Evolutionary Path Planning for Robot Assisted Part Handling in Sheet Metal Bending

Xiaoyun Liao G. Gary Wang[*]
Dept. of Mechanical & Industrial Engineering,
The University of Manitoba
Winnipeg, MB, Canada, R3T 5V6
Tel: 204-474-9463 Fax: 204-275-7507
Email: gary_wang@umanitoba.ca

Abstract

While the bending sequence planning has been intensively studied, design of the motion path of a sheet metal part in the bending operation tends to be ignored by researchers. Because during the bending operation, the space for maneuvering a sheet metal part is very small, collisions between the part and bending tools are likely to occur. When a robot is used to handle the part, the role of an automatic path-planning tool becomes more significant. In this study, an evolutionary path-planning approach for robot-assisted handling of sheet metal parts in bending is firstly proposed and implemented. The proposed approach globally searches the motion path space to identify feasible paths. Collision detection algorithms based on segment intersection are used to check if the generated paths are feasible or infeasible. This method can automatically design feasible handling operations for a robot. Simulation examples on a simple "V" shaped part and a part with multiple bents demonstrate that the approach is efficient and practical.

Keywords: evolutionary algorithms; path planning; collision detection; sheet metal bending; robot-assisted handling, genetic algorithms

[*] Corresponding Author

Introduction

With the development of the manufacturing automation technology, the sheet metal fabrication applies CNC equipment such as a CNC press brake [1] (see Fig.1), which enhances the productivity for small batch manufacturing. In such a system, a programmed robot is often used to handle materials and/or parts during the sheet metal bending process [2] (see Fig 2).

In the robot-assisted sheet metal bending, the most difficult task is to plan the bending sequence[3]. A feasible bending sequence should ensure that the robot grasps the sheet metal part and moves it to a dedicated position without any collision with tools between the bending operations or during the feeding operations[4]. Many researchers focused on the bending sequence planning problem and proposed some methods and algorithms to generate an "optimal" sequence[3-7], but few people have studied the problem of sheet metal part's path of motion in the environment of a bending machine. In fact, the path planning of the sheet metal part should be one of the components in bending sequence planning since it can be applied to detect the feasibility of bending operations. Eyal and Thomas [8] did the early work on the sheet metal part's motion in 1994. They proposed a local search method based on the configuration space and a potential function to generate the motion path by combining special heuristics with the recognition of critical features of the sheet metal component. Their local search method, however, did not check the collision in the whole workspace between the sheet metal part and bending tools, therefore the method might fail to find feasible paths for some parts with complex shape.

Similar to other motion planning problems, such as the robot motion planning, the path planning of sheet metal parts is computationally expensive. It is in fact a NP problem[8]. Recent advances in machine intelligence have led to the application of modern heuristics, such as evolutionary algorithms, to solve the motion-planning problem. Due to their parallel, global and stochastical search mechanism, evolutionary algorithms are expected to improve the efficiency for a motion planner. For example, Page et al. [9] applied traditional genetic algorithms without utilizing domain-specific knowledge to develop a robot path plan. Xiao et al. [10] presented an evolutionary planner/navigator (EP/N) for path planning and navigation that incorporates domain-specific knowledge to deal with different optimization criteria. Hocaoglu and Sanderson [11] proposed a multi-resolution representation for a multi-dimensional path. Based on this approach they developed an evolutionary computation method for the multi-path planning and multi-dimensional planning. However, such robot motion planners cannot be directly applied to design the path of the sheet metal part in the bending operation, because these proposed robot motion planners do not consider the collision of the handled object with the bending machine but only the robot itself within the environment. Moreover, the moving space of sheet metal part with multi-bent profiles is comparatively small, thus collision is a critical issue in designing the motion path of the sheet metal part.

In this paper, an evolutionary path planning approach for the robot-assisted sheet metal handling is firstly presented. The collision detection is performed by using the simple segment intersection algorithm[12]. Several evolutionary operators are also set up for this problem. Test examples demonstrate that this approach is efficient and practical.

Sheet Metal Part Path Planning Problem

During the robot-assisted sheet metal bending process, the sheet metal part, which attaches to a robot, should be able to move freely in a constrained manufacturing environment. This environment is constrained by the bending machine stroke, the top tool (punch), the bottom tool (die), and the robot (see Fig.3). The sheet metal part path planning problem can be stated as follows[8]: "Given a sheet metal part, bending machine and robot, it is desired to find a feasible trajectory for the robot controlled part to undergo during the entire bending operation".

