Swarmbased computing 2015.

Using a hybrid PSO with FCM clustering for retinal image segmentation.

Jelger Kroese & Jaspert de Vries

University Leiden

22-12-2015

Introduction

In today's medical science, imaging is becoming a well-accepted and important tool for both diagnosing diseases as for biomarker research. MRI devices can create a broad range of different images, some easy to interpret and some hard to interpret. Images that are hard to interpret can be MRI images made from the brain. Some of these images are blurry due to noise and in other images the grayscale makes it hard to separate the different brain parts. Also the search for possible tumors in tissue is difficult due to low resolution of the MRI images. Tumors are most of the time small spots on images and are hard to separate from the images. Another field in of medicine research where imaging is a common tool for diagnoses is in the field of eye research. Fundoscopy devices create images of the retina and are often used by ophthalmologist. By looking at these images an ophthalmologist can easily diagnose several eye diseases like *macula edema*. Also by looking at the structure of the blood vessels, diseases as diabetes type 2 can be diagnosed from retina images [1, 2]. So imaging is a well-accepted tool in the field of medicinal sciences. However, image diagnosis is sometimes very hard and not fully automated. Algorithmic image segmentation can be very helpful to automate the investigation of several structures of an image separately. Especially for brain imaging, segmentation of the images can be useful and might improve the interpretation of different brain parts. Segmentation in images of the retina can be used to separate the blood vessels in the image. With the separation of the blood vessels, structures of the blood vessels can be investigated and potential relations with diseases can be obtained from these images. One disadvantage of most of these images is that they are noisy and have inconsistent background lighting.
The aim of this study is to find

In literature, many algorithms are known for image segmentation. The fuzzy c-means algorithm (FCM) is widely used for this. While it is still a successful method for finding clusters within images, the disadvantages of the FCM are its high sensibility to noise and likeness to get stuck in local optima. Proper noise handling in medical images is important, since too much noise reduction in the preprocessing stage cannot be done. This leads to the loss or transformation of finer details in the image. These fine details mostly hold important information about a patient's medical condition. For example, in cancer research you want to find an early stage tumor, because tumors are better to treat in that stage. But the smaller a tumor is, the more likely it is to be indicated as noise. Noise reduction can be feasible when images have different color canals, but in this report only gray scaled images are used. Therefore, instead of focusing on the noise reduction by image processing, in this report the focus is on evaluating the use of new image clustering techniques, which are more likely to find a global optimal clustering.

Recent studies [12, 15, 16] have showed promising results for image segmentation by using Particle Swarm Optimization (PSO) algorithms. Ye et al. [20] developed an PSO that segments images based on their histogram by placing thresholds. Other studies [12, 15, 16] combine a PSO with clustering algorithms in order to improve their performance. These hybrid algorithms are more widely applicable for data clustering than the threshold-placing PSO. The value in using a hybrid PSO is that it combines the global search capabilities of the PSO and the good local search capabilities of the clustering algorithm. The hybridization also makes the output of the PSO also more consistent.

This study focuses on the use of a hybridization of a PSO and an FCM algorithm (FPSO) for the extraction of the blood vessels from the retinal images. This is challenging, since the determination if something is a blood vessel or tissue is hard. The differences in gray intensities that determine if a set of pixels is blood vessel or tissue is very small when the blood vessels will become smaller. This is because smaller blood vessels have very thin blood vessel walls. Therefore the objective is to extract the first – and second order blood vessels from the images. First order blood vessels are the bigger vessels, eventually they will branch in smaller blood vessels which can be named as the second order blood vessels/capillaries. Eventually this smaller blood vessels will branch in the third order blood/vessels. Third order blood vessels are more difficult to extract and are in this situation beyond

the scope, since it requires more advanced imaging techniques. The following sections will first show how an FCM and PSO work separately, then show how we combined the two into a FPSO and then show its performance on retina images.

Fuzzy c-means clustering

Data clustering is an important method for finding structures within datasets and is widely used within the field of Bioinformatics. As a digital image is essentially a matrix of data points it can be used to segment an image into different elements. It is a method that aims to divide data into clusters in such a way that data within the same clusters is as similar as possible and data that belong to different clusters is as different as possible. Shortly said, it segments dataset X into c clusters and returns a new X based on these clusters [3]. Depending on the purpose of the algorithm, different criteria to base the clustering on can be used.

