LOCATION-BASED SEARCH ENGINE FOR MULTIMEDIA PHONES

Pasi Fränti[1], Andrei Tabarcea[1], Juha Kuittinen[1] and Ville Hautamäki[2]

[1]Speech and Image Processing Unit, University of Eastern Finland, Joensuu
{franti, tabarcea, jkuitti}@cs.joensuu.fi

[2]Institute for Infocomm Research, A*STAR, Singapore
vishv@i2r.a-star.edu.sg

ABSTRACT

Location-based search engine is an alternative approach for information retrieval to traditional location-based services based on fixed databases. This is a relatively new concept that aims at utilizing the location of user but without restricting to any fixed location-based service. In this paper, we outline a prototype solution for multimedia mobile phones based on web search, ad-hoc georeferencing, prefix tree structure and gazetteer. Experimental results show that the proposed solution finds search results that have higher or equal mean relevance than that of the GoogleMaps and YellowPages.

Keywords— Search engine, LBS, mobile device, location information, personal navigation, WWW.

1. INTRODUCTION

Exploiting the l*ocation* of the user has become popular during recent years due to increasingly wide availability of GPS positioning in multimedia phones. For instance, according to Nokia's estimate more than half of their phones will include GPS by 2010-2012. Moreover, in case of lacking GPS, positioning can also be based on cellular network or even use IP address for a rough estimation of location.

Locations-based services (LBS) such as Yellow Pages[1], Google Maps [2] and Nokia Ovi Services [3] are therefore expected to emerge very fast to our everyday life via mobile phones and other consumer electronics. Their main limitation, however, is that they are fully or partially based on databases where the entries must be explicitly georeferenced beforehand when added.

Search engines, on the other hand, are efficient in finding information from internet without any prior knowledge or explicit search structure. Their limitation is that the location of the user is not yet well utilized in the current solutions. There are two reasons for this. Firstly, the location of user was not as widely available as it is nowadays. Secondly, the information in WWW is rarely attached with the location for which it would be relevant.

In this paper, we propose an alternative solution based on web search and ad-hoc georeferencing. We denote this as *location-based search engine* to emphasize its seemingly small but significant distinction from location-based services. It aims at combining the benefits of web search and traditional location-based services exploiting the location.

The main problem of this approach is that only very few pages have explicit georeferencing in form of geotagging, using address field or by other means. However, it is rather common that web pages include street or postal addresses as free (non-tagged) text. According to [9] most of relevant services (especially commercial ones) can be found in this way. Based on this observation, the authors have constructed a prototype solution called *MOPSI Search*.

The general idea behind the solution uses the idea originally outlined in [5], but not implemented in practice until now. We describe here the technical solutions for implementing the system on multimedia mobile phone and provide experimental comparison of its search capability in comparison to existing LBS solutions such as GoogleMaps and YellowPages. The technical solution for the ad-hoc georeferencing uses the same principle as in [11].

The workflow of the proposed solution is as follows. Once the location is given, we search for relevant web pages based on given keywords, extract potential address information, and compare them to the entries in a gazetteer. Positive results are returned as search results according to their distance relative to the user location. The location can also be used for plotting the target location on the map or by giving navigational information towards the location.

The rest of the paper is organized as follows. In Section 2, we discuss related work. In Section 3, we give detailed description of our solution called MOPSI search engine (see Fig. 1). It uses Google as a search engine and

[1] http://en.02.fi/yellow+pages/
[2] http:/maps.google.com/
[3] http://www.ovi.com/services/

978-1-4244-7493-6/10/$26.00 ©2010 IEEE

post-processes its search results to obtain information relevant to user's location. A prototype applied for Finland is demonstrated in Section 4. Experimental search results are also shown and briefly compared to two commercial solutions. Conclusions are then drawn in Section 5.

