Segmentation and Border Identification of Cells in Images of Peripheral Blood Smear Slides

Nicola Ritter
School of Information Technology
Murdoch University
Perth, W.A., Australia, 6150
n.ritter@murdoch.edu.au

James Cooper
Department of Computing
Curtin University of Technology
Perth, W.A., Australia, 6102
jc@cs.curtin.edu.au

Abstract

We present an unsupervised blood cell segmentation algorithm for images taken from peripheral blood smear slides. Unlike prior algorithms the method is fast; fully automated; finds all objects—cells, cell groups and cell fragments—that do not intersect the image border; identifies the points interior to each object; finds an accurate one pixel wide border for each object; separates objects that just touch; and has been shown to work with a wide selection of red blood cell morphologies. The full algorithm was tested on two sets of images. In the first set of 47 images, 97.3% of the 2962 image objects were correctly segmented. The second test set—51 images from a different source—contained 5417 objects for which the success rate was 99.0%. The time taken for processing a 2272x1704 image ranged from 4.86 to 11.02 seconds on a Pentium 4, 2.4 GHz machine, depending on the number of objects in the image.

Keywords: erythrocyte, segmentation, border detection, graph algorithm, blood cell.

1 Introduction

For both animals (Reagan, Sanders & DeNicola 1998) and humans (Bell 1997), slides of stained peripheral blood smears are examined to aid diagnosis. The slides include three types of cell: white blood cells (leukocytes), red blood cells (erythrocytes), and platelets. Analysis of these slides by technicians aims to identify pathological conditions that cause changes in the blood cells. For red blood cells (RBC) these changes include incursions (Reagan et al. 1998), as well as changes in cell shape, size and colour (Bessis 1973, Bessis 1977). There may also be erythrocyte fragments (shistocytes) (Lesesve, Salignac, Alla, Defente, Benbih, Bordigoni & Lecompte 2004), joined erythrocytes (rouleaux) (Reagan et al. 1998) or aggregated erythrocytes (agglutination) (Foresto, D'Arrigo, Carreras, Cuezzo, Valverde & Rasia 2000). The changes are linked to specific diseases and conditions (Bacus, Belanger, Aggarwal & Trobaugh Jr. 1976, Bell 1997), making this an important diagnostic indicator. Slides are therefore examined to enable classification of the red blood cell morphology (Walton 1973).

The potential use of automated blood slide examination is large. Ingram and Preston Jr. (1970) estimated that in the US alone there would be over 1 million slide examinations every day. By 1976 the estimate had increased to over 50 million a day (Preston Jr. 1976). This number could be expected to have grown exponentially along with population since then.

However examination of slides by humans has a number of problems. It is time consuming (Preston Jr. 1976, Rowan 1986, Di Ruberto, Dempster, Khan & Jarra 2002) and therefore expensive (Ingram & Preston Jr. 1970). It is subjective (Connors & Wilson 1986, Payne, Bridgen & Edora 1996) and prone to human error (Bacus 1971, Fairbanks 1971), which means that it is difficult to get consistent results from the examination. Human analysis usually only results in a qualitative description of the presence of various morphologies and gives no quantitative results (Robinson, Benjamin, Cosgriff, Cox, Lapets, Rowley, Yatco & Wheeless 1994). This makes it difficult to track the progress of the condition and its treatment.

Currently there are several instruments that do partial differential counting of erythrocytes. A typical example of such an instrument is the DM96, produced by Cellavision (*CellaVision DM96 Technical specifications* 2006). However even this state-of-the-art instrument can only pre-characterise the red blood cells based on six general classifiers: polychromasia, hypochromasia, anisocytosis, microcytosis, macrocytosis, and poikilocytosis. More precise classification into the 30+ possible RBC morphologies (Bessis 1977, Reagan et al. 1998) is still done manually. Currently over 16% of automatically scanned slides are manually reviewed afterwards (Novis, Walsh, Wilkinson, Louis & Ben-Ezra 2006).

To be useful, a computer-based red blood cell differential count will need to be fully automatic (Rowan 1986), faster than a human observer, and at least as accurate as a human observer.

2 Literature Review

The history of research into automated blood slide examination dates back to 1975 (Bentley & Lewis 1975). However it is only recently that digital photography, computer speed, RAM size and secondary storage capacity have made the reality possible.

Analysis of blood slides must be fully automated to be useful (Rowan 1986). However the difficulty of the processing involved (Costin, Rotariu, Zbancioc, Costin & Hanganu 2001) has mostly limited papers to comparisons based on red cells segmented

This work was supported by a grant from the Division of Arts at Murdoch University. The slides and the first set of images were provided by A.Prof Phillip Clark of the Department of Clinical Pathology at Murdoch University. The second set of images were taken by Stanton Smith.

either manually (Bentley & Lewis 1975, Albertini, Teodori, Piatti, Piacentini, Accorsi & Rocchi 2003) or semi-automatically (Bacus et al. 1976, Robinson et al. 1994, Dasgupta & Lahiri 2000, Costin et al. 2001, Gering & Atkinson 2004) from an image. Furthermore, in images from peripheral blood smear slides, the shading of the interior of the red blood cell as well as the overall shape and size of the cell (Reagan et al. 1998) exhibit meaningful variation. This necessitates identification of all points interior to the cell as well as accurate identification of the cell border. However no algorithm so far published finds both the border and the interior points of red blood cells, fully automatically.

Thresholding has been used to pre-process images as an aid to segmentation (Gonzalez & Woods 2002). However with red blood cell images this causes problems due to the pale nature of the interior of the cells, which then necessitates further processing. Adjouadi and Fernandez (2001) find the cell borders using eight-directional scanning within thresholded images of normal blood. However the method would not find the whole of the border of severely deformed cells such as those in the image of Fig. 1(b), as these contain edge points that would not be reached by any of the eight scan-lines. Moreover the process does not result in identification of the points within the contours.

