AN INCREMENTAL APPROACH TO MODEL BASED CLUSTERING AND SEGMENTATION

Chris Brewster, Paul Farmer, James Manners, and Malka N. Halgamuge*

Dept. of Mechanical and Manufacturing Engineering, University of Melbourne, Victoria, 3010, Australia,
*RMIT University, School of Electrical and Computer Systems Engineering, Melbourne, Australia.

Ccb,jmann,pbf@ecr.mu.oz.au, S2005836@student.rmit.edu.au

ABSTRACT

Overlapping a series of adaptive simple mathematical models can be used for image segmentation or data clustering. This paper presents model based evolutionary optimisation segmentation algorithms that incrementally include additional features to the model. Several artificially created images and real images are used to demonstrate the ability of the proposed algorithms.

1. INTRODUCTION

Evolutionary computing techniques have been used for many years by researchers. They are commonly used to effectively search a designated space to find the optimal solution to complex problems.

This paper presents a novel iterative segmentation algorithm that utilises an evolutionary optimisation algorithm (EOA) to determine a satisfactory segmentation. Both Genetic Algorithms (GAs) and Particle Swarm Optimisation (PSO) were used to analyse the affect of each technique on performance.

2. BACKGROUND

2.1. Model based clustering and segmentation

As there are many advantages in obtaining an automated segmentation process, extensive research has been undertaken and many different techniques exist. The most intuitive method is the use of a threshold to class pixels based on their intensity value [1],[2]. More common techniques include the use of edge information contained in the original image or completing boundary or contour tracing [3]. Contour tracing generates an initial contour that is then deformed so that it accurately approximates the real boundary of a region. A similar technique is seed growing. Seed growing generally requires an operator to select appropriate seeds and thresholds. The pixels surrounding the seed are then analysed and included within the region if they fall within the threshold. These newly included pixels are subsequently examined. This technique has been combined with the use of fuzzy logic [4],[5] to handle the uncertainties that occur when processing Magnetic Resonance Imaging (MRI) images.

The technique presented here uses a generic model that can be customised to represent a cluster or a segment of an image. It is expected that a variable number of models can be used to represent a cluster or segment mathematically. This process can also be considered as a way of data compression, as complex clusters or segments can be summarised with a handful of models.

2.2. Evolutionary optimisation algorithms

GAs and PSO are two global solution search methods that are flexible due to their ability to solve many different problems. All that is required is to encode the model used in a suitable manner and evaluate it by the appropriate means.

2.2.1. Genetic algorithms (GA)

Genetic Algorithms involve the generation of an initial population of random guesses at a solution (Chromosomes).

Similar to the partitioning problem [6], the application of GAs presented in this paper requires the ability to deal with variable length chromosomes.

In addition to the widely used genetic operators [7] mutation and crossover, authors have introduced “iterative growth”. Iterative growth introduces an additional model to the existing partially improved chromosome by extending its length to accommodate the parameters of the new model.

2.2.2. Particle Swarm Optimisation (PSO)

This algorithm was developed by researchers who based it upon the swarm behaviour of natural creatures such as fish
or birds. Each “particle” in the search space knows the location of the particle with the best fitness in the population and also its own personal best. The direction this particle moves in the search space is then determined as a function of these two locations. With these processes, the population evolves towards better regions of the search space. Iterative growth is also utilised by PSO to provide additional models to the existing partially improved model.

3. THE INCREMENTAL ALGORITHM

3.1. The Model Ellipse

In this paper, a generic ellipse (oval) with 5 variable parameters is considered as the model for segmentation. The algorithm incrementally introduces ellipses until the clustering or segmentation is satisfactory.

The model parameters were arranged in the following format: \([Oval \, 1, \, Oval \, 2, \, \ldots, \, Oval \, n]\)

Each oval is described with 5 parameters: \([w, \, h, \, x, \, y, \, \theta]\), where \(w\) is oval width, \(h\) is oval height, \(x\) and \(y\) are co-ordinates of oval centre, and \(\theta\) is the angle of rotation.

