Optimal Pipe Routing Techniques in an Obstacle-Free 3D Space

Marvin Stanczak[1,2], Cédric Pralet[1], Vincent Vidal[1] and Vincent Baudoui[2]
[1]*ONERA/DTIS, Université de Toulouse, F-31055 Toulouse, France*
[2]*Airbus Defence and Space, Toulouse, France*

Keywords: Pipe Routing, Automation, Optimization, Constraints.

Abstract: This paper introduces two approaches to automate the design of pipe-systems in a continuous 3D space to assist designers to deal with the high number of constraints of the pipe routing problem. This work assimilates a pipe to a succession of straight sections and bends in which possible bends are restricted to a bend catalog containing orthogonal and non orthogonal bends. As a preliminary work, both methods ignore obstacles but take into account pipe section orientations to deal with asymmetrical pipe sections. To do so, a graph of reachable pipe orientations is generated from the bend catalog. Then, the first approach enumerates valid bend combinations using the previous graph and computes the optimal length of straight sections for each combination using a Linear Program. The second one formulates the pipe routing problem as a Mixed Integer Linear Programming problem based on enumerated reachable orientations. Both approaches are tested on realistic scenarios inspired by industrial problems.

1 INTRODUCTION

For some industries, the design of piping-systems plays an important role because of the large number of pipes to be routed, like in aircraft or spacecraft design, shipbuilding, circuit layout or plant layout. It is an highly time-consuming phase in industry. Traditionally, pipe designers manually define routes and have to deal with a lot of constraints. For these reasons, many researches have been conducted to automate the pipe routing process in order to save time and money during the design phase.

In practice, pipe designers build a route as a succession of rigid straight sections and rigid bends using their expertise to choose the best routing spaces and save length and bends (see Figure 1). The different bends they can use are defined by catalogs from pipe manufacturers. Various kinds of bends are often available, not just orthogonal ones (a bend is said to be orthogonal when the direction of the pipe before the bend is orthogonal to the direction of the pipe after the bend). Historically, the classic pipe routing algorithm, called "Maze algorithm", was introduced by Lee (Lee, 1961). In this approach, the routing environment is discretized with a regular grid and the cells which are occupied by obstacles are labeled. Then, starting from the source cell, the best route is computed by exploring the cells' neighborhood until the target is reached. This kind of cell decomposition approach was improved using different algorithms which require less memory space like Ant Colony Optimization (Fan et al., 2006; Jiang et al., 2015), Genetic Algorithm (Ito, 1999; Kimura, 2011) or Particle Swarm Optimization (Asmara and Nienhuis, 2006). Recently, Belov also introduced a Constraint Programming formulation (Belov et al., 2017). However, cell decomposition is not adapted to non orthogonal bends, even if Ando proposed to introduce additional edges in the cell adjacency graph to model complex bends (Ando and Kimura, 2012). It is also possible to build pipes by adding elements one by one from a catalog containing bends and straight sections

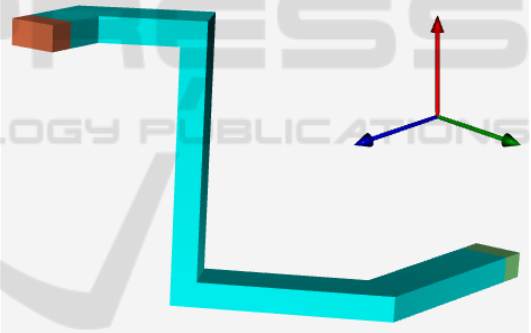


Figure 1: Example of a routed pipe.

Stanczak, M., Pralet, C., Vidal, V. and Baudoui, V.
Optimal Pipe Routing Techniques in an Obstacle-Free 3D Space.
DOI: 10.5220/0008923300690079
In *Proceedings of the 9th International Conference on Operations Research and Enterprise Systems (ICORES 2020)*, pages 69-79
ISBN: 978-989-758-396-4; ISSN: 2184-4372

(Furuholmen et al., 2010), but this approach does not guarantee to find a feasible solution even if there exists one in a continuous space. Indeed, some points of the space are unreachable from a given catalog of pipe elements.

Other methods, called Skeleton approaches, build a graph of possible route candidates using rules inspired by experience. This way, the pipe routing problem can be modeled as a shortest path problem in a graph, solved using the Dikjstra algorithm (S.-H. et al., 2013) or Mixed Linear Integer Programming (Guirardello and Swaney, 2005). A more recent research also explored Evolutionary Algorithms considering multi-objective routing in a skeleton graph (Liu and Liu, 2018).

While cell decomposition approaches discretize the environment and the pipes, skeleton-based algorithms reduce the possible solutions to a finite set. In both cases, these methods lead to sub-optimal solutions in the continuous space. To improve the quality of the solutions, Zhu uses cell decomposition to define a routing channel and then computes the detailed route using a projection procedure (Zhu and Latombe, 1991). Park proposes a method that generates routing cells adapted to the environment and uses patterns inside these cells (Park and Storch, 2002).

Another way to build a route in a continuous environment is to extend lines from the origin point using a set of directions and repeat this extension each time a line intersects an obstacle until the destination is reached. This approach, called "Escape Algorithm" or "Line Search Algorithm", was introduced by Hightower (Hightower, 1969) and allows to deal with non orthogonal bends. But such methods do not take into account the orientation of the pipe sections. Indeed, some types of pipes can have centerline unsymmetrical sections such as rectangular sections. In this case the route must reach the destination with the right orientation, that is with the right angular position of the section around the centerline. This is not possible with line search methods which only consider the centerline.

Last, some parametric-based models were also proposed using adaptable pipe patterns. In this case, the pipe route can be optimized using Mathematical Programming (Sakti et al., 2016; Medjdoub and Bi, 2018) or Genetic Algorithms (Ikehira et al., 2005). However, existing models do not consider bend catalogs with non orthogonal bends.

This paper introduces two pipe routing algorithms which respond to the following three challenges:

- ensure solution optimality in a 3D continuous routing space; optimality is evaluated here with regards to the total cost of the pipe, given by the sum of the costs of each individual bend and each individual straight section used (more details later on these points),
- use non orthogonal bends from a catalog,
- take into account unsymmetrical pipe sections.

For this purpose, as a first necessary step for future works, a simplified version of the Pipe Routing Problem in 3D space (3D-PRP) is considered: the 3D Obstacle-Free Pipe Routing Problem (3D-OFPRP) in which no obstacles are taken into account. Note that in the obstacle-free problem, we consider both that there is no external obstacle and that the pipe cannot be an obstacle for itself. The relaxation of these two assumptions are left for future works.