Di Ruberto *etal.* (2000) follow thresholding with a segmentation method using morphological operators combined with the watershed algorithm. However their work is aimed at segmentation of red blood cells containing parasites and is designed to increase the compact nature and roundness of the cells. Such assumption of roundness is not appropriate for segmentation of RBC images for the purpose of classification, because these may contain deformed red blood cells as well as red blood cell fragments, for which the preservation of shape is important. Their method is also complicated—requiring 9 intermediate steps—and does not result in border identification.

Classical edge operators such as Sobel or Canny produce multiple thick edges as well as multiple edges interior to cells (Adjouadi & Fernandez 2001). Furthermore such an operator is merely a pre-processing technique which leaves the actual edge detection yet to be done. Some success has been found using graph theory (Martelli 1976, Pope, Parker, Clayton & Gustafson 1985, Fleagle, Johnson, Wilbricht, Skorton, Wilson, White, Marcus & Collins 1989, Fleagle, Thedens, Ehrhardt, Scholz & Skorton 1991) to navigate around edge pixels found in an edge image. However this work has involved images of single objects manually located in an image, and does not address the problems of multiple objects in the image; object location; removal of extraneous edges (internal to the cell); or the selection of suitable starting and ending points for the graph search.

Another approach to border detection is that of active contours, or snakes (Kass, Witkin & Terzopoulos 1988), which can be applied either to the original image or to an edge image. However when used to identify cell borders, the resulting contours do not correspond with the exact borders of the cells (Ongun, Halici, Leblebicioglu, Atalay, Beksac & Beksac 2001, Wang, He & Wee 2004), which would cause problems with later RBC classification, where the exact boundary shape is important. Other problems with the use of contours for images of peripheral blood smear slides include the initial positioning of the multiple contours required; the tendency of the contours to find the inner pale cell areas as well as or instead of the outer edges; and the failure of contours to identify pixels interior to the contour.

We present a fully automated algorithm that locates every object—cell or cell group—in an image from a peripheral blood smear slide. For each object it identifies both the pixels within the cell and a 1-pixel wide border of the cell.

3 Materials

As almost all mammalian red-cells are similar in shape, we use canine blood cell slides for our processing as canines have the largest cells of the non-human mammals (Reagan et al. 1998). Their blood slides are also more easily obtained than those of human blood.

The slides were made using the 'wedge technique'. They were then air dried, fixed in alcohol and stained using Wright's and Giemsa stain. The slides were mounted in a Nikon eclipse microscope and viewed at 100x magnification with oil immersion. Parts of the slide were then digitally photographed using a Nikon coolpix camera, to give 2272x1704 colour images. The colour images were converted to greyscale to improve initial processing time. Figure 1 shows typical greyscale images of canine peripheral blood smear slides. Figure 1(a) shows normal blood and Fig. 1(b) shows blood containing irregularly shaped, damaged red blood cells called acanthocytes. The large cell containing darker material—the nucleus—is a white blood cell, the smaller diffuse cells are platelets and the smaller solid pieces are red-cell fragments.

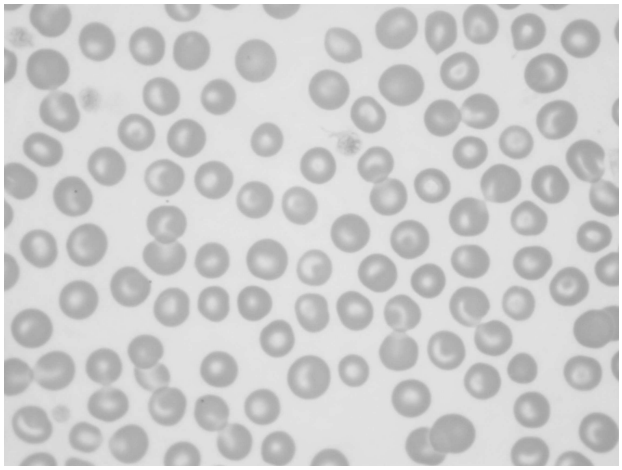
4 A Fully Automated Segmentation and Boundary Identification Algorithm

4.1 Segmentation

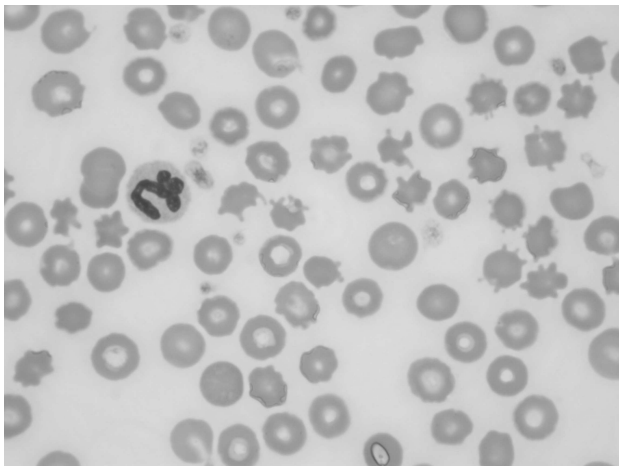
Greyscale histograms of the images in Fig. 1 are shown in Fig. 2. They were calculated using a bin size of 4, which acts to smooth the histogram. As can be seen, there is a marked similarity between the histograms. This similarity holds for all images to which we have access, suggesting that the histograms can be used for automatic selection of a useful threshold (Weszaka, Nagel & Rosenfeld 1974). This is done as follows. A search is made from the right to the left of the histogram, for the first decrease in pixel count. This gives the location of the right peak, which corresponds to the most common greyscale within the background. From there the search continues until the first increase in pixel count, which gives the middle low point. As this low point includes pixels that form part of the border of the cells, the threshold is chosen as the greyscale that falls $\frac{1}{4}$ of the 'distance' from the central minimum towards the right peak. This choice forms a good balance between separation of overlapping cells and 'leakage' of the background into central pale areas close to the border of a cell.

