Integration of Exploration and Search: A Case Study of the M^3 Model

Snorri Gíslason[1], Björn Þór Jónsson[1], and Laurent Amsaleg[2]

[1] IT University of Copenhagen, Copenhagen, Denmark
[2] CNRS-IRISA, Rennes, France
bjorn@itu.dk

Abstract. Effective support for multimedia analytics applications requires exploration and search to be integrated seamlessly into a single interaction model. Media metadata can be seen as defining a multidimensional media space, casting multimedia analytics tasks as exploration, manipulation and augmentation of that space. We present an initial case study of integrating exploration and search within this multidimensional media space. We extend the M^3 model, initially proposed as a pure exploration tool, and show that it can be elegantly extended to allow searching within an exploration context and exploring within a search context. We then evaluate the suitability of relational database management systems, as representatives of today's data management technologies, for implementing the extended M^3 model. Based on our results, we finally propose some research directions for scalability of multimedia analytics.

Keywords: Multimedia Analytics; Exploration-search axis; Scalability.

1 Introduction

Multimedia analytics is a research field that grew from a desire to harness the information and insight that are embedded in today's media collections, which are growing both in scale and diversity. In multimedia analytics, supporting user interaction with media collections is particularly important, both in its own right and as a precursor to applying data mining methods. This user interaction can involve a number of distinct tasks, and Zahálka and Worring [14] defined an exploration-search axis with a range of tasks that multimedia analytics tools need to support in a single interface. Here, we focus on the two extremes of that axis: exploration and search.

Typically, exploration and search are implemented as two different operations that are grounded in disjoint informational contexts. On the Web, for example, exploration consists of clicking on links to jump from one document to the other, whereas searching consists of obtaining ranked lists of relevant documents. Once the user starts clicking on search results, the search context is lost and the only way to revisit that context is to go back to the original search and adjust the query. Then the search starts again from scratch and the only way to observe the previous exploration context is via the color of hyperlinks. Similarly, in current file and media browsers search is implemented as a distinct operation that loses all context of the previous exploration session. To support the exploration-search axis, however, the two user interaction modes must be performed in the same context: we should be able to focus the exploration within search results, or search within the current exploration state.

Multidimensional Media Space. Today, media items are typically associated with a plethora of descriptive and administrative metadata. First, media is commonly generated with technical data about its creation, such as date and time, location, user, and technical specifications. Second, a multitude of methods have been developed to describe the media contents, for example based on deep learning. Third, as users see more and more benefits of annotating media, they likely become more willing to do so.

All this metadata can be seen as defining a multidimensional media space. Many multimedia analytics tasks then boil down to exploring, manipulating and augmenting that media space. Exploration can be seen as applying and updating a set of filters and predicates that outline the current set of multimedia items that a user is interested in. Search can be seen as a reorganization of the space from the reference point of the query.

Contributions. We define *browsing state* as the set of filters and reference points that the user is exploring. The browsing state is an abstract representation of the informational context of the currently displayed media items. Exploration and search tasks gradually update that browsing state, allowing users to alternate tasks while preserving the informational context. We believe that multimedia analytics suites can succeed in seamlessly integrating exploration and search tasks (as well as other tasks along the exploration-search continuum of [14]) if they implement something equivalent to a browsing state.

In this paper, we demonstrate such integration within the context of the Multidimensional Multimedia Model (M^3, pronounced emm-cube) [9]. M^3 was proposed as a way to interactively explore the multidimensional media space, merging concepts from business intelligence (online analytical processing (OLAP) and multidimensional analysis (MDA)) and faceted browsing. The M^3 model, however, was defined as a pure exploration interface, with no support for search. This paper is an initial exploration into the integration of search into the M^3 exploration model, showing that the M^3 model can be elegantly extended to support both extremes of the exploration-search axis.

This paper also highlights the difficulties that the underlying data-retrieval infrastructure runs into when trying to provide an *efficient* implementation of the browsing state and its maintenance. In short, the current state of the technology is unable to dynamically support the unpredictable user-defined sub-collections of media items involved in exploration and search tasks.