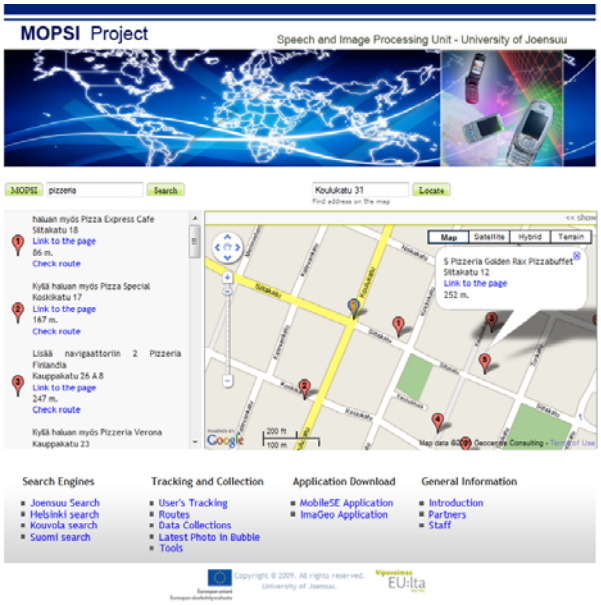

Fig. 1. Web interface of the MOPSI search engine.

2. RELATED WORK

In the recent years there have been related efforts to implement location aware web search engines. Most of the documented efforts are focused on creating explicit location information detectors that are used in the form of tags, in contrast to our approach where no explicit tags are used. Otherwise the approaches are similar: they rank the web pages according to their spatial contents. In some solutions, the found web pages are indexed for future search.

2.1. Web Search

The application in [2] uses web pages as the main source of location-based information. It relies on web crawling, which is targeted to create topical web indexes. The goal is to locate only links relevant to the topic of concern, and avoiding exhaustive crawls. The approach is similar to ours in a sense that its uses open web for detecting postal addresses and descriptive information. However, we don't use a web crawler but we rely on an external search engine.

The application in [7] consists of an indexed collection of web pages and supports both pure text and spatial indexing, which includes spatial relationships such as "inside", "outside" and "near". Conversely, it enables user to specify a search query and a geographical location. The queries are then tested for ambiguity and alternative options are presented for disambiguation, which is done before submission to the search engine by using geographical ontology. Relevance ranking is executed with respect to the non-spatial and spatial elements of the query.

In respect to existing commercial services such as Google Maps, Bing Local [4], Yahoo Local [5] and Yellow Pages, our goal is the same: provide location-relevant information to the user. However, the existing applications are mainly based on commercial databases, and only into limited extent, exploit the results on real-time web search.

2.2. Address Extraction

An essential part of our application is to detect and extract locations (addresses) from web pages. In the reviewed literature, various types of address detection methods have been published during recent years.

Methods of detecting the location of a web resource are initially found in [3, 9]. In [3], "whois" records are used for retrieving phone numbers of network administrators, which are used together with zip code and area database to assign coordinates to Class A and B domains. In [9] hyperlinks, meta tags and postal addresses are also used for additional information. The addresses are detected using a postal code database with latitude/longitude information. We also detect the postal addresses from database with latitude/longitude information, but our granularity is higher, as we use street names and street numbers.

In [8], regular expressions are used to detect patterns of typical address elements and the results are validated using a database, but the focus of the paper is how to rank the search results. Earlier prototypes of our application were also based on regular expression to detect Finnish street names. However, we later change this and started to use gazetteer in order to cover the addresses more thoroughly.

One way to detect addresses from free form text is to build a classifier and let it detect addresses from the web-pages as in [12]. However, customizing the classifier to other languages and countries takes a considerable work as new ground truth tagged text corpus must be created by hand. In our approach, no ground truth tagging is needed. The only things needed are a gazetteer and simple rules on how the street name appears relative to other address fields. Efficient use of the gazetteer is possible because we know the user's current location and the interest area is limited to services close to him. We can therefore build fast access structure to a partial gazetteer.

In [1], database is used to detect and validate addresses from web-page. In addition, different variations of street names are detected and the text is verified for occurrences of address elements such as street names, city names and zip codes. As a result, correspondence between data and text can be examined. We use similar idea for identifying

[4] http://www.bing.com/local/
[5] http://local.yahoo.com/

address elements in our geoparsing algorithm and validating the address by geocoding. The difference is that we use explicit geocoded database and rely on street-name detection, while [1] uses freely available geocoders.

In [4], a syntactic approach to postal address detection is proposed consisting of two steps: vision-based text segmentation and syntactic pattern recognition. The text segmentation analyses the html tags and detects cue blocks (for the purpose of indications, annotation, and explanation) and body blocks (main text body content). Recognition of postal addresses relies on calculating the confidence of the detected blocks, which in turn is based on tokenization of the words. The tokenization process uses city names, state names, street and organization suffixes, but not street names. Our approach is simpler, as we filter out all the html tags, and our address detection relies on street-name detection.

