공 학 석 사 학위논문

모형선 제작을 위한 최적 네스팅 및 절단 경로 생성

Optimal Nesting and Generation of Cutting Paths for Model Ship Production

울 산 대 학 교 대 학 원
조 선 및 해 양 공 학 과
최 준 호

모형선 제작을 위한 최적 네스팅 및 절단 경로 생성

지 도 교 수 오 민 재

이 논문을 공학석사학위 논문으로 제출함

2023 년 08 월

울 산 대 학 교 대 학 원
조 선 및 해 양 공 학 과
최 준 호

최준호의 공학석사학위 논문을 인준함

심사위원 신 동 목 인

심사위원 오 민 재 인

심사위원 김 기 수 인

울 산 대 학 교 대 학 원

2023 년 08 월

모형선 제작을 위한 최적 네스팅 및 절단 경로 생성

최 준호
울산대학교 대학원 조선 및 해양공학과

국문 요약

2차원 네스팅(Nesting)은 조선, 자동차, 의류산업 등에서 자주 사용되는 방법으로, 배치를 효과적으로 함으로써 자원과 재료의 낭비를 최소화 할 수 있으며, 배치 자동화를 통해 공수를 절감하고 생산성을 높일 수 있다. 현재 모형선 제작 과정에서는 피스 배치를 수작업으로 진행하고 있기 때문에 작업 시간이 오래 걸리며 재료의 낭비를 최소로 하는 최적 배치를 찾기 어려운 점이 있다. 이러한 문제를 해결하기 위하여 본 연구에서는 모형선 제작에 필요한 자재의 남김율을 최소화 하는 최적 배치 자동화 방법을 제안한다. 본 연구의 목표는 원자재 목판 안에 여러 개의 피스 형상들이 중첩하지 않고 자재의 남김율을 최소화하며 최대의 피스 수로 최적의 배치를 하는 것이다. 모형선 피스의 특성을 고려하여 다중 판 배치에 대한 네스팅 알고리즘을 구현하였다. 피스를 원자재 판 위에 최적으로 배치하는 방법을 다음과 같이 제시한다. 형상 정보를 표현하는 방법에는 격자 표현법, Non-Fit-Polygon, Phi-function 등의 방법이 있다. 본 연구에서는 격자 표현법을 선택하였다. 격자 표현 방법의 경우 형상을 격자 형태로 간단히 표현할 수 있으며, 다른 방법들에 비해 형상 간 중첩 여부 판단이 쉽다. 그리고 음함수의 특성을 활용해 격자에 피스의 외형선을 표현하는 방법을 제시하였다. 형상 내부의 경우 다차원 배열에서 각 격자를 방문하는 알고리즘인 Breadth-first Search 기법을 사용하여 형상의 내부와 외부를 정의하였다. 형상 회전의 경우 전역 좌표계를 기준으로 회전시켰으며 좌표 값이 음수로 표현되었을 때 그 값 만큼 x축 또는 y축으로 평행이동을 시켜 좌표 값이 최소 좌표인 (0,0) 이상을 가지도록 하였다. 배치 알고리즘의 경우 두 가지 목적 함수(f_y, f_{xy})를 구성하여 배치 결과를 비교하였다. 배치 순서와 피스의 각도에 따라서 배치 결과가 달라질 수 있는데 유전 알고리즘을 사용해 자재의 버림율이 최소 값일 때의 배치 순서를 결정하였을 뿐만 아니라 해당 순서일 때 피스의 최적의 회전 각도도 도출하였다. 본 연구에서 제시하는 절단 경로 생성 알고리즘은 공구의 절단 경로를 최소화하기 위해서 원점에서 가장 가까운 위치의 피스부터 절단된 피스에서 가까운 피스를 선택하는 방식으로 진행된다. 폴리라인(Polyline)에 대한 절단 경로를 생성하기 위해 피스들이 지니고 있는 선분과 호의 시작점과 끝점을 이용하여 점 간 최소 이동거리를 계산하는 알고리즘도 적용하였다. 이는 메타 휴리스틱 알고리즘에 비하여 실행시간이 짧으며 합리적인 경로를 제시할 수 있다.

Optimal Nesting and Generation of Cutting Paths for Model Ship Production

Junho Choi
Department of Naval Architecture and Ocean Engineering, Ulsan University

Abstract

2D nesting, a widely employed technique in industries such as shipbuilding, automotive, and apparel, aims to minimize resource and material waste by effectively arranging components. By automating the layout process, it reduces labor and increases productivity. However, the current manual approach to piece placement in model ship production results in lengthy work hours and difficulty in finding the optimal placement with minimal material waste. To address this challenge, this study proposes an automated method for optimizing nesting in model ship production to minimize material scrap. The objective is to achieve an optimal arrangement within the raw material sheet, maximizing the number of pieces while avoiding overlapping shapes. A nesting algorithm for multiple sheet layouts, considering the characteristics of model ship pieces, is implemented. The chosen representation method is grid-based, as it provides a simple depiction of shapes and facilitates overlap detection compared to alternative methods. Additionally, a novel approach utilizing the properties of negative functions is presented to represent piece outlines on the grid. The interior of shapes is defined using the Breadth-first Search algorithm on a multidimensional array. Rotation of shapes is performed based on a global coordinate system, with adjustments made to ensure positive or zero coordinate values. The placement algorithm utilizes two objective functions to compare results, employing genetic algorithms to determine the optimal placement order that minimizes scrap rate and identify the optimal rotation angle for each piece in that order. Furthermore, a cutting path generation algorithm is introduced to minimize tool cutting paths, employing a selection process based on proximity to the origin and previously cut pieces. The algorithm calculates the minimum travel distance between points using line segments and arcs associated with each piece. The proposed method offers shorter execution times and provides reasonable cutting paths compared to metaheuristic algorithms.

목차

표 목차

그림 목차

제1장 서론

1.1. 연구배경

제조 산업에서 원자재 가격이 연간 꾸준히 상승하고 있으며 생산 비용의 큰 부분을 차지한다. 조선, 항공우주, 의류 및 자동차 등 제조업계는 원자재 판에서 피스들을 잘라내기 위한 가장 효율적인 배치를 결정하는 네스팅 알고리즘의 개발과 맞춤화에 많은 노력을 기울이고 있다.[1]

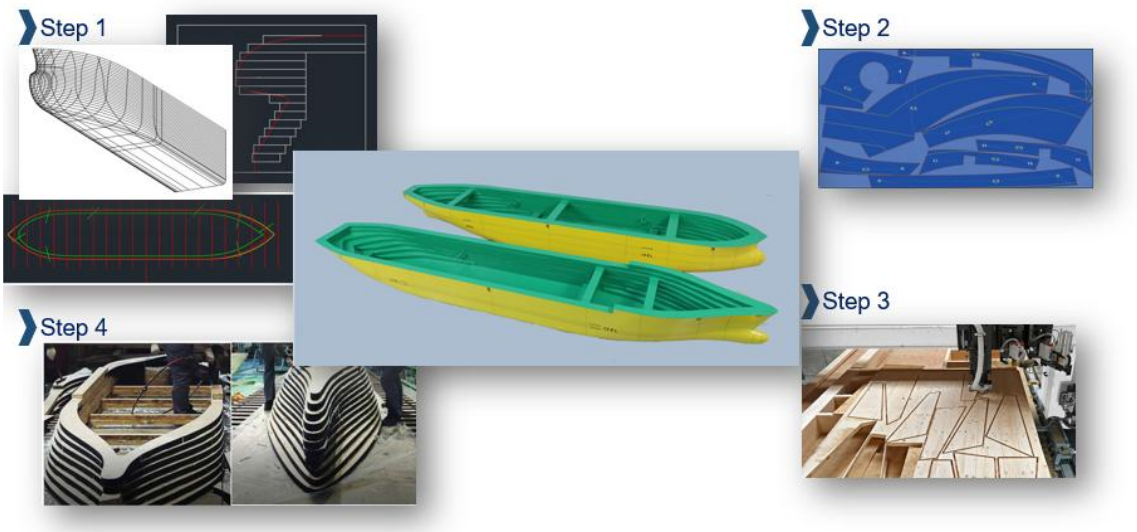

Fig. 1.1. Model ship manufacturing process

모형선 제작 과정은 Fig. 1.1.과 같다. CAD프로그램을 사용하여 수작업으로 모형선 선도를 waterline과 section에 따라 잘라 모형선 피스를 만든다. 그리고 모형선 피스들을 가로 2400mm, 세로 1200mm의 원자재 목판에 원자재의 버림율(Scrap ratio)이 적도록 배치시킨다. 이 후 실제 피스들을 제작하기 위해 완성된 배치에 대한 NC (Numerical Control) 코드를 출력한다. 마지막으로 모형선 피스들을 층별로 쌓아 모형선을 완성한다. 현재 이러한 과정들은 수작업으로 이루어지므로 인건비와 공수절감 및 생산성 향상을 위해 자동화 프로세스를 구축하는 것이 중요하다.

모형선 피스 제작용 네스팅의 경우 다른 산업과 달리 각기 다른 불규칙한 크기와 모양을 가진 수많은 피스들이 존재하며 일반적으로 그 피스들의 크기가 목판의 크기 대비 크기 때문에 단일 판에 모든 피스들의 배치가 불가능하다. 또한 프로그램 내에서 피스 생성 자동화와 컷패스, 비벨, 리드인/아웃 등의 정보를 포함하는 출력용 NC코드를 포함해야 한다는 점에서 보다 복잡하다. 모형선 피스를 제작하는 작업과 더불어 제조업계에서의 네스팅 과정의 주된 목표는 비용과 시간을 최소화하는 것이다. 이를 달성하기 위해서 고려해야하는 요소들에는 자재 버림율를 최소화하는 것, 원자재 판의 남은 부분을 재사용 가능하게 만드는 것과 컴퓨팅 및 커팅 시간을

감소시키는 것 등이 있다.

단일 판에 피스들의 배치를 진행하게 될 경우 배치되지 못한 피스들은 따로 네스팅 과정을 반복해야한다. 다중 판을 이용해 피스들의 배치를 진행할 경우 원하는 피스들을 모두 판 위에 배치시킬 수 있으며 단일 판 알고리즘을 여러 번 반복해 배치를 진행하지 않아도 되므로 시간적인 효용성 측면에서도 높을 것으로 기대된다. 뿐만 아니라 기존에 사용되던 NC 코드 생성 프로그램을 사용하지 않고 절단 경로 생성 알고리즘을 함으로써 피스 배치부터 절단 경로 생성까지 자동화를 통해 작업자의 숙련도를 필요로 하지 않으며 합리적인 결과를 얻을 수 있을 것이라고 기대된다.

1.2. 관련 연구

초기 네스팅 문제는 불규칙한 형상을 직사각형으로 근사시켜 배치하는 문제로 접근하여 해결되었다.[2] 그러나 직사각형 형상과 형상이 다른 불규칙한 형상의 경우 자재 버림율 측면에서 효율이 많이 떨어지게 된다. Dori, D. and Ben-Bassat[3]은 이러한 문제점들을 보완하기 위해 직사각형이 아닌 하나의 다각형 모양을 선택하여 규칙적으로 이어 붙이는 방법을 고안했다. 이 방법은 직사각형 형상으로 근사하여 배치하는 경우보다 자재 효용성 측면에서 높지만 다양하고 불규칙한 형상을 배치하는 알고리즘을 적용하기 어렵다. 이후 Adamowicz, M. and Albano, A.[4][5]에 의해 개발된 Non-Fit-Polygon(NFP)기법은 불규칙한 형상을 더 정확하게 배치하기 위해 볼록 다각형 형태로 근사시켜 피스들을 밀집하여 배치하는 방법이다. Babu A.R and Babu N.R[6]은 형상을 2차원 격자를 통해 표현하였고 배치 알고리즘을 통해 원자재 판에 배치시켰다. 이외에도 시간과 비용의 효율성을 개선하고 피스 형상을 더 정확하게 나타내기 위한 파이 함수[7], 직접 삼각함수[8], 폴리곤 클리핑 기법[9]등 다양한 수학적 모델을 사용한 형상 표현 기법이 있다.

피스 배치와 관련하여 Kang[10]은 격자 표현법을 사용하여 배치 알고리즘을 구성하였으며 판의 왼쪽 아래에 배치 되도록 하는 목적함수를 제안했다. Kim[11]은 격자 표현법을 사용하였으나 탐색 범위를 줄이기 위해 Sliding 기법을 사용했다. 뿐만 아니라, Weng and Kuo[12]는 Bottom-left filling 알고리즘을 사용하여 원자재 사용 길이를 줄이기 위한 알고리즘을 개발하였다. 또한 Genetic algorithm, Greedy algorithm, Tabu search 등의 휴리스틱 알고리즘[13] 또는 메타 휴리스틱 알고리즘

[14][15], 전문가 시스템의 사용[16] 등의 방법과 같이 피스들의 효과적인 배치 방법에 대해 고려되어져 왔다.
배치가 완료된 모형선 피스들의 절단 경로를 생성하는 것은 배치 최적화 과정을 거치는 것과 마찬가지로 가격 경쟁력 및 생산성 향상에 매우 큰 영향을 미친다.[17] 절단 경로 생성 알고리즘의 경우 일반적으로 절단 경로를 최소화 하는 것에 목적이 있다. 하지만 이러한 문제의 경우 NP-hard(Non-deterministic Polynomial-time hard)문제이기 때문에 현실적으로 최적 해를 구하는 것은 불가능하다.[9] Park[18]과 Han[9]의 연구에서는 각 피스들이 지니고 있는 정점(Piercing Point)을 이용하여 유전 알고리즘을 통한 경로 최적화 문제를 해결하였다. Al-Sahib[19]은 절단 경로 문제를 외판원 문제로 모델링한 후 유전 알고리즘을 사용하여 최소 경로를 찾는 알고리즘을 제시했다. Lee[20]는 피스 정점 최적화를 위한 전역 검색과 부품 시퀀싱을 위한 로컬 검색을 결합한 2단계 유전자 알고리즘을 제안하였다. Hajad[21]는 이미지 파일에서 피스 윤곽의 좌표를 할당하였으며 이를 이용해 제안하는 Large Neighborhood Search와 결합된 모의 어닐링(Simulated Annealing) 알고리즘을 사용하여 최적의 절단 경로를 계산하였다. Bang[22]는 절단 경로 생성 휴리스틱 알고리즘을 개발하였다. Manber and Israni[23]은 홀을 포함하지 않는 피스의 최소 정점을 갖는 절단 경로 생성 알고리즘을 개발하였다. Jo[24], Han[25] 는 전역 최적화 알고리즘인 담금질 기법을 적용하여 절단 경로 최적화 문제를 해결하였다.

1.3. 연구 목적

본 연구는 모형선 피스의 자재 버림율을 최소로 하는 네스팅 알고리즘을 개발하는 것을 목적으로 한다. 모형선 피스의 형상 표현은 격자표현법을 이용한다. 세부적으로 Implicit function의 특성을 이용하여 선과 호로 이루어진 모형선 피스의 외형선을 격자에 '1'의 값을 할당하여 표현한다. 그리고 Breath-first Search 알고리즘[26]을 이용하여 이미 외형선이 정의된 격자의 내부를 '1'로 정의한다. 형상 배치 알고리즘의 경우 원자재 판의 왼쪽 아래에 배치되도록 하는 것을 주목표로 하며 Kang[10]이 제시한 목적함수의 일부만 사용하여 배치 후 남는 재료의 효용성을 높일 수 있도록 하였다. 뿐만 아니라, 사이즈가 큰 모형선 피스의 경우 원자재 목판 사이즈와 크기가 비슷하다. 이러한 경우에 단일 판에 피스를 배치하게 되면 해당 피스만 배치되고 다른 피스들이 배치되지 못하는 경우가 발생한다. 문제점을 해결하기 위해 다중 판에 피스들이 배치 될 수 있도록 알고리즘과 목적함수를 재구성하였다. 일반적으로 넓이, 종횡비에 따라서 배치를 진행한다, 또한, 배치 순서와 각도를 유전자로 설정하여 최적의 순서와 각도를 결정하였다. 본 연구에서는 기존의 모형선 제작과정에서 수작업으로 이루어지던 네스팅 작업을 제시한 알고리즘을 통해 자동화시키는 것을 최종 목적으로 한다. 또한, 다중 판 배치를 위한 네스팅 알고리즘을 통해서 여러 개의 피스들을 다중 판에 배치시킴으로써 이전의 배치 알고리즘들 보다 효과적으로 피스들을 배치할 수 있다. 이후 배치된 결과에 대한 절단 경로

를 생성하였다. 본 연구에서 제시하는 절단 경로 생성 알고리즘은 공구의 절단 경로를 최소화하기 위해서 원점에서 가장 가까운 위치의 피스부터 절단된 피스에서 가까운 피스를 선택하는 방식으로 진행된다. 폴리곤(Polygon)이 아닌 호와 선분을 포함하는 폴리라인(Polyline)에 대한 절단 경로를 생성하기 위해 각 피스들이 지니고 있는 선분과 호의 시작점과 끝점을 이용하여 점 간 최소 이동거리를 계산하는 알고리즘도 적용하였다. 이는 메타 휴리스틱 알고리즘에 비하여 실행시간이 상당히 짧으며 합리적인 경로를 제시할 수 있다.

1.4. 논문의 구성

본 논문은 다음과 같이 구성되어 있다. 2장에서는 형상의 표현 알고리즘, 3장에서는 배치알고리즘, 4장에서는 절단 경로 생성 알고리즘 그리고 5장에서는 연구 수행을 통한 결론 및 고찰을 정리하였다.

제2장 형상의 표현

2.1. 배경 이론

네스팅 문제에서 다양한 모양과 크기의 피스 형상을 표현하기 위해 다양한 방법론을 사용한 연구들이 진행되어왔다. No-Fit-Polygon(NFP)는 한 도형을 다른 도형의 경계를 따라 추적하여 얻어지는 경계선의 합으로 이루어진 다각형이다. Fig. 2.1.의 왼쪽 그림과 같이 다각형 A, B가 볼록한 다각형으로 이루어져 있어 NFP를 쉽게 계산할 수 있지만, 오른쪽 그림과 같이 A가 오목한 다각형으로 이루어져 있거나 형상 내에 구멍(Hole)이 존재하는 경우 NFP 알고리즘이 복잡해진다.

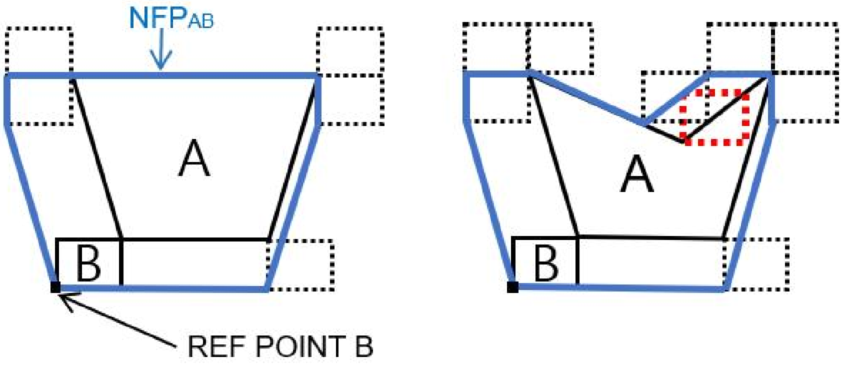

Fig. 2.1. No-Fit-Polygon method

격자 표현법은 Fig. 2.2.와 같이 형상을 격자에 표현하는 방법이다. NFP 방법과는 달리 오목하거나 구멍이 있는 형상도 간단하게 표현이 가능하다. 하지만 호 또는 기울기가 있는 직선을 표현하는 경우 격자의 단위에 따라서 형상 정확도가 달라지기 때문에 격자의 단위를 적절하게 조정해야 한다. 격자의 크기가 작아지게 될 경우 실행 시간과 메모리 사용량이 증가하고 크기가 커지면 정확한 형상표현이 어렵다.

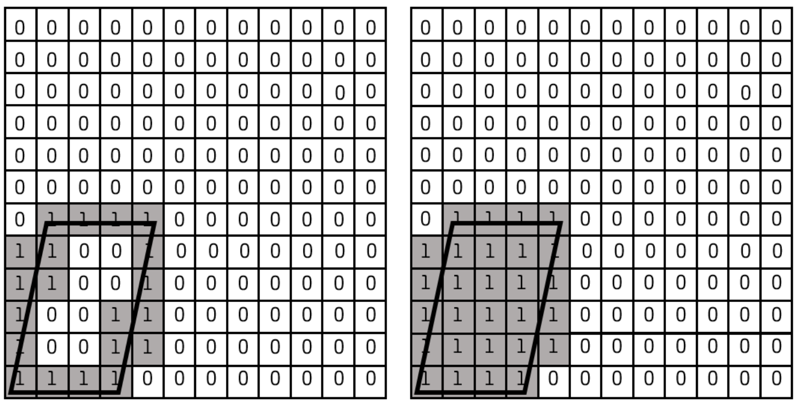

Fig. 2.2. Grid representation method

NFP는 다각형 간의 교차 검사를 수행하는데 탁월한 도구이나, 2차원 배치 문제에

서는 실제 제조 산업에서 널리 적용되지 않았다.[27] 이는 NFP는 구현이 복잡하고 사용 가능한 견고한 알고리즘이 부족하기 때문이다.[27] 모형선 피스의 형상을 표현하기 위해서는 호와 같은 곡선의 형상을 정확히 표현해야한다. NFP 방법의 경우 곡선을 선분으로 근사시켜 표현하므로 오차가 발생 할 수 밖에 없다. 따라서 본 연구에서는 implicit function의 이론을 적용하여 격자 표현법을 이용해 형상을 표현하였다.

모형선 피스 배치 및 절단 경로 생성 자동화 알고리즘의 순서도는 Fig. 2.3.과 같다. 먼저 피스의 형상 정보 데이터를 텍스트 형태로 입력 받은 후 격자에 형상을 표현한다. 이후 다중 판 배치 알고리즘을 통해 피스 배치를 진행하며 유전 알고리즘을 통해서 배치 순서와 각도를 결정한다. 최종적으로 자재 남김율이 가장 낮은 배치 결과를 도출할 수 있다. 배치 결과를 이용하여 G-code 기반으로 한 절단 경로를 생성하게 되면 전체 프로세스가 완료된다.

