# A Review of Modern Blob Detection & Analysis Techniques

Isaac Woodard[1]

[1] *Knight Campus Graduate Internship Program, University of Oregon, 1585 E 13th Ave, Eugene OR 97403*
[*] *isaac.j.woodard@gmail.com*

**Abstract:** Blob detection (BD) is one of the fundamental techniques in image processing and analysis. Blobs are connected regions of pixels which (hopefully) correspond to objects or other structures of interest in an image. BD is a multi-step process: 1) blob extraction to create a binary map of blob pixels from the original image, 2) blob processing to remove noise, 3) blob identification to group the blob pixels into blobs based on connectivity, 4) feature extraction to retrieve desired blob characteristics such as area or position. An overview is made of important BD concepts and common BD algorithms, including thresholding and differential methods. The applications of some algorithms are explored in the field of biomedical research. Also explored is the relevance of machine learning to both BD and the biomedical field. The importance of machine learning continues to grow due to its ability to automate processing of large data sets.

## 1. Introduction

Blob detection (BD) is a common goal in the fields of image processing and image analysis. It is used to identify objects in an image by determining the pixel regions corresponding to the objects. Biomedical research makes extensive use of BD for identifying cells and cellular structures. BD has also seen use in many industrial applications and for traffic and facial recognition systems.

Before discussing what BD is, it may be useful to start by explaining what is meant by a blob. A blob is a region of connected pixels in an image. In comparison to edges and corners, blobs are 2D structures, rather than 1D. Blobs don't have any particular shape, though it is common to try to match blobs to some desired target. When representing a blob, it is usually sufficient to use a binarized copy of the image containing the blob. In this way blobs are sometimes given the acronym of BLOB for "Binary Large Object" [1]. The concept of BLOBs is not unique to image processing, however, so we will stick to using the term "blob".

BD is the combination of *blob extraction* and *blob processing*, followed by *blob identification* or *connected component analysis*. During blob extraction, the initial binarization of the image is performed. In some cases this is enough to find the desired blobs, but often blob processing is necessary to remove noise or further segment the binary regions. The identification step then uses the binary map to group pixels into blobs based on their connectivity [1].

The connectivity of a region can be defined in different ways, with 4-connectivity and 8-connectivity being the most common. Using 4-connectivity, pixels horizontally and vertically adjacent are considered connected, while with 8-connectivity diagonally adjacent pixels are considered connected as well [2]. It may be worthwhile to note that these definitions of connectivity assume a Cartesian grid of pixels. If pixels are arranged differently, such as in a hexagonal grid, other definitions of connectivity must be used.

There are many techniques for extracting blobs–partly due to the lack of a standard definition for what constitutes a blob. While a blob is always a region of connected pixels, deciding how to distinguish those regions from the rest of the image depends on the application. To consider some examples, a region of interest could be a bright spot or a dark spot in an image with near-constant intensity as might be the case for a road, it could be an area centered around a point of illumination as is the case for cells stained with fluorescent dye, or it could be a particular pattern matched to a template as might be the case for facial recognition. Each of these cases is suited to different methods of blob extraction.

After identifying blobs in an image it is possible to determine their properties. Properties can be grouped as statistical or geometrical based on whether they are derived from the individual pixels or the shape of the blob as a whole [2]. A few examples of statistical properties include area, center of mass and orientation, while a few common geometrical properties are bounding box, bounding circle, perimeter and circularity. Determining properties allows blobs to be filtered and sorted in a final *blob classification* step [1, 3].

Blob properties seem to often be referred to as *features*. This can be confusing, though, because *features* is also a general term in the field of image processing for blobs, edges and/or corners identified in an image. To avoid that confusion, we will use the terms *properties* or *blob features* rather than simply *features*.

