A Cell Image Segmentation Algorithm By Simulating Particle Movement

Xijiang Miao

Computer Science and Engineering, University of South Carolina, 29208

Abstract. Accurate segmentation is the base of higher level image analysis. This project report describes an image segmentation algorithm based on simulating the movement of particles. Using cell images as example, this algorithm shows a satisfying quality and reasonable speed. Compare with watershed algorithm, this algorithm does not explicitly require smoothing process or put marks on the image. This algorithm is designed for cell image segmentation, but with proper extension, it can also can be used in images with other concave objects.

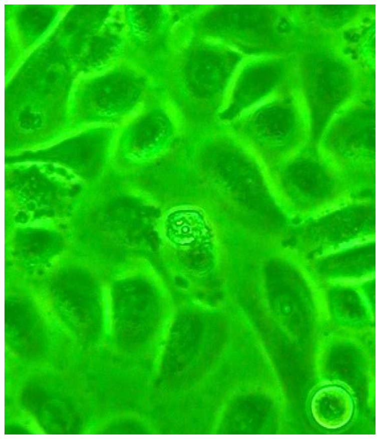
1 Motivation

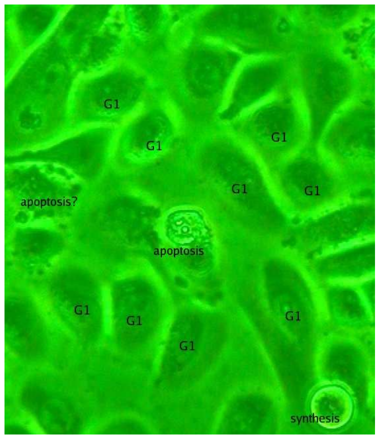
Because cell is the smallest and simplest living system can be manipulated in biology lab, cell level study is extremely important in biology research. Almost all molecular level study result will be bring back to cell level to be verified.

Observing cells under microscope is a routine job in biology lab. But traditionally, only qualitative report are given due to the cells are not easily to be traced by human eye. For example, counting the number of cells in a tray is simple enough, but usually only the order of magnitude is required in the report. The development of computer vision make it possible to observe the live cell quantitatively, which paves the way toward mathematical modeling of protein kinetics and biochemical signaling networks[2].

The morphological properties of cell can provides valuable information worth exploring. For example, in the life of a cell there are several check points. The changes of genes controlling such check points will probably lead to different fate of these cells, for example, apoptosis or imortalization. Because the shape of a cell is different in different phases of cell cycle, the change of check points is quite possible to be observed under microscope. Thus, tracing the morphological changes or movement of cells and their compartments will provide evidence of changes in molecular level.

Although reliable cell image analysis can considerably advance the biology technique, this goal is surprisingly not easy to achieve. One reason is that different types of cells have different shapes. For example, nerve cells tends to have a tree-like shape, while normal muscle cells will have spindle shape. The second reason is that even the cell type are the same, the different phase of the cell will make the cell looks quite different. For example, figure 1 shows cervical cells cultured in South Carolina Cancer Center. We can find the cells in synthesis phase takes the round shape and the cells in G1 phase will take quite different shapes.


(a) The input image



(b) Cells marked with its phase

Fig. 1. The input image

2 Related Work

The cell image segmentation problem can be tackled by some existing algorithms. Among those, voting based technique[1] and watershed algorithm[3] are two algorithms expected to be more successful in solving this problem than other algorithms.

Voting based technique tries to find a common property[1] with which most pixels or features in an image will agree. The typical procedure of a voting based technique is as follows. 1) Define a space of properties, for example a space of ϕ and θ in a line fitting case. (as shown in [4]) 2) For each pixel find its property(ϕ, θ) to vote. 3) Find the most voted property (ϕ, θ) to be the final solution. Our experimental result shows that direct use of the voting based technique is not ideal. Yang [1] improves the voting technique by extending the voting in to several rounds and each round tries to improve the final solution. According to the simple cell example they used in their paper, this improvement works well. But obviously, this heuristic method will suffer from the "local optima" problem, that is, when a property receives votes in the early rounds, it will tend to get more votes in the later rounds. That will make this method not very reliable.

