Layout of Garment Patterns for Efficient Fabric Consumption

Suthep Madarasmi[1] and Phoomsith Sirivarothakul[2]
[1] Department of Computer Engineering, Faculty of Engineering
King Mongkut's University of Technology Thonburi
91 Pracha-Uthit Road, Bangkok 10140, Thailand
Tel. +66-2-470-9085, Fax.: +66-2-872-5050
[2] Department of Industrial Engineering
King Mongkut's University of Technology Thonburi
91 Pracha-Uthit Road, Bangkok 10140, Thailand
Tel. +66-2-586-7892 Fax: +66-2-912-7637
Email : suthep@cpe.eng.kmutt.ac.th, phumsit@hotmail.com

Abstract: This paper presents the use of a Genetic Algorithm to find the optimal layout for the placement of garment patterns on a fabric of fixed width to minimize fabric waste. We developed a program to simulate garment pieces and their layout on a fixed-width fabric. Each piece in the order book is placed with 2 possible orientations: 0 degrees and 180 degrees. The efficiency is measured by the length of fabric used after all the patterns in the order book have been laid out. A comparison is made between the placement using our proposed genetic algorithm to that made by an expert human using our simulation program. The results from our experiments on various pattern designs indicate that our genetic algorithm can effectively be used to obtain highly efficient solutions, comparable to that done by an expert while using a reasonable amount of time. The algorithm can also be adapted for use in other areas related to optimal consumption of sheet material such as metal, paper, and leather.

1. Introduction

The placement of garment pieces or markers to use the minimal amount of fabric is an important concern for the garment production industry, since fabric cost can be relatively high. Each piece of garment part such as collar, pocket, or arm comes in different shapes and sizes, some symmetrical and some asymmetrical, presenting a challenge to finding a good placement solution, even for an expert. Much effort has been devoted to automate this process by using artificial intelligence and optimization techniques. This project presents a methodology to generate a suitable shape layout solution using a Genetic Algorithm.

Genetic Algorithms (GA) [1-5] belong to a class of probabilistic search methods obtained from simulating the evolution of life. The basic idea is that one starts with an initial population of individuals, each corresponding to a solution usually starting with randomly defined values. New solutions to the population are then generated via reproduction, allowing only the fittest n individuals of the population to survive. The fitness of each individual is measured by some function that measures the efficiency of the solution. An offspring from reproduction is created by the crossover or mutation operation. Crossover involves the mixture of traits between 2 or more existing members within a population or sometimes within a single individual. Mutation is usually accomplished by adding some random element to an individual in the population or during the crossover process. After several generations of reproduction and survival of the top n fittest individuals, the fittest individual is reported as the solution. Thus, in a GA system, several solutions represented by the individuals within the population are available at any instance in time, unlike other stochastic optimization methods where usually only one solution is iteratively being improved.

Pargas and Jain [8] describes a stochastic approach to the problem of bin packing two-dimensional figures in a rectangular area efficiently. The techniques employed are similar to those used in genetic algorithms or in simulated annealing algorithms. The shapes used are fairly regular and they achieved 80% efficiency, which in their experiments they used objects that could ideally be packed at 100% efficiency. For garment piece layout, the problem is more complex since each piece to be laid is an irregular shape and cannot be packed at 100% efficiency. Thus, the efficiency must be measured by comparing to the layout done by an expert human. In our research work, we use a very similar representation to their model for the chromosome or strip definition.

Roussel and Maouche [9] presents an algorithm for the shape layout problem particularly fitted to the cutting of garment pieces by an integrated continuous system. They used a heuristic tree search algorithm called ε-admissible algorithm. The algorithm achieved reasonable results, although suffers from time spent back-tracking and has a possibility of getting stuck at a local minimum solution. In the following year, Ismail and Hon [11] researched on the minimal use of raw materials for cutting 2-D shaped objects by using genetic algorithms and heuristics. Their experiments were conducted using simple shapes without limiting the width of the sheet material, making it easier to place the objects, although less realistic since raw material sheets generally do have a fixed width.

