device. Our first idea was to make use of the Microsoft Surface<sup>1</sup>. One problem with the Microsoft Surface is that it is a large device which is hard to move around. This makes it difficult to test and demonstrate our application. Therefore, we have decided to make use of a multi-touch monitor. A multi-touch monitor is relatively cheap to buy and can quickly and easily be used for testing purposes. We will build our application on a Windows 7 system. There are several multi-touch frameworks available for Windows 7 that allow us to control the touch inputs. We chose to use the Microsoft Surface Beta API. With this API it is possible to quickly create visible results and with some minor changes it can also run on the Microsoft Surface.

---

<sup>1</sup><http://www.microsoft.com/surface/>

The Microsoft Surface Beta API contains a collection of resources which allow us to create touch-enabled applications using C# and Windows Presentation Foundation (WPF<sup>2</sup>). To create an application with the API, we need to have Windows 7, Visual Studio and Microsoft .NET 4.0 installed. The API support various input methods like: mouse, stylus and touch. Running an application build with this API requires to have Windows 7, Microsoft .NET 4.0 and the Surface Toolkit Runtime application installed.

## 4.2 Touch Input

The API makes it possible to control the touch events in the application. We can combine these events to create actions. These actions can be divided in two groups: single finger actions and multiple finger actions.

### Single Finger Actions

Single finger actions are similar to mouse actions. With a single finger input the following touch actions can be created:

- **Tab** The tab event will be launched by touching and releasing your finger. See Figure 4.1 (a). This event is similar to a left mouse click event.
- **Double tab** The double tab event will be launched by quickly tab your finger two times on the screen. See Figure 4.1 (b). This event is similar to a double mouse click.
- **Long tab** The long tab event will be launched by holding your finger for a few seconds on the screen. See Figure 4.1 (c). This event is similar to a right mouse click.
- **Gesture** The gesture event will be launched by holding your finger on the screen while dragging your finger. See Figure 4.1 (d). This event is similar to a hold-and-pressed left mouse button event and can be used to drag items or for making flick gestures to navigate through pages.

### Multiple Finger Actions

Multiple finger actions differs more from the standard mouse actions. Combinations of multiple fingers are used to create more advanced gestures. The multi-touch screen we use for our development allows a maximum of two fingers. Therefore, we will only focus on creating combinations of two finger inputs. With two fingers the following touch events can be created:

- **Scale** The scaling event will be launched by pinching two fingers together or apart from each other. See Figure 4.2 (a).
- **Rotation** The rotating event will be launched by touching two spots and twist them. See Figure 4.2 (b).

---

<sup>2</sup><http://msdn.microsoft.com/en-us/library/ms752059.aspx>

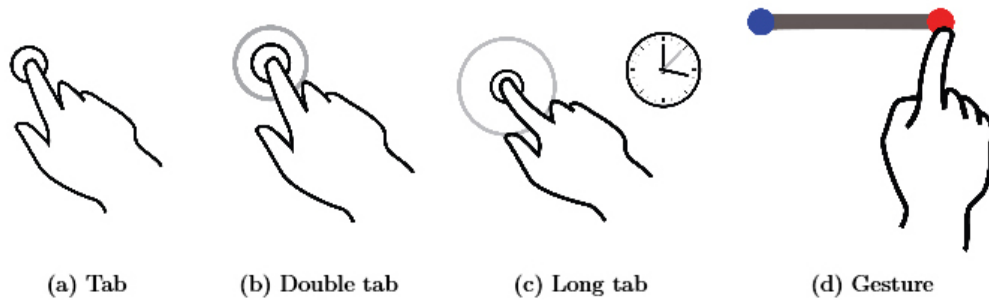


Figure 4.1: Overview of possible actions created with a single finger input. Illustrations provided by Gestureworks ([www.gestureworks.com](http://www.gestureworks.com)).

- **Translation** The translation event will be launched by holding your fingers on the screen and drag them to a direction. See Figure 4.2 (c).
- **Press-and-tap with a second finger** The press-and-tap with a second finger event will be launched by holding one finger on the screen and use another finger to tap on the screen. See Figure 4.2 (d).

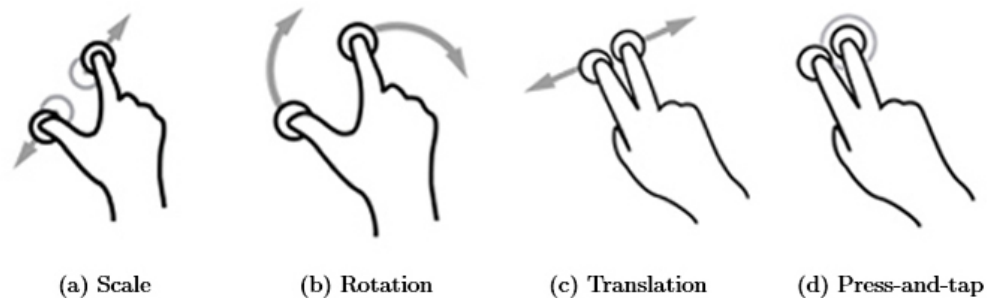


Figure 4.2: Overview of possible actions created with multiple fingers input. Illustrations provided by Gestureworks.

### 4.3 Basic Navigation

In Section 3.7 we proposed a zoomable user interface to browse and navigate through the document collection. A zoomable user interface creates a workable overview of the document collection and makes it possible to zoom in on a specific part of the collection, to get more information about the documents. The longest running effort to create a zoomable user interface has been the Pad++ [Bederson 94]. A zoomable user interface is a type of graphical user interface. Two essential characteristics of a zoomable user interface that affect their usefulness and their usability are [Bederson 09]:

- **Layout flexibility** The user has a creative control over the layout of information in the space. On one hand, user can put anything at any size and place. On the other hand, the environment might be constrained allowing information only in specific places according to a grid or layout algorithm.
- **Navigation mechanisms** There is a tradeoff between flexibility and usability if we look at how a user can move through space. Some interfaces allow users to fly through the space going absolutely anywhere. Almost every document browser and editor limits navigation to the available content. Some other interfaces allow to click on objects to zoom into them and click on a zoom out button to zoom out. This makes it impossible to get lost, but also giving less control over where to exactly look.

We extended the patent overview visualization, defined in Chapter 3, with scaling, rotation and translation gestures to browse and navigate through the document collection. These gestures are by default supported in the Microsoft Surface Beta API. To make use of these gestures, we need to activate the manipulation events. The first thing noticed, after adding gestures to our application, was that rotations have no additional value in our application. With rotations enabled, the interface will also rotate when we only want to zoom in. Rotations are useful for applications that can be approached from different sides. Our application only needs to be approached from one side. We decided to turn off the zooming gestures. The remaining functionality of a zoomable user interface is summarized in Figure 4.3. Each document in the collection is represented by a rectangle object. Only the patents that lay inside the sliding window are visible to the user. A zooming gesture increases or decreases the size of the sliding window and a translation gesture will move the sliding window through the collection.

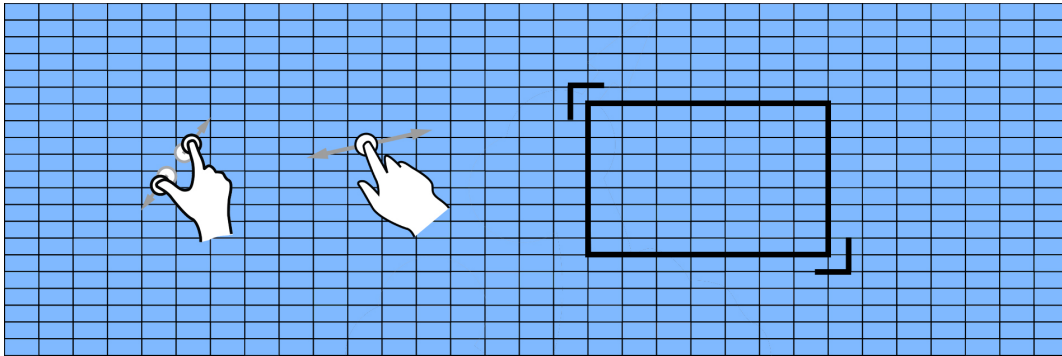


Figure 4.3: Freely explore the document collection with translation and scaling gestures. A zooming gesture increase/decrease the size of the sliding window and a translation gesture will move the sliding window through the collection.

## 4.4 Constraint Zooming and Translation

The user can use scaling and translation gestures to freely explore the document collection. Without any restrictions it is possible that thumbnails are partly visible on the edge of the screen. See Figure 4.4 on the left side. These thumbnails are not able to display all

their information and are therefore not able to give the user a good indication about the relevance of the document. The space can be used more efficiently by displaying only full visible thumbnails. To create a smooth way of exploring the documents, the user is allowed to freely transform the collection. When the user releases his fingers from the screen it will automatically snap to a hidden grid on the background. A linear transition is applied between the current location of the data and the closest cross point on the hidden grid. With flick gestures, the user can quickly move to the next or previous row and column of the grid.

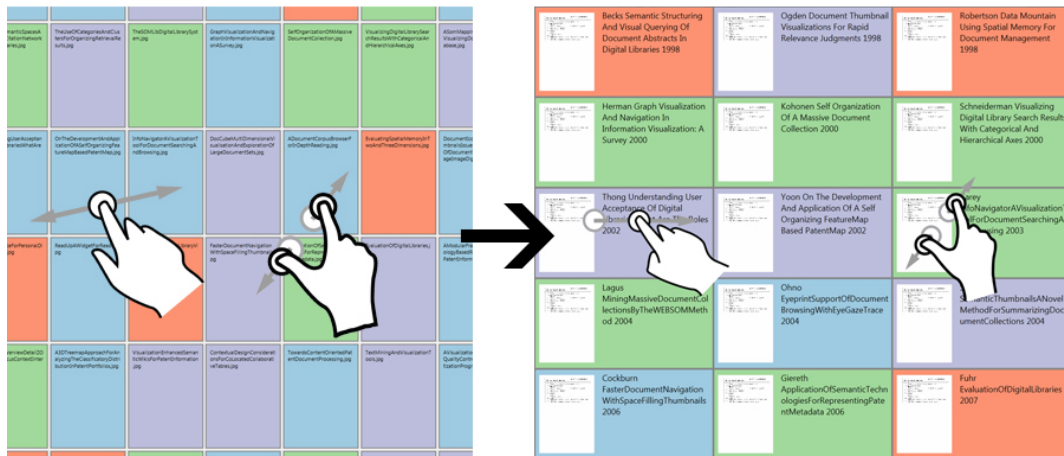


Figure 4.4: The available space can be used more efficiently by adding constraints to the basic navigation. Only fully visible thumbnails will be displayed on the screen.

We added similar constraints for the zooming transformation. Free zooming can result in partly visible thumbnails on the edge of the screen. Therefore, releasing your fingers after a gesture will automatically snap the data to the closest zooming level and apply a linear translation between the current scale factor and the target zoom level. We also added a maximum and minimum limit for zooming. Once the total document collection is visible there is no need to zoom out further. Similar for zooming in, once the application shows one document full screen there is no need to zoom in further.

## 4.5 Detail Patent Viewer

The user should also be able to analyze a single document in more detail. In the detailed document viewer, the full-text information and images of a document can be analyzed. This view will be similar to the document viewer that is used in the currently used viewer. We included some of the basic functionality of the current document viewer like: reading text, selecting text, highlighting keywords and/or sentences and browsing through the images. We improved and optimized these to work well with touch input. Each function is described in more detail below.

### 4.5.1 Text Browsing

Analyzing the full-text information of a patent is one of the most important aspects of the patent examination. By default multi-touch gestures are not enabled for a text field. We had to manually add the scrolling and selection functionality to the text field. We have made a distinction between vertical movements (for scrolling) and horizontal movements (for text selection). The user should be able to highlight keywords inside the text. To add new highlights to a document, we first need to be able to select keyword(s). There are some problems with text selection in combination with touch input. A fingertip is not very precise as selector and will block the view of your selection. This makes it difficult to start and end the selection at the desired positions. One solution is to increase the font size of the text. Increasing the font size is not the best solution since it will decrease the amount of text visible on the screen and it will look ugly. A better solution is to show a popup which displays the information behind your fingertip. In Figure 4.5 three examples of text selection are visible. This preview makes it easier to start and end the selection at the desired positions.



Figure 4.5: Three examples of selections made with touch input. For each a large preview of the selection is visible right above the fingertip.

### 4.5.2 Annotations

Annotations are used to quickly scan a document, to check whether it is relevant or not. We want to display the annotations both in the text viewer and in the annotation bar. When a keyword from the annotation groups occurs in the text it will be highlighted in the text and be visible with a stripe in the annotation bar. The annotation bar provides more information about the frequency of the keywords and where they occur in the document. This bar helps deciding what parts of the document should be read and which can be skipped. We have created a small application to test the text viewer functionality with touch input and improved it where needed. In Figure 4.6 the result of this experiment is visible.

We defined five annotation groups, each contains one or more keywords and a highlight color. The document is scanned, by checking for each word whether it is equal to one of the keywords in the annotation groups. When a match is found, the background color of the word is changed in the corresponding highlight color. The annotation bar is created in a similar way. For the annotation bar we also need to keep track of the row number where

the match is found. This row number is needed to display the annotation stripe on the right location in the bar.