Watershed algorithm is a widely used and well studies method, which simulates the process of raining. In this algorithm, the image is treated as a mountain like 3-D space. When it is rainning, some basin will be filled with water and merge with neighbour ponds when the waters overflow out of its basin. In this algorithm, the condition of merging of ponds is very subjective and very hard to adapt to the changing situations. And smoothing the input image in initialization stage is also crucial to this algorithm. The parameters of the smoothing process (σ in x, y, and intensity level respectively)

[1] Here property means the model to infer. Typically, property or model includes the objective function to fit with and its parameters.

can be tuned to control the number of segments. Once again, the tune of parameters is very subjective. Although there are some improvement in several aspects, such as parallelism, merge condition, the segmentation of cell image simply by watershed still needs some extra efforts. In this report, we tested the watershed algorithm by a watershed plugin[6] in the tool ImageJ[5].

3 Particle Movement (PM) Algorithm

3.1 Mapping the pixels into a particle model

Figure 2 shows part of the input image marked with the gradients of intensity by arrows.

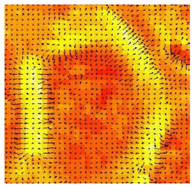


Fig. 2. The gradient of the gray scale.

One observation of this image is that the gradients are all pointing towards the center of corresponding cell body, which is also the basis of watershed algorithm. This is not a occasional phenomenon for cell image. When the lights comes from the bottom of the tray for cell culture, the cell body acts like a lens. With proper adjustment of the microscope, it is very easy to find a proper image letting the center of a cell to be darker than its peripheral area. This phenomenon can be used for the cell segmentation.

Intuitively, we can imagine the pixels in the cell image as particles in a force field. The particles have their own mass and velocity. And velocity of each pixel takes the direction of its gradient. If this force field mapped from cell image is the initial state, at next moment, the particles will move towards their own direction. That is, the particles at the boundary will move towards the center. During the process of movement, the particles will interact with each other in form of attraction and collision, and ideally they will meet in the center of the cell at the end of this process. The particles aggregated in the final state will belong to the same segment.

This process is different from voting based technique in that it will involve not only the result of aggregate but also moving the objects which will decide the segmentaion result. This particle movement (PM) algorithm potentially can deal with more complex situation than voting technique based algorithms. Comparing with watershed algorithm, this algorithm does not explicitly require smooth process. In the initial stage of this algorithm, the pixels at the boundary of an image will move towards their own center without or less considering their surrounding. In the later process, because the collision and attraction simulated in this algorithm, the movement involves stronger and stronger averaging of the surrounding pixels. Essentially, this is the process of smooth. So, this algorithm in fact smoothing the image at different level during the process of segmentation.

In following subsections, we will discuss the detail formulation of the movement simulation.

Parameters and Operations In the particle movement algorithm (PM), there are 3 parameters are maintained for each particles, i.e., position, mass, and velocity. And there are 2 operations are defined, i.e., collision and attraction, which will be described in section 3.1.

The position of particles are specified by the coordinates of a pixel, which are a pair of positive integers, and the offset relative to this pixel coordinates, which are a pair of real number in $[-0.5,\ 0.5)$. That is, position of particle p_i

$$p_i = (x_i + ox_i, y_i + oy_i)$$

where $x_i, y_i \in \mathit{positive\ integer}$, $ox_i, oy_i \in [-0.5,\ 0.5)$ and i is the index of the particle.

The reason for this definition lies in 1) It's relatively easy to store and access the position of a particle, 2) The checking of the collision is greatly simplified, 3) This representation is much more precise than only using the integer coordinates of an image.

The velocity describes the movement of a particle. The initial velocity of a particle is defined as the gradient of the image at position of corresponding pixel, in terms of magnitude and direction. That is,

$$\boldsymbol{v} = \left(\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}\right) = \nabla I$$

And the value of a velocity is subjected to change in the process of movement, which will be described in section 3.1.

The mass describes the mathematical weight of a particle. The value of mass decides the degree of a particle affecting the movement of its surrounding particles.

Interpretation and Initialization The purpose of defining *mass* is to let particles move more accurately. There should be several ways to initialize and change the mass for such a purpose. In PM algorithm, a very simple but effective method is used as follows.