The background of the image is then identified using a 4-adjacency connected components algorithm (Gonzalez & Woods 2002) instead of thresholding. The initial seed point of this algorithm is the top-left most point in the image with a greyscale greater than the calculated threshold. From this seed point all connected pixels with greyscale greater than the threshold value are converted to white. To ensure that the seed point is not interior to a cell, the connected pixels are counted. If this number is much less than the number of pixels between the central minimum of the histogram and the right most peak, then the algorithm iterates using a new seed point, 10 pixels down and to the right of the current seed point.

The algorithm results in a segmented image with the background white and the foreground containing all the cells (red, white and platelets). Incomplete cells that overlap the edge of the image are deleted

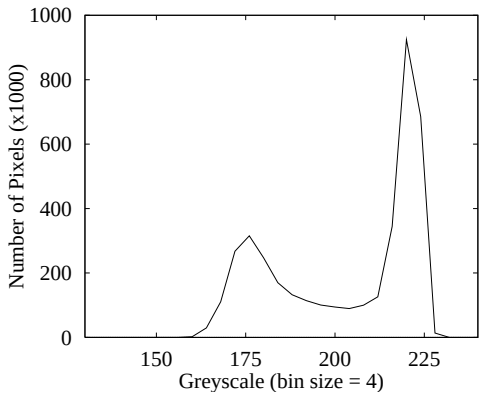


(a) Normal blood.

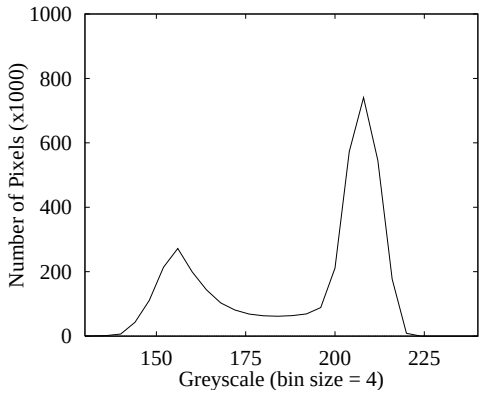


(b) Abnormal blood.

Figure 1: Images taken from slides of peripheral blood smears of canine blood.

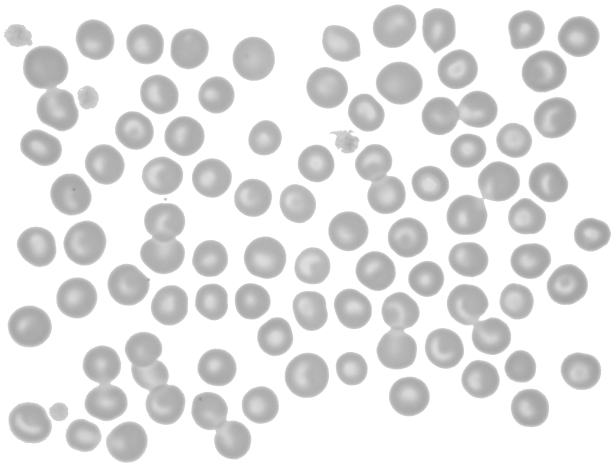


(a) The histogram of the image shown in Fig. 1(a).

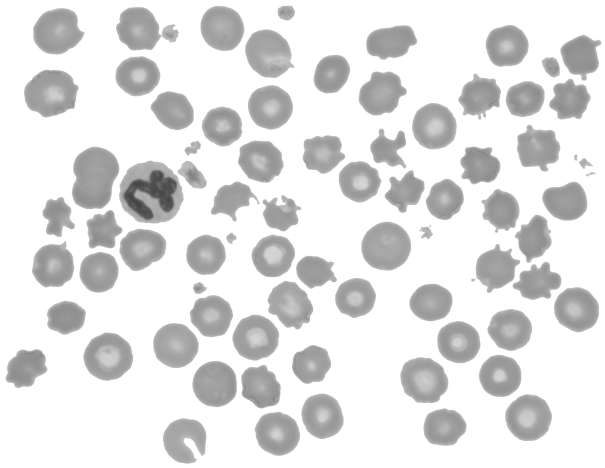


(b) The histogram of the image shown in Fig. 1(b).

Figure 2: Histograms for images taken from peripheral blood smear slides show marked similarities to each other.



(a) Processed from Fig. 1(a).



(b) Processed from Fig. 1(b)

Figure 3: Images in which the objects have been separated from the background, and those that overlap the boundary of the image have been removed.

by doing a 'circuit' of the image removing all pixels that are connected to a non-white boundary point. Figure 3 shows the resulting image for the two original images of Fig. 1.

The objects in the image are then identified as separate entities as follows. A copy is made of the image. A search is then made from the top-left most corner of the copy for the first non-white pixel. This pixel is used as a seed point for a 4-adjacency connected components algorithm that gathers all connected pixels that are not white. Each pixel so gathered is designated either a border pixel—if 8-connected to a white pixel—or an interior pixel, and added to the new object. The choice of 8-connected rather than 4-connected border pixel selection is made due to the improved smoothness of the final borders after the processing described in the next section. All the pixels gathered into the object—both border and internal—are then set to white and the search repeated.