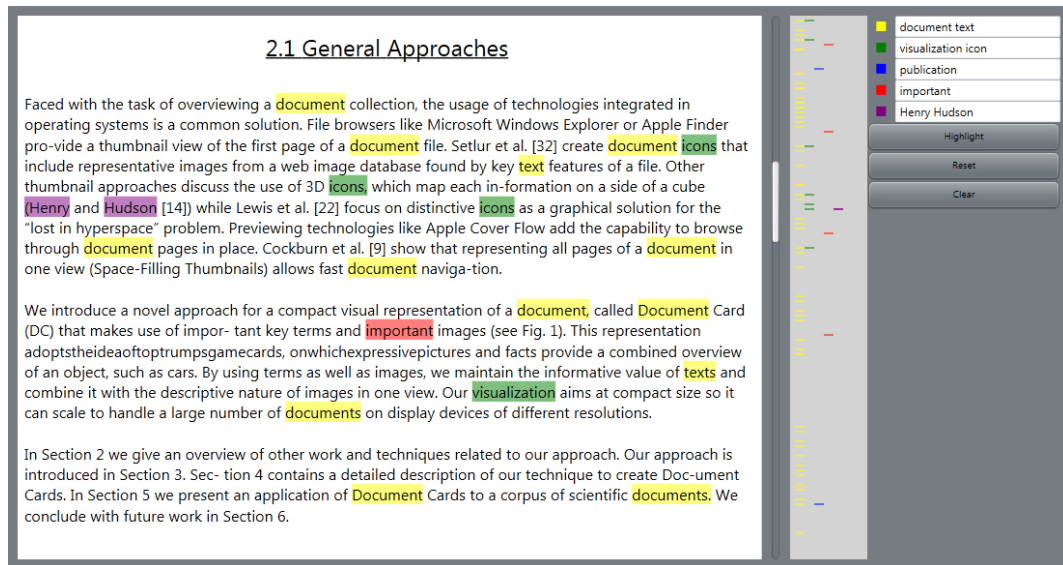


Figure 4.6: Predefined keywords are highlighted both in the text viewer and the annotation bar. Together they give quick insight about the relevance of the document.

### 4.5.3 Image Browser

The last functionality we want to include in our detailed document viewer is an image browser. Most of the patents have a drawing section which contains drawings about the invention. It depends on the field of the examination whether the drawings are useful or not. We have experimented with two different ways of browsing through the drawings.

The first method provides an overview of four drawings from the patent. Horizontal flick gestures are used to go to the next or previous four drawings. Vertical flick gestures are used to switch between a single drawing and four drawings. This method does not provide visual feedback from the gestures on the drawings and might be hard to understand without any experience. On the left side of Figure 4.7 the first method is visible.

The second method provides one drawing at the time, but includes visual feedback of the gestures on the drawings. All the drawings are placed on a horizontal row and only horizontal movements are allowed. Holding your finger on the screen and move it side wards, will have a direct effect on the position of the drawings. Due to the visual feedback of moving drawing, this method is better to understand by new users. On the right side of Figure 4.7 the second method is visible.

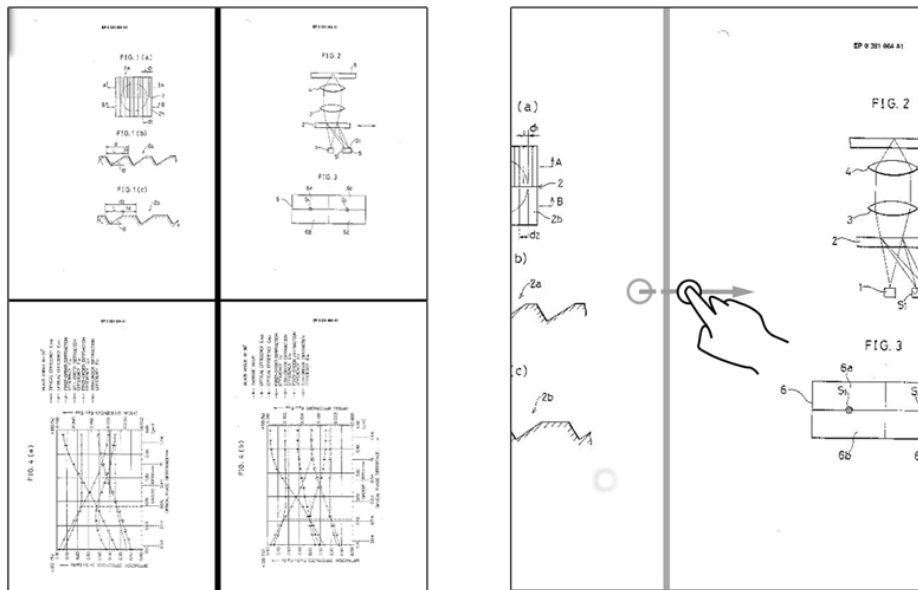


Figure 4.7: Two different methods to browse through the drawings. The left method returns no visual feedback on the drawings and the right method returns visual feedback on the drawings.

## 4.6 Context Menu

Our user interface contains a limited number of controls and elements. However, users should be able to do many actions such as deleting, sorting and grouping. To prevent adding extra buttons to the user interface and still being able to start the actions, we used a context menu. The regular context menu contains a list of items that need to be precisely clicked. These types of menus are hard to control with touch input. We changed the default context menu to work more efficient with touch inputs.

### 4.6.1 Basic Menu

We started with a basic context menu that works with both touch and mouse events. To open the context menu the user should hold his finger for two seconds on the screen. Instead of creating a list we have used circles which are displayed around your finger. This prevents blocking the view with your finger. The context menu is built with menu items and submenus. A menu item starts an action and a submenu will open a new menu, which again contains menu items or submenus. You have to release your finger on one of the circles in order to open the desired action. In Figure 4.8 the basic context menu is visible. Controlling this context menu with mouse input is similar to the context menus in regular applications. The user uses a right click to open the context menu and uses a left mouse click to start the corresponding action. The larger the menu is the less space is available to draw the menu. This is solved by making the circle smaller once you get deeper in the menu, but this also makes it harder to select the actions with touch input.





Figure 4.8: The basic custom context menu requires precisely clicking on the circles in order to active an action.

### 4.6.2 Gesture Based Menu

One problem with this menu is that it requires precisely clicking on the circles, which is hard with touch input. Gesture movements are much easier and quicker. OctoPocus [Bau 08] is an example of a dynamic guide that combines on-screen feed forward and feedback to help users learn, execute and remember gesture sets. OctoPocus inspired us to change the click event for gestures. In OctoPocus all possible gestures are visualized. The number of possible gestures decreases when you start making one of the gestures. Instead of showing all possible options, we decided to create a line that needs to be crossed in order to activate an action. In Figure 4.9 the extended context menu is visible. Crossing one of the lines will start the corresponding action or opens a submenu.

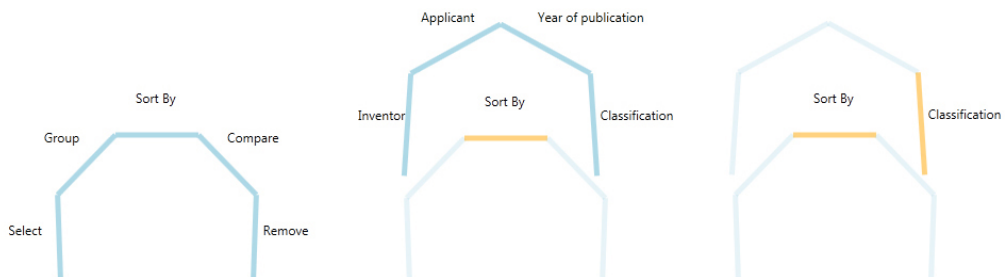


Figure 4.9: The circles in the basic context menu are replaced by lines which need to be crossed with gesture movements.

Once the menu gets bigger the submenus starts to overlap each other and it become unclear what actions are possible. We have structured the context menu more efficient by making use of a hexagon shape. Hexagons can be efficiently connected with each other without overlapping each other. In Figure 4.10 the hexagon based context menu is visible. In this

menu we will use five edges to start an action and one to correct the action. Menus that contain more than five options requires different solutions. For now, we have organized the menu in a way that it contains a maximum of five actions.

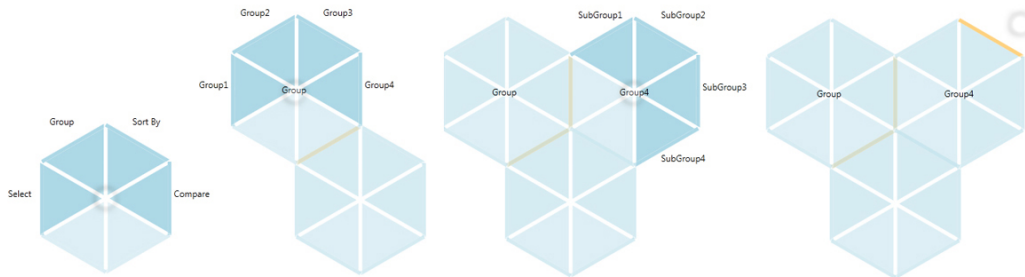


Figure 4.10: The final context menu requires gesture movements to start an action and is built with hexagons to structure them more efficiently.

## 4.7 Conclusions

We decided to build our application with the Microsoft Surface Touch Beta API. This API makes it possible to quickly create visible results and with minor changes the application can also run on the Microsoft Surface. We have limited ourselves to use one and two finger interactions only, since our multi-touch device does not allow more. The touch based interaction experiments are started by extending the visualization technique proposed in Chapter 3. We started by adding basic multi-touch interactions like zooming and scaling to the zoomable user interface. This allows the user to freely explore the patent collection. One problem of letting the user freely explore the collection is that it is possible that thumbnails are partly visible on the edge of the screen. We have added constraints to the translation and zooming to always display the full thumbnails.

The user can open a patent in the detail patent viewer to study it in more detail. This viewer contains three different areas: text viewer, annotation bar and an image browser. Each has its own problems and difficulties. During the experimental phase, we have tried different solutions. For each solution we checked how it worked with multi-touch interactions. We also created a custom context menu that can be efficiently controlled with multi-touch input. It is a hexagon based menu that activates actions by making gesture movements. An interesting aspect of this context menu is that the user starts to remember which gesture movements result in which actions. This makes it possible to quickly make gesture movements to start an action.

The visualization and techniques described in this Chapter are part of the experimental phase of the project. Chapter 3 focuses on the user interface aspects and this Chapter focus on the touch based interactions. In Chapter 5 we combined the results of both into a prototype.

## Chapter 5

---

# The Prototype: TouchPat

We have combined the best solutions of the experiments described in Chapters 3 and 4 into one application *TouchPat* and evaluated it with end-users and experts. Touch input requires unusual design strategies, changes were needed to let the techniques work well with touch input. We used the experience gained from the experiments in our final prototype. One example of an improvement is to make multi-touch interactions run more smoothly. We have combined the best solutions of the experiments and evaluated it with end-users and experts. *TouchPat* visualizes the patent collection with 2D space filling thumbnails in a zoomable user interface. In a zoomable user interface we can semantically change the amount of data to visualize to create smooth transitions between the different zooming levels. We used multi-touch interactions to create natural interactions with the application. Constraints are added to the basic navigating to prevent displaying partly visible thumbnails on the edge of the screen and with our custom context menu the examiner can quickly start an action.

This chapter describes the functionality of the final *TouchPat* prototype. This chapter starts with a description of *TouchPat* in Section 5.1. In Section 5.2 we describe how we have obtained feedback on our prototype. Section 5.3 provides an overview of the discussions during the evaluation. This chapter ends with conclusions on *TouchPat*.

### 5.1 Results

In *TouchPat* a patent examiner can create a stack of two to five relevant patents from a predefined collection up to 1000 documents. *TouchPat* contains three different views: patent collection view, detail patent view and the intermediate thumbnail views. If we open *TouchPat*, it will directly provide an overview of the patent collection in which each patent in the collection is visualized with a symbolic representation of the document. All views contain as few controls and elements as possible to put the focus on the contents.

#### 5.1.1 Patent Overview Viewer

The interface of the patent overview is visible in Figure 5.1. Most of the space is used to visualize the patent collection by a thumbnail representation. At the bottom of the interface

a bar is visible that contains functionality that can be used in all the views. It contains a legend and a button to add and update the annotation groups. Holding your finger on the screen, will open our custom context menu. In this menu gesture movements are used to start actions like; grouping, sorting and removing. The user can also use the context menu to open a single patent in the detail viewer.

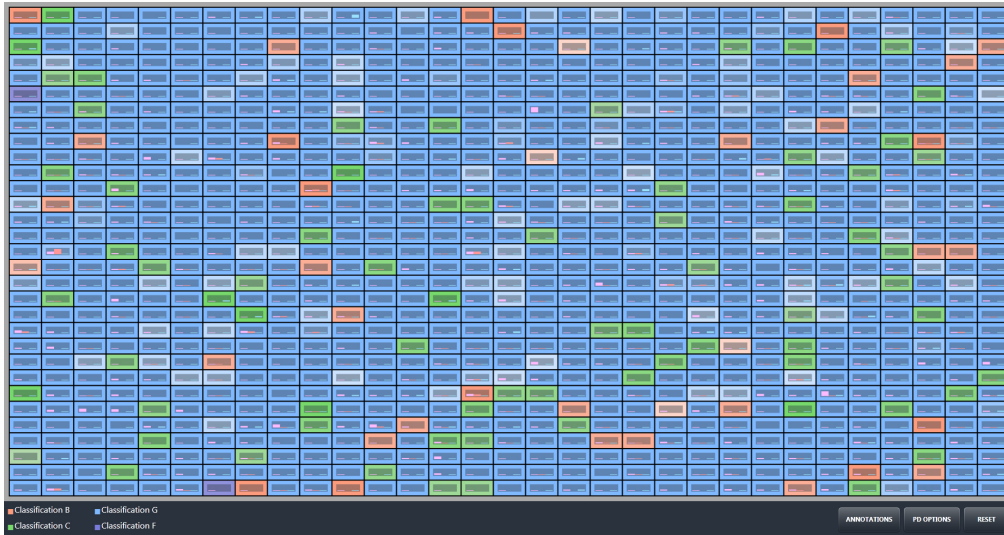


Figure 5.1: The user interface of the patent collection viewer contains as few controls and elements as possible in order to focus on the content itself. Each document in the collection is visualized with a symbolic thumbnail representation.

### 5.1.2 Detail Patent Viewer

The user interface of the detail patent viewer is visible in Figure 5.2. In this viewer a single patent can be studied in more detail. The interface contain three different areas: a text viewer on the left, an annotation bar at the center and an image browser on the right. These areas the patent examiner needs to scan and study. Each area has it own goals. The text viewer text is used can be selected, highlighted and read; the annotation bar is used to scroll through the document and get a quick overview about the relevance and the image browser is used to study the drawings of the patent. It is also possible to navigate through the collection by making use of two finger flick gestures.

