Main subject detection via adaptive feature refinement

Cuong Vu
Damon Chandler
Oklahoma State University
Laboratory of Computational Perception and Image Quality
School of Electrical and Computer Engineering
Stillwater, Oklahoma 74078
E-mail: cuong.vu@okstate.edu

Abstract. Main subject detection (MSD) refers to the task of determining which spatial regions in an image correspond to the most visually relevant or scene-defining object(s) for general viewing purposes. This task, while trivial for a human, remains extremely challenging for a computer. Here, we present an algorithm for MSD which operates by adaptively refining low-level features. The algorithm computes, in a block-based fashion, five feature maps corresponding to lightness distance, color distance, contrast, local sharpness, and edge strength. These feature maps are adaptively combined and gradually refined via three stages. The final combination of the refined feature maps produces an estimate of the main subject’s location. We tested the proposed algorithm on two extensive image databases. Our results show that relatively simple, low-level features, when used in an adaptive and iterative fashion, can be very effective at MSD. © 2011 SPIE and IS&T. [DOI: 10.1117/1.3549884]

1 Introduction

Most photographers convey their ideas via one or more main subjects in their photos. Even an amateur camera operator almost always has a main subject in mind when taking a picture. Given an image, main subject detection (MSD) refers to the process of locating the image’s main subject. In terms of image-processing, the ability to automatically locate the main subject can be beneficial for analysis, coding, and general processing. For example, MSD is almost always used as a first step in autocropping algorithms.1–4 In applications such as image compression and unequal error protection, the ability to find the main subject would allow the system to devote more bits to the data in and around the main subject.5–7 Studies have also shown that MSD can be useful for image quality assessment (e.g., Refs. 8–10) and as a first step in object recognition (e.g., Refs. 11–13).

For a human, locating the main subject in a photo usually requires little effort. Even without color, and even if the scene is devoid of commonplace subject matter (e.g., artistic images), MSD is often effortless for humans. This ability may stem from the fact that photographers usually make the main subject stand out by adjusting the focus and using established rules that lead viewers toward the main subject. In addition, from an evolutionary standpoint, species must be able to quickly locate regions of interest in order to avoid predation and/or to find food. MSD may therefore be an inherent trait which is necessary for survival.

Yet, despite the ease with which a human can perform MSD, this task has proven to be quite challenging for a computer. Current research efforts in MSD generally fall into one of two categories: 1. MSD via visual attention prediction, and 2. direct MSD. In visual attention algorithms (e.g., Refs. 14–21), the goal is to locate the regions in the image upon which humans visually fixate. These predicted fixations can then be used to locate the main subject under the assumption that the main subject receives the greatest number/density of fixations (see, e.g., Ref. 16). Evidence in support of this assumption has recently been published.22, 23 In Ref. 23, Wang et al. collected both eye-tracking data and human ratings of visual importance for a large collection of images. Their results revealed that the main subject in each image always received the greatest number of fixations.

A different class of algorithms has been developed which does not attempt to predict visual fixations but instead addresses the task of MSD more directly (e.g., Refs. 24–32). Algorithms of this latter type generally attempt to locate the main subject via feature extraction followed by feature fusion. Research efforts have traditionally been geared toward either finding better features or finding better models to combine the features, particularly those which can take into account statistical dependencies among features. Indeed, many existing MSD algorithms propose new features (e.g., symmetry, face detection).27 Although such features can be effective, they also usually impose a significant computational cost. Luo et al.27 conducted a study in which the effectiveness of a wide variety of features was tested on a large collection of photographs for which the main subjects were labeled by human observers. They found that the three features of centrality, borderness, and skin color were most effective for MSD when computational complexity was also considered. In addition, many researchers have investigated the use of statistical learning techniques for combining features. Popular methods such as Conditional Random Fields,33 Hidden Markov Models,34, 35 and Bayesian Networks36 have gained recent popularity due to their ability to take into account higher-order statistical dependencies among features. Most of these learning-based approaches do not significantly increase the algorithm’s computational complexity; however,
they do require extensive training on a proper set of ground truth data in order to achieve results that can generalize.

In this paper, we argue that relatively simple features can be effective for MSD if one can adaptively choose which features to use and which features not to use on a per-image basis. A common assumption when combining features (e.g., using Conditional-Random-Field-derived weights) is that the features all contribute to various degrees toward detecting the main subject. However, some features which contribute greatly on some images may actually be detrimental to the detection process on other images. Figure 1 demonstrates this point. In the first row, the main subject is the flower which is distinctive from the background but slightly blurred. In this case, using color will help in locating the main subject, whereas using sharpness will hurt the detection process. In the second row, the main subject is the black 8-ball, while all other balls (the colored balls) belong to the background. For this image, color is now the harmful feature, whereas lightness is a better feature for locating the black 8-ball as the main subject. Yet, in the last row, the main subject (the tree) can be well detected using sharpness, whereas lightness is now the harmful feature. See the fifth paragraph of Sec. 1 for a brief description of this process; see Sec. 3 for details of the algorithm. (Color online only.)

We further argue that MSD does not have to end after the features are combined, but can instead be refined through the use of multiple stages. Namely, the results from an initial application of an MSD algorithm can represent a first guess of the main subject’s location, and this information can then be used to refine the features computed in subsequent iterations of the algorithm. The first row of Fig. 2 illustrates this technique using a feature termed “lightness distance.” The lightness distance is defined as the Euclidean distance between the average lightness of a foreground object and the average lightness of the background. Since the delineation between the foreground and the background is unknown beforehand, the first iteration of the algorithm computes the lightness distance as the Euclidean distance between the average lightness of each image block and the average lightness of the entire image. As shown in Fig. 2(b), the resulting feature map is imperfect since the distance from the lightness of the main subject (the car) to the average lightness of the entire image is about the same as the lightness distance of the snow in the background. However, since we now have a guess of the main subject’s location [denoted by the rectangle in Fig. 2(c)], the feature map can be refined by considering the area outside of the rectangle to be the background and therefore measuring lightness distance to this new background. As shown in Fig. 2(d), the main subject in the refined feature map now stands out. A similar improvement from the baseline feature map to the refined feature map can be seen in the second row for a feature termed “color distance.” In this case, the baseline feature map is computed as the distance between the color (chromaticity) of each block to the average color
of the entire image. Again, this initial feature map [Fig. 2(f)] provides an initial guess of the background [region outside the rectangle in Fig. 2(g)], and this initial guess can then be used to generate a refined feature map by measuring color distance to the new background [Fig. 2(h)]. As these examples demonstrate, multiple stages of refinement can improve the feature extraction process and can thus improve MSD.

In this paper, we present an MSD algorithm which operates using the techniques described in the preceding examples, which we call adaptive feature refinement. Our algorithm operates by first computing five relatively simple feature maps (lightness distance, color distance, contrast, sharpness, and edge strength). These feature maps are then combined via a weighted sum where the weights are selected in an image-adaptive fashion. The resulting combined map is then used to estimate the location of the main subject. Next, this initial estimate of the main subject is used to modify the feature maps, which are, in turn, used to generate a new estimate of the main subject’s location. The modification process is repeated one final time to arrive at the final estimate of the main subject. As we demonstrate, this adaptive, multistage approach can be quite effective at MSD.

