Utah State University
DigitalCommons@USU

All Graduate Theses and Dissertations | Graduate Studies

5-2016

Cluster-Based Salient Object Detection Using K-Means Merging and Keypoint Separation with Rectangular Centers

Robert Buck
Utah State University

Follow this and additional works at: https://digitalcommons.usu.edu/etd

 Part of the Computer Sciences Commons

Recommended Citation

Buck, Robert, "Cluster-Based Salient Object Detection Using K-Means Merging and Keypoint Separation with Rectangular Centers" (2016). *All Graduate Theses and Dissertations*. 4631.
https://digitalcommons.usu.edu/etd/4631

This Thesis is brought to you for free and open access by the Graduate Studies at DigitalCommons@USU. It has been accepted for inclusion in All Graduate Theses and Dissertations by an authorized administrator of DigitalCommons@USU. For more information, please contact digitalcommons@usu.edu.

CLUSTER-BASED SALIENT OBJECT DETECTION USING K-MEANS MERGING AND KEYPOINT SEPARATION WITH RECTANGULAR CENTERS

by

Robert Buck

A thesis submitted in partial fulfillment
of the requirements for the degree

of

MASTER OF SCIENCE

in

Computer Science

Approved:

_____	_____
Dr. Xiaojun Qi Major Professor	Dr. Curtis Dyreson Committee Member
_____	_____
Dr. Vicki Allan Committee Member	Dr. Mark R. McLellan Vice President for Research and Dean of the School of Graduate Studies

UTAH STATE UNIVERSITY
Logan, Utah

2016

Copyright © Robert Buck 2016

All Rights Reserved

ABSTRACT

Cluster-based Salient Object Detection Using K-means Merging and Keypoint Separation with Rectangular Centers

by

Robert Buck, Master of Science

Utah State University, 2016

Major Professor: Dr. Xiaojun Qi
Department: Computer Science

Advances in digital technology have brought more images into daily life through computers, phones, and even automobiles. This significant increase in visual data has increased the necessity for better ways to sift through large amounts of image files. It also has amplified the desire for methods to isolate important objects in images.

Salient Object Detection (SOD) is a computer vision topic that explores ways to distinguish important (salient) object(s) from the background in images. SOD algorithms attempt to mimic the Human Visual System (HVS), by using image characteristics such as color contrast and spatial locations, to separate salient objects. This paper introduces a novel SOD approach to better isolate salient objects by incorporating innovative features including k-means merging preprocessing, keypoint-based postprocessing, subcluster segmentation, and rectangular centers.

Using several publicly available and accepted image databases, I confirm in this thesis the effectiveness of my SOD algorithm. I also demonstrate my method is superior to two state-of-the-art approaches.

(53 pages)

PUBLIC ABSTRACT

Cluster-based Salient Object Detection Using K-means Merging

and Keypoint Separation with Rectangular Centers

Robert K. Buck

The explosion of internet traffic, advent of social media sites such as Facebook and Twitter, and increased availability of digital cameras has saturated life with images and videos. Never before has it been so important to sift quickly through large amounts of digital information. Salient Object Detection (SOD) is a computer vision topic that finds methods to locate important objects in pictures. SOD has proven to be helpful in numerous applications such as image forgery detection and traffic sign recognition. In this thesis, I outline a novel SOD technique to automatically isolate important objects from the background in images.

ACKNOWLEDGMENTS

To my major professor, Dr. Xiaojun Qi, for providing essential guidance throughout the research process.

To Dr. Curtis Dyreson and Dr. Vicki Allan for reviewing my thesis and giving valuable feedback.

To all those that helped me meet a tight defense deadline.

To my family for their endless love and support.

Robert Buck

CONTENTS

Page

LIST OF TABLES

Table Page

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Salient Object Detection (SOD) has become a hot topic in computer vision research. A salient object is basically a visually distinct object in an image. An image can contain zero or many different salient objects, and the purpose of SOD is to separate salient objects from the background. "Perfect" detection, known as Ground Truth (GT), will show salient object(s) in white and everything else in black. Examples are shown in

Figure 1. The first two rows present examples of images from the SED1 database (containing 100 images with one salient object per image) and their corresponding SOD results. The last two rows present example images from the SED2 database (containing 100 images with two salient regions) and their corresponding SOD results.

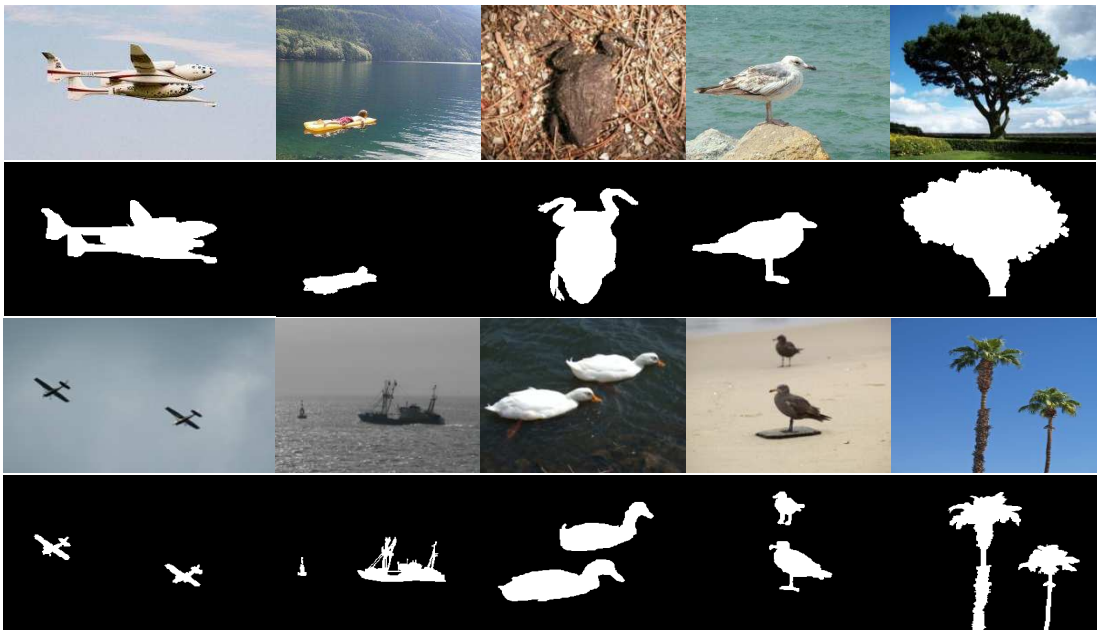


Figure 1 Example images and their corresponding SOD results. First Row: Images from the SED1 database. Second Row: Ground truth for images in the first row. Third Row: Images from the SED2 database. Fourth Row: Ground truth for images in the third row.

SOD is an important topic because it is helpful in a wide range of applications. Z. Ren et al. considered saliency detection as a precursor to object identification [1]. They use a PageRank algorithm to improve saliency; then they attempt to identify flowers, animals, and insects. Anitha and Leveenbose apply saliency detection to image forgery detection [2], which aims to determine whether objects have been added or removed from an image or if the color of an image has been modified. Fu and his team apply saliency detection to traffic sign recognition [3]. They use a machine-learning-based approach to detect and classify traffic signs. Their work is specifically important for autonomous vehicles.

However, SOD is challenging because it attempts to accomplish the same task as the Human Visual System (HVS) - a complex system that is difficult to replicate. The HVS uses a number of visual cues (such as color contrast and spatial locations), combined with prior knowledge, to easily identify salient objects. SOD can model many similar features, but at costs of time and complexity. Part of the challenge is that an object can be salient in one image and background in another. For example, the tree in image 5 of the first row in

Figure 1 is the salient object; however, in image 2 of the first row in Figure 1, the trees are part of the background. Truly SOD is a difficult problem, but the applications are numerous and important.

CHAPTER 2

BACKGROUND

Many approaches have been suggested for accurate SOD. However, they all generally have a few stages in common, as illustrated in

Figure 2. The first step, image segmentation, divides the image into various regions. Prior (also referred to as cue) calculation computes various scores for characteristics of the regions generated in step 1. As the name implies (prior combination), the last step is to combine the scores generated from step 2 into a final saliency map. Each of these stages is explored further in the sections that follow.

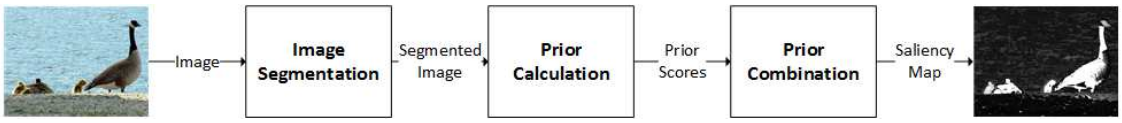


Figure 2 Basic Methodology for Salient Object Detection

2.1 Three Basic Steps for Salient Objects Detection (SOD)

2.1.1 Image Segmentation

Image segmentation, the first and most important step, separates the image into manageable pieces. If large numbers of salient and non-salient pixels are considered together, it is impossible to accurately detect saliency. Thus the choices of region type and methodology are critical.

Much work performed in the last few decades regarding saliency was based at the pixel level ([4], [5]). However, a pixel is not a very accurate portrayal of an image or even a local object. A pixel offers no context or indication to what object it belongs. As such, modern SOD methods work on groups of pixels.

State-of-the-art segmentation methods generally fall into two categories: superpixel-based ([1], [6], [7]) and cluster-based ([8], [9]) segmentation. Superpixels have the advantage of being continuous; clusters, on the other hand, are not necessarily grouped spatially. However, an object that appears in multiple areas of an image can be grouped into one cluster. The choice of segmentation determines what priors can be used to calculate saliency.

2.1.2 Prior Calculation

Following segmentation, calculations are performed on characteristics (called priors) of the regions. The most common prior is color contrast ([8], [10], [11], [12]). Some color contrast calculations compare regions globally to all other regions. Other methods use color contrast related to neighboring regions.

Another common prior has to do with location, sometimes called a spatial or center prior ([13], [14]). It is based on the idea that the HVS focuses on objects closer to the center of an image. Most methods use some form of center bias, even when it is not directly a prior calculation.

A more recent prior is color distribution ([13], [15]). It is defined as the spatial variation of the color in a region. A salient region will typically have compact color distribution.

2.1.3 Prior Combination

Once the priors have been calculated, they must be combined to form a final saliency map. Priors can be combined using a number of methods including: average, weighted average, and multiplication. The average method calculates the average of the scores obtained from the prior calculation. The weighted average method assigns different weights to each prior, and then computes the average of the weighted scores. The multiplication method multiplies the scores obtained from the prior calculation. Multiplication is the most common approach; however, it can be linear ([8]) or non-linear ([15]).

2.2 State-of-the-art Methods

The state-of-the-art method closest to my approach is that in [8]. They use spatial and contrast priors (i.e., cues) to determine saliency. They use clusters for their segmentation and linear multiplication for combination. Further they also apply their method to co-saliency, which looks for saliency across multiple related images. For my final thesis work, I do not extend my method to this application.