The initial mass is assigned with the gradient, that is, same as the initial velocity. Obviously, the noise of the image will contribute a lot to the initial mass.

To deal with this noise problem, we can weighted average the with the surrounding mass during the moving process. The weight of the average in PM algorithm is shown below.

In figure 3, a, b, c are the weight of averages. And c is in the position of the particle in consideration. The mass is averaged in

$$m_f = \frac{a \times \sum_{k=4\ corners}(m_k) + b \times \sum_{k=4\ adjacents}(m_k) + c \times m_{center}}{4 \times a + 4 \times b + c}$$

where m_f is the final interactive mass of m_{center} and $a < b < c$.

a	b	a
b	c	b
a	b	a

Fig. 3. The weight of averaging mass m_{center} with its surroundings

If the noise comes from an independent identical Gaussian distribution, it's easy to show that the average result will reduce the noise to som extent.

A more sophisticated way of averaging is to check the consistency of velocities directions of the surrounding particles. It should further improve the accuracy of the moving mass. And another possible assignment is to consider the p-values of the coefficients of some linear regressions, which is based on a well developed statistical model.

The position of the particles involves two parts, the integer coordinates part and the real number offset part. If the movement of particles does not involves with a collision, the positions move along the direction of their respective velocity. Once there is a collision, and if the resolution of the image is high, the position changes due to the collision will not contribute very much to the final result. So it can be ignored. If the resolution is not high, the position change due to collision can take the mass weighed average of the positions of particles before collision. The experimental result shows that the position changes due to collision are not very important, so assigning the merged position with the center of destination is good enough.

Movement There are two operations of a particle, *collision* and *attraction*. The place of particles are represented by an integer part and a fraction part along its x axis and y axis. Only considering the fraction part, one particle can move within a 1-by-1 small space. Within that space, the particles are considered to move along its velocity and at its own speed. If a particle move out of its space to another small space, the integer and fraction part of its position should be adjusted. If the destination small space already has one particle, these two particles will have a *collision* and will be *merged*.

During the movement, the direction of movement can be adjusted according to their neighbours. It is called *attraction* in this algorithm.

Collision and merge When two particles collide, we think there will be a non-elastic collision, as shown in figure 4.

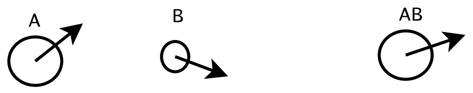


Fig. 4. The movement of particles.

According to the conservation of momentum, we have

$$m_{\boldsymbol{A}} \boldsymbol{v_A} + m_{B} \boldsymbol{v_B} = (m_{A} + m_{B})\, \boldsymbol{v_{AB}}$$

$$\boldsymbol{v_{AB}} = \frac{m_{\boldsymbol{A}}\boldsymbol{v_A} + m_{\scriptscriptstyle B}\boldsymbol{v_B}}{m_{\scriptscriptstyle A} + m_{\scriptscriptstyle B}}$$

From the intuitive meaning, the merged velocity depends on the weighted average of their mass, or reliability. We can extend the formula to 3 or more particles involved:

$$\boldsymbol{v_{merged}} = \frac{1}{\sum_{i=1}^{n} m_i} \sum_{i=1}^{n} (m_i \boldsymbol{v_i})$$

If the original variance of the velocity has variance of $var(\boldsymbol{v})$, the variance after collision is

$$var(\boldsymbol{v_{merged}}) = \sum_{i=1}^{n} \left(\frac{m_i}{summ}\right)^2 var(\boldsymbol{v_i})$$

where $summ = \sum_{i=1}^{n} m_i$.

If we consider the average case, that is, $var(\boldsymbol{v_i})$ and m_i are independent identically distributed, we will have

$$var(\boldsymbol{v_{merged}}) = \sum_{i=1}^{n} \left(\frac{1}{n^2} var(\boldsymbol{v_i})\right) = \frac{1}{n} var(\boldsymbol{v_i})$$

that is, the variance of the merged velocity is reduced along the merge process. It's worth noting that this process is also a process of smoothing.

