MediaFrame: Reclaiming the Shoebox

Steven M. Drucker, Curtis Wong, Asta Roseway, Steven Glenner, Steven De Mar
Microsoft Research
1 Microsoft Way, Redmond, WA. 98052
{sdrucker, wong, astar, stevegl, sdemar}@microsoft.com

ABSTRACT
Applying personal keywords to images and video clips makes it possible to organize and retrieve them, and automatically create thematically related slideshows. MediaFrame is a system designed to help users create annotations by uniting a careful choice of interface elements, an elegant and pleasing design, smooth motion and animation, and a few simple tools that are predictable and consistent. The result is a friendly, useable tool for turning shoeboxes of old photos into labeled collections that can be easily browsed, shared, and enjoyed.

Author Keywords:
Digital photography, organization, annotation, visualization

ACM Classification Keywords
H5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous.

INTRODUCTION
In 1900, Kodak created the field of consumer photography when they introduced the Brownie camera. The camera sold for $1, and a roll of film was 15 cents [1]. Over a century has passed, and though many photos are preserved in albums and journals, untold millions of photographic prints are languishing, unsorted, in bags, trunks, and overstuffed shoeboxes. Many people are preserving their photos by digitizing them. Add to this the new images coming directly from digital cameras, and movie files coming from camcorders, and an enormous and growing number of digital images and film clips are populating the disks of personal computers all over the world.

Organizing all this media is an important problem. If we can organize them robustly, then we can not only find specific images efficiently, but we can easily share our digital media with friends and family. One easy way of sharing is through the creation of video playlists. These are like playlists for MP3 files, but for images and film clips. We can build these playlists by annotating the media file with personally meaningful text labels (also called key words).

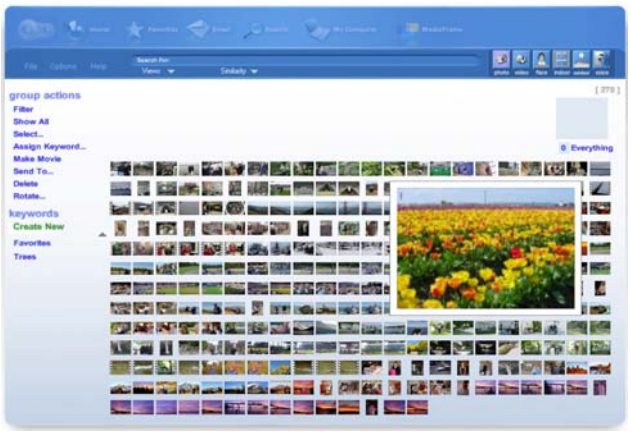

Figure 1a: MediaFrame immediately after starting up with a fresh set of media objects. The user has the cursor hovering over the picture of flowers so it is enlarged.

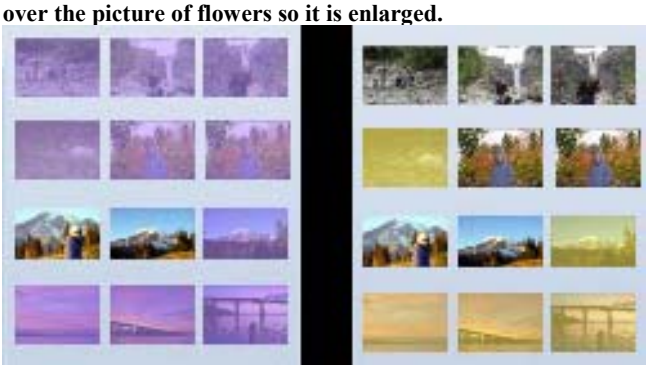

Figure 1b (close up): Objects that have the keyword "outdoors" are tinted purple. Figure 1c (close up): Objects that are "outdoors" but have no "faces" are tinted yellow to show that they are selected.