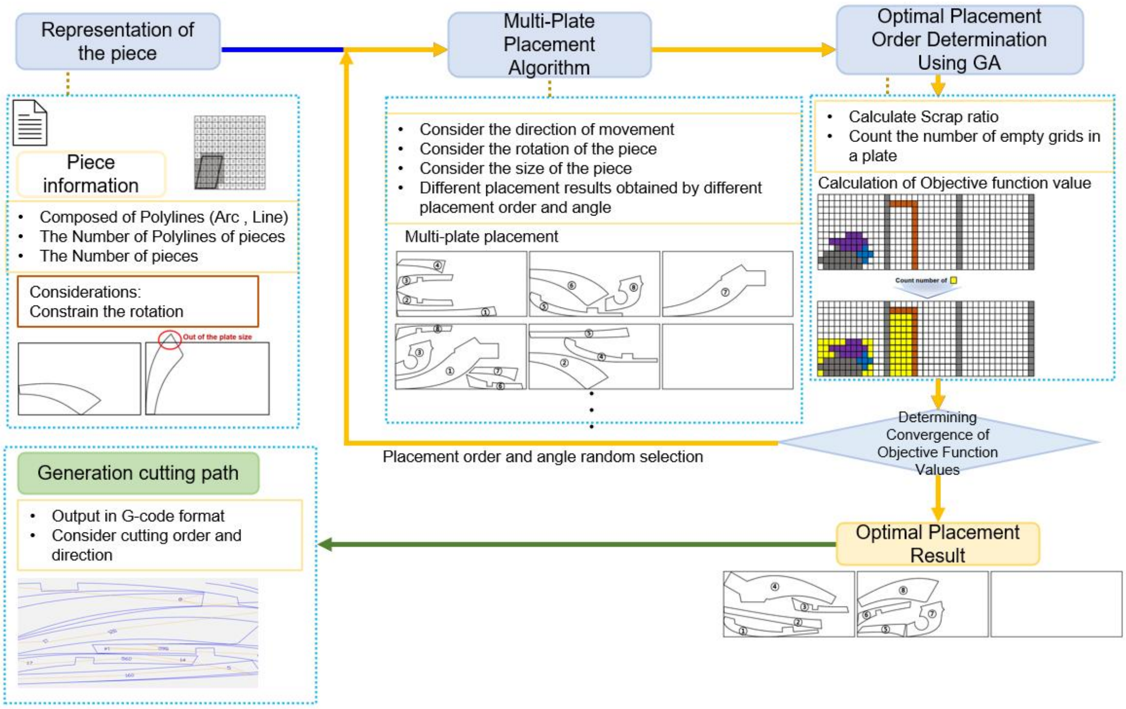

Fig. 2.3. Algorithm flow chart for model ship piece placement and cutting path generation

2.2. 격자 데이터 생성 과정

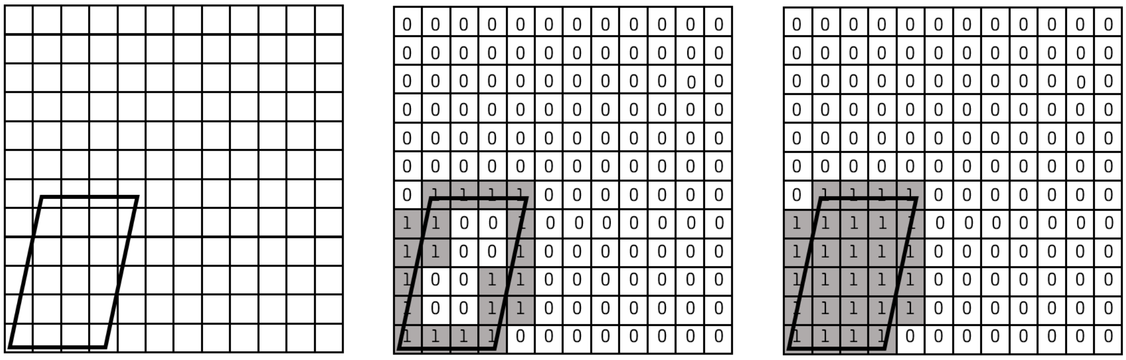

Fig. 2.4. The process of defining the shape of a piece

격자 데이터를 생성과정은 Fig. 2.4.과 같다. 격자는 너비와 높이가 동일한 크기의 2차원 행렬로 구성되며 '0'의 값으로 초기화 되어있다. 모형선 피스의 복잡한 윤곽선이나 내부 구멍들과 같은 외형선 형상이 포함되는 격자에 '1'의 값을 할당한다. 피스를 완벽하게 정의하기 위해서는 내부의 값들도 외형선 픽셀을 이루는 값인 '1'의 값으로 할당해야 한다. 이를 위해 내부 정의 알고리즘을 통해 피스 내부를 '1'로 할당하였다.

2.3. 해상도의 정의

격자 표현 방법에서 해상도를 설정하는 문제는 연산 시간과 배치 정확도를 결정하는 가장 중요한 변수 중 하나이다. 더 높은 해상도를 설정하게 될 경우 배치 정확도가 향상되지만, 더 많은 연산 시간이 필요로 하게 되며 그 반대 또한 마찬가지이다. 따라서 연산 시간과 배치 정확도 사이에서 우선순위를 결정해야한다. Fig. 2.5. 에서 동일한 알고리즘과 4가지의 피스를 이용해 배치를 수행했지만 그림의 왼쪽 그림의 경우 24 by 24로 총 격자수는 576개이고, 그림의 오른쪽의 경우 48 by 48로 2304개이다. 즉, 가로, 세로 방향 격자의 개수 곱의 차이만큼 나기 때문에 더 많은 격자 수를 선택하게 되면 연산 속도 측면에서 기하급수적으로 차이가 나게 된다. 하지만, 오른쪽 판의 경우 크기가 작은 모형선 피스가 크기가 큰 두 개의 피스 사이에 배치되어 왼쪽 판보다 더 효율적인 배치를 하게 된다. 본 연구에서는 연산 시간과 배치 정확도 측면을 실험적으로 고려하여 격자 사이즈를 지정하였다.

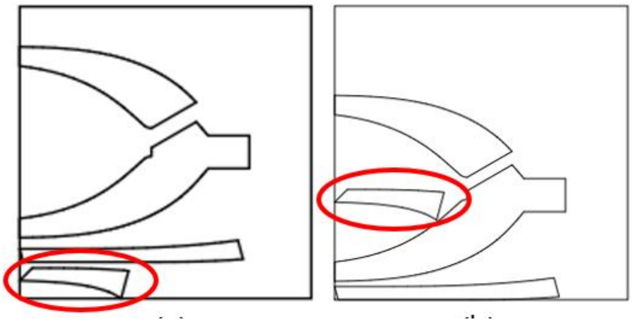

Fig. 2.5. Placement result due to resolution difference

2.4. 입력 데이터

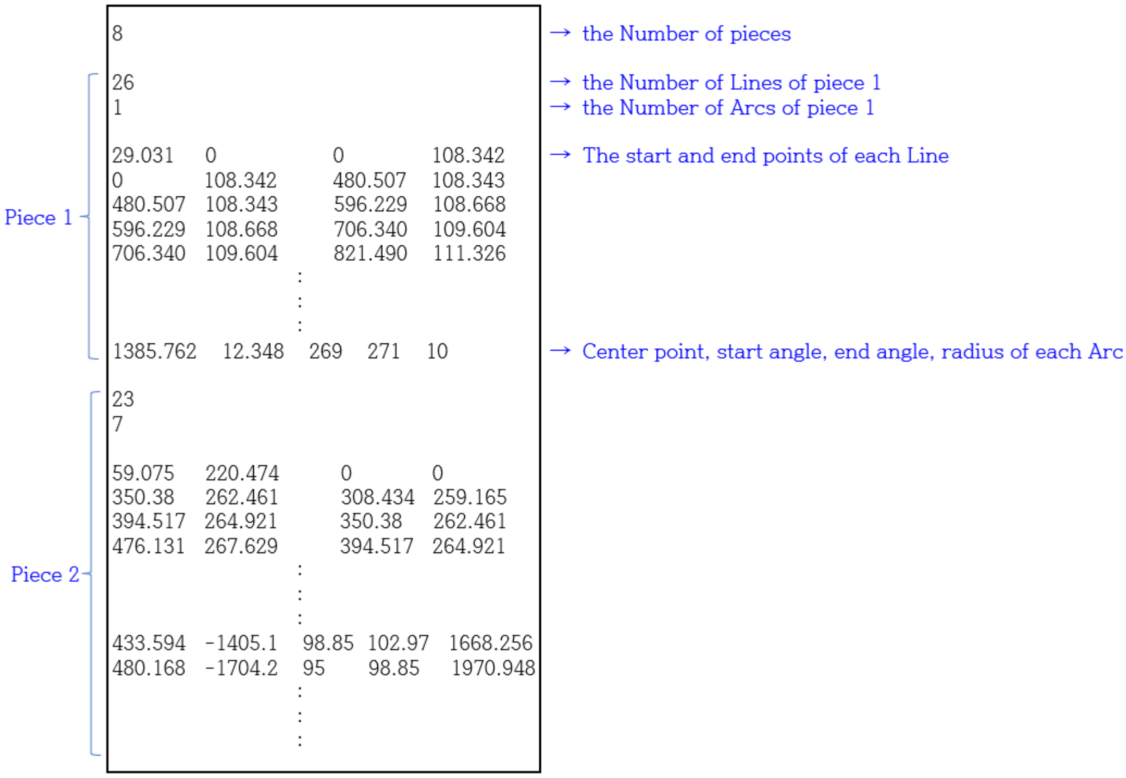

Fig. 2.6. Data of the piece shape

피스를 정의하는 데이터는 CAD 데이터로부터 추출되었으며 Fig. 2.6.과 같이 텍스트 정보로 구성된다. 모형선 피스의 개수와 각 피스의 정보들 순으로 나열된다. 피스의 정보는 피스에 존재하는 선분(line)의 수, 호(arc)의 수, 선분의 시작 점(start point)과 끝점(end point), 그리고 호의 중심점(center point), 시작 각도(start angle), 끝 각도(end angle), 반지름(radius) 순으로 나열된다.

2.5. 피스의 외형선 정의

모형선 피스는 여러 개의 선분과 호로 이루어져 있다. 각각의 선분과 호를 격자에 표현한 후 모든 격자를 합하여 하나의 격자로 표현하였다. 외형선 정의를 위한 이론은 2.5.1절에서 설명하고 선분과 호에 대해서 적용한 내용은 2.5.1.1, 2.5.1.2절에서 자세히 설명한다.

2.5.1. Implicit functions

피스를 표현하는 구성 요소인 선분과 호는 변수 x, y 그리고 어떤 상수로 이루어진 방정식 또는 함수 형태로 표현할 수 있다.

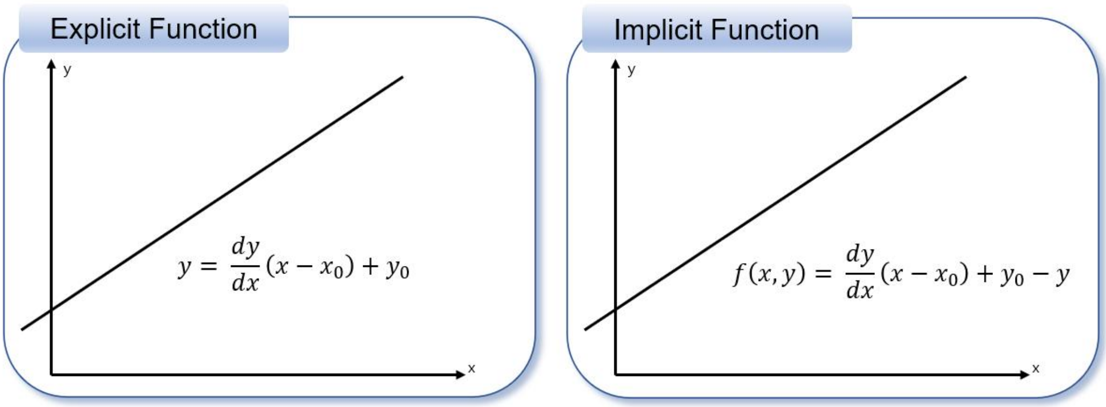

Fig. 2.7. Representation of Explicit functions and Implicit functions

일반적인 형태의 방정식은 Fig. 2.7.의 왼쪽과 같이 표현되며 Explicit function이라고 불린다. Explicit function은 독립 변수 x와 x의 값에 따라 변하는 종속변수 y로 표현되며 $y=f(x)$의 형태로 표현된 함수이다. 반면에, Implicit function은 변수 x,y의 관계를 나타내는 함수로 Explicit function와는 달리 변수들 사이의 관계식으로 주어진다. $f(x,y)$형태로 표현되며 Explicit function에서 종속변수 y항을 우변으로 이항한 형태로 표현할 수 있다.

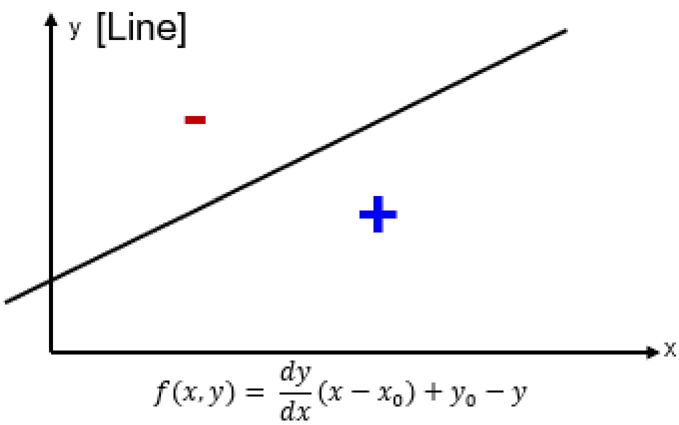

Fig. 2.8. Expression of the implicit function for the equation of a straight line

Table. 2.1. Determining the sign of the equation of a straight line

position of (x_1,y_1)	sign
upper side of the line	$f(x_1,y_1)<0$
down side of the line	$f(x_1,y_1)>0$
on the line	$f(x_1,y_1)=0$

Implicit function을 활용하게 되면 Fig. 2.8.와 Table. 2.1.과 같이 직선의 방정식이 주어졌을 때, 특정 점 (x_1,y_1)를 Implicit function에 대입하면 출력 값이 음수, 양수 또는 '0'으로 표현된다. 해당하는 점 (x_1,y_1)는 출력 값이 음수인 경우 직선의 위쪽, 양수인 경우 직선의 아래쪽, 그리고 0인 경우 직선 위에 위치한다. 이러한 특징을 이용하여 격자에 존재하는 모서리 좌표들을 방정식에 대입하여 부호를 판별한다. 각 모서리 값은 음수 값인 경우 '-1', 양수 값인 경우 '1', '0'인 경우는 그대로 표현한다. 이후에, 한 개의 격자의 4개의 모서리 값을 더하여 조건에 따라 '0'또는 '1'로 할당하여 피스의 외형 선에 해당하는 격자를 '1'로 표현할 수 있다. 호의 경우에도 Fig. 2.9.과 Table. 2.2.에 따라 Implicit function형태로 표현할 수 있다. 시작 각도 θ_1과 끝 각도 θ_1의 범위를 나누어서 모든 호에 대해서 표현이 가능하도록 하였다.

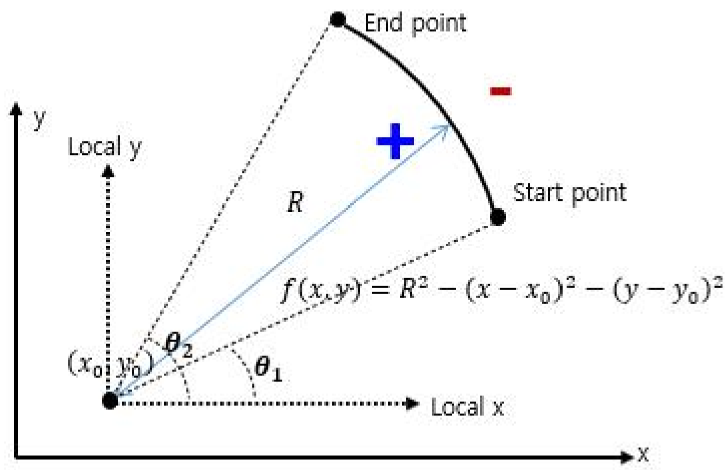

Fig. 2.9. Expression of the implicit function for the equation of an arc

Table. 2.2. Determining the sign of the equation of an arc

position of (x_1,y_1)	sign
outside of the arc	$f(x_1,y_1)<0$
inside of the arc	$f(x_1,y_1)>0$
on the arc	$f(x_1,y_1)=0$

2.5.1.1. Line

선분의 경우 직선의 방정식의 형태로 표현되며 시작점과 끝점이 주어진다. 직선을 표현할 때 사용되는 Bresenham's algorithm[28]의 경우에는 수평선, 수직선, 기울기가 1인 선일 때 표현하기 까다롭기 때문에 따로 예외처리를 해줘야한다. 반면에, implicit function를 이용한 표현 방법을 사용하게 되면 모든 경우의 선분에 대해서 표현이 가능하다.

예를 들어, 시작점이 $(0.5,0.5)$이고 끝점이 $(3.2,3.5)$ 인 선분이 있다. 두 점을 이용해 explicit function 형태로 표현 할 수 있으며 식(2.1)과 같다.

$$y=\frac{10}{9}x-\frac{1}{18} \tag{2.1}$$

implicit function의 형태로 표현하면 식(2.2)과 같다.

$$f(x,y)=\frac{10}{9}x-\frac{1}{18}-y \tag{2.2}$$

가로 칸과 세로 칸의 수가 6으로 동일한 임의의 격자를 생성한다. 이때 앞서 말한 각 격자의 모서리에 대한 배열도 생성해야하므로 모서리의 배열의 경우 임의의 격자보다 1칸씩 더 많으므로 가로 칸과 세로 칸의 수가 7인 배열이 생성된다. (Fig. 2.10.)

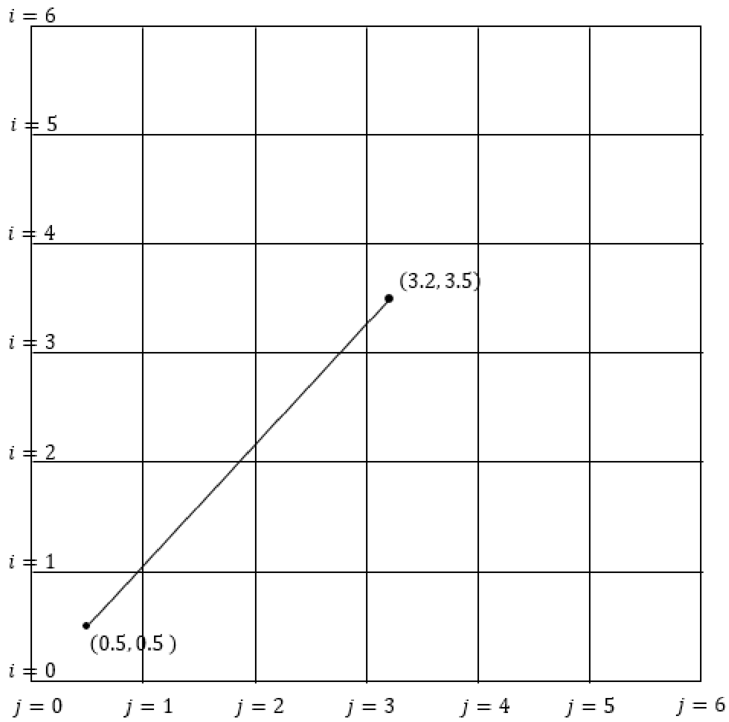

Fig. 2.10. Creation of a grid for representing line segments

식(2.3)의 조건에 따라서 각 모서리 점(a,b)의 값이 '0'보다 크다면 '1'로, '0'보다 작다면 '−1'로, 그리고 '0'과 같다면 '0'으로 값을 할당한다. Fig. 2.11.에서 파란색 점은 '−1'을 뜻하고 빨간색 점은 '1'을 뜻한다.

$$\begin{aligned} &f(a,b) > 0 \ : 1 \\ &f(a,b) < 0 \ : -1 \\ &f(a,b) = 0 \ : 0 \end{aligned} \tag{2.3}$$

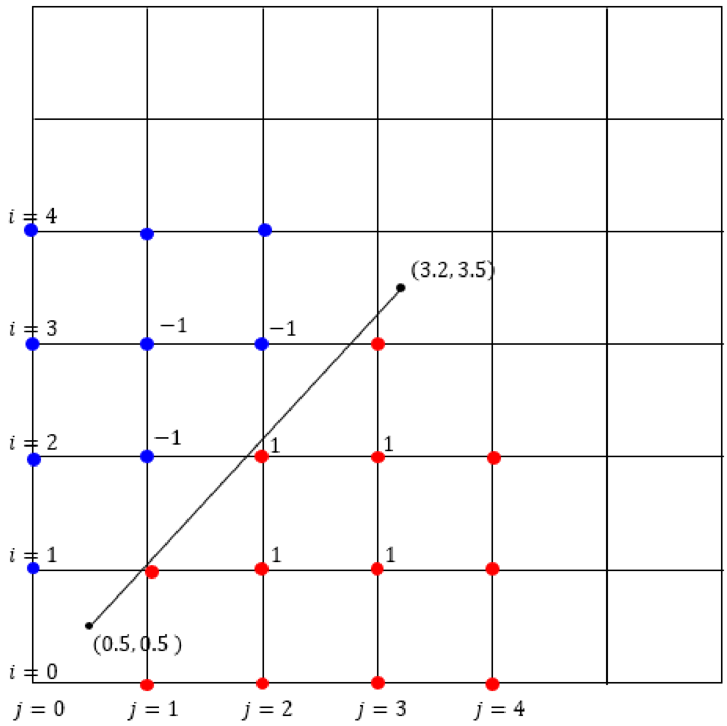

Fig. 2. 11. Assigned values for lattice vertices

각 모서리 점의 격자에 값이 할당되면, 피스의 외형 선을 나타내는 격자를 표현할 수 있다. 격자의 값은 격자 주변의 네 개의 모서리와의 관계를 나타내는 식(2.4)를 통해 정의 할 수 있다. 즉, $grid[i,j]$ 격자의 모서리 점 4개는 $corner[i,j]$, $corner[i,j+1]$, $corner[i+1,j]$ 그리고 $corner[i+1,j+1]$이다.

$$-2 \leq corner[n,n]+corner[n,n+1]+corner[n+1,n]+corner[n+1,n+1] \leq 2 \quad (2.4)$$

if satisfied, $grid[n,n]=1$

if not, $grid[n,n]=0$

격자의 값에 '1'이 할당되는 경우와 '0'이 할당되는 경우에 대해서 Fig. 2.12.와 같다.