Bounsaythip et al [10] aimed to mark all the shapes to be cut onto the sheet by minimizing unoccupied space. As the problem involved many sub-optimal solutions or many local optima. A genetic like algorithm is used to handle the layout process. Each shape is represented as a Comb Code. The efficiency is measured by the length of a fixed width raw material sheet such as fabric used. They compared their results using pant pieces for genetic algorithms, genetic algorithm with simulated annealing, and tree search. The pant pieces they used had fairly regular shapes, obtaining high efficiency results.

In a related work Bounsaythip and Maouche [12] used an evolutionary method to solve a garment shape nesting and placing problem. The structure used by the evolutionary algorithm is organized into a hierarchical tree, which is similar to the representation used in genetic programming. Using the comb code representation, they place the shapes in a nesting operation and use a tree organization to represent the arrangement. They allow each shape to be placed at 0, 90, 180, and 270 degree orientations. The crossover operation is done by combining parts of the tree together, while removing shapes that are more than the number allowed in order book. They tried their experiments on relatively simple shapes, making it difficult to assess the efficiency achieved.

Genetic Algorithms seem to be well-suited for garment placement, especially since it does not suffer from local minima problems. The variations in the various work on the use of GA stems from the representation and the use of differing crossover and mutation methods. In the next sections we present our variation including the problem representation, algorithm, and experimental results.

2. Problem Representation

Figure 1 shows an example of an order book for 1 garment that could be a shirt. If 2 shirts are required, the placement on a fixed width fabric is illustrated in figure 2. In each placement, the marker may be at 0 degrees or 180 degree orientations. In our experiment, each garment piece is modeled as a polygonal object. Objects that have smooth boundaries such as quadratic splines, must be approximated by a polygonal surface such as convex hull [7] prior to starting, since we model each garment piece as a polygonal object.

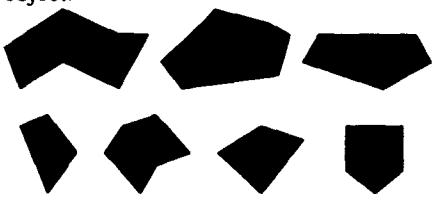

Figure 1. Sample pieces for 1 garment piece.

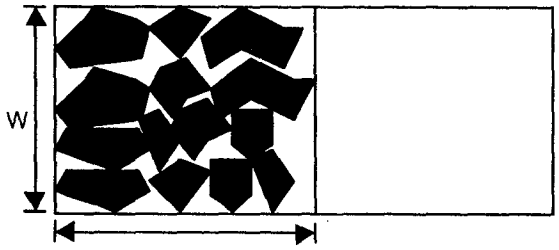

Figure 2. Placement on fabric of fixed width.

For a given problem with N pieces illustrated in figure 1, each piece has a default initial orientation, but may be rotated 180 degrees. A solution or individual is a structure with the following format:

S = [(F1, O2), (F2, O2), (F3, O3), ..., (Fn, On), L]

where S is the solution from the order book, F represents each garment piece, O is the piece's orientation: 0 for 0 degrees or 1 for 180 degrees, and L is the length (cost) of the solution.

Figure 3 shows a particular individual where each chromosome is denoted by a strip of a particular length. Items are randomly selected from the order book and placed on the fabric. This model of chromosome is similar to that in [8]. Each piece is placed starting at the upper-left edge of the fabric. If there is no space to place the piece, we move downwards until there is space or we run out of fabric width. If we reach the bottom of the fabric width without finding space, then we attempt to place the piece towards the right. When placing a new piece we must check that that space is available. We do this by a simple 2-D graphics algorithm of checking that none of the vertices of our polygon is inside another, previously placed polygon [6, 7]. Thus, we do assume that all pieces are made of convex polygons. For non-convex shapes, we convert them to 2 or more pieces of convex shapes before we begin our procedure (see [7] for converting a non-convex polygon to 2 or more convex polygons).