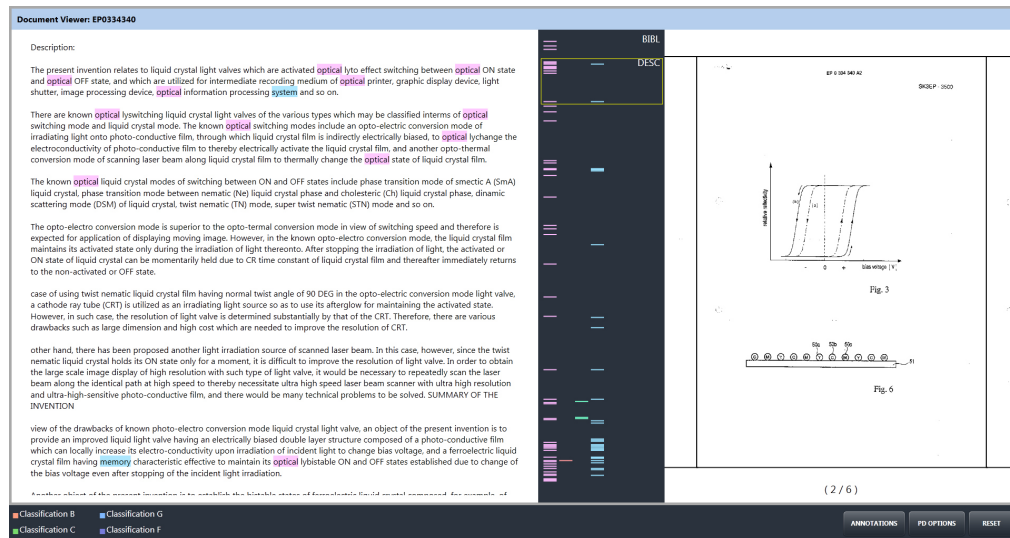


Figure 5.2: The user interface of the single patent viewer contains a text browser on the left, an annotation bar at the center and an image browser on the right.

### 5.1.3 Intermediate Thumbnails

Between the patent overview and the detail patent viewer there are intermediate thumbnail representations. Figure 5.3 gives an overview of the possible thumbnail representations for one patent for different zooming levels. The smallest thumbnail only contain an iconic representation of the patent. Once the thumbnails become larger the more space is available to fill with meta information. The three largest zooming levels also include images. Some intermediate thumbnails are not included in this Figure, since there were too little differences. These thumbnail representations are used to create smooth transitions between the different zoom levels. One improvement of the thumbnail representation is that we added multiple images instead of one image. The thumbnail representations are evolved during the project. We noticed that multiple images are more useful than displaying a single image. We also added a small black dot to the thumbnail representation to indicate whether a patent have already been showed to the user. This information provide insight about what patents have already been shown and which not.

## 5.2 Expert Reviews

To prove something about the effectiveness, efficiency or satisfaction of *TouchPat* evaluations with end-users are needed. There are different evaluation methods that can be used to measure these properties: model-based, observational and co-operative evaluation [Arnowitz 08]. In model-based evaluation the users have to work with *TouchPat* and on the background properties like the number of actions needed, time needed to read a patent document, time differences between the different thumbnail representations, are measured. The other two also require observations, interviews and questionnaires with the end-users. The final *Touch-*

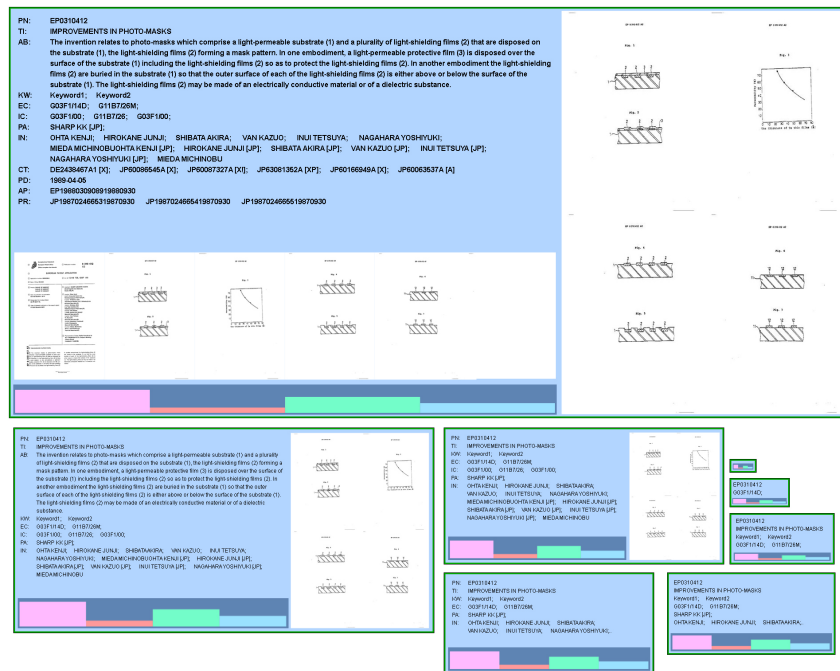


Figure 5.3: An overview of different semantic thumbnail representations of one patent for the different zoom levels. Each thumbnail gives the best summary possible in the available space.

*Pat* application is created from an iterative approach over months.

We demonstrated our work to patent examiners and software developers from the EPO. The presentation ended with a discussion about *TouchPat* in which we let the user think out loud about what they believe and think of *TouchPat*. This discussion provided feedback about how the end-users think, but they also offer us new ideas and solutions. The meeting also included time to let the examiners play with the latest version. Unfortunately, there was too little time left to get useful feedback from this. During another meeting with an ergonomic specialist and a user interface design expert, we again gave a short demonstration of our work and held a discussion about *TouchPat*. These experts have a lot of experience with ergonomics, user interface design and user-profiles of the end-users. They offered us new ideas, improvements and thoughts about the application. The results of both discussions are described in the next Section.

Getting a group of end-users and setting up questionnaires and interviews takes much time. Due to the time constraints were not able to do a more sophisticated end-user evaluation. We have used alternative methods like discussions in order to receive feedback in *TouchPat* from the end-users. At the moment we are arranging a group of end-users so that the EPO can do more extensive and structured evaluations with end-users. A good evaluation for something that is built in a week or a month can quickly take up to several months.

## 5.3 Discussions

The discussions with the end-users and expert offered us great feedback and new ideas for our application. Most ideas and feedback were improvements for the visualization and spatial ordering of the patents. Little feedback was related to the multi-touch interactions, which is also an important aspect of the application but hard to discuss and measure during a discussion. This Section gives an overview of the most interesting discussions during the meetings.

### 5.3.1 Starting Point

*The current application supports different ways to order the resulting patent list. The examiner starts analyzing the collection by starting from the first patent in the ordered list and working to the end of the list. TouchPat will directly provide an overview up to 1000 documents, at what location should the examiner start analyzing the patents?*

In *TouchPat* it is not required to open the patents one by one in the detail patent viewer, when studying a patents collection. In *TouchPat* the patent collection is analyzed by zooming in and out. *TouchPat* provides different strategies to analyze the patent collection in more detail. One way to analyze the patent collection is to directly zoom in on a couple of patents and move through the collection until all patents have been visited. Another approach is to first order the patent collection and then start analyzing the patents from the first one to the last one. The patent collection can be ordered on properties like: classification, inventors, applicant and a ranking. The ranking is based on the frequency of keywords in the patent. The first patents in the list are the patents that contain the most keywords and are most related. Finally, it is also possible to analyze the patent collection by creating different subcollections and analyze these one by one. We have added a small black dot to the thumbnail representation to indicate whether a patent has already been shown to the user. This information provides insight about which patents have already been shown and which not.

### 5.3.2 Thumbnail Representation

*TouchPat contains only one thumbnail representation for all fields. The thumbnail representation combines text information, images and an iconic representation of the document and the images are only visible when we are almost fully zoomed in on the collection. In some fields this thumbnail representation is useful, but in other fields they want to see the images earlier, since in some field the images are more important than the bibliographic information.*

The current thumbnail representation creates a summary of the patent document by making use of all possible data sources. We did not focus on one field, but tried to focus on creating in general the best summary possible in the available space. The application can be improved by adding different thumbnail representations for different fields. In some fields, text is more important while in others the images are more important. The order in which we add the elements to the thumbnail are based on our own experience with articles and feedback from

the end-users. A more advanced user study is required to create a list of the most important elements for each field. These results can be used to create a more specific thumbnail representation for each field. The application can also be extended by letting the user customize the thumbnail representation. The user can then decide which elements should be used and which not. The problem with this approach is that there are too much elements that can be used. The examiner may not know what elements to take. This can be simplified by giving the user a basic thumbnail representation (based on a user study) as starting point.

### 5.3.3 Clustering

*TouchPat gives an overview of 1000 patents, each patent is visualized by a rectangle object that contains information about the patent. Clustering algorithms could directly provide sub-collections of similar patents and need less space to visualize the whole collection. Why did we not make use of clusters?*

Clustering algorithms group similar patents, but also reduce the amount of space available to visualize information. For collections up to 1000 patents, grid displays can make very efficient use of the available space. Once the number of patents in the collection increases, clustering algorithm probably will be needed. The thumbnails for large patent collections become so small that it is not possible to add information. With clusters more patents can be visualized at the same time but the information provided will be different. For example, grouping patents with similar inventors will directly provide an overview of what inventors are involved in the search query. When a large patent collection is displayed on a grid it requires zooming in to see information. A good property of clustering is that it will improve the performance of our application for larger collections. With clusters multiple patents are combined into one object, they no longer have to be displayed separately.

### 5.3.4 Context Menu

*The hexagon based context menu has a maximum of five actions in each menu. What will happen to the menu when it has more than five actions? How can we visualize these? Can the context menu be improved by letting the user customize it?*

A hexagon is a polygon with six edges. Five of these are used to link an action and one is used to undo the action. We have organized the context menu in a way that each submenu has not more than five actions. There are some solutions to use more than five actions in the menu. One solution is to increase the number of edges. This will decrease the size of the edge, which makes it harder to cross a specific edge. Another solution is to group the actions into a single action. If we activate this action, a popup with a list view will appear. A list view can contain more than five options and allows scrolling through the list view in order to select the desired action. We also propose to do a user study to find out what the most frequently used actions in the menu are. Actions that are used most often can be placed on the 'easy' locations. The 'easy' locations are reached quicker and will be remembered easier. For example, two straight gesture movements to the left are easier to remember than making a z-shaped movement. We think that a customizable context menu will only be interesting for more advanced users. Novice users will probably not change the default context menu.



But a user study is needed to see whether there is interest for this option.

### 5.3.5 Spatial Relations

*The patent collection can be ordered in different ways. What is the relation between two horizontal and two vertical patents in the ordered list?*

In *TouchPat* patents can be ordered based on different properties. On the background the patent are stored in a list. Sorting this list only affects one direction. The ordered list is visualized on a grid display, by filling row by row. Horizontal adjacent patents are related and vertical adjacent patent might not have any connection. We suggest improving the ordering algorithm by visualizing similar patents close to each other. For example by starting at the top left and first filling the three places around this point or by making use of treemap or island visualizations. Similar patents are positioned next to each other which allows searching around the patent of interest instead of analyzing the patents row by row.

### 5.3.6 Interface vs. Multi-Touch

*If we look at TouchPat, we see two different things: a new interface and a multi-touch application. Can TouchPat also be controlled with mouse input? Does optimizing the user interface for touch input result in better user interfaces for mouse input?*

Designing a user interface for a multi-touch application requires a different strategy than we are used for regular desktop applications. This resulted in a new user interface for browsing and navigating through patent collections. We had to limit ourselves to a maximum of two fingers as touch input. Therefore, *TouchPat* only has simple touch inputs and gestures. Most of the actions can also be performed with a mouse device. We did not focus on adding mouse events to the application. However, it is possible to do some of the actions with mouse input. Extending the application to be fully controlled with mouse events allows us to create a good evaluation that compare mouse with touch input. We cannot directly say something about the efficiency compared to mouse based user interfaces. We can see that it has resulted in a user interface that focuses on the most important part of the application, the patent collection. We have removed most of the elements to have more space available to display the patent collection. To prove something about the efficiency of multi-touch interfaces compared to mouse based interface, evaluations are needed.

## 5.4 Conclusion

Based on the results above, some conclusions can be drawn. In general most users were surprised by the result of this project. We have created a new interface and received a lot of positive feedback on it. The end-users had many ideas to extend and improve *TouchPat*. In previous research for new user interfaces, the user directly remarked whether less time is needed for patent examination. In our discussions the users did not directly mention

this. With the emergence of smartphones and tablet computers users get familiar with using multi-touch interactions. They seem to be ready for something new, which perhaps multi-touch can bring.

We have focused on visualizing a patent collection containing up to 1000 patents. Most of the remarks were about the relation between the visualization and query and about visualizing larger patent collections. They wanted to see the difference between two or more queries or what the effects of adding an extra keyword to the query have on the resulting collection. One effect of displaying larger patent collections is that we need to reduce the amount of displayed data. This can be done by using clustering algorithms.

We can conclude that creating an application to work with multi-touch interactions has resulted in a drastically new user interface. This user interface is more focused on the most important aspect of the application, which is to visualize the patent collection and being able to browse and navigate through it. Still, more advanced evaluations with end-users are needed. These evaluations will provide feedback about the efficiency and usability of the application. Evaluations are also needed to optimize what information should be added to the thumbnails and to improve the structure of the context menu.

## Chapter 6

---

# Technical Details

During the implementation of our application, we had to solve several problems and challenges. This chapter provides an overview of the major challenges during the implementation phase and explains how they are solved.

Section 6.1 starts with a description of the data used in our application. Section 6.2 explains the different types of events. Section 6.3 describes how the use of diagrams has worked in our favor. Section 6.4 explains how we have applied annotations in the single patent viewer. Section 6.5 describes why we keep track of the path followed and how we use it. Section 6.6 explains how to create a useful ranking based on the annotations. We end this chapter with the performance issues.