Two main types of clustering can be distinguished: *hard* clustering and *fuzzy* clustering (also known as *soft* clustering). Hard clustering divides the dataset into strict clusters, where every element belongs to exactly one cluster. These clusters are called *hard* c-partitions of X. This strict clustering, however, does not give any information about how well the data points fit within the clusters and sees points that are in different clusters as completely unrelated to each other. Since in many cases this does not reflect how the data is divided in reality, *fuzzy* clustering methods have been developed. Here, elements can belong to multiple clusters. Each element has a value for each cluster that indicates their relatedness to that particular cluster. Based on their relatedness the data elements are grouped into different clusters.

Fuzzy c-means clustering is one of the most widely used fuzzy clustering methods. It was developed by Bezdek et al. (1984)[3] and was based on the original idea of Zadeh (1965)[4] to give all data points a value μ_{ik} between 0 and 1 for each cluster, which indicates their relatedness to that particular cluster. With the sum of all μ_{ik} for one data point = 1. The algorithm clusters data based on their relationship to a predefined number of cluster centers c. The cluster centers are randomly generated at the start of each run and are adjusted through iterations, until a stopping criterion is met. This stopping criterion usually is a threshold for the number of iterations or the fitness level. During every iteration a membership matrix U is updated by equation (1). This matrix contains for every data point, in our case an image pixel value, the membership degree to all of the cluster centers. The calculation of this membership degree is based on the Euclidean distance between the data point and one cluster compared to the Euclidean distance between the data point and all the cluster centers. The membership matrix U is then used to compute new cluster centers V_i by equation (2). At the end of every iteration the objective function value $J(U,V)$ of the current solution is checked by equation (3). This value is a measure for intra-cluster variance. The goal of the FCM algorithm is to minimize $J(U,V)$.

Table 1: list of parameters used in an FCM

$X = \{x_1, \dots, x_n\}$		Dataset
c		Number of cluster centers
n		Number of data elements
U	$\mu_{ik} \in [0,1]$, $i = 1, \dots, c, \quad k = 1, \dots, n$	Membership matrix
$V = \{v_1, \dots, v_c\}$		Cluster centers
m	$m \in R, \quad m \geq 1$	Fuzzifier

$$\mu_{ik}^{(t+1)} = \left[\sum_{j=1}^{c} \left(\frac{\left\| x_k - V_i^{(t)} \right\|^2}{\left\| x_k - V_j^{(t)} \right\|^2} \right)^{\frac{1}{m-1}} \right]^{-1} \quad (1)$$

$$V_i = \frac{\sum_{k=1}^{n} [\mu_{ij}]^m . x_k}{\sum_{k=1}^{n} [\mu_{ij}]^m} \quad (2)$$

$$J(U,V) = \sum_{k=1}^{n} \sum_{i=1}^{c} [\mu_{ik}]^m \left\| x_k - V_i \right\|^2 \quad (3)$$

Here m is the 'fuzzifier', which scales the influence of the relatedness. For m = 1 the data will be divided into strict clusters (so no fuzzy clustering). A higher m causes a higher fuzziness in the clustering. $\left\| x_k - V_i \right\|^2$ Represents the Euclidean distance between data point x_k and cluster center V_i .

Pseudocode:

1. T:=0;
2. Initialize fuzzy pseudo-partition randomly;
3. Calculate c cluster centers Vi, using (2);
4. While not terminate do
5. Calculate $\mu_i^{(t+1)}$ using (1);
6. Update cluster centers;
7. Evaluate p using (3);
8. Check if stopping criterion is met;
9. Od

Particle Swarm Optimization (PSO).

The general PSO algorithm was developed by Kennedy and Eberhart [5], Kennedy was a social psychologist and Eberhart an electrical engineer. The original ideas were aimed to produce computational intelligence by exploiting simple analogies of social interaction, rather than only pure individual cognitive abilities. There first work was inspired by the simulation of bird flocks, which was the work of Heppner and Gernanders[6]. This first simulation eventually lead to a new optimization method named: Particle Swarm Optimization (PSO). In a simple entities PSO, all particles are placed in a search space of a problem or function, these particles are evaluated by the objective function at their current location. The determination of the movement through the search space of each particle is done by combining an aspect of the particle its current velocity, its own memory, and information it receives from other particles in the swarm. Different types of information exchange between the