Another interesting approach is given in [16]. Here, Qin and his team discuss saliency using cellular automata. They focus on the background by creating a map based on color and spatial contrast. They also emphasize differences among neighboring regions. Their approach differs in using superpixel segmentation, and their combination method requires many steps.

Li et al. ([17]) create two maps as part of their method, the Salient Object Driven Measure (SODM) map and the Background Driven Measure (BDM) map. SODM uses

priors such as color contrast to isolate salient objects; BDM uses boundary information to deemphasize background pixels. In this way, they focus on highlighting the foreground (salient pixels) and suppressing the background (non-salient pixels).

Zhang and his group ([18]) offer a more complex approach. They first generate sets of superpixels at multiple scales. Priors are then calculated using the regions at each scale and later combined into a single saliency map. In this way the team seeks to better detect objects of varying sizes.

CHAPTER 3

SALIENT OBJECT DETECTION METHOD

My method builds upon previous work with several extensions to state-of-the-art algorithms. The block diagram of the method is shown in

Figure 3. In addition to the general three steps (image segmentation, prior calculation, and prior combination), my system adds many novel features.

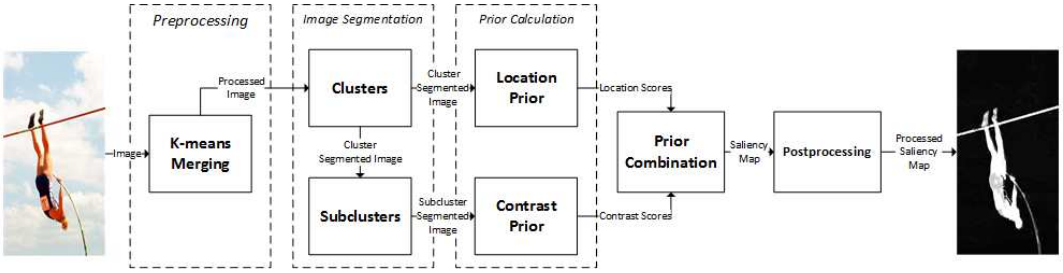


Figure 3 Block Diagram for My Method for Salient Object Detection

Specifically, my approach adds a subclustering process to the image segmentation stage. I calculate a location prior at the cluster level and a contrast prior at the subcluster level. I also define a new image center in the prior calculation stage. Instead of a conventional single point for the center, the new method introduces a rectangular center. Further, my method includes preprocessing and postprocessing stages to further extract the salient objects.

3.1 Preprocessing

The goal of preprocessing is to remove features from an image that may interfere with prior calculations (will be explained in section 3.3) without eliminating necessary regions in the image. To accomplish this goal, I introduce a k-means merging method that combines small regions into larger neighbors. The algorithmic view is shown in Figure 4.

Algorithm 1

Input: Original image I

1: Create a set of clusters L from the input image I using k-means as

$$L = \{C_k\}_{k=1}^{K_c}$$

where K_c is the number of clusters and each cluster is defined by

$$C_k = \{p_n\}_{n=1}^{n_k}, where\ p_n \in I\ and\ n_k = size(C_k).$$

2: Find all smaller clusters C_i^s such that

$$size(C_i^s) \leq \frac{hw}{\alpha K_c}$$

where h is the height of I, w is the width of I, and α is an empirically determined constant that controls the minimum cluster size.

3: For each small region C_i^s :
3.1 Determine the neighbor cluster C^n which has the most pixels that border cluster C_i^s.
3.2: Merge C_i^s into C^n if

$$size(C^n) \leq \frac{hw}{\beta K_c} and\ d_n \geq \frac{3D}{8}$$

where d_n is the number of pixels of C^n that border C_i^s, D is the total number of pixels that border C_i^s, and β is an empirically determined constant.

3.3: Otherwise, keep C_i^s
3.4: Repeat steps 1-3 as long as at least one small cluster is merged into another.

Output: Processed Image J

Figure 4 Algorithmic View of K-means Merging Method

First, I use the k-means method to separate an image I into K_C clusters (for my purposes, K_C=9 is most effective). Each pixel in I is composed of a 3-dimensional vector in the Lab color space, and these vectors are the input for this k-means implementation. Small clusters are isolated by setting α=4. For each small cluster I find the neighbor cluster that borders the small cluster most. The two clusters are then merged if the neighbor cluster is not too large (β=1) and the two clusters share a sufficiently large border. In this way, similarly sized clusters are combined without removing potential salient objects from large backgrounds. This process is then repeated until 9 suitably sized clusters are generated. In other words, the k-means merging algorithm ensures that each of the K_C clusters is not too small. The preprocessing results for three sample images are shown in Figure 5. After applying the k-means merging algorithm, the airplane regions in image 1 become nearly one solid object, the bird in image 3 also becomes more uniform, and the tree in image 2 is barely changed.

Figure 5 Results of Applying K-means Merging. First Row: Original images. Second Row: Processed images.

3.2 Image Segmentation

The goal of image segmentation is to divide the image into manageable pieces that prepare the image for prior calculation. My method employs a cluster-based technique to preserve edge integrity and object coherence. I also add a subclustering process to help separate similar colored objects.

3.2.1 Cluster-based Segmentation

Clusters, for my method, follow directly from the preprocessing stage discussed in section 3.1. Here, I simply formally define the clustering process and provide sample results in Figure 6. A set of clusters is defined by:

$$L = \{C_k\}_{k=1}^{K_c}$$

where a cluster is given by $C_k = \{p_n\}_{n=1}^{n_k}$, such that $p_n \in J$. K-means is used to separate pixels based on the Lab color space. Each pixel is therefore represented by a 3-dimensional vector (*L, a, b*). The individual dimensions are first normalized to values between [0, 1] (in my method Gaussian normalization is used unless otherwise noted).

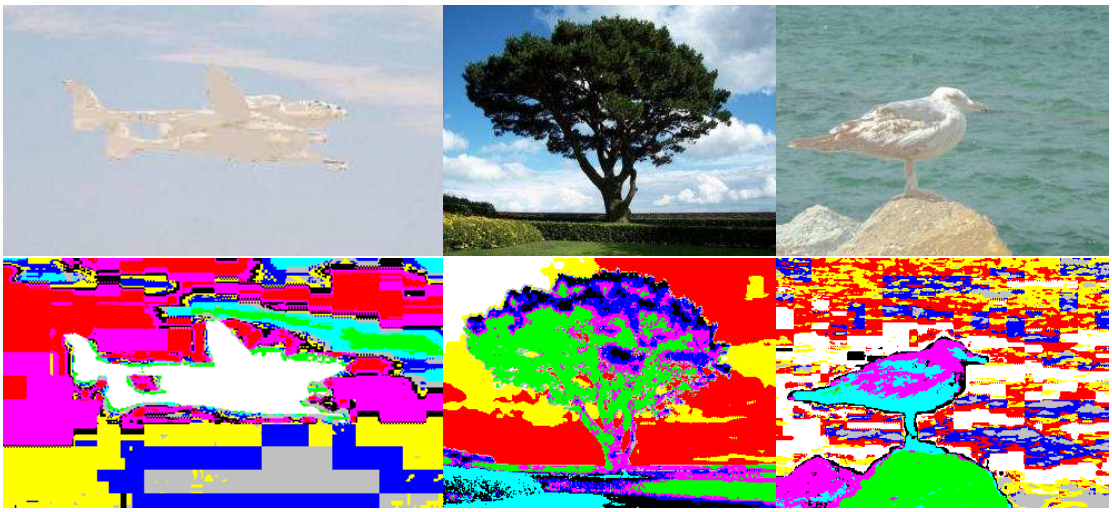


Figure 6 Examples of Cluster-based Segmentation. First Row: Processed images. Second Row: Images segmented into clusters.

3.2.2 Subcluster-based Segmentation

As evidenced clearly in the second image of Figure 6, the clustering process can sometimes group together pixels that are both salient and non-salient. In order to alleviate this problem, I further divide each cluster into subclusters. Formally, an image separates into a set of subclusters:

$$U = \{S_k\}_{k=1}^{K_c K_s}, \text{ where } S_1 \cup S_2 \ldots \cup S_{K_s} = C_1, \ldots S_{K_c K_s - 2} \cup S_{K_c K_s - 1} \cup S_{K_c K_s} = C_{K_c}$$

The k-means method is again applied to divide each cluster into K_s subclusters, where the value of K_s is empirically chosen as 3. Here, I use location information for the k-means method so that each pixel is defined by its 2-dimensional location vector (x, y). Sample subclustering results for an example cluster of the same three sample images are shown in Figure 7. It clearly shows that salient and non-salient pixels are now better separated.

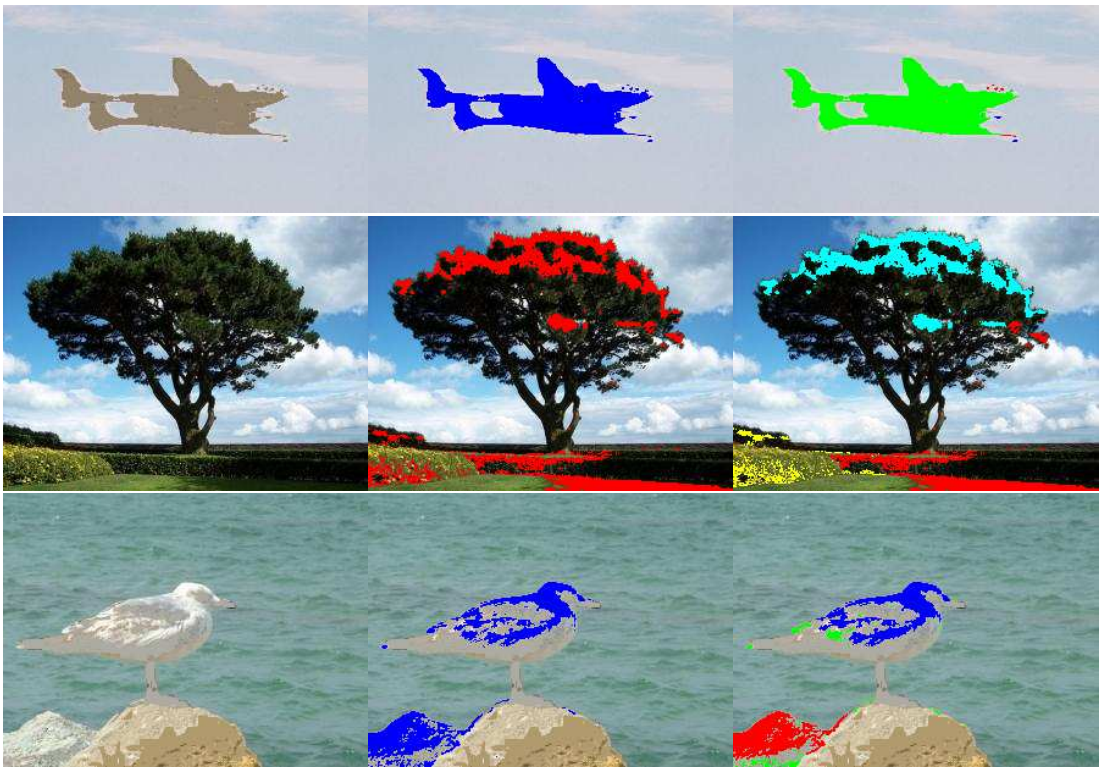


Figure 7 Examples of Subcluster Process. First Column: Processed images. Second Column: Sample cluster. Third Column: Subclusters

3.3 Prior Calculation

Now that images have been separated into appropriate regions represented by nine clusters and 27 subclusters, I next calculate the corresponding cluster and subcluster priors. The goal of prior calculation is to assign salient scores to regions, where a higher score indicates more salient (e.g. important objects) and a lower score indicates less salient (e.g. less important objects and background). My method uses both location and contrast characteristics of the clusters and subclusters to compute the salient scores.

