# Continuous Hierarchical Exploration of Multimedia Collections

Tomáš Grošup, Juraj Moško and Přemysl Čech
SIRET research group, Department of Software Engineering,
Faculty of Mathematics and Physics, Charles University in Prague,
{grosup, mosko, cech}@ksi.mff.cuni.cz,
www.siret.cz

***Abstract*—Preserving continuity between individual exploration steps in a process of multimedia exploration is a concept of natural intuition that sometimes decides if a particular exploration system is usable or not. One of ways how to emulate the continuity of the exploration process is adding some sort of granularity into this process. Then anyone who uses such system can explore particular areas in less or more details.**

**In this paper we proposed new concept, *hierarchical querying*, which keeps consecutive steps of the exploration process more tight to each other. As a second concept, which directly supports the continuity of the exploration process, we proposed preservation of a user context between consecutive steps of the exploration process.**

**In addition, we also presented an evaluation process and architecture design of our multimedia exploration system. For a validity confirmation of our ideas, we have implemented all proposed concepts in a web application that is accessible online.**

## I. Introduction

As a result of rapid expansion of new approaches how to create and acquire multimedia data, the Internet is flooded with huge amount of multimedia data, especially image collections. Therefore, in last decades there arise many techniques that provide support for understanding and searching such collections, like automatic annotations or content based search. One of typical scenarios of the latter one is *query by example*. In the *query by example* scenario a user typically states his/her intent by providing a multimedia query object and some conditions on results of the query, e.g. a threshold on similarity, or a number of most similar objects to the query object. While this scenario is useful when searching for particular objects, it is not suitable in cases when the user does not know in advance what he is searching for. Such a situation happens when the user just wants to explore a collection and/or (s)he wants to get a big picture of what kind of objects the explored collection is containing. Needs for this kind of scenario lead to the uprise of a *multimedia exploration* [2].

From the beginning, the multimedia exploration focused on a user interface and a visualization component that should truly represent an underlying multimedia space. As a result of it, many different content-based exploration systems have been proposed so far. Some of them tries to visualize the collection on a hue sphere [9], other uses a 3D cylinder [10] and other visualizes images on a 2D canvas, where objects can even overlap [3].

Our previous works [6], [7] have used for the visualization component a particle physics model [4], where similarities between objects directly influence a position of objects. Because the inter-objects similarities are transformed into attractive and repulsive forces between particular objects.

Usability of a multimedia exploration system depends also on effectiveness how user actions are evaluated, since the collection can contain hundred thousands or millions of images. Hence it is necessary to provide the multimedia exploration with some index support for query evaluation [1], [8].

## II. Continuous Hierarchical Exploration

A previous version of our demo application [7] focused on efficiency of the exploration process, where support for some *metric access methods* was provided. The work also discussed two different multimedia exploration techniques, see upper part of Figure 1. The first one is called *iterative querying*, where the exploration process starts in some initial view and consecutively uses a well-known query based approach as a basic modality. While in the second one, *iterative browsing*, a user with his/hers queries follows a hierarchy of some exploration structure. As was mentioned in the previous work, both of these techniques have their drawbacks, *iterative browsing* depends on compactness of an underlying structure and *iterative querying* depends on the first selected collection for a zero view. On the other side, both ideas have some advantages. In case when the hierarchical structure that supports *iterative browsing* truly follows the hierarchy of the indexed space, subsequent steps in a browsing process suppose to reveal details more continuously. The problem is in the claim "truly following the hierarchy of indexed space", since to achieve such an illusion is usually difficult in an arbitrary space, because creation of such hierarchical structure can be very expensive. Conversely, the creation of an index support for *iterative querying* seems to be very efficient. Following these ideas we propose a combination of these techniques and named it *hierarchical querying*.

