A Method for Resizing Images by Content Perception

Anish Anil Patankar. Joy Bose
Web Services Group
Samsung R&D Institute India, Bangalore
{anish.p, joy.bose}@samsung.com

***Abstract*—Given the popularity of mobile and wearable devices, it is important to focus on the clarity of the displayed content for the end users. Currently, there is no link between the layout or size of displayed image and the information content in those images. In this paper, we present a method for optimally arranging and displaying a group of images, where the images are resized such that the relative size of the images is proportional to the number of objects in the images, or the number of faces. This ensures that the user gets to view faces or objects in images equally well regardless of the original size of the images. The method can be used for a number of applications in a mobile device, such as the gallery, web browser and video applications. We present implementation details to resize and layout a group of images displayed on a web page in a web browser. We also present results of a user study to see if such an approach might be desirable.**

Keywords— image processing; objectness criterion; object counting; image arrangement; image layout

I. Introduction

A recent survey estimated that 80% of photos were captured on mobile phones or tablets and only 15% on digital cameras, among 1 trillion photos taken in 2015 [1]. With the adoption of social networking platforms, photos are increasingly being consumed on mobile phones. Currently, when viewing images in an image application such as the Gallery app, the relative size of the images is not related to the information contained in those images. This leads to a situation where the user may not be able to see clearly those images where there is a lot of information e.g. more people in the image. This problem is more in devices where the display size is limited, for example on smaller mobile phones or wearable smartwatches.

When displaying a group of images in such a device, one needs a way to resize and layout them so the viewer can get the maximum information content. It would be good if the higher information content images are of a larger size than the smaller information content images. In this paper, we present a method for resizing and displaying images such that the images with higher perceived information content are bigger than the images with less information. We resize and re-layout images in proportion to the number of people or objects in the image. This enables the user to clearly view each person/object in the images. Our method can be applied for a number of applications in the mobile device, including the gallery or web browser.

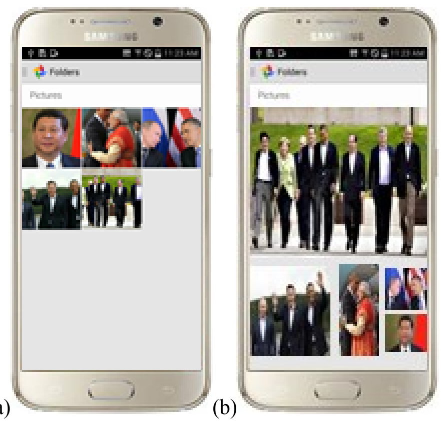

Fig. 1. An illustration of the image gallery application in a mobile phone, where (a) all the images are of the same size, and (b) images are resized in proportion to the number of people in each image

Fig. 1 illustrates the gallery application on a mobile device before and after resizing the images as per our method.

The rest of the paper is organized as follows: in section 2 we look at related work in the area. Section 3 introduces our method for image resizing and re-layouting. Section 4 describes the component blocks of our system. Section 5 includes some implementation details, while section 6 contains a user study. Section 7 mentions some use cases, and section 8 concludes the paper.

II. Related work

There has been quite some interest in automatic photo organization frameworks [2-4]. Commercial software applications and services like Google Photos, Facebook, Flickr etc. do allow users the option of automatic management of images based on various parameters. However, their focus is on organization of photos rather than the display layout of the photos.

The paper by Chen [5] talks of image slideshows where the relative display times of the images during the slideshow are based on some parameters that depend on the content of the image. But it does not mention object count as a parameter, nor do they resize the images based on that. Similarly, operating systems such as Windows 10 that have tiled layouts have the default display tile sizes that can be set and resized manually by the users and are not dependent on the information content of the icons.

A section of the book by Shao et al [6] mentions dynamic layouts of photos. Wang [7] discuss various layouts in image search results based on clustering similar images. Luo et al [8] mention selection of salient images in a group of images based on criterion such as size, sharpness, skin area shown in the image etc. Object detection and counting is, however, missing as a criterion in these works.