3. MOPSI SEARCH ENGINE

3.1. Overall Scheme

MOPSI search engine consists of the following components:

1. User interface for web and mobile devices.
2. Core server software.
3. Geocoded address database.

These components and their relations will be discussed in the following subsections.

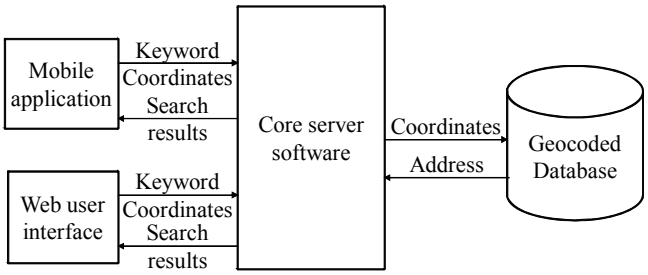

Fig. 2. Overall scheme of the MOPSI search engine.

3.1. User interfaces

The user can access the search engine either by using the mobile application (considering he has no access to a computer) or by using the web interface (considering he has no means of GPS positioning).

The mobile application is based on Java and compatible with most smartphones on the market today. It can use the internal GPS receiver of the phone or external Bluetooth device. The mobile application has also tracking and data collection modules, but these are not the focus of this paper. User can make a query to the MOPSI search engine using his current coordinates (if retrieved from the GPS module) or using previously stored coordinates. The results will be displayed as a list ordered by the distance to the user's location, or displayed on a map.

In order to use the location-based search engine, the mobile application has to be connected to the Internet. Communication between the mobile device with the corresponding client applications and the MOPSI search engine is implemented by HTTP protocol. The mobile application uses Java ME and is compatible with the S60 platform, which in turn supports information about positioning through Location API or through the Bluetooth protocol to connect to an external receiver. A C++ version for Symbian OS is currently under development.

The web interface is designed using PHP language, having AJAX capabilities and is used for accessing the location-based search engine from a computer. User can input the location by the street address or by selecting a location on the map. Search results are shown as a list, and on Google Maps, optionally with a route to the location.

3.2. Core server software

Considering that most web-pages contain street or postal addresses rather than explicit location information, our approach to the location-based search is to detect address portions from web-pages and use the distance from the user's location as an additional relevance criterion.

For implementing this concept we use an external search engine for performing query-based search and post-process its search results by extracting postal addresses, which are then translated into coordinates using the Geocoded database. The core server software performs these operations and is the main component of our location-based search engine.

The software (Fig. 3) has the following components:

1. Relevant municipalities detector
2. Page parser
3. Address and description detector
4. Address validator

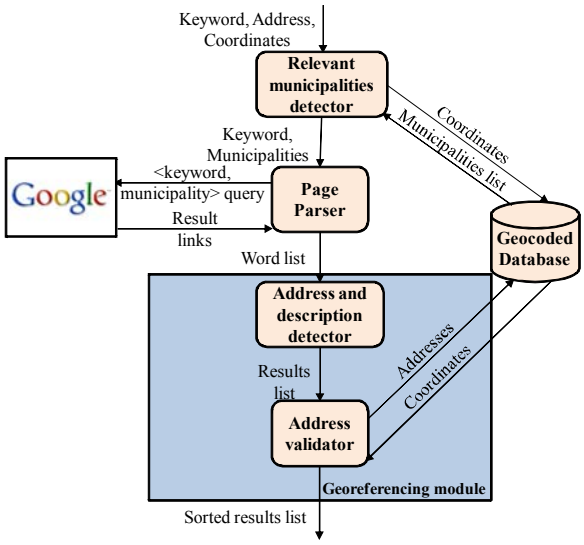

Fig. 3. Overall scheme of the MOPSI search engine.

The *Relevant municipalities detector* uses geocoded database to find all municipalities within a predefined range (e.g. 10 km) user's location.

The *Page parser* then uses an external search engine to perform <keyword, municipality> query for every municipality detected in the previous step. It downloads the web-pages found, strips the html tags, and extracts the text.