Attraction The above process happens only when two particles collides, which is rare comparing with a normal movement without collision. So we need another process to improve the accuracy along the movement. Consider figure 5, we have one particle in the center moving along the red arrow. Obviously, it will move away from the whole group if there is no collision happened, which is quite possible. The attraction is designed to tackle this problem by considering the relationship between a particle and its neighbours.

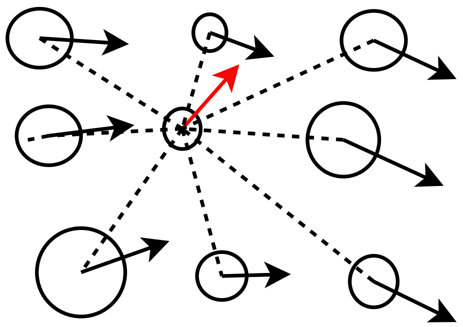


Fig. 5. The interaction between particle and its neighbours

The attraction process of in PM algorithm performs like the collision process. :

$$\boldsymbol{v}_{attraction} = \frac{1}{\sum_{i=1}^{n} d_i m_i} \sum_{i=1}^{n} (d_i m_i \boldsymbol{v_i})$$

where $d_i$take the arrangement of a, b, c defined in figure 3 and normalized to 1.

Follows the variance analysis of previous section, we know this process will also reduce the variance of movement. This process is much more possible to happen during the overall process of segmentation.

Terminate conditions Termination condition is another important choice for this algorithm. Because the summation of forces within segmentation can be non-zero, the final merged particle can still move. The termination of PM algorithm is just as crucial as the merging of ponds in watershed algorithm, although it is easier to be controled.

The possible choices are: 1) limited steps, 2) sand-box.

limited steps One simple termination condition is to let the movement stop after a number of steps. This is the choice of experimental result in section 7.

Sand-box It can be observed in the experiment that the first few steps will outline the overall shape of the segmentation. We can exploit this phenomenon by putting a higher confidence at the earlier steps and lower confidence at the later steps. This is a more sophisticated process under consideration and some more issues are to be addressed to make it practical.

3.2 The Paricle Movement (PM) Algorithm

3.3 Analysis of the algorithm

This section is only for the completeness of the report. The initial implementation of this algorithm is polynomial but in high order. But taking the possible acceleration into consideration, this process should not be slow. In terms of space complexity, it require space in the order of $O(p)$, where p is the number of pixels.

4 Experimental Result

4.1 The input image

The input image, as shown in figure1, is from Dr. Kim Creek's lab in South Carolina Cancer Center. Another more complex image is from a banner of a shop, which is not shown here because the limitation of the space of this report.

Algorithm 1 The particle movement (PM) algorithm

```
Input: A gray scale image of cells
Output: The segmentations of such cells
Begin
  1. Initialize the mass and speed according to section 3.1.
  2. Repeat
     (a) Move particles at their speed and along their directions
     (b) If particles collides
          i. Recalculate the combined speed and mass according to section 3.1
     (c) End if
     (d) Calculate the attraction according to section 3.1
     (e) Record their paths
  3. Until some terminate condition
  4. Segment the image according to the paths
End
```

4.2 The watershed algorithm result from ImageJ to compare with

ImageJ[5] is a software from (National Institute of Health), includes some very useful tools for the biology image processing. Watershed plugin for ImageJ is from [6] and can be freely downloaded.

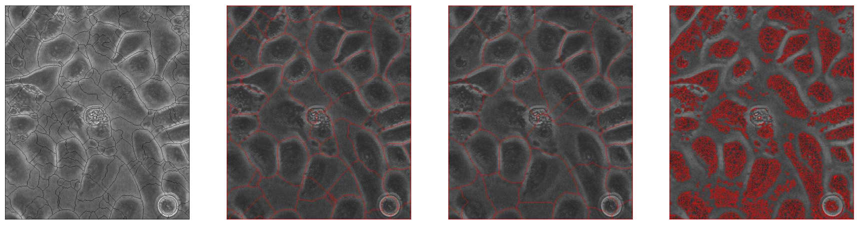

(a) Smooth = 5 (b) Smooth = 10 (c) Smooth = 20 (d) No smooth, Stop in the middle

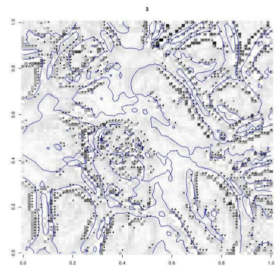
Fig. 6. The output of watershed algorithm in package ImageJ

Figure 6 shows the result of [6]. Panel (a), (b) and (c) is the result of watershed algorithm by smoothing with $\sigma = 5, 10, 20$ beforehand respectively. And panel (d) is without smoothing but stoped in the middle of watershed segmentation. The red or black solid lines are the dams built in watershed algorithm, or the boundary of segmentation.