### A. Hierarchical querying

An idea of *hierarchical querying* takes a concept of the hierarchy from *iterative browsing*, and a concept of querying that does not directly follow any structure from *iterative querying*. At the beginning of the exploration process a user is provided with objects from a *page zero*, the very same

978-1-4673-6870-4/15/$31.00 

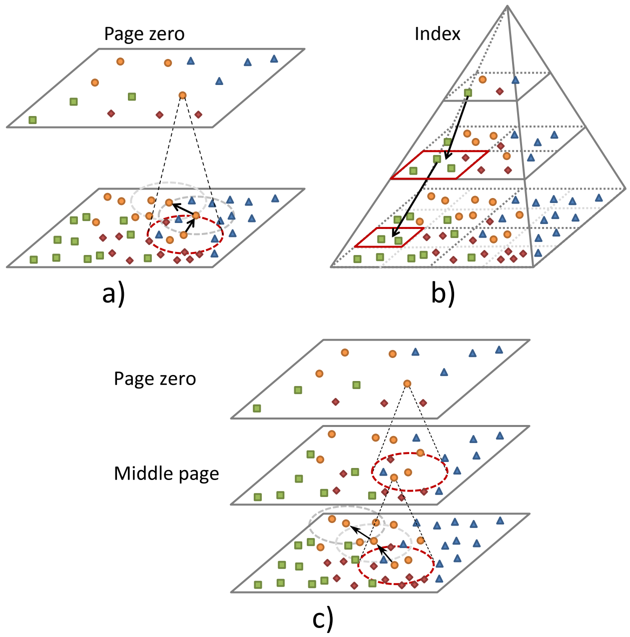


Fig. 1: a) *iterative querying*. b) *iterative browsing*. c) *hierarchical querying*.

approach as in *iterative querying*. But, when it comes to next querying from this *page zero*, instead of returning objects from a whole database, only objects from some subset are returned. This subset, we denoted as a *middle page* (see in the bottom of Figure 1), is a collection of pre-selected objects and is created during a phase of preparing index support for a whole exploration environment. The *middle page* consists of all objects that are in the *page zero* supplemented with additional objects selected from the rest of the database. We used randomly selected objects for both, the *page zero* and the *middle page*, but more complex selection methods known from an area of a hierarchical clustering or a pivot selection can be used. It should be known that the number of objects selected for the *middle page* also influences an illusion of *hierarchical querying*. We derived the object count from the number of objects in the *page zero* and a variable parameter $pow$ that determines the granularity of details in the *middle page*, see Equation 1, where $S$ is the size of the whole collection.

$$objects_{MP} = \left\lceil objects_{PZ} + (|S| - objects_{PZ}) * \left(\frac{1}{2}\right)^{pow} \right\rceil \tag{1}$$

The parameter $pow$ directly controls the number of objects in the *middle page*, in Table I you can see the counts for few values of $pow$ in case when the size of the whole collection is $|S| = 20,000$ and the number of objects in the *page zero* is 20. We added a possibility to parametrize the number of objects in the *middle page* because different multimedia collections have different level of granularity when it comes to classification of images. For example, if most of the images are from the same class, the number of objects in the *middle page* should be higher than in a case when the objects are distributed in the many small classes. Since, it is more desirable to get higher perspective over the collection than just representatives of only one class.

TABLE I: Influence of $pow$ on object count in the *middle page*

| $pow$ | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|
| $objects_{MP}$ | 10,100 | 5,015 | 2,518 | 1,269 | 645 | 333 |

### B. More continuous way

The second major contribution of this paper is an implementation of preserving a user context between a visualization of particular exploration steps. In other words, a state before a query is evaluated should adhere a state after the query evaluation more continuously. The first requirement for more continuous transition is binding of a new result set to the previous one. As you can see in Figure 2 a process of one exploration step starts with selection of the query object (by the user) in step $I$. After the query is evaluated, the back-end sends the new result set to the visualization component, while the old result set still remains on a screen. Both results sets, the new and the old, are compared and objects from their intersection are tagged, they will be preserved in a new visualized state (step $II$). After the rest of objects from the old result set are removed from visualization (step $III$), the rest of the new objects, which are not already visualized, are placed next to the old preserved objects. For a placing method of the new objects, we chose a strategy to place each new object in the position next to the most similar object that is already placed, as depicted in step $IV$. There is also possibility to take into consideration more than only one the most similar object and place the new object depending on distances to more similar objects. When all objects from the new result set are placed in visualization, the process of adjusting a position according to the particle physics model is iteratively run (step $V$) until the position of each object is stabilized (step $VI$).

## III. Demo Application

Our demo application is an online software for content-based exploration of image collections and it is accessible from a web browser at [5]. As you can see in Figure 6, the demo is based on a client-server architecture.