The novel contributions of our system are the blending of different UI techniques and image analysis technologies into a conceptually unified design that lets us select, filter, and name large numbers of photo and video files with ease. Dynamic transitions are used throughout the system since they can help users maintain context when using the system [12]. Our goal in this project was to create a single application that brings together a variety of recent UI advances into a single consistent, visually attractive, and easy to use application for applying labels to image and video files meaningful to the user. Such an application must be sufficiently pleasant to use that it lowers the bar on the daunting task of organizing hundreds or even thousands of media objects.

PRIOR WORK

Until recently, much of the work for organizing, browsing and retrieving digital photographs was devoted to large collections within the business or professional domain. However, with the advent of widespread digital photography (and the huge growth in the number of digital photographs), more image database products have been made available. Some of the most popular are ACDSee [1], Picassa [10], iPhoto [7], and Adobe Photoshop Album [2].

Most of the applications allow retrieval of media objects via one of two ways: keyword based image retrieval or retrieval based on similar images. Some databases focus on the retrieval of images using keywords and categories that were assigned to the image objects by professionals. Subsequent retrieval can proceed by successively narrowing the search using "facets" or properties for media objects. Similarly, our work allows for convenient, successive specifications of properties for rapid retrieval of media objects from the database based on intrinsic metadata (media type, date taken), analyzed metadata (presence of faces, indoor or outdoor content), or extrinsic, explicitly assigned, metadata (keywords).

Several research systems, notably Kuchinsky et al [9], Schneiderman et al [13], Bederson [3] and Rodden & Wood [14] have all looked at using different ways to retrieve pictures based both on applied metadata and other retrieval mechanisms. The MediaFrame application was designed to integrate the best mechanisms to make it easy to rapidly annotate groups of pictures for subsequent rapid retrieval.

DESIGN GOALS

Our goal was to create an application that made it easy and pleasant to assign personally meaningful key words, or labels, to a collection of up to about 600 photographs and video clips. The difficulties of organizing disparate elements into a folder hierarchy are well known, and many commercial products such as ACDSee [1] and Adobe Photoshop Album [2] also support keywords to avoid the problem of strict hierarchical filing. Users can assign any number of keywords to their media, and can then use those labels to create collections and organize which elements are chosen for display.

Clarity, simplicity, and ease of use were our design criteria. We felt that a consistent, predictable, and clean interface was more important than providing a wealth of options. We wanted to create a system that ran comfortably on a contemporary mid-range laptop. We also felt strongly that the application had to have simple but beautiful visual aesthetics that went beyond attractive graphic design. Our interface has different layout schemes, but the presentation doesn't jump from one type of layout to another. Rather, all of our visual elements move smoothly and gracefully in transition from one layout to another to maintain context at all times. Our choices of algorithms and overall architecture were influenced by what we felt we could animate in real-time in ways that would be both informative and pleasing to watch.

To that end, we considered dozens of interface techniques. Here are our principal design choices.

- **Visual clusters**: We allow the user to easily create *bins*, which contain arbitrary collections of objects. Clusters are used elsewhere within the interface to assist in browsing and keyword assigning.
- **Multiple visualizations**: We support three different types of visualization for our objects: the Gallery view, the Time Cluster view, and the Scrollable Grid view. All of our selection, keyword assigning, filtering, and preview tools work the same way in all views. Transitions between views are smooth and continuous which allows users to keep track of the context under which they're operating.
- **Preview**: Users will frequently select objects using different criteria. We provide immediate feedback on the objects that would be selected as the result of any action before that action is confirmed, by tinting affected objects purple.
- **Selection identification**: Objects that are currently selected are tinted yellow.
- **Consistency of object representation**: We wanted images and movies to be easily recognized, yet not jarringly different. Each of these media objects are represented by a thumbnail image. Movie clips are shown by an image surrounded by film sprockets.
- **Undo**: A "back" button lets the user undo any operation.
- **Incentive to begin**: People will be more likely to use the application if there is already some existing metadata for the images. When we first import a set of images and film clips, we analyze them and automatically assign several different keywords.
- **Fuzzy selections**: Much of the time, users will explore different subsets of their images and film clips in order to find just the ones they're after. We offer parameters on most types of selections and provide real-time feedback on what they would select.
- **Smooth motion**: Objects always move, appear, and disappear smoothly which helps the user maintain context.
- **Visual booleans**: Users can create OR and AND Boolean filters implicitly using selection, inverse-selection, and deletion tools. By providing immediate feedback and undo capability, we allow sophisticated Boolean queries to be built up by users not familiar with database query languages.