The result is an array of all objects within the original image, where the objects are cells or cell groups. Each object is itself composed of two arrays: points interior to the cell, and points on the border of the object.

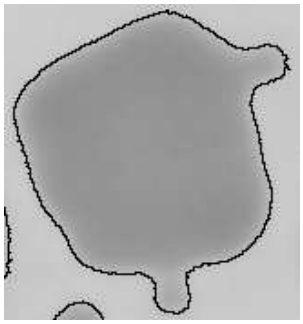


Figure 4: A cell from the top left of the image in Fig. 1(a). The black line shows the initial border formed using the algorithm described in Section 4.1.

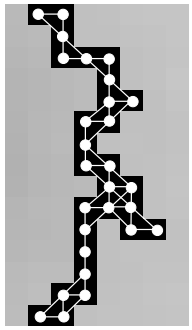


Figure 5: A portion of the border of the cell shown in Fig. 4: it contains 'stubs' that should not be part of the final border. Superimposed on the image is the graph that models the border, with pixels becoming vertices. Arcs are formed from 8-connectedness of the border pixels.

4.2 Border Thinning

The result of the segmentation and initial border selection can be seen in the closeup image in Fig. 4, where the border is shown in black. However the borders are wide and have many small extra stubs and blocks that would interfere with subsequent shape analysis. In particular statistical analysis can be biased by borders that have clusters of pixels. It is therefore useful to refine the borders by pruning stubs and thinning the border to one pixel width.

To refine a border, it is defined as representing a sparse graph, where the vertices are pixel positions. Arcs in the graph are formed between any pair of vertices representing 8-connected pixels. This is illustrated in Fig. 5, which shows a portion of the right-hand side of the border of the cell shown in Fig. 4.

4.2.1 Graph Representation

The traditional way to represent a graph is via an adjacency list or adjacency matrix (Sedgwick 2001). However building such a representation from the array of border points is slow as every point must be compared to every other point to check for connectivity. We therefore use an alternate representation. The border points are sorted—using a row-order index sort—into an array of arrays. In this two dimensional structure, each row stores the indices of border points that fall in the same image row. This allows direct access to all border points in a specific row in the image using the image row value itself—minus the value of the lowest row—as an array index. Each row of indices is also sorted on column order, further improving search time. A schematic diagram of part of the resulting structure is shown in Fig. 6.

```
Use the initial vertex as a one vertex SPT
WHILE less than V-1 vertices added AND
      target not found
   Search the SPT for the shortest path
      between the starting vertex and a
      vertex adjacent to the SPT
   Add that vertex to the SPT
   IF the new vertex is the target
     found = true;
   ENDIF
ENDWHILE
```

Algorithm 1: Dijkstra's shortest path algorithm (Sedgwick 2001), modified to find the shortest path between two specific vertices.

Whilst this representation could be used to build an adjacency list or matrix, this proves to be unnecessary as the search for a connected vertex is already reduced to a search of only the three short arrays containing indices of points above, below and in the same image row as the current vertex. Its use resulted in a 50% reduction in the time taken to refine the border as compared with building an adjacency matrix or list from the original border array.

4.2.2 Graph Algorithm

Dijkstra's shortest path algorithm (Sedgwick 2001) builds a shortest path tree (SPT) from a starting vertex to all other vertices. This is achieved by iteratively searching for the shortest total path from the starting vertex and a vertex adjacent to one already in the SPT, and then adding the new vertex to the SPT. This algorithm can be used to find the shortest path between two specific vertices by prematurely ending the SPT build when the target vertex is found. Algorithm 1 contains pseudo-code for this.

However, searching "the SPT for the shortest path between the starting vertex and a vertex adjacent to the SPT" is time consuming, as it involves iterating through every node in the current tree. However, for our graph this can be simplified. Since the graph only contains previously identified potential border points, at each iteration of the algorithm only an arc of either length 1 or $\sqrt{2}$ can be added. This is because the cost function is based purely on physical distance and does not include image information as for previous users of graph theory (Martelli 1976, Pope et al. 1985, Fleagle et al. 1991). Furthermore, since all distances from the starting vertex are given by $m+n\sqrt{2}$, the tree can be formed using discrete layers sorted on distance from the root. Therefore when building the SPT, only the highest unprocessed layer needs to be processed: it is guaranteed that the next two layers that should be added will be the ones that are 1 unit and $\sqrt{2}$ units longer than that layer. A schematic diagram of the start of the tree-build is shown in Fig. 7 and the simplified algorithm is shown in Algorithm 2. It results in an average time-saving of 73% as compared with using Dijkstra's original algorithm.

4.2.3 Choice of End Points

The final problem to be solved is the choice of the starting and target pixels. Since the purpose is to find the shortest path around the previously selected border pixels, the obvious choice would be two adjacent pixels. However this has several problems. Firstly the SPT would find the path between the two that was just 1 or $\sqrt{2}$ long and secondly any particular pair of pixels chosen might be on a spur that should be removed. We solve these problems by finding a set of