### 6.1 Data Import

The most important ingredient for our application is the data. There are several data sources that we can use in our application. The most obvious one is to use a dataset generated at the EPO based on a real patent application. At the beginning of this project we have directly asked the EPO to provide us a collection of patents in a structured way, so that we could focus us on other aspects of the project. Since we do not have a direct connection the EPO databases from the TU Delft, we have to store these patents locally. A patent examiner in the field of electronics helped us creating three different search scenarios for three unique patent applications. Each scenario included a search query with his results and a set of annotations with their corresponding colors.

At the beginning of the project we did not have access to a dataset with patents yet. Therefore, during the first phase of our project we made use of our collection of scientific papers to test our first visualizations. Scientific papers have similarities with patent documents and are there for a good alternative as starting point. In an early stage, we discovered that the EPO had problems getting all the data in the structured way we want it and had to look for alternatives. Alternative datasets that can be used are: the PatExpert<sup>1</sup> dataset,

---

<sup>1</sup><http://www.patexpert.org/>

the MAREC<sup>2</sup> dataset or create a connection with the Open Patent Service (OPS<sup>3</sup>). The PatExpert dataset is specially created for the PatExpert project. This dataset contains bibliographic information and the full-text information of 7000 patents. This dataset was not structured on the way we wanted it, but it included all information needed in our application. One of the problems with the dataset is that it did not provide the full-text information in a structured way. The text was stored in one field without any structure and a lot of noise. Another problem was that the drawings of the patents were not included in this dataset. The MAREC dataset consists of 100.000 randomly picked patents from each sub-collection of the MAREC dataset (EPO, JPO, USPTO, WIPO). It was targeted at people submitting papers to the AsPIRe'10 workshop at the ECIR. This dataset is well structured and contains the images for each patent as well. The problem with this dataset was that we did not have access to it. In the last weeks of the project we were able to get this data, which was too late to make us of it. A connection with OPS gives direct access to the databases at the EPO. The required input to create this connection is a set of patent numbers and the output are XML files for each patent. The problem with OPS is that we are not sure whether we can extract all the needed information.

Halfway through the project, the EPO provided us the full-text information of all patents from one scenario. Due to time limits and technical problems they were not able to include the bibliographic information and images. Since this dataset did not include bibliographic information which is very important for our application, we decided to stick to the PatExpert dataset. This dataset contains all the information needed. The noise can be removed with filters and the images can be extracted with external tools.

### 6.1.1 Data structure

The data structure of our application is summarized in Figure 6.1. The input data is first converted into our own patent structure. This conversion directly allows us to clean the data by filtering. The Microsoft Surface Touch Beta API is an intermediate level between the viewer and the data and handles the touch inputs. This data structure takes the extensibility of our application into account. There is no need to visualize and update the patents on the background (which are not visible) when zooming in. TouchPat decides what patents are visible and ask the data provider to get only the data of these patents. This directly improves the performance of the application and reduces the amount of memory used. Even when the application is extended later on, the memory usage will remain good, since it only load and visualizes the visible patents.

### 6.1.2 XML data

Before we can make use of the data inside the XML files, we need to create a XML reader that extracts the data. We created a XML reader that reads the XML files and directly convert it into our own patent structure. Using your own structure allows changing the data source easily and makes the application quicker since it only stores the necessary information. The only requirement for a new data source is a new data converter that converts the

---

<sup>2</sup><http://www.ir-facility.org/prototypes/marec>

<sup>3</sup><http://www.epo.org/searching/free/ops.html>

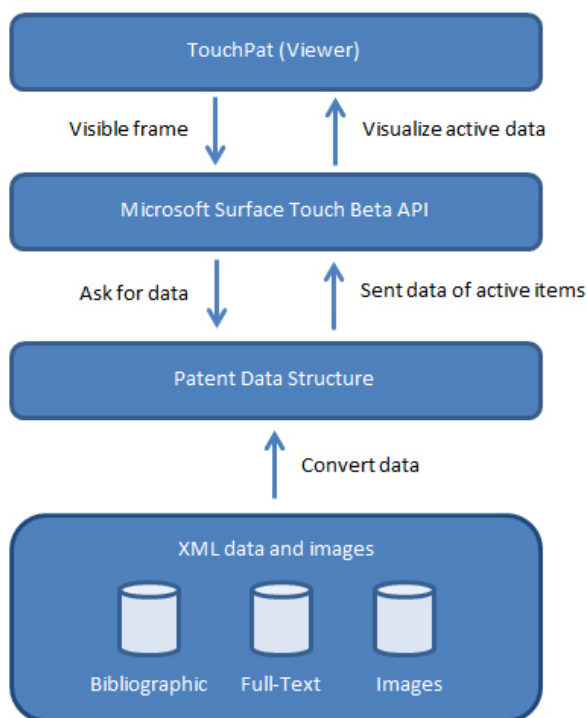


Figure 6.1: An overview of the data structure of our application. XML data is first converted into our own patent structure and the application API will only load and visualize the data of the visible patents.

data into our own structure.

Before the data is converted into our own patent structure, we need to remove the noise. By removing the noise before it is stored in our own structure, we can directly store the information in the way we want to. The noise in the XML files consist mainly of HTML-tags (i.e.: `< RTI >< /RTI >`) and document type indications (i.e.: A1, A2, B1, etc.). Most noise can be removed with standard string replace and regex functions, but some noise requires more advanced steps. As an example, we will take a look at the data from the classification field of the bibliographic information.

```

1 (A2)G11B7/22; G11B7/24 (A3) G11B7/24; G11B7/26; G11B7/28;

```

Listing 6.1: This is an example of the data from the classification field.

This line of code means, that the A2 version of this patent is classified in the G11B7/22 and G11B7/24 classifications and the A3 version of this patent is classified in the G11B7/24, G11B7/26 and G11B7/28 classifications. We can assume that the latest version of the patent contains the classifications that fit the best. To extract the latest classification from this line of code, we first need to split this text into two parts: A2 part and A3 part. Then, we choose the latest part. We can now directly save the data in the format we want.

```
1 string regex = "(\\(A2\\)|\\(A3\\))";  
2 string newtext = Regex.Replace(text, regex, '');  
3 string [] results = newtext.Split(';');  
4 return results[results.Length - 1];
```

Listing 6.2: This is a part of the code which removes some of the noise.

### 6.1.3 Image Abstraction

The generated thumbnail representations include bibliography information, drawings and an icon representation. The bibliography information can directly be extracted from the XML files, the iconic representation can be generated from the full-text information, but the image of the first page need to be extracted from the PDF file. To create an image of the first page of the PDF file, we used the external library called Ghostscript<sup>4</sup>. GhostScript requires a PDF file and a page number as input and the will return an image of the page as output. To extract the drawings from the PDF file some additional steps are needed. The PDF files are poorly structured and contain no information about the text and drawings. They only contain a scanned image of each page in the patent. In order to extract the drawings we can make use of these bookmarks. A drawing page can contain multiple images on one page, ideally we want to extract the drawings individually. There are tools that are able to extract individual drawings from one page. However, the results of these tools are not very good since the images overlap each other. We didn't want to spend too much time on finding or building a tool that is able to extract the drawings individually and decided to use the images of the pages that contain drawings.

We created a separate application that automatizes the process of extracting drawings for a document collection. The application read the patents one by one and generates for each patent the correct images. The application makes use of two external libraries: Ghostscript and PDFtk<sup>5</sup>. PDFtk is used to get the bookmarks from the PDF file, these are necessary to get the page numbers of the page inside the drawing section. Next, GhostScript use these page numbers to create an image of the first page and the pages of the drawing section. Finally, standard C# functions are used to combine multiple images with each other to create an overview of all drawings and a tile with the first four drawings.

## 6.2 Touch Input

TouchPat is built on the Microsoft Surface Touch Beta API and WPF<sup>6</sup>. These frameworks have many different events that can be used for multi-touch interactions. Before we can use the different types of events we first wanted to know more about them.

<sup>4</sup><http://pages.cs.wisc.edu/~ghost/>

<sup>5</sup><http://www.pdfplabs.com/tools/pdftk-the-pdf-toolkit/>

<sup>6</sup><http://msdn.microsoft.com/en-us/library/ms752059.aspx>

### 6.2.1 Events

Most controls in WPF are content controls. Content controls can hold any type and amount of nested content and are derived from the `UIElement` class, which contain Mouse and Touch events. Events in WPF are grouped in three different kinds of events:

- **Direct Events.** These events are like the ordinary .NET events. The event rise in the element and are not passed to any other element. For example, `TouchEnter` (which fires when the touch pointer moves over an element) is a direct event.
- **Bubbling Events.** These events travel up the containment hierarchy. For example, `TouchDown` is a bubbling event. It's raised first by the element that is touched. Next, it's raised by that element's parent, then by that element's parent, and so on, until WPF reaches the top of the element tree.
- **Tunneling Events.** These events travel down the containment hierarchy. They give you the chance to preview (and possibly stop) an event before it reaches the appropriate control. For example, `PreviewKeyDown` allows you to intercept a key press, first at the window level and then in increasingly more specific containers until you reach the element that had focus when the key was pressed.

Name	Routing Type	Description
<code>PreviewTouchDown</code>	Tunneling	Routed event that occurs when a touch presses this element.
<code>PreviewTouchMove</code>	Tunneling	Routed event that occurs when a touch moves while inside this element.
<code>PreviewTouchUp</code>	Tunneling	Routed event that occurs when a touch is released inside this element.
<code>TouchDown</code>	Bubbling	Routed event that occurs when a touch presses this element.
<code>TouchEnter</code>	None	Routed event that occurs when a touch moves from outside to inside the bounds of this element.
<code>TouchLeave</code>	None	Routed event that occurs when a touch moves from inside to outside the bounds of this <code>UIElement</code> .
<code>TouchMove</code>	Bubbling	Routed event that occurs when a touch moves while inside this element.
<code>TouchUp</code>	Bubbling	Routed event that occurs when a touch is released inside this element.

Table 6.1: This table provides an overview of all multi-touch events supported by the Microsoft Surface Beta Touch API.

Table 6.1 gives an overview of all multi-touch events supported by the API. These events are raw events, which does not support two or more finger touch gestures like zooming and rotating. Microsoft Surface Beta API includes high-level support for these gestures movements, called touch manipulations. Basic manipulations like zooming and rotating can be handled with ManipulationCompleted, -Delta, -Started and -Starting event. To allow more realistic, fluid manipulation of elements, WPF has another layer of features that build on its basic manipulation support, called inertia. With inertia enabled elements can get a presence and sense of momentum. For example, if you drag an image and release your finger from the screen, the movement will continue for a short period and decelerate gracefully. Inertia also let elements act like real physical object by letting them bounce back when they are dragged into a boundary which they can't cross. Table 6.2 gives an overview of the manipulation and inertia events.

Name	Description
ManipulationBoundaryFeedback	Occurs when the manipulation encounters a boundary.
ManipulationCompleted	Occurs when a manipulation and inertia on the UIElement object is complete.
ManipulationDelta	Occurs when the input device changes position during a manipulation.
ManipulationIntertiaStarting	Occurs when the input device loses contact with the UIElement object during a manipulation and inertia begins.
ManipulationStarted	Occurs when an input device begins a manipulation on the UIElement object.
ManipulationStarting	Occurs when the manipulation processor is first created.

Table 6.2: This table provides an overview of all manipulation and inertia events supported by the Microsoft Surface Beta Touch API.

## 6.3 Use of Diagrams

The application includes many overlapping actions and gestures that should work smoothly next to each other. If it is not clear what actions and gestures must be activated at what moment, the code can become very complex.

### 6.3.1 States

The main application has four different states: running state, transition state, context menu state and the detail viewer state. Each state allows a limited number of actions. In Figure 6.2 the state diagram of the main application is visible. When the application is started,



it will immediately change the current state to the "application running" state. From this state there are four possible actions: a single click to (de)select a patent; a double click to zoom in on a patent; a long click to open the context menu and a gesture to transform the patent collection. Once a zooming and/or translation gesture is made, the current state will change into the transition state. The application remains in this state until your finger is released and the gesture is stopped. In the transition state it is not possible to start another action besides finishing the gesture. If the context menu is opened, the current state will change into the "context menu" state. In this state all events and actions are blocked besides finishing your gesture movement, by releasing your finger. Once your finger is released the current state is directly changed back to the "application running" state and if the gesture movement crossed one of these lines of the menu it will start the corresponding action.

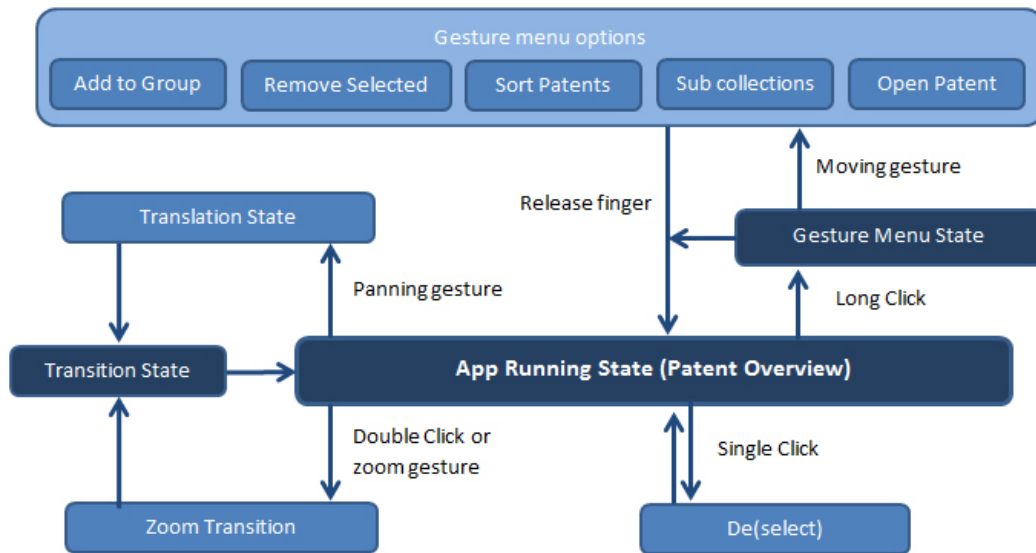


Figure 6.2: An overview of the different states and possible actions of our main application.