The remainder of this paper is organized as follows. We review background work in Section 2, and then summarize the M^3 model in Section 3. We then make the following contributions, before concluding the paper in Section 7:

– We extend the M^3 model to include search results as dynamic dimensions of the multidimensional media space (Section 4).
– Using a proof-of-concept implementation, we then show that relational systems are not suitable for the extended M^3 model (Section 5).
– Based on our experience, we present some research directions towards efficient exploitation of the multidimensional media space (Section 6).

2 Background

Zahálka and Worring [14] surveyed a collection of more than 800 research papers related to user interaction with multimedia collections and compiled into a model of user interaction for multimedia analytics. A key contribution of that work was the exploration-search axis, which consisted of a continuum of tasks that multimedia analytics users must be able to accomplish. In this paper we consider the two extremes of that axis. Here, we review key results related to multimedia search and exploration; this coverage is brief for space reasons.

Multimedia retrieval, in particular high-dimensional feature indexing, has received significant attention in the literature, including several highly scalable methods (e.g., [2, 5, 6, 8]). None of these methods, however, offer any support for integration of search into a dynamic browsing state.

Multimedia exploration tools have typically considered various modes of interacting with static media collections (e.g., [10, 11]). None of these tools consider the integration of dynamic search with exploration. Faceted media browsers create hierarchies (or DAGs) of tags and allow interactively traversing those structures, narrowing down the set of displayed items to match the user needs. Typically, faceted browsers present results in a linear list, thus losing the internal structure of the browsing set [4, 3, 12]. OLAP applications, on the other hand, have long been used to efficiently browse multidimensional numerical data, with support for slicing, drilling in, rolling out, and pivoting. Early applications of the OLAP model to multimedia include [1, 7, 15]. Neither the original OLAP systems, nor the referenced multimedia variants, consider search. Their efficiency is due to pre-computed indexes; including search in their interaction model would invalidate all their pre-computations, as it is impossible to consider all potential query reference points.

Zahálka and Worring also proposed interactive multimodal learning (IML) as an umbrella interaction model for multimedia analytics [14]. More recently, they and others proposed a very efficient system for IML over large-scale collections [13]. IML can be seen as a reorganization of the media space, just as search, and we plan to integrate IML with exploration and search in future work.

3 The M^3 model

The M^3 model was proposed in 2015 by Jónsson et al. [9], as an interaction model for exploration of personal photo collections. The foundation of M^3 is to consider media metadata as defining a multidimensional space that organizes the media items, and to use concepts from faceted browsing and OLAP to explore that space; much of the terminology for user interactions is indeed borrowed from OLAP. The basic data in the M^3 model consists of *objects* and *tags*, which refer to the media items and their descriptive and administrative metadata, respectively. Originally, tags were defined as simple data items, such as alphanumerical strings, dates and timestamps, but tags might also be more complex, such as high-dimensional feature vectors.

The multidimensional aspects of the M^3 model then arise from the ways the tags are organized among themselves. A *concept* in M^3 groups related tags together into sets of tags, which may have an implicit ordering (e.g., for dates and numerical tags). A *hierarchy* then adds an explicit tree structure to (a subset of) the tags in single concept; hierarchies only contain tags from that concept. Together, the concepts and their hierarchies form the dimensions of a *hypercube*; the objects are conceptually present in the *cells* of this hypercube (or a subcube of it) if they are associated with each tag corresponding to the cell.

During exploration, the user uses *filters* over some of the dimensions to define a subcube of the complete hypercube; this subcube is the *browsing state* of M^3. The filters may focus on a specific tag (tag filter), on a range of tags from a concept (range filter), or a subtree of a hierarchy (hierarchy filter). Applying a tag filter or range filter to a concept is called *slicing*, as each filter represents a slice of the entire hypercube; if a filter already exists on that concept it is replaced by the new filter. Traversing up or down a hierarchy is also tantamount to updating a corresponding hierarchy filter; called *rolling up* or *drilling down*, respectively. Note that each of these operations updates the browsing state.