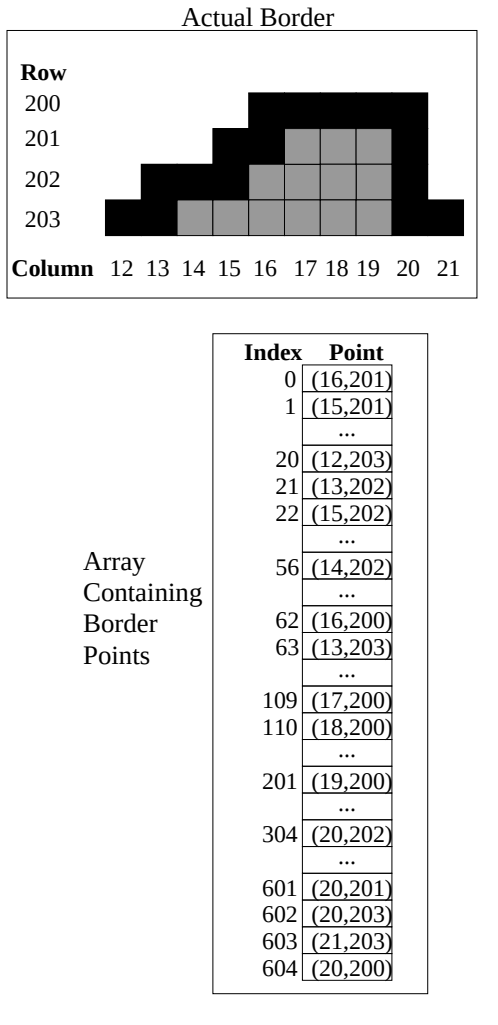


Figure 6: A schematic diagram showing the sorted representation of the original border points. This representation allows the search for two adjacent border points to be reduced to a search of three short arrays.

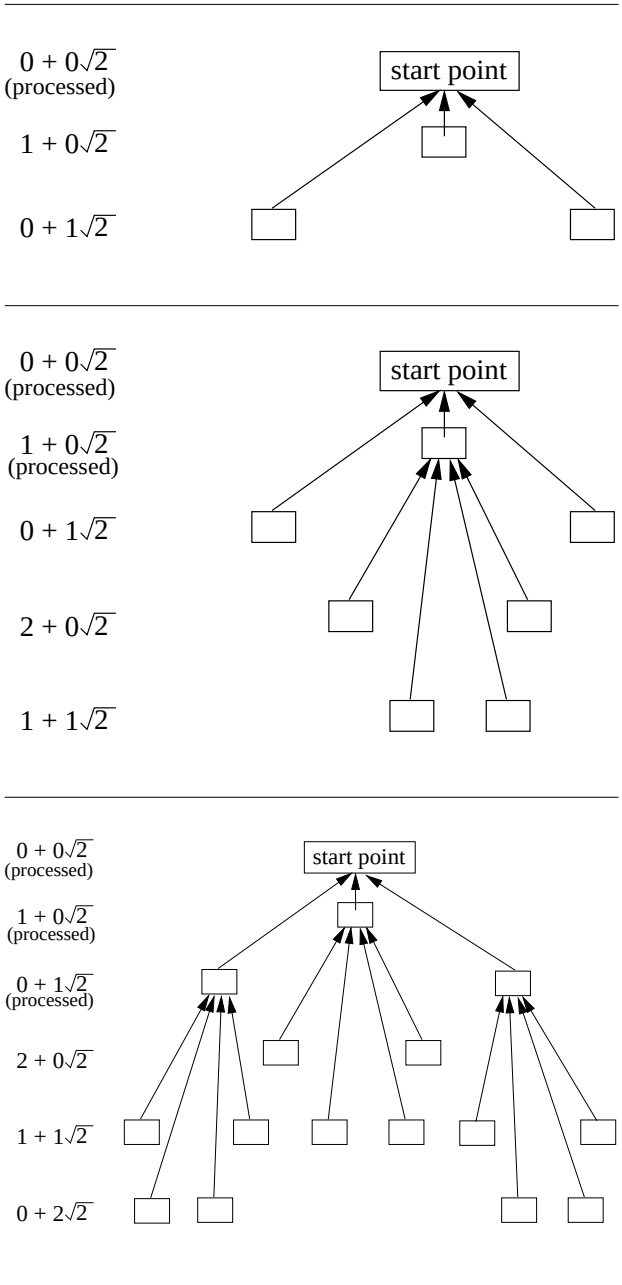


Figure 7: A schematic diagram showing how the layers of the shortest path tree are built using Algorithm 2.

```
Use the initial vertex as a one vertex SPT
WHILE unprocessed layers in the SPT AND
   target not found
  layer = next unprocessed layer in the SPT
  FOR each node in the layer AND
     while target not found
    Add to layer1 all border nodes
       4-adjacent to this node, that are
       not already in the SPT
  ENDFOR
  Add layer1 to correct place in the SPT
  FOR each node in the layer AND
     while target not found
    Add to layer2 all border nodes
       4-diagonal to this node, that are
       not already in the SPT
  ENDFOR
  Add layer2 to correct place in the SPT
  Mark layer as processed
ENDWHILE IF target not found
  return error
ELSE   return noerror
ENDIF
```

Algorithm 2: A modified version of Dijkstra's SPT algorithm. A schematic diagram showing the building of layers in the SPT, can be seen in Fig. 7.

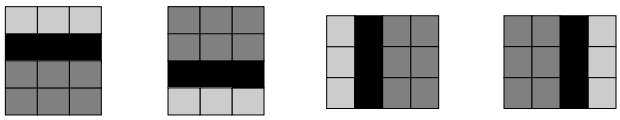


Figure 8: The border point configurations used to find good start and target border points for the SPT. The black pixels represent border pixels, the dark grey pixels are those that are part of the object and the light grey pixels represent the image background.

three points that form a straight line border at a place in the object that is at least two pixels wide. This results in a search for a set of three border points that conform to one of the four configurations shown in Fig. 8. The middle pixel of the three is then marked as already added to the tree and the other two used as the start and target points of Algorithm 2. Once the shortest path between them has been found, the middle point is added onto the end of the path. The result is an ordered one pixel wide border for the object. The points that were in the original border but not in the final border are added to either the interior of the object or the image background, as appropriate. An example of the final border is shown in Fig. 9.