There are also a few patents that mention resizing of images while displaying. The patent by Gindele et al [9] mentions layouting of photos based on face size. US6748097B1 discusses photo layout based on image emphasis appeal. US20140098140 discussed displaying images as tiles, where the tile sizes are ordered as per the number or relevance of the displayed images. US5796401A talks of a layout manager for adapting the size of displayed objects automatically when the screen resolution is altered. Similarly, EP2105930B1 talks of a method for arranging a set of images based on extracting and comparing a number of image attributes, although here too the information content of the image is not considered.

In general, the related work found in the literature does not mention any method to determine the relative size of the images or tiles based on content i.e. by information content or saliency in the images. This is what we aim to cover in this paper.

III. Method for resizing images

As per Bodgan et al [10], the definition of an object in an image rests on characteristics such as a well-defined closed boundary in space and different appearance from their surroundings. Object detection can be performed by using features in the image including color contrast or edge density or by performing Bayesian integration.

Once the object detection or face detection is done, the resizing of the image is performed. There are two alternative approaches for resizing and displaying the images. The following subsections detail these approaches.

A. *Greedy approach for layouting the images*

In the greedy approach, we resize the images to fit into the available space. The steps for the greedy approach are as follows:

- Resize each images as per its resize ratio
- Sort the resized images as per the area
- Take the biggest image and fit into the bounding box that can fit the image
- Repeat for all the other images
- Resize the images if necessary to make a better fit in available space

B. *Bucket approach for layouting the images*

In this approach, we have a constant number of fixed sized buckets into which we fit the images.

The steps for the bucket tiled approach are as follows:

- Sort the resized images into n buckets (as specified by the UX) based on the resize ratio (for example if there are 3 buckets, then image size is different for each bucket but images within each bucket are of same size)
- Resize the images as per its bucket size
- Layout the images by buckets

Algorithm 1: Determination of Image Sizes

```
1. Run face and object detection algorithm on each image in the given set of images and determine bounding rectangles of the faces and objects
2. After all images have been processed, determine mean size (m) of faces/objects in all the images of the set.
3. Compute scaling factor for $i^{th}$ image as $m / m_i$, where $m_i$ is mean size of faces in $i^{th}$ image.
4. Scale each image in the set by the scaling factor determined in step 3.
```

Algorithm 1 provides the method to determine the resizing factor for each image in the given set of images. Algorithm 2 provides a way to modify the layout of all images in a web page.

Algorithm 2: Layouting Images: Bucket approach

```
1. Determine number of buckets (n) and image bounding box sizes for each bucket ($s_0, s_1, .. s_n$)
2. For each image in the given set, determine which bucket it falls into, depending on the size of the image and which bucket's bounding box range it falls into
3. Resize the image to fit into the bounding box range for the selected bucket
4. Layout and display all the images of bucket 1, then all images of bucket 2, and so on till bucket n
```

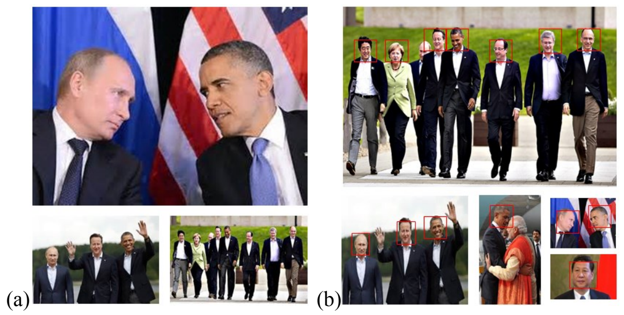

Fig. 2. An illustration of a group of images. (a) Without the image resizing algorithm and (b) After the images are resized. Here individual people could be identified in the group photo, more images could be accommodated in the same area, resulting in higher perceivable content, higher information to the user.

C. *Resizing images in a Gallery application*

The steps of the algorithm for resizing images in the gallery application are as follows:

- Get the new image stored in the gallery.

- Run face detectors on the image.
- Get number of faces and the smallest bounding box that includes all the faces.
- Store the face count and bounding box coordinates with the image file metadata, so that any gallery application can read the information and use it for layouting and display of the images.
- Determine the resize ratio of each image such that the face area/total area is constant (the principle is that each face must be shown at a similar size, irrespective of its actual image size) while keeping the aspect ratio of each of the images unchanged.