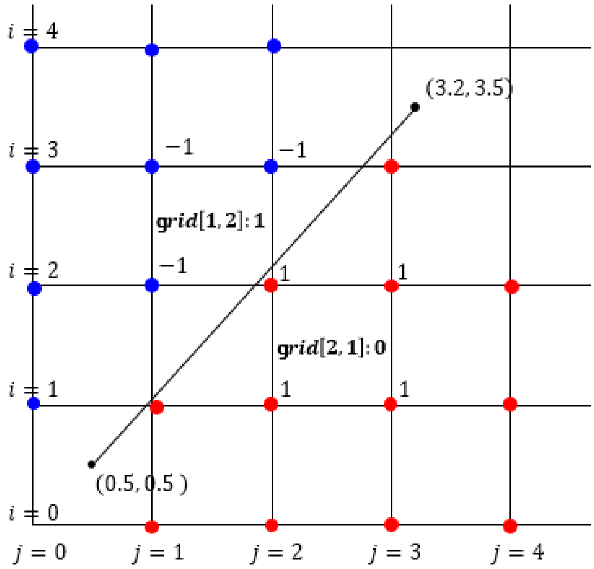

Fig. 2.12. Assigned values for the grid

$i=1, j=2$ 의 값을 가질 때 격자 주변의 네 모서리 값을 합하면 '−2'이므로 위의 조건을 만족한다. 따라서 격자에 '1'로 값이 할당되며 주어진 선분이 지나는 영역이다.

$i=2, j=1$ 의 값을 가질 때 격자 주변의 네 모서리 값을 합하면 '4'이므로 위의 조건을 만족하지 못한다. 따라서 격자에 '0'의 값이 할당되며 주어진 선분이 지나지 않는 영역이다.

마지막으로 지금까지 직선의 방정식을 활용하여 계산했다. 선분의 시작점과 끝점에 대한 계산 범위를 지정해주지 않는다면 격자의 모든 범위에 대해 계산이 수행되므로 원하지 않는 부분까지 '1'의 값이 할당될 수 있다. 따라서 계산범위는 식(2.5)와 식(2.6)에 따라 수행하였다.

$$(\text{int})y_0 \le i < (\text{int})y_1 \ ,(y_0 < y_1) \tag{2.5}$$

$$(\text{int})x_0 \le j < (\text{int})x_1 \ ,(x_0 < x_1) \tag{2.6}$$

격자는 '0'과 자연수 내에서 존재하므로 정수 형태로 변환하였다. 결과적으로 해당 계산범위를 격자에 적용한 결과 Fig. 2.13.과 같이 초록색 박스 내부에서만 계산이 수행된다.

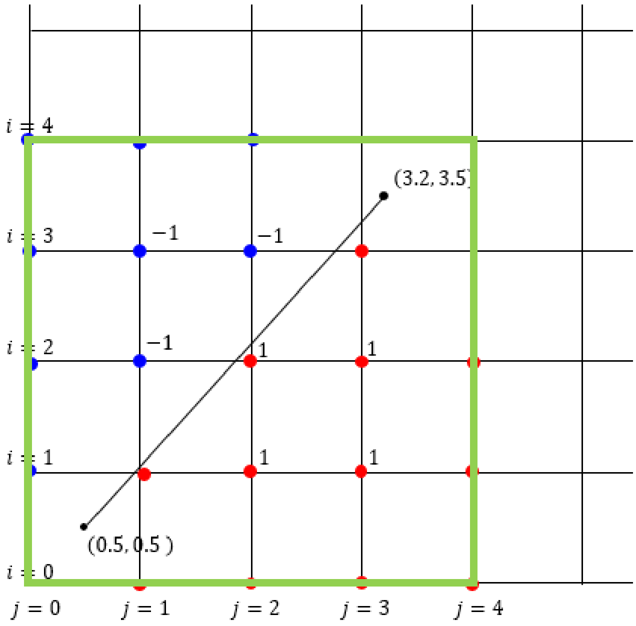

Fig. 2.13. Calculation range of the grid

계산 수행이 완료된 후 격자 표현 결과는 Fig. 2.14.과 같다.

0	0	0	0	0	0
0	0	0	0	0	0
0	0	1	1	0	0
0	1	1	0	0	0
1	1	0	0	0	0
1	0	0	0	0	0

Fig. 2.14. Calculation result of the line segment

2.5.1.2. Arc

호는 원의 방정식 형태로 나타낼 수 있으며 계산 범위에 따라서 호를 표현할 수 있다. 다음은 호의 중심이 (x_0, y_0)이고 반지름이 R인 원을 가로 방향으로 6칸, 세로 방향으로 6칸인 격자에 대해서 표현하는 예제이다. 선분을 표현한 것과 마찬가지로 모서리 좌표에 대한 배열의 수는 격자의 수보다 가로, 세로 방향으로 1개씩 더 많다.

Table. 2.3. Range of θ_1 and θ_2 to represent an arc

θ_1	θ_2
$0^\circ \le R < 90^\circ$	$0^\circ < R \le 90^\circ$
	$90^\circ < R \le 180^\circ$
	$180^\circ < R \le 270^\circ$
	$270^\circ < R \le 360^\circ$
$90^\circ \le R < 180^\circ$	$90^\circ < R \le 180^\circ$
	$180^\circ < R \le 270^\circ$
	$270^\circ < R \le 360^\circ$
	$360^\circ < R \le 450^\circ$
$180^\circ \le R < 270^\circ$	$180^\circ < R \le 270^\circ$
	$270^\circ < R \le 360^\circ$
	$360^\circ < R \le 450^\circ$
	$450^\circ < R \le 540^\circ$
$270^\circ \le R < 360^\circ$	$270^\circ < R \le 360^\circ$
	$360^\circ < R \le 450^\circ$
	$450^\circ < R \le 540^\circ$
	$540^\circ < R \le 630^\circ$

Table. 2.3.은 호의 격자 계산을 명확하게 수행하기 위해서 θ_1과 θ_2의 사분면의 위치에 따라 나눈 표이다. 16가지 경우로 설정하여 모든 호에 대해서 표현이 가능하도록 하였다. 호의 경우 θ_1과 θ_2가 어떤 사분면에 위치하느냐에 따라서 증가 및 감소 방향이 다르기 때문에 4개의 사분면에 대해서 구간을 나누어 계산을 수행하였다. 또한, 호의 경우에도 마찬가지로 격자는 0과 자연수 내에서 존재하므로 정수 형태로 변환하였다. 계산 범위에 대한 예를 Fig. 2.15.을 통해 알 수 있다.

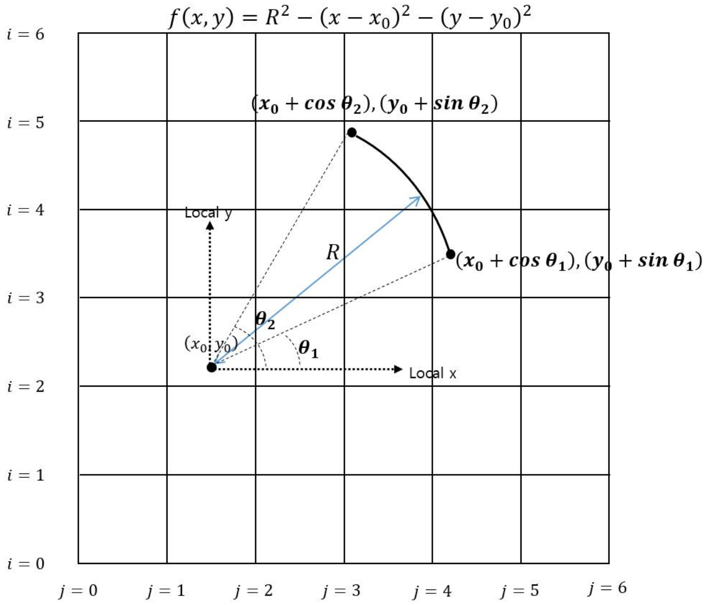

Fig. 2.15. θ_1 and θ_2 in the first quadrant

θ_1과 θ_2가 1사분면에 존재하므로 계산 범위는 식(2.7), 식(2.8)과 같다.

$$(\text{int})(y_0 + R \bullet \sin\theta_1) \le i < (\text{int})(y_0 + R \bullet \sin\theta_2) \quad (2.7)$$

$$(\text{int})(x_0 + R \bullet \cos\theta_2) \le j < (\text{int})(x_0 + R \bullet \cos\theta_1) \quad (2.8)$$

Fig. 2.16.과 같이 한 개의 초록색 박스 내에서만 계산이 수행된다. 따라서 외형선이 지나는 격자에서 '1'의 값이 할당된다.

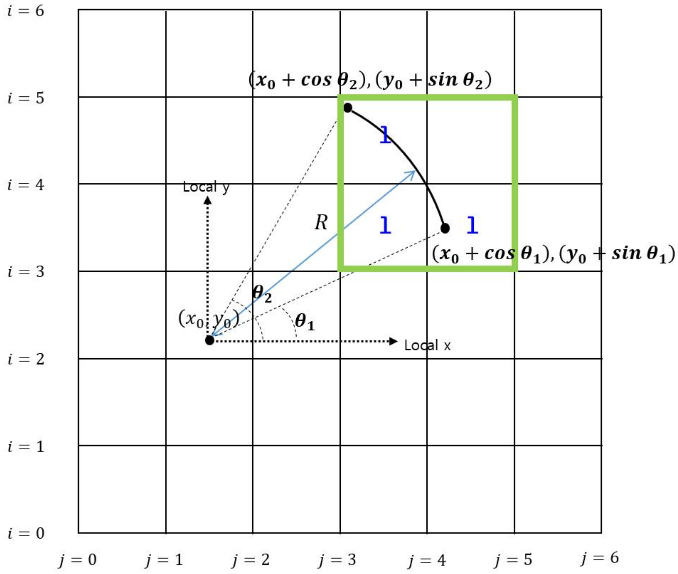

Fig. 2.16. Calculation range when θ_1 and θ_2 are in the first quadrant

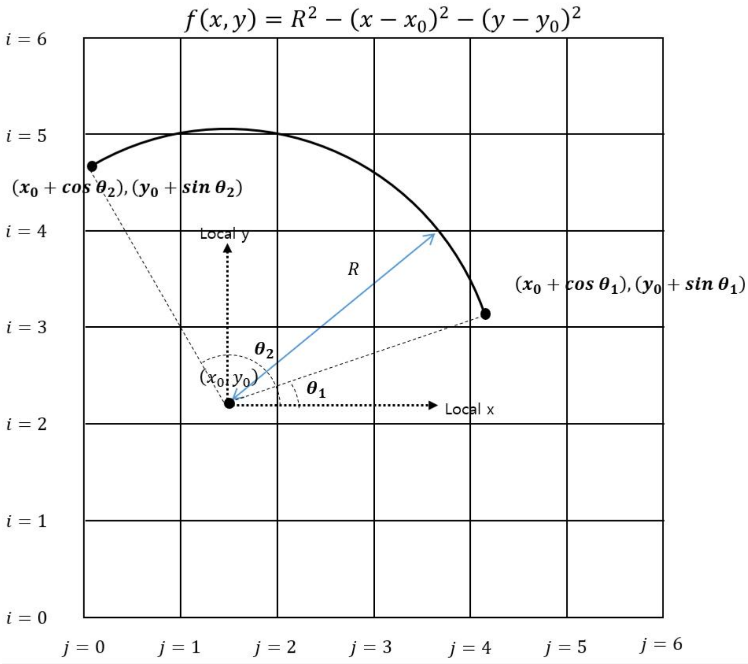

Fig. 2.17. θ_1 in the first quadrant and θ_2 in the second quadrant

Fig. 2.17.과 같이 해당 각도 내에서 호를 표현할 수 있다. 또한 Fig. 2.18.과 같이 계산 범위는 2개의 구간으로 나누어 계산을 수행한다. 초록색 박스 내의 계산 범위 식은 식(2.9), 식(2.10)과 같다.

$$(\text{int})(y_0 + R \bullet \sin\theta_1) \leq i < (\text{int})(y_0 + R \bullet \sin 90^\circ) \tag{2.9}$$

$$(\text{int})(x_0 + R \bullet \cos 90^\circ) \leq j < (\text{int})(x_0 + R \bullet \cos\theta_1) \tag{2.10}$$

하늘색 박스 내의 계산 범위는 식(2.11), 식(2.12)와 같다.

$$(\text{int})(y_0 + R \bullet \sin\theta_2) \leq i < (\text{int})(y_0 + R \bullet \sin 90^\circ) \tag{2.11}$$

$$(\text{int})(x_0 + R \bullet \cos\theta_2) \leq j < (\text{int})(x_0 + R \bullet \cos 90^\circ) \tag{2.12}$$

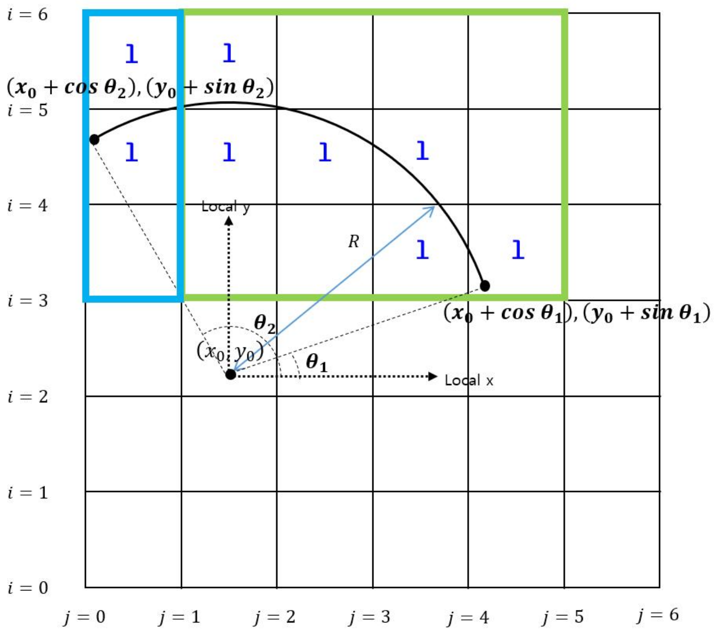

Fig. 2.18. Calculation range when θ_1 is in the second quadrant and θ_2 is in the second quadrant

2.6. 피스 내부 정의 알고리즘

Breath-First Search(BFS) 알고리즘은 격자 탐색 알고리즘 중 하나로, 너비 우선 탐색이라고 불린다.[26] BFS는 격자에서 시작 정점으로부터 거리가 가까운 정점부터 방문하며 탐색한다. 일반적으로 큐(Queue) 자료구조를 이용해 구현된다. 시작 정점을 큐에 넣고, 그 정점과 인접한 정점을 차례로 큐에 넣는다. 그리고 큐에서 하나씩 꺼내어 방문하며, 방문한 정점과 인접한 정점을 다시 큐에 넣는다. 이 과정을 큐가 비어질 때까지 반복하여 완료한다.

본 연구에서는 BFS 알고리즘을 활용하여 격자 내부를 외형 선에 할당된 값과 동일하게 '1'로 채웠다.

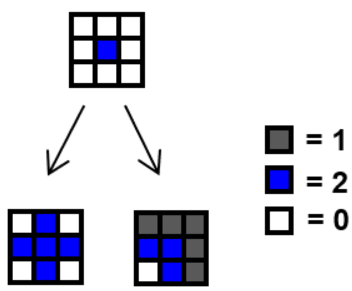

Fig. 2.19. Search direction of BFS algorithm

Fig. 2.19.에서와 같이 격자 내의 값이 '0'인 격자가 존재 할 때, 초기 위치를 지정한다. 초기 위치로 지정된 격자의 값은 '2'로 할당한다. 이때, BFS 알고리즘을 적용시키면 초기 위치를 기준으로 동, 서, 남, 북 방향으로 탐색을 시작한다. 만약, 탐색하고자하는 격자의 값이 '0'이라면 '2'의 값으로 반환하며 왼쪽 아래의 격자 형태를 갖춘다. 하지만 탐색하고자 하는 격자에 '0'이 아닌 '1'의 값이 존재한다면 '2'의 값으로 반환하지 않고 그대로 반환하여 오른쪽 아래의 격자 형태를 갖추게 된다.

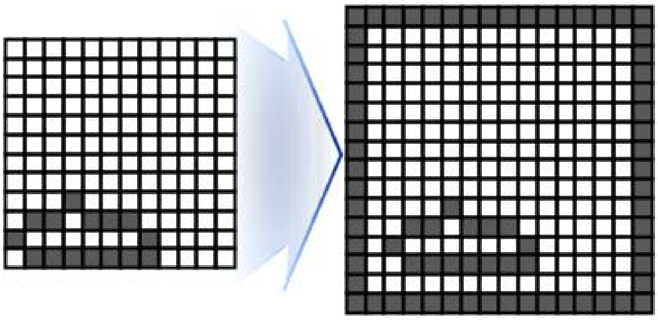

Fig. 2.20. Creation of additional grids

Fig. 2.20.과 같은 모양의 피스의 외형선이 정의 되었을 때 초기 격자 위치를 지정하기 위해서 기존의 격자 사이즈보다 바깥 방향으로 두 줄이 더 추가된 형태의 격자를 정의한다. 그리고 가장자리의 격자들은 모두 '1'의 값을 할당한다.

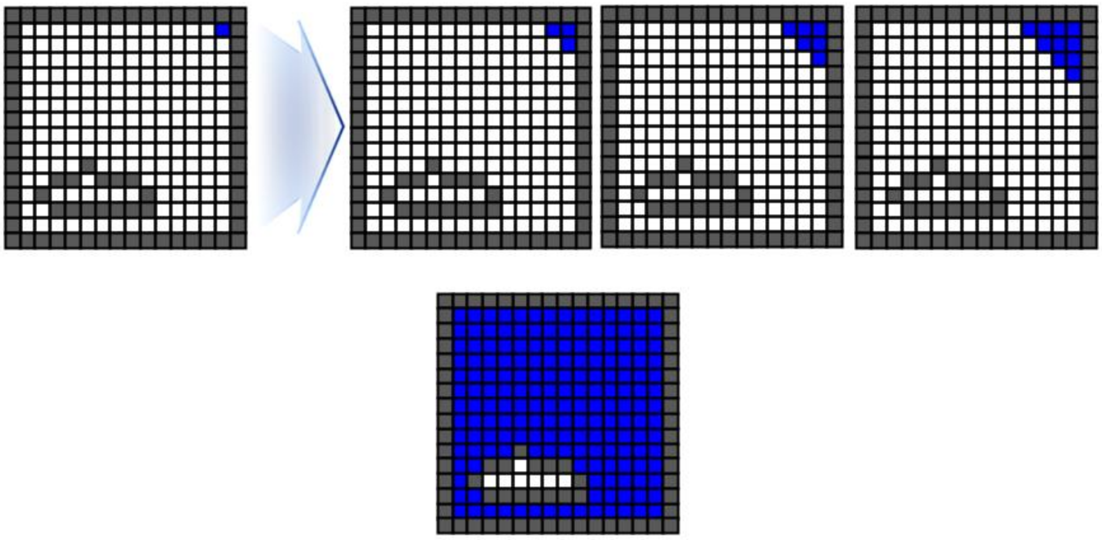

Fig. 2.21. Process of BFS Algorithm

Fig. 2.21.에서 초기 격자 위치는 '0'의 값을 가지는 격자 중 가장 오른쪽 위에 존재하는 격자로 설정하며 값은 '2'로 할당한다. 새로 추가된 격자 위에 초기 격자 위치를 지정하였으므로 피스 형태에 무관하게 해당 프로세스를 적용 할 수 있다. BFS 알고리즘은 재귀함수 형태를 갖추고 있으므로 피스 외부의 격자를 모두 탐색할 때까지 실행된다. 실행이 완료되면 결과 피스 외부가 '2'의 값으로 할당된다.

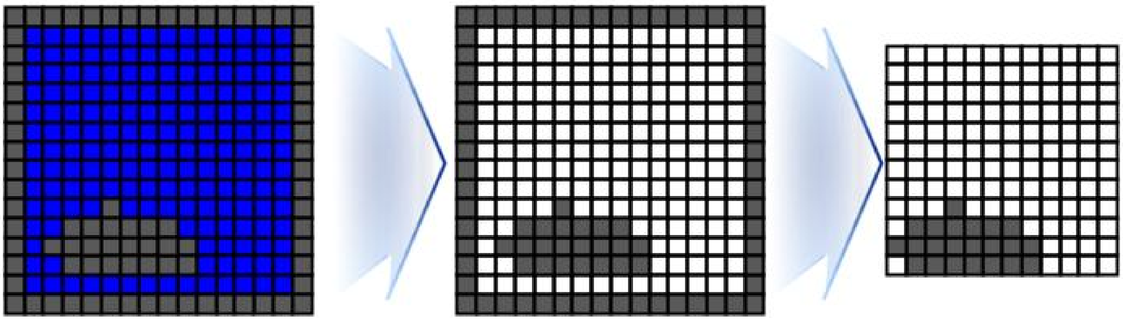

Fig. 2.22. Post-processing after applying the BFS algorithm

외형선 외부 격자가 모두 '2'을 값을 가지게 되므로 내부는 간단하게 '1'로 채울수 있다. '2'의 값을 가지는 격자들을 모두 '0'을 값을 가지도록 바꿔준다. 원본 형태의 격자 사이즈로 변환하면 외형선 내부를 모두 '1'로 채울 수 있다. Fig. 2.22.의 과정을 따른다.
최종적으로 모형선 피스의 일부를 격자로 표현하면 Fig. 2.23.와 같다.

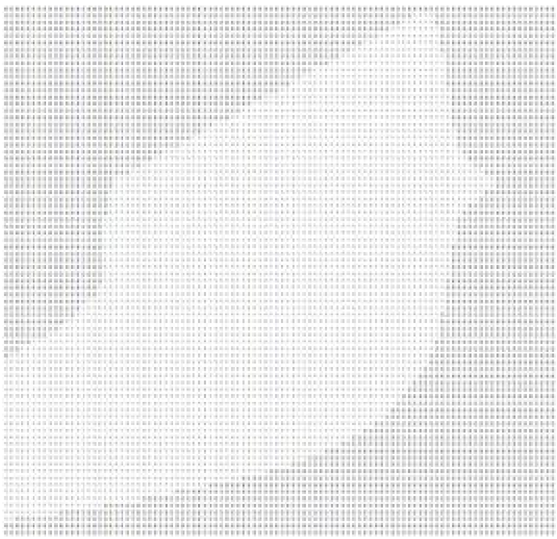

Fig. 2.23. Result of geometric representation of the model ship piece

2.7. 피스의 회전

모형선 피스의 피스들은 불규칙한 모양과 다양한 크기를 가지고 있기 때문에 부품을 회전시키면 배치 과정에서 다양한 배치 방법이 나타난다. 따라서 네스팅 과정의 효율성을 높이기 위해 부품의 회전이 필수적이다. 0˚에서 360˚까지 모든 회전 각도에 대해서 연속적으로 회전시킬 때 최적의 회전 각도를 찾을 수 있다. 하지만 많은 피스들이 이 과정을 거치게 될 경우 막대한 컴퓨팅 시간이 소요된다. 따라서 합리적인 각도 간격을 선정하는 일은 매우 중요하다.