Jónsson et al. [9] also proposed a user interface for the M^3 model. The user interface consists of three axes (called *front*-axis, *up*-axis and *in*-axis for intuitiveness), and the user may assign any dimension from the browsing state with 1, 2 or 3 of these axes, resulting in projection of the browsing state onto a 1D, 2D or 3D representation, respectively. Replacing one visible dimension with another dimension is called *pivoting*; note that if the new visible dimension was already part of the browsing state, then pivoting does not change the browsing state. The following example demonstrates the common operations in the M^3 model.

Example 1 *A mother is sitting down with her children in front of her computer to recall a hiking trip. She first selects the People dimension (a hierarchy over a sub-set of the people concept) as a starting point on the front-axis, which has two tags at the top level: "Adults" and "Kids". Then she selects the Location dimension, which has such nodes as "Cabin" and "River", as the up-axis. Being a photo nerd, she becomes interested in the light conditions and assigns the Aperture value to the in-axis. The current browsing state then has three dimensions, where each cell has (at least) one particular person in one particular location type with one particular aperture value. Note that photos containing kids and adults will show up in two cells (and, if a cabin were situated next to a river, it could show up in four cells) as the photos belong logically in all these cells.*

4 Integrating Exploration and Search

We have identified the following requirements for integrating search within the M^3 model:

Metadata-Based Search: In the M^3 model, the browsing state is based on media metadata, as represented by filters over concepts and hierarchies. The media items themselves are never considered directly in the browsing state, but are represented only through their metadata. Consequently, search operations should also focus on metadata. In order to consider content-based search, the content description must therefore first be extracted into a (new or existing) metadata concept.

Dynamic Result-Dimensions: User interaction in the M^3 model consists of maintaining and projecting the browsing state, which in turn is composed of dimensions. To integrate search into the browsing state, the results of each search operation must therefore define a new dimension in the browsing state. If the query is modified, then the previous result dimension must be replaced by the new result dimension, and the browsing state updated correspondingly.

Single-Concept Search: As described above, a browsing state is composed of dimensions, which are either concepts or hierarchies; hierarchies, in turn, are simply a (relatively) static representation of a concept. There is thus a direct correspondence between browsing state dimensions and concepts. To maintain that correspondence, each search operation should only apply to one tag concept.

Generality: Depending on the metadata type, different search operations may apply, e.g., text search for alphanumeric tags and similarity search for feature vectors. And depending on the search type, different indexes may be required in the media server. In all cases, however, search results should be ordered based on score (e.g., relevance, similarity, or distance). Set-based search can be implemented by assigning the same score to each result. Some search methods only assign scores to a subset of the objects, while others assign a score to each object; range filters can be used to reduce the number of objects returned.

To better illustrate how following these requirements leads to integration of exploration and search, consider the following example.

Example 2 *Recall the final browsing state of Example 1, which had three dimensions: People on the front-axis; Location on the up-axis; and Aperture on the in-axis. The mother now decides she wants to focus on images with colors similar to a particular image. She opens up a search form, where she selects the image, chooses a distance filter to focus on similar images, and assigns this search concept to the in-axis. Note that the tags of this new temporary concept are the color distance values from the search. Also note that by assigning the search dimension to the in-axis she replaces the Aperture concept, but nevertheless only images with an aperture tag are included.*

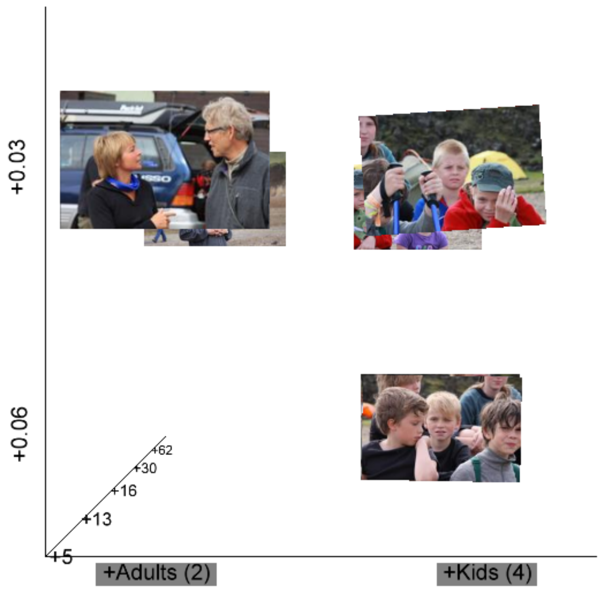

Fig. 1. 3D representation of the browsing state from the scenario in Example 2. See the text for a detailed description of the browsing state axes.

