A Geometric Feature Detection Approach to Particle Picking in Electron Micrographs

Zeyun Yu and Chandrajit Bajaj

The Center of Computational Visualization,
Department of Computer Sciences and Institute of Computational Engineering & Sciences,
The University of Texas at Austin,
Austin, Texas 78712, USA

Abstract

Accurate and automatic particle detection from cryo-electron microscopy (cryo-EM) images is very important for high-resolution reconstruction of large macromolecular structures. In this paper, we present a novel method, based on feature detection, for particle picking. Two fundamental concepts of computational geometry, namely, distance transform and Voronoi diagram, are used for detection of critical features as well as for accurate location of particles from the micrographs. Unlike the conventional template-matching methods, our approach detects the particles based on their boundary features instead of intensities. The geometric features derived from the boundaries provide an efficient way for locating particles quickly and accurately, which avoids a brute-force searching for the best position/orientation. Our approach is fully automatic and has been successfully applied to detect particles with roughly circular or rectangular shapes. Particle detection can be enhanced by multiple sets of parameters used in edge detection and/or by anisotropic filtering. We will also discuss the extension of this approach to other types of particles with certain geometric features.

1. Introduction

The single particle technique [1-3] has been widely used for 3D reconstruction of large molecular complexes from electron micrographs. In contrast to the X-ray diffraction technique, the single particle method does not require formation of crystals. However, the signal-to-noise ratio (SNR) in most cryo-EM images is very low due to various reasons so that high-resolution single particle analysis often has to rely on averaging of a large number of identical particles to improve the signal-to-noise ratio [2, 3]. Therefore, locating most, if not all, of the particles in the digitized cryo-EM images is a crucial step in high-resolution single particle reconstruction. This task, commonly known as *particle picking* or *particle detection* in single particle analysis, can certainly be carried out manually (by mouse clicks). However, as the resolution approaches the atomic level, hundreds of thousands of particles may be necessary [4], which makes it impractical to manually pick the particles. In addition, particle detection by eyes may be inaccurate and subjective.

Lots of methods have been proposed for automatic or semi-automatic particle detection (see [5] for a recent review). The first automatic method was proposed by Heel [6]. In this approach the local variance over a small area around each point is computed, and each local maximum of the variance map is then considered as a particle. Another commonly used approach is the template-matching algorithm (see [7-10]), where the template is chosen as a rotationally averaged image of manually picked particles. The template is cross-correlated with the entire image and the "peaks" of the resulting cross-correlation map are identified as particles. This method, however, may fail for non-spherical particles or for multiple-view particles. A more robust approach is to use multiple references, each of which stands for one of the non-symmetric orientations or one of the views of the same object/particle [11, 12]. But obviously it is computationally intensive when multiple views or orientations are considered [13]. A more recent approach was based on the fast local correlation technique [14]. Due to the numerical scaling, this method proves to be more sensitive than the conventional cross-correlation methods for discriminating peaks from the cross-correlation maps.

The above-mentioned techniques, as well as some other methods such as the crosspoint method [15], the texture analysis method [16], the ring-filter based method [17] and the neural network approach [18], are all explicitly or implicitly based on intensity comparison. In other words, all the intensities within a local window around a pixel have to be considered and compared in order to determine whether or not a particle is located at that pixel.

Another group of particle detection algorithms are based on edge detection. In [19], the authors employed edge detection to estimate the perimeter of the particle being considered, from which the mass of the particle is calculated as a measure for particle selection. However, this method only works for dark field electron micrographs. A more general approach, proposed by Harauz *et al* [20], used edge detection followed by component labeling and symbolic (high-level) processing. This method relies on the connectivity of the detected edges. As the signal-to-noise ratio goes too low, the inaccurate edges may make it difficult to deal with some situations (e.g., edges of two particles are wrongly connected or edges of one particle are wrongly split). Recently, Zhu *et al* proposed a method, based on edge detection followed by Hough transforms, for automatic detection of particles [13, 21]. This method is more robust to noise because Hough transform can extract critical features from the edges and all the features are then integrated to make the decision.

In this paper we present a new method for automatic particle detection. Our method, similar to the one proposed by Zhu *et al* [13, 21], is based on edge detection followed by feature extraction from the edge map. However, our method borrows two fundamental concepts, namely, Voronoi diagram and distance transform, from computational geometry. Unlike conventional template-matching techniques, where the templates are “real” intensity images, we only need to know the geometric information of the true particles (e.g., the radius of circular particles or width/length of rectangular particles). A “virtual” shape is generated from the given geometric information and matched against the edge map of the original electron micrograph. The scoring function of this feature-based approach is evaluated with the help of the distance transform. The speed of the detection process can be largely improved by a good initialization of the “virtual” shape in the edge map with the help of Voronoi diagram and distance transform. This avoids a brute-force searching for the best position/orientation. We also describe how to refine the centers/orientations of the detected particles by moving the particles along the distance map. In case of circular and rectangular particles, as we consider in Section 3, the competitions between particles (*circle-circle*, *rectangle-rectangle* and *circle-rectangle*) are introduced to guarantee a very small number of false positives in most of the images we investigated. We will also discuss several other issues such as the use of multiple sets of parameters for Canny edge detection, the improvement by anisotropic filtering and the extension of our geometric-feature-detection based method to other shapes.

The rest of this paper is as follows. Section 2 gives a brief review of Canny edge detection, Voronoi diagram and distance transform. We describe the details of our method in Section 3. Results and more discussions on several other issues will be given in Section 4 and Section 5, respectively. Finally we give our conclusions and future work.

2. Related Work

2.1 Canny Edge Detection

Edge detection is the first crucial step in our method in order to match a template shape with an electron micrograph. Since it was proposed in 1986, Canny edge detection [22] has drawn a lot of attention for its efficiency of edge detection. The most promising strategies used in this method are non-maximal suppression and double-threshold. The gradient magnitude map can be computed using classical gradient operators [23] and a non-maximal suppression is then applied to the gradient magnitude map in order to obtain “thin” edges and extract candidate edges. Two thresholds are assumed such that candidate edges above the higher threshold are always recognized as true edges and candidate edges that are connected to the true edges by a path of pixels with gradient magnitudes higher than the lower threshold are also recognized as true edges. Besides the two thresholds, a parameter *sigma* is used in Gaussian filter that is applied to the original image before the gradient magnitude map is calculated. However, the noise commonly seen in electron micrographs makes the edge detection very sensitive to the selection of these three parameters, even though Gaussian filtering is applied. In Section 5, we will see how anisotropic filtering can help to improve the edge detection results. Combining multiple sets of parameters, as discussed in Section 5, would be another useful way to reduce the sensitivity to the selection of parameters.

2.2 Distance Transform and Voronoi Diagram

Given a set of discrete points (called *feature points*) P_i, $i = 1,2,\cdots n$ in 2D space $\Re^2$, we define the *nearest neighbor*, denoted as *NN*(*A*), of an arbitrary point *A* as one of the feature points P_k such that $d(A,P_k) \le d(A,P_i)$ for any $i = 1,2,\cdots n$, where the function $d(*,*)$ is Euclidean distance function between two points. Therefore, given a set of feature points in $\Re^2$, we can define a function value for any point *A* as the Euclidean distance between *A* and its nearest neighbor *NN*(*A*). The obtained function map is known as *distance transform* (DT) of the feature points. An example of distance transform can be seen in Fig.1(a), where the feature points are shown by white dots for better illustration although the distance function value at any feature point is always zero.

If we recall the definition for *nearest neighbor*, we would notice that some points in $\Re^2$ might have two or more nearest neighbors. Those points are of great interest in computational geometry. In fact, all of those points form a well-known diagram, called *Voronoi diagram* (VD). A Voronoi diagram generally consists of a set of polygons, each of which corresponds to one feature point, and all polygons exactly form a partition of the space $\Re^2$. Fig.1(b) shows an example of VD.

Both distance transform and Voronoi diagram have been widely used in computer vision, image analysis as well as many other fields. In this paper, we will see how they are applied to particle detection from electron micrographs. Although many algorithms exist for fast computations of DT and VD, we found that the list-processing approach [24] was most suitable for our application. This approach calculates the nearest neighbor of every pixel in the digitized space of $\Re^2$ and stores the results in a number of lists of segments, from which the DT and VD can be easily calculated.

3. Approach

In this section we will describe the details of our particle detection algorithm. We consider only two types of particles here: circular particles and rectangular particles. Particles with other shapes can be detected in a similar way if certain geometric features of the particles are available, as discussed in Section 5. Our method is based on geometric feature detection from image edges and template matching against the edge map. We do not need the “real” intensity templates. This avoids the manual and tedious generation of the reference images. What we need is just the geometric information of the templates. Specifically, we assume that the template circle has a fixed radius *Rd* and the template rectangle has a fixed width *Wd* and a fixed length *Ln*. As an example, a small portion of one Cryo-EM image, shown in Fig.2(a), will be considered in this section.

3.1 Pre-Computations

Edges are first computed using Canny edge detector [22]. To improve the robustness to the noise, an edge-cleaning operation is applied to the detected edge map, where short edges are removed. As seen in Fig.2(b), the most important features are roughly kept in the cleaned edge map.

It has been pointed out that distance transform is very useful for boundary-based template matching [25-27]. The essential idea is to calculate the distance transform of the edge map of the target image. The average distance value calculated along the template contour at a certain location in the target image gives the goodness of the matching between the target image and the template at that location. Obviously a smaller average distance value means better matching. This idea is also used in our method. In addition, as we will see in the following, the distance transform is used in our approach for two more reasons. First, it is used for the initial location of the circles in the circle detection. Second, it is used for the center and/or orientation refinement in both circle detection and rectangle detection. The distance transform of Fig.2(b) is shown in Fig.2(c).