3.3.1 Location Prior

The first prior I consider is the position of a region in an image. Many algorithms use the centroid in calculating center bias. However, for a cluster-based segmentation

method, the centroid is not an appropriate value because a cluster may have all its pixels along the border of an image, and yet have its centroid at the center of the image. Instead, I use the average distance from the center in calculating center bias.

Before defining the distance from center for a region, I first offer a new view of the center of an image. The standard view of the center is a point half way down the rows and halfway across the columns of an image. However, this concept of center ignores the shape of an image. Instead I define center starting at the border. The maximum distance from the image center to any pixel in an image *I* is defined by:

$$m = ceil(\min\left(\frac{h}{2}, \frac{w}{2}\right) - 1), \quad (1)$$

where *h* and *w* are the height and width of an image, respectively, and *ceil(a)* returns the least integer that is greater than or equal to *a*. It should be noted that the maximum distance is actually the distance from any border pixel to the center of an image. To facilitate the calculation of the distance from any pixel to the image center, I also define the distance from any pixel p to the image border to be:

$$d_b(p) = \min(x - 1, y - 1, w - x, h - y), \quad (2)$$

where (x,y) represents the location of pixel p with (1,1) being the location of the upper-left corner of the image. Using equations (1) and (2), the distance from any pixel *p* to the image center is calculated with:

$$d_c(p) = \begin{cases} m - d_b(p) & d_b(p) < m \\ 0 & otherwise \end{cases}, \quad (3)$$

For example, when the image is a square image (i.e. it has the same number of rows and columns) with odd dimensions, the rectangular center is a single center point. When the image is a rectangular image with the smallest dimension as an odd number, the

rectangular center is a single line. An example of a distance matrix which stores the distance from any given pixel to the image center for a 12x27 image is shown in Figure 8 (pixels with the value of 0 are defined as the image center). Figure 8 clearly shows that the image center is a rectangle.

$$\begin{bmatrix}
5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 \\
5 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 5 \\
5 & 4 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 4 & 5 \\
5 & 4 & 3 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 3 & 4 & 5 \\
5 & 4 & 3 & 2 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 2 & 3 & 4 & 5 \\
5 & 4 & 3 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 3 & 4 & 5 \\
5 & 4 & 3 & 2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 3 & 4 & 5 \\
5 & 4 & 3 & 2 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 2 & 3 & 4 & 5 \\
5 & 4 & 3 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 3 & 4 & 5 \\
5 & 4 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 3 & 4 & 5 \\
5 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 4 & 5 \\
5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5 & 5
\end{bmatrix}$$

Figure 8 Illustration of a Rectangular Center Matrix

Lastly, the distance values in the rectangular center matrix are normalized to the range [0, 1] by:

$$z(p) = d_c(p)/m \tag{4}$$

With a new image center defined, I now define the distance from the image center for a cluster (region). Specifically, the distance from the center to a cluster C_k is computed as the mean normalized distance from all pixels within the cluster to the center. This distance can be formally defined as follows:

$$D_s(C_k) = \frac{\sum_{n=1}^{n_k} z(p_n)}{n_k} \tag{5}$$

where $p_n \in C_k$ and n_k is the total number of pixels in C_k.

Finally, I can define the location prior calculation. It consists of two separate components: the distance from a cluster C_k to the center and the border pixel ratio. The location prior is defined as:

$$P_l(C_k) = (1 - D_s(C_k))(1 - \frac{b_k}{B}), \quad (6)$$

where B is the total number of border pixels of an image and b_k is the number of pixels of C_k on the border of an image. In conclusion, a location saliency map for an image I is generated from the location prior where each pixel p maps as follows:

$$M_l(p) = P_l(C_k), p \in C_k \text{ and } k \in [1,9] \quad (7)$$

Each map value is then normalized to [0, 1]. The maps generated from the location prior for three sample images are shown in the top row of Figure 9.

3.3.2 Contrast Prior

Next, I define a color contrast prior by first expressing a "color distance" between two subclusters S_i and S_j as follows:

$$D_c(S_i, S_j) = \sqrt{(l_i - l_j)^2 + (a_i - a_j)^2 + (b_i - b_j)^2}, \quad (8)$$

where (l_i, a_i, b_i) is the mean color value for subcluster S_i in the Lab color space (i.e., l_i is the mean value of the L color component for subcluster S_i). I then multiply the color distance with the region size ratio and the center bias as defined in Eq. (5). This leads to the contrast prior of a subcluster S_k:

$$P_c(S_k) = \frac{\sum_{i=1, i \neq k}^{K_c * K_s} D_c(S_k, S_i) \frac{n_k}{N} (1 - D_s(C_k))}{K_c K_s - 1}, \quad (9)$$

where N is the total number of pixels in an image. As with location, a contrast saliency map is created for image I using each pixel p as follows:

$$M_c(p) = P_c(S_k), p \in S_k \; and \; k \in [1, K_c K_s] \tag{10}$$

where values of M_c are again normalized to [0, 1]. The maps generated from the contrast prior for three sample images are shown in the second row of Figure 9.

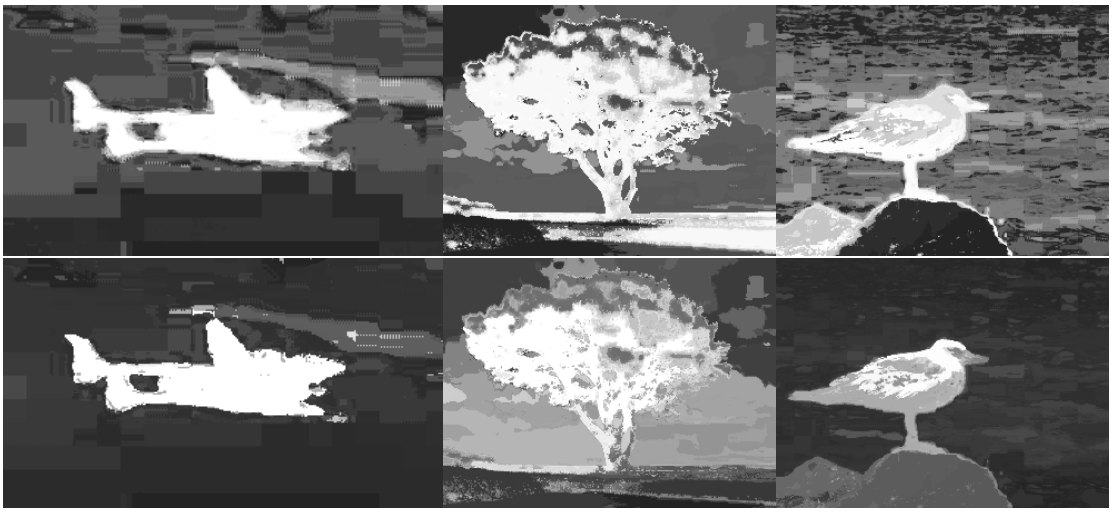


Figure 9 Prior Saliency Maps. First Row: Location Maps. Second Row: Contrast Maps

3.4 Prior Combination

After computing the two priors, namely location and contrast, I combine the priors with using point-wise multiplication. My method employs multiplication in order to help suppress noise ([8]). The value in the combined saliency map for pixel p in image I is generated by:

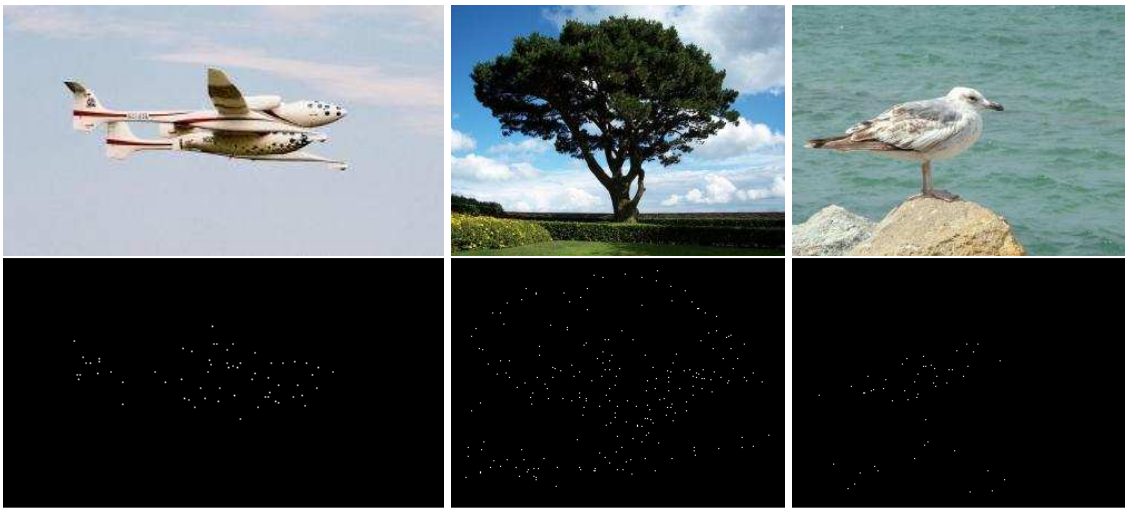
$$M_o(p) = \prod M = M_l(p) M_c(p) \tag{11}$$

The values in the combination map are then normalized to [0, 1]. Combination maps for three sample images are shown in Figure 10.

Figure 10 Combination Saliency Maps.

3.5 Postprocessing

The goal of postprocessing is to further separate salient objects from the background. Though keypoints are not necessarily good in predicting saliency, they do isolate important regions and thus can help separate salient regions. Many keypoint detector algorithms exist including Speeded Up Robust Features (SURF), Minimum Eigenvalue Features (MEF), Features from Accelerated Segment Test (FAST), Harris-Stephens (HARRIS), and Binary Robust Invariant Scalable Keypoints (BRISK). SURF is intended to find blobs in an image; the latter four algorithms are designed to detect corners. The keypoints detected by each algorithm for three sample images are shown in Figure 11.