In this study, we simplify the path planning problem for the sheet metal part as a Two-Dimensional (2D) problem, and assume the robot has enough degrees of freedom (DOFs) to handle the part.

Collision Detection

Collision detection plays an important role in the path-planning problem. When robot drags the sheet metal part from one location to another, we should ensure that there is no collision between the moving part and tools during the motion. Therefore collision detection algorithms are applied. A lot of collision detection algorithms in literatures can be used[11]. In the present study, since we simplify the problem as a 2D problem, so the sheet metal part and tools can be respectively considered as a group of many 2D line segments. A simple collision detection algorithm based on segment intersection[12] is developed to check the collisions.

If we view a motion path as line segments defined by many node points along the path, at each motion node the segments of the sheet metal part should not intersect with the line segments of tools. If there is an intersection detected, the current motion node is thus infeasible; otherwise, the node is feasible. Therefore for a path consisting of many nodes, if all nodes along a candidate path are feasible, the path is thus feasible; otherwise, the path is infeasible. The collision detection algorithm is given as below.

Collision_detection algorithm:

Preparation model/file: the sheet metal part model, punch model, and die model

Input: a path (presented by a sequence of points)

Output: if the path is feasible or infeasible

BEGIN:

```
1) N      ← the number of nodes on a given path;
2) Bool ← a Boolean number initialized as 0;
3)   While (n < N) do
4)        C_Part(P) ← the configuration of sheet metal part at node P;
5)        If ( intersection between C_Part(P) with tools)
6)        Bool ← Bool+1;
7)   EndWhile;
8) Output Bool; ← Bool=0 indicates a feasible path; otherwise an infeasible path
END
```

In the proposed evolutionary path planning approach, the above collision detection algorithm is developed to check if the nodes on a path are feasible or infeasible, and search for feasible paths. As the search for a feasible path is challenging and computationally intensive, a special evolutionary algorithm is designed and implemented in this study.

Evolutionary Path Planning

An evolutionary algorithm normally consists of three important components: the encoding scheme, fitness function, and evolutionary operators. The encoding scheme and fitness function transform a real physical problem to the "language" of an evolutionary algorithm. The evolutionary operators mainly influence the efficiency and convergence of the algorithm. Following sections will describe the three components of the proposed evolutionary approach for the sheet metal path-planning problem.

Encoding Scheme

In current study, a "real" scheme is used to form chromosomes[10]. A chromosome is defined as an ordered list of path nodes as shown in Fig 4. Each node of a path, i.e. *gene* of a chromosome, is presented by the "real" x and y coordinates at the node, its rotation angle θ, which denotes the rotation of the sheet metal part with respect to a fixed global coordinate system, and a Boolean variable *b*, which indicates if the given node is feasible or not. At each node on the path (motion node) defined by (x, y, θ), collision detection will be performed. If no collision between the sheet metal part and bending tools, this node is denoted to be feasible, otherwise, it is infeasible.

Fitness Function

To measure the feasibility of a given path, each path will be assigned a "fitness" value. For the sheet metal part path-planning problem, the fitness value of path P is defined by Eq. (1).

$$\text{Fitness} = \text{fea\_N}(P) / N(P) \tag{1}$$

where fea_N(P) is the number of feasible nodes of path P; N(P) is the total number of nodes of path P.

Thus, the fitness value of a candidate path should be 1.0 if the path is feasible; for any infeasible path, its fitness value is between 0.0 and 1.0.

Having setting up the coding scheme and fitting function, the goal is to search for feasible paths having a fitness value equal to 1 through evolutionary operations.

Evolutionary Operators

Several evolutionary operators are set up, which include one reproduction, one crossover operator and four mutation operators. These operators manipulate the genetic material, *gene*, in the encoded path representation. Their features are described below[10,11,13,14].

Reproduction: It uses the roulette wheel method to generate mating pool from the previous generation[14]. That is, the higher the fitness value, the more likely the chromosome being selected to the mating pool. Sometimes, in order to increase the likelihood of the chromosome with a higher fitness value being selected, a linear fitness-scaling scheme is usually applied, which was suggested by Goldberg[13] .