The Voronoi diagram is computed in order to guess the initial locations and orientations of the rectangular particles. More details on how to do this will be given in the following. Fig.2(d) shows the Voronoi diagram of the edge map. Both distance transform and Voronoi diagram are computed using the list-processing approach [24].

3.2 Circle Detection

The circle detection includes the following steps.

Initial Circle Detection. From the distance transform of the edge map (shown in Fig.2(c)), we can see that most circular particles have a local maximum near their centers in the distance map. Therefore, we begin circle detection by detecting the local maxima of the distance map. Our goal here is to find all possible circles, although local maxima in the distance map do not always mean the center of a circle (for example, the rectangles also have local maxima near their centers). The local maximum is defined in our algorithm within a 3×3 local window, which means that a pixel is said to be a local maximum if its distance value is no less than the distance values of its surrounding (eight) neighbors. Besides this, the distance values at (valid) local maxima must be less than the radius *Rd* of the template circle and greater than a small fixed value (say, 5). Fig.3(a) shows the initial circles that we detect using these criteria. Remember that the reason we see so many ``local maxima" in Fig.3(a) is that the edges obtained by Canny detector do not show perfect circles or rectangles due to the noise seen in the original images.

Circle-Circle Competition. From the initial circle detection, we see too many circles that do not correspond to the true circular particles. Hence we need to delete most of them. One of the criteria for deleting false circles is the assumption in the *single particle reconstruction* that all particles should be isolated. This assumption immediately gives the following rule for deleting false circles: if two circles intersect, then we must delete one of them. It is quite easy to determine whether two circles intersect or not. But the remaining question is how to choose the ``winner" from two intersecting circles. As we said before, the average distance value along a template contour indicates the goodness of the matching. The average distance value of a circle with center $\vec{C}$ is defined by:

$$Adv(\vec{C}) = \frac{1}{n} \sum_{|d(\vec{P},\vec{C})-Rd|\le 1} DT(\vec{P}) \tag{1}$$

where n is the number of pixels satisfying $|\, d(\vec{P},\vec{C}) - Rd \,| \le 1$ and $d(*,*)$ is Euclidean distance between two pixels. $DT(\vec{P})$ is the distance value at $\vec{P}$ in the distance map. Therefore, the ``winner" of two intersecting circles should be the one with smaller average distance value. By this way, we can delete most of the false circles seen in the initial circle detection. The remaining circles are shown in Fig.3(b), from which we can see that the number of circles is largely reduced. But we still see two problems. One is that several circles are more or less away from their true centers due to the noise in the edge map. The other is that some rectangles are wrongly recognized as circles. Both problems will be addressed in the following. Although we said above that all circles must be completely isolated, in Fig.3(b) we allow the circles to partially intersect with each other. The reason is that, at this moment, some circles are not well centered. If all survival circles were required to be completely isolated, then some true circles that are not well centered might be wrongly deleted.

Center Refinement. As we said before, some circles seen in Fig.3(b) do not have correct centers. It is necessary to refine the centers of those circles. How do we do that? Recall the definition of distance transform that we give in Section 2. Before we calculate the distance transform, we need to identify the nearest neighbor of every pixel in the image domain. Therefore, given a circle with center $\vec{C}$, we can define a force on that circle as follows:

$$\vec{F}(\vec{C}) = \frac{1}{n} \sum_{|d(\vec{P},\vec{C})-Rd|\le 1} (NN(\vec{P}) - \vec{P}) \tag{2}$$

where n and $d(*,*)$ are defined the same as in equation (1). $NN(\vec{P})$ is the nearest neighbor of $\vec{P}$. The force $\vec{F}(\vec{C})$ indicates how far the circle should be moving to the correct center. This is an iterative process. Once the circle moves to a new position $\vec{C}'$, the new force $\vec{F}(\vec{C}')$ is calculated and the circle keeps moving in this way until $\|\vec{F}(\vec{C}')\|$ is smaller than a given value. Generally two or three iterations should be enough for a circle to reach its correct center. Fig.3(c) shows the result after center refinement. An interesting observation is that some circles may move to the same location such that the total number of circles is reduced.

Further Refinement. From Fig.3(c) we can see that some circles intersect again due to the center refinement. So we need to run the *Circle-Circle Competition* again. But this time we allow no intersection at all between any two circles. In addition, we delete those circles that have too large average distance value. The final result for the circle detection is shown in Fig.3(d). Remember that we still have the problem that some rectangles are wrongly recognized as circles. This problem will be easily resolved after the rectangle detection described below.

3.3 Rectangle Detection

The rectangle detection is performed separately from the circle detection. It also includes four steps: *initial rectangle detection*, *rectangle-rectangle competition*, *center/orientation refinement* and *further refinement*. We will explain them step-by-step, but special treatment will be given to the first step, the most different part from the circle detection.

Initial Rectangle Detection. As we saw before, the local maxima of the distance map indicate not only the centers of the circles but the centers of the rectangles as well. Hence, a simple way to detect the initial rectangles is again by detecting the local maxima of the distance map. However, since every rectangle has its center as well as its own orientation, it is obviously too much time-consuming for us to try every possible orientation at each local maximum for the best matched rectangle. Therefore, we need to seek for other efficient ways for initial rectangle detection. In the following we will consider a method based on Voronoi diagram of the edges (see Fig.2(d)). This method begins with *corner detection* followed by conversion from corners to centers/orientations.

A corner of a rectangle is illustrated in Fig.4(a), where the edges (dark dots) are discrete points forming a right angle at *C*. We construct the Voronoi diagram (dashed lines in Fig.4(a)) of those edge points and check every point *A* on the Voronoi diagram. If *A* happens to be on the ``bisector'' of the corner as shown in Fig.4(a), then *A*, together with its two nearest neighbors *B* and *D*, should roughly form a right angle (that is, $\angle BAD \approx 90^o$). In addition, the directions of the edges at *B* and

D should be roughly perpendicular to the vector $\overrightarrow{AB}$ and $\overrightarrow{AD}$, respectively. In this case, we can compute the corner *C* by $\vec{C} = \vec{B} + \vec{D} - \vec{A}$. On the other hand, if *A* is not on the ``bisector" of the corner, then *A* and its nearest neighbors do not satisfy the above conditions. Therefore, we have the following algorithm to detect all the corners in an edge map based on Voronoi diagram.

Algorithm for Detecting Corners:

1. Check every point *A* on the Voronoi diagram (VD).
2. Find its nearest neighbors *B* and *D*. This is readily available after we compute VD [24].
3. If $\angle BAD \approx 90^o$ and both vectors, $\overrightarrow{AB}$ and $\overrightarrow{AD}$, are roughly perpendicular to the edges at *B* and *D*, respectively, then mark $\vec{C}$ ($\vec{C} = \vec{B} + \vec{D} - \vec{A}$) as a corner.
4. Repeat (1)-(3) until all points on the VD are checked.

It is worth noting that many corners detected by the above algorithm often correspond to the same corner although they may stay a little bit away from the true corner due to the noisy edges. The above method for corner detection is quite robust to noise because, even if part of the edges around a corner are damaged, other undamaged edges can still contribute to the detection of that corner.

After we detect the corners, we then need to convert each of the corners into the center and orientation that uniquely represent each of the rectangles (remember that we assume the width and length of all the rectangles are fixed). Given a corner, we actually have two cases, as shown in Fig.4(b), which correspond to two possible centers/orientations. Hence we need to choose one of them. Again, the ``winner" is chosen by calculating the average distance value on each of these two rectangles and the one with smaller average distance value is the ``winner". It is worth noting that every rectangle has four corners and thus a rectangle can be detected from any one of its four corners even if the others are damaged. Fig.5(a) shows the result of the initial rectangles detected by our method. Similar to the initial circle detection, there are many false rectangles due to the noisy edges detected by Canny edge detector. Therefore, we need to let all the initial rectangles compete with each other as described in the following. Compared to the simple approach that searches for the local maxima of the distance map and checks every possible orientation to detect the initial rectangles, the method described above tremendously reduces the number of the candidate rectangles for further processing and hence improves the performance.

Rectangle-Rectangle Competition. The competition between rectangles is similar to what we saw for circle-circle competition. However, determining whether two rectangles intersect or not is more complicated than the circle-circle intersection. In addition, the average distance value for each rectangle is now computed along the rectangular contour.

Center/Orientation Refinement. The center refinement of the remaining rectangles is similar to the center refinement of circles seen above. It is possible that the orientations of the rectangles should also be refined. But our experience shows that the orientation refinement (and sometimes even the center refinement) is often unnecessary since the results after the first two steps always give accurate orientation and/or center for each detected rectangle.

Further Refinement. Similar to the circle detection, we need to run the rectangle-rectangle competition again for those rectangles that intersect after the above steps. Again, it is useful to delete the rectangles that have too large average distance value. The final result of the rectangle detection is shown in Fig.5(b), from which we can see that there are still a small number of false rectangles. Remember that in circle detection we have an unsolved problem that some rectangles are wrongly recognized as circles. Now it is time for us to solve both problems together by *circle-rectangle competition* as described below.

3.4 Circle-Rectangle Competition

We first put together the detected circles and rectangles as shown in Fig.5(c). Then we check each of the circles and see if it intersects with any of the surrounding rectangles. If it does, then the circle-rectangle competition is applied. The circle-rectangle competition is similar to the circle-circle competition or rectangle-rectangle competition seen before. The ``winner'' is also chosen to be the one that has the smaller average distance value. The final result is shown in Fig.5(d). We summarize the overall procedure of our particle detection approach in Fig.6.