particles can be implemented. These are algorithms with different topologies. In this report we do not focus too much on the different kinds of topologies, because there is not a real rule thumb for it. But topology can be divided in two different kinds named the static topologies and dynamic topologies. Whereby the static topologies focus more on the individual particles and trying to get fixed parameters, while the dynamic topologies focus more on the swarms and their neighbors to create a more dynamic selection criteria with no fixed parameters[7, 12]. In this study the topology is used where particles adjust their velocity through their own best position, local best position (pi) and the best position in the whole swarm, global best position (pg). All particles are moved through the search space with every iteration. This leads to new best locations and happens in all particles in the swarm[7]. Eventually the swarm will move to an optimum of the fitness function. An individual in a PSO has four D-dimensional vectors, the vectors are the current position x_i, the local best position pi, the global best position pg and the velocity v_i. The current position x_i can be seen as point in space with a set of coordinates, which represents a problem solution. Every iteration, the x_i can move, after which its new position is evaluated. The current position can be better or worse than its previous position. If the position is an improvement, it will be stored in the p_i vector. If the new position is an improvement of the all positions that were visited by all particles until then it is also stored in pg *[7].* The velocity vector v_i determines the movement directions and speed of x_i. During the iteration the algorithm adjust v_i to let x_i move to the optimum. The velocity vector v_i can also be seen as the step size of the algorithm. x_i, p_i and v_i only describe the problem solving of one individual, but in a PSO problem solving is a population-wide phenomenon. Individuals have their own behavior and interactions with other individuals in the population. This organization in the population has a communication structure that can be seen as a social network[7].

In the following part, the different parameters and components of the PSO will be briefly discussed. Parameters Ø1 and Ø2 are called acceleration coefficients. These determine the magnitude of the random forces that determine the influence of the local best position and global best position on the velocity of a particle. Ø1 and Ø2 have major influences on the behavior of a PSO. The second components that can be evaluated from the original algorithm are and. These two components are the attractive forces which are produced by springs of random stiffness. The motion that they cause can be interpreted as the integration of Newton's second law. The mean stiffness of the springs of pulling a single particle can be represented by Ø1 and Ø2 divided by 2. Changing Ø1 and Ø2 can have major effects and eventually it can make the whole algorithm unstable. To control the velocities in the PSO it is useful to keep them within a range. This range is mostly described as [-Vmax, +Vmax], the value of Vmax is crucial for the balance between exploration and exploitation. The choice of specific Vmax is not possible and there are no rule of thumb to predict a Vmax. Vmax is in most of the cases a problem-specific and by creating a general term for it can create an algorithm that emphasis exploration instead of exploitation[7]. To create a better control of the search and to reduce the dependency of Vmax, a modification can be made for the equation to create an 'inertia weight" with the symbol ω. This was proposed in the study of Shi and Eberhart in 1998 [8] (equation 4).

$$\begin{cases} \vec{v}_i \leftarrow \omega\vec{v}_i + \vec{U}(0,\phi_1) \otimes (\vec{p}_i - \vec{x}_i) + \vec{U}(0,\phi_2) \otimes (\vec{p}_g - \vec{x}_i), \\ \vec{x}_i \leftarrow \vec{x}_i + \vec{v}_i, \end{cases} \quad (4)$$

Here $\vec{U}\ (0, \emptyset i)$ represents a vector of random numbers uniformly distributed in $[0, \emptyset i]$, which is randomly generated at each iteration and for each particle. $\otimes$ is component-wise multiplication. $\vec{U}\ (0, \emptyset 1) \otimes (\overrightarrow{pi} - \overrightarrow{xi}) + \vec{U}\ (0, \emptyset 1) \otimes (\overrightarrow{pg} - \overrightarrow{xi})$ are the external force that influences a particle. This can also be notated as $\overrightarrow{fi}$, by using $\overrightarrow{fi}$ the function for the change in particle velocity can be

written as $\Delta \overrightarrow{vi} = \overrightarrow{fi} - (1 - \omega) \overrightarrow{vi}$. With this transformation of the equation the constant 1 –ω is created, which act as a friction coefficient. The inertia weight constant is better able to have some high values like 0.9. Higher values suggest that a system is moving with low resistance (can also be pronounced as low viscosity), what makes it able to do more extensive exploration. Eventually the value of ω will be lower and the system will have more resistance, it will then go from an explorative state to a more exploitative state. The combination of the well-chosen ω and acceleration coefficients Ø1 and Ø2 will make the PSO more stable. PSO also needs some restraining velocities to decrease the unacceptable levels that were created in a few iterations (Kennedy 1998 [7, 9]). Therefore Kennedy proposed a strategy for the placement of 'constriction coefficients'. These coefficients can control the convergence of the particle and is a method to prevent the explosion, ensuring convergence and elimination of the arbitrary Vmax parameter. Still for the analysis there is some guessing for the values of the acceleration coefficients. Kennedy and Clerc[7, 10] proposed a more simplified method for incorporation the constriction coefficient (equation 5). This is the equation that is used to update particle positions and velocities in this study.