The mother now recalls a name from the trip and wants to focus on images tagged with that name. She opens up a second search form for the People dimension, types in the name "Mick Junior" and assigns the new search concept to the up-axis. The tags of this search concept are the similarity scores from the search. By assigning the search dimension to the up-axis she now replaces the Location hierarchy, but as before only images with a location tag are included.

Figure 1 shows the browsing state resulting from Example 2, which has one hierarchical dimension and two search dimensions. A few notes are in order.

- The scores on the up-axis indicate the relevance of image tags to the query, as computed by PostgreSQL (the relevance values are $+0.03$ and $+0.06$, presumably using some form of TFIDF scoring). Three participants in the family hike were named Mick (two kids, one adult), but only one was called Mick Junior. Images with Mick Junior receive higher relevance scores than images with one of the other Micks.
- The hierarchy on the front-axis divides the browsing state based on whether the Micks shown in images are adults or kids. In this case the images in the lower right corner contain Mick Junior, images in the upper left corner contain Mick Senior, and images in the upper right corner contain the third (young) Mick. The collection does contain some images with two and three Micks, but these were filtered by the color similarity query.

– The distances on the in-axis reflect the color similarity, computed using squared Euclidean distance. Note that in this case a filter of 75 was applied, and hence images with distance higher than 75 are not reflected in the browsing state.
– The example image from the first (text) search has Euclidean distance of 0 from itself. That image did not have a Mick in it, however, and hence is not included in the set of images retrieved by the browsing state. Therefore there is no image with distance 0 in the browsing state. The most similar image has nearly identical average color, however, with a squared Euclidean distance of only 5.

In the preceding example, the interactions of the user were represented by a sequence of browsing states, each dependent on the previous browsing state. We believe that any efficient implementation of the extended M^3 model must take into account this incremental nature of the browsing state and interactions with users, but we are not aware of any existing algorithms or indexing strategies that do so. In Section 5 we describe and evaluate a proof-of-concept implementation of the extended M^3 model, using relational database technology as a representative of the current state of the art, and show why performance suffers when the previous browsing state is ignored. In Section 6 we then propose some research directions towards scalable implementation of the extended M^3 model.

5 Prototype Evaluation

The M^3 model was initially implemented by Jónsson et al. [9] as a server to deliver the objects in a browsing state (O^3) and a photo browsing client (P^3). We extended this prototype by integrating search functionality, as described below. As the O^3 server was implemented on top of a relational database management system (RDBMS), we decided to evaluate whether an RDBMS is a suitable technology for implementing this integration.

5.1 Proof-of-Concept Implementation

Following the requirements of Section 4, integrating search results into the browsing state of the M^3 model can conceptually be done in the following steps:

1. Create a temporary concept for the search results and add to browsing state:
   – Retrieve the relevant objects and their relevance score, applying a range filter if required.
   – Create new metadata tags for the relevance score and assign the appropriate objects to each score tag.
   – Update the browsing state description to include the search concept.
2. Retrieve the browsing state.

Since relational systems provide no efficient support for integrating knowledge of the previous browsing state into the first step, the query to retrieve the relevant objects is completely independent of the existing browsing state and only affected by the criteria applied to that particular search dimension, which leads to sub-optimal performance.

We decided to focus on two types of search that are well supported by many relational systems:

- Keyword search over alphanumeric tags, supported by an inverted index.
- Similarity search over low-dimensional features, supported by an R-tree.

For the latter, we use a very simple similarity-based search on color, using the average RGB color in each image (computed by averaging the R, G, and B values across all pixels). As feature vectors did not exist in O^3 as a tag type, we added a table to store these features, and used an R-tree to index the three-dimensional feature vectors.