As shown in Fig.5, in this fitness-scaling scheme, the paths in the pool are firstly classified into two groups by the average fitness value of the population. The average fitness u_{avg} is mapped to f_a, and the maximum original fitness u_{max} is mapped to f_b. If the original fitness is smaller than u_{avg}, it is linearly mapped to $[0, f_a]$. If the original fitness is greater than u_{avg}, it is linearly mapped to $[f_a, f_b]$. Thus, the new fitness function is defined as:

$$new\_fitness = \begin{cases} (f_a / u_{avg}) * u \, , & u \le u_{avg} \\ f_a + (u - u_{avg}) * (f_b - f_a) / (u_{\max} - u_{avg}), & u > u_{avg} \end{cases} \qquad (2)$$

where u is the original fitness. The f_a and f_b should be selected to be adaptable to the problem at hand. As one can see from Fig. 5, via the scaling, the difference between the original fitness value zero to u_{avg} is shortened to $[0, f_a]$; while the difference between u_{avg} and u_{max} is increased to $[f_a, f_b]$. Since such a difference represents the likelihood of selection, thus the chromosomes having a fitting value larger than u_{avg} will have a larger chance of being selected after the scaling. The scaling constants, f_a and f_b, can also be understood as "greediness" control factors. If the scaling effect is strong, the evolutionary algorithm might converge quicker at an increased risk of missing the global optimum. On the other hand, if the scaling effect is weak, the probability of reaching the global optimum is higher but the convergence speed might be too slow. Thus there is always a trade-off between the convergence speed and the accuracy of the final solution. In our testing examples we set f_a=0.5 and f_b=3.5 (See Section Implementation and Testing).

Uniform crossover: This operator selects two parent chromosomes (paths) based on the crossover probability. For a selected pair of paths, a template of equal length whose position value is "1" or "0" is randomly generated. Starting from the first position on the template, if the position value is "1", the two children will inherit the gene from the two parents, respectively; if the position value is "0", then the first child will receive the corresponding gene from the second parent and the second child will receive the gene from the first parent. The process continues until all the genes of the parents have been processed. For example, as shown in Fig.6 A), by using the generated template, the parent1 (i.e. Path: $P_1P_2P_3P_4P_5$) and parent2(i.e. path: $Q_1Q_2Q_3Q_4Q_5$) yield two children, Path: $Q_1P_2P_3Q_4P_5$ and path:$P_1Q_2Q_3P_4Q_5$. The 2nd, 3rd, and fifth genes are directly inherited from its corresponding parent; but the 1st and fourth genes are switched, so comes the name "crossover".

Inversion mutator: It's a reordering operator applied to the genes of a chromosome. This operator reverses the order of genes between two randomly chosen positions within the selected chromosome based on a given inversion probability. For example, in Fig.6 B), the path: $P_1P_2P_3P_4P_5P_6$ is changed into $P_1P_5P_4P_3P_2P_6$ by inversing the nodes between Position 2 and Position 5.

Flipping mutator: It operates on the infeasible node (gene) in the selected path. It flips the node P_k with respect to the line segment P_{k-1} P_{k+1}. For example, in Fig.6 C), the node P_4 in Path $P_1P_2P_3P_4P_5P_6$ is flipped into P'_4 with respect to P_3P_5, thus generating a new path $P_1P_2P_3$ P'_4 P_5P_6.

Perturb_1 mutator: It perturbs the randomly selected infeasible node (gene) in the selected path by a small amount based on a given operation probability. For example, in Fig.6

D), perturbing the node P_4 in path $P_1P_2P_3P_4P_5P_6$ with a small amount generates the new path $P_1P_2P_3\ P'_4\ P_5P_6$. This operator is helpful to gradually make the path smooth during the evolutionary process.

Perturb_2 mutator: It perturbs the infeasible node (gene) in the selected path by a large amount. As shown in Fig.6 E), this operator gives the candidate node a larger perturbing amount. This operator enables a relatively big adjustment on an infeasible node, thus improves the algorithms' speed of convergence.

Evolutionary Path Planning Algorithm

The evolutionary path-planning algorithm for sheet metal parts in a constrained workspace uses a "real" path representation and incorporates traditional genetic algorithm with the proposed evolutionary operators and evaluation method.

Each individual in the population represents a feasible or infeasible path, along which the sheet metal part moves out from the workspace. Each individual path is assigned a fitness value by checking the number of the feasible nodes on the path.