The first proposed approach, presented in Section 3, decomposes the problem into a shortest path problem in an oriented graph and a Linear Programming (LP) problem. The second one, presented in Section 4, is based on a Mixed Integer Linear Programming (MILP) model. Both approaches require geometric notations introduced in Section 2 to make the linear formulations possible. Section 5 describes experimental results for both methods. Last, Section 6 gives perspectives on the way obstacles can be taken into account.

2 PROBLEM DEFINITION

2.1 Geometric Preliminaries

In what follows, a pipe is described as a succession of straight sections and bends. This representation is formalized in this section.

2.1.1 Local Pipe Configurations

We consider a 3D Cartesian coordinates system defined by its origin point and its standard orthonormal basis $(\overrightarrow{e_x}, \overrightarrow{e_y}, \overrightarrow{e_z})$ where $\overrightarrow{e_x} = (1,0,0)$, $\overrightarrow{e_y} = (0,1,0)$, and $\overrightarrow{e_z} = (0,0,1)$. In this system, the vector of coordinates of a point P are referred to as $\overrightarrow{p} = (P_x, P_y, P_z)$ and the coordinates of a vector are referred to as $\overrightarrow{v} = (v_x, v_y, v_z)$.

For a pipe with any section and any point P of the centerline of the pipe, it is possible to define the *local configuration* of the pipe at point P as the pair $\theta = (P,R)$ where R is an orthonormal basis $(\overrightarrow{x_R}, \overrightarrow{y_R}, \overrightarrow{z_R})$ such that (1) vector $\overrightarrow{z_R}$ is collinear with the centerline and (2) the intersection between the volume of the pipe and plane $(P, \overrightarrow{x_R}, \overrightarrow{y_R})$ matches exactly with the section of the pipe (see Figure 2). Equivalently, R corresponds to a rotation matrix which defines how

to transform the orthonormal basis of the 3D coordinates system into the orthonormal basis associated with the local configuration.

In the following, R is called the *orientation* of local configuration θ and the set of pipes' local configurations is referred to as Θ. Note that we consider here that there is a unique shape for the pipe section all along the pipe (for example, not possible to transform a square pipe section into a rectangular pipe section).

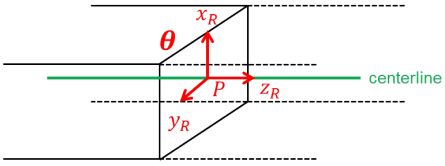


Figure 2: A local pipe configuration.

For any part of a pipe, the input configuration (respectively the output configuration) is the local configuration associated with the first end section (respectively the last end section) of the pipe part.

2.1.2 Straight Sections

With the previous notations, the output configuration of a straight section of a pipe of length $L_s \in \mathbb{R}^+$ applied from the local configuration $\theta = (P,R) \in \Theta$ is the local configuration $\theta' = (P',R)$ where P' is the point P translated of the distance L_s along vector $\overrightarrow{z_R}$ (see Figure 3). Thus, a straight section can be represented by an application s from Θ into Θ defined by a length L_s. $\mathcal{S}$ refers to the set of straight sections of pipes. The cost C_s of a straight section $s \in \mathcal{S}$ is αL_s where α is a constant linear cost.

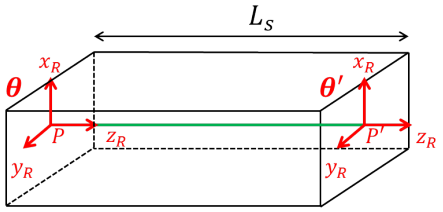


Figure 3: A straight section.

2.1.3 Bends

As for straight sections, the output configuration of a bend of rotation matrix R_b and half length $L_b \in \mathbb{R}^+$ applied from the local configuration $\theta = (P,R) \in \Theta$ is the local configuration $\theta' = (P',R')$ where $R' = RR_b$ and P' is the point P translated of the distance L_b along vector $\overrightarrow{z_R}$ and then translated of the distance L_b along vector $\overrightarrow{z_{R'}}$ (see Figure 4). Rotation matrix R_b expresses the orientation of the output configuration of the bend in the basis associated with its input configuration.

Thus, a bend can be represented by an application b from Θ into Θ defined by a rotation matrix R_b and a half length L_b. By extension, a bend b also transforms an input orientation o into an output orientation o' referred to as $o' = b(o)$. In the following, $\mathcal{B}$ refers to the set of bends. The definitions provided are also valid for bends leading to a twisted pipe section between two straight sections.

The cost of a bend $b \in \mathcal{B}$ is referred to as $C_b \in \mathbb{R}^+$.

For each bend $b \in \mathcal{B}$ applied from configuration $\theta = (P,R) \in \Theta$, the transition configuration is the local configuration of the pipe $T_b = (P_{T_b}, R_{T_b})$ where $R_{T_b} = RR_b$ and P_{T_b} is the point P translated of the distance L_b along vector $\overrightarrow{z_R}$ (see Figure 4). In other words, the transition configuration of bend b is the local configuration with the break point of the centerline as origin and the output orientation of the bend as orientation.

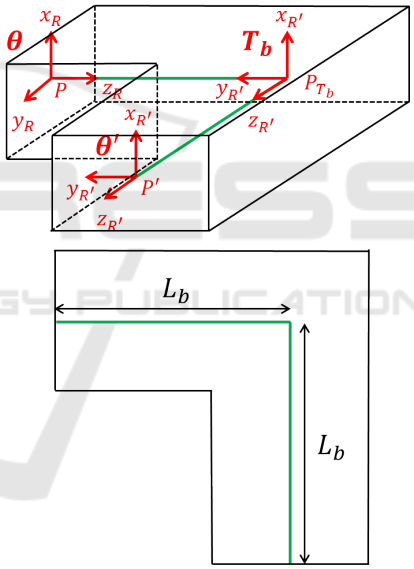


Figure 4: A bend b and its transition configuration T_b.

Note that in practice, a bend is not a discontinuous change of the centerline direction. The centerline follows a circular arc depending on the angle of the bend and on the bend radius. But the introduced representation guarantees that, between two transition configurations, the centerline section is a segment of $\mathbb{R}^3$, which is used in what follows.