5.2 Evaluation

In this section we evaluate the performance of our prototype for three extremely simple browsing states. Each of these three browsing states correspond to a user selecting search as the first (and only) operation to apply to the collection. The goal of these experiments is to gauge the potential performance of relational systems, to establish baseline performance numbers and to identify performance bottlenecks, with an aim towards inspiring and supporting subsequent research into indexing and query processing.

Experimental Collections. In the following we describe three experiments: keyword search with long text annotations; keyword search with short (name) tags; and color similarity search. For each experiment, a new collection with a single tag concept was created, allowing detailed control of the tag concept properties. We now describe the tag collections and concepts created for each experiment.

Text Search: In this experiment, we created collections with 1K, 10, 100K, and 400K objects, and used the Amazon review data set[1] to create a concept with one review tag associated with each object. Reviews exceeding 512 characters were truncated; as some reviews are shorter, the average tag length varied from 434 character for the smallest collection to 458 characters for the largest collection.

Tag Search: In this experiment, we created collections with 1K, 10K, 100K, and 1M objects. We then created a collection of 200 randomly chosen surnames and associated each object with three surname tags. The surname tags were chosen by a) assigning selected tags to 1, 10, 100 and 1,000 random objects, to facilitate the controlled experiment, and b) randomly assigning the remaining tags to give three tags per object.

Color Search: In this experiment, we again created collections with 1K, 10K, 100K, and 1M objects. We then created a random RGB tag for each image and inserted into the RGB concept.

[1] Available at `https://www.kaggle.com/bittlingmayer/amazonreviews`.

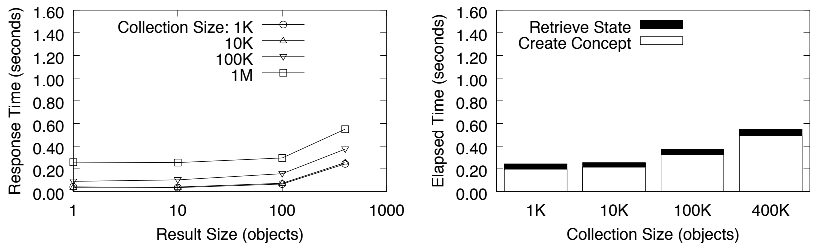

Fig. 2. Text Search: Performance of browsing state retrieval (left); Time breakdown for retrieval of 1,000 objects (right).

Experimental Method. In each experiment, we consider retrieval of browsing states with 1, 10, 100 and 1,000 objects, and study how the performance of browsing state retrieval varies depending on both browsing state size and object collection size. In all cases, the experiments focus on the retrieval of the browsing state information and exclude the retrieval of the objects (images) themselves.

The experiments were run on a DELL Latitude E7440 laptop, with an Intel dual core i7-4600U 2.10GHz processor, 4MB CPU cache, 16 GB of RAM and a 256GB solid state drive. The laptop runs Windows 8.1, but the experiment was run on a Linux Ubuntu 16.10 64-bit virtual machine with allocated base memory of 4 GB and 1 processor, running on Oracle VM VirtualBox Manager. In each experiment, we started with the smallest object collection and continued to the largest object collection. We repeated this process five times and report the average times from these five runs. Before each such run, the virtual machine was shut down and the laptop restarted.

Text Search. Figure 2(left) shows the performance of browsing state retrieval for long text tags, as the browsing state size varies from 1 object to 1,000 objects, and as the collection size grows from 1K to 400K. The figure shows that the time increases both with collection size and result size, but in all cases remains under 0.6 seconds, which is sufficient for interactive workloads.

Figure 2(right) shows a more detailed analysis when the browsing state contains 1,000 objects, breaking the response time into a) the creation of a temporary concept with the search results, and b) the retrieval of the resulting browsing state. As the figure shows, the majority of the time is spent on the former. The increased time, as the collection grows, is due to the increased size of the inverted index; for even larger collections, the response time is likely to increase linearly.

Tag Search. Figure 3(left) shows the performance of browsing state retrieval for short name tags, as the browsing state size varies from 1 object to 1,000 objects, and as the collection size grows from 1K to 1M. As with the longer text tags, time increases both with collection size and result size, but remains interactive with less then 1 second to retrieve 1,000 objects from the 1M collection.