One of the possible actions in the context menu is to open a patent in the detailed viewer. Once this option is chosen, the current state will change to the "patent viewer" state. In Figure 6.3 the state diagram of the patent viewer is shown. The single patent viewer contains three different areas: text field, annotation bar and image field. Each window has its own events and actions. In the text field, vertical movements are made to scroll through the document and horizontal movements are made to select the text. In annotation bar only click and move events are allowed to scroll through the document. In the image field horizontal movements flicks are made to browse through the drawings and click events are made to manipulate an image. The last possible action in the detail patent viewer is patent navigation. Horizontal flick movements with two fingers are used to move to the previous or next patent of the collection.

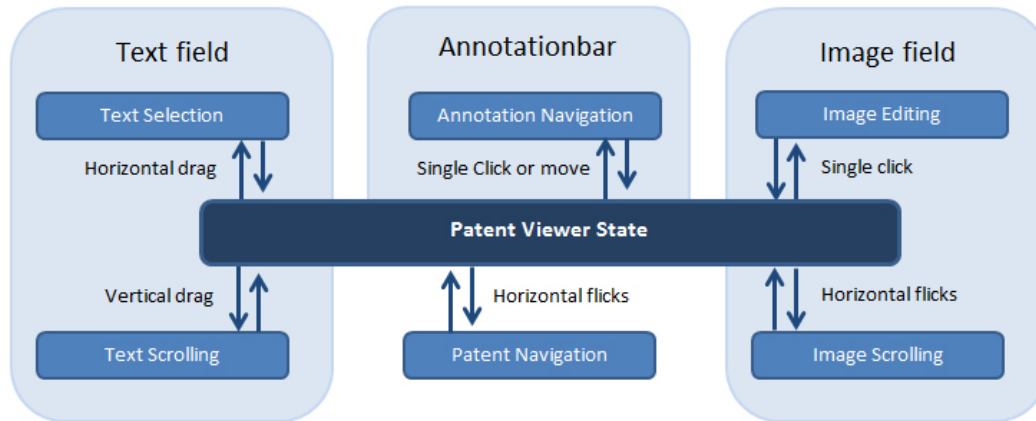


Figure 6.3: An overview of the different states and possible actions in the single patent viewer.

### 6.3.2 Events

You can also use state diagrams at the event level of the application. The application contains many overlapping touch events that should work next to each other. In order to make use of these events, we first need to know the order in which these events are launched. The order in which the events are launched is visualized in Figure 6.4. On the right side of the Figure it shows manipulation events. These events will only rise when the manipulationEnabled property is set to true.

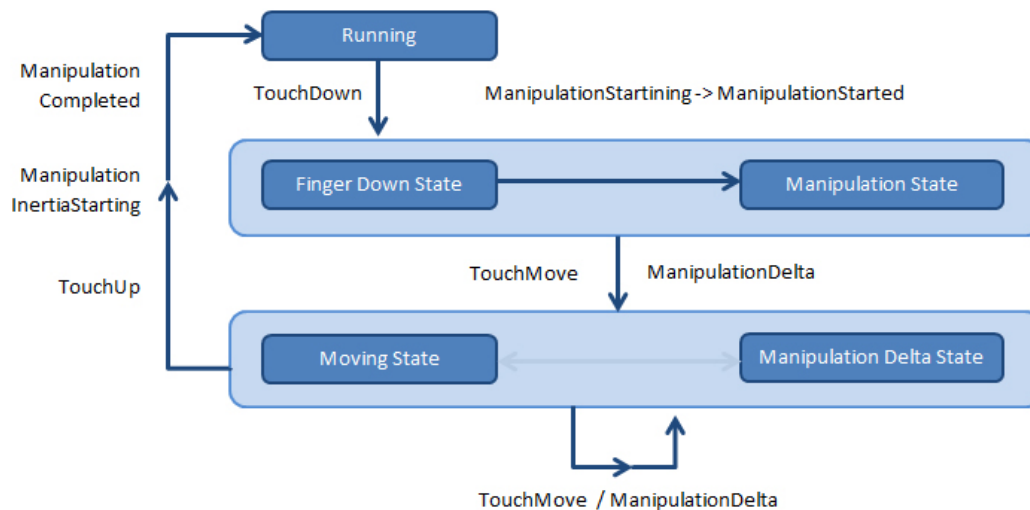


Figure 6.4: Overview of the states and events in the application.

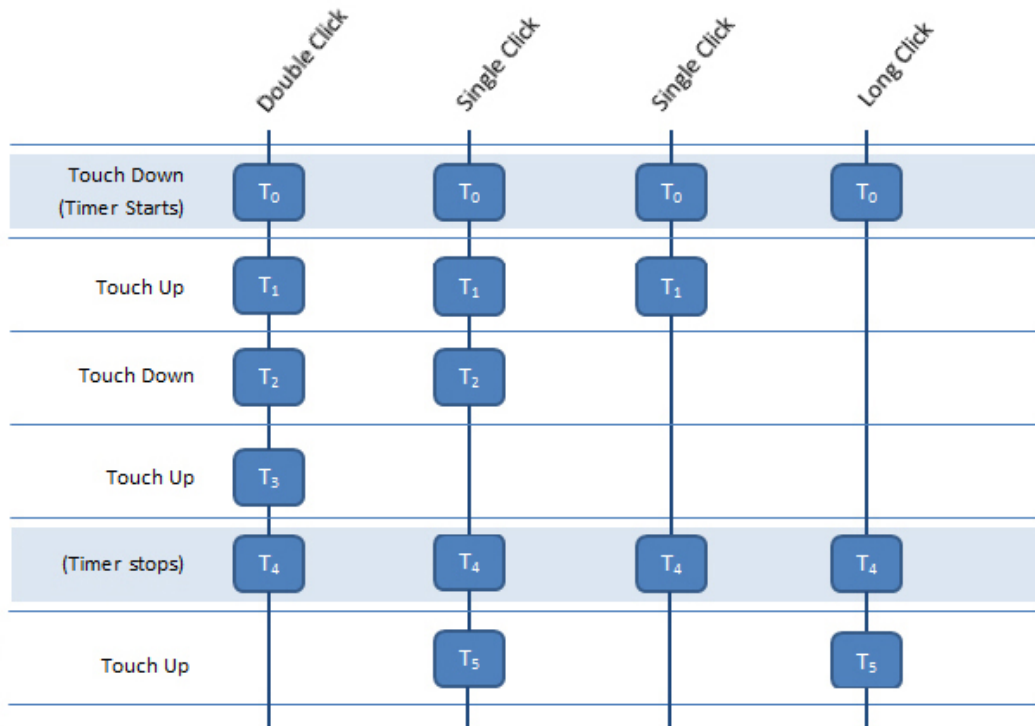


Figure 6.5: Event timeline. The order of the low-level events (shown in columns) indicates the type of click (shown in rows).

This schema can be used as reference to create custom events and gestures. For example, multi-touch does not support double click events by default. We can add a double click event in our multi-touch application by making use of the default `OnTouchDown` and `OnTouchUp` events in combination with a timer. Normal timers depend on the frame rate of the application. A low frame rate increases the duration of the interval. To create a double click event, we want to make use of a background timer. The interval of the background timer does not depend on the frame rate of the application and run on its own. The timer starts when you finger touches the screen for the first time and end when the interval expires. The status of your finger and the number of taps that have been made decides whether a single click, double click or a long click is made. Figure 6.5 describes the reasoning of deciding which type of click is made.

## 6.4 Annotations

The single patent viewer contains two areas that make use of annotations. Both require to have a predefined annotation set. This annotation set contains one or more annotation groups and each annotation group contains a color and one or more keywords. Text highlighting requires going through the text and checking for each keyword whether it is similar to one of the words in the annotation groups. If a match is found the background color

property of the keyword is changed in the corresponding color.

Creating the annotation bar requires a different approach. We first need to know the number of rows in the full-text. This number is needed to define the thickness of the annotation stripes. Next, we need to check row by row whether the row contains one of the keywords from the annotation groups. If there is found a match, we will use the row number to draw the annotation stripe at the right location in the annotation bar. Instead of using the default scrollbar of the text viewer, we have removed the scrollbar and added the scroll functionality to the annotation bar. The scrollbar functionality can be added by handling the touchDown, Move and Up events of the annotation bar. In order to scroll to the right place in the text viewer we need to convert the location in the annotation bar to the right location in the text viewer, by making use of the length of the controls. Standard scrollbars have a bar that indicates the active part of the document, in our annotation bar we used a yellow frame to indicate the active part of the document.

Description:

The present invention relates to liquid crystal light valves which are activated **optical** lyto effect and **optical** OFF state, and which are utilized for intermediate recording medium of **optical** printer, shutter, image processing device, **optical** information processing **system** and so on.

There are known **optical** lyswitching liquid crystal light valves of the various types which may be switching mode and liquid crystal mode. The known **optical** switching modes include an opto-electric irradiating light onto photo-conductive film, through which liquid crystal film is indirectly electrically electroconductivity of photo-conductive film to thereby electrically activate the liquid crystal film, and conversion mode of scanning laser beam along liquid crystal film to thermally change the **optical** state

The known **optical** liquid crystal modes of switching between ON and OFF states include phase trans liquid crystal, phase transition mode between nematic (Ne) liquid crystal phase and cholesteric (Ch) scattering mode (DSM) of liquid crystal, twist nematic (TN) mode, super twist nematic (STN) mode and

The opto-electro conversion mode is superior to the opto-thermal conversion mode in view of switching expected for application of displaying moving image. However, in the known opto-electro conversion maintains its activated state only during the irradiation of light thereonto. After stopping the irradiation

view of the drawbacks of known photo-electro conversion mode liquid crystal light valve, an object of provide an improved liquid light valve having an electrically biased double layer structure composed of which can locally increase its electro-conductivity upon irradiation of incident light to change bias crystal film having **memory** characteristic effective to maintain its **optical** lybistable ON and OFF states the bias voltage even after stopping of the incident light irradiation.

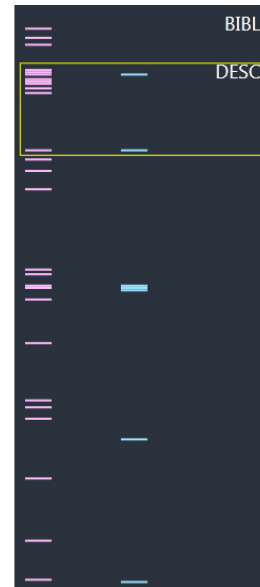


Figure 6.6: We have used annotations both in the text viewer and in the annotation bar. The yellow frame visualizes the active part of the document.

## 6.5 Path Tracking

The path followed by the user gives us much information about used search strategies. It provides information about patents that have been visited, how the current subcollection of patents is reached and it is possible to modify previously made decisions. The current implementation of path tracking is limited by only keeping track of actions made, the time patents have been showed and what patents are visible to the user. Each action is logged in a log file that describes the specific action and the time the action is executed. This

information is used to indicate what patents have already been shown. We can't say that a patent has been read in detail, but we can say that the patent has been shown to the user. The more you are zoomed in on a patent the higher the effect on the time showed is, since there is a higher chance that you have seen it. When the user is completely zoomed in, there is a one on one effect on the time and when four patents are showed the effect on time is one divided by the number of visible patents. Path tracking can also be used for the evaluation of the application, for example by reconstructing different search strategies and compare them with each other.

## 6.6 Ranking

The patents can be ordered based on different properties. The standard way of patent sorting uses properties such as: year of publication, title and classification. Another way to sort patents is to create a relevance value based on the annotations. Inverse Document Frequency (IDF)<sup>7</sup> creates such a ranking value, that can be used to sort the patents. The IDF is calculated based on the length of the patent and the frequency of the annotation keywords. The IDF can be calculated with the following three equations:

$$tf_{i,j} = n_{i,j} / \sum n_{k,j} \quad (6.1)$$

with,  $n_{i,j}$  as the number of occurrences of a considered keyword in document  $d_j$  and the denominator as the sum of number of occurrences of all terms in document  $d_j$ .

$$idf_i = \log(D / \text{numberOfDocumentsContainingKeyword}) \quad (6.2)$$

with, D as the number of documents in the collection and d as the number of documents that contain the considered keyword.

$$(tf - idf)_{i,j} = tf_{i,j} * idf_i \quad (6.3)$$

## 6.7 Performance

Multi-touch interactions work the best if they run smoothly. The application should be able to visualize up to 1000 documents without any delay. Each document is represented by a rectangular object on the canvas. Applying zooming and translation gestures on the collection will constantly update the visualization based on a delta value. Updating 1000 rectangle objects takes so much time that the application will temporary freeze.

<sup>7</sup><http://en.wikipedia.org/wiki/Tf%E2%80%93idf>

### 6.7.1 Data Structure

To increase the performance of the application, we experimented with the way patent are stored. The patent information is stored in our own patent class. This class contains many different properties like background color, meta information and location. The WPF performance, manual provided by the MSDN Library<sup>8</sup>, suggests several ways to improve the performance of WPF applications. One suggestion is to use data binding instead of manually add properties. We have created a benchmark to verify the effect of changing this property on 1000 documents. The results, see Figure 6.7, shows that the update time of the document is indeed improved. Still, it takes on average 124 milliseconds to update all documents. For an application that has multiple updates per second, this number is still too high.

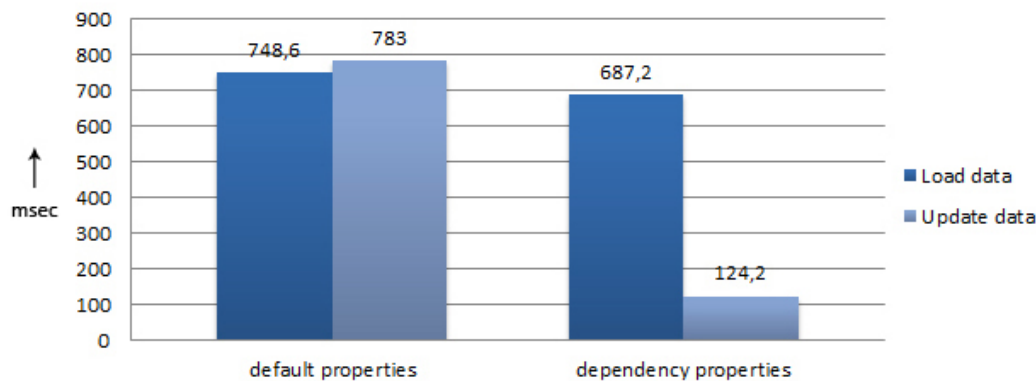


Figure 6.7: Load and update times for 1000 rectangles (documents) in milliseconds.