모형선 피스의 피스들의 대부분은 가로 방향의 길이가 세로 방향의 길이 보다 긴 형태를 띄고 있다. 가로 방향의 길이가 세로 방향 길이와 비슷한 경우 모형선의 강도 측면에서 취약해 질 수 있기 때문이다. 또한, 이러한 특징으로 인해 가로 방향 길이가 긴 피스가 90˚를 회전하게 되었을 때 2D 시트의 세로 방향 길이보다 커진다. 배치가 될 수 없는 경우가 발생하기 때문에 적절한 각도를 선정하는 것이 중요하다. 회전 각도는 0˚, 5˚, 10˚, 15˚, 20˚, 30˚, 60˚, 90˚, 120˚, 150˚, 180˚, 210˚, 240˚, 270˚, 300˚, 330˚, 340˚, 345˚, 350˚, 355˚로 설정하였다. 또한 Fig. 2.24.에서 확인할 수 있듯 90˚와 같이 피스가 배치 될 수 없는 경우의 각도도 포함하며 3장에서 자세히 설명한다.

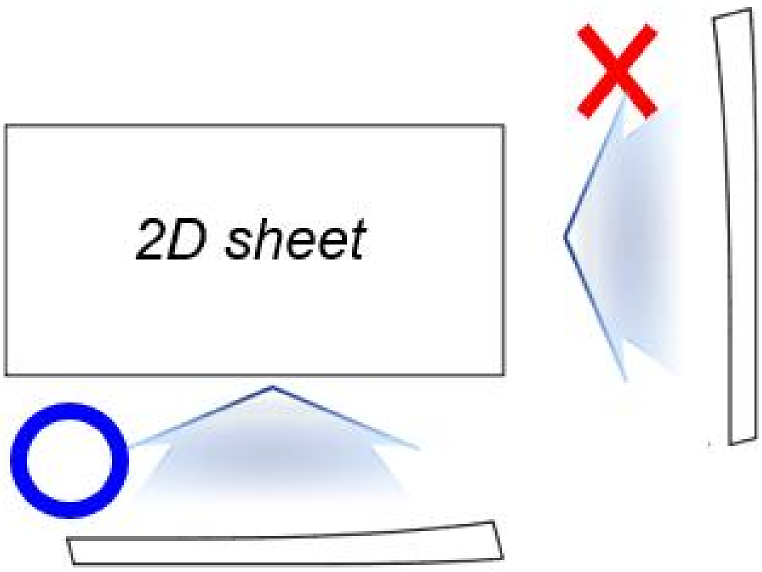

Fig. 2.24. Arrangement at angles that cannot be placed

피스의 회전은 식(2.13)에 따라 초기에 입력된 점 좌표(x_R,y_R)들을 글로벌 좌표계의 원점을 기준으로 하여 반시계 방향으로 변환한다.

$$\begin{pmatrix} x_N \\ y_N \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} x_R \\ y_R \end{pmatrix} \tag{2.13}$$

하지만 Fig. 2.25.과 같이 회전변환 과정을 거치게 되면 피스의 좌표 값이 ‘0’ 또는 자연수가 아닌 음수가 되는 경우도 발생한다. 이렇게 될 경우 격자는 ‘0’을 포함한 자연수 영역 내에서 생성되므로 격자에 피스를 표현하는데 있어 문제점이 발생한다. 이러한 문제점을 해결하기 위해 x축 또는 y축 방향으로 평행이동 하여 피스의 좌표 값이 0 또는 최소 자연수 값을 갖도록 하였다.

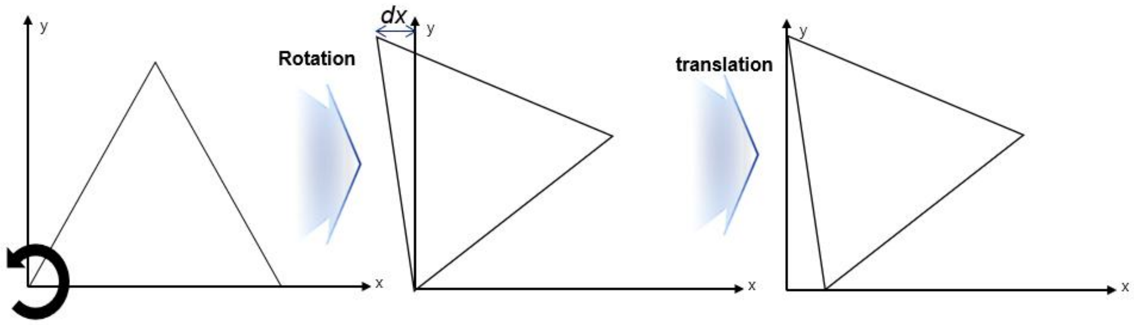

Fig. 2.25. Rotation of the piece

제3장 배치 알고리즘

3.1. 단일 판(single plate) 배치

단일 판에 모형선 피스를 배치하는 경우 모든 피스들이 배치되지 않고 남는 경우가 발생한다.(Fig. 3.1.)

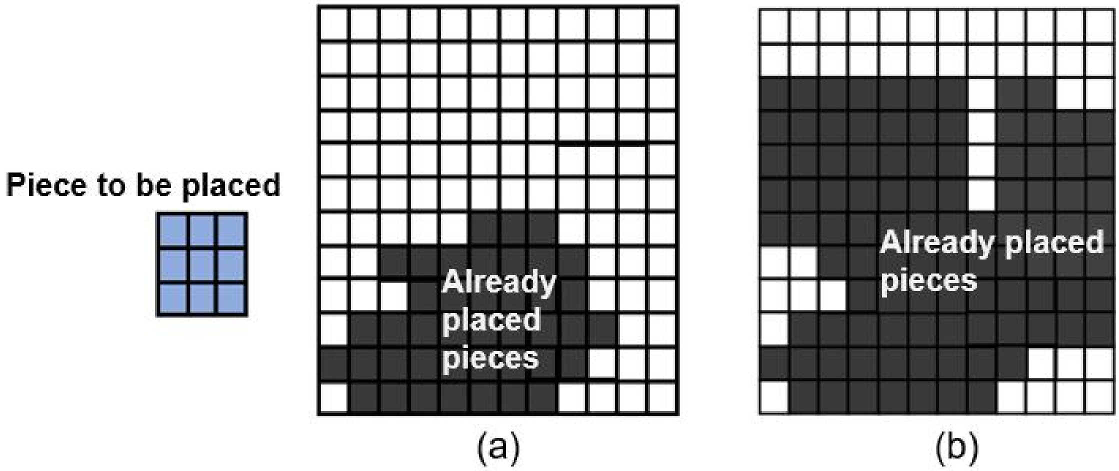

Fig. 3.1. Placing a piece on a single plate

Fig. 3.1.의 (a)의 경우 배치될 피스가 빈 격자 내부에서 배치 될 수 있지만 (b)의 경우 빈 격자 내부에서 배치가 불가능하다. 즉, 남은 피스들을 다시 배치 알고리즘을 통해서 배치를 진행해야한다는 것을 의미한다. 또는 사용자가 단일 판 내에 배치될 만한 피스들을 선택하고 배치 알고리즘을 적용하는 과정을 거치므로 추가적인 공수가 들기 때문에 완전한 자동화라고 볼 수 없다. 따라서 본 논문에서는 다중 판에 대한 배치 알고리즘을 제시한다.

3.2. 다중 판(multiple plate) 배치

모형선 피스의 피스를 2D 판에 배치하는 경우 2D 판의 크기가 피스의 크기에 비해 작기 때문에 다중 판에 배치 할 수 있는 알고리즘을 구현하는 것이 필수적이다. 다중 판의 개수는 사용자 변수로 지정하였으며 다중 판은 Fig. 3.1.과 같은 형태의 격자로 표현된다.

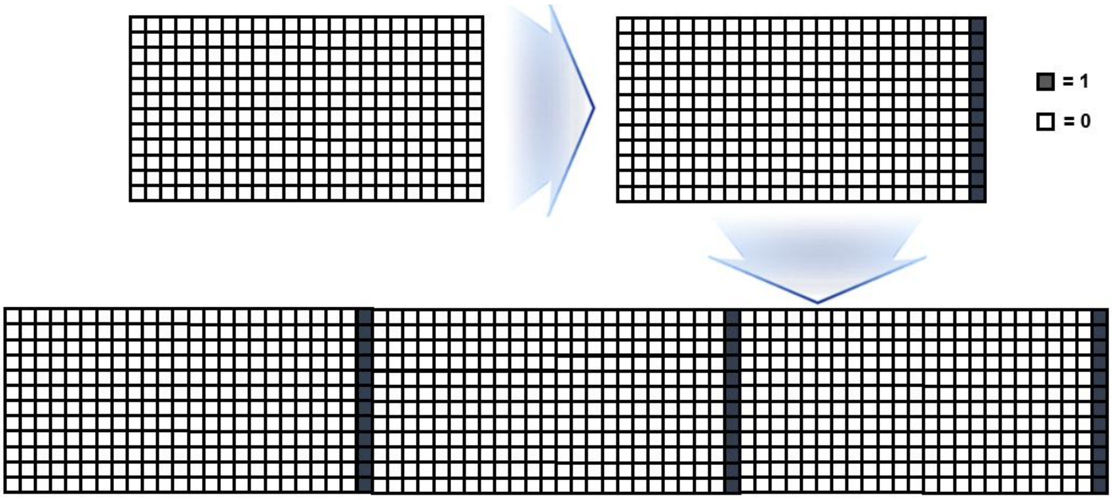

Fig. 3.2. Representation of multiplate grids

Fig. 3.2.과 같이 하나의 배치 격자 판이 있을 때, 오른쪽 가장자리에 값이 '1'로 할당된 격자를 추가한다. 그리고 사용자가 원하는 만큼 격자 판을 생성한다. 본 연구에서는 모형선 피스를 다중 판에 배치하는 알고리즘을 구현한다.

3.3. 배치 순서의 결정

3.3.1. 유전 알고리즘

유전 알고리즘(GA, Genentic Algorithm)이란 유전학 개념을 기반으로 하여 주어진 문제의 잠재해 사이에서 자연선택과 유전법칙을 적용해 목적함수를 최대화 또는 최소화 시켜 최적의 해를 찾는 전역 최적화 기법이다.[29] 유전 알고리즘은 후보해 집단(Population)을 가지고 있으며 각 후보 해는 이진문자 열로 나타내어진다.

유전 알고리즘은 후보 해들의 모집단(Population)으로 구성되어 있으며 이진 문자열에서 하나의 문자를 유전자(Gene)이라고 한다. 또한, 이진 문자열로 표현된 하나의 후보 해를 염색체(Chromosome)이라고 하며 Fig. 3.3.과 같다.

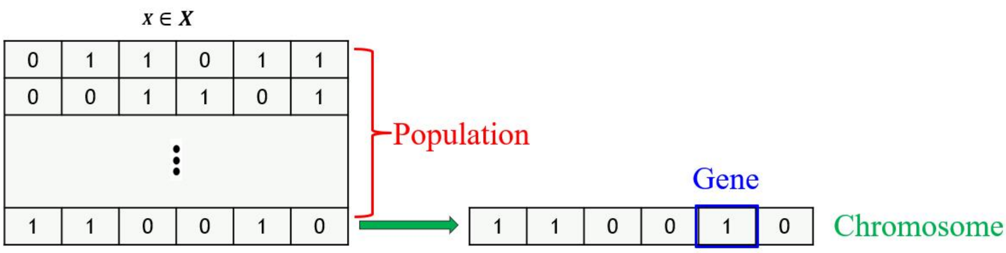

Fig. 3.3. The components of a poluation

유전 알고리즘은 선택 연산(Selection), 교차 연산(Crossover), 돌연변이 연산(Mutation)과 같은 개념을 사용한다. 선택 연산은 유전 알고리즘의 적합도 평가 값에 따라 순차적으로 나열하여 좋은 개체들만 선택하는 방법이다. 교차 연산은 두 후보 해에서 일부 유전자를 교환하여 새로운 후보 해를 생성하는 방법이며 돌연 변이 연산은 일부 유전자를 무작위로 변형하여 새로운 후보 해를 만드는 방법이다. Fig. 3.4은 일반적인 유전 알고리즘의 기본 구조를 나타낸다.

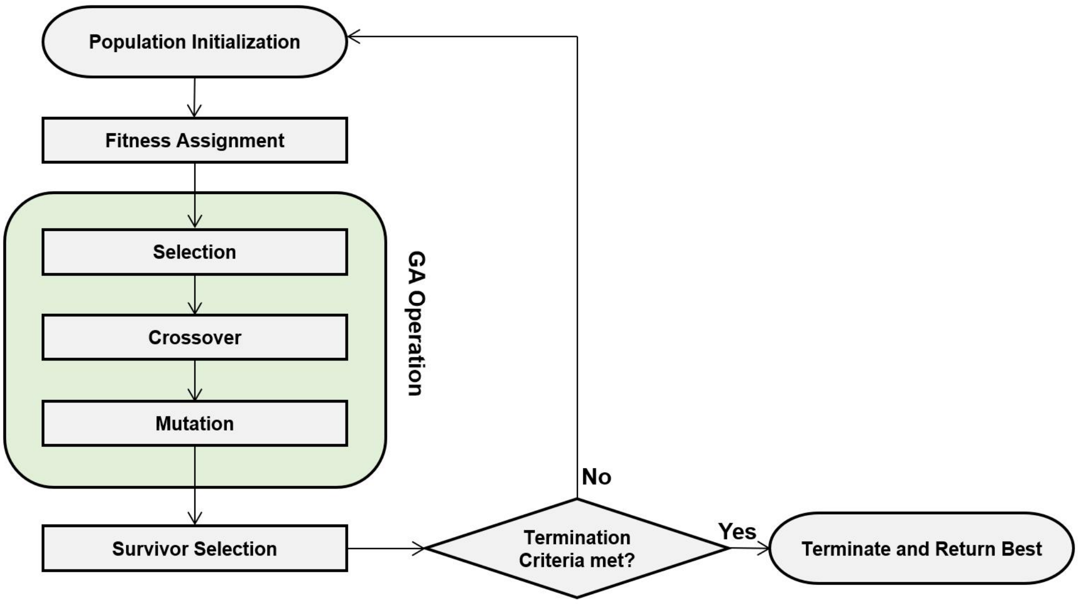

Fig. 3.4. Basic steps of genetic algorithm

유전 알고리즘의 순서는 Table. 3.1.과 같다.[29] 초기에 염색체로 구성된 해 집단을 구성한다. 그리고 적합도 평가 함수를 통해 적합도를 계산한 후 선택 연산, 교차 연산, 돌연변이 연산을 순차적으로 진행한다. 적합도 값이 수렴할 때까지 반복 연산을 진행한다.

Table. 3.1. The procedure of a general GA[30]

Algorithm. Genetic Algorithm
$Step1$: Set $t=1$. Randomly make N solutions to form the chromosome, P_1. Evaluate the fitness of solution in P_1.
$Step2$: ***Fitness assignment***: Evaluate and obtain a fitness value to each solution $x \in Q_t$ using its objective function value.
$Step3$: ***Selection***: Select N solutions from Q_t based on their fitness and copy them to P_{t+1}
$Step4$: ***Crossover***: Generate an offspring chromosomes Q_t as follows: 4.1 Select several solutions x from P_t based on the fitness values 4.2 Using a crossover operator, generate offspring and add them to Q_t
$Step5$: ***Mutation***: Mutate each solution $x \in Q_t$ with a predefined probability
$Step6$: If the object function is converged, terminate the search and return to the current best solution, else, set $t=t+1$ go to $Step2$:

본 연구에서는 유전 알고리즘을 이용하여 임의의 배치 순서와 각도를 결정하고 최종적으로 배치 후 판의 잔재의 넓이를 최솟값으로 갖는 배치 순서와 각도를 구한다. Kim[31]의 연구에서는 크기가 큰 피스와 작은 피스를 나누어 각도 변화에 대한 폭을 다르게 주었으나 본 연구에서는 두 가지 영역을 나누지 않고 각도 변화 크기를 동일하게 주어 계산하였다. 또한, 단일 판에 대한 배치와 다중 판에 대한 배치의 경우 적합도 평가 함수의 구성이 달라지며 3.2.1.1절에서 자세히 설명한다.

3.3.1.1. Fitness function

모형선 피스들의 배치가 완료되면 Fitness function을 통해서 적합도 평가를 진행한다. 모형선 피스의 배치를 위한 적합도 평가 함수는 식(3.1)과 같다.

$$fitness = \sum_{i=1}^{i=NumberOfPlates} i^2 \cdot f_i \tag{3.1}$$

식(3.1)의 i는 판의 개수를 뜻하며 f_i는 i번째 판에서의 판의 스크랩 면적을 최소화하는 함수이다. f_i의 계산 과정은 Fig. 3.5. 같다. 삼중 판에 4개의 피스가 Fig. 3.5.와 같은 형태로 배치가 완료되었다. 이때, 각 판에서 피스들이 차지한 격자의 최대 행과 열의 수를 계산한다. 최대 행과 열이 만드는 직사각형 내에서 피스가 차지한 격자를 제외한 나머지 격자의 수를 계산한다. 첫 번째 판에서 구한 나머지 격자수는 f_1, 두 번째 판에서 구한 나머지 격자수는 f_2가 된다. $fitness$ 값이 낮을수록 살아남을 확률이 높다. i^2항은 일종의 가중치 항이며 판의 개수를 최대한 적게 사용하도록 만드는 항이다. 이 항이 존재하지 않다면 Fig. 3.5.의 나머지 격자수가 Fig. 3.6.보다 적으므로 세대를 거듭하게 될 경우 Fig. 3.5.의 배치 형태가 살아남게 될 것이다. 하지만 i^2항에 의해 Fig. 3.5.에서 배치 후 계산된 적합도 값은 Fig. 3.6.에서 계산된 적합도 값보다 크다. 따라서 세대를 거듭하게 될 경우 Fig. 3.6.의 배치 형태만 존재하게 된다.

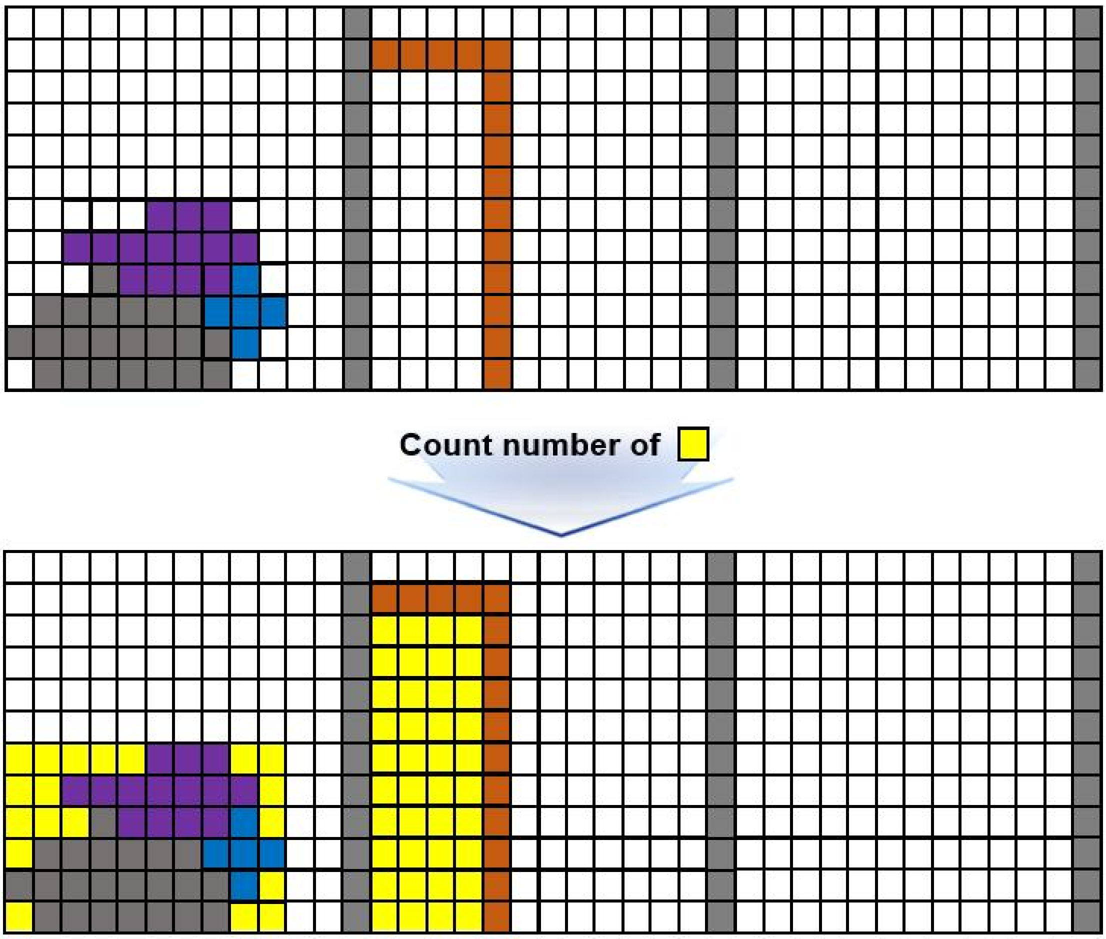

Fig. 3.5. Case1 : Calculation of the fitness function

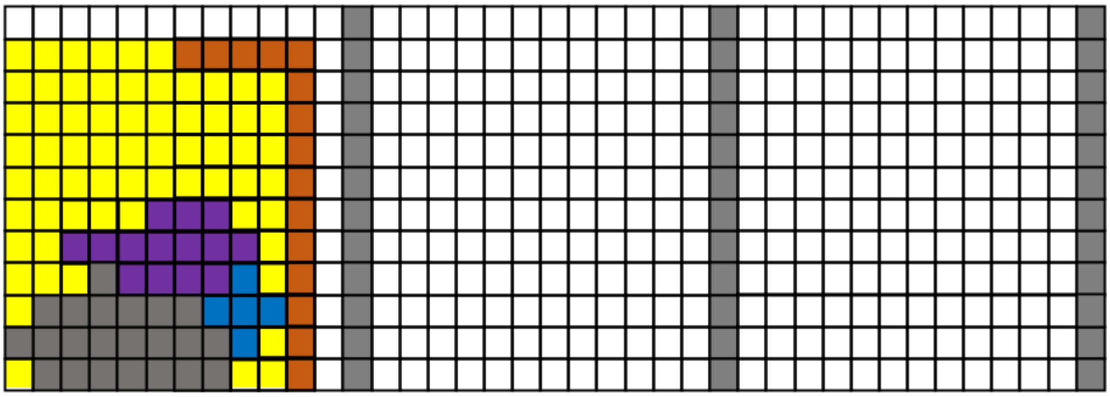

Fig. 3.6. Case2 : Calculation of the fitness function

3.3.1.2. Selection

선택연산은 각 세대에서 좋은 형질의 개체를 잘 유지하는 것이 목적이다.[29] 선택연산 방법은 다음과 같다. 먼저, 적합도 평가 함수 계산 결과를 오름차순으로 정렬한다. 범위 내의 빈 격자의 수와 적합도 평가 함수의 결과 값은 비례한다. 즉, 적합도 평가 함수의 결과 값이 낮을수록 해당 개체가 살아남을 확률이 높다. 본 연구에서는 부모세대에서 적합도 평가 함수를 계산하고 이들 중 상위 50%의 결과는 자식세대로 채택하여 구성하였고 상위 결과 값들의 교차연산과 돌연변이 연산을 통해 적합도 평가 함수 값이 낮은 개체들을 자식세대에 추가하였다. 적합도 함수 값이 수렴할 때까지 반복 수행한다.