Fixed Grid View

Figure 1a shows our application in the fixed grid view immediately after loading up a folder of previously unlabeled images. The user has moved the cursor over the

thumbnail of the flowers and the image has dynamically expanded.

The central area of the screen holds about 1000 thumbnails from either still images or video clips. More thumbnails can be displayed, but at this point, the size of the thumbnail becomes too small and too numerous to be useful for rapid browsing and searching. In the upper-right are buttons that represent each of our automatically-assigned metadata. These are created by plug-ins that we can easily add and revise over time. Each automatic module looks at a media object as it's loaded in, and gives it one or more appropriate keywords or calculates metadata that can be used for later comparisons (for example, the similarity detector computes feature analysis and color histograms). Some examples include an indoor/outdoor detector and a face detector (this simply notes whether there is a face in the image, and doesn't try to guess the identity of the person).

In the lower left is a scrolling list of keywords. This list always contains the union of all the keywords assigned to all the media in the main window. The list in the figure therefore is simply the collection of terms that were assigned by the automatic modules.

Labels are saved with the media when they're applied. If a user wanted to do nothing more than apply the automatic labels, then he or she could exit the application now.

But of course most users will want to apply at least some of their own labels to their media, such as descriptions (e.g. "birthday party" or "picnic"), names (e.g. "Grandparents"), or locations (e.g. "Maui").

Suppose that you want to add the label "trees" to pictures of trees taken on a recent hike. You'll first select the images that were taken on the hike, narrow that down just to the trees, and then apply the label.

To select the hiking images, you would probably first move your mouse to the upper-right part of the screen over the "outdoors" icon. As soon as you're hovering over the icon, we preview the selection by temporarily tinting those objects purple that the system thinks were taken outdoors as shown in Figure 1b. Clicking on the outdoors icon causes the purple objects to become selected. To show their new state, the purple tint becomes a yellow one.

Of course, only some of these outdoors pictures are pictures of trees. Suppose that looking through your pictures, you see that one way to narrow down the selection would be to remove those images that have lots of faces in them. One way to do this is to first invert your current selection, so that now the selected objects are all those that weren't labeled "outdoors" (note that this would include not just the ones labeled "indoors", but also those objects that the indoor/outdoor algorithm wasn't sure about and gave no label to). In Boolean terms, the selection is now "not-outdoors". Now you'd extend the selection to include faces. As in many Windows applications, holding down the Control key while selecting inverts the state of the selected objects, and holding down Shift adds (or subtracts) them from the current selection.

So to add in all the pictures with faces in them, you'd hold down the shift key and click on the "faces" icon. Inverting the selection again, you've now got those images that are outdoors, but don't have faces. You could then clean up the result by control-clicking on individual objects, selecting or deselecting them as appropriate. The process feels natural in process, in large part thanks to the real-time previewing of selections, which lets users casually hunt for the best group of objects that satisfy some criteria they have in mind.

Now you would simply click on the Add New Keyword button and type in “tree” in the newly created field. The tag is immediately added to the other tags for the selected objects, and the keyword "tree" is added to the list at the left. Figure 1c shows the result.

Selecting and Filtering

The typical workflow of our application is to first select some objects, refine the selection, and then apply a label to the selection. So we provide a variety of tools for selecting and refining.

Let's continue our previous example. Suppose you wanted to make another choice that refined your set of trees, giving just some of them the label "evergreen trees." To remove clutter you would probably want to get rid of the non-tree objects. When you press the "Filter" button at the left, the objects that are not selected are removed from the display. Double-clicking on the “Trees” label selects and performs a filter in one step, so everything that is not labeled “trees” is removed.

Filtering is not deleting. Rather, filtered objects are simply removed from the display, but they remain part of the collection. When you filter objects away, they fade out, which can leave a collection that looks messy, as in Figure 2a. The system automatically enlarges the objects and smoothly moves them into a neat grid, resulting in Figure 2b. At any time, you can press "Show All" to restore display of the objects in the collection.