In the evolutionary loop, a set of paths is selected for generating new paths by crossover and mutation. An evolutionary operator is selected on the basis of its probability. The crossover operator changes two parent paths into two new offspring paths, while the mutator mutates a single path as a new child. The new offspring are added into the existing population and evaluated based on their fitness value. Then the fitness value of each path is modified by using the linear scale scheme, and the best ones are chosen as a new population. This evolutionary process terminates after a certain number of generations.

A high-level description of this algorithm is as follows:

Step 1: Randomly generate the initial population having *PopSize* individuals.

Step 2: Perform the collision detection for candidate paths.

Step 3: Assign a fitness value for each candidate path.

Step 4: If the termination criterion is not satisfied, go to Step 5; otherwise output the search results.

Step 5: Use the reproduction operator to form a "mating pool".

Step 6: Apply the crossover operator to the selected paths based on the crossover probability.

Step 7: Apply mutation operators to the selected paths based on its operation probabilities.

Step 8: Perform the collision detection for the newly generated paths.

Step 9: Assign a fitness value for each new path.

Step 10: Select the best *PopSize* individuals from the previous paths and the new paths.

Step 11: Go back to Step 4.

Implementation and Testing

The proposed algorithm has been implemented with Visual C++ and OpenGL in the PC environment. Two cases were studied to test the proposed approach.

Case 1: The sheet metal part has a "V" shape, which has to be moved out from a constrained workspace. The running parameters are: *POPSIZE* =90; *Max_Iteration*=150, *the probability of crossover* is 50%, *inversion*'s 20%, *flipping*'s 15%, *perturb_1*'s 5% and *perturb_2*'s 10%. Each path is defined with 25 nodes. Fig.7 shows one of the obtained feasible motion paths. As can be seen from Fig. 7, the "V" shaped part successfully moves out the constrained space without collision with the bending tools. Fig.8 gives the relationship between the number of generations and the average fitness value of each generation during this evolutionary search process. From Fig.8 we can see that, after only 8 iterations the evolutionary planner successfully finds feasible paths.

Case 2: This case is for a sheet metal part with a multiple-bent profile to represent a more realistic bending part. . The algorithm uses the same control parameters as in Case 1. Fig. 9 depicts one feasible motion path. Again, the path allows the part to be moved out of the space without collision with the tools. Fig.10 shows that the proposed algorithm is convergent and it takes only 60 iterations to find a feasible path.

From the testing results on Case 1 and 2, it is clear that the presented evolutionary path-planning algorithm for sheet metal parts moving in a constrained workspace is efficient and can be used to design the path plan for the robot-assisted handling of sheet metal parts.

Conclusion and Future Work

This paper proposes a new evolutionary approach to plan the sheet metal part's moving path in a constrained bending workspace. Tests of the approach demonstrate that the proposed evolutionary path planner is efficient and practical. It can automatically generate feasible paths for robot-assisted handling.

The future work might incorporate the consideration of the constraints from the robot to simultaneously obtain the optimal control law of robot. The developed method can also be incorporated into the bending sequence planning to fully automate the bending and handling operations.

References

1. Raabe J. The Fascinating World of Sheet Metal (in German). Verlags-GmbH, Stuttgart, Germany, 1996.

2. FANUC Robotics UK, 16/12/2002, Http://www.fanucrobotic.co.uk/products/system/

3. Inui M, Terakado H. Fast Bending Sequence Planning for Progressive Press-Working.Proceedings of the 1999 IEEE International Symposium on Assembly and Task Planning, Port, Portugal, July 1999: p.344-349.

4. Aomura S, Koguchi A. Optimized Bending Sequences of Sheet Metal Bending By Robot. Robotics and Computer Integrated Manufacturing Vol.18, 2002: p.29-39.

5. Duflou JR, Oudheusden DV., Methods for the Sequencing of Sheet Metal Bending Operations. International Journal of Product Research, Vol.37, No.14, 1999: p.3185-3202.

6. Gupta SK. Sheet Metal Bending Operation Planning: Using Virtual Node Generation to Improve Search Efficiency. Journal of Manufacturing Systems, Vol.18, No.2, 1999: p.127-139.