This paper is organized as follows. Section 2 provides an overview of current methods of MSD which operate based on either computational models of visual attention or more direct approaches. In Sec. 3 we describe our algorithm. The performance of our algorithm on two separate ground-truth databases is analyzed and discussed in Sec. 4. General conclusions are provided in Sec. 5.

2 Background

Modern methods of main subject detection can generally be classified into two main trends: 1. those which attempt to predict visual attention maps and then use those maps to perform MSD; and 2. those which directly attempt to find the main subject in the image. This section provides an overview of these methods.

2.1 Methods Based on Visual Attention Prediction

Researchers have developed various models for predicting where in an image a human observer’s attention will be drawn. Such visual attention maps are typically compared to eye-tracking data. Although visual attention is not specifically designed for MSD, the maps can also potentially be used to locate main subjects, under the assumption that the main subject will receive the most attention.

Itti et al.14 introduced a model of visual saliency (defined as the extent to which a region stands out from its background in terms of one or more dimensions such as intensity or orientation.) based on intensity, color, and orientation, at different scales and orientations. Harel et al.15 used a graph-based approach and Frintrop et al.17 applied integral images to extend the work of Itti et al.14. Le Meur et al.16 built an algorithm for detection of bottom-up visual attention by modeling three aspects of visual processing: visibility, perception, and perceptual grouping. Bruce and Tsotsos18 developed a model based on maximizing information sampled from a scene by quantifying the self-information of each local image patch. Gao and Vasconcelos19 found the saliency of a given location by discriminating between the visual appearance of the center of the location and the surrounding region. Kadir and Brady20 and Kadir et al.21 introduced a saliency detector which automatically selects appropriate scales for analysis and highlights a salient region if it shows unpredictability in both its attributes and spatial scale.

2.2 Direct Methods for Main Subject Detection

A different class of algorithms has been developed which attempts to address the task of MSD more directly. Ma and Zhang24 used a fuzzy growing method to extract the saliency map based on local contrast analysis. Hou and Zhang25 developed an algorithm which analyzed the spectrum of the image and then constructed a saliency map from the extracted spectral residual of the image. Wang and Li26 extended the work of Hou and Zhang25 in two stages. In the first stage,
they introduced an automatic channel-selection module and a decision-reversal module. In the second stage, they modified the potentially incomplete salient regions based on region similarity and proximity.

Luo et al.\textsuperscript{27} were among the first researchers who tried to bring top-down cues to MSD. Features extracted in their algorithm consist of both structural features, which are bottom-up cues including color, texture, size, shape, etc.; and semantic features, which are top-down cues including semantic object classes such as sky, grass, skin, face, etc. A saliency map is obtained using a Bayesian-network-based approach to combine the features.

Achanta et al.\textsuperscript{28} used luminance and color to implement a contrast determination filter that operates at various scales to generate saliency maps. In Ref. 29, Achanta et al. introduced a frequency-tuned approach to estimate center-surround contrast. By filtering an image with a difference of Gaussian (DoG) filter, they argued that such filtering will help uniformly highlight whole salient regions with well-defined boundaries. The filtering also serves to remove high frequencies from textures, noise, and blocking artifacts. Their algorithm operated in a frequency-tuned manner via selection of frequency parameters for the DoG filter. Both algorithms\textsuperscript{28,29} produced a clustering map, obtained from either K-means clustering\textsuperscript{37} or mean-shift segmentation,\textsuperscript{38} to their saliency map for main subject segmentation.

Hu et al.\textsuperscript{30} brought the idea of estimating the contribution of each feature map to saliency before combining them. They proposed what they call the composite saliency indicator, which measures the density of saliency points of the feature map and the area of the convex hull polygon of these saliency points. The features used in Ref. 30 are color, intensity, and orientation. Gopalakrishnan et al.\textsuperscript{31} modified the work of Hu et al.\textsuperscript{30} with only two features, color and orientation, arguing that salient regions in most images can be determined by either color or orientation. Gopalakrishnan et al. also propose a different technique of estimating the contribution of each feature. For color, they measured the compactness and isolation of salient clusters in the feature map; for orientation, they used a local orientation histogram. Their saliency map was finally chosen as the feature map which has the highest estimated contribution.

Liu et al.\textsuperscript{32} used three features: multiscale contrast, center-surround histogram, and color spatial distribution, which attempted to capture local, regional, and global aspects of the salient object. Multiscale contrast is the linear combination of contrast in a Gaussian image pyramid. The center-surround histogram measures how distinct a region is with respect to its surroundings in terms of RGB color histograms. The global spatial distribution of each color in the image is also used as a color-based feature. Liu et al. trained a Conditional Random Field to obtain weights for combining the features into a final saliency map. Along with their algorithm, Liu et al. also published the largest ground-truth database for MSD. Further details of this database are described in Sec. 4.

In the following section, we present our algorithm for MSD which operates based on adaptive feature refinement. As we demonstrate, by adaptively combining and iteratively refining the features on a per-image basis, it is possible to be effective at MSD using relatively simple low-level features.

### 3 Algorithm

This section describes the details of our algorithm. First, we describe five low-level features which we use for MSD. Next, we describe a method for adaptively combining the features on a per-image basis based on an estimate of the usefulness of each feature. Finally, we refine the features via two refinement stages in order to obtain a more accurate MSD map. An outline of the algorithm is provided in Fig. 3.

#### 3.1 Features

Viewing MSD as a low-level vision problem, we might choose an object as the main subject because it is in focus; it is different from the background in color, lightness, or contrast; or it has more edge pixels than other regions. Therefore, to perform main subject detection, we measure, for each block in the input image, five low-level features:

1. **Lightness distance:** Measures the difference in lightness/color between a region and its surroundings.
2. **Color distance:** Measures the difference in color between a region and its surroundings.
3. **Contrast:** Measures the difference in contrast between a region and its surroundings.
4. **Sharpness:** Measures the difference in sharpness between a region and its surroundings.
5. **Edge strength:** Measures the difference in edge strength between a region and its surroundings.

In this section, we describe how each feature is computed.

Let $X$ denote the $M_1 \times M_2$-pixel input image. We divide $X$ into blocks of size $m \times m$ pixels with 50% overlap between neighboring blocks. For all feature maps and results presented in this paper, we have used $m = 8$. Let $x$ denote a block of $X$, and let $f_i(x)$ denote the $i$th feature value measured for $x$. From all $f_i(x)$, $x \in X$, we form the $i$th feature map, which we denote as $f_i(X)$.

#### 3.1.1 Lightness and color distance

Lightness/color distance measures how the lightness/color of each region differs from the average lightness/color of the background of the image. [Later when we modify these features, we measure the lightness and color distance to the foreground (see Sec. 3.3.3).] We argue that a region with a large lightness/color distance can be a candidate for the main subject.