4. Results

In this section we demonstrate the performance of our particle detection algorithm on several examples of Cryo-EM images, known as *Keyhole Limpet Hemocyanin* (KLH) [13]. The edges are detected using Canny edge detector with the following parameters: *tlow* = 0.9, *thigh* = 0.8, *sigma* = 4. Here both *tlow* and *thigh* are in the range of $[0.0-1.0]$. Two thresholds in Canny method are set as follows. We first count the number (denoted by n) of pixels that pass the non-maximal suppression. The higher threshold is set as the magnitude value of the pixel whose magnitude ranks the $\lceil n \times (1 - thigh) + 0.5 \rceil$-*th* among all the pixels that pass the non-maximal suppression. The lower threshold is simply set as the higher threshold multiplied by *tlow*. We clean all the edge maps by removing short edges of length below 20 pixels. The radius of the template circle is 35 pixels. The width and length of the template rectangle are 65 pixels and 80 pixels, respectively.

We tested our method on SGI Onyx2 with single processor (400 MHz MIPS R12000) and the total computational time is about 20 seconds for each image with a size of 1024×1024 pixels. About 18%, 2% and 6% of the total time are used for edge detection, edge cleaning and computations of DT&VD, respectively. The particle detection (including circle detection, rectangle detection and circle-rectangle competition) takes about 74% of the total time, which is roughly four times of that for Canny edge detection. The false negatives/positives for each example, evaluated against visual detection, can be found from Fig.7 - Fig.9. From these examples we can see that our approach gives accurate centers and/or orientations for the detected particles with few false positives/negatives.

5. Discussions

5.1 Using Multiple Sets of Parameters

As mentioned before, the edges detected by Canny method are very sensitive to the selection of the three parameters (*thigh*, *tlow*, *sigma*). Although a certain level of noise in the edge map does not affect the performance of our feature-based particle picking algorithm, the edges with completely wrong parameters may exclude some geometric features and hence some particles may be misrecognized. One way to reduce the sensitivity-to-parameter is to utilize multiple sets of parameters and the results from each of them are then combined for better results. We demonstrate this strategy in Fig.10, where two sets of parameters are used in Canny method. It is obvious that each set of parameters alone cannot correctly detect particles (one rectangular particle is wrongly recognized as circular particle). However, these two sets of parameters are complementary to each other in the sense that wrong particles in one case are correctly detected in the other. The combined particles shown in Fig.10 give correct result. Two sets of results are combined by inter-competition, which is similar to the competitions we introduced in Section 3. Each of the particles from one result competes with each of the particles from the other one if both particles intersect. The "winner" is again the one with smaller average distance value along its own shape contour. Fig.11 shows particles detected using three sets of parameters. One can compare it with Fig. 8 and see the improvements. A disadvantage of using multiple sets of parameters is that the computational time may increase by several times depending on how many sets of parameters are considered.

5.2 How Anisotropic Filtering Helps?

Edges play a crucial role in our particle detection approach but they are often corrupted by noise commonly seen in the electron micrographs. Therefore, it is expected that noise reduction would be in great demand as a pre-processing step. Traditional image filters include Gaussian filtering, median filtering, and frequency domain filtering [23]. Most of recent research has been devoted to anisotropic filters that smooth out the noise without blurring the geometrical details such as edges and corners. Several categories of anisotropic filters have been proposed in the area of image processing. Bilateral Filtering [28-30] is a straightforward extension of Gaussian filtering by simply multiplying an additional term in the weighting function. A partial differential equation (PDE) based technique, known as anisotropic heat diffusion, has also been studied [31, 32] and a fundamental relationship was discussed between this technique and the bilateral filtering [33]. Another popular technique for anisotropic filtering is by wavelet transformation [34]. The basic idea is to identify and zero out wavelet coefficients of a signal that likely correspond to image noise. Finally, the development of nonlinear median-based filters in recent years has also resulted in promising results. One of those filters is called mean-median (MEM) filter [35].

We have tested several methods for noise reduction, including Perona-Malik model [31], bilateral filtering [28] and anisotropic diffusion [36]. Fig. 12 shows the difference between the results on the original image and the results enhanced by bilateral filter [28] and by anisotropic diffusion [36]. The particle detection on the filtered data gives more accurate results due to the anisotropic filtering that smoothes the noise while preserving sharp edges.

It is worthy of noting that Gaussian filtering seems quite necessary in Canny edge detection even if the data is pre-smoothed by anisotropic filters. The reason is that Canny edge detector requires calculation of gradient vector for every pixel and the gradient directions should vary smoothly along edges. Sharp edges obtained by anisotropic filtering may make sudden changes of the gradient directions and hence bring difficulties to the edge-tracing. In addition, the gradient vectors are usually calculated within a small neighborhood (e.g., 3×3 pixels), which may also cause sudden changes of the gradient directions along edges. Gaussian filtering, however, can more or less alleviate this problem. An alternative way is to smooth the gradient vectors instead of the gray-scale intensities. For example, one can use gradient vector diffusion [37, 38].

5.3 Particles with Other Shapes

The particles we have been working on so far are restricted to circular and rectangular shapes. However, a similar pipeline also works for particles with other types of shapes if some kind of geometric features can be derived from the shapes. The general pipeline is described as follows:

1. From the given electron micrographs, determine the template shapes (e.g., radius for circle, width/length for rectangle, and so on).
2. Smooth original images using anisotropic filtering.

3. Apply Canny edge detection followed by edge cleaning.
4. Compute distance transform and Voronoi diagram of the edges.
5. Detect geometric features from the distance transform and/or Voronoi diagram. In case of circular/rectangular particles, the geometric features are centers/corners, respectively.
6. From the geometric features, guess the initial particles (including the centers and, if applicable, the orientations). Every candidate particle is attached with a “survival” factor, which is defined by the inverse of the average distance value along the particle’s contour.
7. Any two candidate particles, if they intersect, should compete with each other. The “winner” in a competition is the one that has a larger “survival” factor than its competitor. As described in Section 3, particle refinement may be necessary to improve the detection accuracy. In addition, multiple sets of parameters for Canny edge detection can be helpful.
8. The output is the final “winners” of all the particle competitions.

Detecting particles of arbitrary shapes is much more challenging. However, particle detection in the single particle analysis does not require accurate boundary segmentation. Edge detection, coupled with anisotropic filtering and/or distance transform, may still be a feasible way in such cases.

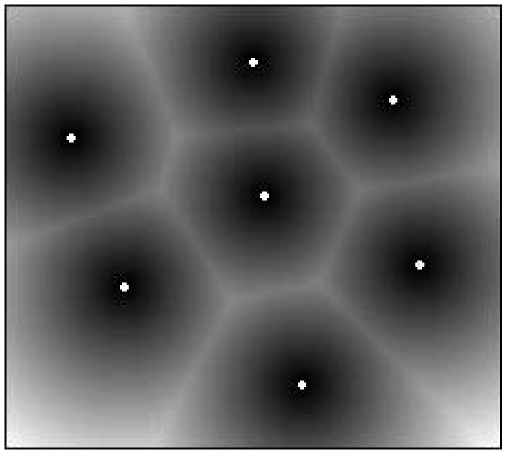
6. Conclusions and Future Work

In this paper we described an automatic method for particle detection from electron micrographs. Our method is based on shape matching between the edge map calculated from the original/filtered image and the template shapes of the true particles (what we call “virtual” templates). Unlike the conventional template-matching method that is based on cross-correlation of intensities, our method only needs the geometric features of the particles and hence avoids the manual and tedious generation of the template images. Multiple views of the particles are allowed by simply applying different geometric features to each view (e.g., the circular/rectangular examples as we considered in Section 3). In addition, with the help of the distance transform and Voronoi diagram, we can quickly identify the geometric features from the edges and make a good guess on the initial positions/orientations of the candidate particles. Good initialization is a crucial part of our algorithm in order to reduce the computational time as well as improve the detection accuracy. Since our method is based on boundaries (or edges), which are said to be the most important features of an image, other irrelevant details (or noise) have a very small influence on the detection of a particle from the original image. This “weighted” matching approach differs a lot from the conventional template-matching methods where all intensities around a pixel under consideration evenly influence the matching score.

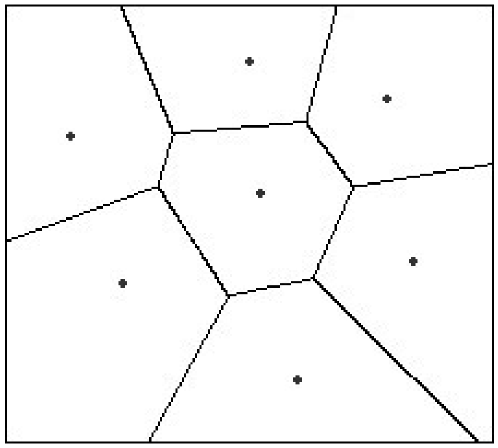
Part of our future work is *flexible shape matching*, meaning that, in particular, the radius of the template circle and the width/length of the template rectangle could be flexible. We expect that this could further reduce the sensitivity of our particle detection algorithm to the determination of template shapes. Another interesting work is that the geometric features of each particle may help a lot to enhance the *particle alignment* and *particle classification* in the single particle analysis.

Acknowledgments

This research is supported in part by NSF grants ACI-9982297, CCR-9988357, and from grant UCSD 1018140 as part of NSF-NPACI, Interactive Environments Thrust. Thanks are also due to Qiu Wu at UT-ECE for implementations of the bilateral and anisotropic diffusion filtering, and to Yuanxin Zhu, Bridget Carragher at The Scripps Institute for providing the KLH datasets.