$$\begin{cases} \vec{v}_i \leftarrow \chi(\vec{v}_i + \vec{U}(0, \phi_1) \otimes (\vec{p}_i - \vec{x}_i) + \vec{U}(0, \phi_2) \otimes (\vec{p}_g - \vec{x}_i)), \\ \vec{x}_i \leftarrow \vec{x}_i + \vec{v}_i, \end{cases} \quad (5)$$

where $\phi = \phi_1 + \phi_2 > 4$ and

Using the constriction coefficients the particles can converge without Vmax, only in several experiments, Ebenhart and Shi[11] concluded that it is more appropriate to use a prudent rule of thumb to limit Vmax to Xmax. This is the dynamic range of each variable on each dimension, what eventually creates a PSO with no problem-specific parameters.

Here the pseudo code of a PSO is shown.

1: Initialize population array of particles with random positions and velocities on D dimensions in the search space.
2: **loop**
3: For each particle, evaluate the desired optimization fitness function in D variables.
4: Compare particle's fitness evaluation with its *pbest.* If current value is better than $pbest_i$, then set $pbest_i$ equal to the current value, and $\overrightarrow{pi}$ equal to the current location in $\overrightarrow{xi}$ in D-dimensional space.
5: Identify the particle in the neighborhood with the best success so far, and assign it index to the variable g.
6: Change the velocity and position of the particle using (5)
7: If a criterion is met (usually a sufficiently good fitness or a maximum number of iterations), exit loop
8: **end loop**

FCM and PSO hybrid

Mehdizadeh et al. (2008)[15-17] in their study showed that combining an FCM and PSO algorithm in to a hybrid FPSO can both improve the solution quality and reduce CPU time compared to the standard FCM algorithm. This makes it especially useful for clustering relatively large datasets with image data.

In the FPSO a swarm of particles is created, where each particle represents a solution for the clustering of the dataset. Each particle contains c cluster centers (V) and all particles move through a c dimensional search space by adjusting their V values. During every iteration equation (5) is used to calculate the fitness of all particles. After that, the velocity and position of the particles is updated like in the PSO algorithm.

The standard FCM is quicker at finding a reasonable solution than a PSO when starting out from a random initial solution. It is however likely to get stuck in local optima. The PSO is better at global search. Therefore it is quicker and more accurate at optimizing the solution when the initial population has a reasonably good starting point already (Yang et al., 2011)[16]. In order to combine these two strengths the hybrid algorithm of an FCM and PSO works in two steps:

1. The FCM algorithm runs once until it meets certain stopping criteria.

2. The FPSO uses the solution of the FCM algorithm as one particle in the swarm. The other particles are randomly initialized. Then the FPSO iteratively runs in order to improve the original solution.

The FPSO algorithm works as follows:

1. t := 0; Select the initial parameter values for c, $\emptyset 1$, $\emptyset 2$, X, w, m and a stopping criterion.
2. Run normal FCM once, to provide FPSO with initial particle positions;
3. Calculate $\mu_{ik}^{(t)}$ for all particles by equation (1) and update $p^{(t+1)}$.
4. Calculate the fitness of all particles by (3).
5. Update global best positions and local best positions.
6. Update $Vel_l^{(t)}$ and $V_l^{(t)}$ for all particles by (5)
7. Go to step 3. Compare $p^{(t)}$ and $p^{(t+1)}$. If $\left|p^{(t+1)} - p^{(t)}\right| \leq \varepsilon$, then stop. Otherwise go to 4.