7. Thanapandi CM, Walairacht A., Ohara S. Multi-Component Genetic Algorithm For Generating Best Bending Sequence and Tool Selection in Sheet Metal Parts. Proceedings of the 2001 IEEE International Conference on Robotics & Automation, Seoul, Korea, May 21-26, 2001: p.830-835.

8. Eyal Z., Thomas H. A planning Approach For Robot-Assisted Multiple-Bent Profile Handling. Robotics and Computer Integrated Manufacturing, vol.11, No.1, 1994: p.35-40.

9. Page WC, McDonnell JR, Anderson B., An Evolutionary programming Approach To Multi-dimensional Path Planning. Proceedings of 1st Annual Conference on Evolutionary Programming, May 1992: p.63-70.

10. Xiao J, Michalewicz Z, Zhang L. Adaptive Evolutionary Planner/Navigator for Mobile Robots. IEEE Transactions on Evolutionary Computation, Vol.1, April 1997: p.18-28.

11. Hocaoglu C, Sanderson AC. Planning Multiple Paths With Evolutionary Speciation. IEEE Transactions on Evolutionary Computation, Vol.5, No.3, June 2001: p.169-191.

12. O'Rourke J. Computational Geometry in C (Second Edition). Cambridge University Press, Cambridge, UK, 1998.

13. Goldberg DE. Genetic Algorithms in Search, Optimization and Machine Learing, Addision-Wesley, Boston, MA, USA.,1989.

14. Gen M, Cheng RW. Genetic Algorithms and Engineering Design. John Wiley & Son Inc., New York, USA., 1997.

List of Figures

Fig. 1 A CNC press brake[1].

Fig. 2 A robot-assisted sheet metal bending system[2].

Fig. 3 The constrained environment of the moving sheet metal part.

Fig. 4 A chromosome representing a path.

Fig. 5 The linear fitness-scaling scheme[13].

Fig. 6 Designed evolutionary operators.

Fig. 7 An obtained feasible motion path of the “V” shaped part.

Fig. 8 The number of generations vs. the average fitness value for Case 1.

Fig. 9 An obtained motion path of a multiple-bent sheet metal part.

Fig. 10 The number of generations vs. the average fitness value for Case 2.

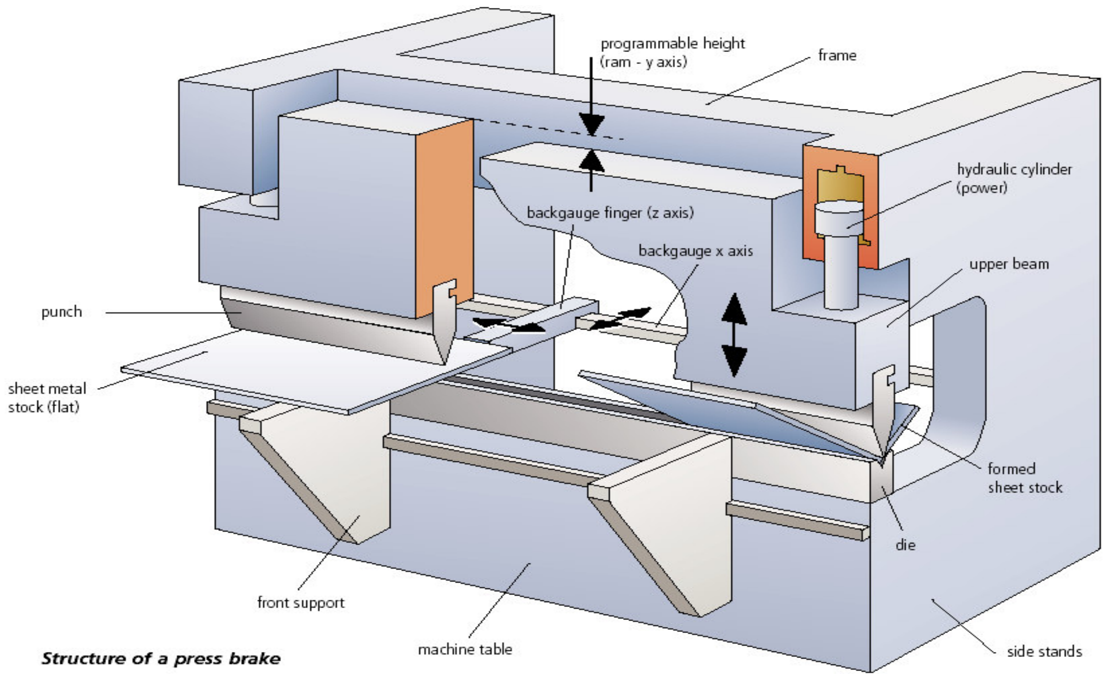

Fig 1 A CNC press brake[1].