3.3.1.3. Crossover

교차연산은 두 개의 부모개체의 유전자로부터 재조합을 통해 새로운 자식 유전자를 생성하는 과정이다.[29] 일반적으로 교차연산이 수행되는 방법은 부모 유전자 중 랜덤하게 선택된 지점을 기준으로 하여 두 부분으로 나눈다. 그리고 두 부모 유전자의 교차점을 기준으로 하여 한 쪽 부분을 서로 교환시키게 되면 교환된 두 부분을 조합해 새로운 자식 유전자를 생성하게 된다. 이러한 과정을 반복하여 다수의 자식 유전자를 생성하게 된다. 본 연구에서도 마찬가지로 하나의 부모 유전자를 선택된 저짐을 기준으로 하여 자식 유전자에 복사시킨다. 그리고 다른 부모 유전자에서 동일하게 선택된 지점을 기준으로 하여 반대쪽 나머지 유전자 절반을 복사시킨다. 하지만, 배치 순서를 결정하는 문제의 경우에 이 방법을 적용해서 교차연산을 진행하게 될 경우 같은 피스가 유전자에 복사되는 경우가 발생한다.

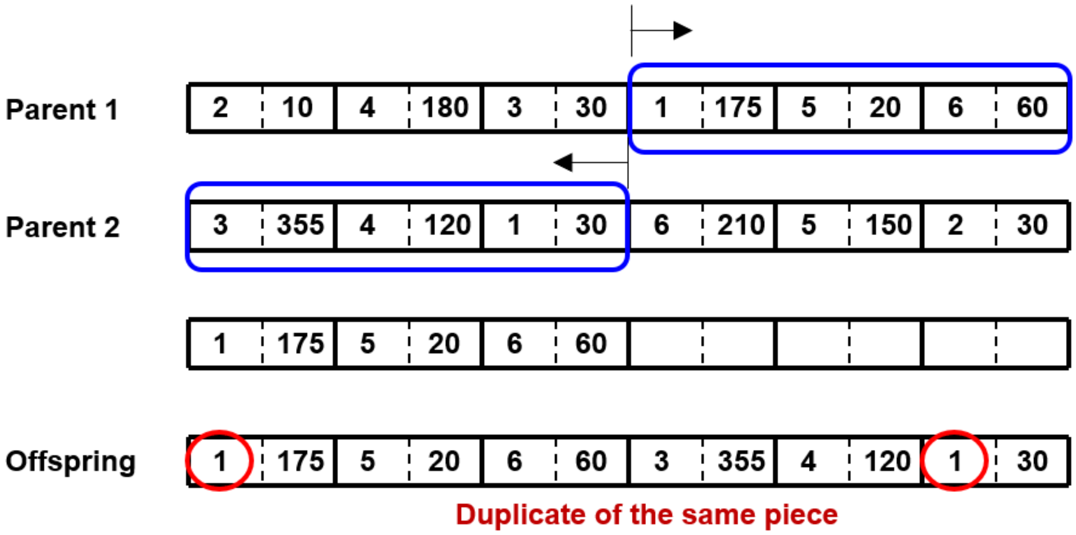

Fig. 3.7. Duplication of genes

Fig. 3.7.과 같이 배치 순서 결정 문제에 맞도록 다음과 같은 방법을 적용하였다. 한 개의 부모 개체의 유전자를 선택된 지점을 기준으로 복사하고 다른 부모 개체의 유전자 중 이미 자식 개체에 복사되지 않은 유전자를 순차적으로 복사함으로써 동일한 피스가 중복되는 현상을 제거했다. (Fig. 3.8.)

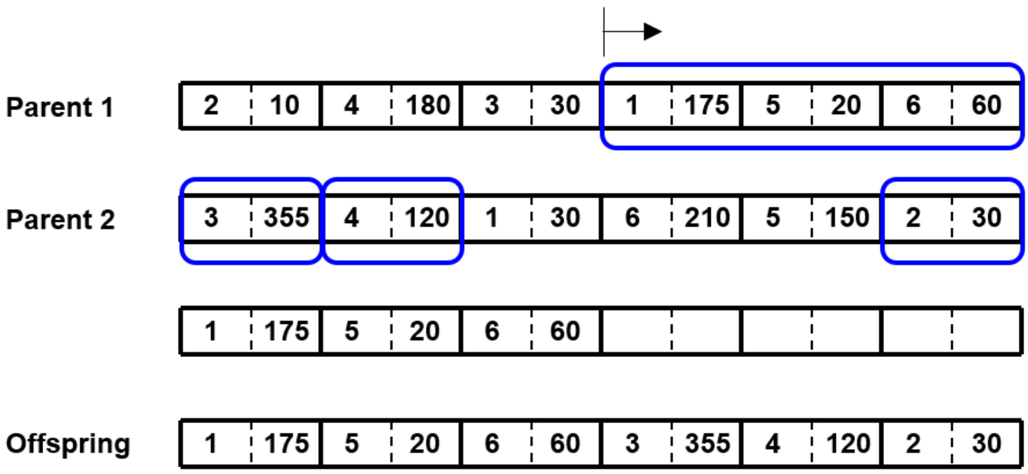

Fig. 3.8. Elimination of gene duplication

3.3.1.4. Mutation

선택연산이 부모 개체에서 존재하는 유전자를 조합하여 새로운 자식 개체를 만드는 과정이었다면 돌연변이 연산은 부모 개체에서 존재하지 않는 유전자를 무작위로

도입하여 새로운 유전자 조합을 생성하는 과정이다. 이러한 과정은 후보 해 집단의 다양성을 유지하는데 큰 역할을 한다. 다양성은 유전 알고리즘에서 중요한 요소 중 하나로, 후보 해 집단의 다양성이 유지되지 않는다면 최적 해를 찾는 것이 어려워 진다.

본 연구에서는 개체의 선택된 유전자마다 각도와 처음 주어진 피스의 번호가 속성 값으로 내장되어있다. 각도의 경우 다른 유전자와 관계없이 난수를 발생시켜 무작위로 변환이 가능하다. 하지만 해당 피스에서 발생된 난수에 의해 각도 값이 변환되었을 때 피스가 격자에 표현되지 못하는 경우가 발생할 수 있다. (Fig. 3.9.)

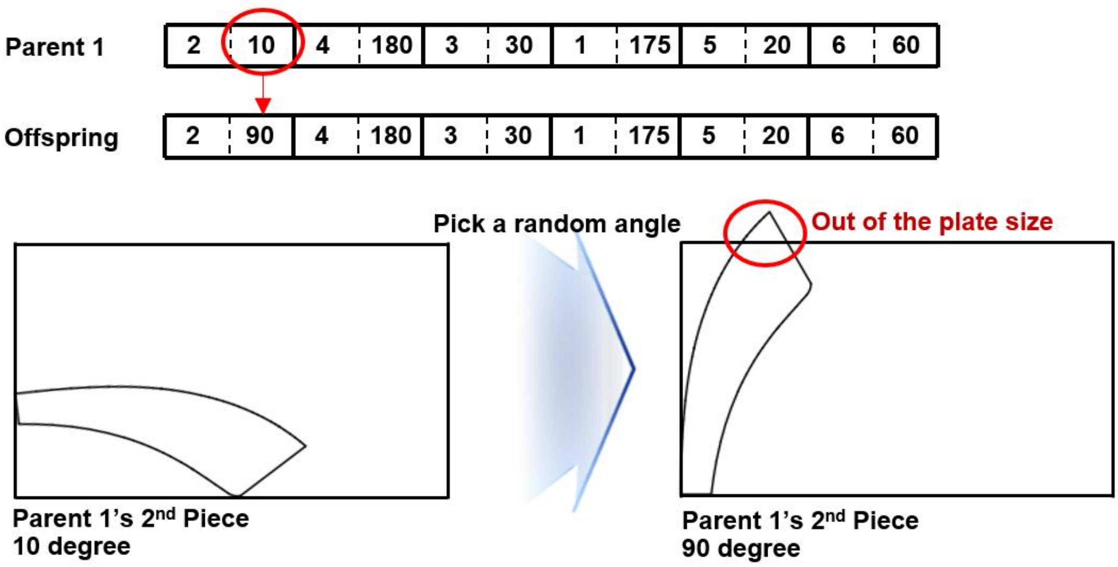

Fig. 3.9. Increased piece vertical size due to rotation of angle

이러한 문제점 발생할 경우 난수를 다시 발생시켜 격자에 표현 가능한 난수가 할당되도록 설정하였다. 그리고 피스의 번호의 돌연변이 연산인 경우 난수를 발생시켜 무작위로 번호를 변경할 때 피스 번호의 중복 현상이 발생할 수 있다. 따라서 무작위 난수를 통해 두 개의 유전자를 선택하고 두 유전자 정보 중 피스 번호를 서로 교환하였다.(Fig. 3.10.)

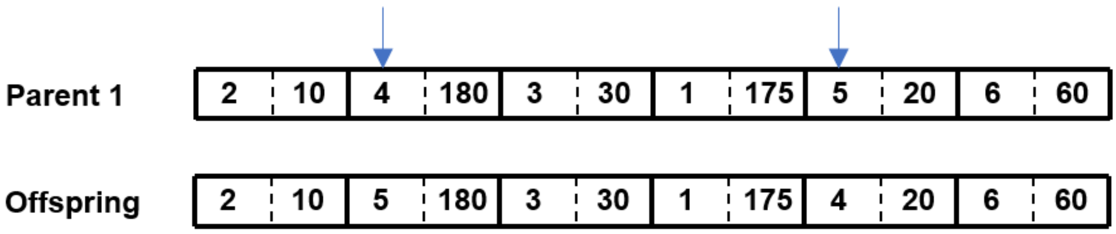

Fig. 3.10. Mutation operation on layout order

3.4. 피스의 배치

3.4.1. 배치 알고리즘

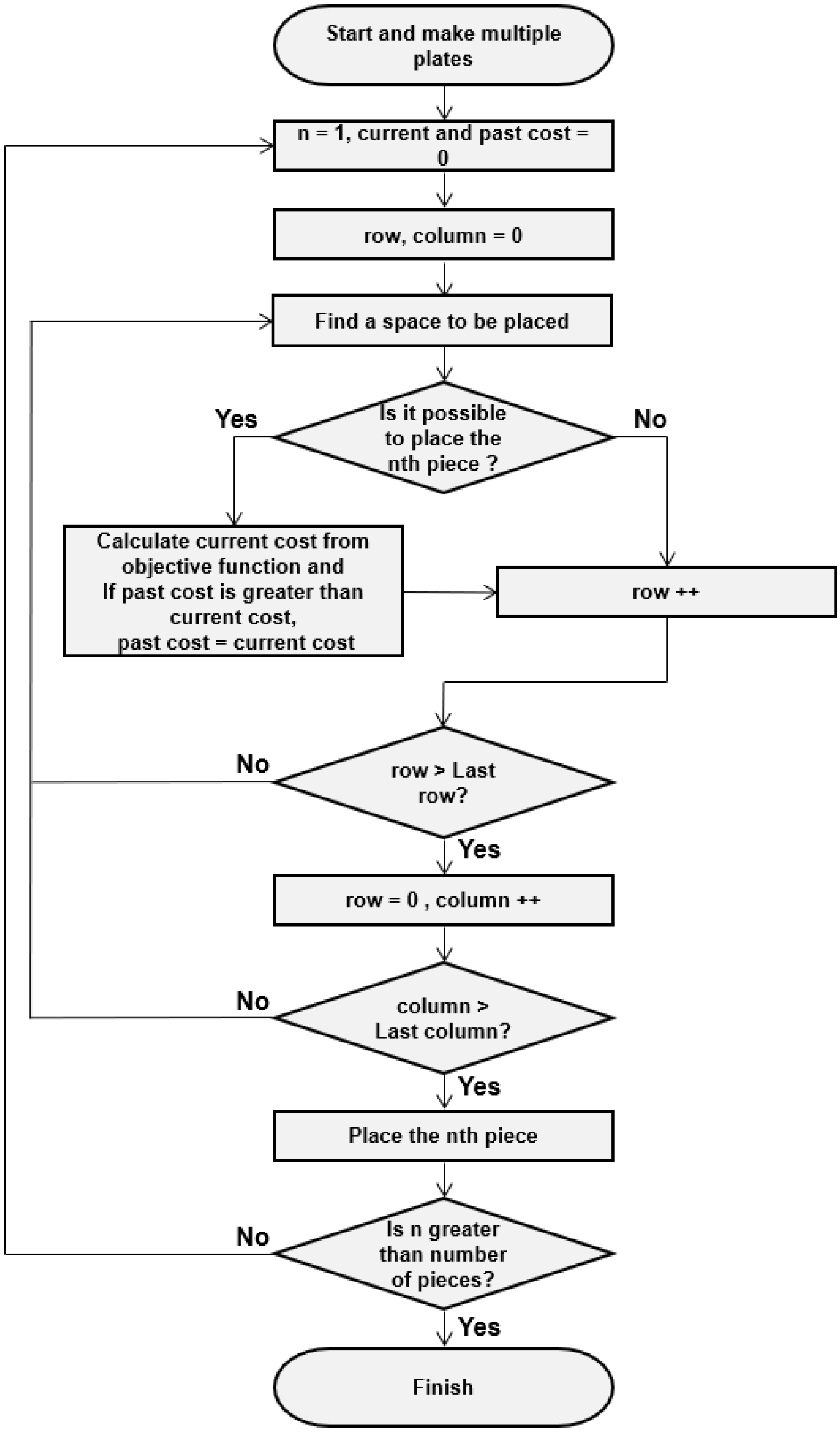

Fig. 3.11. Layout algorithm flow chart

배치 알고리즘은 기본적으로 Kang[10]이 제안한 내용을 따른다. 일반적인 경우 단일 판에 피스들을 배치하는 것을 목적으로 하나 본 연구에서는 다중 판에 피스들이 배치되는 것을 목적으로 진행되기 때문에 만약 피스들이 첫 번째 판에 모두 배치되지 못한다면 두 번째 판에 배치할 수 있도록 알고리즘을 설정하였다. 배치 알고리즘에 대해 설명하면, 첫 번째 순서로 배치되는 피스는 첫 번째 판의 왼쪽 하단에 배치되도록 설정하였다. 이 후 배치될 피스들은 알고리즘의 순서에 맞게 순차적으로 행 또는 열을 한 칸씩 이동시키며 겹침 여부를 판단한다. 이때 겹침 여부 판단은 각 피스들의 형상은 격자마다 '1'로 정의되어있으므로 배치되었을 때 판의 '1'의 개수가 배치되기 전 피스들의 형상에 대한 '1'의 개수와 동일하지 못하면 겹쳤다고 판단하고 행 혹은 열 방향으로 이동시킨다.(Fig. 3.12.)

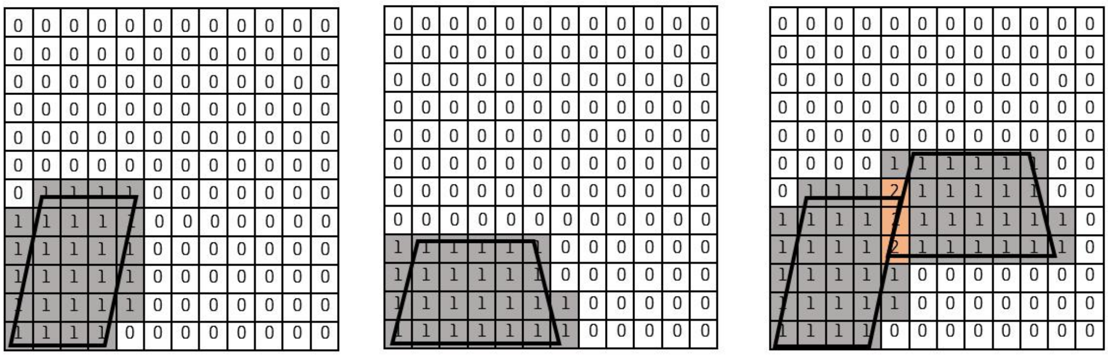

Fig. 3.12. Overlap detection of two pieces

배치 될 피스의 탐색 범위는 Fig. 3.13.과 같다. 만약 탐색 범위를 설정하지 않고 모든 격자에 대해서 탐색을 진행한다면 탐색에 걸리는 시간은 크게 증가한다. 따라서탐색 범위는 이미 배치된 피스들의 최대 행과 최대 열을 값과 배치될 피스의 최대 행과 최대 열의 값을 합하여 그 범위에서만 탐색하도록 설정하였고 그림의 Case1과 같다. 이때, 배치를 진행하다보면 판의 격자 사이즈를 초과하여 배치되는 경우가 발생한다. 해당 문제를 해결하기 위해 탐색 범위를 격자 자체의 최대 행 또는 열까지 탐색하도록 설정하였으며 Case2의 경우 최대 열만 초과하였으므로 첫 번째 판 내에 피스가 배치된다. 마지막으로 행과 열 모두 값이 격자 사이즈를 초과하게 되면 우선 첫 번째 판 전체에서 탐색을 진행하나 배치가 되지 않을 경우 두 번째 판에 배치를 하게 된다.

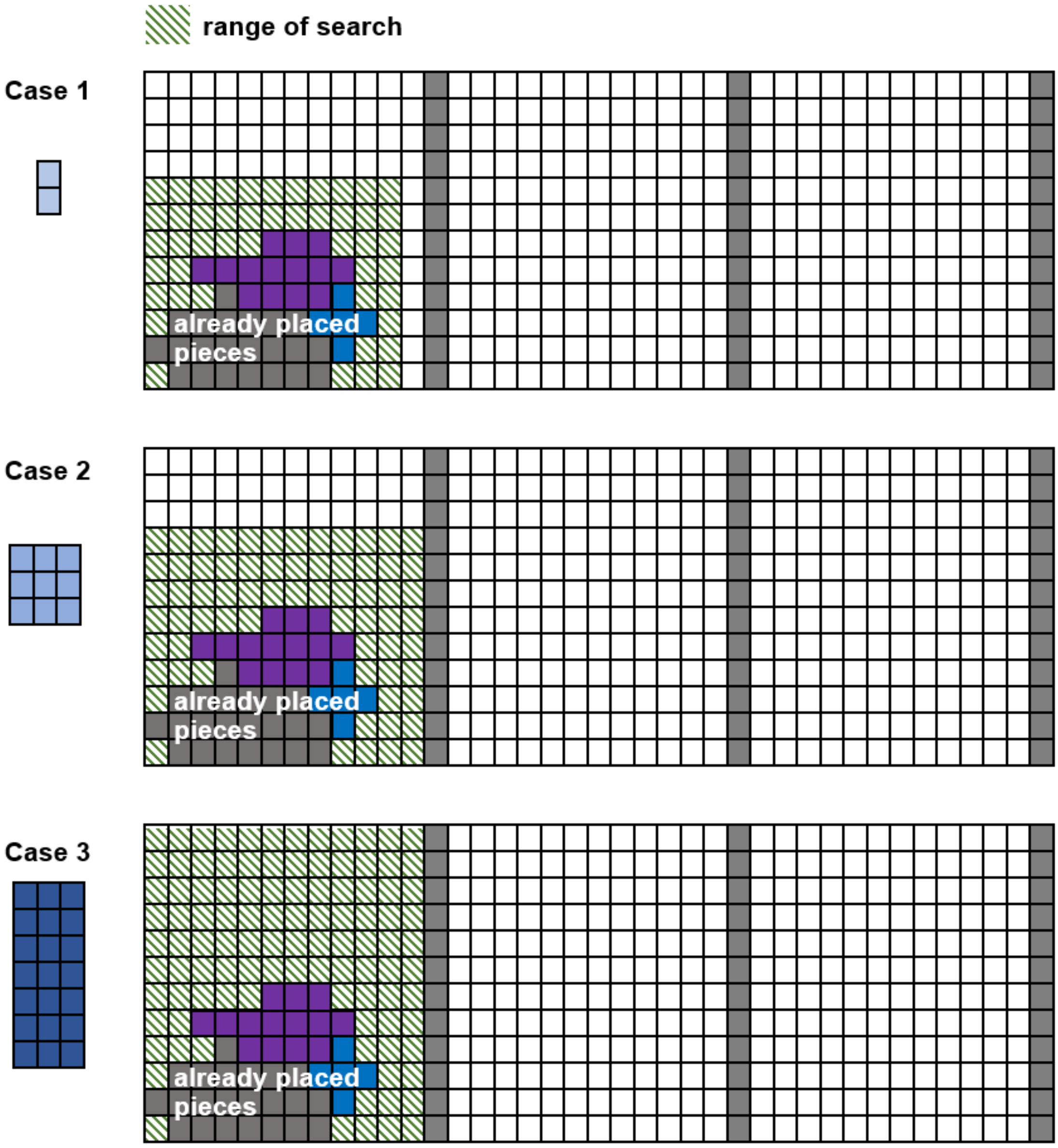

Fig. 3.13. Search Range of pieces to be placed

3.4.2. 목적함수

배치 될 피스가 겹침이 발생하지 않는다면 목적함수(Objective functions)를 통해 비용(Cost)을 계산한다. 탐색 범위 내의 모든 격자들을 탐색하고 비용함수를 계산하여 비교한 후 비용 값이 가장 작은 경우의 배치 결과를 선택한다. Kang[10]이 제시한 목적 함수의 경우 단일 판에 대한 목적함수로써 다음과 같이 구성된다.

$$Min(F) = f_x + f_y + f_{xy} \quad (3.2)$$

f_x, f_y, f_{xy}항은 그림과 같다.

f_x항의 경우 피스가 아래쪽으로 배치되게 하는 항이며 배치 될 피스의 아래 방향의 빈 격자의 수로 계산된다. f_y항의 경우 피스가 왼쪽으로 배치되게 하는 항이며 배치 될 피스의 왼쪽 방향의 빈 격자의 수로 계산된다. 마지막으로 f_{xy} 항은 피스가 직사각형 형태를 이루며 배치되도록 하는 항이며 배치 될 피스의 최대 격자 내의 빈 격자의 수이다. (Fig. 3.14.)