As noted before, the keywords that are shown are only the keywords that are present in the currently display subset of the collection. This makes successive refinement extremely convenient and clear. In addition, those keywords that are present on every thumbnail are displayed in a dimmer color and are not clickable since they do not refine the collection further.

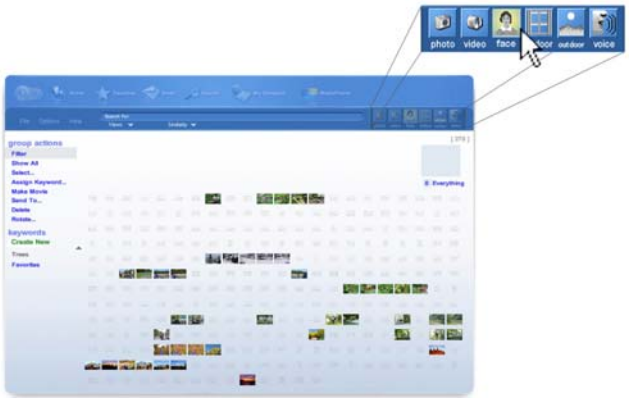

Figure 2a: Nearly at the end of the transition when non-tree objects have been filtered away, the screen is messy.

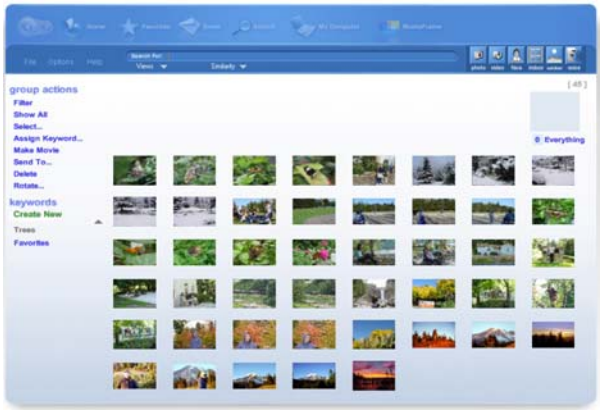

Figure 2b: After smooth and automatic rearrangement.

In addition to clicking on both user-created and automatic keywords, and clicking individually on objects, we provide several other selection techniques, all of which can be used to grow or shrink selections:

- **Sweep**: Sweep out a rectangle with the mouse to select all objects inside the box.
- **Similarity**: Selects other objects that are similar to those chosen. For still images, we use color histograms and some simple feature analysis to find close matches.
- **Key text**: In addition to complete keywords, users can also type in short text, which match anywhere in the word. For example, the text "ons" would match the keywords "onset," "monster," and "spoons".
- **Time**: Objects can be selected according to the timestamp assigned by the camera when they were shot. We provide sliders for choosing the center of the time range, and the size of the symmetrical range around that center.

Time Cluster Views

The Fixed Grid View organizes objects into a grid, which looks pleasing but doesn't carry any information about the objects themselves.

The Time Cluster view, shown in Figure 4, organizes objects into groups according to when they created (either time-stamped by the camera or camcorder, or by file-creation time on the computer). Beneath each cluster is a label that shows the time range spanned by that cluster. Clicking on a label selects all of the objects in its cluster. You can roll your mouse over any object, and it will show a large image or play the video, as always

Figure 4: The Time Cluster view. Objects are grouped together by their proximity to one another in time.

The user controls this view by specifying the number of clusters to be used. The system looks at all the media objects, and finds the largest gap between any two objects. This becomes the first splitting point, creating two clusters. If the user wants three clusters, the next largest gap is found, and the appropriate cluster is split at that point. The process repeats, always finding the biggest remaining gap between objects and splitting the appropriate cluster at that point, until the desired number of clusters have been created. We've also tried incorporating some of the same algorithms for clustering used in Platt [11] to help manage clusters and to help choose a representative image for a cluster.