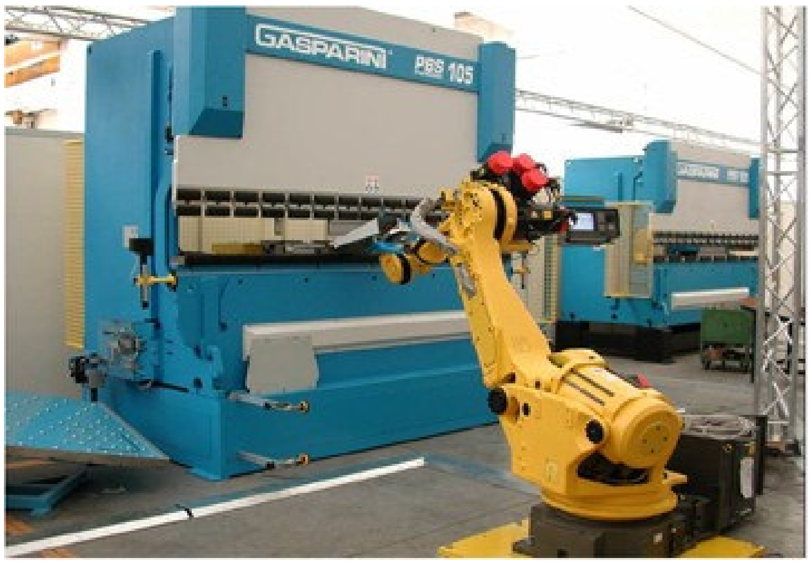

Fig 2 A robot-assisted sheet metal bending system[2].

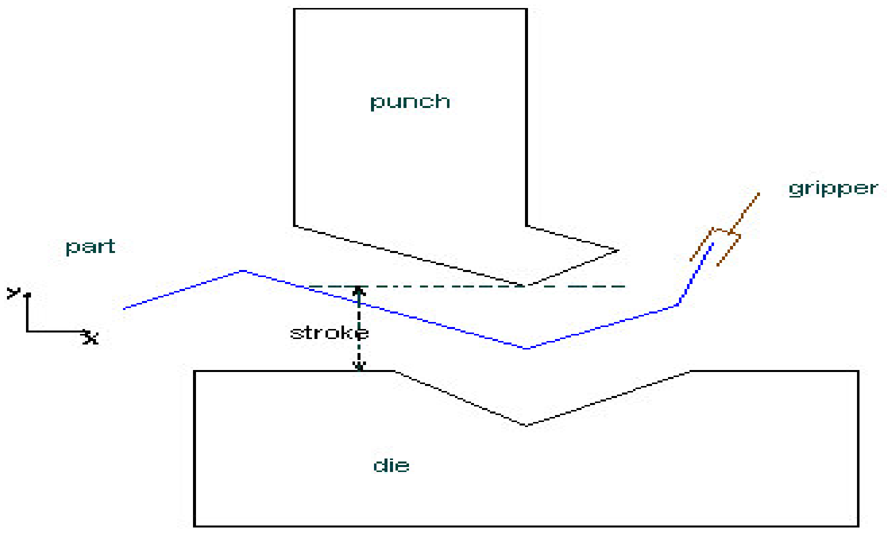

Fig.3 The constrained environment of the moving sheet metal part.

X_1	Y_1	θ_1	b_1	X_2	Y_2	θ_2	b_2	…	X_n	Y_n	θ_n	bn

Fig.4 A chromosome representing a path.

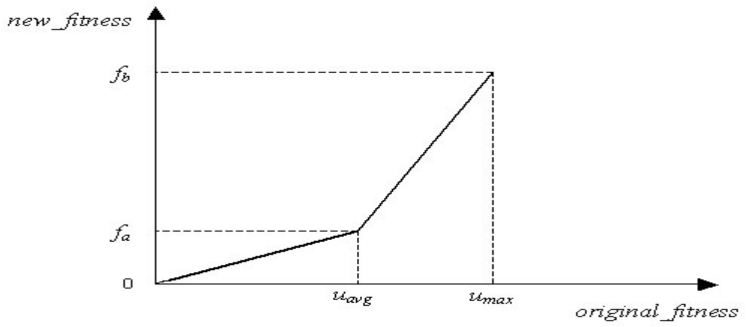

Fig.5 The linear fitness-scaling scheme[13].