Let $B$ denote the guessed background region in the image. In the first stage of the algorithm, we consider the entire image as the background region (see Sec. 3.1.1). Let $f_1(x)$ denote the Euclidean distance between the average lightness of block $x$ and the average lightness of $B$. Let $f_2(x)$ denote the Euclidean distance between the average color of block $x$ and the average color of $B$. These two features are given by:

$$ f_1(x) = |\bar{L}(x) - \bar{L}(B)|, $$

$$ f_2(x) = \sqrt{(\bar{a}(x) - \bar{a}(B))^2 + (\bar{b}(x) - \bar{b}(B))^2}, $$

where $\bar{L}$, $\bar{a}$, $\bar{b}$ denote the average $L^*$, $a^*$, $b^*$ measured in the Commission Internationale de l’Eclairage (CIE) 1976 ($L^*, a^*, b^*$) color space (CIELAB). Let $R'$, $G'$, $B'$ denote the nonlinear RGB values of the image, the conversion from RGB color space to $L^*a^*b^*$ is implemented by first linearizing the $R'$, $G'$, $B'$ values to be proportional to light energy, assuming sRGB color space:

$$ A = \begin{cases} A'/12.92, & A' \leq 0.04045 \\ [(A' + 0.055)/1.055]^{2/3}, & A' > 0.04045 \end{cases} $$

where $A = R$, $G$, or $B$.\textsuperscript{33}
Fig. 3 Outline of the algorithm. Starting with the baseline stage, five features are measured and adaptively combined to obtain the baseline MSD map and then the baseline bounding box around the main subject. This initial bounding box provides the first guess of the main subject and background region. This guess of the background allows the algorithm to modify features, which are again adaptively combined to obtain the refined MSD map and refined bounding box. The new bounding box can be considered as the new foreground and used to modify the features again. These features are combined adaptively into the final MSD map and thus the final bounding box.

The linearized $R, G, B$ values are then converted to the CIE XYZ color space as:

$$X = 0.412453 \times R + 0.357580 \times G + 0.180423 \times B,$$  
$$Y = 0.212671 \times R + 0.715160 \times G + 0.072169 \times B,$$  
$$Z = 0.019334 \times R + 0.119193 \times G + 0.950227 \times B.$$  

Finally the $L^*, a^*, b^*$ values are given by:

$$L^* = 116 \times g(Y/Y_r) - 16,$$  
$$a^* = 500 \times [g(X/X_r) - g(Y/Y_r)],$$  
$$b^* = 200 \times [g(Y/Y_r) - g(Z/Z_r)].$$
where \( X_r = 0.950456, Y_r = 1, Z_r = 1.088754 \) are the CIE XYZ tri-stimulus values of the D65 reference white point; and the function \( g \) is given by:

\[
g(t) = \begin{cases} 
1/3, & t > 0.008856, \\
7.787t + 16/116, & \text{otherwise}.
\end{cases}
\]  
\tag{10}

3.1.2 **Contrast**

Main subjects tend to be of greater contrast than their surrounding regions. Accordingly, we measure root-mean-square (rms) luminance contrast for each block \( x \). A high contrast region is a possible candidate or part of the main subject.

Let \( f_1(x) \) denote the rms luminance contrast of block \( x \). In order to compute \( f_1(x) \), we first convert the image \( X \) into grayscale image \( X_g \):

\[
X_g = 0.299X_1 + 0.587X_2 + 0.114X_3,
\]
\tag{11}

where \( X_1, X_2, X_3 \) denote \( R, G, B \) layers of \( X \), respectively.

Let \( x_0 \) denote the corresponding block of \( x \) in \( X_g \). Let \( I(x) = (b + kx_0)^\gamma \) denote the luminance-valued block, with \( b = 0.7297, k = 0.037644, \) and \( \gamma = 2.2 \) assuming sRGB display conditions. The quantity \( f_1(x) \) is then computed via:

\[
f_1(x) = \left\{ \begin{array}{ll}
\frac{\sigma(x)}{\mu(x)}, & \mu(x) > 0, \\
0, & \mu(x) = 0,
\end{array} \right.
\]
\tag{12}

where \( \sigma(x) \) and \( \mu(x) \) denote the standard deviation and the mean of \( I(x) \), respectively.

3.1.3 **Sharpness**

The main subject in an image is rarely blurred; usually it is in focus and is sharp. In Ref. 40 we describe an algorithm for generating a local sharpness map, which indicates the degree of sharpness for each region in an image. Given an image \( X \), let \( S(X) \) denote the sharpness map of \( X \). We summarize the process of computing \( S(X) \) below; for further details, see Ref. 40.

1. Convert the input image \( X \) to grayscale image \( X_g \) via Eq. (11).
2. Divide \( X_g \) into blocks of size 32\( \times \)32 with 75\% pixels overlap. Let \( x_i \) denote one block.
3. Measure the slope of power spectrum for block \( x_i \), denoted as \( \Delta \). Let \( \tilde{S}_1(x_i) = 1 - 1/[1 + e^{-3(\Delta^2/2)^2}] \); \( S_1(x_i) \) represents the sharpness measure based on the slope of power spectrum of block \( x_i \).
4. From all the \( S_1(x_i) \), we form the sharpness measure based on the slope of power spectrum \( S_1(X) \) for the whole image. We call \( S_1(X) \) the spectral sharpness map.
5. Divide \( X_g \) into blocks of size 8\( \times \)8 with no overlap. Let \( x_i \) denote one block.
6. Let \( S_2(x_i) \) denote the sharpness measure based on local total variation of block \( x_i \), computed via:

\[
S_2(x_i) = \max_{x_0 \in \xi} \nu(\xi),
\]

where \( \xi \) is a 2\( \times \)2 sub-block of \( x_i \), and \( \nu(\xi) = \sum_{i,j} |\xi_i - \xi_j| \), where \( \xi_i, \xi_j \) are pixels in \( \xi \).

7. From all the \( S_2(x_i) \), we form the sharpness measure based on local total variation \( S_2(X) \) for the whole image. We call \( S_2(X) \) the spatial sharpness map.
8. The final sharpness map \( S(X) \) is computed as:

\[
S(X) = S_1(X)^{1/2} \cdot S_2(X)^{1/2}.
\]

From the sharpness map \( S(X) \), we compute the sharpness of block \( x \), denoted as \( f_2(x) \), via averaging values of the sharpness map in each block:

\[
f_2(x) = \mu \left[ S(x) \right] = \frac{1}{m^2} \sum_j s_j,
\]
\tag{13}

where \( s_j \) is a pixel of \( S(x) \).

3.1.4 **Edge strength**

Since edge detection plays an important role in early vision, we argue that a region which contains more edge pixels can be a candidate for the main subject. Thus, we also use edge strength, which denotes the average number of edge pixels in each block, as another feature for locating the main subject.

Let \( f_3(x) \) denote the edge strength of block \( x \). Let \( E(X) \) denote the binary edge image computed by running the Roberts edge detector on \( X \) [since we want to detect only strong edges, Roberts edge detector is a reasonable choice.]: The feature \( f_3(x) \) is then given by:

\[
f_3(x) = \mu_{E(X)} = \frac{1}{m^2} \sum_j e_j,
\]
\tag{14}

where \( E(X) \) is the corresponding block of \( x \) in \( E(X) \) and \( e_j \) is a pixel of \( E(X) \). Basically, the edge strength feature is computed via averaging values of the edge map in each block.

3.1.5 **Center-weight modification and normalization of each feature**

Assuming that we have a guess of the main subject’s location, we can weigh the feature maps across space such that greater weight is applied to this guessed main-subject location. As a first guess, a viewer would likely suspect that the main subject is located near the center of the image; our algorithm also makes this assumption. Later, in Sec. 3.3, we describe how this guess is refined.