If the user wants to get a closer look at objects in just some smaller range of time, he or she would select the desired clusters (or pieces of clusters) and then press "filter" to remove the unselected objects from the display (or, as in the case for all filtering, the user can double click and the default operation is to filter). The remaining objects then divide themselves up and smoothly animate into the same number of clusters, only now of course on a finer scale.

Scrollable Grid View

Another viewing mode is the Scrollable Grid view, shown in Figure 5. This offers a conventional two-dimensional grid, which can be scrolled horizontally and vertically. As objects move off the grid, they fade away, and as they are to be included, they fade in.

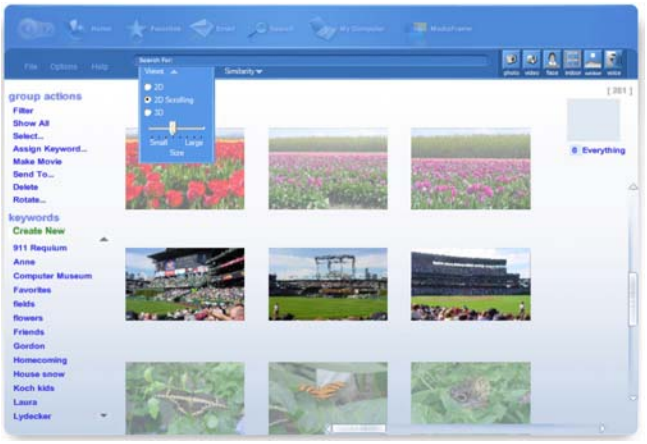

Figure 5: The Scrollable Grid view. The grid of objects may be scrolled horizontally or vertically and thumbnails can be smoothly resized.

All of the selection and labeling tools work the same in all three views.

IMPLEMENTATION

A short note on implementation of the system is warranted. We used the capabilities of the built-in graphics hardware in order to achieve smooth fading, scaling, and fluid animation of the thumbnails on the screen. Since we wanted the system to work on laptops, we made sure that we implemented an efficient texture management system and isolated the UI thread from the rendering thread (for rapid interactive responses). We also used a higher level language (C#) for development which allowed more rapid iteration on design ideas than lower level C++. The managed code bindings for DirectX performed well enough for high frame-rates (averaging 40 fps on an 850 MHz Pentium III with an NVIDIA GeForce2Go).

LESSONS LEARNED

We have used our prototype application ourselves to label several thousand images, usually in batches of about 250 or so. We've shown the prototype to a huge number of individuals who all have remarked on how intuitive the feature set and behavior of the system is. More importantly, we've also watched 10 users sort and label their own images. While not a formal and longitudinal study, this has been sufficient to learn some lessons.

- **Design experience is critical:** The appealing, fluidic motion of the interface, images smoothly fading in and out, and the colored glows for previews and selections are not mere design flourishes, but give the application a feeling of motion and energy. When the motion or color changes weren't well tuned, people were confused, or tired of the task more quickly and were more inclined to quit before they'd completed all the labeling they original intended**.**
- **Automatic label assignment important:** Since users knew that simply opening up the application and loading their objects would result in several useful labels being applied automatically, they were more inclined to cross that threshold and get started.
- **Time clusters:** Some like absolute time, while others prefer relative breaks.
- **Keyword Display:** We showed all the keywords that were present in the current working set. This often produced more keywords than was explicitly filtered for, sometimes confusing the users. After explanation, users understood, but we'd like to make it apparent without explanation.

FUTURE WORK

There are many ways to extend this system, but our primary motivation is not to make it larger, but rather even easier and more pleasant to use, by leveraging the mechanisms and ideas that are already in place.

Because our automatic labels are assigned by plug-in modules, we can easily experiment with other algorithms that are fast and create labels that are useful. The ability to recognize specific faces and label them with their owner's names would useful. We'd also like to have a more robust similarity matcher.

We'd like to support hierarchies of keys. Unlike folder hierarchies, a tree of keys simply means that any object given a label also is assigned the labels above it on the tree. Thus assigning the label "kitchen" to an image might automatically also give it the labels "remodeling project" and "house."