In case where the faces in the images are all of different sizes, there are various alternatives for resizing:

- resize the images according the mean face size
- resize the images according to biggest face size
- resize the images according to smallest face size

D. Resizing and relayouting images in a Web Page

A similar method as described for a gallery application can be used to resize the images in a web page as per the number of objects. Here, changes will be made to the web browser application. Here, JavaScript and DOM APIs are used to perform all the required steps. We detect human faces or objects using the method described previously. After this, we resize and re-layout the images in the web page. The script to do this can be embedded in the web browser or added as an extension or add-on to the web browser. Based on the sizes, positions and number of detected objects, the layout engine harmonizes the recommended sizes of all images on the web page. This constraint is applied so that the overall web page layout including non-image component does not get deformed.

IV. COMPONENTS OF THE SYSTEM

The components of our system for image resizing, as illustrated in the block diagram of fig. 3, are as follows:

- Object/ Face detection block: This implements the algorithm for detecting objects and/or faces.
- Object analysis block: This takes the detected face/ object coordinates from the previous block and analyses the same to detect the count, size and other features of the faces or objects. This block processes each image and generates three outputs: Size s, count c, and the feature vector for each object or face. Based on the outputs and the display parameters for image, the optimal image size is calculated.
- Image Size Calculator Block: This calculates the final image sizes based on the c and s values returned from the previous block as per the weights provided, as well as the display info.
- Display Info block: This provides the display parameters such as the screen size.

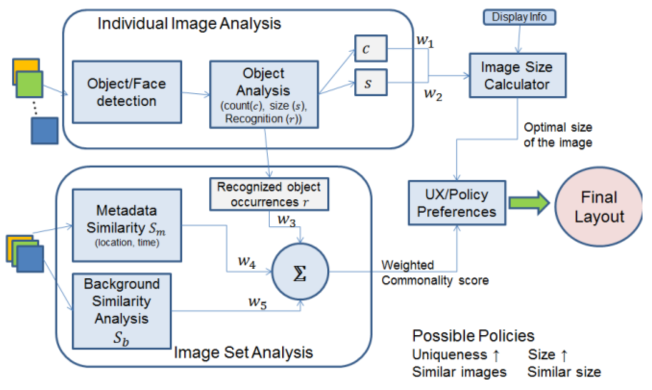
Individual Image Analysis
Object/Face detection
Object Analysis (count(c), size (s), Recognition (r))
c
s
w_1
w_2
Display Info
Image Size Calculator
Optimal size of the image
Recognized object occurrences r
w_3
Metadata Similarity S_m (location, time)
w_4
Σ
Background Similarity Analysis S_b
w_5
Image Set Analysis
Weighted Commonality score
UX/Policy Preferences
Final Layout
Possible Policies
Uniqueness ↑ Similar images
Size ↑ Similar size

Fig. 3.   Block diagram for the system of resizing images

- Recognized object occurrences block: This outputs the feature vector of the identified person or object in the image.
- Metadata Similarity block: This block takes metadata information from the images such as location and time when the image was captured. This metadata is used to find the similarity between the images.
- Background Similarity Analysis Block: This analyzes the backgrounds of the images from the set, and finds the similarity between the backgrounds. For example, the backgrounds of two images can both be ‘forest’ or ‘garden’.
- The weighted output of the blocks is taken and summation is done to calculate a value which we call the “weighted commonality score”. This value represents how much common the image in consideration is to other images in the same set of images or folder or album of images.
- User Experience (UX) policy preferences block: This is a module which applies the UX policy for how exactly the images should be displayed, taking input from the previous blocks. Based on the UX policy, the final layout of the images is decided and displayed.

V. IMPLEMENTATION AND RESULTS

In order to demonstrate the effectiveness of our method for resizing images, we implemented our method on a web server with a few chosen images, and then conducted a usability study to see if users preferred the original layout or the layout with resized images.