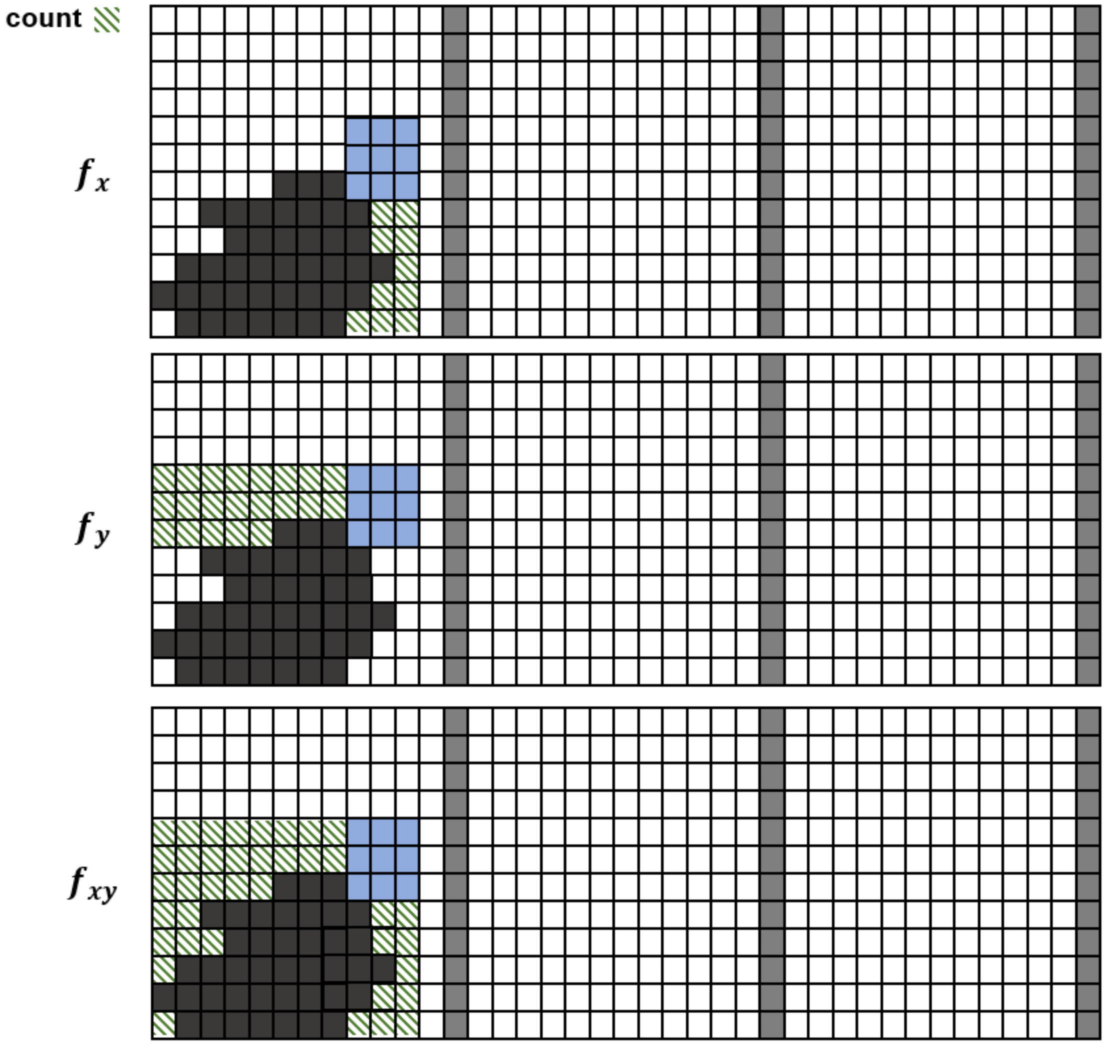

Fig. 3.14. Calculation of the objective function f_x, f_y, and f_{xy}

Lee[32]는 판의 스크랩 율(Scrap ratio)를 고려하여 각 항에 가중치계수를 곱해 수치실험을 진행하였다. 그 결과 남는 판의 원재료의 재사용적인 측면에서 바라볼 때, f_y항과 f_{xy}항이 주요하게 관여한다는 것을 알 수 있다.

본 연구에서는 f_y항과 f_{xy}항의 개념을 이용하였으며 다중 판의 배치에 맞도록 목적함수를 재구성하였다. 각 판에 대해서 목적함수를 계산하여 합하는 형식으로 구성된다. 재구성된 목적함수는 식(3.3)이다.

$$Min(F) = \sum_{i=1}^{NumberOfPlates} (f_y)_i + (f_{xy})_i \qquad (3.3)$$

3.5. 배치 결과

본 연구에 적용된 실제 판의 크기는 세로 : 1200mm, 가로 : 2400mm 이며 25mm와 50mm 두 가지 경우로 나누어 격자사이즈를 지정하였고 비교하였다. 회전 각도는 0˚, 5˚, 10˚, 15˚, 20˚, 30˚, 60˚, 90˚, 120˚, 150˚, 180˚, 210˚, 240˚, 270˚, 300˚, 330˚, 340˚, 345˚, 350˚, 355˚로 설정하였다. 길이 방향 부채들이 많이 존재하기 때문에 20˚도 까지 5˚간격으로 지정하였고 다른 각도의 경우 30˚도 간격으로 설정하였다. Fig. 3.15.와 같이 8개의 모형선 피스들로 다중판 배치를 진행하였고 배치 완료된 피스들은 post script 형식으로 출력하였다. 또한 실험이 진행된 컴퓨터 사양은 Intel(R) Core(TM) i9-10900 CPU @ 2.80GHz와 16.0GB RAM이다.

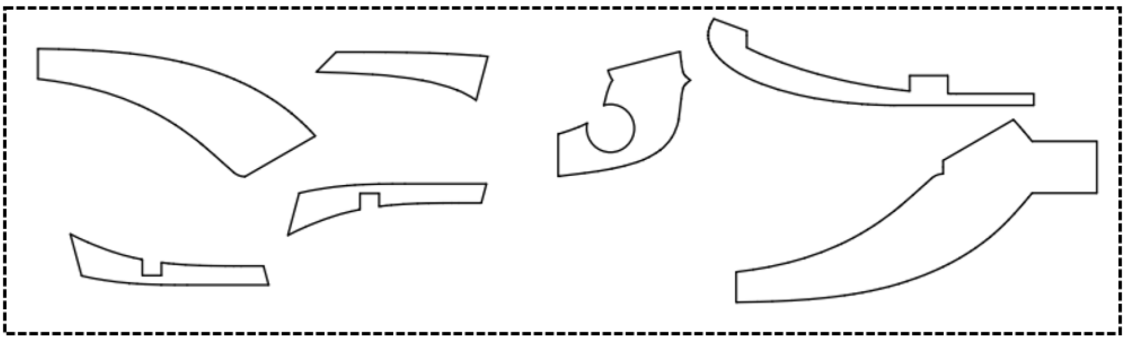

Fig. 3.15. Pieces of the Model ship

5가지의 다른 조건에 따라 모형선 피스들을 배치하였고 적합도 평가 값은 낮을수록 자재의 버림율이 적다. 적합도 평가 값은 5가지 경우 모두 동일하게 적용하였으며 앞서 제안된 식(3.1)과 같다. Fig. 3.16과 Fig. 3.17은 격자사이즈 25mm일 때 계산된 배치 결과이다. Fig. 3.16.는 제안된 배치 알고리즘을 사용하였고 모형선 피스의 종횡비 순서대로 배치한 결과이며 적합도 평가 값은 3.19381이다.

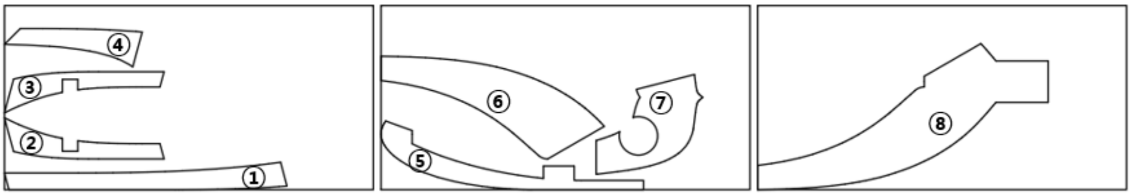

Fig. 3.16. Arrangement in order of aspect ratio

Fig. 3.17.은 모형선 피스들의 넓이 순서대로 배치한 결과이며 적합도 평가 값은 1.62284이다.

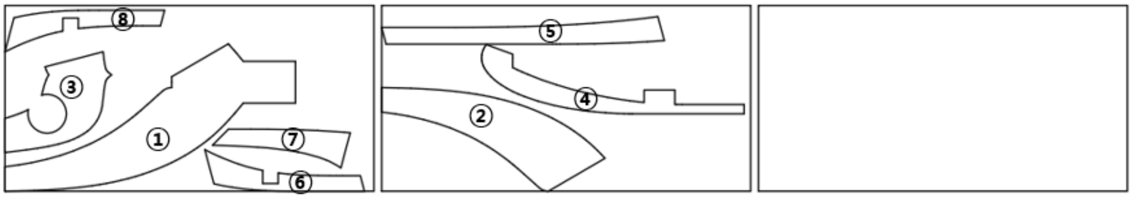

Fig. 3.17. Arrangement in order of area

Fig. 3.16.과 Fig. 3.17과 같이 순서와 각도를 지정해놓고 배치를 진행할 경우 다른 순서나 각도를 고려하지 않고 배치 알고리즘을 적용하였기 때문에 결과 값이 최적의 배치인지 고려할 수 없다. 하지만 이미 순서와 각도를 정한 후 배치를 진행하여 최적화 알고리즘을 적용하지 않기 때문에 시간적인 측면에서 이점이 있다.

Fig. 3.18.은 유전 알고리즘을 통해 배치 순서를 결정하였으며 선행 이동을 열 방향으로 지정하였고 배치 알고리즘의 목적함수는 f_{xy}만 설정하였다. 적합도 평가 값은 1.27155이다.

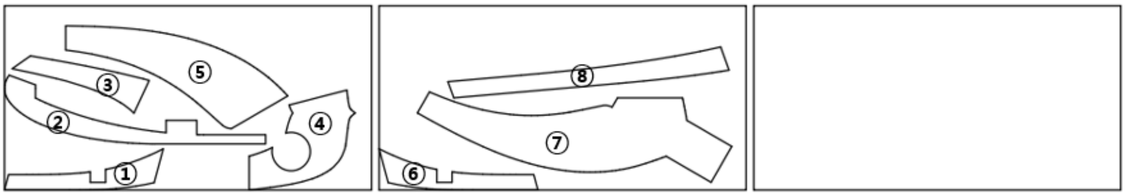

Fig. 3.18. Case 1 of Arrangement using genetic algorithm

Fig. 3.19.도 마찬가지로 유전 알고리즘을 통해 배치 순서를 결정하였으며 선행이동 방향과 목적함수는 Fig. 3.18과 동일하다. 적합도 평가 값은 1.17361 이다.

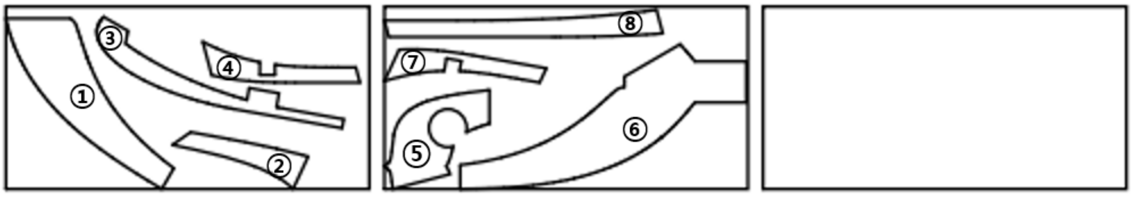

Fig. 3.19. Case 1 of Arrangement using genetic algorithm when the grid size is 50mm

Fig. 3.18.과 Fig. 3.19.의 배치 결과 모두 두 개의 판만 사용하였다. 또한 피스가 배치된 최대 직사각형 격자 내의 빈 격자의 수를 세어 적합도 평가 값을 계산하였으므로 격자 사이즈가 서로 다른 경우 적합도 평가 값의 비교가 불가능하다. 하지만 두 경우의 배치 결과를 보면 격자 사이즈가 더 작을 때 각 판에서 직사각형 모양으로 남는 넓이의 크기가 더 큰 것을 확인할 수 있다.

Fig. 3.20.은 유전 알고리즘을 통해 배치 순서를 결정하였으며 선행 이동을 행 방향으로 지정하였고 배치 알고리즘의 목적함수는 f_{xy}만 설정하였다. 적합도 평가 값은 1.10652이다.

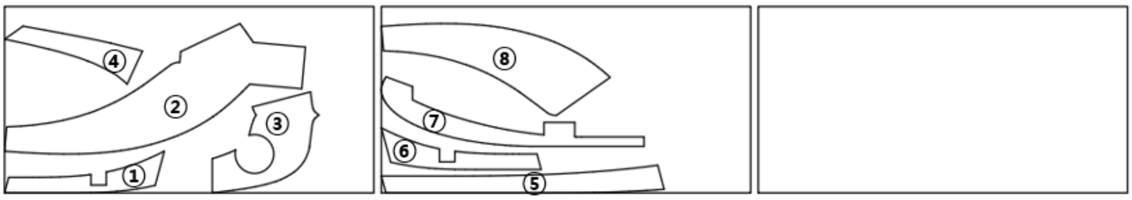

Fig. 3.20. Case 2 of Arrangement using genetic algorithm when the grid size is 25mm

Fig. 3.21.은 유전 알고리즘을 통해 배치 순서를 결정하였으며 선행이동 방향과 목적함수는 Fig. 3.20과 동일하다. 적합도 평가 값은 1.17361 이다.

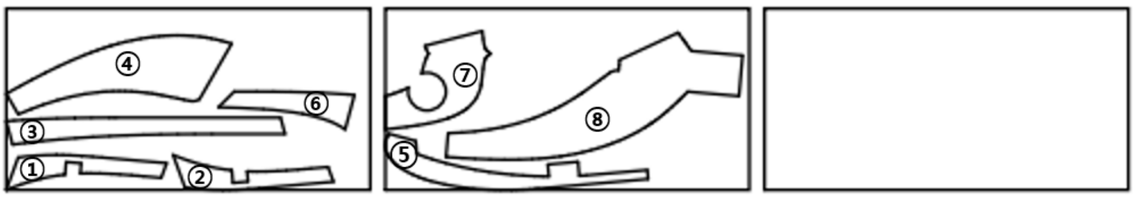

Fig. 3.21. Case 2 of Arrangement using genetic algorithm when the grid size is 50mm

Fig. 3.21.를 통해서 5번 피스가 두 번째 판에 배치가 되었지만 6번 피스는 첫 번째 판에 되는 것을 확인할 수 있다. 이를 통해서 가능한 적은 수의 판을 사용하고자 하는 알고리즘의 특성을 확인할 수 있다. 또한 Fig. 3.20.과 Fig. 3.21.에서도 Fig. 3.18.과 Fig. 3.19와 마찬가지로 자재 사용 범위가 격자의 사이즈가 클 때 더 큰 것을 확인 할 수 있다. 뿐만 아니라, 모형선 피스 특성상 길이 방향 피스가 많이 존재한다. Case 1과 Case 2의 경우를 비교하였을 때 길이 방향 피스가 많이 존재할 때 행 방향 선행이동보다 열 방향 선행이동을 통한 배치가 적합도 측면에서 낮은 값을 가지는 것을 확인할 수 있다.

Fig. 3.22.은 유전 알고리즘을 통해 배치 순서를 결정하였으며 선행 이동을 열 방향으로 지정하였고 배치 알고리즘의 목적함수는 f_y 와 f_{xy}의 합으로 설정하였다. 적합도 평가 값은 1.07757이다.

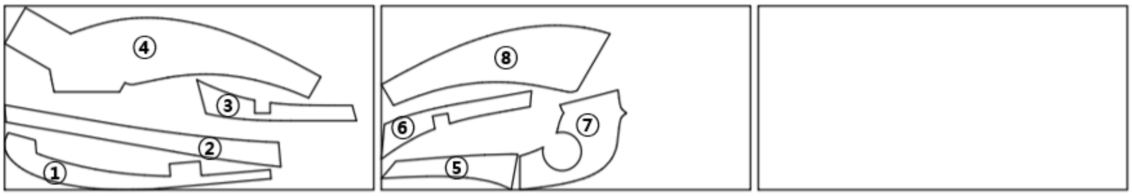

Fig. 3.22. Case 3 of Arrangement using genetic algorithm when the grid size is 25mm

Fig. 3.23.은 유전 알고리즘을 통해 배치 순서를 결정하였으며 선행이동 방향과 배치 알고리즘의 목적함수는 Fig. 3.22.와 동일하다. 적합도 평가 값은 1.05012이다.

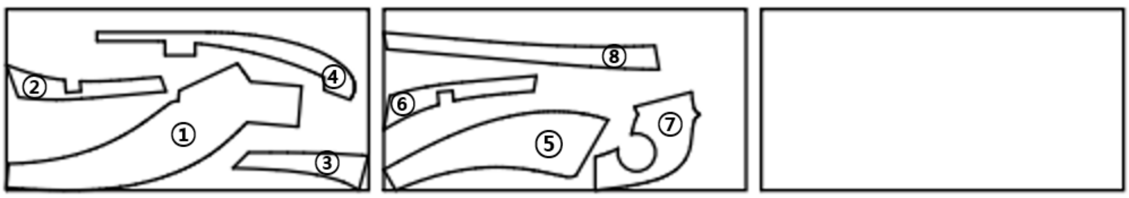

Fig. 3.23. Case 3 of Arrangement using genetic algorithm when the grid size is 25mm

Fig. 3.22와 Fig. 3.23의 배치 결과를 비교 했을 때, 격자사이즈를 25mm로 한 경우 남는 자재의 재사용성 측면에서 더 효과적으로 자재를 남긴다고 볼 수 있다. 자재의 재사용성 측면은 Fig. 3.24을 통해 설명할 수 있다. 각 판의 길이 방향 사용범위를 고려했을 때 (b)가 (a)보다 직사각형으로 남긴 넓이가 더 넓다 따라서 (b)가

남는 자재의 재사용성이 높다고 판단할 수 있다. (a), (b)는 동일한 배치 알고리즘과 유전 알고리즘을 사용했지만 격자 사이즈가 더 작을 때 더 큰 넓이를 남김을 알 수 있다.

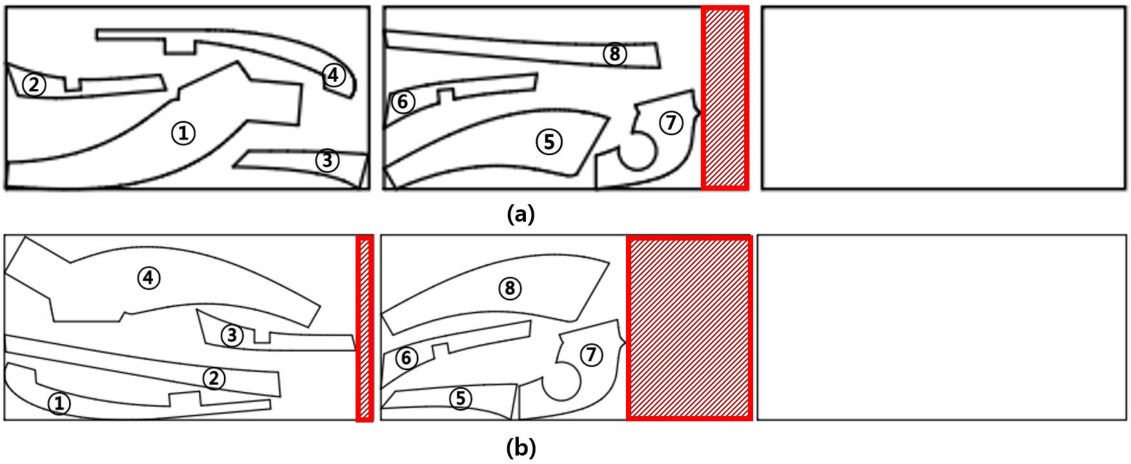

Fig. 3.24. Comparison of reusability when grid sizes are different

유전 알고리즘을 적용하여 순서를 결정하지 않고 종횡비 순(Fig. 3.16.), 넓이 순(Fig. 3.17.)으로 제안된 배치 알고리즘을 사용하였을 때 자재의 버림율이 유전 알고리즘을 사용했을 때(Fig. 3.18.)(Fig. 3.19.)(Fig. 3.20.)의 결과보다 높았다. 단순하게 배치순서를 결정했을 경우에 유전 알고리즘을 적용했을 때보다 빠르게 결과값을 얻을 수 있었지만 사용자가 순서와 각도를 결정하고 계산되었기 때문에 최적화 된 배치 결과를 얻을 수 없었다. 유전 알고리즘을 적용하여 순서를 결정한 경우 Case1(Fig. 3.18., Fig. 3.19.)일 때, 배치 알고리즘의 목적함수를 f_{xy}로 설정하였다. 또한 탐색 범위 내에서 배치 될 피스의 이동 방법을 열 방향으로 이동시키는 방법을 적용했는데, 남은 자재의 재사용적인 측면에서 봤을 때 적합한 결과 값을 얻지 못했다. Case2(Fig. 3.20., Fig. 3.21.)에서는 남은 자재의 재사용성 측면을 보안하기위해 이동 방법을 행 방향으로 이동시켜 결과 값을 얻었다. Case3(Fig. 3.22., Fig. 3.23)에서는 Case1, Case2과는 달리 목적함수를 f_y 와 f_{xy}의 합으로 설정하여 자재들이 왼쪽에 배치될 수 있도록 하였고 가장 적은 적합도 결과 값인 1.07757과 1.05012을 얻었다.

제4장 절단 경로 생성

본 연구에서 다루는 모형선 피스들의 기본적인 형상은 불규칙한 형태이다. 각 피스들은 선분과 호를 포함하는 폴리라인(Polyline)이다. 각 피스들에 대한 정보는 시계방향으로 나열되어 있으며, 레이저 토치의 절단 경로를 이루기 위하여 폐곡선 형태로 이루어져있다. 피스의 각 둘레를 형성하고 있는 선분의 시작점과 끝점, 그리고 호의 시작점과 끝점 좌표 데이터를 이용하여 G-code 형식으로 절단 경로를 생성할 수 있다. 4.1절에서는 본 연구에서 사용된 G-code에 대해 설명한다.