The *Address and description detector* searches for address blocks in the word list returned by the Page parser. It uses optimized prebuilt data structures (prefix trees) which contain all the street names for each municipality used for the search. It uses a simple text-matching algorithm to detect the address block, the description of the search result and the telephone number.

The *Address validator* verifies the addresses from the result list from the previous step using the Geocoded database. The addresses which are not found in the Geocoded database are discarded. The remaining search results are sorted by the distance to user's location, and filtered using a distance threshold in order to avoid excess information. The results are then shown as a sorted list on the mobile display.

3.3. Geocoded database

The process of assigning geographic coordinates is known as geocoding. Finding corresponding coordinates of a given street address requires a gazetteer, which is a geocoded street-name database that connects any given addresses to its exact locations (coordinates). Such databases are commonly available (although not necessarily free), and can be purchased for given (or specified) regions.

The *Geocoded database* is used to store all the addresses in the form of <street, number, municipality>, and their corresponding geographical coordinates. It is used every time an address needs to be converted into a location and vice versa. Furthermore, the Geocoded database is used for creating data structures (arrays of prefix trees) needed in the detection of the address blocks in the web-pages.

The speed and accuracy of the database is enhanced by using a database management system that implements *Open Geographical Information System* (OpenGIS) specification, or other spatial and location-data standards for faster query results. The database management systems that can be used include MySQL with spatial extensions, PostgreSQL with PostGIS or Oracle Spatial. In MOPSI, we use MySQL.

4. PROTOTYPE SEARCH ENGINE FOR FINLAND

We implemented a prototype application for location-based web search in Finland. The user interface of the engine was first a web page, accessible both from PC and multimedia phone, but we later developed Java-application for the mobile version. In PC (Fig. 1), user gives location as physical address, which is converted to coordinates in server-side using the Geocoded database. In mobile phone (Fig. 4), the software obtains user's coordinates by GPS, and then queries the server using HTTP Post. The mobile version also displays the nearest known address of the user.

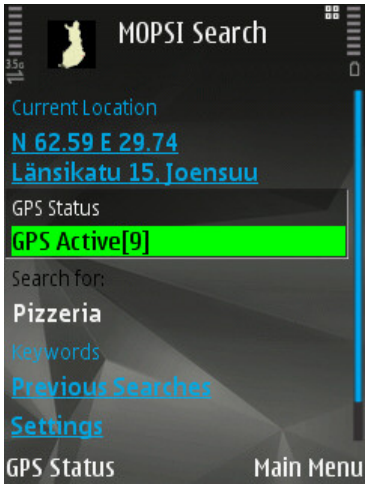

Fig. 4. Mobile application

4.1. Workflow of the prototype

When user inputs keyword(s), the server side application then searches all municipalities within a certain distance from user's location. The application creates *keyword-municipality* pairs for every commune and executes Google search using these as the search query. The contents of the first 10 returned web links are downloaded to the server and analyzed to find addresses. We use pattern matching technique that relies on *prefix trees* to detect the street names. If a match is found, the application searches for other typical address elements, such as street numbers, postal codes and municipal names followed by the candidate street name. If other elements are found, they are collected together as a street address candidate.

Unlike traditional web search, the search result is not necessarily the entire web page but can be only a part of it, most likely in case of a service directory that lacks a formal structure. In this case, the title of the entire page is not enough to identify the relevant part. However, the text closer to the address element can contain a description that could separate the search result from the rest of the web page, and possibly from several other non-relevant entities in the same page. In the current version, we extract description simply by taking a part of the text preceding the address.

The search result is built from the following: description phrase, address, web link, map link and distance from user's location. The current prototype uses free map service provided by Google Maps on server-side and commercial map service provided by the Finnish National Land Survey on mobile-side. Finally, the application

displays the results as a list ordered by the distance, see Fig. 5.