Table 2: parameters for FPSO

n		Number of data elements
c		Number of cluster centers
n_part		Number of particles
$P_l = \{v_1, \ldots, v_c\}$		Particle l, consisting of all cluster centers.
$Vel_l^{(t)}$		Velocity of particle l at time t
$X = \{x_1, \ldots, x_n\}$		Dataset
$\emptyset 1$		Acceleration constant 1
$\emptyset 2$		Acceleration constant 2
w		Inertia weight
β		Constriction factor for velocity
U	$\mu_{ik} \in [0,1]$,	Membership matrix

	$i = 1, \dots, c, \quad k = 1, \dots, n$	
$V = \{v_1, \dots, v_c\}$		Cluster centers
m	$m \in R, \quad m \geq 1$	Fuzzifier

Methods

The main objective for this study is to test the possibilities and limitations of an FPSO algorithm to recognize blood vessels in retina pictures. In our experiments, the image segmentation was based on the intensity levels of the image. Figure 1 shows how we retrieved our images to test the FPSO on. Two pictures of the retina were shot at different wavelengths at the same time. A wavelength of 569 nm shows the red bloodcells (figure 1b), a wavelength of 602 nm shows all other elements, except for the red blood cells (figure 1a). By calculating the absolute difference between these two image a clearer image of the bloodflow through the retina remains (figure 1c). This image is used for the segmentation in all experiments. In the experiments, two implementations of the FPSO are used. One, where the whole image is segmented in one piece and another implementation where the image is divided into 32 blocks, which are processed separately and concatinated afterwards. The following parameter settings are used for the FPSO:

Table 3: parameter settings for experiment

n_part	10
c	3
β	1
$\emptyset 1$	$\sqrt{2}$
$\emptyset 2$	$\sqrt{2}$
w	0.95
m	2
ε	0.0001
Max. number of iterations	40

The clustering results are then compared to the threshold-placing PSO of Yen et al. *[20].* All computations are executed on a laptop with an Intel it HQ4700 processor and 8 GB of RAM.

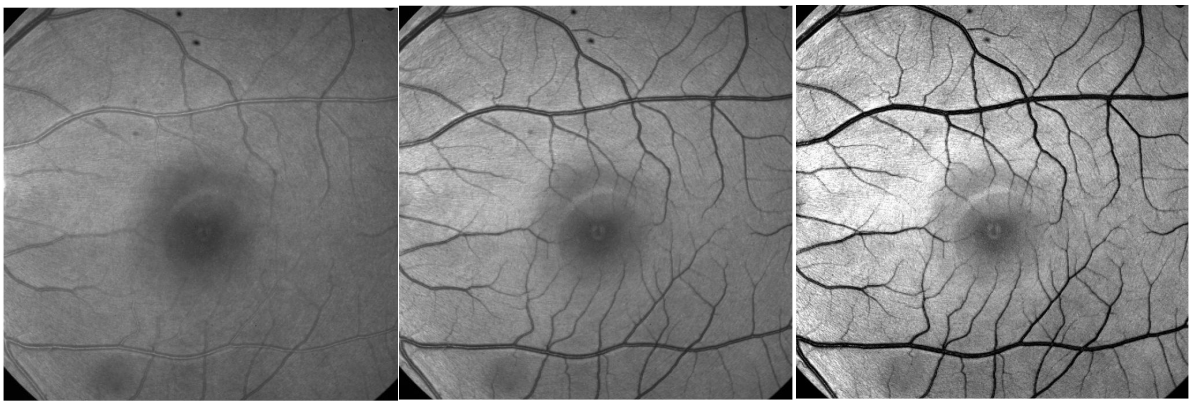

Figure 1: a. retina images captured at 602 nm (left) and b. 569 nm (middle). C. Absolute difference between a and b (right).

Results

Figure 2 compares the original image (a) with the FPSO segmented (b), the block processed FPSO (c) and the threshold-placing PSO (d). It shows the FPSO performs well on detecting the first order blood vessels. However, because of the noise and differences in background lighting intensities it fails to segment many of the smaller second order blood vessels. The Block processed FPSO (c) manages to keep the background noise lower and makes some smaller blood vessels better visible. The fovea in the center of the Retina however, appears as one bright spot, showing less detail. The threshold-placing PSO (d) is also block processed and shows very similar results to th block processed FPSO. Average CPU times were 22.5 minutes for the FPSO, 23.4 minutes for the block processed FPSO and 2.2 minutes for the threshold-placing PSO.