We execute JavaScript in the context of the webpage to grab all the IMG elements from the webpage. We applied a simple rule based filter to grab only the thumbnails so as to exclude logos and ads and other image elements. After that, we executed a face and object detection algorithm on the images. The details of the algorithm are already covered earlier. We gave preference to face detection over object detection i.e. if a face is detected in the image, we did not run object detection. This flows from the fact that human faces are usually more important than the objects. However, more studies may be required to handle corner cases.

The object detector and face detector modules both have identical APIs. They return the co-ordinates and sizes of the bounding boxes for all detected objects and faces respectively. The size estimator computes the most suitable size for a given image based on the sizes, positions and the number of detected objects within the image. Then the layout engine puts the resized images into a layout such that overall dimensions of all images put together are maintained as the original. A component called styler applies styles to specific elements i.e. actual images and their containers to achieve the overall desired effect using DOM APIs along with styles.

The implementation of the method can be adapted for different platforms and applications. We provide the core solution as a library to be used by various third party applications. The library has a convenient API for applications to integrate it in the layout module. We here provide the details of the web browser based implementation. We choose Google Chrome browser for its flexible and well documented extension APIs. However, the same implementation can easily be adapted for other web browsers as well.

For our implementation, we have used a version of the YEF real time object detection algorithm as explained by Abramson et al [11, 12] and implemented by LiuLiu [13] as a Javascript library, using canvas and DOM APIs.

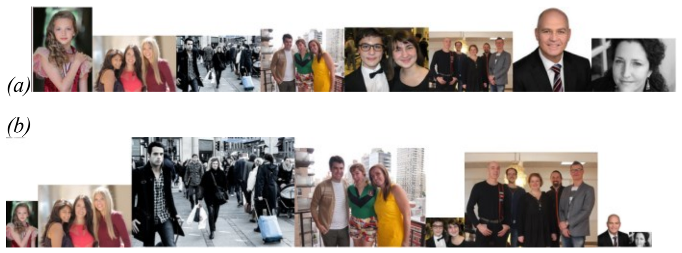

Fig. 4. Screenshot of a set of images with faces (a) before and (b) after our method of resizing images is applied.

Fig. 4 shows the results for a set of images shown on a webpage before and after our resizing plugin, embedded in the web browser, is applied. We can see that after our method is applied, all the faces are of the same size in the resized set of images regardless of the original size of the images.

VI. User Study

In order to gauge the usefulness of our method, we performed a user study on 330 technologically adept mobile phone users, aged between 23-41, of which 80 are female and the rest male, reflecting the gender mix of our organization. In our study we showed the users 5 images in a gallery arranged in the default layout and in the layout with images resized as per our method, on a desktop computer as well as on a smartphone.

We asked the following three questions from the users:

- Do you have eye strain or discomfort out of viewing images or application icons in a smartphone?
- Would you prefer if the images you view in a smartphone in a web browser are made bigger?
- Showing the users a gallery of 5 images with the default and our modified resized layout, we asked the users which of these two layouts they preferred.

Fig. 5 plots the results from our study. As we can see, an overwhelming number of people preferred the resizing of images, and also most of the users found the current image sizes to be too small for comfort, even though most people experienced little or no eye strain while viewing the images. This shows that our method can be useful to such users.

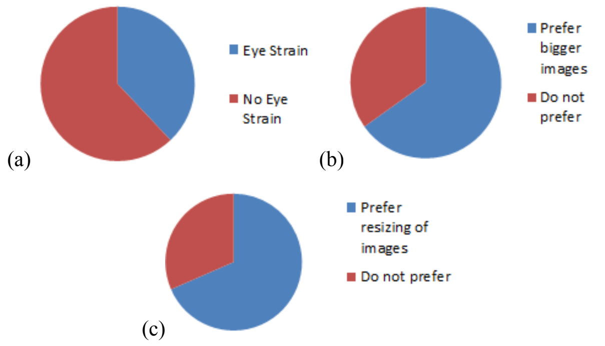

Fig. 5. Plotted results of a user survey with 330 users to gauge the usefulness of our system for resizing images. Majority of the users prefer bigger images, even though they do not have eye strain, and prefer the resized version of images as per the number of objects.