### 6.7.2 Graphics and Imaging

The WPF performance manual also suggests using `DrawingVisual` instead of normal rectangular objects. The rectangular objects provide layouting and event handling, which decrease its performance. The `DrawingVisual` object is a lightweight drawing class that can draw shapes, images and text. This class is considered lightweight because it does not provide layouting or event handling, which improves its performance. To test the difference between `Rectangles` and `DrawingVisuals`, we created a benchmark. For both classes we have drawn 1000 rectangles and updated the collection 500 times. We measured the time it takes to update the 1000 documents and calculate the average over 500 tries. For rectangular objects it took on average 167 milliseconds to update the 1000 documents and for the `DrawingVisuals` it took on average 26 milliseconds to update the 1000 documents. The results of the update benchmarks can be found in Figure 6.8. Using `DrawingVisuals` is almost 6 times faster than using rectangle objects. This gave the application a real performance boost.

<sup>8</sup><http://msdn.microsoft.com/en-us/library/aa970683.aspx>

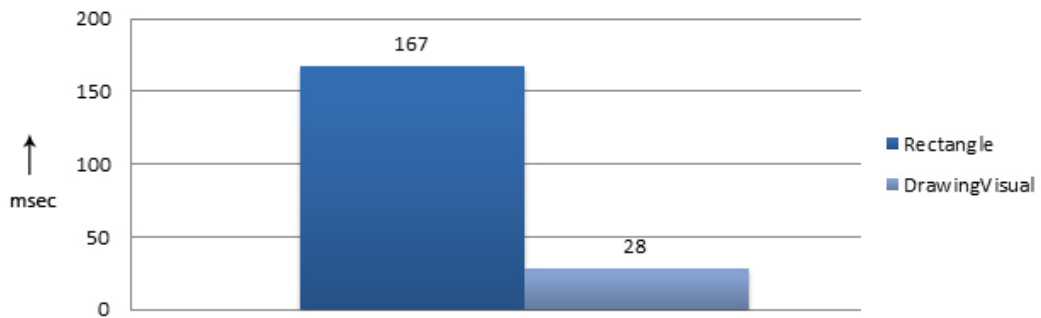


Figure 6.8: Average milliseconds needed to update 1000 rectangles (documents).

### 6.7.3 Scalability

The previously described performance boost makes it possible to display 1000 documents smoothly. But what will happen, when the number of patents increases later on. Increasing the number of documents has a linear effect on the time needed to update all documents. Figure 6.9 shows the influence of the number of documents on the update time of all documents. This is not very strange, since we need to draw more rectangles. We have added different techniques to improve the scalability of our application.

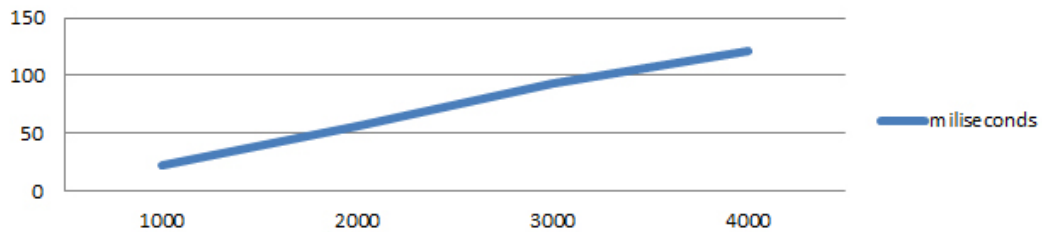


Figure 6.9: The number of documents has a linear effect on the update time in milliseconds.

When zooming in on the patent collection it is not necessary to update all patents, only the visible patents. In Figure 6.10, a box is visible that indicates what patents from the collection are visible on the screen. Only the patents within this border have to be loaded and updated. From the whole update process, image loading takes the most. We have ensured that the images are loaded before the text, to let the application directly react on touch input.

To keep the application running smoothly we also preload the data of patents that lay just outside this box. Less information need to be loaded when the visible frame is moved, this keeps the performance of the application more stable. Finally, we changed the update frequency for different zooming levels. For the lowest range of the zoom levels less updates are needed than for the highest range of zoom levels, as the thumbnails of the lowest range of zoom levels does not change much.

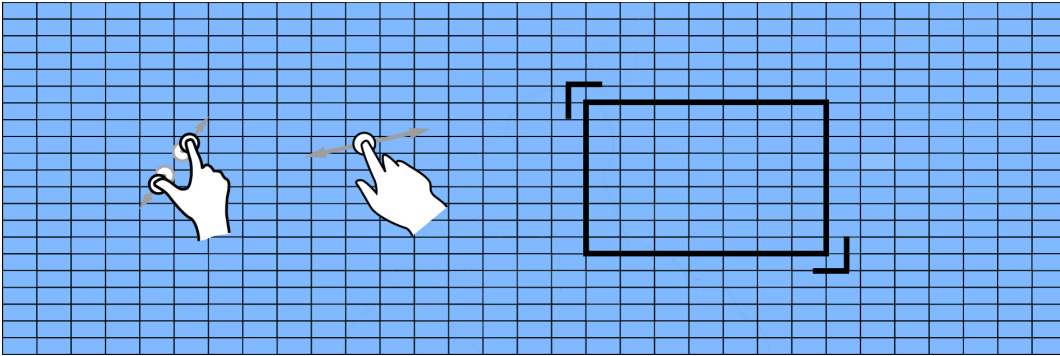


Figure 6.10: There is no need to update the patents that lay outside the border. This improves the performance and scalability of the application.

Unfortunately, we were not able to do a comparison with other software. There are not many patent examination applications available to create a good comparison. The only possibility was to do a comparison between different methods. This is something we have done during the experimental phase of our project.



## Chapter 7

---

# Guidelines for Multi-Touch Interfaces

One of the sub goals of this master's thesis project was to formulate guidelines that can be used to build a multi-touch application. We limit the focus of our guidelines on using maximum two finger interactions. Our testing device does not allow more than two finger inputs and two point interactions are a good base case for more general problems. Our guidelines aims to help developers and designers create applications and user interactions that can be controlled with multi-touch interactions. We have based our guidelines on our own experience and that from literature study [Ridder 11] [Lao 09] [Moscovich 07] [Kewaley 08] [Benko 09] and [Microsoft 09].

In this chapter, we will first describe the core principles for optimal user interactions in multi-touch applications in Section 7.1. In Section 7.2 we describe the core principles for designing the user interface of a multi-touch application.

### 7.1 Interaction Guidelines

Interaction defines the interplay of the software with user's behavior, response and gestures. Interaction provides a focus on how well and naturally the user interacts with the application. This Section describes several aspects of developing and/or designing natural user interactions within a multi-touch application.

#### 7.1.1 Events

Before you can start designing and developing a multi-touch application, you need to know what kind of events you can use in your application. The number of possible actions depends on the framework used and the number of focus points your input device allows. This guideline focuses on creating an application that can be controlled with one or two-finger interactions. One or two finger events allows creating many different combinations of actions.

We provide an overview of all possible one and two finger interactions in Table 7.1 and 7.2.







Image	Movement	Description
	Tap	Press and release with one finger
	Double tap	Quickly press and release two times with one finger
	Drag	Press with one finger and slide or push
	Flick	Press with one finger, slide quickly, and then release
	Press and hold	Press and hold one finger
	Tap	Press and release with two fingers

Table 7.1: All possible events that can be created with one or two finger input. Illustrations provided by Gestureworks ([www.gestureworks.com](http://www.gestureworks.com)).








	Double tap	Quickly press and release two times with two fingers
	Hold and tap	Hold one finger and press and release with another finger
	Drag	Press with two fingers and slide or push
	Flick	Press with two fingers, slide quickly, and then release
	Press and hold	Press and hold two fingers
	Scale	Bring two fingers together or pull them apart on one hand
	Rotate	Press with two fingers and twist them

Table 7.2: All possible events that can be created with one or two finger input. Illustrations provided by Gestureworks ([www.gestureworks.com](http://www.gestureworks.com)).

Each event in this list can be mapped on one of the actions of your application. Most users are getting familiar with the use of multi-touch interactions. This mapping has to be chosen wisely, since the user expects that certain types of events results in specific actions. For example, it is not logical to use scaling gestures to translate the data. More natural is to use the scaling gesture to scale the data, since the scaling gestures would stretch or squeeze the object in the real world. In Figure 7.1 we demonstrate this by mapping four different gestures on three different actions.

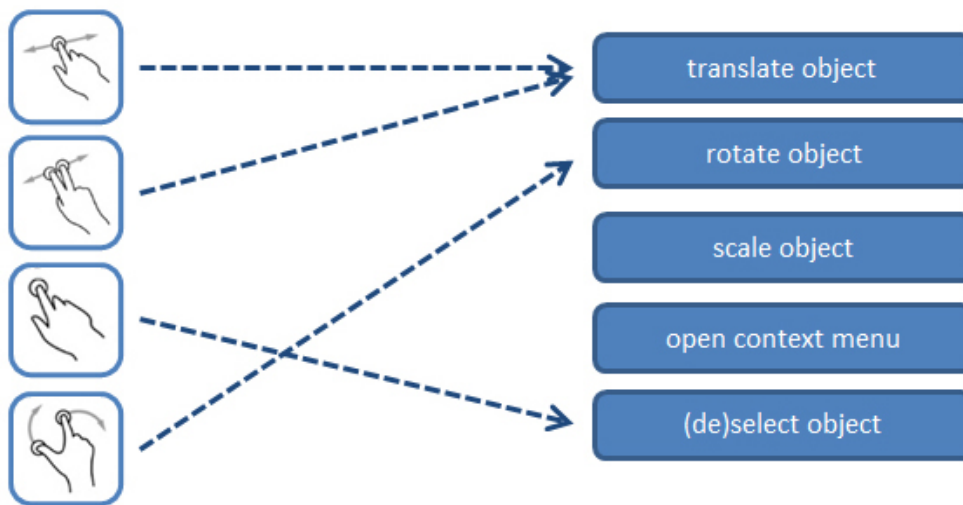


Figure 7.1: Four different events are mapped on three different actions.

### 7.1.2 Real World Behavior

The best way to create a complete and user friendly user interface is by making objects on the screen behave like objects in the real world. An object needs to have certain properties in order to act like a real world object.

#### Behavior

To let objects behave like physical objects you need to add motions, inertia of movement and natural-feeling collisions to the objects. We can add some points that will go beyond the real world behaviors. For example, in a multi-touch application you can scale photos while this is not possible in the real world. Every object and visible property change must smoothly animate and transition to the next state. You should prevent objects from abruptly appear or disappear. Do not give each object all possible behaviors. Think carefully about whether the behaviors are really needed. For example, in an image collection application you want to be able to manipulate the image in all possible ways. But in a zoomable user interface that is controlled from one side, such as a document viewing application, we do not need to rotate the space and only allow zooming and translations.

## Response

Create immediate responses to all user input that will receive a response. Pre-buffer content, provide a transition, or use other mechanisms to make sure that every touch receives an immediate and meaningful response. Place visual responses immediately at the position of the contacts, so that the touch appears like a direct and physical reaction. An application without immediate responses distracts significantly from the user task and will not feel natural.

## Discrete vs. Continuous

Graphical user interface applications mostly have discrete actions like single-click actions that are performed in a sequence in order to complete a task. Touch gestures and direct manipulations allow us to do continuous actions instead of discrete actions. For example, to change the size of an object, we first need to select the object, select the right command and then scale the object. With multi-touch input we can directly apply a scaling gesture on the object, the zooming gesture will continuously give visual feedback of the effect of the zooming gesture on the object.

### 7.1.3 Single vs. Multi-user Experiences

Multi-touch devices allow us to do collaborative tasks by gathering around the multi-touch input device. You should consider whether your application will be used by just a single user or by multiple users. There are some differences between single and multiple user applications. The interface for single user applications only need to be viewed from one direction, while multiple user require approaching the interface from different sides. The interface of a multiple user experiences application should be able to rotate or have a 360 degree interface.

Input devices like the Microsoft Surface are very suitable for creating multiple user experiences. Do not limit yourself by only focusing on multi-user interactions. A multiple user application should be controllable by a single user, without the need of other users. A typical property of an application for multiple users is that the application should be able to let new users join the application but also leave the application. Next, the users should be able to divide their tasks and decide whether they want to share their display or have two separate windows.

### 7.1.4 Constraints

Most multi-touch applications allow the users to freely explore the environment. The user can interact with the application when comfortable, instead of telling them what to do. Instead of limiting the possible actions, you should add constraints that prevent user from doing "wrong" actions. There are no guidelines about which constraints are needed for which application, the constraint should be chosen based on the type of application and the possible actions. For example, in a grid based zoomable user interface we do not want to visualize items that are partly visible. The user is able to freely browse through the application but

when he releases his fingers, the data will automatically snap to the grid.

## 7.2 Design Guidelines

Designing a multi-touch user interface requires a different approach to think about the user interface and overall graphical screen design than the development of traditional graphical user interface applications. The interaction guidelines provide some rules to create natural behaviors, gestures and responses. This Section will focus on designing the user interface. It is the visual design that brings the elements to life on-screen.

### 7.2.1 Layout and Orientation

There are basically two types of layouts that can be used in a user interface design: gridless and gridded layouts. It depends on the type of application which you can choose. Gridded layouts are ideal for productivity or focused activities, linear sorting of data, or simply to create a visual rhythm to certain screen states, while gridless layouts are ideal for free manipulations environments like a photo album. Gridded layouts can be used global (full-screen) or local (within an object).