Two concepts fundamental to BD are histograms and binary images. In image processing, histograms are used to display the number of pixels with each intensity value in an image [4]. This is simplest to image for grayscale images but can be applied to color images as well. Histograms are often a useful in blob extraction as a way to identify pixel intensities corresponding to blobs, since blob pixels will create peaks on the histogram. In the ideal case, the histogram is bimodal, with one peak corresponding to the background of the image and the other to the blob(s) [2, 5].

Following blob extraction, binary images are used to label pixels as part of a blob or the background. If storage efficiency is important, true binary images should be used, but otherwise it is common to use pseudo-binary images where 8-bit pixels are assigned to either 0 (black) or 255 (white) [4]. When dealing with binary images, it is important to recognize the assignment of blob pixels and background pixels to true and false values is quite arbitrary. Similarly, there is no universal convention for assigning colors to true and false values [5].

## 2. Current State of the Art

### 2.1. Blob Detection Process

BD begins with the extraction of a binary map from an image. When needed, this is followed by blob processing to remove noise. Blob identification is then performed to group the blob pixels in the binary image into connected regions. With the blobs identified, blob features can finally be determined. Now that we've discussed the basics of BD let's look at the first three of these steps in more detail.

We'll consider blob extraction in the context of gray-scale images and global image thresholding–perhaps the simplest method of generating the binary map. Global image thresholding assigns all pixels below a certain intensity threshold to a single value and all pixels above the threshold to another value [5]. This creates a binary map with blob pixels corresponding to either bright or dark regions in the original image.

Good lighting is essential for image thresholding to be effective. Generally blobs correspond to objects or patterns in an image which need to be identified. For those regions to be distinguished from the background there must be clear contrast between the two and the overall brightness must be consistent across the image. It is possible to gain more flexibility by using dynamic thresholding. This allows different thresholds to be used in different parts of the image based on the local brightness. However, there must still be sufficient contrast between the regions of interest and the background.

Some issues following from blob extraction can be addressed by morphological noise reduction techniques. Small spots in the binary map as well as noisy blob edges can by removed through *opening* [2]. *opening* is the combination of *erosion* and *dilation*. Small features are first removed by erosion and then the remaining features are grown back to their original size [5]. On the other hand, holes or gaps in the binary map can be filled using *closing* [2]. Like opening, closing uses both erosion and dilation but in reverse order. Features are grown, closing the gaps, before being shrunk back down [5].

At this point, blobs should be clearly visible in the binary map and blob identification can be performed. The task of grouping pixels into connected regions is easy enough for humans to perform by inspection, but it is not trivial to perform algorithmically. One approach is the so-called Grass-Fire Algorithm [1]. The algorithm works by checking every pixel in the binary map. Whenever it finds a blob pixel, it starts a "grass-fire"–checking every adjacent pixel based on the chosen connectivity definition. It continues with the fire until it is unable to find another adjacent pixel, marking all found pixels as part of the current blob along its way. The algorithm can be implemented recursively or iteratively, each with their own trade-offs.

### *2.2. Overview of Other Algorithms*

There are many other algorithms which can be used in the BD process. We'll start by going over differential methods for blob extraction. These are some of the most powerful and flexible BD methods available, avoiding some of the shortcomings of simple thresholding. We'll then discuss the watershed algorithm for blob segmentation following blob extraction. We'll finish with a niche method for blob detection known as template matching.

Using a gradient filter to calculate the local intensity derivative across an image is a fundamental part of edge detection. Gradient filters can also be used for BD, however. A formal definition for a blob used by Tony Lindeberg, an influential BD theorist, treats a blob as the combination of a local extremum and a saddle point [6]. Both of these elements can be identified by creating a gradient intensity map of an image.

Several different gradient filters are commonly used. A long-standing filter is the Laplacian of Gaussian (LoG), which combines smoothing with the gradient calculation to improve efficiency [4]. Increased efficiency can be obtained using the Difference of Gaussians (DoG) at the cost of being an approximation of the gradient [7]. For scale-selection algorithms, Lindeberg found the Determinant of the Hessian (DoH) is less biased than the LoG [8].