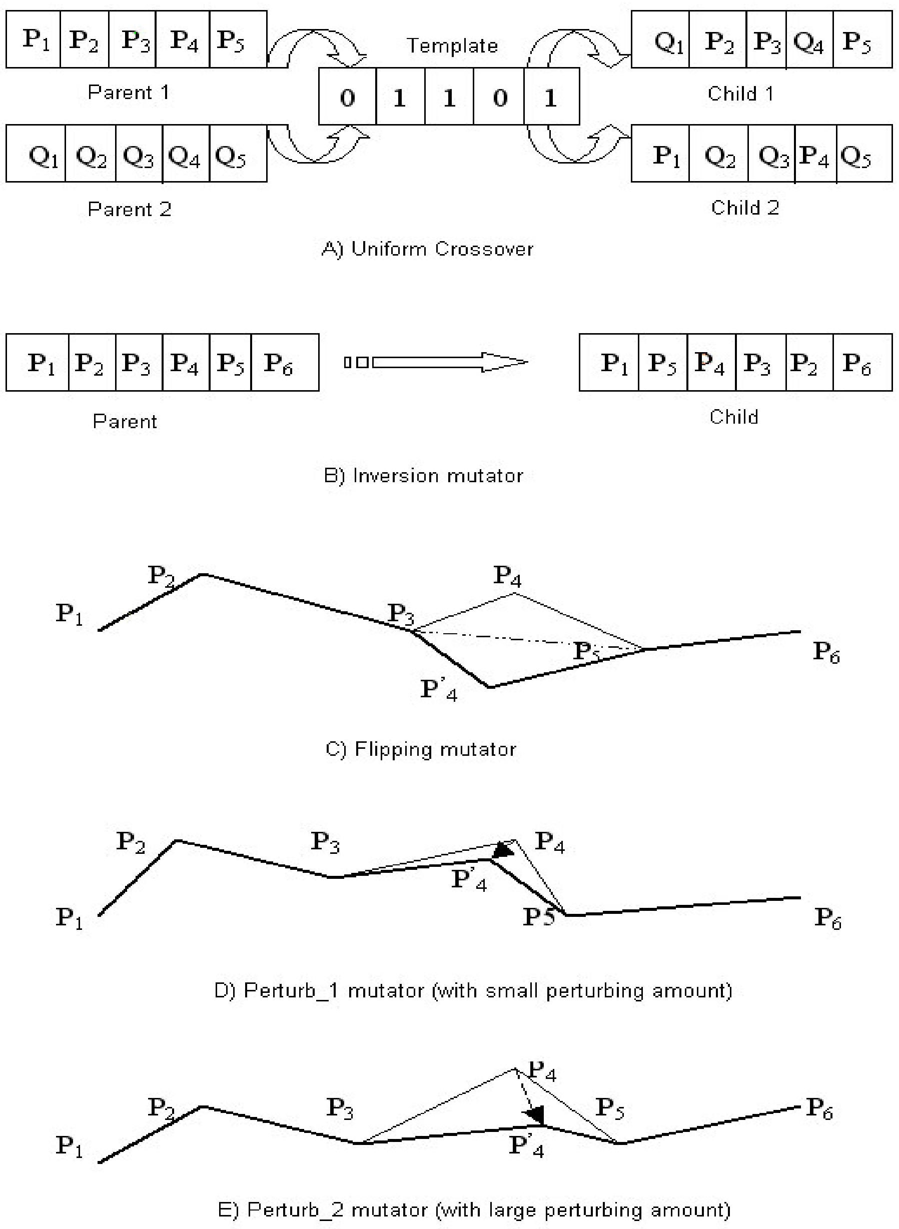

Fig. 6 Designed evolutionary operators.