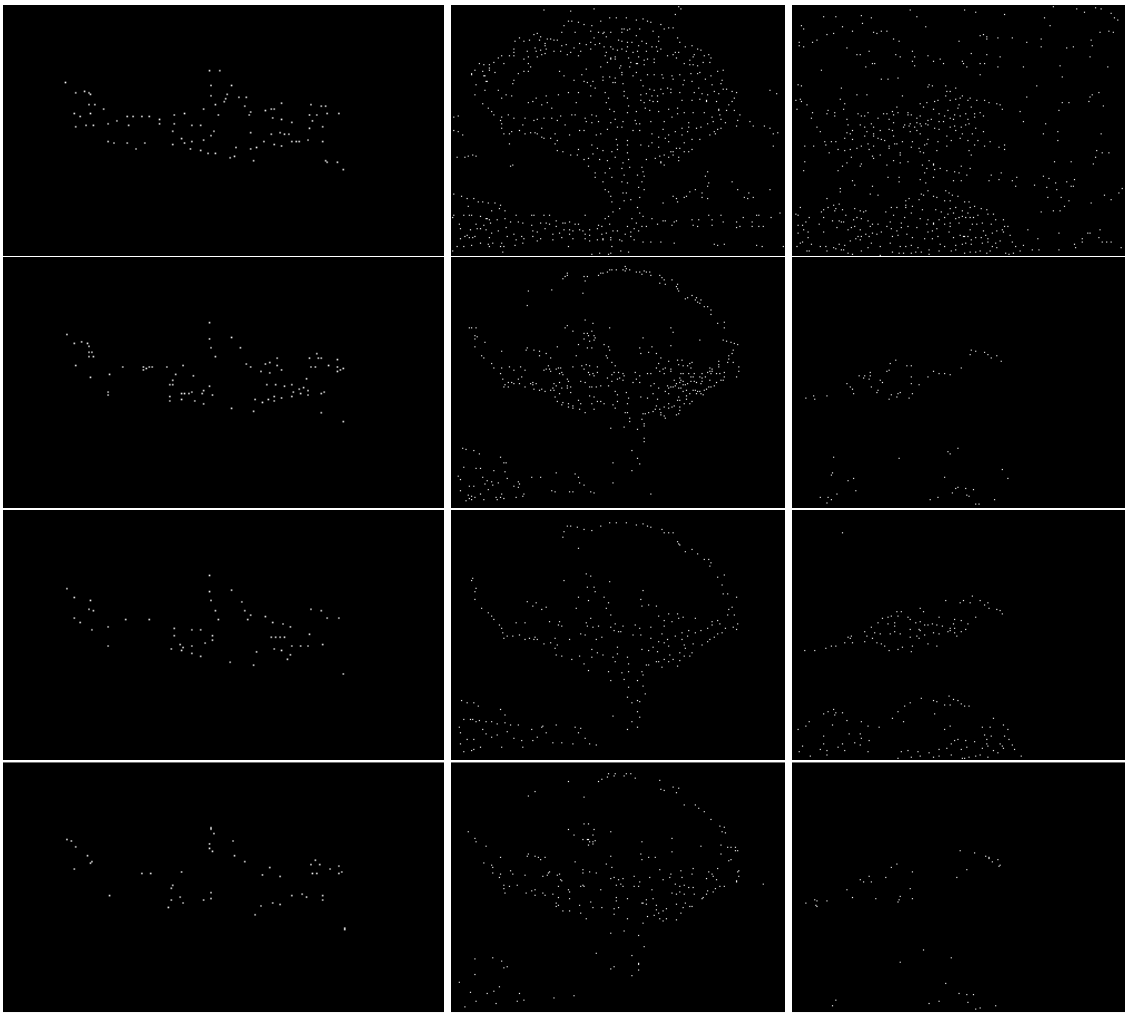


Figure 11 Results of Keypoint Detectors. First Row: Original images. Second Row: SURF results. Third Row: MEF results. Fourth Row: FAST results. Fifth Row: HARRIS Results. Sixth Row: BRISK Results

In order to incorporate the keypoint detectors, I first define a keypoint significance matrix $W(s)$, whose size is (2s-1) x (2s-1):

$$W(s) = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & & \vdots & \vdots & \vdots & & 1 \\ 1 & \cdots & s-1 & s-1 & s-1 & \cdots & 1 \\ 1 & \cdots & s-1 & s & s-1 & \cdots & 1 \\ 1 & \cdots & s-1 & s-1 & s-1 & \cdots & 1 \\ 1 & & \vdots & \vdots & \vdots & & 1 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \tag{12}$$

where *s* is the significance value predetermined based on the chosen keypoint type. In my system, I use the blob detector (SURF) combined with the most effective corner detector (empirically determined to be MEF) with *s* empirically determined to be 7 and 3 for SURF and MEF, respectably. In this way, each located keypoint has its associated keypoint significance matrix to cover its neighboring region. Specifically, the center of the keypoint significance matrix is positioned at the located keypoint. If *k* number of keypoints are found by a keypoint detector, their corresponding *k* keypoint significance matrices are positioned at their designated locations. The values in the overlapped keypoint significance matrices are accumulated to obtain the final value in the keypoint map, which has the same size as the original image. This is equivalent to applying the keypoint significance matrix over keypoints identified in the original image *I* by a given keypoint detector (SURF or MEF) such that:

$$M_{type}(I) = W(s_{type})[kp] \tag{13}$$

where *type* can be SURF or MEF, *kp* represents a keypoint, and $W(s_{type})$ indicates applying the keypoint significance matrix for *type* on keypoint *kp*. The final keypoint map M_k for an image *I* combines the two keypoint maps, namely, M_{SURF} and M_{MEF}, as follows:

$$M_k = M_{SURF} + M_{MEF} \tag{14}$$

and again the values of M_k are normalized to [0, 1].

Keypoint maps (e.g., keypoint map obtained by using SURF detector, keypoint map obtained by using MEF detector, and the final combined keypoint map) for three sample images are illustrated in Figure 12.

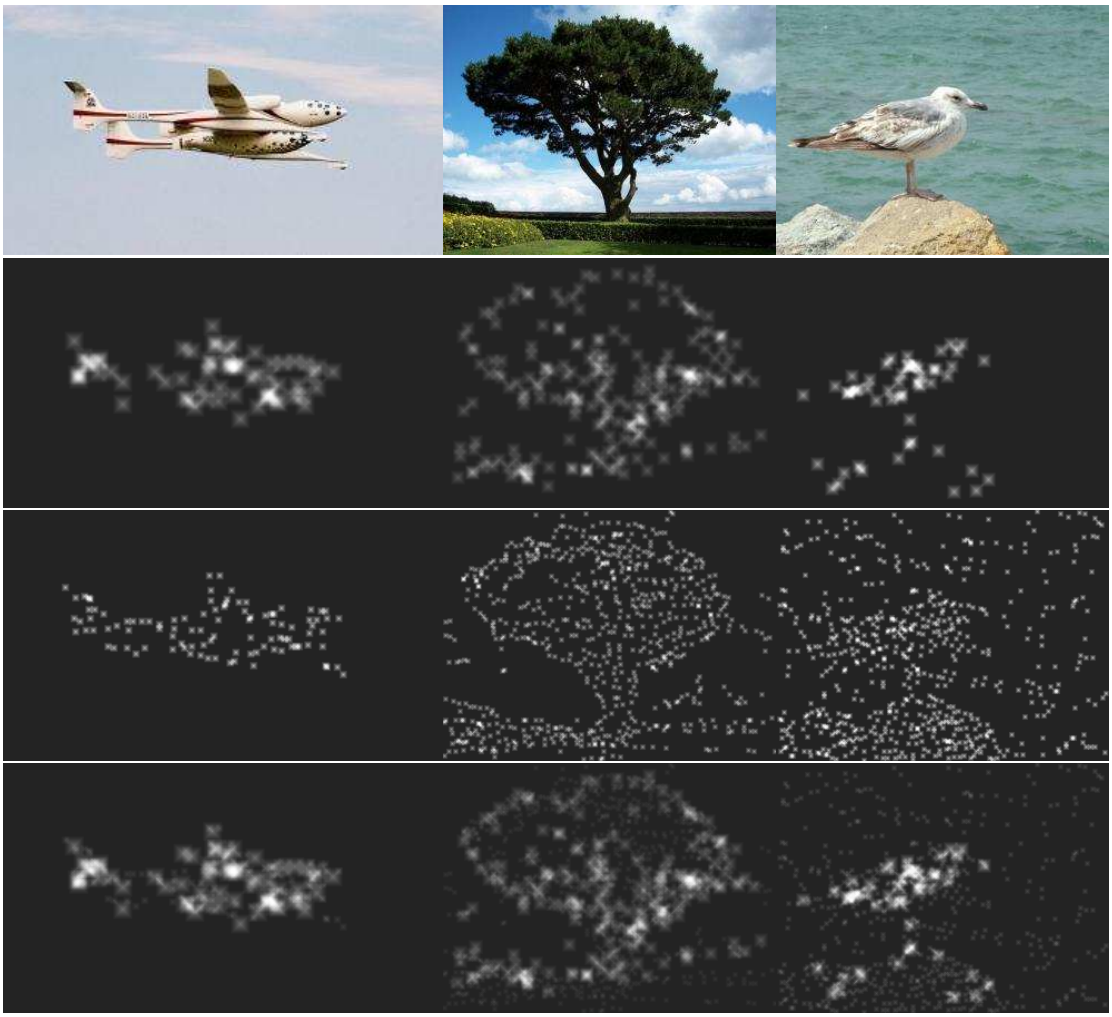


Figure 12 Examples of Keypoint Maps. First Row: Original images. Second Row: SURF maps. Third Row: MEF maps. Fourth Row: Combined keypoint maps.

As the last step, I generate the final saliency map for an image by point-wise multiplying the normalized M_k with the combination map M_o derived in Eq. (11):

$$M = M_o M_k \tag{15}$$

The saliency maps for three sample images are given in the last row of Figure 13 along with intermediate step results (e.g., combination map M_o in the second row and the keypoint map M_k in the third row).