Global grids provide a structured way of displaying the information, which allow the user to easily scan and organize the data. Global grids also force to orientate the interface into one direction which makes collaboration more difficult. Local grids give more structure to controls, content and objects. Controls and content are the most usable from one orientation, such as a list view control, or any content that is primarily text-based. Therefore, it is important to ensure that they remain floating and draggable in order to be rotated easily to the right orientation for a particular user.

### 7.2.2 Properties

Each element in the user interface has a couple of properties like the location, size, color, etc. Multi-touch input have effect on the way the elements should be visualized or where they should be positioned. We will discuss the effect of multi-touch input on these properties.

#### Positioning

It is important to choose the location of your elements and controls with care. Placing the controls on the top of the screen (or opposite side of the screen) will block the view of most of the screen when you need to click it. The best location to place the elements depend on whether the user is left or right handed. For right handed user the best location to place clickable elements is on the right or bottom side of the screen, while for left handed user the best location are on the left or bottom side of the screen. So the bottom side is best location for both.

## Size

The traditional graphical user interface contains small buttons which need to be precisely clicked by the mouse. Touch input is less precise than mouse input, the center of your fingertips decide where your focus point is. The size of the fingertips differs for different users and depend on the angle you will place your fingers on the screen. The elements need a larger clickable surface than we are used to, to be able to work well for a large variety of users.

## Depth

Touch input will only have effect on the selected item or collection. Using depth cues to separate the foreground elements from the background and helps to make clear what the content or control have the focus and what is in the background. To create depth in your application you can use a combination of scaling, drop shadows and depth cueing to the content or control.

## Colors

There are no strict rules about choosing colors for your application. However, there are some guidelines which can be used to optimize the effect of colors. The way colors are displayed depends on the type of input device. Colors may be displayed differently on monitors than on tablet computers or the Microsoft Surface. Important is to choose colors with high contrast, such as pure black on pure white. This reduces the effect of anti-aliasing and making text look crisp. There are some websites and applications which suggest color maps that contain colors with high contrast. For example, *colorbrewer*<sup>1</sup> suggests a couple of color maps that have high contrast and are colorblind safe. You should also experiment and test the colors on your output device to make sure they really have a high contrast on your output device.

### 7.2.3 Content as Interface

The user interface should contain a limited number of controls that do not belong to the content. The main part of the user interface should be the content itself. The users should be able to directly start working, for example by moving objects on the screen. The controls should be relevant to the data and should be recognized easily. Focusing on minimalistic interfaces prevent adding a lot of nonsense controls. For example, to zoom in on the data, the user should use a zooming gesture instead of a button. By minimalistic we do not mean small elements, but rather elegant, simple and little elements.

To create a minimalistic user interface we can make use of icons. Icons need little space and provide hints or clues to users so that they can infer what result a touch or gesture will create. Icons need to encourage exploration and help the interface stay learnable, discoverable, and natural. Text is also very important in a minimalistic design. By using casual, comfortable, clear and personal text messages we create an interface that stays close to the real-world and feels naturally.

---

<sup>1</sup><http://www.colorbrewer2.org/>





## Chapter 8

---

# Conclusions and Future Work

This chapter outlines the conclusions regarding this master's thesis project. First we describe the main conclusions regarding our research goal and whether we achieved this goal in Section 8.1. Section 8.2 describes the contributions we made during this project. This chapter concludes with an overview of identified future work.

### 8.1 Conclusions

This section reviews our research goal, and determines whether this goal has been reached or not. The research goal of this thesis project was defined as:

*Investigate and construct a prototype that gives the user a visual overview of a patent collection up to 1000 documents and allow the user to browse and navigate through this collection by using a multi-touch input device and be able to create a subcollection of two to five patents that are most relevant based on the patent application..*

In order to achieve this goal, we had to increase our knowledge about patents, multi-touch interactions and previous research. We started this project with a literature study in which we have searched for related work in the field of document collection visualization and patent visualization. The EPO have arranged several workshops with patent examiners from different fields. These workshops provided us information about the patent structure, the current application and search strategies. In order to create a multi-touch application, we decided to make use of the Microsoft Surface Touch Beta API. With this API you can quickly create visible results and small changes are needed to export the application to the Surface.

During the literature study we have found different solutions to visualize large patent collections. During an experimental phase we have created several small applications to experiment with these solutions and solved individual problems related to visualization, navigation and interaction. The applications gave us early feedback about how multi-touch interactions work. At the end of the experimental phase we have chosen our main visualization technique, the zoomable user interface. The zoomable user interface was the starting point of our prototype.

Before we started implementing the application, we created a list of functions and sketches for our prototype. The sketches gave early feedback and ideas to improve the application. Implementing the prototype resulted in a number of implementation challenges and choices which needed to be solved. The application was continuously improved based on new ideas and feedback. Once the prototype was finished, we arranged some discussions with end-users and experts to get feedback on our application.

We are convinced that we have come up with an interesting and new way of navigating and browsing through large patent collections. We are able to provide an overview of up to 1000 patents and use touch input for natural interactions. The user can customize the application by adding annotations, creating subcollections and order the collection based on different properties. Together, these techniques will help the user in analyzing the patent collection and create a subcollection of two to five most relevant patents based on the patent application. The custom menu has shown potential. The examiners will begin to remember the gestures required for each action when they work for longer time with this menu. Further evaluations with end-users should provide feedback about the usability and whether it improves the examination time of a patent application.

The EPO had visions of a new user interface for patent examination that uses multi-touch interactions. We created one possible solution to indicate that it is possible to create such a new user interface which is controlled with multi-touch interactions. The application does not run on other operation systems than Windows. But this was not the purpose of this project, we wanted to see if it was possible to make these vision reality through providing a solution. An important lesson learned from this project is that you should not expect to receive a dataset, in limited time and in the way you want it, from a large company. Large companies have stored their data in many different databases, which all provide their data in a different format. Pre-processing is needed to store the data in the format we want to have it. Pre-processing the data is a step that every application should make, since we are responsible for what data is visualized, but we have no control of how the data is provided by the databases. It is also important to start evaluating the application at an early stage of the project.

The three hypotheses (defined in Section 1.2), we thought would help in creating a good prototype were defined as:

- Zoomable User Interface creates a workable overview.
- Patents are recognized by their visual representation and spatial location.
- Multi-touch input devices create natural interactions.

Concluding, we can say that these three hypotheses helped creating a good prototype. The zoomable user interface creates a workable overview of the whole patent collection, but also includes a detail view of the patents. Different visual aspects of a patent document are used to create a good thumbnail representation, which describes the relevance of the patent. To stimulate the spatial memory of the patent examiner, patents can be sorted and subcollections can be created. The spatial memory also plays an important role in the context menu. The examiner starts to remember where specific actions are located. Creating an application that is controlled with multi-touch interactions creates more natural interactions than

in default desktop applications.

## 8.2 Contribution

The main contribution of this thesis project is a prototype that allows a new way of browsing and navigating through large patent collections by providing an overview of the collection and making use of multi-touch interactions. We had to start from scratch and rethink the user interface in order to let it work optimally with multi-touch interactions. This has led to a minimalistic user interface with few controls and elements in which the content itself is the main part of the user interface. In the past, little research was done in visualization techniques in the field of patents and patent collections. With the emergence of smartphones and tablet computers, the time seems to be right for a new approach.

The starting point of our application was a zoomable user interface. A zoomable user interface provides an overview of the patent collection in which the user can change the scale of the viewed area in order to see more detail or less, and browse through the collection. Zoomable user interfaces already exist for a long time, but have never been used to display large patent collections in combination with multi-touch interactions. We created a semantic thumbnail representation for each patent. Each patent document is visualized by a set of symbolic thumbnail representations. For each zooming level different thumbnail representations are generated that create the best summary possible in the available space. This allows us to create smooth transitions between the different zooming levels.

Touch input creates a more natural way of interacting than we are used to with mouse input. The combination of patent visualization and multi-touch is new. Our application lets the user browse through a large patent collection with touch input and gestures and the mouse is no longer needed. More users are familiar with multi-touch interactions, they have accepted touch input. The gap between multi-touch and mouse interaction is closing slowly.

## 8.3 Future work

Following the work presented in this thesis document, a number of open issues was identified, each of which is now briefly discussed. We have categorized them in three groups: work done that could be done if we had a few extra months, work that could be done if we had a few extra years and a vision of what the future might bring.

### 8.3.1 Within a few months

#### Open Patent Services (OPS)

The current implementation makes use of a locally stored dataset, containing XML files and images. This limits the possibilities of testing the application on different datasets. A

connection with OPS<sup>1</sup> gives us direct access to the databases and makes it possible to test our application on different datasets. In order to create an OPS connection, an OPS reader and an XML converter are required. The OPS reader needs a set of patent numbers as input and return a set of XML files as output. The XML converter will convert the XML files directly into our own patent structure. The only problem with OPS is that we are not sure whether it can provide all the required information.

### Patent sorting

The current implementation sorts the patents in a row. It does not include any relation between two adjacent patents that are located on top of each other. Different ordering algorithms, like creating islands, will provide more useful results. In island ordering, relevant patents are positioned close to each other. This makes it easier for the user to visually compare multiple patents with each other.

### Non-patent literature

The current implementation only works with patent documents. In some fields, the patent examiners have to search for non-patent literature like books, scientific articles and websites as well. The application needs to be extended to work with non-patent literature. For this, a connection with non-patent literature databases is needed and the thumbnail generation tool should be able to create a summary for non-patent documents.

## 8.3.2 Within a few years

### Thumbnail customization

Different fields require different thumbnail representations. Our focus was on creating the best summary possible in the available space, making use of the drawings, text and annotation icon. It depends on the field, whether text or images are more important. We think that different thumbnail representations for different field are needed. User evaluations are needed to determine what information is most important in each field. The end-users also have shown interest in customizing the thumbnail representation based on their own preferences. This way each user can decide how the thumbnails can be customized for an individual. This might be useful for recognizing patents.

### Patent and images citations

Inside the full-text information of a patent document there can be referred and/or cited to related patents and drawings. Cited patents are important for an examiner, since they might contain relevant information about whether a claim is exclusive or not. A link to cited patents, allow the user to easily study these. In some fields, the drawings are of high importance. A link to the referred drawings will directly show the right drawing, instead of manually searching through all drawings. Agatonovic [Agatonovic 08] propose an automatic approach for large-scale, parallel patent citations.

<sup>1</sup><http://www.epo.org/searching/free/ops.html>

### **End-user evaluations**

We created an application which allows the user to browse and navigate through a large patent collection on a different way as they are used. The only evaluation we have done, are discussions with end-users and experts. These evaluations do not give enough insight to prove anything about the usability. To prove this, we need to do more extensive and structured end-user evaluations. These evaluations contain questionnaires, interview and a set of tasks that the user should perform. We also need to evaluate the most important elements of a patent. This information is needed to improve the thumbnail representation and creating a better summary of the patents.

### **Different visualization**

This master thesis project was limited to display up to 1000 patents. For larger patent collections, the thumbnails become so small that you cannot add any information. For larger patent collections, more research is needed in other techniques like clustering and graphical visualization.

### **8.3.3 Vision**

If we look to the future, we believe that our application will be integrated in the current system. Important is that there come one single application instead of using many individual applications. The application will have a direct connection with all the different databases and will be extended so that the examiners can create search queries in this application as well. It will be possible to visually compare different search queries with each other, smooth animations are used to see the effect of removing or adding keywords to the search query. The application will be extended to work with much larger patent collections and all multi-touch interaction will run very smooth to get a direct feedback on an action.

We also think that multi-touch applications will have a good chance to be integrated in current systems. There is already a clear signal that indicates that more tablet and touch devices are sold. There is a high certainty that companies will upgrade their applications to work on these devices as well. This way the employees can work on different locations and will always have all their information and documents carried with them.



---

# Bibliography

- [Agatonovic 08] Milan Agatonovic, Niraj Aswani, Kalina Bontcheva, Hamish Cunningham, Yaoyong Li, Ian Roberts & Valentin Tablan. *Large-scale , Parallel Automatic Patent Annotation Categories and Subject Descriptors*. Proceeding of the 1st ACM workshop on Patent information retrieval - PaIR '08, 2008.
- [Arnowitz 08] Jonathan Arnowitz, Michael Arent & Nevin Berger. *Effective Prototyping for Software Makers*. The Morgan Kaufmann Series in Interactive Technologies, 2008.
- [Bau 08] Olivier Bau & Wendy E. Mackay. *OctoPocus: a dynamic guide for learning gesture-based command sets*. Proceedings of the 21st annual ACM symposium on User interface software and technology - UIST '08, pages 37–46, 2008.
- [Bederson 94] Benjamin B Bederson, South Street Mre & James D Hollun. *Pad ++ : A Zooming Graphical Interface for Exploring Alternate Interface Physics*. Proceeding of the 7th annual ACM symposium on User interface software and technology (UIST 94, Marina del Rey, pages 17–26, 1994.
- [Bederson 09] Benjamin B Bederson. *The Promise of Zoomable User Interfaces*. Human-Computer Interaction, 2009.
- [Benko 09] Hrvoje Benko, Meredith Ringel Morris, A. J. Bernheim Brush & Andrew D. Wilson. *Insights on Interactive Tabletops: A Survey of Researchers and Developers*. Microsoft Research, 2009.
- [Bier 04] Eric Bier, Lance Good, Kris Popat & Alan Newberger. *A document corpus browser for in-depth reading*. Proceedings of the 2004 joint ACM/IEEE conference on Digital libraries - JCDL '04, pages 87 – 96, 2004.
- [Buxton 07] Bill Buxton. *Multi-touch systems that i have known and loved*. Microsoft Research, pages 1–10, 2007.
- [Cockburn 06] Andy Cockburn, Carl Gutwin & Jason Alexander. *Faster Document Navigation with Space-Filling Thumbnails*. Proceedings of the SIGCHI conference on Human Factors in computing systems - CHI '06, pages 1–10, 2006.
- [Cockburn 08] Andy Cockburn, A M Y Karlson & Benjamin B Bederson. *A Review of Overview + Detail , Zooming , and Focus + Context Interfaces*. ACM Computing Surveys (CSUR), vol. 41, no. 1, pages 1–42, 2008.