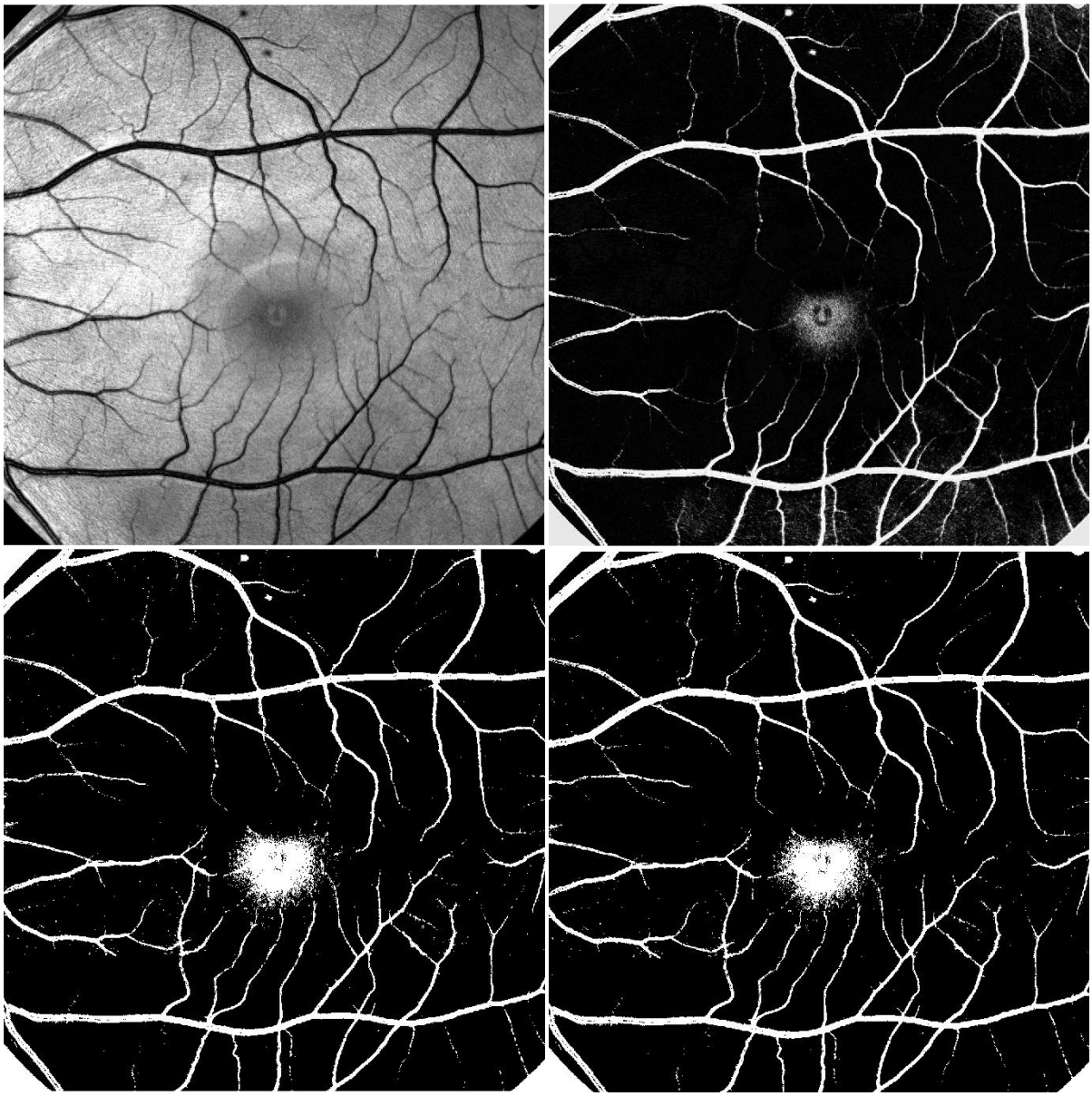

Figure 2: a. (top left) source image. b. (top right) blood vessel segmentation by FPSO. c. (bottom left) blood vessel segmentation by block porcessed FPSO. d. (bottom right) blood vessel segmentation by threshold-placing PSO.

Discussion

Results show that all algorithms manage to segment first order blood vessels quite good. Second order blood vessels are in all algorithms present but are not very detailed. The block processed segmentations show a more detailed representation of the second order blood vessels but still some are missing or some have an intact shape. In the middle of all the segmented images, there is a bright whit spot in the middle. This bright spot is the foveal zone also named the yellow spot. In this area of the eye the cones and rods are present to translate the incoming light in to color spectrum that will be visualized in the cerebellum. This area is bright because in the foveal zone is a very blood perfused area in the eye. This is needed to transport the necessary nutrition to the densely packed cons and rods. The blood for the foveal zone is transported by the blood vessels of the retina, but is transported by the blood vessels of the cornea. The cornea is a small tissue layer that can be found under the retina. In the image that was obtained after the FPSO, this area is less bright and show some more detail. The block processed FPSO performs slightly better than the regular FPSO, since it offers more detail and less noise. All in all this shows that the FPSO is capable of segmenting retinal images to some extent, however the high CPU time it takes to run makes it a very inefficient approach. Since basing the clustering of the FPSO on the histogram values of the image is essentially the same as placing thresholds, the threshold-placing PSO is a better choice since it produces similar results but is much more efficient.