2.1.4 Bend Combinations

A bend combination B is a list $[b_1,...,b_{N_B}]$ of bends in $\mathcal{B}$. N_B refers to the number of bends of combination B. By extension, B also transforms an input

orientation o into an output orientation $B(o)$ defined by:

$$B(o) = b_{N_B} \circ ... \circ b_1(o) \quad (1)$$

The cost of a bend combination B is defined by:

$$C_B = \sum_{i=1}^{N_B} C_{b_i} \quad (2)$$

2.1.5 Pipes

With the previous notations, a pipe Π with $N_\Pi \in \mathbb{N}$ bends can be described by a list $[s_1, b_1, s_2, ..., s_{N_\Pi}, b_{N_\Pi}, s_{N_\Pi+1}]$ alternating $N_\Pi + 1$ straight sections $s_i \in \mathcal{S}$ and N_Π bends $b_i \in \mathcal{B}$. Thus, any pipe can be represented by an application Π from Θ into Θ that verifies the following conditions:

$$\Pi(\theta) = s_{N_\Pi+1} \circ b_{N_\Pi} \circ s_{N_\Pi} \circ ... \circ s_2 \circ b_1 \circ s_1(\theta) \quad (3)$$

$\mathcal{P}$ refers to the set of pipes. For $N \in \mathbb{N}$, $\mathcal{P}(N)$ refers to the set of pipes which contain exactly N bends and $\mathcal{P}_{max}(N)$ refers to the set of pipes which contain at most N bends.

The cost $C_\Pi \in \mathbb{R}^+$ of such a pipe is defined by the cost of its bends and straight sections:

$$C_\Pi = \sum_{i=1}^{N_\Pi} C_{b_i} + \sum_{i=1}^{N_\Pi+1} \alpha L_{s_i} \quad (4)$$

If the input configuration θ and the output configuration $\Pi(\theta)$ of a pipe $\Pi \in \mathcal{P}$ are considered as transition points, the centerline of Π can be described as a sequence of transition configurations:

$$\Pi = [T_0, ..., T_{N_\Pi+1}] \quad (5)$$

with

$$T_0 = \theta,$$
$$T_{N_\Pi+1} = \Pi(\theta),$$
$$\text{and } \forall i \in \{1, ..., N_\Pi\}, \ T_i = T_{b_i}$$

2.2 Constraints

The 3D-OFPRP can be subject to various kinds of constraints depending on the application field (Park and Storch, 2002). This section enumerates the constraints taken into account in our study.

2.2.1 Routing Space

A pipe should be routed within a finite subset $\mathcal{A}$ of $\mathbb{R}^3$ called the routing space. In what follows, it is assumed that $\mathcal{A}$ is a convex subset of $\mathbb{R}^3$ described by a set of inequalities:

$$\forall i \in \{1, ..., k_\mathcal{A}\} \ \ A_i x + B_i y + C_i z + D_i \leq 0 \quad (6)$$

So, because of the convexity of $\mathcal{A}$, the centerline of the pipe $\Pi \in \mathcal{P}$ is contained in the routing space if and only if all the transition configurations of Π are in the routing space.

2.2.2 Catalog of Bends

Pipe designers can be led to use a restricted set of bends in order to reduce production costs. This set of possible bends is referred to as $\mathcal{B}_{cat} \subseteq \mathcal{B}$. It contains $N_{\mathcal{B}_{cat}}$ bends.

2.2.3 Pipe Attachability Constraints

In practice, the way pipes are routed is subject to physical constraints. For instance, pipes are supported by brackets which are fitted to fixed planar structures referred to as walls. $\mathcal{W}$ is the set of walls. A pipe configuration $\theta = (P, R) \in \Theta$ is attachable to a wall $w \in \mathcal{W}$ of normal $\overrightarrow{n_w}$ if and only if:

$$\overrightarrow{x_R} \cdot \overrightarrow{n_w} = 0 \ \vee \ \overrightarrow{y_R} \cdot \overrightarrow{n_w} = 0 \quad (7)$$

This means that, for a pipe whose section is rectangular, at least one edge of the section is parallel to wall w. This way, a bracket can be used to fix the pipe at the point corresponding to pipe configuration $\theta = (P, R)$. Also, the actual distance between the pipe and the wall to which it is attached can be limited just by restricting the routing space.

By extension, a pipe $\Pi \in \mathcal{P}$ is attachable to $\mathcal{W}$ if and only if all the transition configurations of Π are attachable to at least one wall $w \in \mathcal{W}$. So, as a pipe configuration constraint, the routed pipe should be attachable to $\mathcal{W}$.

2.2.4 Minimal Length of Straight Sections

In order to respect the pipe manufacturing constraints, each straight section of a pipe $\Pi \in \mathcal{P}$ must have a minimal length L_{min}.

2.3 Goal

For a maximum number of bends $N_{max} \in \mathbb{N}$, the 3D-OFPRP consists in connecting a source configuration $\theta_s = (P_s, R_s) \in \Theta$ to a destination configuration $\theta_d = (P_d, R_d) \in \Theta$ using a pipe $\Pi \in \mathcal{P}_{max}(N_{max})$ minimizing the pipe cost C_Π and respecting the previous constraints.

Furthermore, for $n \in \{0, ..., N_{max}\}$, the 3D-OFPRP($n$) consists in the same problem but with pipes $\Pi \in \mathcal{P}(n)$ containing exactly n bends. This way, the optimal solution of the 3D-OFPRP is the best solution among the optimal solutions of the set of 3D-OFPRP(n) for n from 0 to N_{max}.

3 COMBINED GRAPH-LP SOLVER

This section introduces a formulation of the 3D-OFPRP which uses a graph structure representing all pipe orientations that can be reached from the source orientation with at most N_{max} bends of catalog $\mathcal{B}_{cat}$.

3.1 Preprocessing: Reachable Orientation Graph

To enumerate the reachable orientations, an oriented graph $G(O,E)$ is built iteratively using the reachable orientations as vertices and the bends that allow to reach an orientation from another one as arcs.

The generation procedure of the graph starts by adding the source orientation R_s to $G(O,E)$. Then, the graph is extended by applying each bend b of catalog $\mathcal{B}_{cat}$ to each node $o \in O$ added at the previous step. If the output orientation $o' = b(o)$ is not already in O, then node o' is added to O. Moreover, arc (o,b,o') is added to E. Furthermore, the attachable configuration constraints are taken into account at this step by filtering the reachable orientations that violate them. This reduces the bend combinations that will be explored. By extending graph $G(O,E)$ N_{max} times, all valid orientations that can be reached with at most N_{max} bends of $\mathcal{B}_{cat}$ are enumerated in O.

The 3D-OFPRP can have a solution only if the destination orientation R_d is in the vertices of $G(O,E)$, that is to say only if $R_d \in O$.