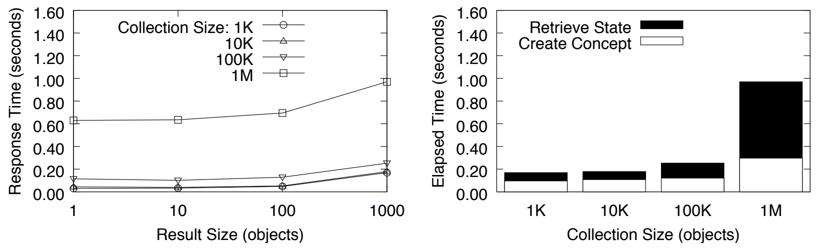

Fig. 3. Tag Search: Performance of browsing state retrieval (left); Time breakdown for retrieval of 1,000 objects (right).

Somewhat unexpectedly, however, retrieval from the 1M collection is significantly more expensive than before. Figure 3(right) shows the more detailed analysis when the browsing state contains 1,000 objects. As the figure shows, the creation of the new temporary concept is less expensive than with the larger text tags, due to the smaller inverted index. On the other hand, the creation of the resulting browsing state is significantly more expensive, for two reasons. First, the collection is larger (1M compared to 400K). Second, since each object is associated with more tags (3 name tags compared to 1 text tag), the number of tag-object associations is actually 7.5x larger for the short name tags, resulting in a more expensive query to assemble the browsing state.

Color Search. Figure 4(left) shows the performance of browsing state retrieval for RGB color tags, as the browsing state size varies from 1 object to 1,000 objects, and as the collection size grows from 1K to 1M. For the most part, time increases both with collection size and result size. The retrieval time is no longer interactive, however, when returning 1,000 objects from the 1M collection.

Figure 4(right) shows the detailed analysis when the browsing state contains 1,000 objects. As the figure shows, the creation of the new temporary concept is responsible for the majority of the response time, due to the inefficiencies of the R-tree index. Interestingly, the time to retrieve the browsing state shrinks as the collection grows. The reason for this is that as the collection grows, more and more objects share each distance value and hence the browsing state has fewer and fewer distinct distance tags, resulting in reduced computation time.

5.3 Summary

In summary, the performance of the relational server suffers as the size of both the collection and browsing state grow. The key reason is that the RDBMS offers no support for using the previous browsing state to facilitate the search, in some cases leading to high cost of computing the search, and in other cases to high cost of retrieving the browsing state.

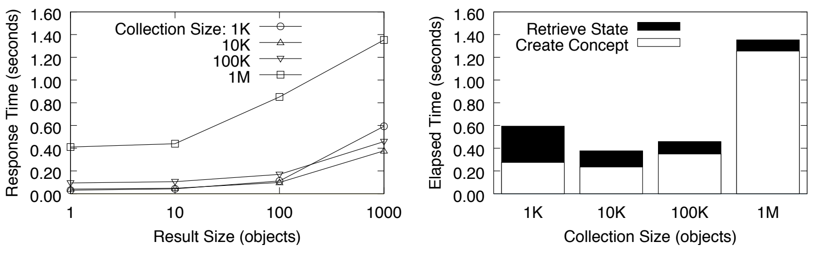

Fig. 4. Color Search: Performance of browsing state retrieval (left); Time breakdown for retrieval of 1,000 objects (right).

6 Discussion

In this section we highlight the major lessons we have learned from this case study about integrating exploration and search for multimedia analytics.

- At the conceptual level, we have presented an elegant way to integrate exploration and search for multimedia analytics. As Example 2 shows, our model allows searching within an exploration context and exploring within a search context. This integration warrants further exploration, however, for example with respect to the meaning of k-NN search and approximate search within exploration contexts, as well as other modes of interaction, such as interactive multimodal learning. We believe that this field is ripe for investigation.
- The performance results show that relational database systems are not a suitable tool for multimedia exploration, as they do not support the multidimensional nature of the application well and fail to provide interactive performance, even with a relatively small collection of 1M objects. Note that some systems integrate efficient support for traditional OLAP applications, which might appear more appropriate for multimedia analytics. This support is predicated on the static nature of such applications, however, and does not address dynamic searches within an exploration context.
- In the prototype, the creation of a search dimension is done without any consideration of the browsing state; the browsing state is then updated using the new search dimension. For better integration, however, the index structures and algorithms supporting search must be aware of the multidimensional nature of the data. Currently, no such index structure or algorithms exist and developing those algorithms is another field that is ripe for investigation.