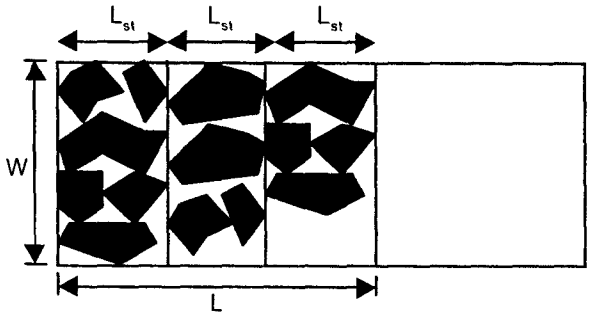

Figure 3. Pieces are placed to form strips representing chromosomes. The entire order book placement forms a single individual or solution.

3. Our Genetic Algorithm

The genetic algorithm used in this paper is outlined in Figure 4.

3.1. The Initial Population

The initial population is created by randomly generating N individuals. Each individual is created by randomly picking items from the Order Book and placing them on the fixed width fabric.

3.2. The Fitness Values

Each strip shown in figure 3 represents a chromosome. The fitness of each chromosome is defined as percent area efficiency of each chromosome. For each individual, the fitness of the individual is the length of the fabric used. Only the fittest N individuals are maintained in the population.

3.3. Reproduction: Creating a New Individual

A new individual may be created by either crossover or mutation. In the crossover step, we first begin with a new individual with no chromosomes corresponding to a fabric with no marker pieces. We then use a linearly biased random number generator [8] to pick a chromosome or strip section from our population. After selecting a chromosome to become part of a new individual, the pieces in that chromosome will decrease the pieces required in our order book. Thus, we must compute the efficiency of each strip

or chromosome in the remaining population again. For example, it is possible that our new individual already has all the collar pieces needed and any population containing a chromosome with a collar will now be rendered unfit and undesirable. Once the efficiency is recomputed, we can again sample for a fit chromosome to continue to create our new individual.

After picking a few chromosomes the items in the order book will be reduced and we will be left with no more chromosomes in our population with a good enough efficiency. At that point we simply take the remaining items in the order book and insert them in, to complete our new individual.

Table 1. The Genetic Algorithm used in this paper.

```
1. Set and Initialize the population by
   generating a random solution,
   determining its length and keep a sorted
   list.
2. Divide each individual into chromosome
   strips (Figure 3) using the width of the
   largest piece. Compute the efficiency
   of each strip.
3. Crossover, Mutation, and Selection:
Repeat
   If (r = random (0, 1)) < (p = crossover
   probability)
      Create a New Individual by Crossover:
      Repeat
         • Sample to obtain a chromosome from
           the solution that has high
           efficiency (more efficient more
           likely to be picked)
         • Recompute the efficiency of each
           chromosome based on what is still
           needed in the order book
      Until No Efficient Chromosome
      Available
      Fill Remaining Solution Randomly by
      remaining items in the Order Book
   Else
      Create New Individual by Mutation:
      Fill Solution Randomly by Order Book
   Insert the solution into the new
   population
   Check for Natural Selection (Only Top N
   survive, Least fit individual dies)
Until(Population has converged)or(No
improvement in Length of Best
Solution)or(NumIterations > MaxIterations)
```

3.4. Reporting a Solution

Once the system has converged, we pick the individual with the best fitness and report its configuration as the solution.

4. Experimental Results

We first experimented with rectangular marker pieces of 7 different sizes, each of 6 sets, totaling 42 pieces. Figure 4 shows our algorithm results, which happens to be very similar to the human expert. The efficiency for both is 93.9%.

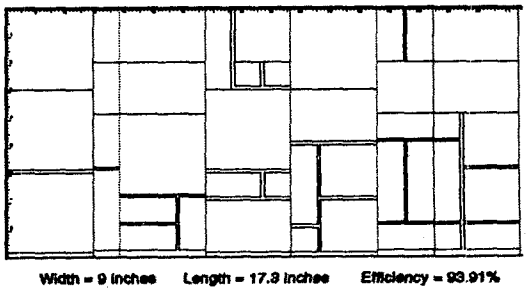

Figure 4. Result of the GA for rectangular pieces.

Next, we experimented with 20 marker pieces for an actual shirt shown in figure 5, used to produce 10 shirts. Figure 6 and 7 shows results for GA vs. Human Expert with 74.7% vs. 78% efficiency, respectively.