The reachable orientation graph $G(O,E)$ contains in the worst case $O\left((N_{\mathcal{B}_{cat}})^{N_{max}}\right)$ nodes. Filtering the reachable orientations with the pipe attachability constraints reduces the complexity of the enumeration.

In the following, N_O (respectively N_E) refers to the number of vertices (respectively the number of arcs) in $G(O,E)$.

3.2 Problem Decomposition

To solve the 3D-OFPRP, we use a decomposition approach that exploits two solvers, as shown in Figure 5. The first solver, called *combination enumerator*, enumerates the possible bend combinations B_{sol} containing $N_{B_{sol}} \in \{0,...,N_{max}\}$ bends of catalog $\mathcal{B}_{cat}$ and ensuring that the destination configuration can be reached from the source configuration. Combination B_{sol} can also be the empty combination here (case $N_{B_{sol}} = 0$). The second solver, called *length optimizer*, takes as an input a bend combination B_{sol} produced by the combination enumerator and computes the optimal route Π_{sol} for B_{sol}. This optimal route is computed through a linear program that finds the optimal length of straight sections between each pair of bends. This way, by iterating between the two solvers, it is possible to compute the optimal pipe route.

As soon as the combination enumerator does not have any more solution (see the branching on condition $B_{sol} \neq nil$ in Figure 5) the current best solution pipe Π_{best} is returned. Otherwise, the length optimizer is called to find a solution which is better than Π_{best}. When the length optimizer finds such a solution Π_{sol}, Π_{best} is updated. The combination enumerator is then requested for the next possible bend combination.

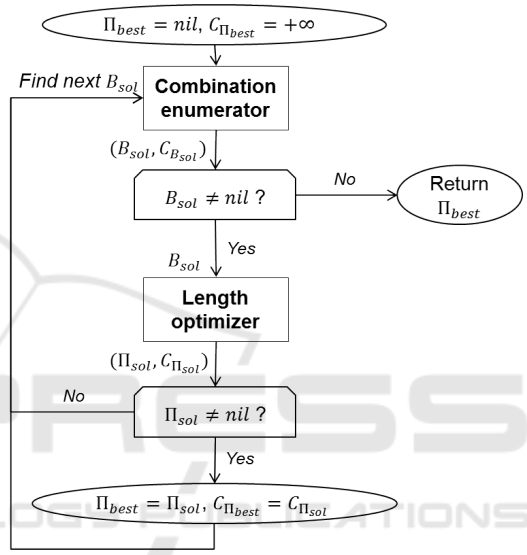


Figure 5: Architecture of the solver.

The time complexity of this 2-step pipe routing algorithm corresponds to the time complexity of the combination enumerator multiplied by the time complexity of the length optimizer which, as shown in Section 3.4, is based on a linear program solvable in polynomial time.

3.3 Combination Enumerator

The combination enumerator enumerates the bend combinations B_{sol} that allow to reach the destination orientation from the source orientation using at most N_{max} bends of catalog $\mathcal{B}_{cat}$. It is an enumeration solver: for an instance of a bend combination problem between o_s (corresponding to orientation R_s) and o_d (corresponding to orientation R_d), at each request, the solver returns a possible bend combination which has not been previously returned, if such a combination exists.

To do so, the bend combination problem is formulated as a reachability problem in the reachable orien-

tation graph $G(O,E)$. For this reachability problem, the revisit of a node is allowed in order to have a complete enumeration of the possible bend combinations. A bend combination can indeed contain several times the same orientation $o \in O$ as a transition orientation. The enumeration algorithm is an A* search algorithm based on successive calls to the following procedure, starting with the list *OpenList* of combinations to explore, initialized with the empty combination. This procedure is presented in Algorithm *Find next* B_{sol}.

To avoid the exploration of the uninteresting bend combinations, a lower bound $C_{LB}(B)$ of the best pipe route which can be obtained with bend combination B is used. This way, the bend combinations B such that $C_{LB}(B)$ is greater than or equal to the current best pipe cost $C_{\Pi_{best}}$ are not explored. A valid lower bound takes into account the cost of the bend combination and the minimal cost of the straight sections:

$$C_{LB}(B) = \max\left(C_B + (N_B+1)\alpha L_{min},\ \alpha\|\overrightarrow{P_sP_d}\|\right)$$

In the previous expression, we implicitly assume that using a unique straight section cannot be more costly than using bends. The bound proposed could be made tighter by considering the minimum cost of the bends required to reach the goal orientation in the reachable orientation graph.

Furthermore, the list *OpenList* of bend combinations to explore is sorted by increasing order of lower bounds $C_{LB}(\cdot)$. To do this, each time a new bend combination B must be added to *OpenList*, its corresponding bound $C_{LB}(B)$ is computed so that it can be inserted at the right position in the list. Then, function *PopFirstElement* used in Algorithm 1 removes and returns a bend combination B whose $C_{LB}(\cdot)$ value is the lowest, which allows to explore the more promising bend combinations first.

Figure 6 shows the evolution of the search tree for a simple reachable orientation graph containing 4 orientations and 3 bends such that $C_{b_1} < C_{b_2} < C_{b_3} < 2C_{b_1}$, with orientation o_1 as source orientation and orientation o_4 as destination orientation. In this case, for $N_{max} = 2$, the procedure returns bend combination $B_1 = [b_1]$ on the first call and bend combination $B_2 = [b_2, b_3]$ on the second one. For $N_{max} = 4$, a bend combination such as $[b_1, b_1, b_3]$ traversing orientations $[o_1, o_4, o_3, o_4]$ could also be returned.

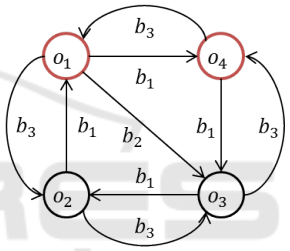
It can be noted that at this step, both the bend catalog constraints (Section 2.2.2) and the pipe attachability constraints (Section 2.2.3) are taken into account.

3.4 Length Optimizer

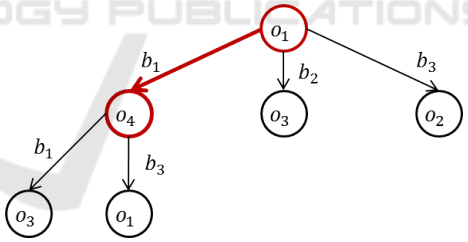
The length optimizer defines the 3D-route of the pipe for a given bend combination $B_{sol} = [b_1, ..., b_n]$ by

Algorithm 1: Find next B_{sol}.

```
Data: G(O,E), OpenList, o_s, o_d, C_{Π_best},
      N_Max ≥ 0
Result: B_sol
begin
    while OpenList ≠ ∅ do
        B ← PopFirstElement(OpenList)
        if C_LB(B) < C_{Π_best} then
            o_B ← B(o_s)
            if N_B < N_Max then
                for (o_B, b, o) ∈ E do
                    Add B ∪ [b] to OpenList
            if o_B = o_d then
                return B
        else
            return nil
    return nil
```



(a) Reachable orientation graph $G(O,E)$.