This paper highlights the difficulties that the underlying data-retrieval infrastructure runs into when trying to provide an *efficient* implementation of the browsing state and its maintenance. In short, the current state of the technology is unable to dynamically support the unpredictable user-defined subcollections of media items involved in exploration and search tasks. We believe that discovering the algorithms and index structures required to provide that support is a major research direction within the field of multimedia analytics.

7 Conclusion

Effective support for multimedia analytics applications requires exploration and search to be integrated seamlessly into a single interaction model. In this paper, have presented an initial case study of using the multidimensional media space of media metadata to integrate exploration and search. We have extended the M^3 model, initially proposed as a pure exploration tool for the multidimensional media space, and shown that it can be elegantly extended to allow searching within an exploration context and exploring within a search context. We have then presented and evaluated a proof-of-concept prototype and derived some major research directions for multimedia analytics.

References

1. Arigon, A.M., Miquel, M., Tchounikine, A.: Multimedia data warehouses: A multiversion model and a medical application. Multi. Tools and Apps. 35(1) (2007)
2. Babenko, A., Lempitsky, V.S.: The inverted multi-index. IEEE Transactions on Pattern Analysis and Machine Intelligence 37(6) (2015)
3. Bartolini, I., Ciaccia, P.: Integrating semantic and visual facets for browsing digital photo collections. In: Proc. SEBD (2009)
4. Diao, M., Mukherjea, S., Rajput, N., Srivastava, K.: Faceted search and browsing of audio content on spoken web. In: Proc. CIKM (2010)
5. Guðmundsson, G.Þ., Amsaleg, L., Jónsson, B.Þ., Franklin, M.J.: Towards engineering a web-scale multimedia service: A case study using Spark. In: Proc. MMSys. Taipei, Taiwan (2017)
6. Jégou, H., Tavenard, R., Douze, M., Amsaleg, L.: Searching in one billion vectors: Re-rank with source coding. In: Proc. ICASSP. Prague, Czech Republic (2011)
7. Jin, X., Han, J., Cao, L., Luo, J., Ding, B., Lin, C.X.: Visual cube and on-line analytical processing of images. In: Proc. CIKM (2010)
8. Jónsson, B.Þ., Amsaleg, L., Lejsek, H.: Scalability of the nv-tree: Three experiments. In: Proc. SISAP. Lima, Peru (2018)
9. Jónsson, B.Þ., Tómasson, G., Sigurþórsson, H., Eiríksdóttir, Á., Amsaleg, L., Lárusdóttir, M.K.: A multi-dimensional data model for personal photo browsing. In: Proc. MMM. Sydney, Australia (2015)
10. Shneiderman, B., Bederson, B.B., Drucker, S.M.: Find that photo! interface strategies to annotate, browse, and share. Communications of the ACM 49(4) (2006)
11. Worring, M., Koelma, D.C.: Insight in image collections by multimedia pivot tables. In: Proc. ACM ICMR. Shanghai, China (2015)
12. Yee, K.P., Swearingen, K., Li, K., Hearst, M.: Faceted metadata for image search and browsing. In: Proc. CHI (2003)
13. Zahálka, J., Rudinac, S., Jónsson, B.Þ., Koelma, D.C., Worring, M.: Blackthorn: Large-scale interactive multimodal learning. IEEE Trans. on Multi. 20(3) (2018)
14. Zahálka, J., Worring, M.: Towards interactive, intelligent, and integrated multimedia analytics. In: Proc. IEEE VAST. Paris, France (2014)
15. Zaïane, O.R., Han, J., Li, Z.N., Hou, J.: Mining multimedia data. In: Proc. CASCON (1998)