3.1.1. GA Chromosome structure

Each parameter is encoded as an 8-bit binary number hence \(w\) and \(h\) take values between 0 to 255 pixels. \(x\) and \(y\) are calculated as a percentage of the image width and height and \(\theta\) as a percentage of \(2\pi\).

The chromosome size is number of ovals \(\times 5\) (parameters) \(\times 8\) bits (or number of ovals \(\times 40\) bits).

3.1.2. PSO position structure

Continuous PSO was used, and as such the particle position took the form of an array of floating point numbers. The array contained number of ovals \(\times 5\) (parameters) numbers.

The pseudo-code for the generation of the model image is shown below.

```
for( i; i < imageHeight){
    for( j; j < imageWidth){
        for( k=1, k < numberOfOvals){
            if((j-x)^2 + (i-y)^2 < (w/h)^2) {
                model.pixel{ i,j } = white;
            } else{
                model.pixel{ i,j } = black;
            }
        }
    }
}
```

3.2. Evaluation of fitness

The evaluation function returns a fitness value, indicating how well the model matches the original image/cluster.

3.2.1. Standard evaluation

A simple evaluation is used initially to evaluate models. This evaluation simply compared every pixel in the model and original image. It then determined how similar they are as a percentage. The result is squared to improve separation of fit models.

3.2.2. Evaluation with penalty

A second evaluation function was developed that applies a penalty for expanding the ovals into the black regions of the original image.

Following is the pseudo code for evaluation with penalty.

```
Input: original image, model
Output: fitness value
{
    for( every pixel in originalImage){
        if( originalImage.pixel = model.pixel) count++;
        if(( originalImage.pixel = black) && (model.pixel = white)) count -= penalty;
    }
    percentSimilar = 100 × count / width×height
    return percentSimilar^2
}
```

3.3. Incremental segmentation with thresholding

The algorithm implemented utilises an evolutionary optimisation algorithm (EOA). In this case either a Genetic Algorithm [7],[8] or a Continuous Particle Swarm Optimisation [9],[10], is used to fit a number of model ellipses to the original image/cluster. Initially one model oval is utilised and a population of such models is evolved using the implemented EOA. This evolution continues until an initial threshold is reached or the maximum number of evolutions has been reached. Having passed either condition the population is re-initialised without sacrificing any of the parameters of the existing fittest model. An additional model ellipse is also added to each member of the new population.

The EOA is then run again on the new population until a defined increase on the fitness value of the previous iterations fittest individual is achieved. If this is not achieved the population stops evolving when the maximum number of evolutions is reached.

The pseudo-code for this algorithm is shown below.

```
Main loop,
Inputs: population size
Output: fittest chromosome/model
```
MAX_NUM_OVALS;
MAX_NUM_EVOLUTIONS;
INITIAL_THRESHOLD;
CUTOFF[MAX_NUM_ITERATIONS];
//array indicating the point at which
//the evolution should be terminated.
//Each position indicates a number
//greater than 1, i.e. 1.15 corresponds to
//a 15% increase on the previous
//iterations fittest chromosome

initialize EOA with
1) Appropriate Operators
2) Evaluation (Fitness) Function

for( n=1; n <= MAX_NUM_ITERATIONS){
    initialise model(n);
    if( n = 1){
        population = randomInitialPopulation(
        populationSize);
    }
    else{
        population = seededPopulation(
        fittestModel, populationSize);
    }
    e=0;
    do{
        population.evolve();
        fittestVal =
        population.getFittestValue();
        e++;
    }while( (e <= MAX_NUM_EVOLUTIONS)
    && (( fittestVal < THRESHOLD) ||
    (fittestVal<
    CUTOFF[n]×prevFittest)));
    prevFittest =population.getFittestValue();
    fittestModel =
    population.getFittestModel();
}

4. RESULTS

Four images were used to evaluate the performance and
effectiveness of the algorithms. The test images consisted
of two artificial images. These tested the algorithms' ability to accurately match ovals to images with sharp
corners and other abnormal shapes. The other two test
cases were obtained from actual MRI brain scans. They
were filtered and thresholded [11], to obtain a binary
representation. These brain images test the effectiveness
of the algorithms on complex real life images. The
maximum number of ovals the incremental algorithms
used was 6. Therefore, for comparison, the model used in
the regular algorithms was initialised with 6 ovals. Figures
1, 2, 3 and 4 show a selection of results from each of the
algorithms.