(b) First solution.

Figure 6: Example of bend combination enumeration.

computing the length of the straight sections. As the bend combination is known, the orientations $O_i = R_{T_i}$ of each transition configurations can be calculated.

Then, the 3D-OFPRP with a given bend combination B_{sol} can be formulated as an LP problem that contains two types of variables:

- the length variables ℓ_i such that, for $i \in \{0, ..., n\}$, real variable ℓ_i is the length of the i^{th} straight section (see Figure 7);
- the transition point variables $\overrightarrow{p_i} = (p_{x_i}, p_{y_i}, p_{z_i})$ such that, for $i \in \{0, ..., n+1\}$, real variable p_{x_i} (respectively p_{y_i} and p_{z_i}) is the coordinate of transition point p_{T_i} (see Figure 7) on the X-axis (re-

spectively on the Y-axis and on the Z-axis).

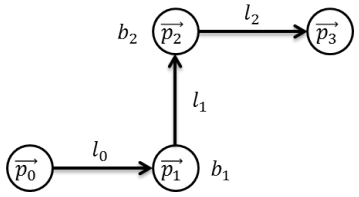


Figure 7: Indices of the transition points, lengths and bends for a 2-bend pipe.

The linear problem to solve is then:

$$\text{minimize} \sum_{i=0}^{n} \alpha \ell_i \tag{8}$$

subject to:

$$\overrightarrow{p_0} = \overrightarrow{p_s} \tag{9}$$

$$\overrightarrow{p_{n+1}} = \overrightarrow{p_d} \tag{10}$$

$$\forall i \in \{0,...,n+1\},\ \forall j \in \{1,...,k_{\mathcal{A}}\},$$
$$A_j p_{x_i} + B_j p_{y_i} + C_j p_{z_i} + D_j \leq 0 \tag{11}$$

$$\forall i \in \{0,...,n\},\ \ \ell_i \geq L_{min} \tag{12}$$

$$\forall i \in \{1,...,n+1\},$$
$$\overrightarrow{p_i} = \overrightarrow{p_{i-1}} + \left(\ell_{i-1} + L_{b_{i-1}} + L_{b_i}\right) \overrightarrow{z_{O_{i-1}}} \tag{13}$$

$$C_{B_{sol}} + \sum_{i=0}^{n} \alpha \ell_i \leq C_{\Pi_{best}} - 1 \tag{14}$$

$$\forall i \in \{0,...,n\},\ \ \ell_i \in \mathbb{R}^+ \tag{15}$$

$$\forall i \in \{0,...,n+1\},\ \ \overrightarrow{p_i} \in \mathbb{R}^3 \tag{16}$$

Constraints 9 and 10 are the origin and destination constraints. Constraints 11 are the routing space constraints. Constraints 12 model the minimal straight section length constraints. Last, Constraints 13 are the position succession constraints. In this equation, terms $L_{b_{i-1}}$ and L_{b_i} represent respectively the cost contribution of the half lengths of the previous and next bends to the length of the segment between transition points $i-1$ and i, and with the convention $L_{b_0} = 0$ and $L_{b_{n+1}} = 0$. Last, Constraint 14 forces to find a solution that is strictly better than the best solution known (Π_{best}).

Contrarily to most of the existing algorithms, such an LP formulation allows to route in a continuous 3D space. In particular, it does not restrain the pipe route to candidate paths using nodes in a discretized 3D representation.

This part of the combined solver takes into account the spatial constraints, including the routing space constraints and the design constraints about straight sections. It is also responsible for optimizing the straight section part of the criterion.

4 MILP MODEL

The second method proposed in this paper is based on a MILP formulation of the 3D-OFPRP(n) problem, which also uses the transition points and the graph of reachable orientations $G(O,E)$. More precisely, the problem contains four types of variables :

- the bend variables $x_{i,b}$ such that, for $i \in \{1,...,n\}$, integer variable $x_{i,b}$ equals 1 if the i^{th} bend of the pipe (see Figure 7) is bend $b \in \mathcal{B}_{cat}$, 0 otherwise;
- the length variables ℓ_i such that, for $i \in \{0,...,n\}$, real variable ℓ_i is the length of the i^{th} straight section;
- the transition orientation variables $y_{i,o}$ such that, for $i \in \{0,...,n\}$, integer variable $y_{i,o}$ equals 1 if the orientation of transition configuration T_i is orientation $o \in O$, 0 otherwise;[1]
- the transition point variables $\overrightarrow{p_i} = (p_{x_i}, p_{y_i}, p_{z_i})$ such that, for $i \in \{0,...,n+1\}$, real variable p_{x_i} (respectively p_{y_i} and p_{z_i}) is the coordinate of transition point p_{T_i} on the X-axis (respectively on the Y-axis and on the Z-axis).

This way, the 3D-OFPRP(n) can be formulated as the following Mixed Integer Linear Program:

$$\text{minimize} \sum_{i=1}^{n} \sum_{b \in \mathcal{B}_{cat}} x_{i,b} C_b + \sum_{i=0}^{n} \alpha \ell_i \tag{17}$$

subject to:

$$\forall i \in \{1,...,n\},\ \sum_{b \in \mathcal{B}_{cat}} x_{i,b} = 1 \tag{18}$$

$$\forall i \in \{0,...,n\},\ \sum_{o \in O} y_{i,o} = 1 \tag{19}$$

$$y_{0,o_s} = 1 \tag{20}$$

$$y_{n,o_d} = 1 \tag{21}$$

$$\overrightarrow{p_0} = \overrightarrow{p_s} \tag{22}$$

$$\overrightarrow{p_{n+1}} = \overrightarrow{p_d} \tag{23}$$

$$\forall i \in \{0,...,n+1\},\ \forall j \in \{1,...,k_{\mathcal{A}}\},$$
$$A_j p_{x_i} + B_j p_{y_i} + C_j p_{z_i} + D_j \leq 0 \tag{24}$$

$$\forall i \in \{0,...,n\},\ \ \ell_i \geq L_{min} \tag{25}$$

$$\forall i \in \{1,...,n+1\},\ \forall o \in O,$$
$$\overrightarrow{p_i} \leq \overrightarrow{p_{i-1}} + (\ell_{i-1} + L(i))\, \overrightarrow{z_o} + \overrightarrow{M}\,(1 - y_{i-1,o}) \tag{26}$$