Differential methods have the advantage of performing well even when noise is present [7]. Apart from thresholding, another simple approach is to just use local intensity extrema to detect blobs and forgo use of a gradient filter. This approach performs pourly with noise, however; a single blob might be flagged as multiple smaller blobs [7].

There are cases, though, where splitting a single blob into multiple smaller blobs is the goal. An example is identifying cells in the process of dividing. Contrast between the dividing cells is often too small for thresholding or local extrema BD algorithms to identify them as separate blobs. In these cases a *watershed* algorithm can be used to separate the blobs.

A common analogy for describing the watershed algorithm is rain falling on a mountian range [3, 7]. Imagine the image as a topographic map where the local extrema used to identify the blobs are peaks. Now imagine rainfall running down the peaks into the background of the image. The water streaming down peaks within the same blob will meet somewhere in between them, creating a boundary. This boundary can then be used to segment the blob.

Another blob detection method unlike those already discussed is template matching [7]. Template matching uses a reference image–smaller than the image to perform BD on–and scans it across every pixel location. Rather than searching for exact matches, it is generally more useful to calculate the percentage match at each pixel location. Still, template matching is particularly poor at identifying blobs of different sizes so use cases are more limited.

### *2.3. Other Topics*

Some mention has been made of multi-scale algorithms. Differential BD approaches use filters which can only be tuned to detect blobs of one size at a time. This crippling flaw can be addressed by processing the image with a set of filters of different sizes [9]. The blobs detected by each filter are then compared with each other and resolved to avoid duplicate tags of the same region.

Such multi-scale designs are the de-facto approach when using differential BD algorithms. They have been greatly influence by the work of Tony Lindeberg, who has published more than a dozen papers relating to scale-space and BD extending as far back as 1990. An additional benefit is the removal of the need for a separate segmentation step when contrast between blobs is low.

The focus of this paper is on 2D BD algorithms, but it is also possible to detect blobs in 3D images. In a study on magnetic resonance imaging (MRI) of kidney glomeruli, Zhang et. al. proposed the Hessian-based Difference of Gaussians (HDoG) for efficient detection of blobs in 3D images [10]. Their motivation for designing a new differential algorithm was the poor segmentation performance of standard algorithms like the LoG. Conceptually, 3D blobs are quite similar to 2D blobs; they contain a local intensity maxima and can be any arbitrary shape. The issue of lighting becomes more complicated, particularly in cases like MRI where there is no "light" to speak of, but contrast between blobs and their surroundings remains important.

Most BD algorithms are limited to pixel-precision due to their nature of being defined as a group of connected pixels. It is possible to define a subpixel precise center for a blob using a center of mass calculation. Defining the boundaries of blobs with subpixel precision, however, either requires fitting a geometric shape to the blob as in the work of Stefan Hinz [3], or performing complementary edge detection to extract subpixel contours for each blob [2]. In this way, BD and edge detection can sometimes be used to achieve the same goal: construction of 2D regions in an image.

## 3. Technology

BD sees much use in biomedical research and applications. Large sets of data frequently encountered in the field are often impractical to process manually. Using BD allows items of interest such as cells to be identified across an entire image. This can allow hundreds of items to be identified and analyzed at once rather than one at a time.

A study on BD of kidney glomeruli imaged with magnetic resonance was previously mentioned. To provide some medical context, the glomeruli of the kidney form a network of blood vessels which are responsible for transporting waste out of the bloodstream. Though they are blood vessels, they appear as spots in MRI images [10]. This study highlights the ability of BD to be used with non-optical imaging (as well as 3D imaging).