It can be seen that the Regular GA found the general
shape but seemed to have difficulty finding the wheels.
Incremental GA always found the wheels. Increasing the

Figure 1: Test image (car.pgm) and the best results for the four algorithms

Figure 2: Test image (dog.pgm) and the best results for the four algorithms
number of evolutions would have improved this result. Regular PSO quickly converges on a local maximum and gets stuck with little or no chance of improvement. The result for incremental PSO clearly shows the affect of introducing the ovals one at a time as more components of the shape have been identified. Despite this, it still does not achieve as good a result as that for the GAs.

Regular GA had more difficulty with this image. It always managed to place ovals in roughly correct positions, but did not manage to achieve very close approximations in most of the results. Of all the tests performed, Incremental GA easily performed the best out of the four algorithms. It did, however, have some difficulty with the sharp edges.

Regular PSO was clearly the worst result for this test image, with the other three all achieving similar results. Both incremental algorithms found the small protrusion on the left hand side, while regular GA only found it once. Another aspect that should be noted is that both incremental algorithms “overfit” the earlier ovals. This shows that when the algorithms are manipulating models with one oval, the best approximation tends to be an over-simplified approximation, resulting in loss of finer detail. The modified evaluation function prevented this occurring. The results from using this evaluation function are by far the best.

This image has a number of distinct regions that are not significantly separated, making it difficult for all the algorithms to approximate. The incremental algorithms
again suffer from overfitting, while regular GA performed reasonably well. The best results were obtained by using the modified evaluation function with incremental GA.

<table>
<thead>
<tr>
<th>Table 1: Average fitness value for all algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Table" /></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Incremental GA (Max)</th>
<th>Incremental PSO (Max)</th>
</tr>
</thead>
<tbody>
<tr>
<td>119.5 (200)</td>
<td>55.75 (100)</td>
</tr>
<tr>
<td>408.75 (600)</td>
<td>209.25 (300)</td>
</tr>
<tr>
<td>414.5 (600)</td>
<td>166.25 (300)</td>
</tr>
<tr>
<td>453.5 (600)</td>
<td>244 (300)</td>
</tr>
<tr>
<td>547.25 (600)</td>
<td>278.5 (300)</td>
</tr>
</tbody>
</table>

Table 1 summarises the results discussed above and allows a direct comparison between the different algorithms.

<table>
<thead>
<tr>
<th>Table 2: Average number of evolutions performed by incremental algorithms</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Table" /></td>
</tr>
</tbody>
</table>

5. CONCLUSIONS

It has been shown that the two algorithms developed do provide accurate segmentation of images. Of the four algorithms developed, incremental GA works the best. Incremental PSO performs nearly as well. Regular GA and regular PSO do not perform as well as either the incremental algorithms. This shows that in general, irrespective of the optimisation algorithm used, the incremental version performs better than the regular version. This result shows the benefit of the algorithm being able to focus on fitting one oval at a time rather than being swamped when trying to manipulate the entire model at once.

It also appears that the Genetic Algorithms worked better than Particle Swarm Optimisation. This is due to the diversity maintained by the GA. PSO tends to quickly converge on local maxima, decreasing the search space covered.

Although the incremental algorithm outperformed the regular algorithm, it suffered from the problem described as overfitting. This occurs early in the iterations when the algorithm is trying to obtain the best solution using only one oval. The evaluation function with penalty prevents this problem by applying a penalty if the model is outside the white regions of the original image. This modified evaluation improved the results obtained using the incremental GA on complex images.

The biggest limitation of the algorithms are the running times which were around 25-30 minutes each when run on a 1GHz Pentium. With further work this run time could be reduced but due to the evolutionary nature of the algorithms it is unclear by how much.

7. REFERENCES