[1]The use of such variables reduces the number of variables compared to a formulation that would use variables $x_{i,o,b} \in \{0,1\}$ taking value 1 if and only if the i^{th} bend of the pipe is bend b and the orientation of i^{th} transition point is orientation o.

$$\forall i \in \{1,...,n+1\},\ \forall o \in O,$$
$$\overrightarrow{p_i} \geq \overrightarrow{p_{i-1}} + (\ell_{i-1} + L(i))\,\overrightarrow{z_o} - \overrightarrow{M}\,(1 - y_{i-1,o}) \quad (27)$$

$$\forall i \in \{1,...,n\},\ \forall o \in O,$$
$$\forall b \in \mathcal{B}_{cat} \mid\ \nexists o' \in O\ \ (o,b,o') \in E,$$
$$y_{i-1,o} + x_{i,b} \leq 1 \quad (28)$$

$$\forall i \in \{1,...,n\},\ \forall o' \in O,$$
$$\forall b \in \mathcal{B}_{cat} \mid\ \nexists o \in O\ \ (o,b,o') \in E,$$
$$x_{i,b} + y_{i,o'} \leq 1 \quad (29)$$

$$\forall i \in \{1,...,n\},\ \forall o \in O,$$
$$\forall o' \in O \mid\ \nexists b \in \mathcal{B}_{cat}\ \ (o,b,o') \in E,$$
$$y_{i-1,o} + y_{i,o'} \leq 1 \quad (30)$$

$$\forall i \in \{1,...,n\},\ \forall (o,b,o') \in E,$$
$$x_{i,b} + y_{i-1,o} \leq 1 + y_{i,o'} \quad (31)$$
$$y_{i,o'} + x_{i,b} \leq 1 + y_{i-1,o} \quad (32)$$
$$y_{i-1,o} + y_{i,o'} \leq 1 + x_{i,b} \quad (33)$$
$$\forall i \in \{1,...,n\},\ \forall b \in \mathcal{B}_{cat}\ \ x_{i,b} \in \{0,1\} \quad (34)$$
$$\forall i \in \{0,...,n\},\ \ \ell_i \in \mathbb{R}^+ \quad (35)$$
$$\forall i \in \{0,...,n\},\ \forall o \in O\ \ y_{i,o} \in \{0,1\} \quad (36)$$
$$\forall i \in \{0,...,n+1\},\ \ \overrightarrow{p_i} \in \mathbb{R}^3 \quad (37)$$

Constraints 18 and 19 ensure respectively the bend and the transition orientation unicity. Constraints 20 and 21 are the origin and destination constraints for the orientation. Constraints 22-25 are the same as in the LP formulation. Constraints 26-27 are the position succession constraints. The successive transition points are defined using a big-M formulation that ensures that transition point $\overrightarrow{p_i}$ corresponds to the previous transition point $\overrightarrow{p_{i-1}}$ translated along the previous transition orientation $\overrightarrow{z_{o_{i-1}}}$. The translation distance takes into account the previous straight section length ℓ_{i-1} and the half lengths of the previous bend $i-1$ and of bend i, if they exist. Term $\overrightarrow{M} \in (\mathbb{R}^+)^3$ is an arbitrary value that should be greater than the bigger dimension of the routing space in order not to reduce the possible positions for each transition point. Moreover, to simplify Constraints 26-27, term $L(i)$ is defined by:

$$L(i) = L^-(i) + L^+(i)$$

where:

$$L^-(i) = \begin{cases} \sum_{b \in \mathcal{B}_{cat}} x_{i-1,b} L_b \text{ if } i > 1 \\ 0 \text{ otherwise} \end{cases}$$

$$L^+(i) = \begin{cases} \sum_{b \in \mathcal{B}_{cat}} x_{i,b} L_b \text{ if } i < n+1 \\ 0 \text{ otherwise} \end{cases}$$

Constraints 28-30 avoid the use of bend-orientation triplets $(o,b,o') \in O \times \mathcal{B}_{cat} \times O$ which do not match a possible arc in the reachable orientation graph. This formulation of the constraints with three parts generates less constraints than a formulation with only one constraint $y_{i-1,o} + y_{i,o'} + x_{i,b} \leq 2$. Last, Constraints 31-33 are the orientation succession constraints. This formulation ensures that if at least two elements of bend-orientation triple $(o,b,o') \in E$ are chosen for the i^{th} segment of the pipe, then the last one is also chosen. This supposes that there exists at most one bend b such that $(o,b,o') \in E$, which is true if all the bends of $\mathcal{B}_{cat}$ have different rotation matrices. If this hypothesis is not verified, Constraints 31-33 can be removed.

5 RESULTS

The MILP approach and the Combined Graph-LP approach have been compared on four realistic test cases requiring different numbers of bends to be solved.

5.1 Test Cases

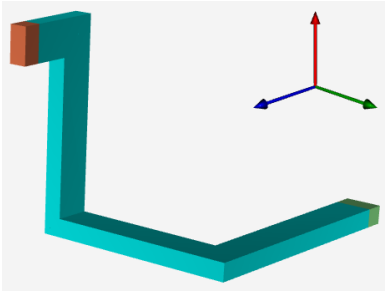
The test cases are as follows:

- **Case 1** contains source and destination in the same plane with the same orientation and has a 2-bend optimal solution (see Figure 8a).
- **Case 2** contains source and destination in different planes with a rotated orientation along the Z-axis and has a 3-bend optimal solution (see Figure 8b).
- **Case 3** contains source and destination in different planes with the same orientation and has a 4-bend optimal solution (see Figure 8c).
- **Case 4** contains source and destination in different planes with a rotated orientation along the Z-axis and destination at the back of the source. This case has a 5-bend optimal solution (see Figure 8d).

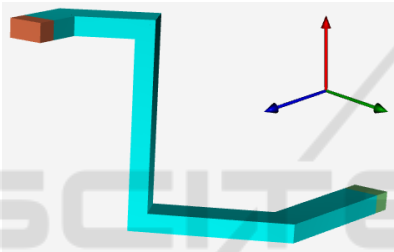
The minimal length of the straight sections used is $L_{min} = 2$. The routing space is a cube of width 10000 centered in $(0,0,0)$. The linear cost of straight sections is $\alpha = 1$.