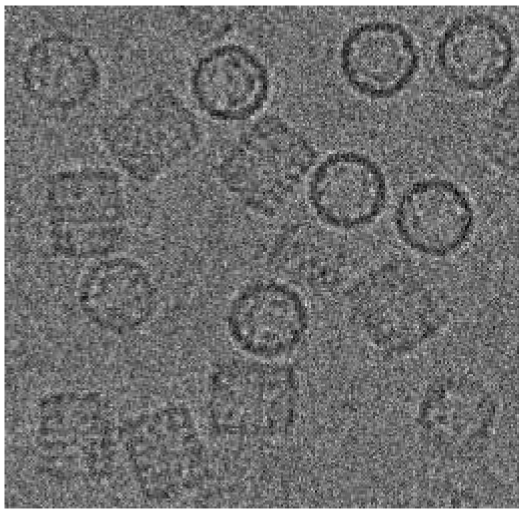


(a) Distance transform

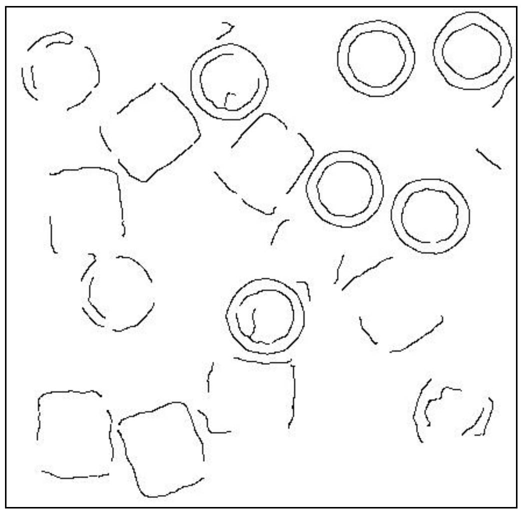


(b) Voronoi diagram

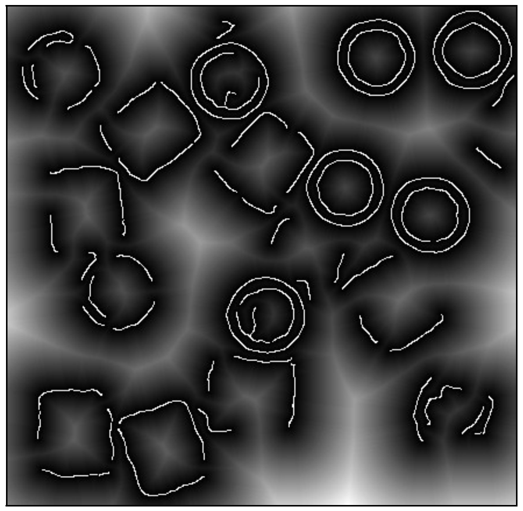
Figure 1 An example of distance transform and Voronoi diagram



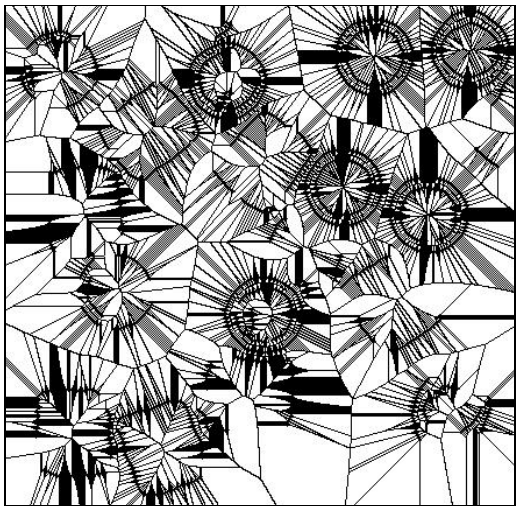
(a) Original image



(b) Edge map

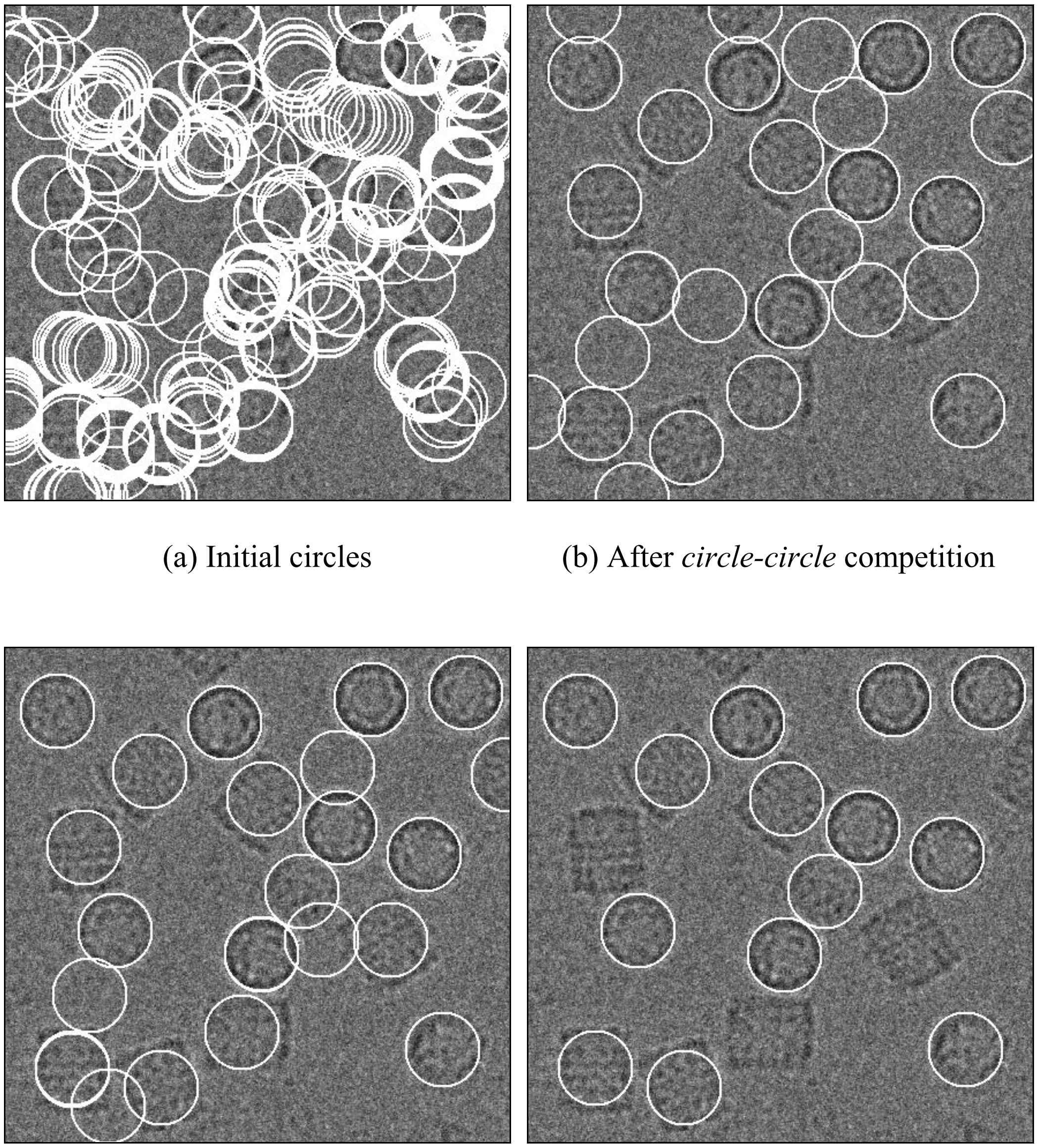


(c) Distance transform



(d) Voronoi diagram

Figure 2 Illustration of the edges, distance transform and Voronoi diagram. The edge map is obtained by Canny edge detector [22] followed by edge cleaning (we remove short edges with length below 20 pixels). The distance transform and Voronoi diagram are calculated by list-processing approach [24].