4.1 G-code

Table. 4.1. G-code list

G-code	Group	Property	Function
G00	01	modal	Rapid postioning
G01			Linear interpolation
G02			Circular interpolation CW
G03			Circular interpolation CCW

G-code란 단순 기계 또는 드릴링 기능과 같은 기계의 특정 동작을 명령할 때 사용되는 코드로써 모형선 피스 절단과정에서 사용되는 레이저 토치는 G-code형식에 따라 작동한다. Table. 4.1.에서 보이는 4가지 G코드를 사용해 절단 경로를 생성하였으며 G00은 원하는 좌표에 절단을 하지 않고 빠르게 이동하는 코드이다. G01은 직선가공 코드로써 절단을 하며 이동한다. 마지막으로 G02와 G03은 원호 가공 코드로써 시계 방향 혹은 반시계 방향으로 절단을 하며 이동한다.

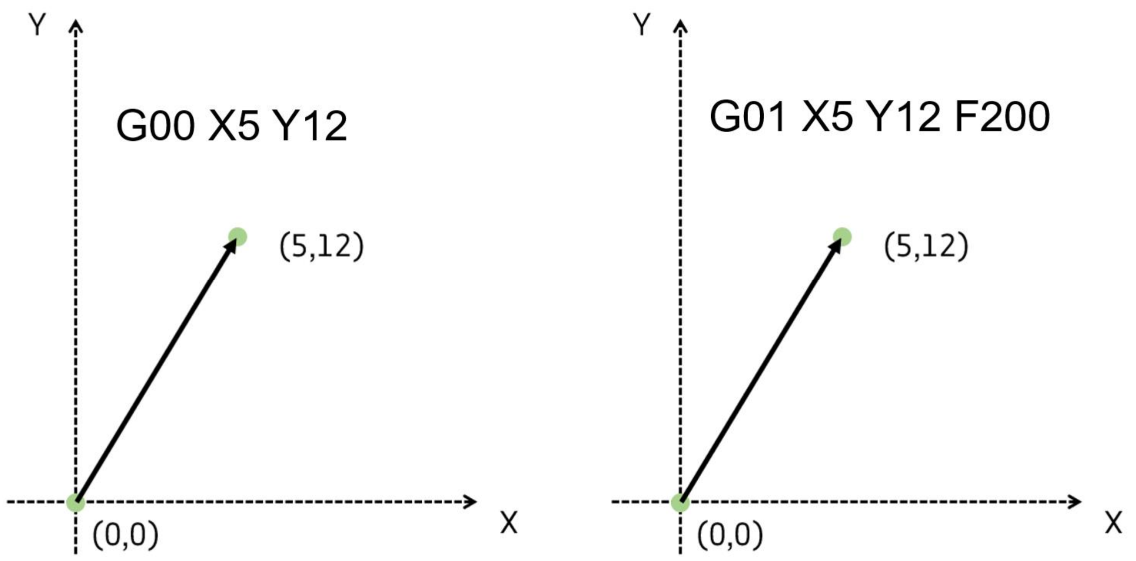

Fig. 4.1. Example for G00 and G01

Fig. 4.1.에서 왼쪽은 G00, 오른쪽은 G01 코드를 나타내고 그림으로 표현하였다. G00의 경우 원하는 X좌표와 Y좌표 값을 텍스트 형식으로 입력한다. G01도 마찬가지로 원하는 좌표 값만 입력하면 직선 가공을 진행할 수 있지만 좌표 값 뒤에 F200과 같이 이송속도를 추가하면 공구의 이동속도를 조절할 수 있다.

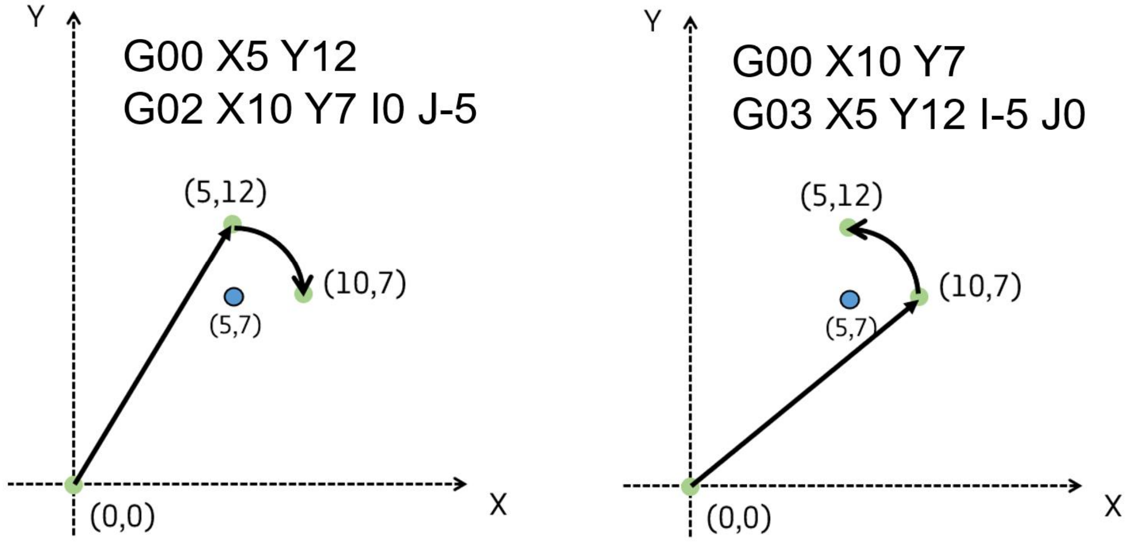

Fig. 4.2. Example for G02 and G03

Fig. 4.2.은 왼쪽은 시계방향 그리고 오른쪽은 반시계 방향 절단에 대한 G-code 예시이다. 시계방향 절단의 경우 공구가 직선 가공 후 (5,12)에 위치하고 있으며 호의 중심점이 (5,7)이고 (10,7)로 이동하고자 할 때 Fig. 4.2.의 왼쪽과 같이 G-code형식으로 작성할 수 있다. I와 J는 이전 공구 위치에서 호의 중심점 위치까지의 상대 좌표 값을 나타낸다. 반시계 방향도 마찬가지로 G-code 형식으로 작성할 수 있다.

4.2 피스의 절단 순서 결정

4.2.1 호의 볼록성 판별

본 연구에서 모형선 피스는 다각형(Polygon)형태가 아닌 호와 선분의 합으로 이루어진 형태(Polyline)으로 이루어져있다. 호가 존재하는 경우 볼록성을 판단하여야 적절한 G코드 형태로 표현할 수 있다.

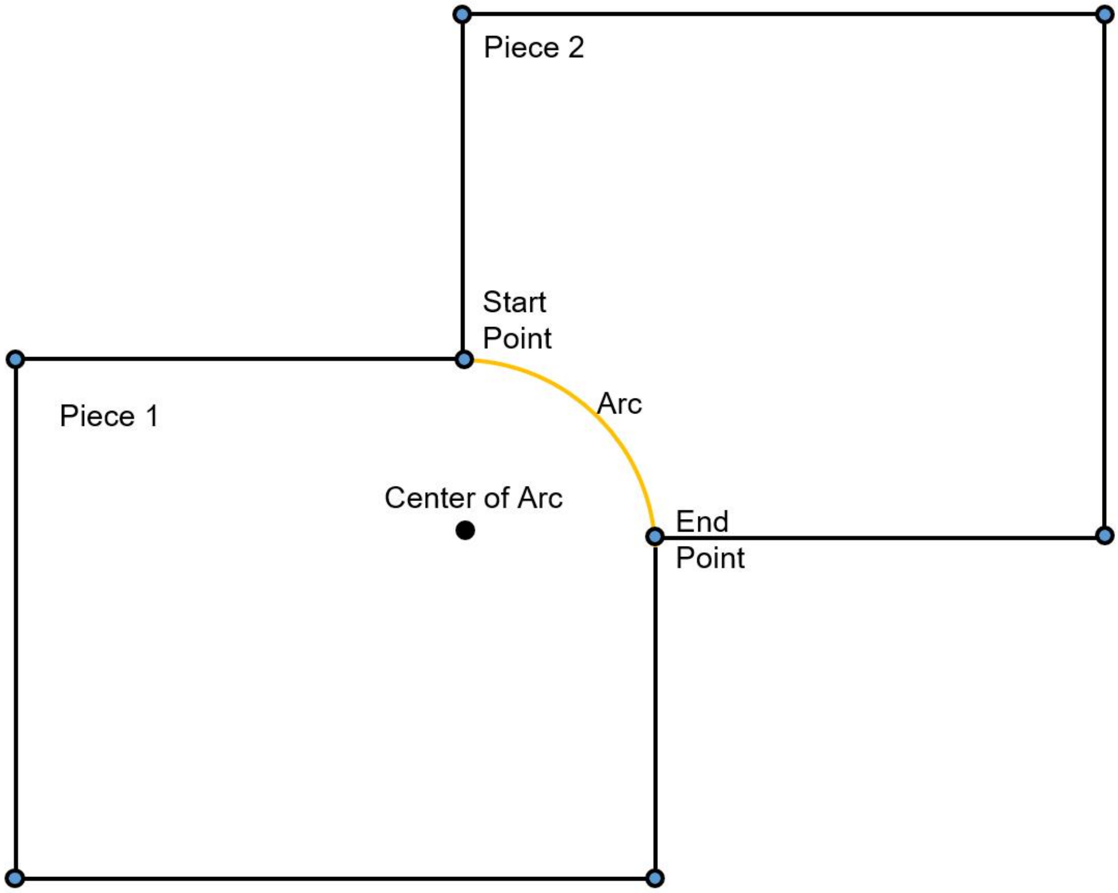

Fig. 4.3. Examples of reasons for judging convexity

Fig. 4.3.과 같이 Arc의 시작점, 끝점 그리고 중심점이 동일하지만 피스의 다른 점들의 위치에 따라서 볼록 혹은 오목일 수 있다. Piece 1에 대해서는 Arc가 볼록하므로 피스를 시계방향으로 절단한다고 가정하였을 때, Arc는 G02 코드를 사용하여 시계방향으로 절단하여야 한다. 하지만 Piece 2의 경우에는 Arc가 오목하므로 피스를 시계방향으로 절단한다고 가정하였을 때, G03코드를 사용하여 반시계 방향으로 절단하여야 한다. 따라서 G02 혹은 G03 코드를 적절하게 선택하는 방법을 아래에서 자세히 설명한다.

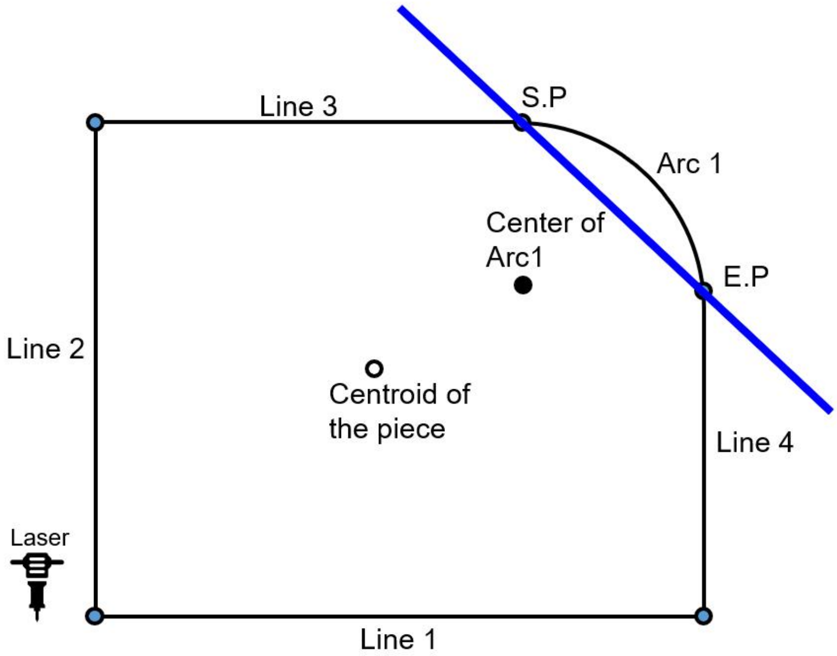

Fig. 4.4. Example when the arc of the piece is convex

Fig. 4.4.와 같이 볼록한 호가 피스에 존재하는 경우 호의 시작점과 끝점을 이용하여 음함수 형태로 표현할 수 있다. 음함수 형태로 표현된 직선의 방정식의 x,y좌표에 'Arc1'의 중심점과 피스의 중심점 값을 대입하였을 때 같은 부호의 값이 도출되면 볼록하다고 판단한다. 위의 그림에서 'Arc1'의 중심점과 피스의 중심점은 모두 직선의 아래쪽에 위치하므로 같은 부호 값으로 표현된다. 따라서 볼록하다고 판단한다. 이때 피스의 중심점은 주어진 선분과 호의 시작점과 끝점의 좌표를 합한 후 평균을 내어 계산한다.

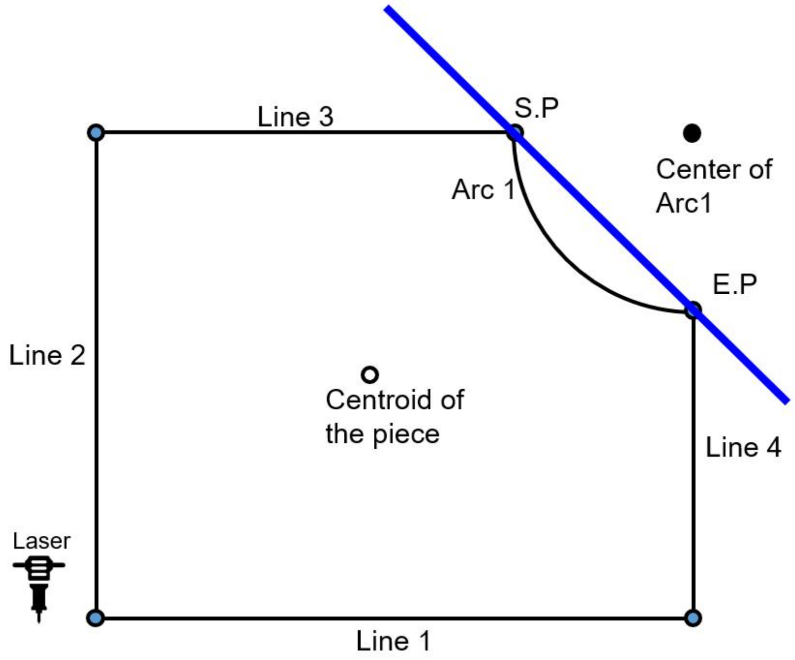

Fig. 4.5. Example when the arc of the piece is concave

Fig. 4.5.의 경우에는 반대의 경우로써 피스의 중심은 직선의 아래에 위치하고 'Arc1'의 중심점은 직선의 위쪽에 존재한다. 따라서 음함수로 표현된 직선의 방정식에 각각의 중심점 좌표 값들을 대입하면 다른 부호의 값이 나오므로 해당 피스에서 'Arc1'은 오목하다고 판단한다.

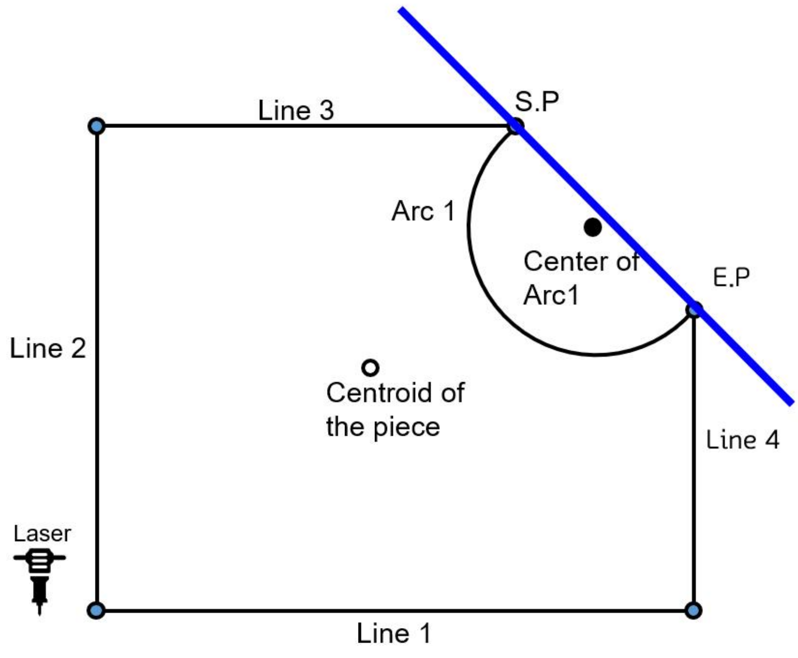

Fig. 4.6. Exception Case example when the arc of the piece is concave

예외의 경우로써 Fig. 4.5.과 다르게 Fig. 4.6.에서는 오목한 형상이지만 음함수로 표현된 직선의 방정식에 피스의 중심점 값과 아크의 중심점 값을 대입하게 되면 같은 부호의 값이 도출된다. 이는 아크의 시작점과 끝점의 각도가 180°보다 클 때 아크의 중심점이 직선의 아래쪽에 위치하게 되므로 같은 부호임에도 불구하고 오목한 형상이 표현된다. Fig. 4.7.와 같이 반대의 경우도 똑같이 적용된다.

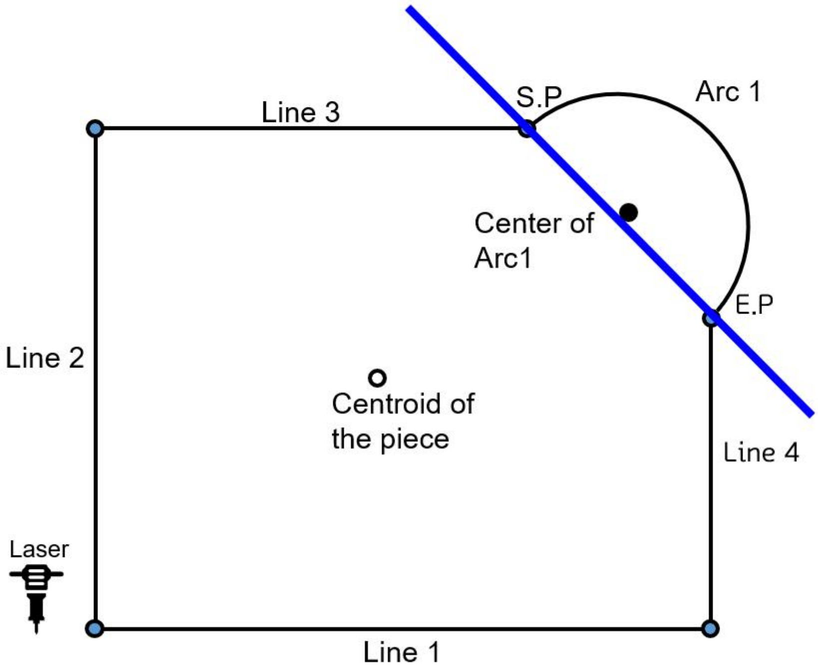

Fig. 4.7. Exception case example when the arc of the piece is convex

해당 경우들을 각도에 따라 분류하여 표로 표현하면 Table. 4.2.과 같다.

Table. 4.2. Determination of convexity according to arc angle

degree	Sign	Convex or Concave
less than 180°	Different Sign	Concave
	Same Sign	Convex
over 180°	Same Sign	Convex
	Different Sign	Concave

4.2.2 절단 경로 알고리즘

절단 경로를 생성하는 경우 유전알고리즘(GA)와 같이 메타 휴리스틱 계열 알고리즘을 이용하여 레이저 토치의 최소 이동경로에 대한 해를 구하는 것이 일반적이다. 하지만 최적화 과정을 거치게 될 경우 각 모형선 피스들의 점 데이터의 수가 많거나 모형선 피스의 수가 많아지게 되면 해에 대한 수렴속도와 반비례하는 결과를 얻게 된다. 따라서 본 논문에서는 메타휴리스틱 계열 알고리즘을 사용하지 않고 합리적인 절단 경로를 생성하는 알고리즘을 제시한다.

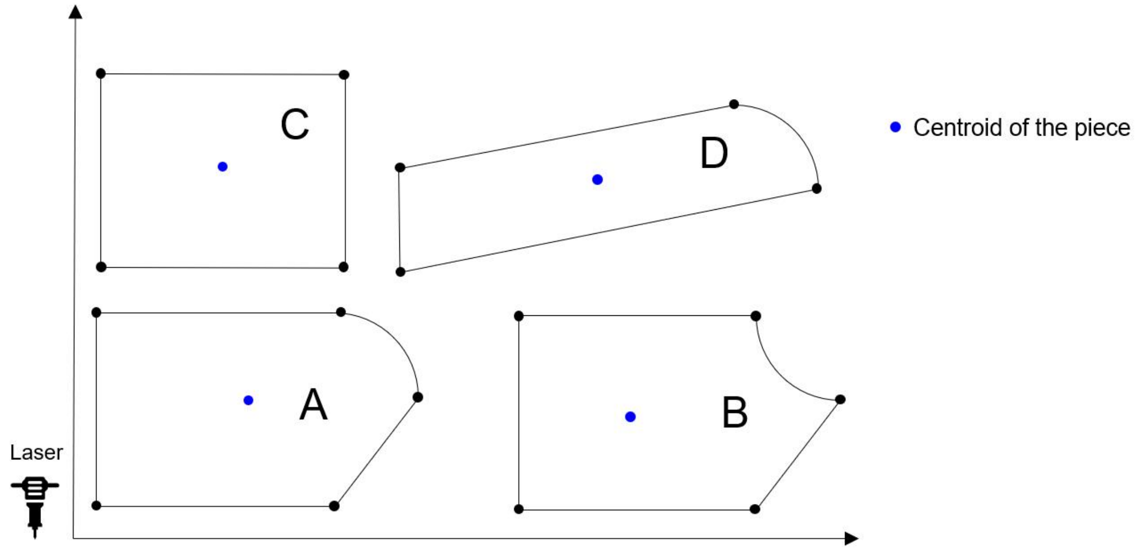

Fig. 4.8. Example of arrangement result including arc and line

Fig. 4.8.에서와 같이 먼저 각 피스들의 시작점과 끝점 데이터를 이용하여 평균값을 내어 중심점 좌표를 구한다.