Let \( (r_c, c_c) \) denote the center of the estimated main-subject region. The features \( f_i(x), i = 1, \ldots, 5 \) are modified as follows to put more weight on the region around \( (r_c, c_c) \):

\[
f_i(x) = f_i(x) \cdot f_c(x),
\]
\tag{15}

where \( f_c(x) \) denotes the relative distance of block \( x \) from the estimated center of the main subject. The quantity \( f_c(x) \) is given by:

\[
f_c(x) = 1 - \frac{\sqrt{(r - r_c)^2 + (c - c_c)^2}}{\sqrt{M_1/2^2 + M_2/2^2}},
\]
\tag{16}

where \( r \) and \( c \) denote the row and column value of the top-left pixel of \( x \). Note that the set of \( f_c(x), x \epsilon X \), can be arranged into a map, which we call the center-weighted map. The ultimate feature maps that we use is \( f_i(X) \) which is the normalized version of \( f_i(X) \) and given by:
The coordinate \((\mathbf{r}, \mathbf{c})\) quantity can determine on a per-image basis which feature to use and images. Thus, the MSD task can be very effective if one may actually be detrimental to the detection process on other images. However, some features which contribute greatly on some images to various degrees toward detecting the main subject. How-when combining features is that the features all contribute determine the usefulness of each map. The standard assumption After computing the five feature maps, the next step is to de-
termines the usefulness of each feature is cluster density, which we argue is a simpler approach. We also argue that the algorithm does not have to end after the features are combined, but can instead obtain a more accurate MSD via refining features in multiple stages. Later in Sec. 4 we show that along with the adaptive feature selection, the multistage refinement steps contribute greatly to the performance of our algorithm.

3.2 Adaptive Feature Selection Based on Cluster Density

After computing the five feature maps, the next step is to determine the usefulness of each map. The standard assumption when combining features is that the features all contribute to various degrees toward detecting the main subject. However, some features which contribute greatly on some images may actually be detrimental to the detection process on other images. Thus, the MSD task can be very effective if one can determine on a per-image basis which feature to use and which feature not to use.

Here, we describe how our algorithm makes this determination on a per-image basis. The criterion we propose to determine the usefulness of a feature is cluster density, which measures how clustered are the high valued pixels of the feature map. The smaller the cluster density value of a feature map, the more useful it is in this MSD task.

Let \(a_i\) denote the cluster density of feature map \(i\). The quantity \(a_i\) is given by:

\[
\alpha_i = \frac{\sum_{(r,c) \in P_i} \sqrt{(c_0 - c)^2 + (r_0 - r)^2}}{|P_i|^\beta},
\]

where \(P_i\) is the set of all coordinates \((r, c)\) corresponding to locations in the \(i\)th feature map with values greater than 0.5. The coordinate \((r_0, c_0)\) denotes the centroid of these locations. The value \(\beta\) was chosen empirically as 1.25 to compensate for large but high density clusters.

After \(a_1, \ldots, a_5\) are computed, these values will later be used to determine the weight \(w_i\) for each feature map \(f_i(\mathbf{X})\) (as described next in Sec. 3.3). Examples of the usefulness of each feature along with its cluster density value in different images are given in Fig. 4.

The idea of adaptive feature selection we propose here is similar to the idea of Hu et al. in Ref. 30 and Gopalakrishnan et al. in Ref. 31, in which they estimate the contribution of each feature map to saliency before combining them (see Sec. 2). Here we estimate the usefulness of each feature based on cluster density, which we argue is a simpler approach. We also argue that the algorithm does not have to end after the features are combined, but can instead obtain a more accurate MSD via refining features in multiple stages. Later in Sec. 4 we show that along with the adaptive feature selection, the multistage refinement steps contribute greatly to the performance of our algorithm.

3.3 Baseline MSD Map and Multistage Refinement

After the feature maps and corresponding cluster densities are computed, the next step is to combine the maps. We first assign weights for each feature map based on its cluster density to make what we call the baseline MSD map. We then modify these features and again combine them adaptively to refine the map in order to achieve a more accurate MSD result.

3.3.1 Stage 1: Baseline MSD map

At this first stage, we consider the whole image as the background \((\mathbf{B})\) to compute lightness and color distance in Eqs. (1) and (2). In order to apply center-weight modification for all features, we use the center of the image as our first guess of the center of the main subject; that is \((r_c, c_c) = (M_1/2, M_2/2)\) in Eq. (16).

The weight for each feature is then assigned based on the cluster density value:
Fig. 5 Stage 1 of the algorithm. Five features are measured and combined via adaptive feature selection to make the baseline MSD map and thus the bounding box around the main subject. Lightness and color distance are measured to the background which is considered as the whole image. In order to apply center-weight modification to all features, the center of the image is used as the first guess of the center of the main subject.

\[ w_i = \begin{cases} 
1, & \text{if } \alpha_i = \bar{\alpha}_1 \\
\frac{2}{3}, & \text{if } \alpha_i = \bar{\alpha}_2 \\
\frac{1}{3}, & \text{if } \alpha_i = \bar{\alpha}_3 \\
0, & \text{otherwise}
\end{cases} \]

where the sorting operation is used to sort the cluster density values in ascending order. The feature with the smallest cluster density value is assigned the greatest weight of 1, the feature with the second smallest cluster density value is assigned a weight of 2/3, the feature with the third smallest cluster density value is assigned a weight of 1/3, and the other features are assigned weights of 0.

Let \( R_X \) denote the baseline map of \( X \). \( R_X \) is computed as the weighted sum of all five feature maps:

\[ R_X = \frac{\sum_i w_i f_i(X)}{\sum_i w_i}. \]

The final baseline map that we use is \( \bar{R}_X \) which is the normalized version of \( R_X \) whose values have been rescaled to occupy the range \([0, 1] \):

\[ \bar{R}_X = \frac{R_X - \min(R_X)}{\max(R_X) - \min(R_X)}. \]

We next draw the smallest bounding box which contains at least 99% of pixels whose values in \( \bar{R}_X \) are greater than:

\[ T_1 = 1.5 \times \text{mean}(\bar{R}_X). \]

This adaptive threshold was chosen empirically. We use the percentage of 99% for this stage in an attempt to remove some unexpected background objects in the bounding box, which would otherwise be captured if we were to use 100%. Figure 5 summarizes the process to obtain the baseline map and bounding box.

### 3.3.2 Stage 2: Refinement based on new background

The result from Stage 1 can represent a first guess of the main subject region. We argue that this information can assist subsequent iterations of the algorithm. More specifically, the guess of the main subject region can be used to refine the feature maps and thus obtain a more accurate MSD map.

In this second stage, using \( \bar{R}_X \) and the bounding box from Stage 1, we modify all the features and again combine them adaptively. Specifically, the lightness and color distance of each block \( x \) are recomputed in the same way via Eqs. (1) and (2) except the background now is considered as locations outside of the baseline bounding box.