The four test cases have been solved with three different bend catalogs containing bends around the X-axis and Y-axis (no twisted section around the Z-axis in the catalogs). The first catalog $\mathcal{B}_{cat.1}$ contains four orthogonal bends ($N_{\mathcal{B}_{cat.1}} = 4$). The second one $\mathcal{B}_{cat.2}$ contains the four orthogonal bends and four 45°

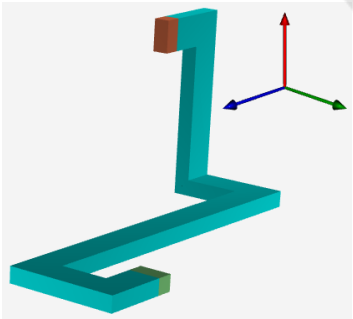
(a) Case 1 (source:$(0,0,0)$, destination: $(3000,-2000,0)$, same orientation for the source and the destination).



(b) Case 2 (source:$(0,0,0)$, destination: $(3000,-2000,2000)$, rotated orientation along the Z-axis for the source and the destination).



(c) Case 3 (source:$(0,0,0)$, destination: $(3000,-2000,2000)$, same orientation for the source and the destination).



(d) Case 4 (source:$(0,0,0)$, destination: $(-3000,-2000,2000)$, rotated orientation along the Z-axis for the source and the destination, and destination at the back of the source).

Figure 8: Test cases with catalog $\mathcal{B}_{cat.3}$, with for each test case the (x,y,z) coordinates of the source and the destination.

bends ($N_{\mathcal{B}_{cat.1}} = 8$). The last one $\mathcal{B}_{cat.3}$ contains the four orthogonal bends, four 30° bends and four 60° bends ($N_{\mathcal{B}_{cat.3}} = 12$).

In order to minimize the number of bends first and then the total length, all the bends have a cost of 20000 which is high compared to the euclidean distance between the source and the destination.

5.2 Generation of the Reachable Orientations

The graph of reachable orientations has been generated for each catalog. Figure 9 shows, for each bend catalog, the evolution of the number of reachable orientations with the maximum number of bends N_{max} allowed in a pipe.

It should be noted that the number of reachable orientations has an asymptotic trend when the number of bends grows. This is explained because first the attachability constraint is taken into account during the enumeration, and second there is a finite number of possible orientations in a given plane due to the bend catalog. In the end, the number of reachable orientations is quite small and the enumeration is easy and fast, as shown in Table 1 which presents average times obtained on 10 generations of graph $G(O,E)$ for each catalog.

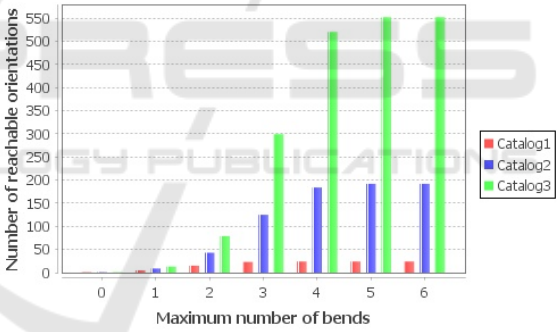


Figure 9: Evolution of the number of reachable orientations with the number of bends for each catalog.

Table 1: Average time required to generate the reachable orientation graph (in milliseconds).

Catalog	Maximum number of bends					
	1	2	3	4	5	6
$\mathcal{B}_{cat.1}$	2.	1.	2.	2.	2.	1.
$\mathcal{B}_{cat.2}$	0.	1.	6.	18.	27.	32.
$\mathcal{B}_{cat.3}$	0.	1.	24.	185.	310.	303.

5.3 Comparison of Both Approaches

The MILP model has been implemented using the OPL language and the instances have been solved using CPLEX 12.9 with an Intel Xeon 1.90 GHz processor and 64 GB RAM. The Combined Graph-LP solver has been implemented in Java and uses the LP

Table 2: Results of the Combined Graph-LP and MILP approaches.

Case	N_{max}	Catalog	Optimal value	Average time to optimum (in s)			
				Combined Graph-LP			MILP
				Total	Comb.Enum.	Len.Optim.	**Total**
1	2	$\mathcal{B}_{cat}.1$	43400	**0.118**	0.025	0.093	**0.304**
		$\mathcal{B}_{cat}.2$	43165.69	**0.028**	0.017	0.011	**2.571**
		$\mathcal{B}_{cat}.3$	43230.94	**0.024**	0.011	0.013	**1.265**
2	3	$\mathcal{B}_{cat}.1$	64600	**0.013**	0.006	0.007	**0.324**
		$\mathcal{B}_{cat}.2$	64600	**0.053**	0.047	0.006	**0.681**
		$\mathcal{B}_{cat}.3$	64600	**0.162**	0.153	0.009	**2.321**
3	4	$\mathcal{B}_{cat}.1$	83800	**0.058**	0.013	0.045	**10.435**
		$\mathcal{B}_{cat}.2$	83565.69	**1.000**	0.939	0.061	**12.606**
		$\mathcal{B}_{cat}.3$	83630.94	**13.413**	13.303	0.110	**9.693**
4	5	$\mathcal{B}_{cat}.1$	105400	**0.207**	0.126	0.081	**29.873**
		$\mathcal{B}_{cat}.2$	105400	**45.747**	45.574	0.173	**27.369**
		$\mathcal{B}_{cat}.3$	105400	**2573.032**	2572.765	0.267	**28.385**

solver of the `Apache Commons Math 3.6.1` library with an Intel 2.70GHZ precessor and 16 GB RAM. As the complexity of this approach comes from the enumeration of bend combinations, the use of the LP solver of `Apache Commons Math 3.6.1` rather than CPLEX 12.9 has no significant impact on computation times. Both solvers have been run on the four test cases with the different catalogs. The results of the Combined Graph-LP solver and of the MILP model are shown in Table 2 which presents the average computation times obtained on 5 executions for each catalog.

Both approaches find the optimal solutions and prove the optimality on each test case. Although the MILP model is slower than the Combined Graph-LP solver with a small number of bends N_{max} or small catalog size $N_{\mathcal{B}_{cat}}$, the MILP approach is more robust when N_{max} and $N_{\mathcal{B}_{cat}}$ increase. Actually, for small problem instances, the resolution is immediate for both methods, but the initialization time of the MILP model (time before the search actually starts) is greater than the initialization time of the Linear Program with fixed bends. Then, it can be noticed that when the bends added to the catalog allow to improve the optimal solution, the resolution of the MILP is faster even if the problem instance contains more variables. This is due to better cuts in the MILP resolution which avoid to unnecessarily consider branches of the search space. There is no equivalent mechanism in the Combined Graph-LP solver. Furthermore, the number of bend combinations to explore in the Combined Graph-LP solver is in $O\left((N_{\mathcal{B}_{cat}})^{N_{max}}\right)$ and increases exponentially with N_{max} as shows the time spent in Combination Enumerator in Table 2.