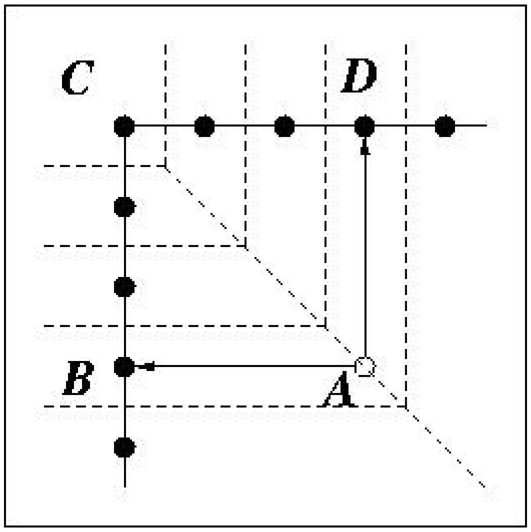
(a) Initial circles

(b) After *circle-circle* competition

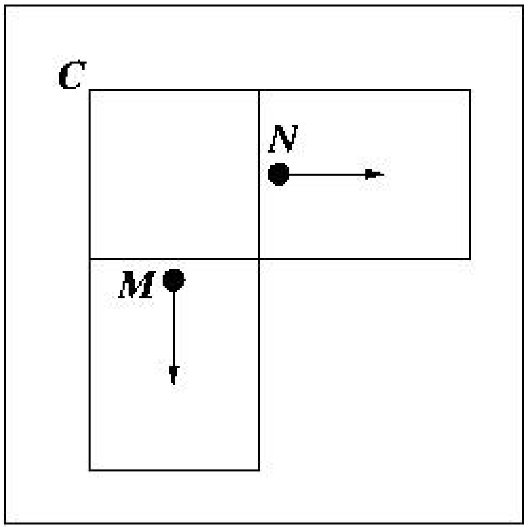
(c) After center refinement

(d) After further refinement

Figure 3 Illustration of the four steps seen in circle detection

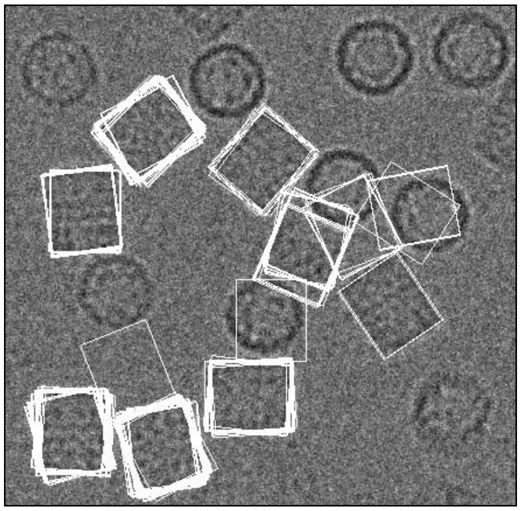


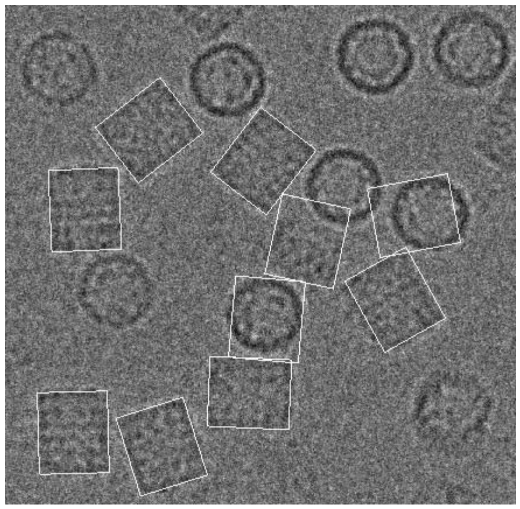
(a) Corner detection

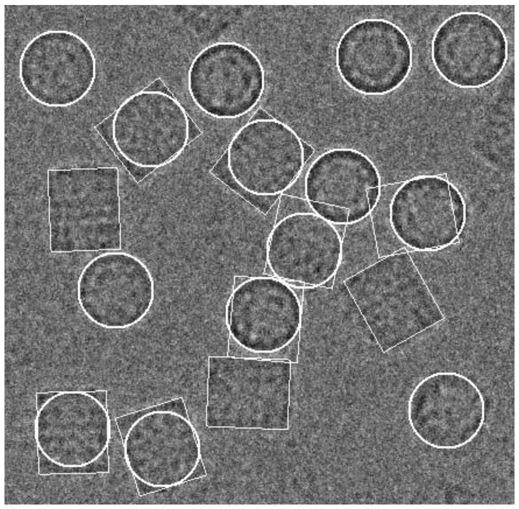


(b) From corner to center

Figure 4 Illustration of corner detection by Voronoi diagram and the two cases for converting corners to centers.


(a) Initial rectangles


(b) Refined rectangles


(c) Circle-rectangle combined

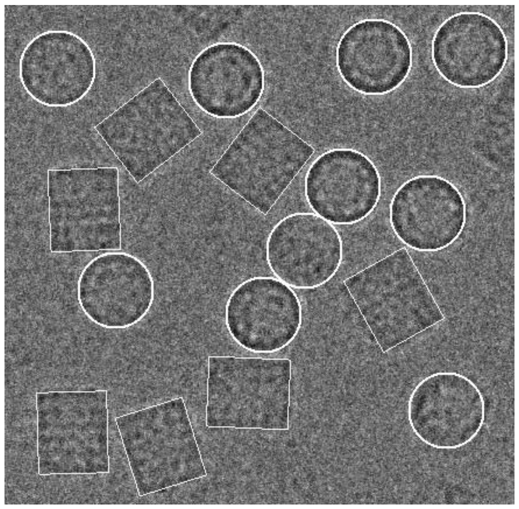

(d) Final results

Figure 5 Illustration of rectangle detection and the circle-rectangle competition

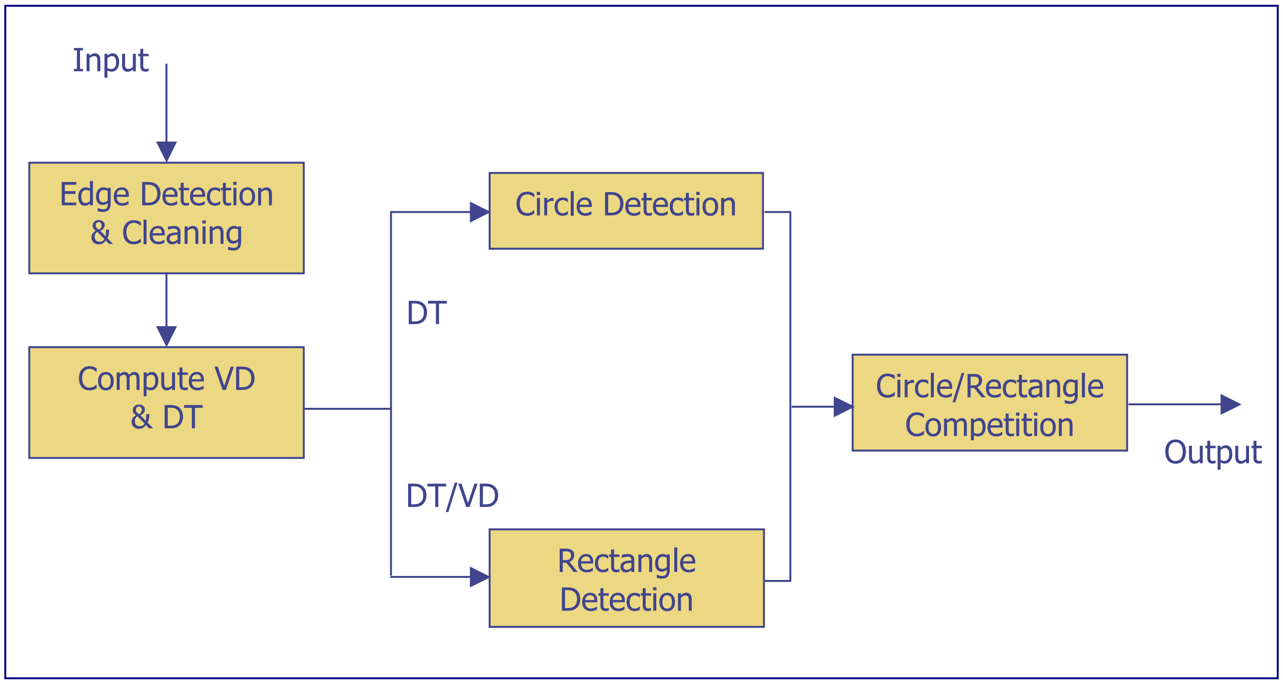


Figure 6 Pipeline of our particle detection approach.

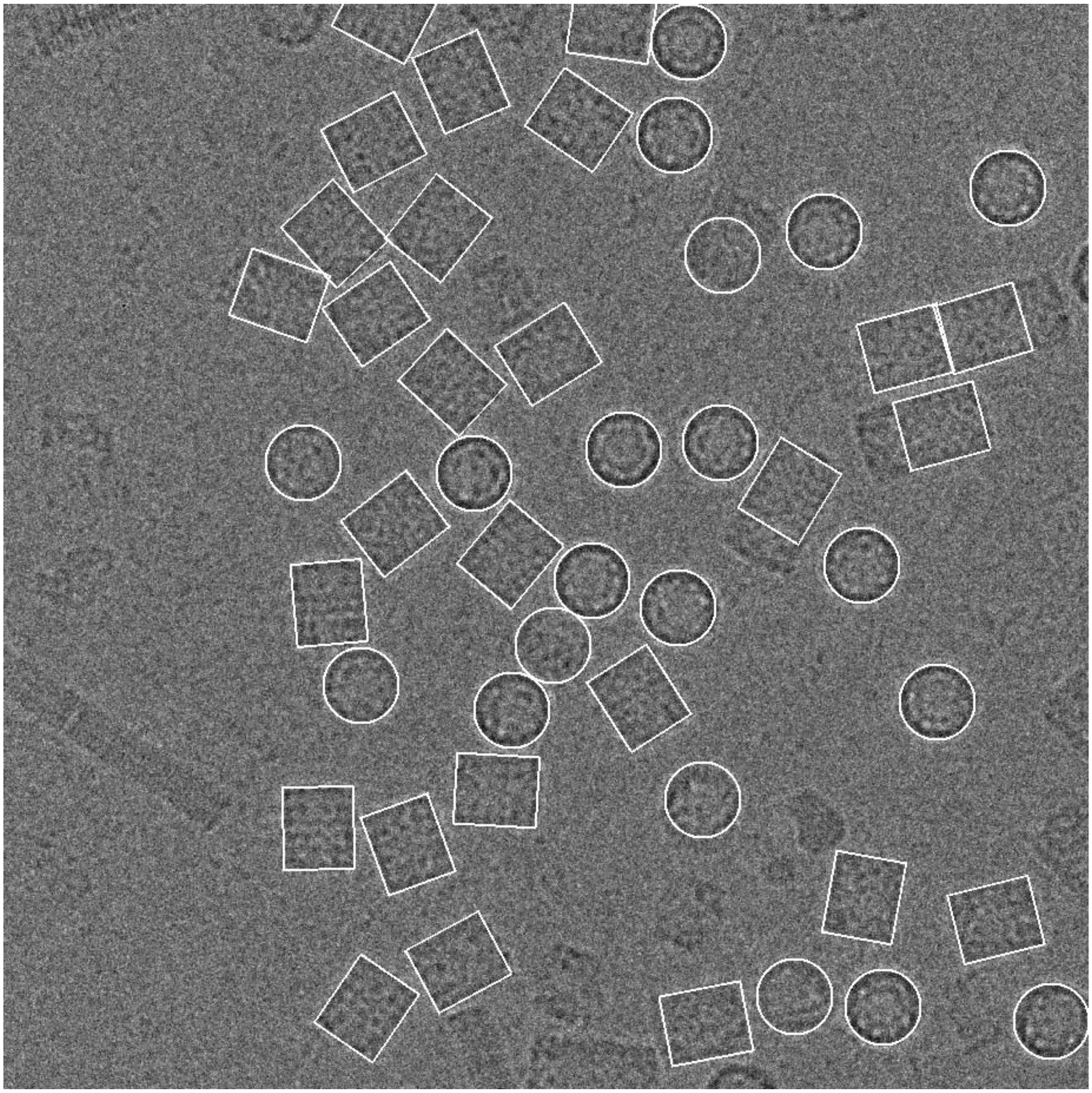


Figure 7 Example I:

Circles: **20** particles detected; **1** false positive; **0** false negative.