All five features are then center-weight-modified via Eq. (15) with \((r_c, c_c)\) assigned as the coordinate of the center of the bounding box. Again, these features are adaptively combined based on their cluster density values, except this time the weight \( w_i \) for each feature map \( f_i(X) \) is determined via:
Fig. 6 Stage 2 of the algorithm. The algorithm uses the bounding box from Stage 1 as a guess of the main subject’s location. Lightness and color distance are now measured to the new background (region outside the bounding box), and all features are modified with the new center-weighted map. These modified features are combined via adaptive feature selection to make the refined MSD map and thus the refined bounding box around the main subject.

\[ w_i = \begin{cases} 1, & \text{if } \alpha_i = \tilde{\alpha}_1 \\ \frac{\tilde{\alpha}_5 - \tilde{\alpha}_i}{\tilde{\alpha}_5 - \tilde{\alpha}_1}, & \text{if } \alpha_i = \tilde{\alpha}_2, \text{ where } \tilde{\alpha} = \text{sort} \{\alpha_i\}, \\ 0, & \text{otherwise} \end{cases} \]  

where the sorting operation is in ascending order. Only two features with smallest cluster density values are used, the other features are ignored. The new \( \tilde{R}_X \) is computed via Eq. (20) with the new sets of feature maps and weights, and then normalized to occupy the range \([0, 1]\) via Eq. (21). We then consider all pixels in this new \( \tilde{R}_X \) greater than the threshold \( T_2 \) as corresponding to main-subject locations. This threshold \( T_2 \) is computed as:

\[ T_2 = 2 \times \text{mean}(\tilde{R}_X). \]

The bounding box is selected to be the smallest rectangle which contains at least 75% of the values in \( \tilde{R}_X \) greater than \( T_2 \). Locations within this rectangle are considered a refined guess of the main subject. We choose a larger threshold \([2 \times \text{mean}(\tilde{R}_X)]\) and a smaller percentage (75%) to draw the bounding box in this stage because we want to capture the primary lightness and color of the main subject. We will use this information in the next refinement stage. Figure 6 summarizes the process to obtain the Stage 2 refined map and bounding box.

3.3.3 Stage 3: Refinement based on new foreground

Given that we now have a better guess of the primary lightness and color of the main subject, we modify lightness and color distance features via measuring the lightness and color distance of each block \( x \) to the foreground (region inside the bounding box from Stage 2). These two features are recomputed, respectively, via:

\[ f_1(x) = -|\tilde{L}^*(x) - \tilde{L}^*(F)|, \]  

\[ f_2(x) = -\sqrt{|\tilde{a}^*(x) - \tilde{a}^*(F)|^2 + |\tilde{b}^*(x) - \tilde{b}^*(F)|^2}, \]  

where \( F \) denotes the region of \( X \) within the bounding box. Similar to Stage 2, all five features are then center-weight-modified via Eq. (16) with \((r_c, c_c)\) assigned as the coordinate of the center of the bounding box in Stage 2.

The adaptive feature selection gives rise to a final set of feature weights in the same way via Eq. (22), and thus a final map \( \tilde{R}_X \) via Eqs. (20) and (21). From this final \( \tilde{R}_X \), we consider locations corresponding to the main subject as values of \( \tilde{R}_X \) greater than the threshold:

\[ T_3 = 1.5 \times \text{mean}(\tilde{R}_X). \]

We then draw a bounding box around all these pixels; locations within this bounding box are considered the final guess of the main subject. Figure 7 summarizes the process to obtain the final map and bounding box.
Stage 3 of the algorithm. The algorithm uses the bounding box from Stage 2 as a guess of the main subject's location. Lightness and color distance are now measured to the foreground (region inside the bounding box), and all features are modified with the new center-weighted map. Features are modified again and combined via adaptive feature selection to make the final MSD map and thus the final bounding box around the main subject.

4 Results and Analysis

In this section, we evaluate the performance of our algorithm based on two image databases. The first database is the MSRA Salient Object Database, which contains 5000 images. These are 24 bits/pixel color images with sizes ranging from 222×165 to 400×400 pixels. Each image in this set contains only one main subject that has been consistently labeled by nine human observers. The ground truth bounding box surrounding the main subject was averaged from results of observers as described in Ref. 32. The middle row of Fig. 8 shows some examples from this database.

The second database from Achanta et al. provides accurate object-contour-based ground truth instead of the
Precision and Recall are computed via:

\[ \text{Precision} = \frac{\text{A}(D \cap G)}{\text{A}(D)}, \]

(25)

\[ \text{Recall} = \frac{\text{A}(D \cap G)}{\text{A}(G)}, \]

(26)

where the \( \text{A}(\cdot) \) operator computes the area of a region. F-measure is the weighted harmonic mean of Precision and Recall given as:

\[ F_\alpha = \frac{(1 + \alpha) \times \text{Precision} \times \text{Recall}}{\alpha \times \text{Precision} + \text{Recall}}, \]

(27)

with \( \alpha = 0.5 \) in evaluation on the bounding-box-based ground truth database according to Ref. 32, and \( \alpha = 0.3 \) in evaluation on the object-contour-based ground truth database according to Ref. 29.

\( \text{BDE} \) measures the average displacement error between the boundaries of two regions (see Refs. 32 and 42) and is used here for evaluation on the bounding-box-based ground truth. Let \( B_D \) and \( B_G \) represent the boundary point set of the detected and ground truth rectangle, respectively. The BDE from \( B_D \) to \( B_G \), denoted as \( E(D, G) \), is computed as the average of distances from every point \( p \) in \( B_D \) to its closest point in \( B_G \):

\[ E(D, G) = \frac{1}{|B_D|} \sum_{p \in B_D} d(p, B_G), \]

(28)

where \( |B_D| \) represents the number of points in set \( B_D \), and \( d(p, B_G) \) represents the minimum Euclidean distance from \( p \) to all points in \( B_G \):

\[ d(p, B_G) = \min_{q \in B_G} \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2}, \]

where \( (p_1, p_2) \) and \( (q_1, q_2) \) are coordinates of \( p \) and \( q \), respectively.

The BDE from \( B_G \) to \( B_D \), denoted as \( E(G, D) \), is computed similarly. The final BDE between \( B_D \) and \( B_G \) is computed as the average of these two BDEs:

\[ \text{BDE}(B_D, B_G) = \frac{1}{2} [E(D, G) + E(G, D)]. \]

(29)

4.2 Representative Results

Figure 9 shows several examples with bounding-box-based ground truth, object-contour-based ground truth, our baseline and final MSD maps, and our final MSD bounding boxes. Notice that each refined MSD map in Fig. 9(e) demonstrates a marked improvement over the corresponding baseline map in Fig. 9(d); this refinement is a crucial step for MSD.

Figure 10 shows our MSD maps compared with maps from other algorithms, and Fig. 11 shows the bounding boxes around main subjects based on each map. Algorithms used
Fig. 10 Comparison of different algorithms’ final maps. Other algorithms usually detect not only the main subject regions but also several regions in the background. Overall, our MSD maps show clear main subject regions with well-suppressed background regions.

As shown in Fig. 11, most algorithms can yield an accurate bounding box for some images. However, as shown in Fig. 10, these accurate bounding boxes do not necessarily stem from accurate maps. The maps from other algorithms generally detect not only the main-subject regions but also several regions in the background. Note, however, that the maps from visual-attention-based algorithms such as ones from Itti et al., Hou and Zhang, Harel et al., Le Meur et al., and Achanta et al. were designed to denote regions of visual attention rather than main-subject regions. Overall, our maps show clear main subject regions and good suppression of background regions; our bounding boxes around main subjects are therefore easy to draw. Figure 12 shows additional representative results of our algorithm.
4.3 Overall Performance

Table 1 shows the Precision, Recall, F-measure, and BDE of our algorithm and others on the bounding-box-based ground truth database. Algorithms for comparison consist of the work of Itti et al., Harel et al., Le Meur et al., Hou and Zhang, Ma and Zhang, Achanta et al., and Liu et al. The algorithms from Refs. 24 and 32 directly output bounding boxes around the main subjects. (Code from Refs. 24 and 32 were not available for us to reproduce their results; the results presented in this comparison were obtained from their papers.) All other approaches output saliency maps; therefore, we need to draw bounding boxes around salient objects. Here, we apply an adaptive threshold in order to detect the main subject from the saliency maps as suggested in Ref. 25. The threshold was chosen as \( T = 2 \times \text{mean(Saliency Map)} \) as recommended in Ref. 29 for all algorithms’ saliency maps except maps from Le Meur et al. The threshold we chose for saliency maps from Le Meur et al. was \( T = 1.5 \times \text{mean(Saliency Map)} \) (similar to ours), as we found this threshold yields better MSD performance. Also, each bounding box was selected to contain only 95% of main-subject pixels according to Ref. 32. (See the Appendix for our testing results with different thresholds and percentages.)