5.4 Discussion

Both approaches are based on the enumeration of reachable orientations. This enumeration can be time-consuming with large bend catalogs and without pipe orientation constraints to limit the admissible orientations. In this case, if several pipes must be routed based on the same catalog, the enumeration can be done only once by considering relative routing problems that express the destination, the routing space and the reachable orientations relatively to the origin pipe configuration which is assimilated to the reference of the 3D coordinates system.

Furthermore, with test cases 1 and 3, it can be noticed that the introduction of non orthogonal bends in the catalog, such as 30°, 45° or 60° bends, reduces the cost of the optimal solution by saving length. This shows that extending existing pipe routing approaches by using non orthogonal bends is relevant. Other catalogs using only non orthogonal bends (e.g. a catalog containing only the 30°, 45°, and 60° bends) could also be tested. Also, approaches using bend catalogs can easily be extended to the use of more complex patterns with several transition configurations, which is profitable for the reuse of existing patterns in the providers' catalog.

6 CONCLUSIONS AND PERSPECTIVES

This paper introduced two approaches to route a pipe in a 3D continuous space using bends from a catalog. Bends can be described by a rotation matrix and can be non orthogonal. Both approaches are based on the enumeration of reachable orientations at a preprocess-

ing step, which takes into account the orientation of the source and destination sections of the pipe. Also, the techniques proposed make pipe routing possible for pipes that have a non symmetrical section such as a rectangular one.

Both presented approaches are sensitive to the increase in the number of bends, even if the MILP approach seems more robust for delivering an optimal solution. The MILP resolution might also be boosted by using column generation techniques with a master problem considering bend variables. The Combined Graph-LP solver could also be accelerated by improving the combination enumeration procedure with, for example, a bidirectional A* enumeration starting from the source and destination orientations.

As a result, the new methods introduced in this paper can be efficiently used as a first routing algorithm. If the optimal pipe route obtained by solving the 3D-OFPRP problem collides with obstacles of the 3D-PRP problem, Branch-and-Cut techniques can be integrated to the MILP approach by (a) analyzing the solution generated, (b) adding new integer variables and new linear constraints to enforce that pipe sections must not traverse the obstacle sides for which collisions are detected, (c) solving the problem again, and so on until a valid pipe routing is found. One difficulty though will concern the management of collisions with the pipe itself, since the pipe sections cannot be considered as fixed obstacles from one resolution to the next.

REFERENCES

Ando, Y. and Kimura, H. (2012). An automatic piping algorithm including elbows and bends. *Journal of the Japan Society of Naval Architects and Ocean Engineers*, 15:219–226.

Asmara, A. and Nienhuis, U. (2006). Automatic piping system in ship. In *International Conference on Computer and IT Application (COMPIT)*.

Belov, G., Czauderna, T., Dzaferovic, A., de la Banda, M. G., Wybrow, M., and Wallace, M. (2017). An optimization model for 3d pipe routing with flexibility constraints. In *International Conference on Principles and Practice of Constraint Programming*, pages 321–337. Springer.

Fan, X., Lin, Y., and Ji, Z. (2006). The ant colony optimization for ship pipe route design in 3d space. In *2006 6th World Congress on Intelligent Control and Automation*, pages 3103–3108. IEEE.

Furuholmen, M., Glette, K., Hovin, M., and Torresen, J. (2010). Evolutionary approaches to the three-dimensional multi-pipe routing problem: a comparative study using direct encodings. In *European Conference on Evolutionary Computation in Combinatorial Optimization*, pages 71–82. Springer.

Guirardello, R. and Swaney, R. E. (2005). Optimization of process plant layout with pipe routing. *Computers & chemical engineering*, 30(1):99–114.

Hightower, D. W. (1969). A solution to line-routing problems on the continuous plane. In *Proceedings of the 6th annual Design Automation Conference*, pages 1–24. ACM.

Ikehira, S., Kimura, H., Ikezaki, E., and Kajiwara, H. (2005). Automatic design for pipe arrangement using multi-objective genetic algorithms. *Journal of the Japan Society of Naval Architects and Ocean Engineers*, 2:155–160.

Ito, T. (1999). A genetic algorithm approach to piping route path planning. *Journal of Intelligent Manufacturing*, 10(1):103–114.

Jiang, W.-Y., Lin, Y., Chen, M., and Yu, Y.-Y. (2015). A co-evolutionary improved multi-ant colony optimization for ship multiple and branch pipe route design. *Ocean Engineering*, 102:63–70.

Kimura, H. (2011). Automatic designing system for piping and instruments arrangement including branches of pipes. In *International Conference on Computer Applications in Shipbuilding (ICCAS)*, pages 93–99. IEEE.

Lee, C. Y. (1961). An algorithm for path connections and its applications. In *IRE transactions on electronic computers*, number 3, pages 346–365. IEEE.

Liu, L. and Liu, Q. (2018). Multi-objective routing of multi-terminal rectilinear pipe in 3d space by moea/d and rsmt. In *3rd International Conference on Advanced Robotics and Mechatronics (ICARM)*, pages 462–467. IEEE.

Medjdoub, B. and Bi, G. (2018). Parametric-based distribution duct routing generation using constraint-based design approach. *Automation in Construction*, 90:104–116.

Park, J.-H. and Storch, R. L. (2002). Pipe-routing algorithm development: case study of a ship engine room design. *Expert Systems with Applications*, 23(3):299–309.

S.-H., K., W.-S., R., and S., J. B. (2013). The development of a practical pipe auto-routing system in a shipbuilding cad environment using network optimization. *International journal of naval architecture and ocean engineering*, 5(3):468–477.

Sakti, A., Zeidner, L., Hadzic, T., Rock, B. S., and Quartarone, G. (2016). Constraint programming approach for spatial packaging problem. In *International Conference on AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, pages 319–328. Springer.

Zhu, D. and Latombe, J.-C. (1991). Pipe routing-path planning (with many constraints). In *Proceedings. 1991 IEEE International Conference on Robotics and Automation*, pages 1940–1947. IEEE.