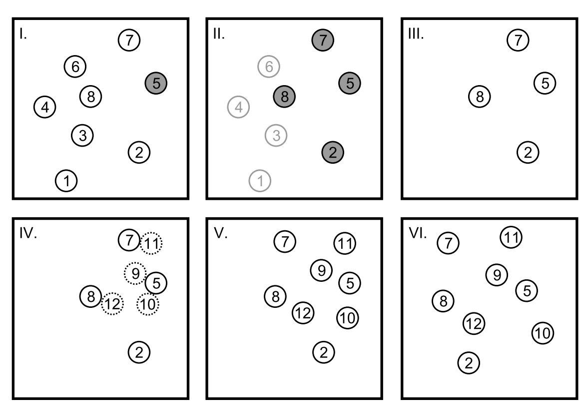


Fig. 2: Schema of preserving a user context during one exploration step.

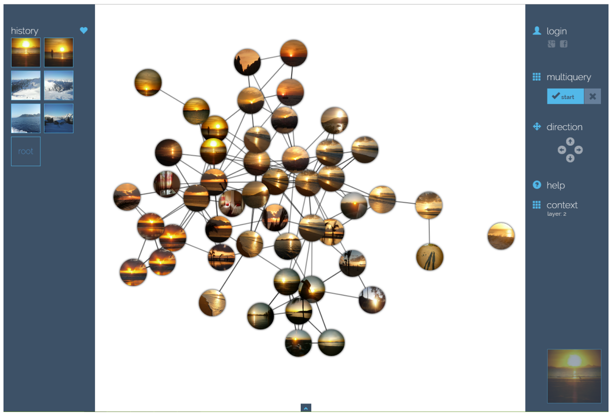


Fig. 3: View of the whole demo application. You can see a history panel on the left side, some information about a current context on the right side and a current exploration view on the main canvas.

The client is a single-page application and is responsible for handling user actions, for preserving a context between individual exploration queries and for visualization of result data.

The server is an ASP.NET web application which handles HTTP requests, creates and manages a collection of metric indexes and prepares data for the client part in form of a similarity graph.

When a user opens the demo, a *Query Controller* sends a request to the server for the *page zero* via an AJAX call. This request is handled by a *Request Handler* that retrieves result objects from the *page zero*. The result objects are a collection of multimedia objects containing an identifier of an object, a specific URL address where the multimedia object itself can be retrieved from and index-specific metadata that is preserved between individual exploration steps. After the objects are retrieved, the result collection is passed to a *Similarity Graph Generator* that is responsible for computation of distances between the objects in the result and for saving them in a similarity matrix. The matrix is subsequently transformed into a similarity graph by keeping only edges which a similarity value is higher than a system-specific threshold. The similarity graph is returned back to the client in a form of a JSON structure containing the nodes and the edges.

On the client side, the received similarity graph is passed to a *Context Controller* that compares a new result with the previous one. Objects which are not present in the new result are removed from a screen, the rest of objects stay at the same position where they had been located before the query started. After that, each remaining object of the new result is placed next to its most similar object already added in a visualized space. This can be computed very quickly, since the result set already contains the edges with their similarity weights. The graph with $x$, $y$ coordinates adjusted by the *Context Controller* is passed to a *Force Directed Visualization* component. Its responsibility is to tune the position of the visualized objects by using the particle physics model until a stable distribution of images on the screen is found.

In addition to visualization, the client provides a user interface for exploration of the visualized collection by selecting objects of a user interest. After one or more objects are selected, the *Query Controller* generates an exploration query from them and sends a query request to the server. Since each of the visualized objects contain the index-specific metadata, the server can retrieve a current context from the query request - a state in which user exploration currently is - and then, according to the context, the sever determines what layer (the *middle page* or the whole database) of an underlying index should be queried.