VII. Use Cases

The principle of resizing images based on their content can be applied in a number of use cases, such as the following:

- Image search results can be displayed with the images resized as per the actual information in each image. Multi-party video conferences
- For a multi-party video conference, the relative size of each party can be dependent on the number of people detected.
- The same principle can be used in social networking and chat sites, or tiled layouts in general, where the images or videos or live streams are resized as per the number of faces or any other measure of information content.

VIII. Conclusion and Future Work

In this paper we have presented the design of a system to resize images as per the number of objects or faces in each image. This system has been implemented for faces in a webpage using existing face detection plugins in a web browser. A patent for the system has also been filed [14]. We believe that widespread usage of such a system can help the users to see images more clearly and thus increase the usability of mobile phones and wearable smartwatches and other such devices.

In future, we plan to extend the implementation of our system for objects other than faces and for other applications on mobile devices such as the gallery application.

REFERENCES

[1] Stephen Heyman. Photos, Photos Everywhere. July 29 2015. [Online]. Available: http://www.nytimes.com/2015/07/23/arts/international/photos-photos-everywhere.html? Retrieved on 28 April 2016.

[2] Li, Cheng-Hung, et al. Image content clustering and summarization for photo collections. In Proc. Multimedia and Expo, 2006 IEEE International Conference on. IEEE, 2006.

[3] Cooray, Saman H. and O'Connor, Noel E. and Gurrin, Cathal and Jones, Gareth J.F. and O'Hare, Neil and Smeaton, Alan F. (2006) Identifying person re-occurrences for personal photo management applications. In: VIE 2006 - IET International Conference on Visual Information Engineering, 26-28 Sept. 2006, Bangalore, India.

[4] Abdel-Mottaleb M, Chen L. Content-based photo album management using faces' arrangement. InMultimedia and Expo, 2004. ICME'04. 2004 IEEE International Conference on 2004 Jun 27 (Vol. 3, pp. 2071-2074). IEEE.

[5] Jun-Cheng Chen, Wei-Ta Chu, Jin-Hau Kuo, Chung-Yi Weng, and Ja-Ling Wu. 2006. Tiling slideshow. In Proceedings of the 14th ACM International Conference on Multimedia (MM '06). ACM, New York, NY, USA, 25-34. DOI=http://dx.doi.org/10.1145/1180639.1180653

[6] Multimedia Interaction and Intelligent User Interfaces: Principles, Methods and Applications. Eds: Shao, L., Shan, C., Luo, J., Etoh, M. (Eds.). Springer. 2010

[7] Wang, C., Reese, J.P., Zhang, H., Tao, J. and Nemiroff, R.J., 2013, February. iMap: A stable layout for navigating large image collections with embedded search. In IS&T/SPIE Electronic Imaging (pp. 86540K-86540K). International Society for Optics and Photonics.

[8] J. Luo, A. Singhal, A. Savakis. Efficient Mobile Imaging Using Emphasis Image Selection. In Proc PICS 2003.

[9] E.B. Gindele et al. US Patent US 6748097 B1. Method for varying the number, size, and magnification of photographic prints based on image emphasis and appeal

[10] Alexe, Bogdan, Thomas Deselaers, and Vittorio Ferrari. "What is an object?." Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on. IEEE, 2010

[11] Yotam Abramson, Bruno Steux, Hicham Ghorayeb. YEF (Yet Even Faster) Real-Time Object Detection. In Proc. International Workshop on Automatic Learning and Real-Time, ALaRT 2005, September 7-8, Siegen, Germany.

[12] Abramson, Y., Steux, B. and Ghorayeb, H., 2007. Yet Even Faster (YEF) real-time object detection. International Journal of Intelligent Systems Technologies and Applications, 2(2-3), pp.102-112.

[13] LiuLiu. JavaScript Face Detection Explained. Feb 19, 2012. [Online]. Available: liuliu.me/eyes/javascript-face-detection-explained/

[14] Anish Anil Patankar, Joy Bose, “Electronic device for displaying a plurality of images and method for processing an image”, WIPO Patent WO 2016204449 A1, filed 9 June 2016.