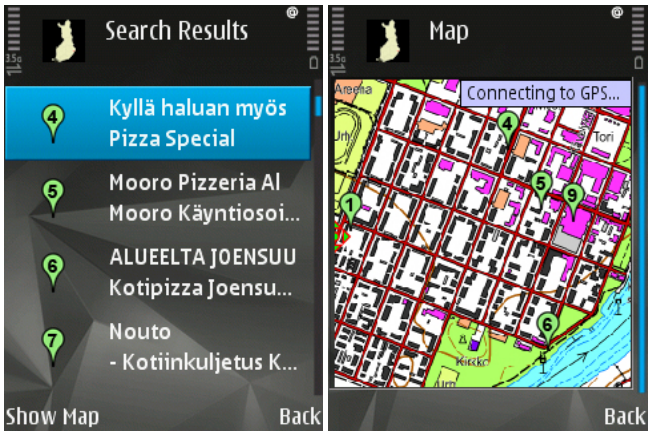

Fig. 5. Example of search results for keyword pizzeria.

4.2. Experiments

In order to test our prototype application, we simulated the following scenario: the user is in the center of an urban or rural municipality and his search is restricted to that municipality. His targets can be commercial (i.e. restaurant) or non-commercial (i.e. police station).

We selected 10 Finnish urban municipalities and 10 Finnish rural municipalities (see Table 1). The urban municipalities were selected according to their size, and geographical location was taking into account when selecting the rural municipalities. The search was performed using 10 test keywords which were divided into 5 commercial ones (hotel, restaurant, pizzeria, cinema, car repair) and 5 non-commercial ones (hospital, museum, police station, swimming hall, church).

Table 1. Municipalities used for testing

Type	Municipalities
Urban	Helsinki, Espoo, Lahti, Turku, Tampere, Jyväskylä, Vantaa, Oulu, Kuopio, Joensuu
Rural	Kuhmo, Ulvila, Lapua, Pieksämäki, Sodankylä, Forssa, Somero, Laihia, Kitee, Salla

In addition, the same queries were submitted to two other location based web services: Google Maps and Finnish Yellow Pages. According to [10], Google Maps uses multiple sources for providing the results. For instance, information submitted by local business owners or public directories, enhanced content such as reviews, photos submitted by users to various services, user generated content and other websites crawled. Yellow Pages, on the other hand, is a public directory of local business and the data is maintained and verified by users and administrators.

The search results are sorted by distance and filtered according to each municipality. Duplicates were manually removed and the distance was calculated using the gazetteer. There is no standard methodology for testing the overall relevance of the location-based search results considering relevance both by topic and distance. We therefore formulated our own criterion based on the evaluation methodology proposed in [6]. First, we compare the total number of search results from the tested services without considering their relevance. Secondly, we compare the relevance of the search results by evaluating them as (1) *relevant*, (2) *somewhat relevant* or (3) *not relevant*.

Table 2. Number of results for our test scenario

Query type	MOPSI prototype	Google Maps	Yellow Pages
Rural non-commercial	69	29	0
Rural commercial	245	92	189
Urban non-commercial	148	413	37
Urban commercial	1412	813	1337
Total number of results	2352	1405	1597
Overall mean relevancy	1.59	1.66	1.28
Overall std. deviation	0.84	0.89	0.54
Overall std. error	0.02	0.02	0.01

As shown in Table 2, our prototype provides more results than Google Maps or Yellow Pages, and is slightly more relevant (smaller the better) than Google Maps, but less relevant than Yellow Pages. With the exception of urban non-commercial queries, our prototype gives more results. Unsurprisingly, Google Maps has more results than Yellow Pages for non-commercial queries and less for commercial ones.

Contrary to Google Maps and Yellow Pages, our prototype relies on the relevance of the results of the external search engine. The search results are basically the addresses found in the results of the external search engine, whilst Google Maps has multiple criterions for relevance and Yellow Pages is controlled by human evaluators. We therefore compare relevance of the search engines on an average basis using a combination of two keyword categories for each comparison.

A comparison of rural municipals and non-commercial keywords resulted in our search engine (MOPSI) having mean value less than that of Google Maps (2.35 comparing to 2.48), whilst Yellow Pages did not return any results. This indicates a relatively high number of relevant results obtained by our search engine. The same method was used in Table 3, in which the results show a diverse number of relevant links by the MOPSI search.