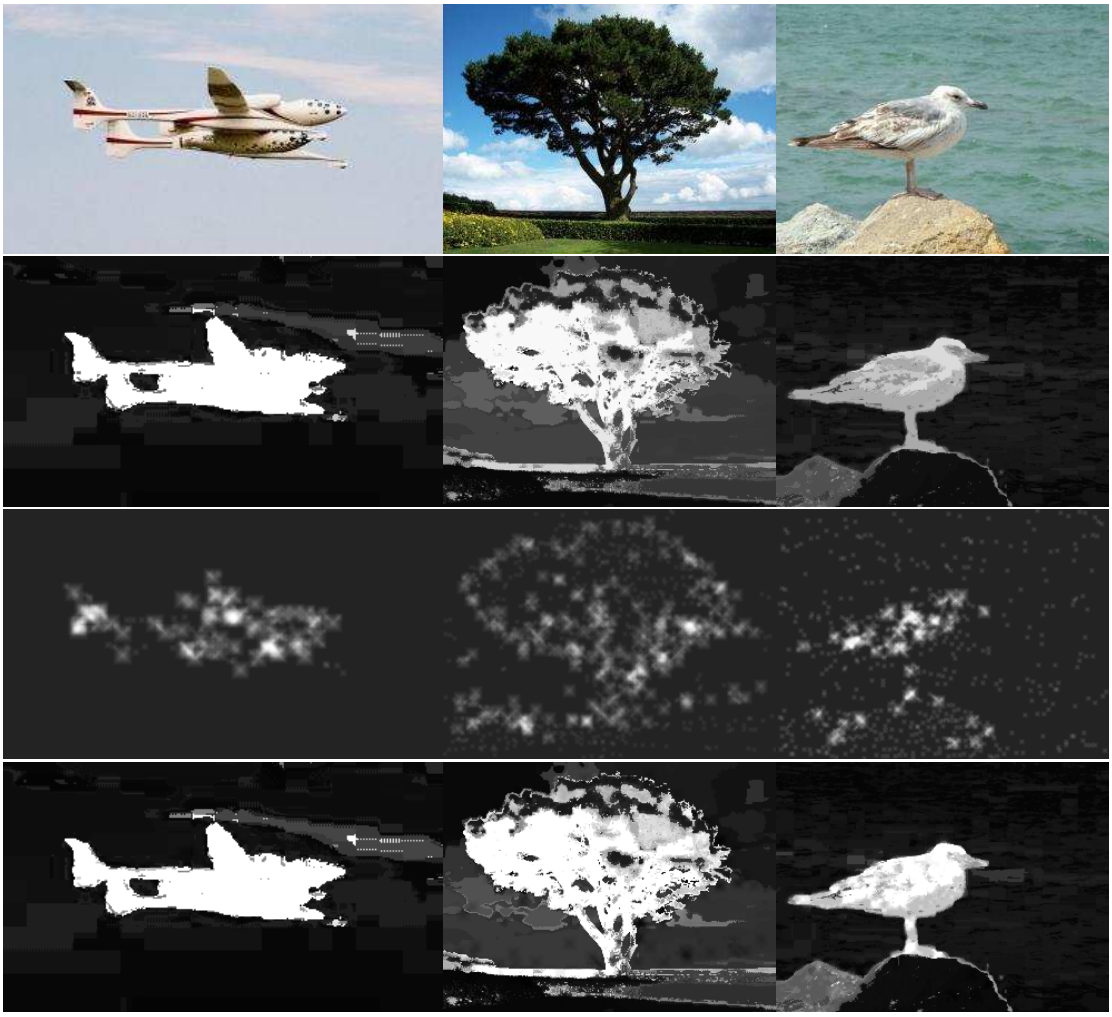


Figure 13 Final Saliency Results. First Row: Original images. Second Row: Combination maps. Third Row: Keypoint maps. Fourth Row: Saliency maps.

CHAPTER 4

EXPERIMENTS

I here perform multiple experiments to show the effectiveness of my method. In section 4.1 I define several variations of my full system and show comparison results of all the systems. I then compare my method with two state-of-the-art methods in section 4.2.

4.1 Variant Systems

In order to show the usefulness of each stage of my method, I first define four variant systems, in sections 4.1.1 through 4.1.4, with selected steps removed from the full system. Then full system is shown in section 4.1.5. Further the results of the variant systems are visually compared with the results of the full system in section 4.1.6 and quantitatively compared in section 4.1.7.

4.1.1 Variant System 1 (Baseline)

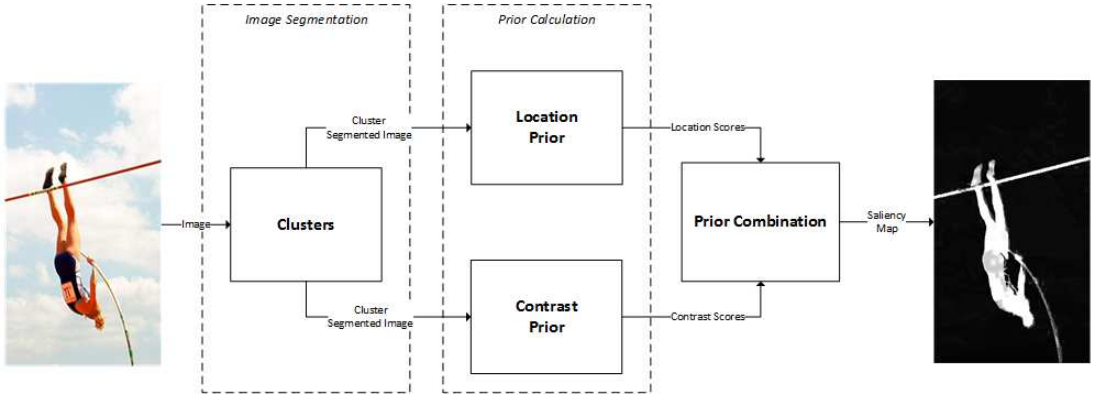


Figure 14 Block Diagram of System 1

The block diagram of the first variant system is demonstrated by Figure 14. Both the pre and post processing steps are removed. A single point replaces the rectangular center for center bias calculations and subclusters are not used. This systems serves as a good baseline as it is very similar to the system in [8], yet still improved. Sample map results for the initial system are shown in the first row of Figure 17-Figure 19.

4.1.2 Variant System 2 (Baseline + Preprocessing)

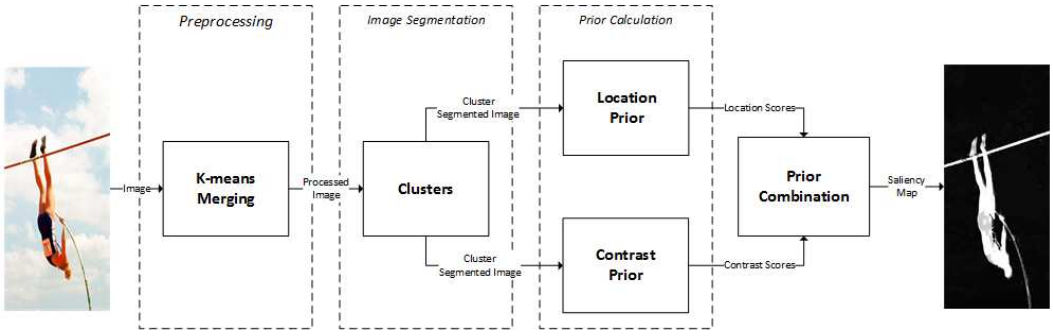


Figure 15 Block Diagram of Systems 2 and 3

For variant system 2, k-means merging preprocessing is added to variant system 1 as illustrated in Figure 15. Map results for this system are shown in the second row of Figure 17-Figure 19.

4.1.3 Variant System 3 (Variant System 2 + Rectangular Centers)

For variant system 3, the block diagram is the same as in Figure 15. However, rectangular centers are again employed to the system instead of the conventional center approach. Sample map results for this system are shown in the third row of Figure 17-Figure 19.

4.1.4 Variant System 4 (Variant System 3 + Subclusters)

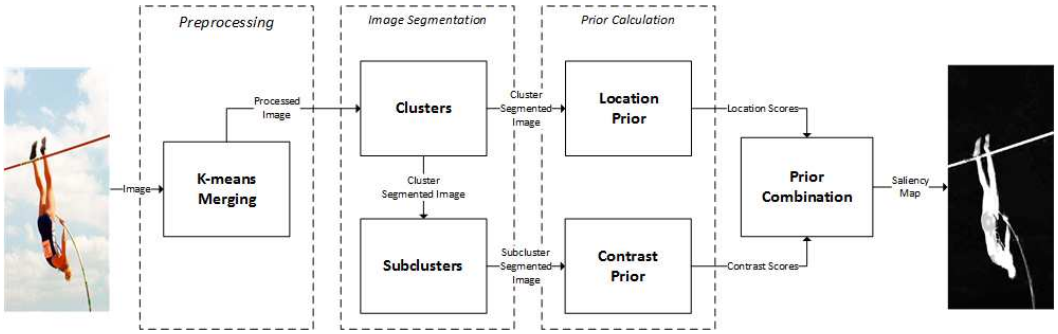


Figure 16 Block Diagram of System 4

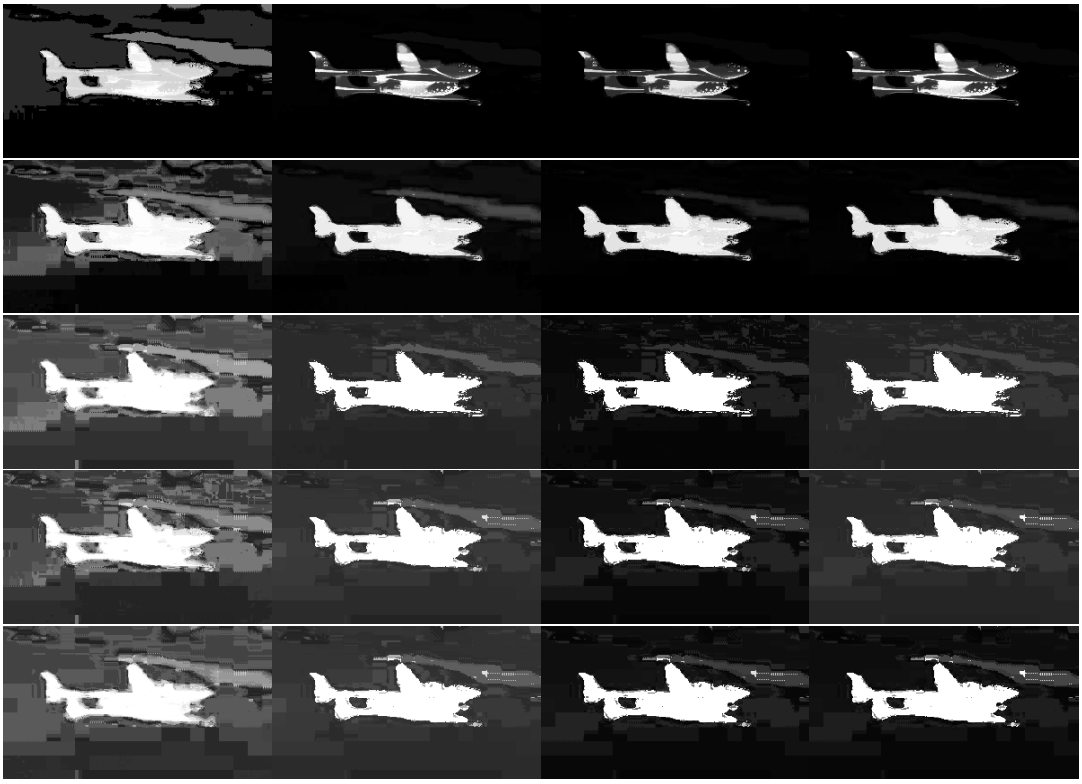
For variant system 4, I add the subclustering process as shown in Figure 16. Sample map results for this system are shown in the fourth row of Figure 17-Figure 19.

4.1.5 Full System (Variant System 4 + Postprocessing)

Finally, the full system is again generated by adding postprocessing with keypoint separation to variant system 4. This system was illustrated earlier in Figure 3. Sample map results for this system are shown in the last row of Figure 17-Figure 19.