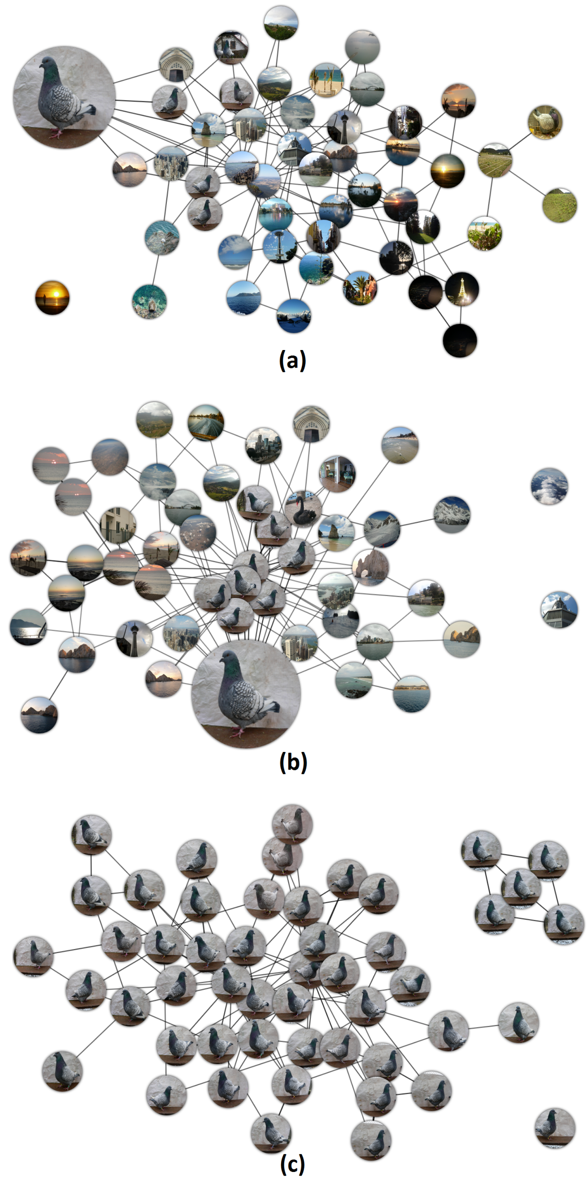


Fig. 4: Exploration in *hierarchical querying*. (a) Initial view of the *page zero*. (b) View of the *middle page* after clicking on a pigeon image. (c) View from the bottom of the hierarchy after clicking on the pigeon image in the *middle page*.

A layout of the client demo application is shown in Figure 3. A user can explore a collection by clicking on some images of his interest. For a return back in a querying history, (s)he can use a history panel in the left part and then continue

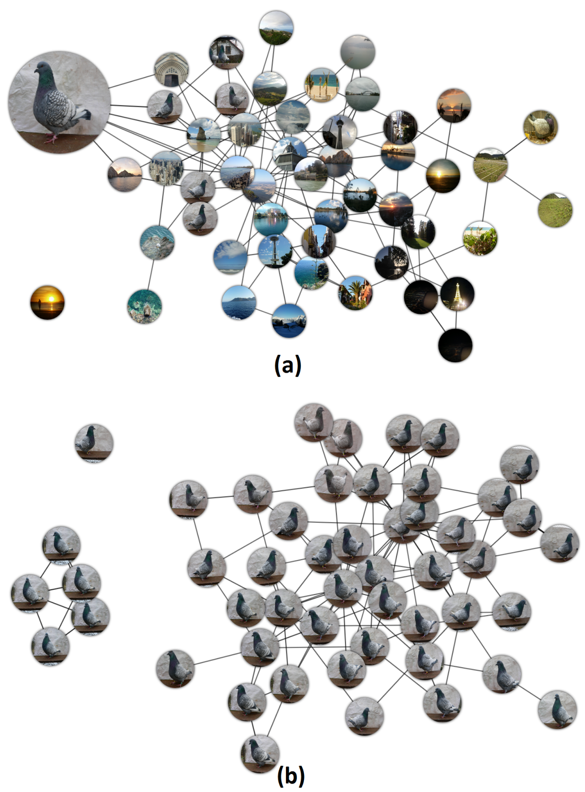


Fig. 5: *iterative querying*. (a) Initial view. (b) View after performing a $k$NN query with a pigeon image as a query object.

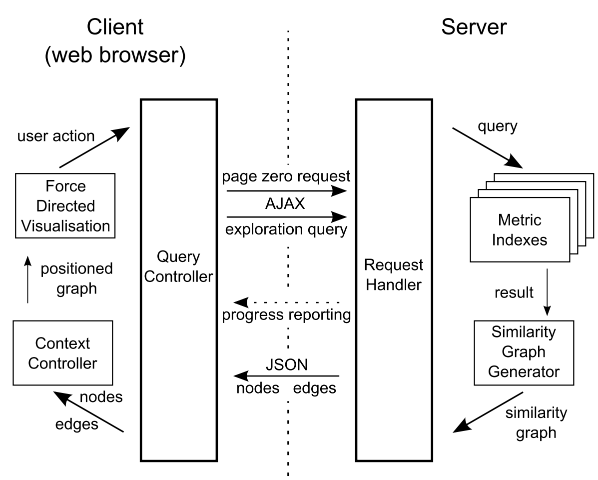


Fig. 6: Architecture of the demo application.

by exploring a different part of the visualized collection. And in the right part the user can see some information about the current context.