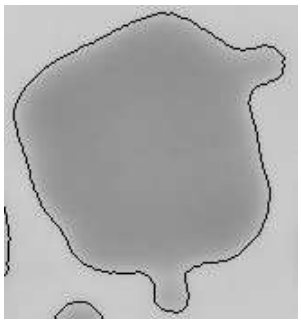
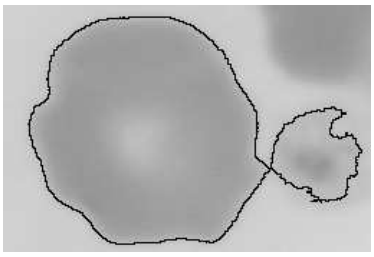
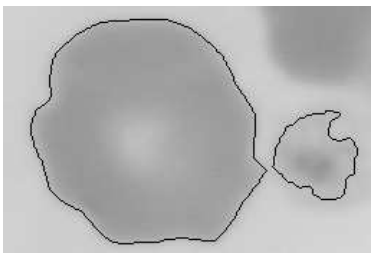


Figure 9: The same cell as that shown in Fig. 4 after the border has been refined.



(a) An object composed of two touching objects: the border forms a figure of 8.



(b) The two objects separated by eroding the original object, and reprocessing the two separate ones.

Figure 10: Two just-touching objects shown before and after the separation algorithm has been performed.

4.2.4 Correcting Figure-of-Eight SPT Failure

Occasionally two objects are just touching which results in a border with a figure-of-eight shape. An example of this can be seen in Fig. 10(a). Clearly the shortest path algorithm will fail in this case as the node at the intersection of the two border loops needs to be included twice, but can only be selected once. In this situation the algorithm returns an error code. Upon receiving this error code the object will be isolated from the rest of the image, eroded by removing the original border and reprocessed to produce two separate objects with smoothed borders. The result of this process can be seen in Fig. 10(b).

5 Results

The end result of the entire process described in Section 4 is an array of objects with both internal and border points identified. Figure 11 shows examples of the results, with the borders shown in black.

The method has been tested on 47 images of both normal (14) and diseased (33) canine blood. The objects in the images include normal red blood cells (discoytes), deformed red blood cells (echinocytes, acanthocytes, codocytes, pre-keratocytes, polychromatophils and rouleaux), red blood cell fragments (shistocytes), platelets, white blood cells and overlapping cells. The images are all 2272x1704 pixels in size. In these images there are 2962 objects that do not touch the boundary of the image. Our segmentation algorithm finds 2961 (99.97%) of them. The one missed is a pale platelet. Of the 2961 objects found, 97.33% are correctly segmented and bordered using our algorithm. The source of the errors can be seen in Table 1. The average processing time per image is 6.045 seconds, or 0.095 seconds per object, using a Pentium 4, 2.4 GHz machine with 1 GB of RAM.

Description	Number of Objects	% of Total
Outer region of platelet not included	41	1.36%
Platelet segmented as more than one object	12	0.41%
Pale outer tips of cell protruberances not included in cell segmentation	11	0.37%
Pale outer tips of cell protruberances segmented as separate objects	5	0.17%
'Blisters' on pre-keratocytes not included in cell segmentation	5	0.17%
Part of blister on pre-keratocytes segmented as separate object	1	0.03%
'Leakage' of the background into the pale inner area of the cell	4	0.16%
Correctly segmented with correct border identification	2882	97.33%
TOTAL	2961	100.00%

Table 1: Results of the segmentation and border identification algorithm when run on 47 images which include normal red blood cells, deformed red blood cells, red blood cell fragments, white blood cells, and platelets.

The algorithms were then run on 51 new images—17 normal, 34 diseased—taken from different slides by a different photographer but with similar equipment. These images are also all 2272x1704 pixels in size. The camera settings were varied to get different contrast, brightness and magnification. In this set of images there are 5417 objects of which our algorithm again misses one (a platelet). The success rate of correct segmentation and border identification of the objects found is 98.98%: a similar rate as with the first set of images. Information about the failures is given in Table 2. For this set of images the average processing time per image is 8.00 seconds, or 0.075 seconds per object. The time taken per image is greater due to the larger numbers of objects in the images, however the processing time per object is less due to the smaller pixel size of the average object.

6 Discussion

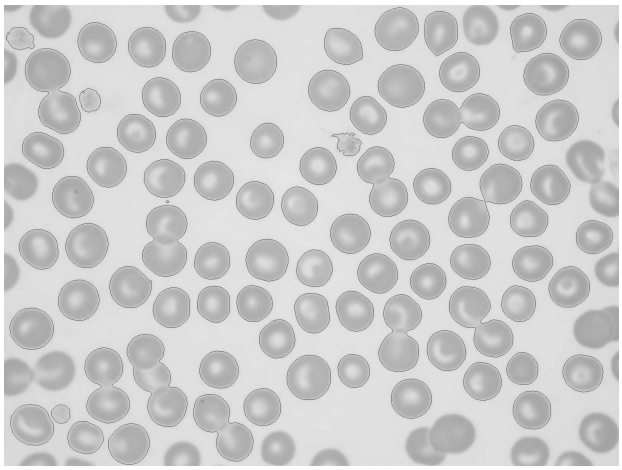
The number of fails for the segmentation and border detection of objects in the image is very small, especially when compared to the very high rates of observer error for human detection (Fairbanks 1971). Furthermore—as described below—most of the errors are not significant or can be dealt with in later processing.

Over two thirds of those objects not correctly segmented are platelets, which have diffuse areas that are not picked up by the segmentation algorithm. As the main work is aimed at differentiating red blood cells which means that platelets will be discarded, this is not a major problem. The automatic identification of platelets will be dealt with in future work.