Rectangles: **27** particles detected; **0** false positive; **1** false negative.

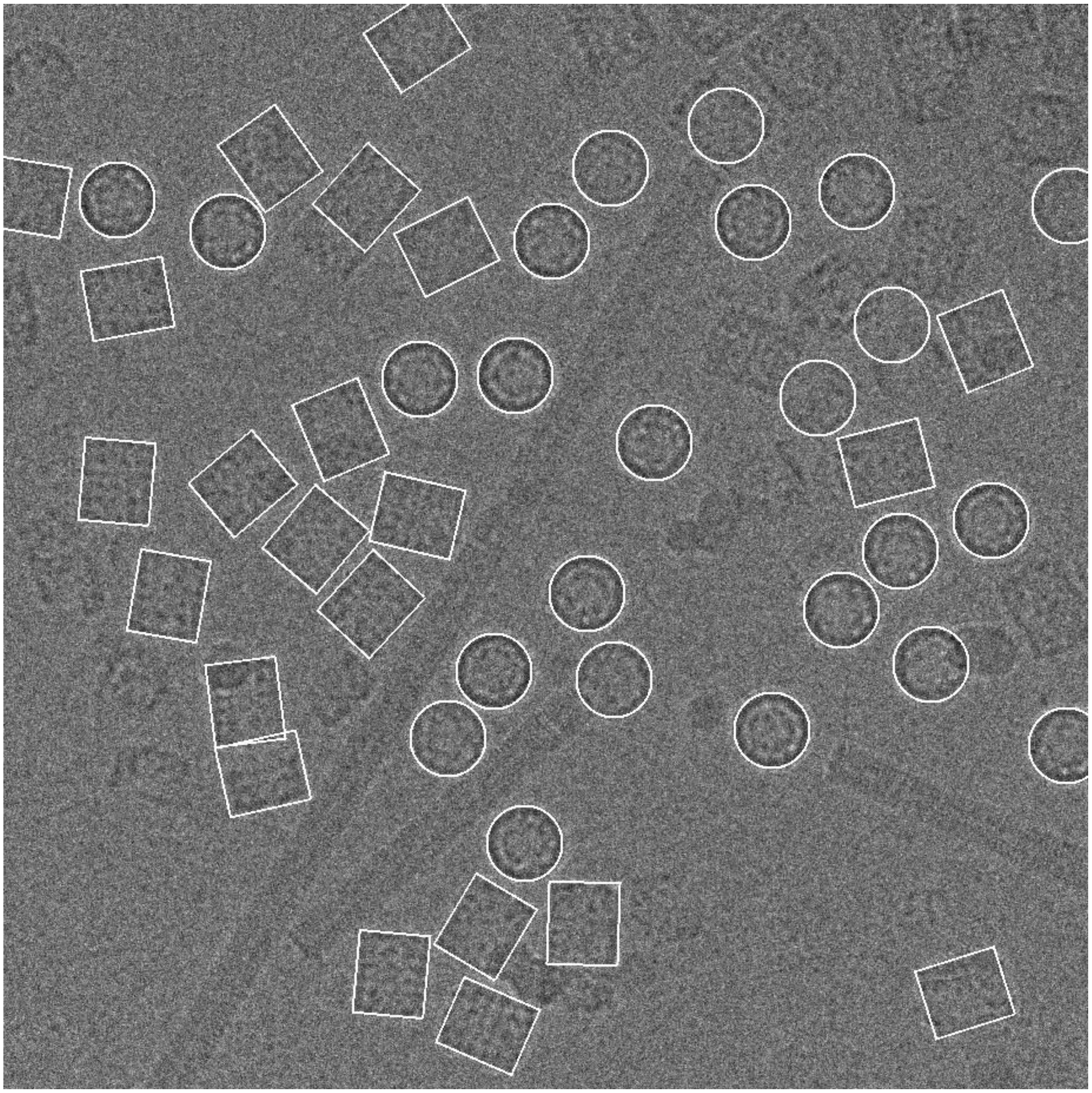


Figure 8 Example II:

Circles: **24** particles detected; **2** false positives; **1** false negative.

Rectangles: **22** particles detected; **1** false positive; **7** false negatives.

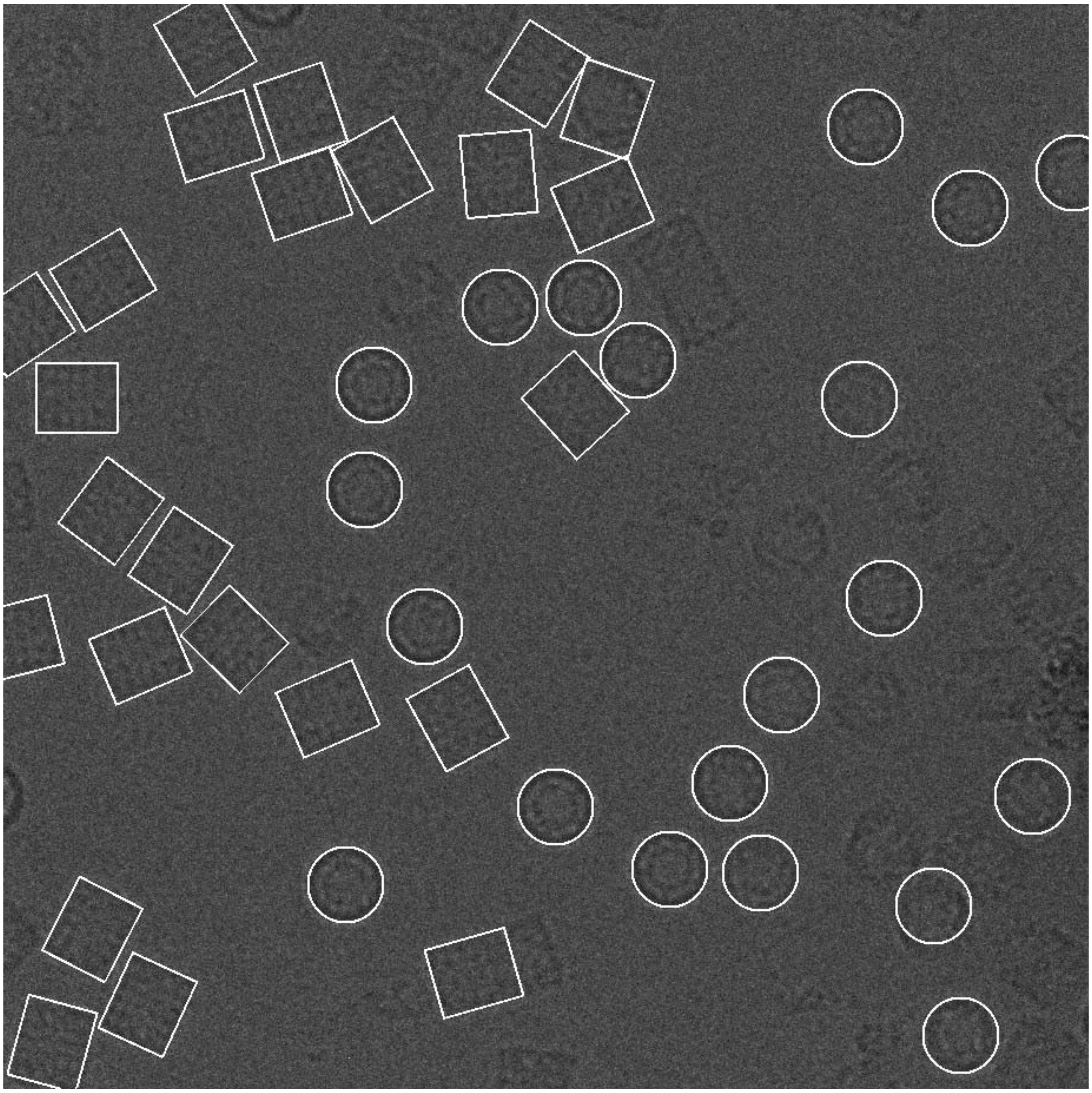


Figure 9 Example III (with low contrast):

Circles: **20** particles detected; **0** false positive; **4** false negatives.

Rectangles: **24** particles detected; **0** false positive; **3** false negatives.