However, for using these techniques in real life, the segmentation must be much more accurate and reliable. This is not possible when the clustering is only based on histogram levels. One of the main difficulties to extract the blood vessels from the image is that the background intensity of the image is not the same everywhere. In the middle around the fovea, the background is darker gray while around the optical nerve the background is bright. These differences are caused by the reflection of the light, when the images were taken. These lighting conditions also differ through different images. In order to make the segmentation more accurate and reliable other features should be extracted from the image to base the clustering on. Since the threshold-placing PSO strictly focuses on placing thresholds in an image histogram, it cannot be used to deal with other features. Our FPSO, however, could be used to cluster the data of multiple features of an image.

An approach for this would be to segment the image into different blocks and extract a number of features per segment. The image in this report was quite big and had the size of 1024 pixels by 1024 pixels, so creating 4 segments will not do the trick. More will be needed, but optimal amount of segments will be hard to predict. This can be validated by doing more research on this topic.

You could then create a matrix of all segments and their corresponding features and find clusters within that. Relevant features to base the clustering on with retina images are blobs and ridges. Several feature extraction methods are available to detect ridge- and blob-like structures from an image. Examples of these are laplacian kernel and principle curvature-based region detector (PCBR). Were Laplacian kernel is an algorithm that is more specialized in finding blob looking structures, the PCBR function is able to find more strait ridges in the image. A laplacian matrix in combination with hierarchical kernel clusterfication can be used to find specific pattern in a search space. A laplacian matrix is a representation of a matrix. This matrix can be used to find new properties of a graph or image. The laplacian matrix will be used together with Kirchoff's theorem to calculate the number of spanning trees. An advantage of laplacians for interactive segmentations is that it uses random walks[18, 19]. During interactive segmentations, the researcher will draw the area that must segmented. This will become difficult if this area is present in the whole image. A problem for image segmentation is to find the smooth-function for a seed of pixels. The laplacians method can provide a solution for this problem, the values of the smooth-function in a seed of pixels

must be comparable to the associated labels. This will allow to vary only on low-density regions in the input space[18, 19].

The PCBR is a structure-based detector, which is designed to handle within-class variance [21]. This makes it robust against intensity differences throughout an image, which is necessary in the case of retina image segmentation. It uses curvilinear structures to match the image data, which matches the shape of the blood vessels [21].

Apart for making the FPSO more accurate by adding features to base the clustering on, the time it takes to run should be decreased. Test for optimizing the algorithm should be done in order to achieve this.

Future perspectives

In the field of image clustering much there are many possibilities to improve images and create higher quality images. In this report we only suggested a possible solution to improve the clustering of a gray scaled retina image. But not only image analyses of retina images can be improved, there are type of medicinal imaging. Microscope images of dividing cultured cells can be optimized using hybridization of the PSO with the fuzzy c-mean clustering image. In these in vitro studies cells are cultured in special culturing liquids and can exposed to several molecules. These molecules can by cytokines to trigger inflammation cells but the molecules can also be the potential drug to cure a disease. For fluorescent microscopy special fluorescent proteins are created that will light up when they are exposed with special wavelength. Special fluorescent protein in some studies will can only be active if special molecule is bound to the protein. This can be used to see if specific receptors are activated after binding of molecule/ligand. It is not always easy to the start of these receptor activation, because at the start only a few receptors are activated. This is also the issue when receptors loses their activated state. By using a hybridization PSO fuzzy c-mean clustering algorithm it can be possible to improve the segmentation of the activated receptors, released cytokines or the distribution of the potential drug. In the microscopy field there are many possible applications were a hybridization of the PSO and fuzzy c-means clustering can be used as an image improving tool.

Conclusion

Hybridization of FCM and PSO is a promising tool for the segmentation of retinal images, but the hybridization is CPU time intensive. Secondly the hybridization of the FCM and PSO algorithm is able extract the first order blood vessels an in less detail the second order blood vessels. PSO in combination with thresholding and block processing decreases the CPU time drastically. For accurate and reliable retinal image segmentation the clustering should be based on other features than grey scale values. More research is needed in order to make an FPSO that works well with these features and is optimized for processing time.