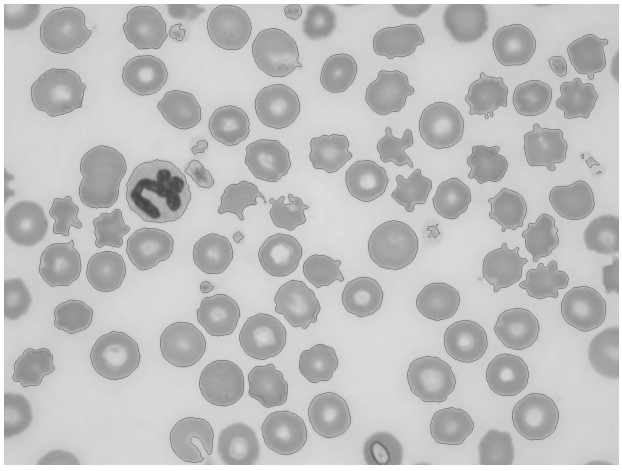
The loss of pale tips of protuberances is also not considered a major problem as in all but one case the cell has multiple other protuberances which would clearly define it as being deformed. The loss of the outer edge of a 'blister' in a pre-keratocyte cell, as shown in Fig. 12, would cause the cell to be later classified as a keratocyte, rather than a pre-keratocyte. Again this does not constitute a major problem, as later algorithms could check all cells classified as keratocytes for pale borders remaining around the blister. Furthermore, both morphologies are indicative of the same diseases (Reagan et al. 1998).

Description	Number of Objects	Percentage of Total
Outer region of platelet not included	39	0.72%
Platelet segmented as more than one object	3	0.06%
Pale outer tips of cell protruberances not included in cell segmentation	1	0.02%
Pale outer tips of cell protruberances segmented as separate objects	0	0.00%
'Blisters' on pre-keratocytes not included in cell segmentation	10	0.18%
Part of blister on pre-keratocytes segmented as separate object	1	0.02%
'Leakage' of the background into the pale inner area of the cell	1	0.02%
Correctly segmented with correct border identification	5361	98.98%
TOTAL	5416	100.00%

Table 2: Results of the segmentation and border identification algorithm when run on 51 new images which again include normal red blood cells, deformed red blood cells, red blood cell fragments, white blood cells, and platelets.



(a) The final borders found in the image of Fig. 1(a).



(b) The final borders found in the image of Fig. 1(b).

Figure 11: Two images from peripheral blood smear slides showing the final borders in black. All pixels inside the borders have been collected as part of the object.

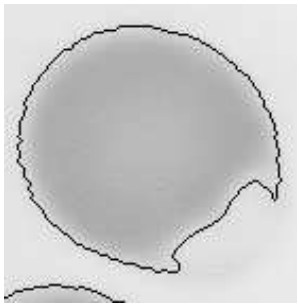


Figure 12: A pre-keratocyte cell where the outer pale edge of the blister has not been included by the segmentation algorithm. The cell therefore looks like a keratocyte instead.

Finally it is worth noting that it is important that the algorithm does not separate overlapping cells. This is because overlapping cells may in fact be joined cells (rouleaux or aggregates) which are caused by disease. It is therefore necessary that cell groups be left intact for later classification.

7 Conclusion

We present a fully automatic method for segmentation and border identification of all objects that do not overlap the boundary in an image taken from a peripheral blood smear slide. Unlike prior algorithms the method is fully automated, fast and accurate. It can separate cells that just touch and has been shown to work with both normal, deformed and joined red blood cells, as well as white blood cells and red cell fragments. It has been tested on a total of 98 images from two different sources with a high success rate.

The algorithm combines automatic threshold selection with connected-components and a novel adaption of Dijkstra's shortest path algorithm and an alternate graph representation to the standard adjacency list or matrix. The result is an average processing time of 7.06 seconds per image, with an average of 84 objects per image.

Further work will involve use of shape factors and interior greyscale analysis to classify all of the red blood cells. This will enable a full red blood cell morphology count.

References

Adjouadi, M. & Fernandez, N. (2001), 'An orientation-independent imaging technique for the classification of blood cells', *Particle & Particle Systems Characterization* **18**(2), 91–98.

Albertini, M., Teodori, L., Piatti, E., Piacentini, M., Accorsi, A. & Rocchi, M. (2003), 'Automated analysis of morphometric parameters for accurate definition of erythrocyte cell shape', *Cytometry Part A* **52A**(1), 12–18.

Bacus, J. (1971), 'The observer error in peripheral blood cell classification', *American Journal of Clinical Pathology* **59**, 223–230.

Bacus, J., Belanger, M., Aggarwal, R. & Trobaugh Jr., F. (1976), 'Image processing for automated erythrocyte classification', *Journal of Histocemistry & Cytochemistry* **24**(1), 195–201.

Bell, A. (1997), Morphological evaluation of erythrocytes, *in* E. A. Stiene-Martin, C. A. Lotspeich-Steininger & J. A. Koepke, eds, 'Clinical Hema-

tology: Principles, Procedures, Correlations', 2nd edn, Lippincott Williams & Wilkins, chapter 8, pp. 87–105.

Bentley, S. & Lewis, S. (1975), 'The use of an image analysing computer for the quantification of red cell morphological characteristics', *British Journal of Haematology* **29**, 81–88.

Bessis, M. (1973), Red cell shape: An illustrated classification and its rationale, *in* M. Bessis, R. Weed & P. Leblond, eds, 'Red Cell Shape: Physiology, pathology, ultrastructure', Springer-Verlag, New York, pp. 1–26.

Bessis, M. (1977), *Blood smears reinterpreted*, Springer International, Berlin. Translated by G. Brecher.