It is important to note that Recall is not necessarily an appropriate measure for MSD, since a 100% Recall can be easily obtained by selecting the whole image. The main challenge in MSD is to simultaneously obtain high Precision and F-measure, and low BDE. As can be seen from Table 1, on these three criteria, our algorithm gives very competitive results compared to the best results (as we know) from Liu et al., given that we use only low-level features. Other algorithms generally yield higher BDE and lower Precision and F-measure. However, note that the algorithm of Achanta et al. and Hou and Zhang were not designed to yield a bounding box around the main subject. Also, given that algorithms from Itti et al., Harel et al., and Le Meur et al. were designed to predict visual attention, their performance on this database is noteworthy.

Table 2 shows the Precision, Recall, and F-measure of our algorithm compared with others on the object-contour-based ground truth database. Note that in order to detect salient objects, Achanta et al. augment their saliency maps with Mean-Shift segmentation. Given the segmentation map and the saliency map of an image, they average saliency values within each segment; then, an adaptive threshold is used to obtain a binary map of salient object(s). For comparison, we applied this Mean-Shift segmentation technique to all algorithms’ saliency maps. We also show results of our algorithm both with and without Mean-Shift segmentation. As can be seen from Table 2, our algorithm gives better results than all other algorithms in this comparison; and our algorithm’s performance improves if we use segmentation.

4.4 Discussion

Given that our algorithm operates based on five low-level features in an adaptive and iterative fashion, it is informative to analyze the contribution of each of these three aspects.
Table 2 Precision, Recall, and F-measure comparison of different algorithms on the object-contour-based database (Ref. 29). Mean-Shift segmentation was applied to all algorithms. Numbers in parentheses are results given in Ref. 29; we could not reproduce these results [see the Appendix for our testing results with our adaptive threshold, $T = 1.5 \times \text{mean(Saliency Map)}$, and the adaptive threshold proposed in Ref. 29, $T = 2 \times \text{mean(Saliency Map)}$]. In general, our algorithm without using Mean-Shift segmentation outperforms others, and our algorithm's performance improves if we use segmentation. Our performance without segmentation is very competitive to the results reported in Ref. 29, and we surpass their performance when segmentation is used. Note that Liu et al. in Ref. 32 tested their algorithm only on the bounding-box-based database; their code was not available to generate results on this object-contour-based database.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Precision</th>
<th>Recall</th>
<th>F-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Itti et al.</td>
<td>0.61 (0.78)</td>
<td>0.63 (0.49)</td>
<td>0.58 (0.64)</td>
</tr>
<tr>
<td>Le Meur et al.</td>
<td>0.70</td>
<td>0.70</td>
<td>0.67</td>
</tr>
<tr>
<td>Hou et al.</td>
<td>0.61 (0.67)</td>
<td>0.58 (0.57)</td>
<td>0.57 (0.61)</td>
</tr>
<tr>
<td>Harel et al.</td>
<td>0.66 (0.71)</td>
<td>0.70 (0.72)</td>
<td>0.64 (0.68)</td>
</tr>
<tr>
<td>Achanta et al.</td>
<td>0.69 (0.82)</td>
<td>0.77 (0.75)</td>
<td>0.68 (0.78)</td>
</tr>
<tr>
<td>Our algorithm (without segmentation)</td>
<td>0.79</td>
<td>0.77</td>
<td>0.77</td>
</tr>
<tr>
<td>Our algorithm with segmentation</td>
<td>0.86</td>
<td>0.85</td>
<td>0.84</td>
</tr>
</tbody>
</table>

Fig. 13 Histograms of the weights assigned to each feature in Stage 1 when running the algorithm on the bounding-box-based database (Ref. 32). The percentage values represent the number of times the features are used at each weight bin divided by 5000 (the total number of images). For each image, the weights of the five features in this stage have been normalized such that they sum to 1. Here, the normalized weight of each feature can only be either 0, 1/6, 1/3, or 1/2, according to Eq. (19).
Table 3 Average weight of each feature in each stage and over all stages when running the algorithm on the bounding-box-based database (Ref. 32). For each image, the weights of the five features in each stage have been normalized such that they sum to 1. The ± values denote ± one standard deviation of the respective mean.

<table>
<thead>
<tr>
<th></th>
<th>Sharpness</th>
<th>Lightness distance</th>
<th>Contrast</th>
<th>Color distance</th>
<th>Edge strength</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stage 1</td>
<td>0.319 (± 0.155)</td>
<td>0.208 (±0.191)</td>
<td>0.107 (±0.153)</td>
<td>0.278 (±0.202)</td>
<td>0.089 (±0.155)</td>
</tr>
<tr>
<td>Stage 2</td>
<td>0.276 (±0.245)</td>
<td>0.242 (±0.257)</td>
<td>0.080 (±0.180)</td>
<td>0.332 (±0.261)</td>
<td>0.069 (±0.177)</td>
</tr>
<tr>
<td>Stage 3</td>
<td>0.331 (±0.249)</td>
<td>0.170 (±0.237)</td>
<td>0.137 (±0.227)</td>
<td>0.265 (±0.258)</td>
<td>0.097 (±0.204)</td>
</tr>
<tr>
<td>Overall</td>
<td>0.309 (±0.190)</td>
<td>0.207 (±0.198)</td>
<td>0.108 (±0.164)</td>
<td>0.292 (±0.209)</td>
<td>0.085 (±0.161)</td>
</tr>
</tbody>
</table>

4.4.1 Contribution of each feature

Because we combine features adaptively, in each stage of the algorithm there are only two or three features that are used and all the others are ignored. It is informative to analyze the average contribution of each feature in each stage. Table 3 shows the average weight of each feature in each stage and the average weight over all stages. For each stage, the weights of the five features have been normalized such that they sum to 1. The ± values denote ± one standard deviation of the respective mean.
Table 4 Overall performance of our algorithm when using a set of optimized fixed weights (first row), using only the baseline map (second row), using no color information (third row), and using our full algorithm (last row). The ground truth database used here was the bounding-box-based database (Ref. 32).