Fluorescent dye microscopy is a common technique for imaging cells and even particular components within cells. A dye is selected which binds to the desired cells or components. Kong et. al. explored the use of their generalized Laplacian of Gaussian (gLoG) filter for different sets of fluorescent microscopy images [11]. One of the sets showcased the filter's ability to determine the orientation of cell nuclei. Another set illustrated the ability of their BD algorithm to segment an image based on the locations and number of cells. A curious quality of their segmentation approach was the choice not to distinguish between cells and the background. In their study, the extra information of precise cell boundaries wasn't needed, so the extra processing effor would have been a waste.

While BD is powerful, on its own it still requires the selection of suitable parameters. Optimizing parameters for a handful of images can be tedious and prohibitive for large image sets. A natural next step to the problem of too much data is machine learning. Machine learning creates the possibility of training a program with a representative set of sample images to select the parameters automatically.

Martin Ter Haak described the use of machine learning in the context of ribonucleic acid (RNA) sequencing using fluorescent dye microscopy [7]. The overall goal was to outline a machine learning BD algorithm capable of automatically selecting parameters for blob detection of large sets of images. Additional goals were to handle 3D images as well as images too large to fit in memory.

## 4. Future Directions

There are several areas where further research would be valuable. Several different BD algorithms were discussed but limited attention was given to the advantages of one over another. Studies comparing the performance of one algorithm to another could provide further insight. Similarities were also drawn between blob extraction and contour extraction. Given the ability of these two methods to both outline regions of interest, considering trade-offs between the two would shed light on when it is more appropriate to use one over the other.

Many other applications of BD bear exploration as well. BD is used for facial recognition, tracking cars in traffic and identifying roads in satellite images. There are also likely applications in the semiconductor industry given the importance of image processing for analysis of microscopy data. Parallels might even be found between biomedical and semiconductor research.

In terms of development in the field of BD, there is still untapped potential in machine learning. Haak cited two popular software packages for biomedical research, CellProfiler and Fiji, which at the time of publication (2018) did not support or only partially supported machine learning [7]. A number of other packages were available which allowed partial automation through machine learning, but none which offered full automation.

Another potential use for machine learning is in automation of threshold selection for the blob extraction step of BD. Przemysław Pietrzkiewicz broke down several different low-level paradigms for selecting a threshold appropriate for a variety of objects and patterns under different lighting conditions [2]. Rather than choose a method manually, machine learning could be used to automate the selection.

## References

1. T. B. Moeslund, *BLOB Analysis* (Springer London, London, 2012), pp. 103–115.
2. P. Lowej, “Image analysis techniques for industrial inspection systems,” (2012).
3. S. Hinz, “Fast and subpixel precise blob detection and attribution,” (2005), pp. III – 457.
4. W. Burger and M. J. Burge, *Principles of Digital Image Processing: Fundamental Techniques* (Springer Publishing Company, Incorporated, 2009), 1st ed.
5. J. D. Gibson and A. Bovik, *Handbook of Image and Video Processing* (Academic Press, Inc., USA, 2000), 1st ed.
6. T. Lindeberg, “Detecting salient blob-like image structures and their scales with a scale-space primal sketch: A method for focus-of-attention,” Int. J. Comput. Vis. **11**, 283–318 (1993).
7. M. ter Haak, “Machine learning for blob detection in high-resolution 3d microscopy images,” Ph.D. thesis (2018).
8. T. Lindeberg, “Scale selection properties of generalized scale-space interest point detectors,” J. Math. Imaging Vis. **46**, 177–210 (2013).
9. T. Lindeberg, “Feature detection with automatic scale selection,” Int. J. Comput. Vis. **30**, 77–116 (1998).
10. M. Zhang, T. Wu, S. Beeman, L. Cullen-McEwen, J. Bertram, J. Charlton, E. Baldelomar, and K. Bennett, “Efficient small blob detection based on local convexity, intensity and shape information,” Med. Imaging, IEEE Transactions on **PP**, 1–1 (2015).
11. H. Kong, H. C. Akakin, and S. Sarma, “A generalized laplacian of gaussian filter for blob detection and its applications,” Cybern. IEEE Transactions on **43**, 1719–1733 (2013).