The biggest issue to further develop is scaling. Our system works well for about 500 or 600 objects, up to a half-dozen automatic labels, and a few dozen user-created labels. When the numbers go significantly beyond those points, the system slows down, objects become harder to distinguish by eye, scrolling through the list of labels becomes a chore, and the animation begins to crawl. The problems of scaling are common to all programs that try to organize and display massive amounts of visual information. Taking a cue from other visualization systems, we have started to experiment with a "firefly" representation, where our objects are simply points of light. Users can get a closer view on any object by hovering over it, or they can make a selection of a few hundred points and then edit them using the techniques described in this paper. Managing longer lists of labels, and hierarchies of labels, in an easy way is a more challenging task.

SUMMARY

Labeling images and video clips with appropriate key words turns unorganized heaps of media into collections of related objects, which may then be easily browsed, selected, and displayed.

Our system integrates a variety of visualization and interface techniques. The workflow is simple, feedback is

immediate, and the smooth animation and other visual design elements help to enliven an inherently repetitious process.

We have found that users like our system enough to assign useful labels to their collections, and informal observations suggest that they can assign meaningful labels to a body of about 500 objects in less than 5 minutes.

The MediaFrame browser unites a careful choice of interface elements, an elegant and pleasing design, smooth motion and animation, and a few simple tools that are predictable and consistent. The result is a friendly, useful system for turning shoeboxes of old photos into labeled collections that can be easily shared, distributed, and enjoyed.

HARDWARE REQUIREMENTS

The system can be demoed on any system that has DirectX9 and a minimum of a GeForce2MX graphics card. The author will demonstrate the system using a Dell 8100 Laptop.

REFERENCES

1. ACDSee. http://www.acdsystems.com/

2. Adobe Photoshop Album, http://www.adobe.com/

3. Bederson, Benjamin B, (2001), PhotoMesa: a zoomable image browser using quantum treemaps and bubblemaps, Proceedings of the 14th annual ACM symposium on User interface software and technology, November 11-14, Orlando, Florida

4. Canon ZoomBrowser, http://www.powershot.com/

5. Flickner, M., H. Sawhney, W. Niblack, J. Ashley, Q. Huang, and B. Dom et al. *Query by image and video content: the qbic system.* IEEE Computer, 28(9):23--32, 1995.

6. History of Kodak: Milestones 1878 to 1932. http://www.kodak.com/US/en/corp/aboutKodak/kodakHistory/milestones78to32.shtml

7. iPhoto, http://www.apple.com/iphoto/

8. Kang, H., & Shneiderman, B. (2000). Visualization Methods for Personal Photo Collections Browsing and Searching in the PbotoFinder. In Proceedings of IEEE International Conference on Multimedia and Expo (1CME2000) New York, pp. 1539-1542.

9. Kuchinsky, A., C. Pering , M. Creech , D. Freeze , B. Serra , J. Gwizdka, FotoFile: a consumer multimedia organization and retrieval system, Proceedings of SIGCHI. pp.496-503, May 15-20, 1999, Pittsburgh, PA.

10. Picassa Softare, http://www.picassa.net/

11. Platt, J. (2000). AutoAlbum: Clustering Digital Photographs Using Probabalistic Model Merging. In Proceedings of IEEE Workshop on Content-based Access of Image and Video Libraries (CBAIVL-2000) IEEE Press

12. Robertson, G., Cameron, K., Czerwinski, M., Robbins, D., Polyarchy visualization: visualizing multiple intersecting hierarchies. Proceedings of SIGCHI, pp. 423-430, 2002. Minneapolis, Minnesota, United States.

12. Rodden K., Wood K., How do people manage their digital photographs?, Proceedings of the SIGCHI conference on Human factors in computing systems, p.409-416, March 2003, Ft. Lauderdal, Florida, United States

13. Shneiderman, B., & Kang, H. (2000). Direct Annotation: A Drag-and-Drop Strategy for Labeling Photos. In Proceedings of IEEE Conference on Information Visualization (1V2000) New York: IEEE, pp. 88-98.

14. Virage Incorporated, http://www.virage.com/

The columns on the last page should be of approximately equal length.