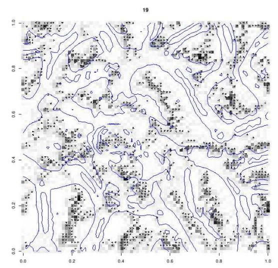
From above result, we can make following observations: 1) The number segmentations are very sensitive to the level of smoothness. 2) Some parts of the segmentation are separated without any good reason. 3) The process itself simulates the process of rising of water level, which is a good compensation of the PM algorithm described.

4.3 The result of PM algorithm

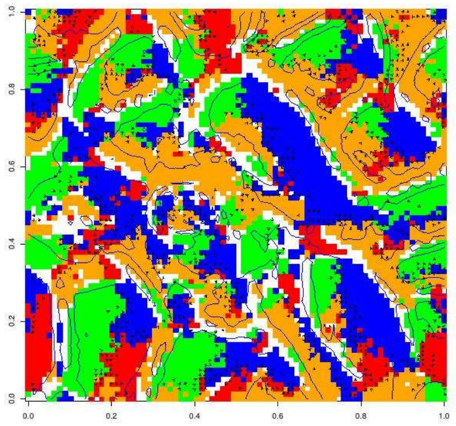
The following figure is the result from PM algorithm as described in section 3.2. It is worth noting that this result is from a simple implementation of this algorithm. With fine tune of parameters and enhanced features, the result is definitely better than what is shown here.



(a) Particles at the beginning of movement.



(b) The particles at the end of movement



(c) The colorizing result.

Fig. 7. The result of particle movement (PM) algorithm

The panel (a) and (b) in figure 7 show the particles which are moving along their direction. The arrows show their direction and the darkness of pixels showes their mass. Panel (a) is in the beginning stage while panel (b) is in the ending stage. Panel (c) shows the colored segmentation of the result. In panel (c), the colors shows the direction of the movement. That is, green is from left to right, blue is from left to right, and so on. A segmentation is composed of an ordered colored blocks. The contour lines shows the position of the cells. Because the limitation of tools and knowledge, a "real" colored segmentation is not drawn here. The coarse resolution is due to some compress of the input image for speed.

Compare the final colored segmentation with the input image marked with phases, we can realize that although without smooth process, we can identify most cells except those in apoptosis phase, which is very difficult for image segmentation.

This algorithm is not limited in the cell image segmentation. Although there are no elaborately test in concrete example, it should be able to segment images with similar characteristics, that is, with a concave shape in the intensity axis.

5 Conclusion

The PM algorithm is promising for cell image segmentation, in that it can segment the cell image with satisfying quality and reasonable speed. It is also simple enough that I can implement it without using any sophisticated tools. With proper knowledge and effort, this algorithm should be able to be extended to other image segmentation problems.

References

1. Yang, Q. et al, *Perceptual Organization of Radial Symmetries*, Proceedings of CVPR, 2004
2. R. Elis and C. Athale, *Computational imaging in cell biology*, JCB, Vol 161, No 3, May, 2003 477-481
3. Vincen t, L. and Soille, P. *Watersheds in Digital Spaces: An Efficient Algorithm Based on Immersion Simulations.* IEEE Transactions on pattern analysis and machine intelligence, Vol. 13, No. 6, June 1991
4. Forsth and Ponce. *Computer Vision: A modern approach*. Pearson Education.
5. National Institute of Health, *ImageJ*, http://rsb.info.nih.gov/ij/
6. Biomedical Imaging Group, *Watershed Plugin*, http://bigwww.epfl.ch/sage/soft/watershed