CellaVision DM96 Technical specifications (2006). `http://www.cellavision.com/`.

Connors, D. & Wilson, M. (1986), 'A new approach to the reporting of red cell morphology', *Journal of Medical Technology* **3**(2), 94–785.

Costin, H., Rotariu, C., Zbancioc, M., Costin, M. & Hanganu, E. (2001), 'Fuzzy rule-aided decision support for blood cell recognition', *Journal of Fuzzy Systems & Artificial Intelligence* **7**(1–3), 61–70.

Dasgupta, A. & Lahiri, P. (2000), 'Digital indicators for red cell disorder', *Current Science* **78**(10), 1250–1255.

Di Ruberto, C., Dempster, A., Khan, S. & Jarra, B. (2000), Segmentation of blood images using morphological operators, *in* 'Proceedings of the 15th International Conference on Pattern Recognition (ICPR 2000)', IEEE, Barcelona, Spain, pp. 397–400.

Di Ruberto, C., Dempster, A., Khan, S. & Jarra, B. (2002), 'Analysis of infected blood cell images using morphological operators', *Image And Vision Computing* **20**(2), 133–146.

Fairbanks, V. F. (1971), 'Is the peripheral blood film reliable for the diagnosis of iron deficiency anaemia?', *American Journal of Clinical Pathology* **55**, 447–451.

Fleagle, S., Johnson, M., Wilbricht, C., Skorton, D., Wilson, R., White, C., Marcus, M. & Collins, S. (1989), 'Automated analysis of coronary arterial morphology in cineangiograms: geometric and physiologic validation in humans', *IEEE Transactions on Medical Imaging* **8**(4), 387–400.

Fleagle, S., Thedens, D., Ehrhardt, J., Scholz, T. & Skorton, D. (1991), 'Automated identification of left ventricular borders from spin-echo magnetic resonance images', *Investigative Radiology* **26**(4), 295–303.

Foresto, P., D'Arrigo, M., Carreras, L., Cuezzo, R., Valverde, J. & Rasia, R. (2000), 'Evaluation of red blood cell aggregation in diabetes by computarized image analysis', *Medicina-Buenos Aires* **60**(5), 570–572.

Gering, E. & Atkinson, C. (2004), 'A rapid method for counting nucleated erythrocytes on stained blood smears by digital image analysis', *Journal of Parasitology* **90**(4), 879–881.

Gonzalez, R. & Woods, R. (2002), *Digital Image Processing*, Prentice Hall, New Jersey.

Ingram, M. & Preston Jr., K. (1970), 'Automatic analysis of blood cells', *Scientific American* **223**(5), 72–82.

Kass, M., Witkin, A. & Terzopoulos, D. (1988), 'Snakes: Active contour models', *International Journal of Computer Vision* **4**, 321–331.

Lesesve, J., Salignac, S., Alla, F., Defente, M., Benbih, M., Bordigoni, P. & Lecompte, T. (2004), 'Comparative evaluation of schistocyte counting by an automated method and by microscopic determination', *American Journal of Clinical Pathology* **121**(5), 739–745.

Martelli, A. (1976), 'An application of heuristic search methods to edge and contour detection', *Communications of the ACM* **19**(2), 73–83.

Novis, D., Walsh, M., Wilkinson, D., Louis, M. S. & Ben-Ezra, J. (2006), 'Laboratory productivity and the rate of manual peripheral blood smear review - a College of American Pathologists q-probes study of 95141 complete blood count determinations performed in 263 institutions', *Archives of Pathology & Laboratory Medicine* **130**(5), 596–601.

Ongun, G., Halici, U., Leblebicioglu, K., Atalay, V., Beksac, S. & Beksac, M. (2001), 'Automated contour detection in blood cell images by an efficient snake algorithm', *Nonlinear Analysis-Theory Methods & Applications* **47**(9), 5839–5847.

Payne, N., Bridgen, M. & Edora, F. (1996), 'A redesign of RBC morphology reporting', *Medical Laboratory Observer* **28**(4), 60–65.

Pope, D., Parker, D., Clayton, P. & Gustafson, D. (1985), 'Left ventricular border recognition using a dynamic search algorithm', *Radiology* **155**(2), 513–518.

Preston Jr., K. (1976), Clinical use of automated microscopes for cell analysis, *in* K. Preston Jr. & M. Onoe, eds, 'Digital Processing of Biomedical Images', Lenum Press, New York, pp. 47–58.

Reagan, W., Sanders, T. & DeNicola, D. (1998), *Veterinary hematology: atlas of common domestic species*, Iowa State University Press.

Robinson, R., Benjamin, L., Cosgriff, J., Cox, C., Lapets, O., Rowley, P., Yatco, E. & Wheeless, L. (1994), 'Textural differences between AA and SS blood specimens as detected by image-analysis', *Cytometry* **17**(2), 167–172.

Rowan, R. (1986), Automated examination of the peripheral blood smear, *in* 'Automation and quality assurance in haematology', Blackwell Scientific, Oxford, chapter 5, pp. 129–177.

Sedgwick, R. (2001), *Algorithms in C++ Part 5: Graph Algorithms*, 3rd edn, Addison-Wesley.

Walton, J. (1973), 'Uniform grading of hematologic abnormalities', *American Journal of Medical Technology* **39**(12), 517–523.

Wang, X., He, L. & Wee, W. (2004), 'Deformable contour method: A constrained optimization approach', *International Journal of Computer Vision* **59**(1), 87–108.

Weszaka, J. S., Nagel, R. N. & Rosenfeld, A. (1974), 'A threshold selection technique', *IEEE Transactions on Computing* **23**, 1322–1326.