4.1.6 Qualitative Results

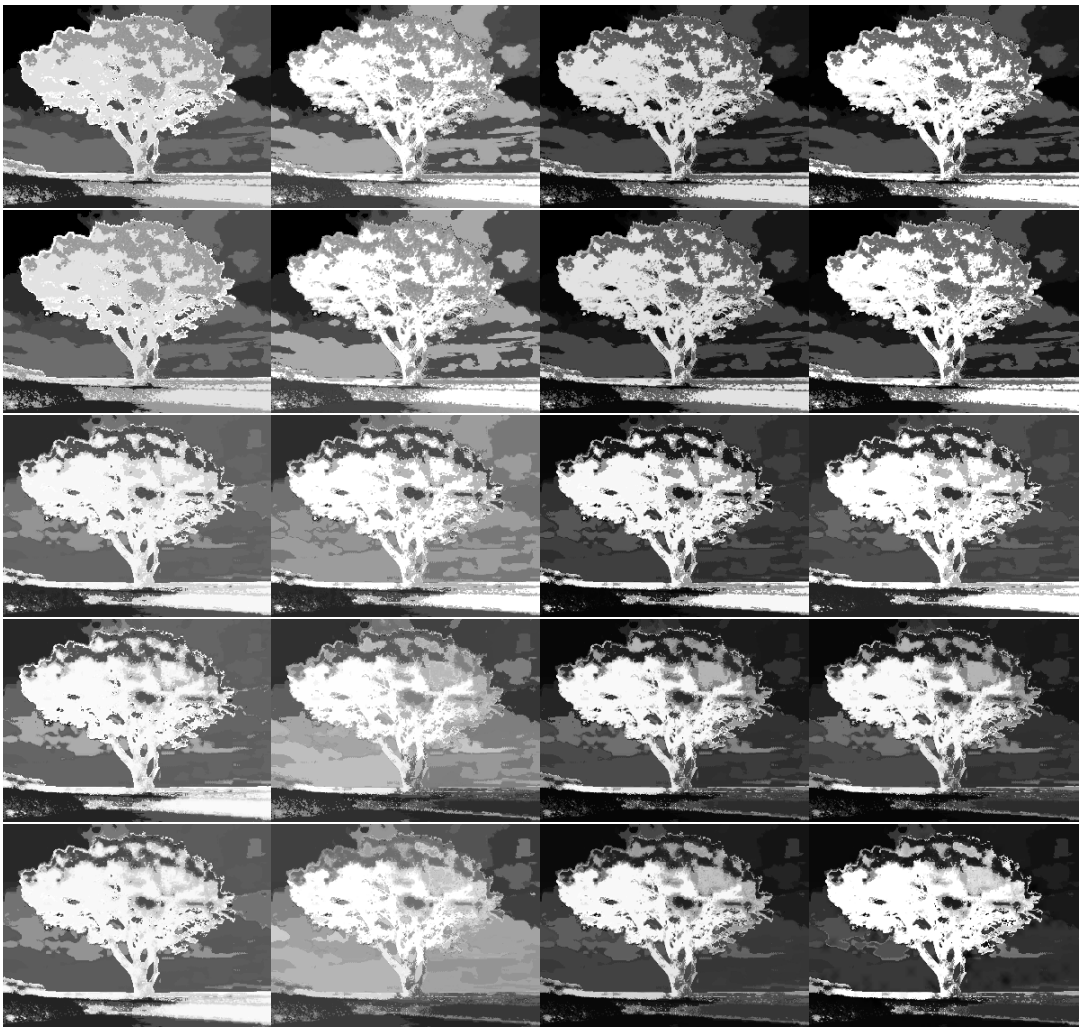
In this section, I show results for the systems discussed in sections 4.1.1-4.1.5 for three sample images. Figure 17 presents results of the four variant systems and the full system for the plane image. The most notable difference occurs between the contrast maps of variant systems 1 and 2. The preprocessing stage merges the regions of the plane into essentially one cluster. Thus, the entire plane is then calculated to be salient with both location and contrast priors.



a) Location Map b) Contrast Map c) Combined Map d) Saliency Map

Figure 17 Comparison of Variant System Results for the Airplane Image. First Row: Variant system 1 results. Second Row: Variant system 2 results. Third Row: Variant system 3 results. Fourth Row: Variant system 4 results. Fifth Row: Full system results.

In Figure 18, I show results of the five systems for the tree image. The subclustering process has the most important effect on the tree saliency map. As exhibited between the saliency maps of variant systems 3 and 4, the salient tree has been separated from the ground vegetation.



a) Location Map b) Contrast Map c) Combined Map d) Saliency Map

Figure 18 System Results for Tree Image. First Row: Baseline systems results. Second Row: Preprocessing results. Third Row: Rectangular center results. Fourth Row: Subclustering results. Fifth Row: Postprocessing results.

Results of the five systems for the bird image are displayed in Figure 19. The fourth row shows subclustering unlinks the bird from the small rock. However, the subclustering process also deemphasized some parts of the bird. Postprocessing, though, helped to again highlight most of the bird (fifth row).

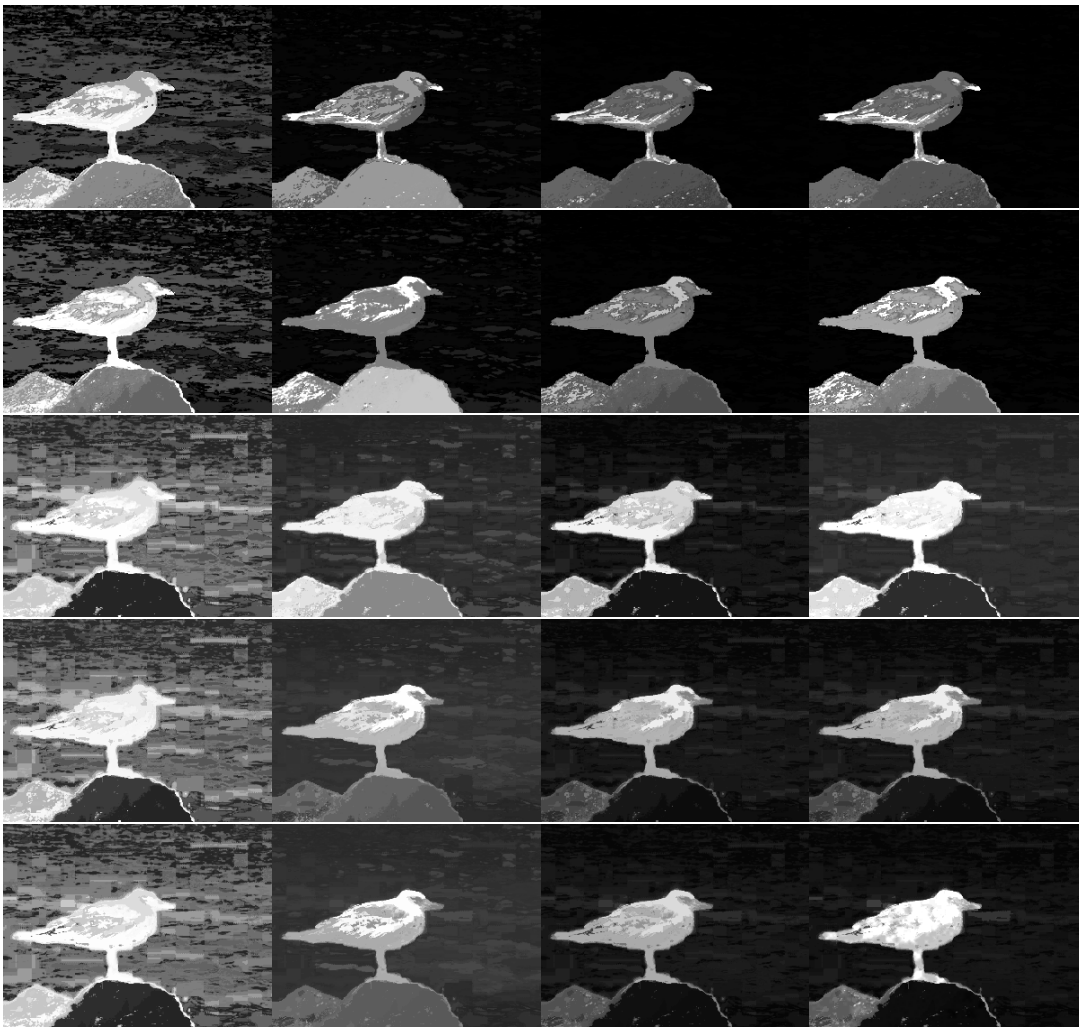


Figure 19 System Results for Bird Image. First Row: Baseline system results. Second Row: Preprocessing results. Third Row: Rectangular center results. Fourth Row: Subclustering results. Fifth Row: Postprocessing results.

4.1.7 Quantitative Results

Lastly, I quantitatively compare the four variant systems with the full system. Here I use Precision/Recall (PR) curves and F-number to evaluate all five systems. It should be noted that Receiver Operating Characteristic (ROC) curves measure similar features to PR curves and are common in other research fields. However, PR curves are more commonly employed in saliency detection. The algorithm to generate a PR curve is shown in Figure 20.

Algorithm 2

Input: Saliency Maps M_n and Ground Truth G_n where n is the number of images

1: For each threshold t such that

$$t \in \{0, 0.02 \dots 0.98, 1\}$$

1.1: For each saliency map M create a threshold map M_t as follows:

$$M_{n,t}(p) = \begin{cases} 1 & p \geq t \\ 0 & p < t \end{cases}$$

where p is a pixel in M and 1 represents salient.

1.2: Calculate precision $p_{n,t}^r$ and recall $r_{n,t}^e$ for the nth image at threshold t by:

$$p_{n,t}^r = \frac{\sum\{tp\}_{n,t}}{\sum\{tp\}_{n,t} + \sum\{fp\}_{n,t}}$$

$$r_{n,t}^e = \frac{\sum\{tp\}_{n,t}}{\sum\{tp\}_{n,t} + \sum\{fn\}_{n,t}}$$

where tp (true positive), fp (false positive), and fn (false negative) are defined for a given pixel as:

$$\begin{cases} G_n(p) = 1 \; and \; M_{n,t}(p) = 1 & tp \\ G_n(p) = 0 \; and \; M_{n,t}(p) = 1 & fp \\ G_n(p) = 1 \; and \; M_{n,t}(p) = 0 & fn \end{cases}$$

1.3 Precision p_t^r and recall r_t^e for t are then calculated by:

$$p_t^r = \frac{\sum_{i=1}^n p_{i,t}^r}{n}$$

$$r_t^e = \frac{\sum_{i=1}^n r_{i,t}^e}{n}$$

2: Create a precision/recall curve by plotting a curve through all points (p_t^r, r_t^e) created in step 1.

Output: Precision/Recall Curve

Figure 20 Algorithmic View of Generating a Precision/Recall Curve

Figure 21 shows PR curves for each of the five systems discussed in the previous sections. As is presented, there is consistent improvement from each variant system to

another since the area covered by the respective curve is increased with each variant system.