<table>
<thead>
<tr>
<th></th>
<th>Precision</th>
<th>Recall</th>
<th>F-measure</th>
<th>BDE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Using a set of fixed weights for our five features</td>
<td>0.67</td>
<td>0.85</td>
<td>0.68</td>
<td>28.5</td>
</tr>
<tr>
<td>Using the baseline maps only</td>
<td>0.73</td>
<td>0.84</td>
<td>0.74</td>
<td>25.3</td>
</tr>
<tr>
<td>Our algorithm without color distance</td>
<td>0.75</td>
<td>0.83</td>
<td>0.74</td>
<td>24.8</td>
</tr>
<tr>
<td>Our algorithm</td>
<td>0.81</td>
<td>0.82</td>
<td>0.81</td>
<td>20.8</td>
</tr>
</tbody>
</table>

deviation of the respective mean. Note that the standard deviations are quite large due to the fact that these data include times when the features are assigned the weight of 0. Figures 13–15 show histograms of the weights assigned to each feature in Stages 1, 2, 3, respectively, when running the algorithm on the bounding-box-based database.32

As shown in Table 3 and Figures 13–15, the three most useful features are sharpness, color distance, and lightness distance. It is important to note that even though contrast and edge strength receive the lowest average weights, there are certain images for which these two features are crucial. Figure 16 contains examples of images in which contrast and/or edge strength play important roles, along with the respective weights in each stage. In general, we believe that contrast and edge strength contribute most on images in which: 1. the main subject is desaturated; or 2. the main subject contains multiple colors and/or lightnesses, while there are some background objects also in focus.

Fig. 15 Histograms (using 100 bins) of the weights assigned to each feature in Stage 3 when running the algorithm on the bounding-box-based database (Ref. 32). The percentage values represent the number of times these features are used at each weight bin divided by 5000 (the total number of images). For each image, the weights of the five features in this stage have been normalized such that they sum to 1. To promote visibility, the histogram only shows the number of times when a feature is actually used (weight greater than 0). For reference, the number of times that each feature gets a weight of zero is: Lightness Distance: 3270, Color Distance: 2371, Contrast: 3612, Sharpness: 1705, and Edge Strength: 4042.
Fig. 16 Examples of images in which contrast and/or edge strength play crucial roles. The weights (before normalized) for contrast and edge strength, respectively, for these images are: (a) Stage 1: 1, 2/3; Stage 2: 1, 0.4; Stage 3: 1, 0.42; (b) Stage 1: 1, 0; Stage 2: 0.77, 0; Stage 3: 0.82, 0; (c) Stage 1: 2/3, 1; Stage 2: 0.76, 1; Stage 3: 1, 0.71; (d) Stage 1: 2/3, 1; Stage 2: 0.72, 1; Stage 3: 0.99, 1; and (e) Stage 1: 1, 0; Stage 2: 1, 0; Stage 3: 1, 0.

Fig. 17 Examples of our algorithm on grayscale images. From top to bottom: original grayscale image, our MSD map, and our bounding box around the main subject.

Fig. 18 Some failure cases of our algorithm.
Table 5 Results of other algorithms on the bounding-box-based database (Ref. 32) with two different adaptive thresholds. The threshold of $2 \times \text{mean(Saliency Map)}$ was proposed in Ref. 29 and the threshold of $1.5 \times \text{mean(Saliency Map)}$ is our own value. For each threshold, we also show the results of using a bounding box which contains 100% salient pixels, and the smallest bounding box which contains at least 95% salient pixels (as proposed in Ref. 32).

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Threshold and Percentage</th>
<th>Precision</th>
<th>Recall</th>
<th>F-measure</th>
<th>BDE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Itti et al.</td>
<td>$2 \times \text{mean(Saliency Map)}$, 95%</td>
<td>0.639</td>
<td>0.808</td>
<td>0.655</td>
<td>34.602</td>
</tr>
<tr>
<td></td>
<td>$2 \times \text{mean(Saliency Map)}$, 100%</td>
<td>0.538</td>
<td>0.922</td>
<td>0.600</td>
<td>42.060</td>
</tr>
<tr>
<td></td>
<td>$1.5 \times \text{mean(Saliency Map)}$, 95%</td>
<td>0.591</td>
<td>0.889</td>
<td>0.638</td>
<td>36.708</td>
</tr>
<tr>
<td></td>
<td>$1.5 \times \text{mean(Saliency Map)}$, 100%</td>
<td>0.470</td>
<td><strong>0.973</strong></td>
<td>0.550</td>
<td>49.538</td>
</tr>
<tr>
<td>Achanta et al.</td>
<td>$2 \times \text{mean(Saliency Map)}$, 95%</td>
<td><strong>0.583</strong></td>
<td>0.811</td>
<td>0.601</td>
<td>41.979</td>
</tr>
<tr>
<td></td>
<td>$2 \times \text{mean(Saliency Map)}$, 100%</td>
<td>0.438</td>
<td><strong>0.966</strong></td>
<td>0.513</td>
<td>56.316</td>
</tr>
<tr>
<td></td>
<td>$1.5 \times \text{mean(Saliency Map)}$, 95%</td>
<td>0.541</td>
<td>0.865</td>
<td>0.582</td>
<td>44.562</td>
</tr>
<tr>
<td></td>
<td>$1.5 \times \text{mean(Saliency Map)}$, 100%</td>
<td>0.492</td>
<td>0.912</td>
<td>0.546</td>
<td>50.565</td>
</tr>
<tr>
<td>Harel et al.</td>
<td>$2 \times \text{mean(Saliency Map)}$, 95%</td>
<td>0.690</td>
<td>0.764</td>
<td>0.676</td>
<td>32.975</td>
</tr>
<tr>
<td></td>
<td>$2 \times \text{mean(Saliency Map)}$, 100%</td>
<td>0.570</td>
<td>0.915</td>
<td>0.623</td>
<td>39.599</td>
</tr>
<tr>
<td></td>
<td>$1.5 \times \text{mean(Saliency Map)}$, 95%</td>
<td>0.610</td>
<td>0.880</td>
<td>0.651</td>
<td>35.828</td>
</tr>
<tr>
<td></td>
<td>$1.5 \times \text{mean(Saliency Map)}$, 100%</td>
<td>0.466</td>
<td><strong>0.980</strong></td>
<td>0.545</td>
<td>51.047</td>
</tr>
<tr>
<td>Hou et al.</td>
<td>$2 \times \text{mean(Saliency Map)}$, 95%</td>
<td>0.609</td>
<td>0.883</td>
<td><strong>0.650</strong></td>
<td>35.515</td>
</tr>
<tr>
<td></td>
<td>$2 \times \text{mean(Saliency Map)}$, 100%</td>
<td>0.474</td>
<td>0.972</td>
<td>0.551</td>
<td>49.559</td>
</tr>
<tr>
<td></td>
<td>$1.5 \times \text{mean(Saliency Map)}$, 95%</td>
<td>0.567</td>
<td>0.926</td>
<td>0.625</td>
<td>38.704</td>
</tr>
<tr>
<td></td>
<td>$1.5 \times \text{mean(Saliency Map)}$, 100%</td>
<td>0.424</td>
<td><strong>0.990</strong></td>
<td>0.509</td>
<td>56.435</td>
</tr>
<tr>
<td>Le Meur et al.</td>
<td>$2 \times \text{mean(Saliency Map)}$, 95%</td>
<td><strong>0.746</strong></td>
<td>0.553</td>
<td>0.632</td>
<td>35.726</td>
</tr>
<tr>
<td></td>
<td>$2 \times \text{mean(Saliency Map)}$, 100%</td>
<td>0.702</td>
<td>0.668</td>
<td>0.657</td>
<td>33.125</td>
</tr>
<tr>
<td></td>
<td>$1.5 \times \text{mean(Saliency Map)}$, 95%</td>
<td>0.682</td>
<td>0.724</td>
<td><strong>0.670</strong></td>
<td><strong>31.764</strong></td>
</tr>
<tr>
<td></td>
<td>$1.5 \times \text{mean(Saliency Map)}$, 100%</td>
<td>0.612</td>
<td><strong>0.842</strong></td>
<td>0.652</td>
<td>33.389</td>
</tr>
</tbody>
</table>
4.4.2 Contribution of adaptive feature selection and multistage refinement

Table 4 shows results of using a set of fixed weights for our five features, where the weights were optimized over a training set that we used previously in Ref. 43. Also shown in Table 4 are the results obtained using our algorithm without multiple stages of refinement (i.e., using only the baseline MSD maps). By comparing the first row in Table 4 to the second row in this table, it is clear that adaptive feature selection (i.e., adaptive weights) does improve the performance of the algorithm. By comparing the second row in Table 4 to the fourth row in this table, it is clear that the multistage nature of our algorithm also contributes greatly to its performance. These results support the notion that adaptivity and multistage refinement can be effective strategies for MSD.