Figure 4 shows the exploration process of *hierarchical querying* in particular exploration steps, starting from the *page zero* (a), continuing with the *middle page* containing only some sample images (b) and ending in the bottom of the hierarchy where all objects from a dataset are stored (c). For a comparison, a process of performing the same exploration query in a concept of *iterative querying* is depicted in Figure 5. As you can see, absence of the *middle page* in *iterative querying* means that only the most similar objects from the whole dataset are retrieved, and such a very detailed view does not give the user an option to see results of a query from a bigger perspective.

## IV. Conclusion

In this demo paper we introduced continuity and hierarchy into our previous work. Our ideas came out of two previously discussed concepts *iterative querying* and *iterative browsing*. We tried to adopt advantages from both of them and proposed the new technique - *hierarchical querying*. From our perception, after *hierarchical querying* was implemented into our system the exploration process became more directed by a user. Moreover, introducing a preservation of the user context made the exploration process notably more fluent.

For further research we plan to implement more than one *middle page* into the hierarchy, to create multi-layer exploration system. We also plan to define and implement additional exploration operations so besides a current traverse (zooming-in) operation we could, for example, zoom-out of a specific part in the visualized space which should be useful for following the hierarchy also upwards. And as a next step we plan to verify proposed ideas with performing a user study.

## Acknowledgment

This research has been supported in part by Czech Science Foundation project 15-08916S and by Grant Agency of Charles University projects 201515 and 910913.

## References

[1] C. Beecks, P. Driessen, and T. Seidl. Index support for content-based multimedia exploration. In *ACM Multimedia'10*, pages 999–1002, 2010.

[2] C. Beecks, T. Skopal, K. Schöffmann, and T. Seidl. Towards large-scale multimedia exploration. In *DBRank 2011, Seattle, WA, USA*, pages 31–33, 2011.

[3] C. Beecks, M. S. Uysal, P. Driessen, and T. Seidl. Content-based exploration of multimedia databases. In *Proc. of International Workshop on Content-Based Multimedia Indexing (CBMI 2013), Veszprém, Hungary, 17-19 June 2013*, pages 59 – 64, 2013.

[4] T. M. J. Fruchterman and E. M. Reingold. Graph drawing by force-directed placement. *Softw. Pract. Exper.*, 21(11):1129–1164, Nov. 1991.

[5] T. Grošup, J. Moško, and P. Čech. Continuous hierarchical exploration of multimedia collections. http://www.siret.cz/HierarchicalExploration, 2015. SIRET Research Group.

[6] J. Lokoč, T. Grošup, and T. Skopal. Image exploration using online feature extraction and reranking. In *ICMR '12*, pages 66:1–66:2, New York, NY, USA, 2012. ACM.

[7] J. Lokoč, T. Grošup, P. Čech, and T. Skopal. Towards efficient multimedia exploration using the metric space approach. In *Content-Based Multimedia Indexing (CBMI), 2014 12th International Workshop on*, pages 1–4, June 2014.

[8] J. Moško, T. Skopal, T. Bartoš, and J. Lokoč. Real-time exploration of multimedia collections. In H. Wang and M. Sharaf, editors, *Databases Theory and Applications*, volume 8506 of *Lecture Notes in Computer Science*, pages 198–205. Springer International Publishing, 2014.

[9] G. Schaefer. Interactive browsing of image repositories. In *Proceedings of the 2012 International Conference on Computer Vision and Graphics*, ICCVG'12, pages 236–244, Berlin, Heidelberg, 2012. Springer-Verlag.

[10] K. Schoeffmann, D. Ahlstrom, and C. Beecks. 3D Image Browsing on Mobile Devices. In *ISM '11*, pages 335–336, Washington, DC, USA, 2011. IEEE Computer Society.