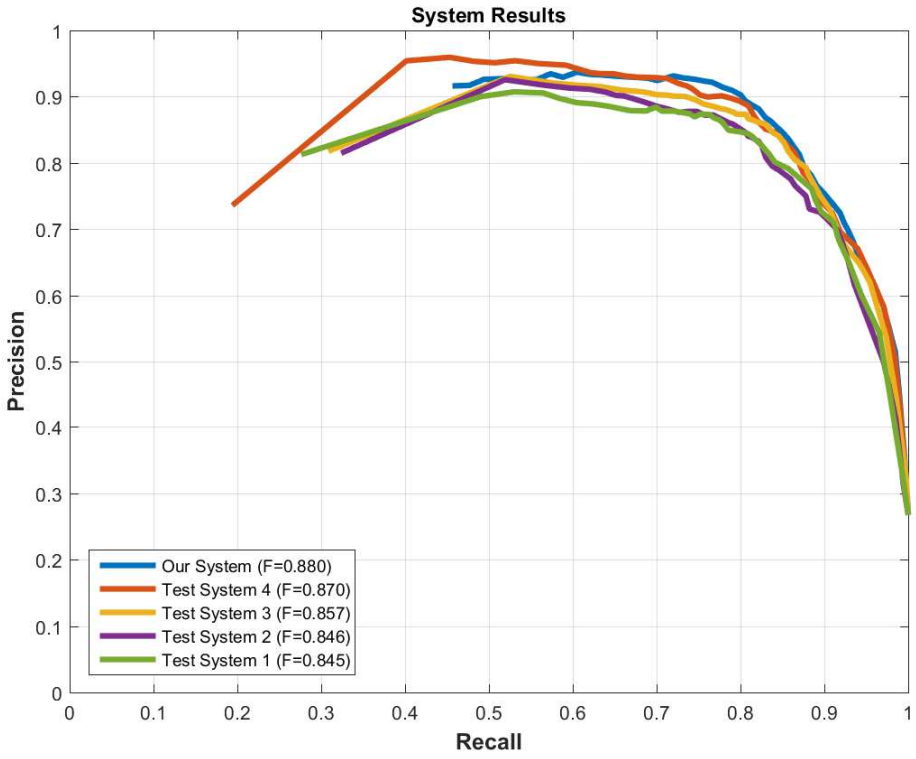


Figure 21 Comparison of PR Curves for the Five Systems

I also explore another measurement commonly used in the literature, the F-number, to compare the various systems. This quantity combines precision and recall into a single

quantity for a system. The higher the F-number, the better the performance. The general equation is:

$$F_\beta = \frac{(1+\beta^2)*precision*recall}{\beta^2*precision+recall} \quad (16)$$

A common practice in many fields is to set $\beta^2 = 1$ to create an F_1 number that weights precision and recall evenly. However, a recent trend in computer vision research is setting $\beta^2 = .3$ so as to weigh precision above recall. I apply this practice here and the result for each system is summarized in the legend of Figure 21. This value shows consistent progress from one variant system to another. Additionally, the largest improvements occur between variant systems 2 and 3 and variant systems 3 and 4. These larger increases indicate significant performance improvements occurred when rectangular centers and the subclustering process were added.

4.2 State-of-the-art Comparisons

I now compare my method with other state-of-the-art algorithms. Subsequently I compare my approach with two other methods described previously in section 2.2 ([16], [8]). These methods were chosen because, along with being unique, their code was publicly available. The first is based on cell automata (CA) and the latter is a cluster-based (CB) approach similar to my own.

In order to compare the three methods I will use three publicly available image databases - SED1, SED2, and ASD. The SED1 database, as described previously, is a collection of 100 images with 1 salient object per image, SED2 contains 100 images with 2 salient regions per image, and the ASD database is a group of 1000 images with

varying number of salient regions. The ASD database has the advantages of being larger and more diverse. The SED databases have the advantage of being annotated by multiple people, making them more reliable. It is important to note that the ASD database contains some images with frames that interfere with prior calculations. To solve this problem my method uses the same automatic frame removal technique as employed by CB.

4.2.1 Qualitative Results

Figure 22 shows a visual comparison among the methods. The results indicate CA typically does the best job of suppressing background noise, though it struggles with edges and often includes lots of gray (e.g., less salient portions) within salient objects. CB preserves distinct objects better; however, it often includes a large amount of non-salient pixels along with the salient object. My method shows an excellent amount of edge integrity, while clearly isolating salient objects in a number of different settings.

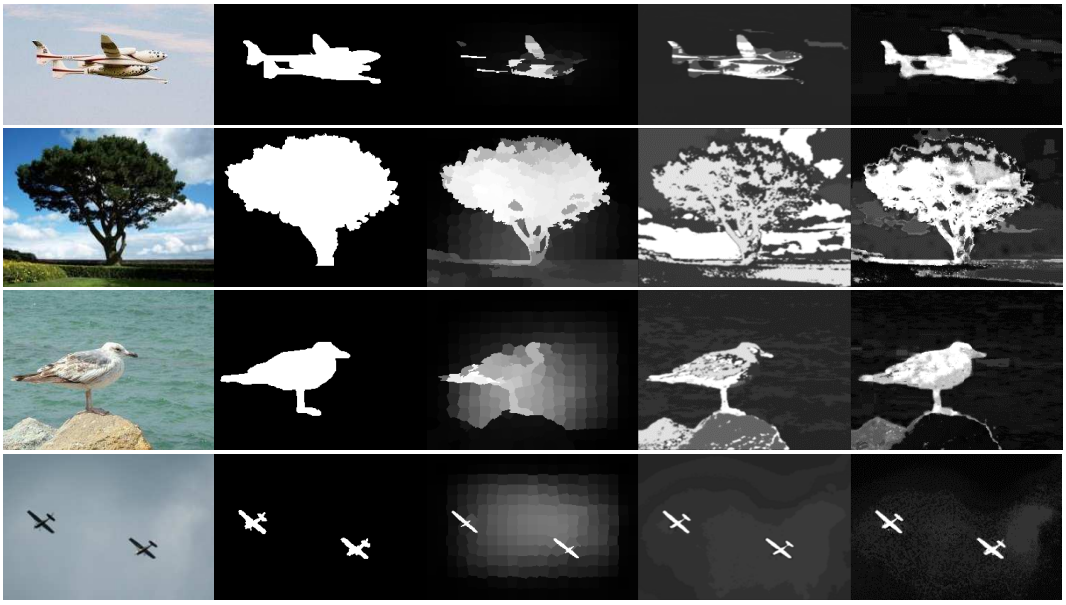


Figure 22 Comparison of Three Algorithms. The first three rows are from the SED1 database. The middle rows are SED2 images. The final three rows come from ASD.

4.2.2 Quantitative Results

I also compute quantitative measures (e.g., PR curve F-number, and Standard Deviation) to compare the performance of my algorithm with two other state-of-the-art algorithms. Figure 23 plots precision/recall curves for each of the three methods on the

SED1 database, and Table 1 compares standard deviation of F-numbers among the algorithms at multiple segmentation thresholds. Clearly CB falls short of the two other methods in all three categories. Specifically, the area covered by the curve from the CB method is the smallest and the F-number of 0.756 is the lowest. The CA method significantly improves over the CB method by covering a bigger area under its curve and achieving a higher F-number of 0.855. My method performs best as the area covered by its curve is the greatest, and the F-number is the largest (0.880). It should be mentioned that the baseline system also achieves better performance than CB with an F-number of 0.845, and variant system 3 achieves better results than CA with an F-number of 0.857.

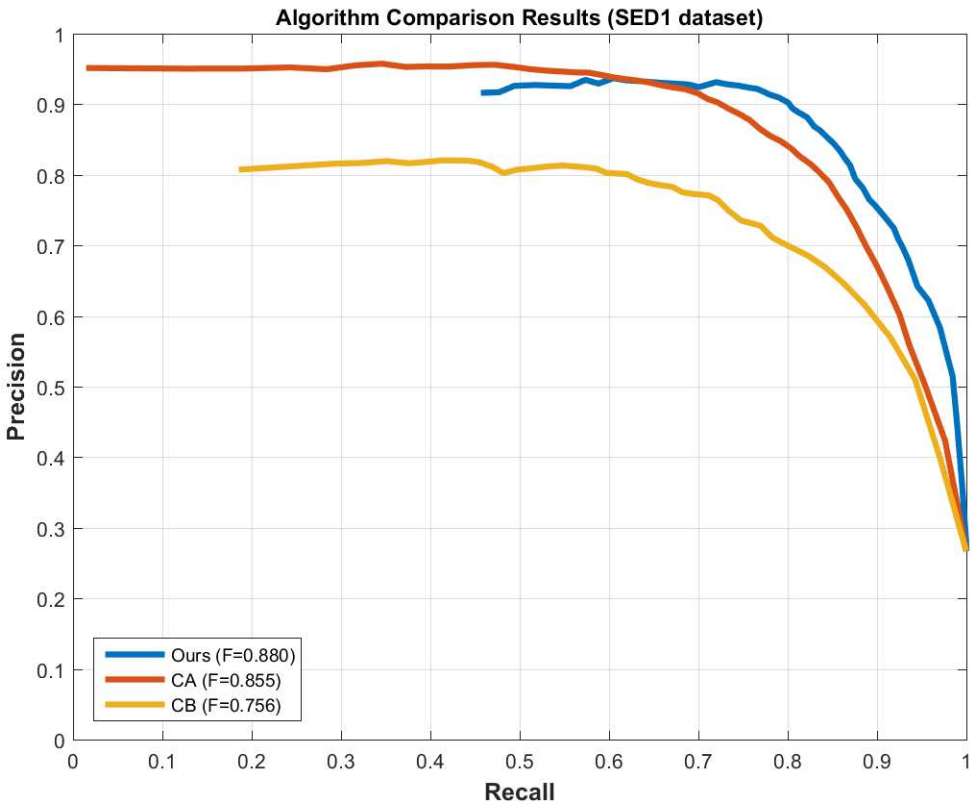


Figure 23 Precision/Recall Curve Comparison for the SED1 Database

Table 1 Standard Deviation Comparison of F-Numbers for the SED1 Database

	Threshold									
Algorithm	.1	.2	.3	.4	.5	.6	.7	.8	.9	1
Mine	0.264	0.254	0.214	0.193	0.176	0.180	0.204	0.232	0.255	0.269
CA	0.240	0.208	0.183	0.153	0.151	0.179	0.205	0.217	0.240	0.050
CB	0.158	0.256	0.243	0.215	0.222	0.225	0.220	0.231	0.262	0.281

Results from using the SED2 database, shown in Figure 24 and Table 2, highlight other differences among the algorithms. For this database with two salient objects, the CA method performs worst with the smallest area under the curve, and the lowest F-number of 0.775; however, again it generally has the lowest standard deviation measures. CB achieves better results with a larger area under the curve, and higher F-number of 0.796. My method is again superior with the largest area under the curve and the highest F-number at 0.804. Though, my method also has slightly higher standard deviation measures. These results suggest CA struggles more with multiple salient objects, and my method works great with single and multiple salient objects.