4.4.3 Main subject detection for grayscale images

Even though color is one of the three most useful features as revealed by our previous analysis, and even though we believe that humans also use color when detecting the main subject, our approach can be utilized to perform MSD on grayscale images. Figure 17 shows our MSD results on several grayscale images. Note that Liu et al. 32 conclude that their algorithm relies heavily on their color spatial distribution feature (weight of 0.54, while multiscale contrast and center-surround RGB histogram used weights of 0.22 and 0.24, respectively). Table 4 also shows results of our algorithm on the bounding-box-based database, 32 without using color information. This result is very promising; in fact, our algorithm without color information is better than all others except results from Liu et al. 32 and our own approach using color. This grayscale-only performance highlights the advantage that adaptivity and multistage refinement provide: in the absence of color, the next best feature will be used and refined in an attempt to fill the gap.

4.4.4 Limitations and future work

Figure 18 shows several failure cases of our algorithm. These failures result from the fact that either all features fail to highlight the main subject, or the algorithm fails to select the most useful features. To handle these types of failures, we would need either additional features, an improved means of measuring the usefulness of each feature, or possibly additional refinement stages.

Another important note is that we have designed our algorithm to detect one main subject in an image. The two databases used to analyze the performance of our algorithm also contain images with predominantly one main subject. What happens if an image contains two (or more) main subjects? Or, what if we need to detect the secondary, potentially less important subjects in an image? One probable solution for these problems is to run the entire algorithm iteratively. After the algorithm finds the first main subject, we can run the algorithm again, this time assigning less weight to the previously detected main subject region. Figure 19 demonstrates our initial results on detecting multiple main subjects.

5 Conclusion

In this paper we presented an adaptive, multistage algorithm for main subject detection. The algorithm first computes feature maps corresponding to lightness distance, color distance, contrast, sharpness, and edge strength. Next, the algorithm quantifies the usefulness of each feature on a per-image basis in order to assign a weight to each feature before combining the features via a weighted sum. The resulting MSD map is used to make an initial estimate of the location of the main subject, and then this initial estimate is used to refine the feature maps. The refined maps are again combined adaptively to generate a new estimate of the main subject’s location, followed by another level of feature refinement. Finally, the doubly refined feature maps are adaptively combined to generate a final estimate of the main subject’s location.

We tested the proposed algorithm on two image databases, one with bounding-box-based ground truth and one with object-contour-based ground truth. Our results showed that the proposed algorithm performs well on both databases. We believe that this performance is noteworthy given that our algorithm uses only low-level features. A further analysis revealed that the accuracy of our algorithm can be attributed to: 1. its ability to determine, on a per-image basis, which features to use and which features not to use; and 2. its multistage nature. Furthermore, our algorithm shows promise on grayscale-only images and on the task of multiple MSD. We believe that this strategy of adaptive feature refinement can be useful for a variety of image-processing applications.

Appendix

This section provides results of other algorithms with different adaptive threshold values on both databases. Table 5 shows the results of other algorithms on the bounding-box-based ground truth database. 32 Table 6 shows the results of other algorithms on the object-contour-based ground truth database. 32
Table 6 Results of other algorithms on the object-contour-based ground truth database (Ref. 29) with two different adaptive threshold values. The threshold of $2 \times \text{mean(Saliency Map)}$ was proposed in Ref. 29 and the threshold of $1.5 \times \text{mean(Saliency Map)}$ is our own value.

<table>
<thead>
<tr>
<th>Threshold</th>
<th>Precision</th>
<th>Recall</th>
<th>F-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>$2 \times \text{mean(Saliency Map)}$</td>
<td>0.612</td>
<td>0.449</td>
<td>0.515</td>
</tr>
<tr>
<td>$1.5 \times \text{mean(Saliency Map)}$</td>
<td>0.606</td>
<td>0.630</td>
<td>0.583</td>
</tr>
</tbody>
</table>

Achanta et al.

<table>
<thead>
<tr>
<th>Threshold</th>
<th>Precision</th>
<th>Recall</th>
<th>F-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>$2 \times \text{mean(Saliency Map)}$</td>
<td>0.722</td>
<td>0.576</td>
<td>0.631</td>
</tr>
<tr>
<td>$1.5 \times \text{mean(Saliency Map)}$</td>
<td>0.711</td>
<td>0.704</td>
<td>0.681</td>
</tr>
</tbody>
</table>

Harel et al.

<table>
<thead>
<tr>
<th>Threshold</th>
<th>Precision</th>
<th>Recall</th>
<th>F-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>$2 \times \text{mean(Saliency Map)}$</td>
<td>0.696</td>
<td>0.479</td>
<td>0.560</td>
</tr>
<tr>
<td>$1.5 \times \text{mean(Saliency Map)}$</td>
<td>0.663</td>
<td>0.701</td>
<td>0.645</td>
</tr>
</tbody>
</table>

Hou et al.

<table>
<thead>
<tr>
<th>Threshold</th>
<th>Precision</th>
<th>Recall</th>
<th>F-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>$2 \times \text{mean(Saliency Map)}$</td>
<td>0.616</td>
<td>0.412</td>
<td>0.505</td>
</tr>
<tr>
<td>$1.5 \times \text{mean(Saliency Map)}$</td>
<td>0.610</td>
<td>0.576</td>
<td>0.571</td>
</tr>
</tbody>
</table>

Le Meur et al.

<table>
<thead>
<tr>
<th>Threshold</th>
<th>Precision</th>
<th>Recall</th>
<th>F-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>$2 \times \text{mean(Saliency Map)}$</td>
<td>0.723</td>
<td>0.501</td>
<td>0.595</td>
</tr>
<tr>
<td>$1.5 \times \text{mean(Saliency Map)}$</td>
<td>0.699</td>
<td>0.696</td>
<td>0.666</td>
</tr>
</tbody>
</table>

Acknowledgments

This work was supported by the Army Research Office, “Enabling Battlefield Situational Awareness through a Cooperative and Intelligent Video Sensor Network,” 56940-CS-DPS, and by the National Science Foundation, “Content-Based Strategies of Image and Video Quality Assessment,” Award No. 0917014; PI: Damon Chandler, Oklahoma State University.

30. Y. Hu, X. Xie, W. Ma, L. Chia, and D. Rajan, “Salient region detection using weighted feature maps based on the human visual