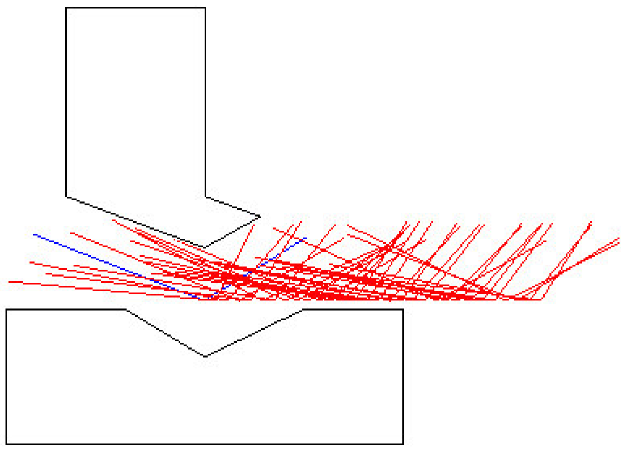

Fig.7 An obtained feasible motion path of the “V” shaped part.

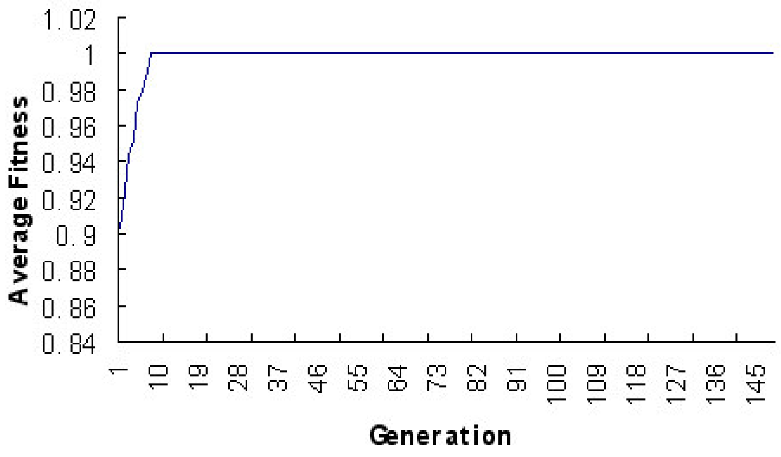

Fig.8 The number of generations vs. the average fitness value for Case 1.

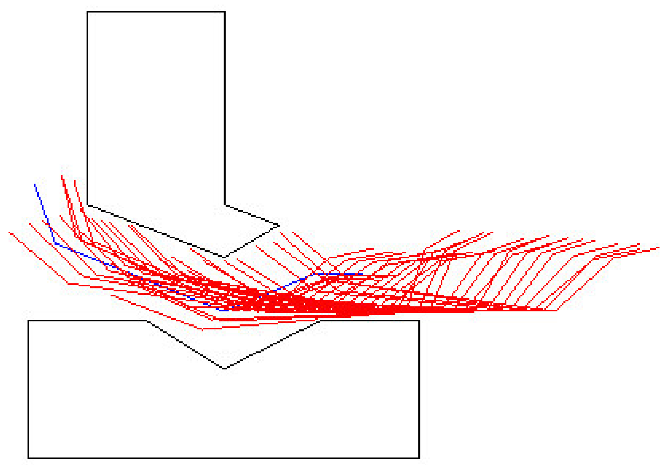

Fig.9 An obtained motion path of a multiple-bent sheet metal part.

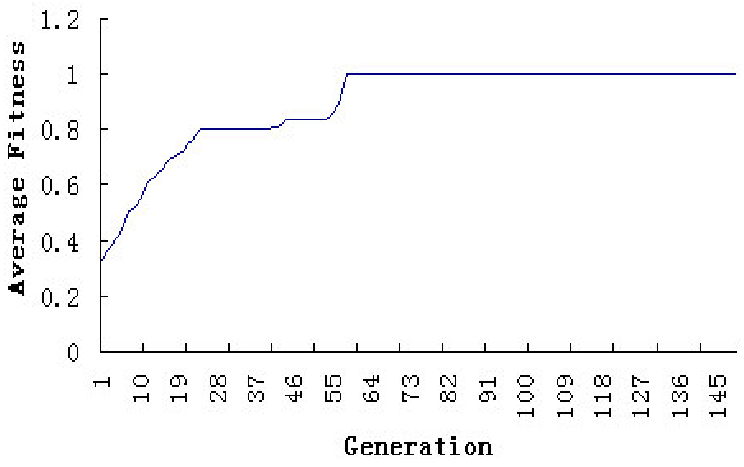

Fig.10 The number of generations vs. the average fitness value for Case 2.