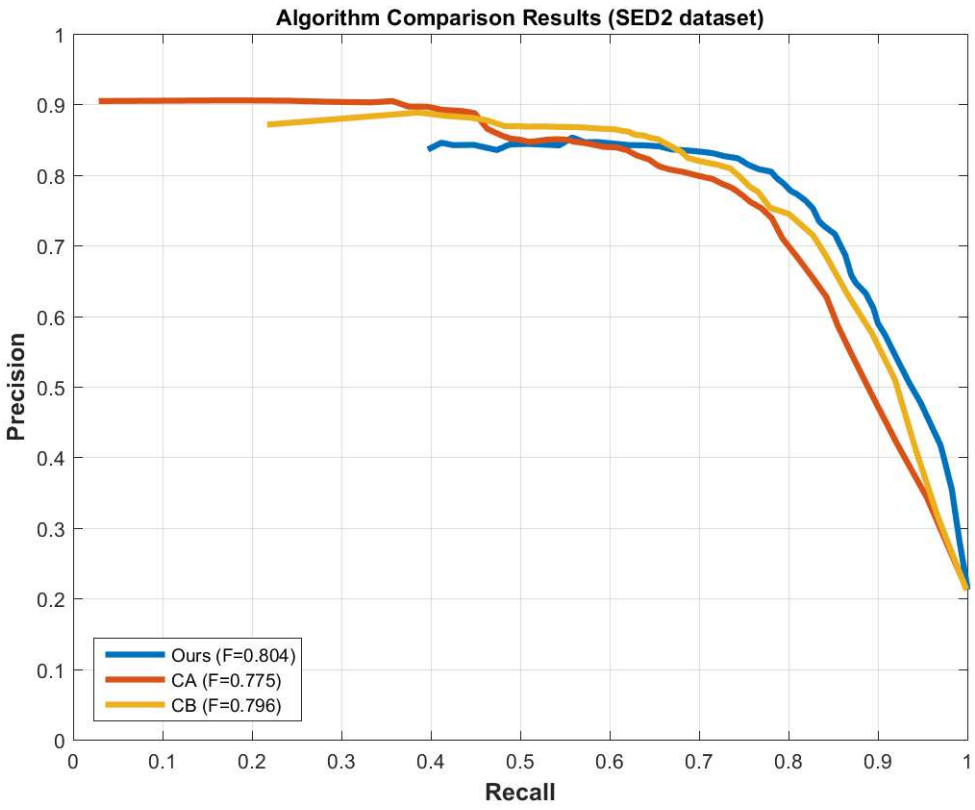


Figure 24 Precision/Recall Curve Comparison for the SED2 Database

Table 2 Standard Deviation Comparison of F-Numbers for the SED2 Database

	Threshold									
Algorithm	.1	.2	.3	.4	.5	.6	.7	.8	.9	1
Mine	0.304	0.312	0.296	0.279	0.279	0.267	0.278	0.302	0.323	0.334
CA	0.274	0.283	0.287	0.279	0.269	0.271	0.279	0.279	0.292	0.105
CB	0.198	0.298	0.256	0.249	0.261	0.264	0.272	0.282	0.288	0.309

The ASD database provides some peculiar results (shown in Figure 25 and Table 3). CB again performs worst with the smallest area under the curve and lowest F-number of 0.849. However the CA method has a slightly larger area under the curve than my method, and it also achieves a better F-number at 0.907 to my 0.883. Further research would be needed to understand the variations among the three databases. Fortunately my

method performs well with all three databases and in most cases is superior to the other two state-of-the-art algorithms.

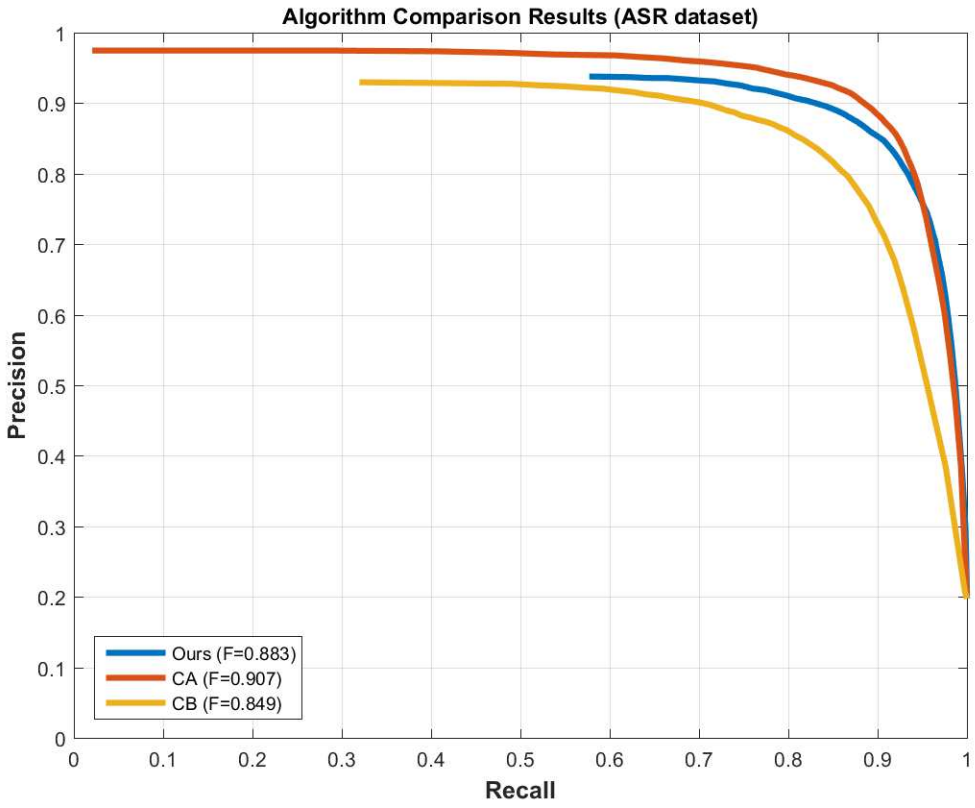


Figure 25 Precision/Recall Curve Comparison for the ASD Database

Table 3 Standard Deviation Comparison of F-Numbers for the ASD Database

	Threshold									
Algorithm	.1	.2	.3	.4	.5	.6	.7	.8	.9	1
Mine	0.280	0.261	0.239	0.214	0.189	0.178	0.176	0.182	0.203	0.235
CA	0.255	0.224	0.199	0.177	0.161	0.155	0.159	0.172	0.217	0.060
CB	0.108	0.263	0.230	0.209	0.202	0.196	0.200	0.207	0.220	0.258

CHAPTER 5

CONCLUSION

In this paper I introduced the topic of Salient Object Detection (SOD). I then identified a few applications of the topic and also pointed out some of the challenges. I further discussed the characteristics of current and past algorithms.

I next presented a new approach to SOD that includes not only the three general steps of image segmentation, prior calculation, and prior combination, but also includes novel concepts that improve upon state-of-the-art methods. My contributions include:

1) preprocessing using k-means merging,
2) performing contrast calculations on the subcluster level,
3) defining a rectangular image center, and
4) postprocessing using keypoint separation.

Lastly, I compared my proposed approach with two state-of-the-art methods, namely cellular automata and cluster-based. My extensive experimental results on the SED1, SED2, and ASD databases demonstrated the effectiveness of my innovative concepts. Further results showed the new method achieves better results for many images and similar results for other images compared to the two state-of-the-art methods. Overall, my method outperforms the other two systems in terms of PR curve and F-number. It also outperforms its four variant systems for the SED1 database.

Future work includes exploring alternate preprocessing techniques, relying less on center bias, and incorporating color distribution. The major drawback to the k-means method is that its results are slightly random. Thus, a method less reliant on k-means would help provide more consistent results. Though humans often focus more towards the center of an image, there is no requirement that a salient object be in the center. Therefore a method less reliant on the center would be helpful for more applications. One way to use center bias less is to incorporate color distribution. Consequently, the next step would be to add the color distribution prior to the method discussed in this thesis.

REFERENCES

[1] Z. Ren, S. Gao, L.-T. Chia and I. W.-H. Tsang, "Region-Based Saliency Detection and Its Application in Object Recognition," *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 24, no. 5, pp. 769-779, May 2014.

[2] K. Anitha and P. Leveenbose, "Edge Detection based Salient Region Detection for Accurate Image Forgery Detection," in *IEEE International Conference on Computational Intelligence and Computing Research*, 2014.

[3] K. Fu, I. Y. Gu and A. Odblom, "Traffic Sign Recognition using Salient Region Features: A Novel Learning-based Coarse-to-Fine Scheme," in *IEEE Intelligent Vehicles Symposium*, Seoul, Korea, 2015.

[4] L. Itti, C. Koch and E. Niebur, "A Model of Saliency-Based Visual Attention for Rapid Scene Analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 20, no. 11, pp. 1254-1259, November 1998.

[5] X. Hou, S. J. Tong and L. Zhang, "Saliency Detection: A Spectral Residual Approach," in *IEEE Conference on Computer Vision and Pattern Recognition*, Minneapolis, MN, 2007.

[6] L. Zhou and Z. Yang, "Salient Region Detection based on Spatial and Background Priors," in *IEEE International Conference on Information and Automation*, Hailar, China, 2014.

[7] A. Singh, C.-H. H. Chu and M. A. Pratt, "Multiresolution superpixels for visual saliency detection," in *IEEE Symposium on Computational Intelligence for Multimedia, Signal and Vision Processing*, Orlando, FL, 2014.

[8] H. Fu, X. Cao and Z. Tu, "Cluster-Based Co-Saliency Detection," *IEEE Transactions on Image Processing,* vol. 22, no. 10, pp. 3766-3778, October 2013.

[9] L. Zhou, Y. Li, Y. Song, Y. Qiao and J. Yang, "Saliency Driven Clustering for Salient Object Detection," in *IEEE International Conference on Acoustic, Speach and Signal Processing*, 2014.

[10] R. Manke, S. C. Raikwar and A. S. Jalal, "A Robust Approach for Salient Region Detection," in *9th International Conference on Industrial and Information Systems*, Gwalior, 2014.

[11] N. Tong, H. Lu, L. Zhang and X. Ruan, "Saliency Detection with Multi-Scale Superpixels," *IEEE Signal Processing Letters,* vol. 21, no. 9, pp. 1035-1039, September 2014.

[12] M.-M. Cheng, N. J. Mitra, X. Huang, P. H. Torr and S.-M. Hu, "Global Contrast Based Salient Region Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 37, no. 3, pp. 569-582, March 2015.

[13] R. Kannan, G. Ghinea and S. Swaminathan, "Salient Region Detection Using Patch Level and," *IEEE Signal Processing Letters,* vol. 22, no. 6, pp. 686-690, June 2015.

[14] P. Manipoonchelvi and K. Muneeswaren, "Region-based saliency detection," *IET Image Processing,* vol. 8, no. 9, pp. 519-527, 2014.

[15] L. Kong, L. Duan, W. Yang and Y. Dou, "Salient region detection: an integration approach," *IET Computer Vision,* vol. 9, no. 1, pp. 85-97, 2015.

[16] Y. Qin, H. Lu, Y. Xu and H. Wang, "Saliency Detection via Cellular Automata," in *IEEE Conference on Computer Vision and Pattern Recognition*, Boston, MA, 2015.

[17] Y. Li, K. Fu, L. Zhou, Y. Qiao and J. Yang, "Saliency detection via foreground rendering and background exclusion," in *IEEE International Conference on Image Processing (ICIP)*, Paris, France, 2014.

[18] G. Zhang, Z. Yuan, Y. Liu and N. Zheng, "Salient object detection based on multi-scale super-pixels and saliency propagation," *Electronic Letters,* vol. 50, no. 9, pp. 675-677, 24 April 2014.