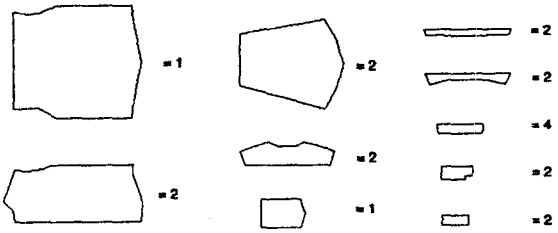

Figure 5. Shirt pieces used in our first test case.

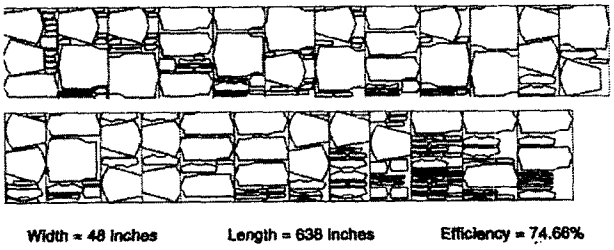

Figure 6. Actual shirt parts experimental result.

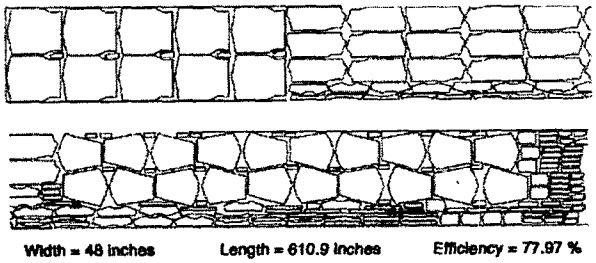

Figure 7. Actual shirt result done by human expert.

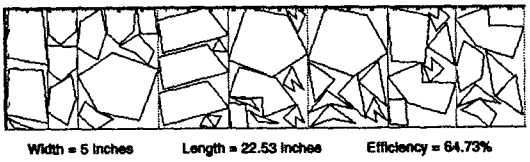

Figure 8. GA results for multi-edged shapes.

We also tested our GA on complex, multi-edged pieces of 15 templates in 3 sets. Figure 8 and 9 compares GA vs. Human expert with efficiency 64.7% vs. 70.8%.

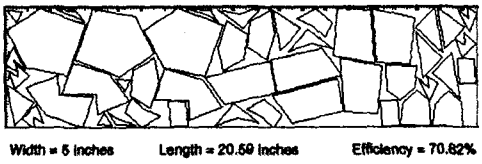

Figure 9. Human expert on multi-edged shape.

5. Discussion and Conclusion

There are many parameters that need to be determined for our algorithm, the two most important being the population size and the number of strips to crossover. Before deciding these values we performed an analysis using several experiments. Figure 10 shows the results of our analysis to determine the population size, showing the efficiency after each iteration for population size of 10, 15, 20, 25, 30. From the graph it is clear that the results of population sizes 15 through 30 are very similar. Thus, to reduce computation time, we used this information to fix our population size throughout our experiments to 15.

Figure 11 shows the graph of efficiency per iteration for our choices of the number of strips to select to crossover at one time. From the graph, we concluded that 3 strips would be most efficient.

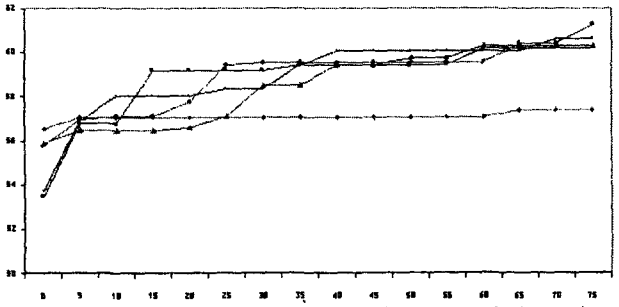

Figure 10. Finding the appropriate population size. The lowest graph is for size 10, so we pick 15.