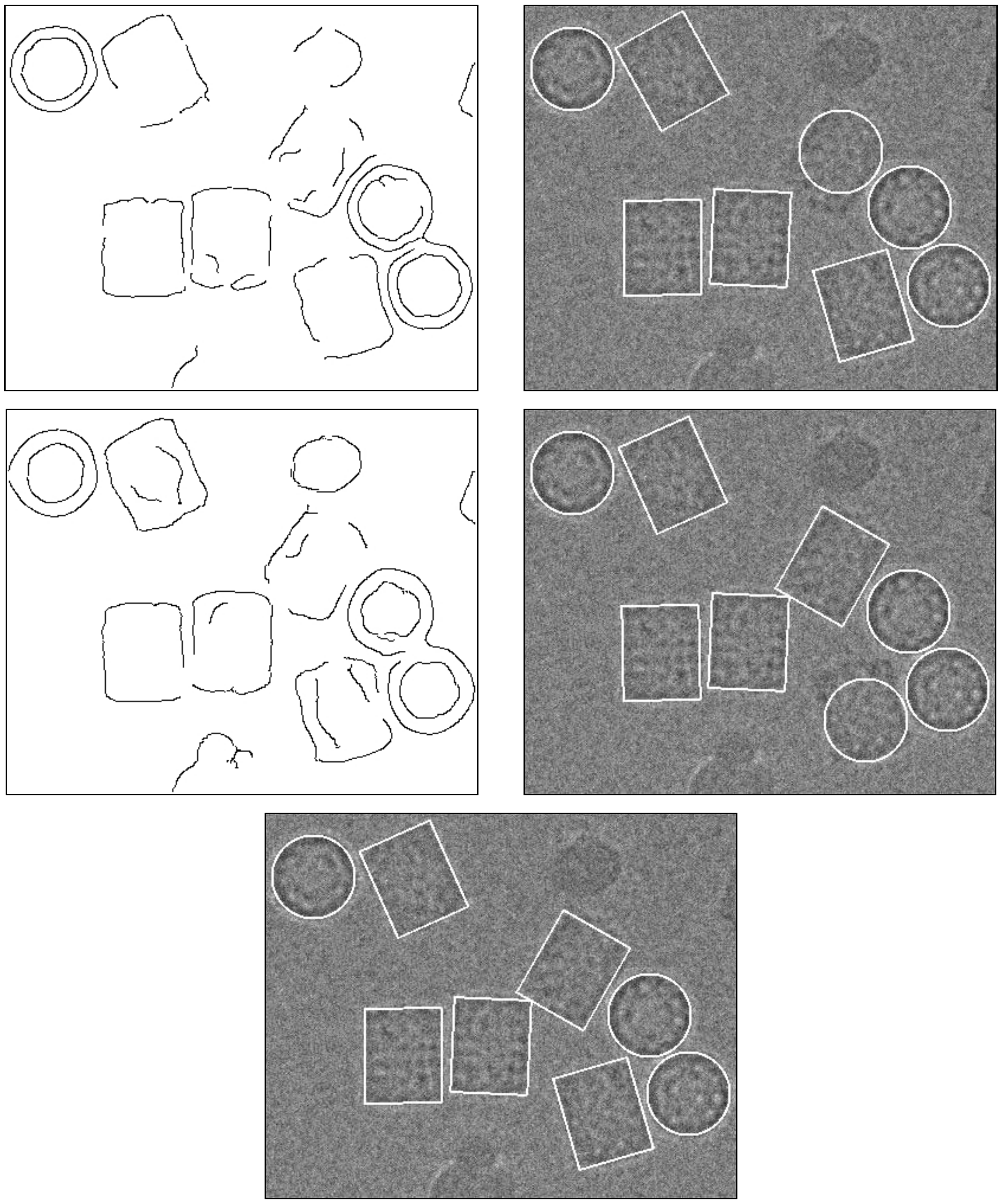


Figure 10 Particle detection can be enhanced by multiple sets of parameters in Canny edge detection. Top row: results (edges and particles detected) using one set of parameters (*sigma* = 4; *tlow* = 0.9; *thigh* = 0.8). Middle row: results using another set of parameters (*sigma* = 6; *tlow* = 0.7; *thigh* = 0.8). In both cases, there is one rectangle that is wrongly recognized as circle. Bottom row: combined results using the above two sets of parameters. All edges are cleaned by removing short edges with length below 20 pixels.

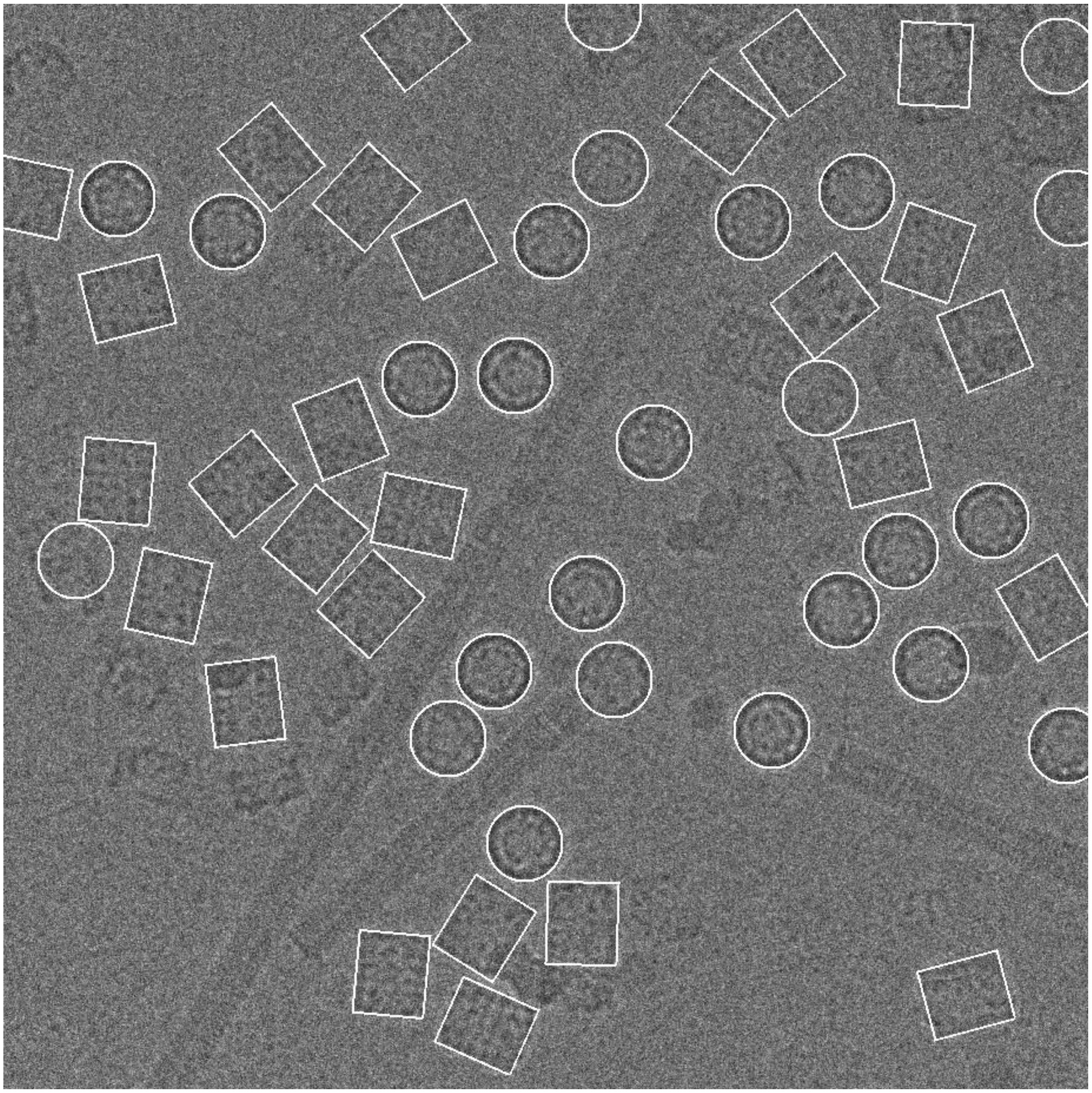


Figure 11 Particle detection is enhanced by three sets of parameters in Canny edge detection. Set I: *sigma* = 4; *tlow* = 0.9; *thigh* = 0.8. Set II: *sigma* = 4; *tlow* = 0.7; *thigh* = 0.8. Set III: *sigma* = 4; *tlow* = 0.85; *thigh* = 0.75. Edges are cleaned by removing short edges with length below 20 pixels.