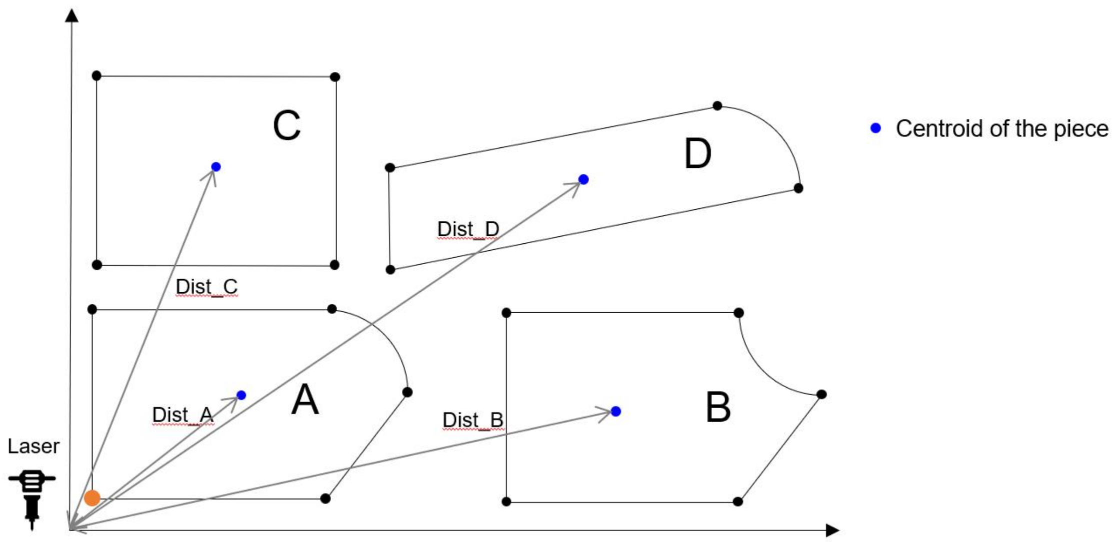

Fig. 4.9. Example of how to select the first piece

그리고 각 피스들의 중심점에서 원점까지의 거리를 계산하여 원점에서 거리가 가장 가까운 피스를 선택한다. Fig. 4.10.에서는 A가 선택된다. 피스 A가 지니고 있는 점 데이터들 중 원점과의 거리가 가장 가까운 점인 주황색 점을 선택하여 시계방향으로 절단을 완료한다.

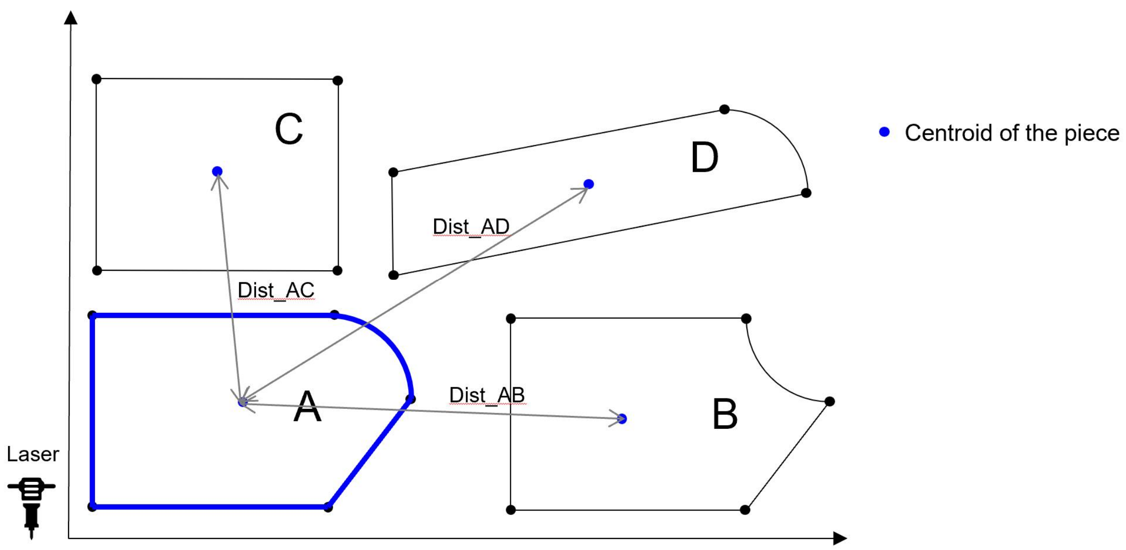

Fig. 4.10. Example of how to select the second piece

A를 시계방향으로 절단을 완료한 후, 아직 절단이 완료되지 않은 피스와 피스 A의 중심점 사이의 거리를 계산한다. 계산된 중심점 사이의 거리 중 가장 가까운 피스 C를 선택한다.(Fig. 4.11)

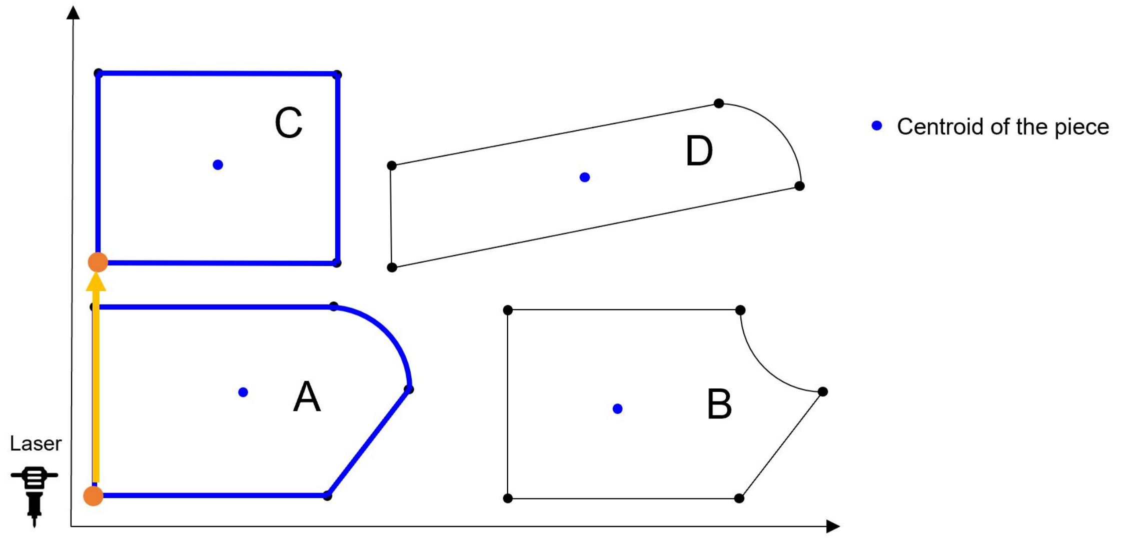

Fig. 4.11. method for calculating the shortest path between the first piece and the second piece

피스 C가 선택된 이후 레이저 토치가 위치한 지점에서 절단될 피스 C의 점 데이터 들 중 가장 가까운 거리를 계산한다. Fig. 4.11.과 같이 피스 A에서 피스 C로 가장 가까운 거리로 이동하게 된다. 이때 G-code의 G00 명령어를 통해 레이저 토치는 신속한 직선이동을 하게 된다.

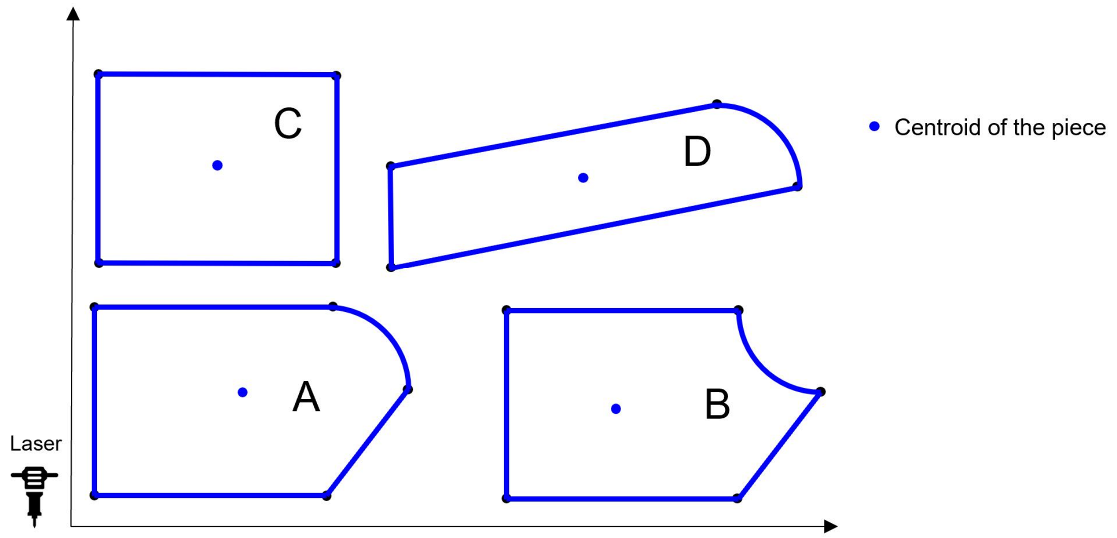

Fig. 4.12. Result of piece cutting completion

호를 포함하는 Polyline 형태의 피스 A,B,C,D의 절단을 완료할 수 있다. (Fig. 4.12.)

4.2.3 실제 모형선 피스에 대한 절단 경로 생성 알고리즘 검증

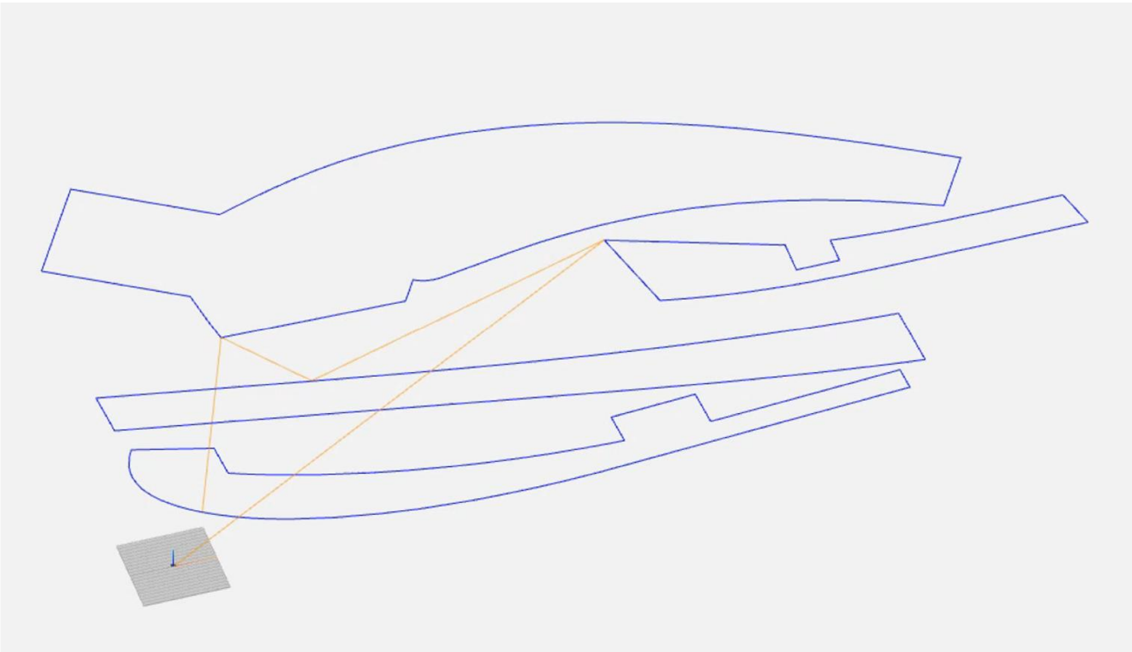

Fig. 4.13. Verification of cutting path generation algorithm of model ship pieces

실제 배치가 완료된 모형선 피스 데이터들을 이용하여 절단 경로 생성 알고리즘을 통해 절단 경로를 생성하였다.(Fig. 4.13.) 실제 모형선 점 데이터의 경우 일정한 간격을 두고 점 데이터가 존재 하지 않기 때문에 실제 피스 중심점의 위치가 다른 경우가 발생한다. 해당 문제를 해결하기 위해서 일정한 간격으로 선분과 호를 잘라 피스 외형선의 점 데이터를 확보하여 구하게 되면 피스 절단 순서를 알고리즘의 목적에 맞도록 구할 수 있다.

제5장 결론 및 고찰

본 연구는 기존의 수작업으로 진행되던 모형선 피스 배치 단계와 프로그램 숙련도를 요구하는 NC 코드 생성 단계를 자동화하기 위하여 연구를 진행하였다. 또한 배치 과정과 절단 경로 생성 과정을 모두 자동화하여 공수를 절감할 수 있다. 먼저 모형선 피스 배치의 경우 모형선 피스의 자재 버림율을 최소로 하는 효율적인 배치를 찾기 위해 배치 알고리즘과 유전 알고리즘을 사용하였다. 이외에도 넓이 순 배치, 종횡비 순 배치 등을 고려하였다. 큰 사이즈의 모형선 피스의 경우 원자재 목판의 크기와 유사하기 때문에 해당 피스가 먼저 배치되는 경우 단일 판에 다른 피스들이 배치되지 못하는 경우가 발생한다. 이러한 문제를 해결하기 위해 단일 판 사이에 '1'로 할당된 격자를 추가하여 다중 판을 구성하였다. 또한 다중 판 배치 문제에 알맞게 배치 알고리즘과 유전 알고리즘의 목적함수를 새롭게 구성하였다.

유전 알고리즘을 사용하지 않고 배치 알고리즘만 사용하여 피스들을 종횡비 순으로 배치한 경우 적합도 평가 값이 3.19381이고 판을 3개 사용하여 배치하였다. 또한 넓이 순으로 배치를 진행한 경우 적합도 평가 값이 1.62284이고 판을 2개 사용하여 종횡비 순 배치보다 나은 결과 값을 얻었지만 남은 자재의 재사용성 측면에서 낮은 결과를 얻었다.

격자 사이즈를 25mm와 50mm인 두 경우를 계산하였으며 유전 알고리즘을 사용하고 배치 알고리즘의 목적함수를 f_{xy}만 사용한 경우에는 2가지 케이스로 분류하였다. 첫 번째는 열 방향을 선행해서 이동 탐색하는 경우 적합도 평가 값이 각각 1.27155과 1.17361이고 판은 2개를 사용하였다. 두 번째는 행 방향을 선행하여 이동 탐색하는 경우 적합도 평가 값이 1.10652과 1.07411이고 판은 2개를 사용하였다. 격자 사이즈가 25mm 일 때 두 경우 모두 넓이 순, 종횡비 순 배치 결과보다 나은 적합도 평가 값이 도출되었다. 하지만 첫 번째 케이스는 원자재의 남김이 윗 방향이고 두 번째 케이스는 원자재의 남김이 오른쪽 방향으로써 두 번째 케이스가 길이 방향으로의 원자재 남김율 측면에서 더 크므로 재사용성 측면에서 효과적이라는 결과를 얻었다. 이러한 결과를 바탕으로 배치 알고리즘의 목적함수를 f_{xy}, f_y 로 구성하였고 행 방향을 선행하여 이동 탐색하는 알고리즘을 구성하였다. 구현 결과는 적합도 평가 값은 1.07757과 1.5012이고 판은 2개를 사용하였다. 이때 원자재의 재사용 측면에서 3가지 케이스들 중 가장 유리하다.

모형선의 네스팅의 경우 일정한 waterline에서 일정한 간격으로 잘린 피스들의 배치 결과를 한번에 얻어야 한다. 또한, 피스들의 크기가 원자재 판의 크기와 비교했을 때 크기가 비교적 크기 때문에 다중 판에 피스들을 효과적으로 배치 할 수 있는 알고리즘의 적용이 필수적이다. 본 연구는 다중 판에 피스들을 배치할 때 효과적인 네스팅 방법을 제시한다. 유전알고리즘을 이용해 배치순서를 무작위로 선택하고 배치 알고리즘을 적용하여 자재 버림율을 최소화는 배치 결과를 얻었다. 그리고 도출된 배치 결과를 이용하여 모형선 피스의 절단 경로를 생성하였다. 절단 경로는 공

구의 절단 경로를 최소화하기 위해서 원점에서 가장 가까운 위치의 피스부터 시작한다. 이후 절단된 피스에서 가장 가까운 피스를 선택하는 방식으로 절단 경로 생성 알고리즘이 진행된다. 또한 피스 간 이동시 최단거리로 이동하기 위하여 현재위치에서 다음 절단될 피스의 점 데이터들 중 가장 가까운 점을 선택할 수 있도록 하였다. 실제 배치 결과에 대한 피스 데이터를 알고리즘에 적용하여 검증하였다.

향후 연구로는 모형선 피스의 강도에 영향을 미치는 목재 판의 결을 고려하여 최적 배치를 계산하는 방법이 있다. 또한 격자 사이즈에 대해 최적화를 통해 시간 단축 문제를 해결하고자 한다. 뿐만 아니라 이동 경로 측면에서 NC머신 공구의 경로 최적화를 통해서 공구가 최적의 경로로 움직일 수 있도록 할 것이며 추후 제시하는 방법을 이용하여 모형선 피스 생성과정의 자동화 뿐 아니라 조선소에서 사용되는 철판 네스팅으로 확장 할 수 있을 것이다.

참고문헌

[1] Guo, Baosu, et al. 2020. "Automatic layout of 2D free-form shapes based on geometric similarity feature searching and fuzzy matching." Journal of Manufacturing Systems 56 : 37-49.
[2] Adamowicz, M., & Albano, A. 1976. A solution of the rectangular cutting-stock problem. IEEE Transactions on Systems, Man, and Cybernetics, (4), 302-310.
[3] Dori, D., & Ben-Bassat, M. 1984. Efficient nesting of congruent convex figures. Communications of the ACM, 27(3), 228-235.
[4] Adamowicz, M., & Albano, A. 1976. Nesting two-dimensional shapes in rectangular modules. Computer-Aided Design, 8(1), 27-33.
[5] Albano, A. 1977. A method to improve two-dimensional layout. Computer-Aided Design, 9(1), 48-52.
[6] A. R. Babu and N. R. Babu, 2001. "A generic approach for nesting of 2-D parts in 2-D sheets using genetic and heuristic algorithms," Computer-Aided Design, vol. 33, pp. 879-891,
[7]Konopasek, M. 1981. Mathematical treatments of some apparel marking and cutting problems. US Department of Commerce Report, 99(26), 90857-10.
problems," US Department of Commerce Report, vol. 99, pp. 90857-10, 1981.
[8]Y. Stoyan, G. Scheithauer, N. Gil, and T. Romanova, 2004. Phi-functions for complex 2D-objects, Quarterly Journal of the Belgian, French and Italian Operations Research Societies, vol. 2, pp. 69-84,
[9] 한윤근. 2000. 임의 형상 부재의 자동 네스팅 시스템에 관한 연구. 박사학위논문, 서울대학교.
[10] 강현구. 1998. 불규칙한 선박용 부재의 최적배치. 전남대학교 산업공학과 석사논문.
[11] Kim, Y., Gotoh, K. and Toyosada, M., 2003. “Automatic twodimensional layout using a rule-based heuristic algorithm”,Journal of Marine Science and Technology,Vol.8, pp. 37-46,
[12]Weng, W.C. and Kuo, H.C. 2011. “Irregular Stock Cutting System Based on AutoCAD”, Advances in Engineering Software, Vol 42, pp 634-643
[13]Liu Q, Zeng J, Zhang H, Wei L . 2020. A heuristic for the twodimensional irregular bin packing problem with limited rotations. In: Fujita H, Fournier-Viger P, Ali M, Sasaki J (eds.) Trends in artificial intelligence theory and applications. Artificial intelligence practices, pp 268-279. Springer
[14] Han, G.C. and Na S.J., 1996. Two-stage approach for nesting in

two-dimensional cutting problems using neural network and simulated annealing Proceedings of the Institution of Mechanical Engineers, Journal of Engineering manufacture, Vol.210, pp.509-519,
[15]Luo Q, Rao Y, Peng D. 2022. Ga and gwo algorithm for the special bin packing problem encountered in field of aircraft arrangement. Applied Soft Comput, 114, 108060.
[16]Sheen, D. M. 2012. Nesting expert system using heuristic search. Journal of Ocean Engineering and Technology, 26(4), 8-14.
[17]한윤근, & 장창두. 2000. 개선된 유전자 알고리즘을 이용한 부재 절단경로 최적화에 관한 연구. 대한조선학회 논문집, 37(3), 90-98.
[18]박주용, & 서정진. 2009. 유전자 알고리즘을 이용한 부재 절단 경로 최적화. 한국해양공학회지, 23(6), 67-70.
[19]Al-Sahib, N. K. A., & Abdulrazzaq, H. F. 2014. Tool path optimization of drilling sequence in CNC machine using genetic algorithm. Constraints (2a), 2, 2b.
[20]Lee, M. K., & Kwon, K. B. 2006. Cutting path optimization in CNC cutting processes using a two-step genetic algorithm. International Journal of Production Research, 44(24), 5307-5326.
[21]Hajad, M., Tangwarodomnukun, V., Jaturanonda, C., & Dumkum, C. 2019. Laser cutting path optimization using simulated annealing with an adaptive large neighborhood search. The International Journal of Advanced Manufacturing Technology, 103, 781-792.
[22]방기범. 1989. 판재 소모를 최소화하는 이차원형상의 최적 배치, 서울대학교 기계설계학과 대학원 박사학위논문, pp3-15.
[23]Mander, U. and Israni, S., 1984, “Piece Point Minimization and Optimal Torch Path Determination in Flame Cutting,” Journal of Manufacturing Systems, Vol.3, No. 1, pp.81-89,
[24]조경호, 이건우, 1993, “풀림모사 기법을 이용한 NC 터릿 작업에서의 공구경로 최적화”, 대한기계학회 논문집, 17(5), pp. 1183-1192,
[25]한국찬, & 나석주. (1996). 레이저 절단공정에서의 토치경로 생성에 관한 연구. 대한기계학회 논문집, 20(6), 1827-1835.
[26]Silvela, Jaime, and Javier Portillo., 2001, "Breadth-first search and its application to image processing problems.", *IEEE Transactions on Image Processing* 10.8 , 1194-1199.
[27] E. K. Burke, R. S. Hellier, G. Kendall, and G. Whitwell, 2007. Complete and robust no-fit polygon generation for the irregular stock cutting problem, European Journal of Operational Research, vol. 179, pp. 27-49,
[28] Bresenham, J.E., 1965. Algorithm for Computer Control of a Digital

Plotter. IBM System Journal, 4(1), 25-30
[29] Goldberg D.E., 1989. “Genetic Algorithms in Search, Optimization and Machine Learning (1st. ed.)”, Addison-Wesley Longman Publishing Co., Inc., USA.
[30] Konak A., Coit D.W., Smith A.E., 2006. “Multi-objective optimization using genetic algorithms: A tutorial”, Reliability Engineering & System Safety, Vol. 91, Issue 9, Pages 992-1007, ISSN 0951-8320,
[31]Kim, D. S., & Lee, C. S. 2009. Determination of an optimal sequence for nesting with heuristic algorithm. In CAD/CAM Conference.
[32] Lee, H,B,. & Ruy, W,S. 2013. Determination of Nesting Algorithm Fitness Function through Various Numerical Experiments. Journal of Ocean Engineering and Technology, 27(5), 28-35.