- [Divoli 10] Anna Divoli, Michael Wooldridge & Marti Hearst. *Full text and figure display improves bioscience literature search*. PloS one, vol. 5, no. 4, January 2010.
- [Dunsmuir 09] Dustin Dunsmuir. *Selective Semantic Zoom of a Document Collection*. Technical report, 2009.
- [Feldman 97] Ronen Feldman, Willi Klosgen & Amir Zilberstein. *Visualization Techniques to Explore Data Mining Results for Document Collections*. Proceedings of the 3rd Annual Conference on Knowledge Discovery and Data Mining - KDD '97, pages 16–23, 1997.
- [Freeman 09] Dustin Freeman, Hrvoje Benko, Meredith Ringel Morris & Daniel Wigdor. *ShadowGuides : Visualizations for In-Situ Learning of Multi-Touch and Whole-Hand Gestures*. Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces, pages 165–172, 2009.
- [Giereth 07] Mark Giereth, Steffen Koch, Martin Rotard & Thomas Ertl. *Web Based Visual Exploration of Patent Information at Stuttgart*. Proceedings of the 11th International Conference Information Visualization - IV '07., pages 150–155, 2007.
- [Giereth 08a] Mark Giereth, Harald Bosch & Thomas Ertl. *A 3D treemap approach for analyzing the classificatory distribution in patent portfolios*. IEEE Symposium on Visual Analytics Science and Technology - VAST '08., pages 189–190, 2008.
- [Giereth 08b] Mark Giereth & Thomas Ertl. *Visualization Enhanced Semantic Wikis for Patent Information*. Proceedings of the 12th International Conference Information Visualisation - IV '08., pages 185–190, 2008.
- [Gress 10] Bernard Gress. *Properties of the USPTO patent citation network: 19632002*. World Patent Information, vol. 32, no. 1, pages 3–21, 2010.
- [Harrower 03] Mark Harrower & Cynthia a. Brewer. *ColorBrewer.org: An Online Tool for Selecting Colour Schemes for Maps*. The Cartographic Journal, vol. 40, no. 1, pages 27–37, June 2003.
- [Hearst 95] Marti Hearst. *TileBars : Visualization of Term Distribution Information in Full Text Information Access*. Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '95, pages 59–66, 1995.
- [Herman 00] Ivan Herman, Guy Melancon & M. Scott Marshall. *Graph visualization and navigation in information visualization: A survey*. IEEE Transactions on Visualization and Computer Graphics, vol. 6, no. 1, pages 24–43, 2000.
- [Isenberg 09] Petra Isenberg & Danyel Fisher. *Collaborative Brushing and Linking for Co-located Visual Analytics of Document Collections*. Computer Graphics Forum, vol. 28, no. 3, pages 1031–1038, June 2009.
- [Isenberg 10] Petra Isenberg, Danyel Fisher, Meredith Ringel, Morris Kori & Inkpen Mary. *An Exploratory Study of Co-located Collaborative Visual Analytics Around a Tabletop Display*. IEEE Symposium on Visual Analytics Science and Technology (VAST), pages 179–186, 2010.



- [Keim 02] Daniel Keim. *Information visualization and visual data mining*. IEEE Transactions on Visualization and Computer Graphics, vol. 8, no. 1, pages 1–8, 2002.
- [Kewaley 08] Susheel Kewaley. The Effect of Multi-Touch Technology on User Interface Design. , 2008.
- [Koch 10] Steffen Koch, Harald Bosch, Mark Giereth & Thomas Ertl. *Iterative Integration of Visual Insights during Scalable Patent Search and Analysis*. IEEE transactions on visualization and computer graphics, pages 1–14, 2010.
- [Lao 09] Songyang Lao & Peng Wang. *A Gestural Interaction Design Model for Multi-touch Displays*. Proceedings of the 23rd British HCI Group Annual Conference on People and Computers: Celebrating People and Technology - BCS-HCI '09, pages 440–446, 2009.
- [Lin 92] Xia Lin. *Visualization for the Document Space*. Proceedings Of The 3rd Conference On Visualization - VIS '92, pages 274–281, 1992.
- [Lupu 11] Mihai Lupu, Katja Mayer, John Tait & Anthony J. (Eds.) Trippe. Current Challenges in Patent Information Retrieval. The Information Retrieval Series, Vol. 29, 2011.
- [Microsoft 09] Microsoft. User experience guidelines: User interaction and design guidelines for creating microsoft surface applications. , 2009.
- [Morris 01] S.a. Morris, Z. Wu & G. Yen. *A SOM mapping technique for visualizing documents in a database*. Proceeding on International Joint Conference on Neural Networks - IJCNN '01, vol. 3, pages 1914–1919, 2001.
- [Moscovich 07] Tomer Moscovich. Principles and applications of multi-touch interaction. , 2007.
- [Rauber 01] Andreas Rauber & Alexander Muller-Kogler. *Integrating Automatic Genre Analysis into Digital Libraries*. Proceedings of the 1st ACM/IEEE-CS joint conference on Digital libraries - JCDL '01, pages 1–10, 2001.
- [Ridder 11] Michel de Ridder. Interaction and Visualization for Mining Large Document Collections. , 2011.
- [Robertson 98] George Robertson, Mary Czerwinski, Kevin Larson, Daniel Robbins, David Thiel & Maarten Van Dantzich. *Data Mountain : Using Spatial Memory for Document Management*. Proceedings of the 11th annual ACM symposium on User interface software and technology - UIST '98, pages 153–162, 1998.
- [Sengupta 04] Arijit Sengupta, Mehmet Dalkilic & James Costello. *Semantic Thumbnails - A Novel Method for Summarizing Document Collections*. Proceedings of the 22nd annual international conference on Design of communication: The engineering of quality documentation - SIGDOC '04, pages 45–51, 2004.
- [Siek 05] Katie A Siek, Yvonne Rogers & Kay H Connelly. *Fat Finger Worries : How Older and Younger Users Physically Interact with PDAs*. nternational Federation For Information Processing, pages 267–280, 2005.

- [Sun 08] Taotao Sun & Steven A Morris. *Timeline and Crossmap Visualization of Patents*. Proceedings of WIS, 2008, pages 1–12, 2008.
- [Wanner 08] L Wanner, R Baezayates, S Brugmann, J Codina, B Diallo, E Escorsa, M Giereth, Y Kompatsiaris, S Papadopoulos & E Pianta. *Towards content-oriented patent document processing*. World Patent Information, vol. 30, no. 1, pages 21–33, 2008.
- [Wigdor 07] Daniel Wigdor, Gerald Penn, Kathy Ryall, Alan Esenther & Chia Shen. *Living with a Tabletop: Analysis and Observations of Long Term Office Use of a Multi-Touch Table*. Second Annual IEEE International Workshop on Horizontal Interactive Human-Computer Systems (TABLETOP'07), pages 60–67, October 2007.
- [Wigdor 10] Daniel Wigdor. *Architecting next-generation user interfaces*. Proceedings of the International Conference on Advanced Visual Interfaces - AVI '10, 2010.
- [Yoon 02] Byung-Un Yoon, Chang-Byung Yoon & Yong-Tae Park. *On the development and application of a self-organizing feature map-based patent map*. R and D Management, vol. 32, no. 4, pages 291–300, 2002.

---

# List of Figures

1.1	First page of patent with number EP 0392853 A2, published 17 October 1990.	3
1.2	The user interface of the Viewer tool contains three sections: a text viewer, an annotation bar and an image browser.	4
1.3	A patent application is summarized with a visualization that describes the main concept of the application.	6
1.4	A claim tree shows the relations between the claims in a patent application.	7
1.5	The patent examination process is summarized in five steps.	8
1.6	An overview of different multi-touch devices that can be used as platform for our prototype.	8
1.7	The focus of this project lays on analyzing a subcollection up to 1000 patents in more detail.	10
2.1	Space filling thumbnails fills the screen with a thumbnail for each document in the collection. The mouse pointer indicates which document should be previewed larger [Cockburn 06].	12
2.2	Combining thumbnails with meta information of the document (title, author and year of publication) [Bier 04].	12
2.3	Force-directed layout of a patent family on the left and 2D matrix visualization on the right [Giereth 07].	14
2.4	Visual representation of combined search expressions on the left and an example of text query with Boolean operators on the right [Koch 10].	14
2.5	Overview of the workspace of Cambiera during an analysis session. Both analysts have arranged several search boxes and documents in the space related to their current hypotheses. [Isenberg 09].	15
2.6	With semantic zoom levels you can specify for each zoom level how much information should be visualized [Dunsmuir 09].	16
3.1	For one patent document we created different thumbnail layouts by changing the size and combining text and images in different ways. The highlighted thumbnail representation was used as starting point.	21
3.2	Different sketches for the user interface of the patent collection viewer based on the functions and tasks of the application. The focus was on displaying the patents and at the edges there is space for buttons and a legend. The highlighted user interface was used as starting point.	22

3.3	Different sketches for the user interface of the single patent viewer. They all contain the three most important aspects of the viewer: a text browser, an image browser and an annotation bar, but display these in different ways. The highlighted user interface was used as starting point. . . . .	23
3.4	Space filling thumbnails are used to create an overview of the patent collection by making optimal use of the whole screen. This figure provides the result of an experiment with space filling thumbnail. By hovering the thumbnails, a larger window with details of the document will appear. . . . .	25
3.5	In a list view representation of a document collection, each document is visualized by its most important meta information. This figure provides the result of an experiment with a list view. . . . .	26
3.6	A space filling symbolic representation of the patent collection in which each document is visualized by a rectangle and a background that indicates the classification of the patent. . . . .	26
3.7	The design space of the symbolic representation contains different elements that can be used to add information to the thumbnail. . . . .	27
3.8	Different types of tilebars are created by changing the layout, size and background color. The tilebars are generated based on five predefined annotation groups, each group contains one or more words and a highlight color. . . . .	28
3.9	Overview of the selection of representative colors. The top row show eight different colors used to indicate the classified Section of a patent and the bottom row shows eight different colors used to indicate the classified group of the a patent. . . . .	30
3.10	Thumbnail representation of an article in four different scales. We used the first page of the article as image. . . . .	31
3.11	Thumbnail representation of an article in four different scales. The amount of meta information and font size will gradually increase when the thumbnails become larger. . . . .	32
3.12	Thumbnail representation of an article in five different scales. The thumbnail representation combines meta information with images. The larger the thumbnail is the more information can be displayed. . . . .	33
3.13	Thumbnail representation of an article in five different scales. The thumbnail representation combines meta information, images and an annotation icon. The larger the thumbnail, the more information can be displayed. . . . .	34
4.1	Overview of possible actions created with a single finger input. Illustrations provided by Gestureworks ( <a href="http://www.gestureworks.com">www.gestureworks.com</a> ). . . . .	37
4.2	Overview of possible actions created with multiple fingers input. Illustrations provided by Gestureworks. . . . .	37
4.3	Freely explore the document collection with translation and scaling gestures. A zooming gesture increase/decrease the size of the sliding window and a translation gesture will move the sliding window through the collection. . . . .	38
4.4	The available space can be used more efficiently by adding constraints to the basic navigation. Only fully visible thumbnails will be displayed on the screen. . . . .	39

4.5	Three examples of selections made with touch input. For each a large preview of the selection is visible right above the fingertip. . . . .	40
4.6	Predefined keywords are highlighted both in the text viewer and the annotation bar. Together they give quick insight about the relevance of the document. . . . .	41
4.7	Two different methods to browse through the drawings. The left method returns no visual feedback on the drawings and the right method returns visual feedback on the drawings. . . . .	42
4.8	The basic custom context menu requires precisely clicking on the circles in order to active an action. . . . .	43
4.9	The circles in the basic context menu are replaced by lines which need to be crossed with gesture movements. . . . .	43
4.10	The final context menu requires gesture movements to start an action and is built with hexagons to structure them more efficiently. . . . .	44
5.1	The user interface of the patent collection viewer contains as few controls and elements as possible in order to focus on the content itself. Each document in the collection is visualized with a symbolic thumbnail representation. . . . .	46
5.2	The user interface of the single patent viewer contains a text browser on the left, an annotation bar at the center and an image browser on the right. . . . .	47
5.3	An overview of different semantic thumbnail representations of one patent for the different zoom levels. Each thumbnail gives the best summary possible in the available space. . . . .	48
6.1	An overview of the data structure of our application. XML data is first converted into our own patent structure and the application API will only load and visualize the data of the visible patents. . . . .	55
6.2	An overview of the different states and possible actions of our main application. . . . .	59
6.3	An overview of the different states and possible actions in the single patent viewer. . . . .	60
6.4	Overview of the states and events in the application. . . . .	60
6.5	Event timeline. The order of the low-level events (shown in columns) indicates the type of click (shown in rows). . . . .	61
6.6	We have used annotations both in the text viewer and in the annotation bar. The yellow frame visualizes the active part of the document. . . . .	62
6.7	Load and update times for 1000 rectangles (documents) in milliseconds. . . . .	64
6.8	Average milliseconds needed to update 1000 rectangles (documents). . . . .	65
6.9	The number of documents has a linear effect on the update time in milliseconds. . . . .	65
6.10	There is no need to update the patents that lay outside the border. This improves the performance and scalability of the application. . . . .	66
7.1	Four different events are mapped on three different actions. . . . .	70



---

# List of Tables

1.1	This table shows the structure of the table of concepts. This table contains a title, classification and keywords for the three most important concepts of the patent application. . . . .	6
3.1	Text and Thumbnail (TAT) is used to semantically change the amount of meta information to visualize for each zooming level [Bier 04]. . . . .	29
6.1	This table provides an overview of all multi-touch events supported by the Microsoft Surface Beta Touch API. . . . .	57
6.2	This table provides an overview of all manipulation and inertia events supported by the Microsoft Surface Beta Touch API. . . . .	58
7.1	All possible events that can be created with one or two finger input. Illustrations provided by Gestureworks ( <a href="http://www.gestureworks.com">www.gestureworks.com</a> ). . . . .	68
7.2	All possible events that can be created with one or two finger input. Illustrations provided by Gestureworks ( <a href="http://www.gestureworks.com">www.gestureworks.com</a> ). . . . .	69