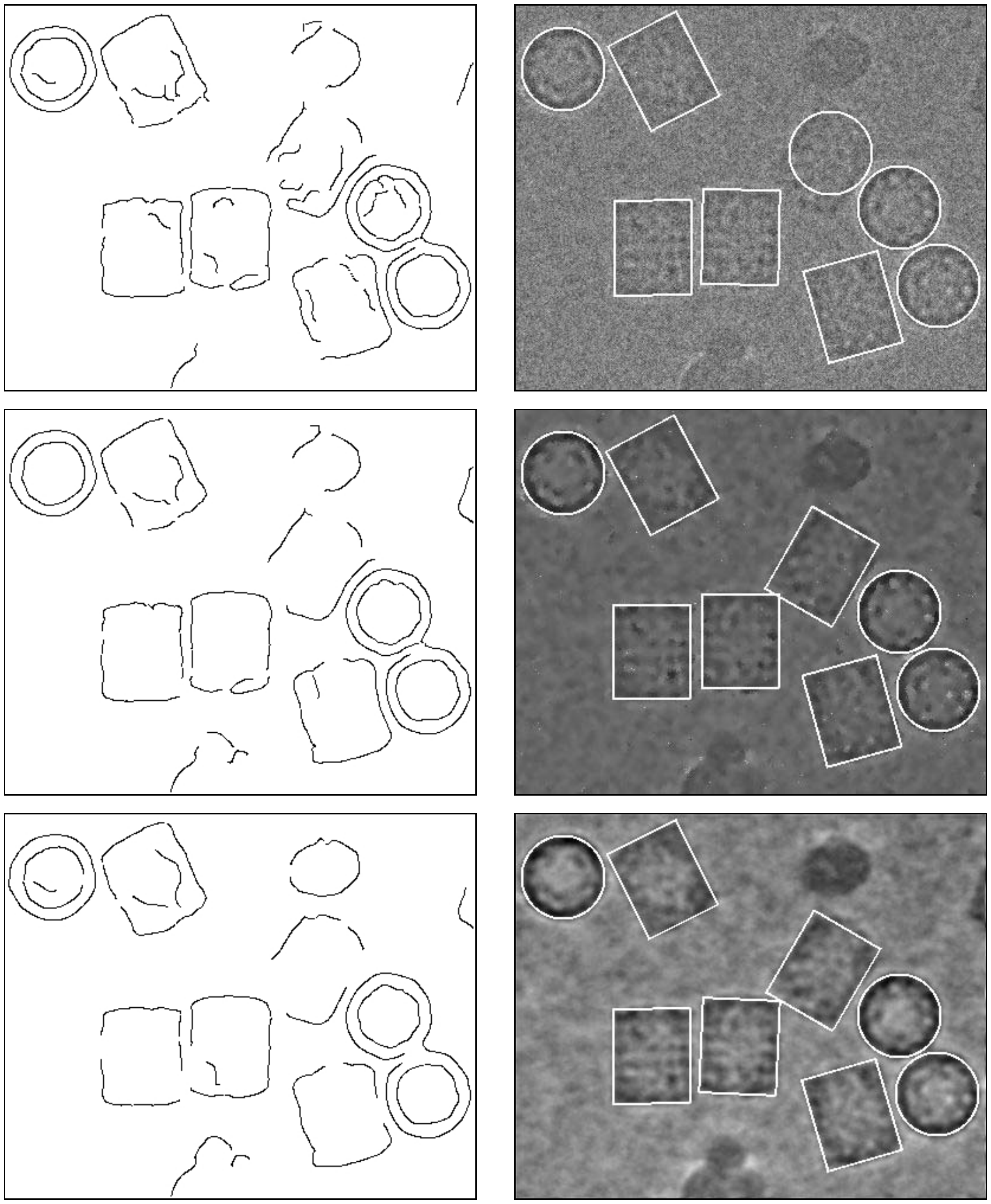


Figure 12 Particle detection can be enhanced by anisotropic filtering. Top row: results (edges and particles detected) without pre-filtering. Middle row: results (edges and particles) enhanced by bilateral filtering. Bottom row: results (edges and particles) enhanced by anisotropic diffusion filtering. In all three cases, the parameters for edge detection are chosen as: *sigma* = 4; *tlow* = 0.8; *thigh* = 0.8. All edges are cleaned by removing short edges with length below 20 pixels.

References

1. Baker, T.S., N.H. Olson, and S.D. Fuller, *Adding the third dimension to virus life cycles: three-dimensional reconstruction of icosahedral viruses from cryo-electron micrographs.* Microbiology and Molecular Biology Reviews, 1999. **63**(4): p. 862-922.

2. Frank, J., *Three-dimensional Electron Microscope of Macromolecular Assemblies*. 1996, San Diego: Academic Press.

3. Heel, M.v., et al., *Single-particle electron cryo-microscopy: to-wards atomic resolution.* Quarterly Reviews Biophysics, 2000. **33**(4): p. 307-369.

4. Henderson, R., *The potential and limitations of neutrons, electrons and X-rays for atomic resolution microscopy of unstained biological macromolecules.* Quart. Rev. Biophys., 1995. **28**: p. 171-193.

5. Nicholson, W.V. and R.M. Glaeser, *Review: automatic particle detection in electron microscopy.* Journal of Structural Biology, 2001. **133**(2-3): p. 90-101.

6. Heel, M.v., *Detection of objects in quantum-noise-limited images.* Ultramicroscopy, 1982. **7**: p. 331-342.

7. Frank, J. and T. Wagenknecht, *Automatic selection of molecular images from electron micrographs.* Ultramicroscopy, 1984. **12**: p. 169-176.

8. Thuman-Commike, P. and W. Chiu, *Automatic detection of spherical particles from spot-scan electron cryomicroscopy images.* J. of Micros. Soc. Am., 1995. **1**: p. 191-201.

9. Thuman-Commike, P. and W. Chiu, *PTOOL: a software package for the selection of particles from electron cryomicroscopy spot-scan images.* Journal of Structural Biology, 1996. **116**(1): p. 41-47.

10. Saad, A., W. Chiu, and P. Thuman-Commike, *Multiresolution approach to automatic detection of spherical particles from electron cryomicroscopy images.* Proceedings of IEEE International Conference on Image Processing, 1998: p. 8-10.

11. Ludtke, S.J., P.R. Baldwin, and W. Chiu, *EMAN: semiautomated software for high-resolution single-particle reconstructions.* Journal of Structural Biology, 1999. **128**(1): p. 82-97.

12. Stoschek, A. and R. Hegerl, *Automated detection of macromolecules from electron micrographs using advanced filter techniques.* J. Microsc. (Oxford), 1997. **185**: p. 76-94.

13. Zhu, Y., B. Carragher, and C.S. Potter, *Fast detection of generic biological particles in cryo-EM images through efficient Hough transforms.* Proc. 2002 IEEE International Symposium for Biomedical Imaging, 2002: p. 205-208.

14. Roseman, A.M., *Particle finding in electron micrographs using a fast local correlation algorithm.* Ultramicroscopy, 2003. **94**: p. 225-236.

15. Martin, I.M.B., et al., *Identification of spherical virus particles in digitized images of entire electron micrographs.* Journal of Structural Biology, 1997. **120**(2): p. 146-157.

16. Lata, K.R., P. Renczek, and J. Frank, *Automatic particle picking from electronic micrographs.* Ultramicroscopy, 1995. **58**: p. 381-391.

17. Kivioja, T., et al., *Local average intensity-based method for identifying spherical particles in electron micrographs.* Journal of Structural Biology, 2000. **131**(2): p. 126-134.

18. Ogura, T. and C. Sato, *An automatic particle pickup method using a neural network applicable to low-contrast electron micrographs.* Journal of Structure Biology, 2001. **136**: p. 227-238.

19. Andrews, D.W., A.H.C. Yu, and F.P. Ottensmerer, *Automatic selection of molecular images from dark field electron micrographs.* Ultramicroscopy, 1986. **19**: p. 1-14.

20. Harauz, G. and A. Fong-Lochovsky, *Automatic selection of macromolecules from electron micrographs by component labeling and symbolic processing.* Ultramicroscopy, 1989. **31**(4): p. 333-344.

21. Zhu, Y., et al., *Automated identification of filaments in cryo-electron microscopy images.* Journal of Structural Biology, 2001. **135**(3): p. 302-312.

22. Canny, J., *A computational approach to edge detection.* IEEE Transactions on Pattern Analysis and Machine Intelligence, 1986. **8**: p. 679-698.

23. Gonzalez, R.C. and R.E. Woods, *Digital Image Processing*. 1992: Addison-Wesley.

24. Guan, W. and S. Ma, *A list-processing approach to compute Voronoi diagram and Euclidean distance transform.* IEEE Transactions on Pattern Analysis and Machine Intelligence, 1998. **20**(7): p. 757-761.

25. Huttenlocher, D.P., G.A. Klanderman, and W.J. Rucklidge, *Comparing images using the Hausdorff distance.* IEEE Transactions on Pattern Analysis and Machine Intelligence, 1993. **15**(9): p. 850-863.

26. Rucklidge, W.J., *Locating objects using the Hausdorff distance.* Proc. 1995 IEEE Conference on Computer Vision, 1995: p. 457-464.

27. Gavrila, D.M., *Multi-feature hierarchical template matching using distance transforms.* Proc. 1998 International Conference on Pattern Recognition, 1998: p. 439-444.

28. Tomasi, C. and R. Manduchi, *Bilateral filtering for gray and color images.* In Proc. IEEE International Conference on Computer Vision, 1998: p. 836-846.

29. Durand, F. and J. Dorsey, *Fast bilateral filtering for the display of high-dynamic-range images.* In Proc. ACM Conference on Computer Graphics (SIGGRAPH), 2002: p. 257-266.

30. Elad, M., *On the bilateral filter and ways to improve it.* IEEE Transactions On Image Processing, 2002. **11**(10): p. 1141-1151.

31. Perona, P. and J. Malik, *Scale-space and edge detection using anisotropic diffusion.* IEEE Trans. on Pattern Analysis and Machine Intelligence, 1990. **12**(7): p. 629-639.

32. Weickert, J., *Anisotropic Diffusion In Image Processing*. 1998: ECMI Series, Teubner, Stuttgart, ISBN 3-519-02606-6.

33. Barash, D., *A fundamental relationship between bilateral filtering, adaptive smoothing and the nonlinear diffusion equation.* IEEE Trans. Pattern Analysis and Machine Intelligence, 2002. **24**(6): p. 844-847.

34. Xu, Y., et al., *Wavelet Transform Domain Filters: A Spatially Selective Noise Filtration Technique.* IEEE Trans. Image Processing, 1994. **3**(6): p. 747-758.

35. Hamza, A.B., et al., *Removing noise and preserving details with relaxed median filters.* Journal of Mathematical Imaging and Vision, 1999. **11**(2): p. 161-177.

36. Bajaj, C., Q. Wu, and G. Xu, *Level-set Based Volumetric Anisotropic Diffusion for 3D Image Denoising.* ICES Techical Report #301, University of Texas at Austin, 2003.

37. Xu, C. and J.L. Prince, *Snakes, shapes, and gradient vector flow.* IEEE Trans. Image Processing, 1998. **7**(3): p. 359-369.

38. Yu, Z. and C. Bajaj, *Image segmentation using gradient vector diffusion and region merging.* In Proc. International Conference on Pattern Recognition, 2002: p. 941-944.