Table 3. Mean relevance for our test scenario

Query type	MOPSI prototype	Google Maps	Yellow Pages
Rural non-commercial	2.35	2.48	0
Rural commercial	1.71	1.33	1.36
Urban non-commercial	2.20	2.17	1.59
Urban commercial	1.46	1.41	1.27
Overall mean relevancy	1.59	1.66	1.28

The results show that the relevance of MOPSI search engine is close to Google Maps, except for the rural municipalities with non-commercial keywords, for which we get higher number of results, but their overall relevance is smaller. Yellow Pages is the most relevant service, but having the lowest number of results (except for commercial urban keywords). In urban areas, the number of results by MOPSI search engine is close to Google Maps and Yellow Pages, whilst in rural areas it is higher.

4.3. Observations and known problems

One of the main problems is that the search can produce vast amount of irrelevant results. Mobile devices have restricted resources (data transfer bandwidth, small display) and often poor usability. The application should therefore be able to filter out flawed or less relevant data much better than a web-based application to make the browsing of the search results easier. Because of this, the current version downloads only a limited amount of links but further ideas to improve this would be desirable.

Another problem is that, unlike normal web searches, we allow the results include parts of web-page. This is very useful for finding services that do not have their own web page but exist in ad hoc service directories such as: http://www.pizza-online.fi/. However, an open problem is how to extract only the relevant part from a web page without any prior knowledge about its structure.

Despite previously mentioned problems, a lot of positive results are also achieved. When it comes to non-commercial services, the web pages are good repository of location relevant information. In rural areas of Finland where business is small, more services are found from web pages than from commercial databases.

5. CONCLUSIONS

The concept of location-based search engine was outlined, and its design issues and problems were discussed. MOPSI prototype implementation in Finland was demonstrated with qualitative comparison using typical search examples. The idea itself can be generalized worldwide although practical implementation would need a local gazetteer, or at least enough knowledge to be able to extract addresses from the web pages with reasonable accuracy.

The results indicate that the proposed approach has a lot of potential for practical applications. Most of the problems are related to technical matters and implementation issues. For instance, we use real-time search and page parsing whereas commercial solutions such as Google can use large computer capacity and avoid computational problems by pre-processing, huge storage (cache), and indexing.

6. REFERENCES

[1] Ahlers D. and Boll S. 2008. Retrieving address-based locations from the web. Int. Workshop on Geographic Information Retrieval, 27-34, Napa Valey, CA.

[2] Ahlers D. and Boll S. (2008b). Urban Web Crawling. ACM Int.workshop on Location and the web. Vol. 300, 25–32. Beijing, China.

[3] Buyukkokten O., Cho J., Garcia-Molina H., Gravano L. and Shivakumar N. (1999). Exploiting geographical location information of web pages. WebDB (Informal Proceedings), – dbpubs.stanford.edu

[4] Can L., Qian Z., Xiaofeng M. and Wenyin L. (2005). Postal address detection from web documents. Web Information Retrieval and Integration. Int. Workshop on Challenges in Web Information Retrieval and Integration, 40 - 45

[5] Hariharan G., Fränti P. and Mehta S. (2002). Data mining for personal navigation. SPIE Conf. on Data Mining and Knowledge Discovery, vol. 4730, 355-365.

[6] Jansen B.J., Molina P.R. 2006. The effectiveness of Web search engines for retrieving relevant ecommerce links, Inf. Processing & Management, 42 (4), 1075-1098

[7] Jones C.B., Abdelmoty A.I. , Finch D., Fu G. and Vaid S. (2004). The SPIRIT spatial search engine: Architecture, ontologies and spatial indexing. LNCS Lecture Notes in Computer Science, Springer.

[8] Lee H.C., Liu H. and Miller R.J. (2007). Geographically-Sensitive Link Analysis. IEEE/WIC/ACM Int. Conf. on Web Intelligence, Silicon Valley, CA, 628–634.

[9] McCurley, K.S. (2001). Geospatial mapping and navigation of the web. Int. Conf. on WWW, 221-229.

[10] Pasztor E., Egnor D. 2006. Generating structured information, US Patent Application 20060200478

[11] Tabarcea A., Hautamäki V. and Fränti P. (2010). Ad-Hoc georeferencing of web-pages using street-name prefix trees. WEBIST Int. Conf. on Web Inf. System and Technology, Valencia, Spain.

[12] Viola P. and Narasimhan M. (2005). Learning to extract information from semi-structured text using a discriminative context free grammar. ACM SIGIR Conf. on Research and Development in Inf. Retrieval, Salvador, Brazil, 330–337.