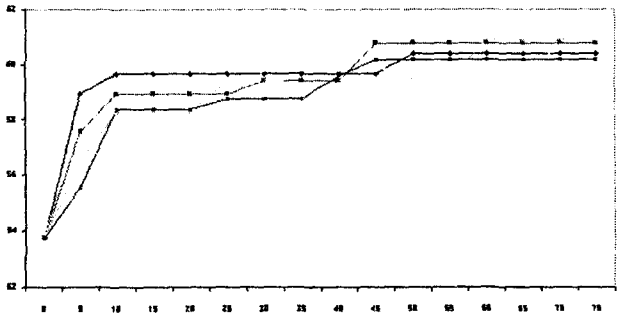

Figure 11. Finding the number of strips to crossover at once. The choice of 3 strips yields highest efficiency.

Our experimental results indicate that our GA is reasonably good, although human experts generally obtain a 3%-5% higher efficiency. This is consistent with the comparisons in other AI algorithms of this kind. In our experiments we observed that during crossover, when a new individual is created by choosing good chromosomes or strips from the available population, only a few good strips or chromosomes can be used. After the first few strips the pieces in the order book is reduced and the new individual started to have pieces that were already available in the other good strips among the available population. Thus, the available good strips or chromosomes were soon rendered useless. In our future work we intend to use a new crossover scheme that will avoid this problem by finding partial solutions using the good strips or chromosomes and then restarting the GA to find good solutions for the remaining garment pieces in the order book that have not been used.

A comparison of our results with other researchers is a rather difficult one, since the overall efficiency depends on the shape of the patterns used. The more complex the shape, the less the overall efficiency. We need to compare our technique with that of other researchers by using the same test data. Before we do that a standard test data set has to be established. Meanwhile, we choose to compare our results to that of human experts and attempt to improve the efficiency by 3-5 percent. In the future we plan to fine tune our system to achieve this objective.

References

[1] T. M. Michell, *Machine Learning*, New York, McGraw-Hill, pp. 249-250, 1997.

[2] E. Falkenauer, *Genetic Algorithms and Grouping Problems*, Chichester, John Wiley & Sons, 25-58, 1998.

[3] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, Berlin, Springer-Verag, 1992.

[4] D. E. Goldberg, *Genetic Algorithms in Search, Optimization, and Machine Learning*, Reading MA, Addison-Wesley, pp. 1-23, 1989.

[5] A. M. S. Zalzala and P. J. Fleming, *Genetic Algorithms in Engineering systems*, London, The Institution of Electrical Engineers, pp. 1-19, 1997.

[6] S. Harrington, *Computer Graphics*, New York, McGraw-Hill, pp. 3-5, 70-76, 1987.

[7] D. Hearn and M.P. Baker, 1994, *Computer Graphics C Version*, NJ, Prentice-Hall, pp. 235-236, 1994.

[8] R. P. Pargas and R. Jain "A Parallel Stochastic Optimization Algorithm for Solving 2D Bin Packing Problems", *IEEE Proc. of the 9th Int. Conf. on Artificial Intelligence for Applications*, pp. 18-25, 1993.

[9] G. Roussel and S. Maouche, "Improvements About Automatic Lay-Planning For Irregular Shapes on Plain Fabric" *IEEE Proc. of Int. Conf. on Systems Man and Cybernetics, System Engineering in the Service of Humans*, v. 3., pp. 90-97, 1993.

[10] C. Bounsaythip, S. Maouche and M. Neus "Evolutionary Search Techniques Application in Automated Lay-Planning Optimization Problem", *IEEE International Conference on Intelligent Systems for the 21st Century*, v. 5, pp. 4497-4502, 1995.

[11] H.S. Ismail and K.B. Hon, "The Nesting of two-dimensional Shapes Using Genetic Algorithms" , *Proceedings of The Institution of Mechanical Engineers Part B, Journal of Engineering Manufacture*, v. 209 (B2), 1995, pp. 115-24.

[12] C. Bounsaythip and S. Maouche, "Irregular Shape Nesting And Placing With Evolutionary Approach", *IEEE Int. Conf. On Systems Man and Cybernetics, Computational Cybernetics and Simulation*, v. 4, pp. 3425-3430, 1997.