References

1. Podoleanu, A.G. and R.B. Rosen, *Combinations of techniques in imaging the retina with high resolution.* Prog Retin Eye Res, 2008. **27**(4): p. 464-99.
2. Klefter, O.N., A.O. Lauritsen, and M. Larsen, *Retinal hemodynamic oxygen reactivity assessed by perfusion velocity, blood oximetry and vessel diameter measurements.* Acta Ophthalmol, 2014.
3. Bezdek, J.C., R. Ehrlich, and W. Full, *FCM: The fuzzy c-means clustering algorithm.* Computers & Geosciences, 1984. **10**(2): p. 191-203.
4. Zadeh, L.A., *Fuzzy sets.* Information and control, 1965. **8**(3): p. 338-353.
5. Kennedy, J. and R. Eberhart. *Particle swarm optimization*. in *Neural Networks, 1995. Proceedings., IEEE International Conference on*. 1995.
6. Heppner, F. and U. Grenander, *A stochastic nonlinear model for coordinated bird flocks.* AMERICAN ASSOCIATION FOR THE ADVANCEMENT OF SCIENCE, WASHINGTON, DC(USA). 1990., 1990.
7. Poli, R., J. Kennedy, and T. Blackwell, *Particle swarm optimization.* Swarm Intelligence, 2007. **1**(1): p. 33-57.
8. Yuhui, S. and R. Eberhart. *A modified particle swarm optimizer*. in *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*. 1998.
9. Kennedy, J. and W.M. Spears. *Matching algorithms to problems: an experimental test of the particle swarm and some genetic algorithms on the multimodal problem generator*. in *Proceedings of the IEEE international conference on evolutionary computation*. 1998. Citeseer.
10. Clerc, M. and J. Kennedy, *The particle swarm - explosion, stability, and convergence in a multidimensional complex space.* Evolutionary Computation, IEEE Transactions on, 2002. **6**(1): p. 58-73.
11. Eberhart, R.C. and Y. Shi. *Comparing inertia weights and constriction factors in particle swarm optimization*. in *Evolutionary Computation, 2000. Proceedings of the 2000 Congress on*. 2000.
12. Saka, E. and O. Nasraoui, *Improvements in Flock-Based Collaborative Clustering Algorithms*, in *Computational Intelligence*, C. Mumford and L. Jain, Editors. 2009, Springer Berlin Heidelberg. p. 639-672.
13. Parsopoulos, K.E. and M.N. Vrahatis, *Particle swarm optimizaer in noisy and continuously changing evironments.* LASTED/ACTA, 2001. **Artificial intelligence and soft computing**: p. 289-294.
14. Pugh, J., Y. Zhang, and A. Martinoli. *Particle swarm optimization for unsupervised robotic learning*. in *Swarm Intelligence Symposium*. 2005.
15. Mehdizadeh, E., S. Sadinezhad, and T. Tavakkolimoghaddam, *Optimization of fuzzy clustering criteria by a hybrid PSO and Fuzzy C-means clustering algorithm.* Iranian Joernal of Fuzzy Systems, 2008. **5**: p. 1-14.
16. Yang, Y., G. Chen, and Y. Guo. *SVM Combined with FCM and PSO for Fuzzy Clustering*. in *Computational Intelligence and Security (CIS), 2011 Seventh International Conference on*. 2011. IEEE.
17. Alam, S., et al., *Research on particle swarm optimization based clustering: A systematic review of literature and techniques.* Swarm and Evolutionary Computation, 2014. **17**: p. 1-13.
18. Milyaev, S. and O. Barinova, *Learning Graph Laplacian for Image Segmentation.* GraphiCon., 2012.
19. Kang, S.H., B. Shafei, and G. Steid, *Supervised and transductive multi-class segmentation using p-laplacians and RKHS methods.* Elsevier, 2014. **25**(5): p. 1136-1148.

20. Ye, Z., Chen, H., Liu, W., & Zhang, J. (2008, October). Automatic threshold selection based on particle swarm optimization algorithm. In *Intelligent Computation Technology and Automation (ICICTA), 2008 International Conference on* (Vol. 1, pp. 36-39). IEEE.

21. Deng, H., Zhang, W., Mortensen, E., Dietterich, T., & Shapiro, L. (2007, June). Principal curvature-based region detector for object recognition. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on* (pp. 1-8). IEEE.