INTERNATIONAL JOURNAL OF FLUID POWER, 2016
VOL. 17, NO. 2, 102–113
http://dx.doi.org/10.1080/14399776.2016.1155945

Optimal routing of pipes in a virtual environment using nonlinear programming

Friedrich Kohlmai[a], Volker Baumbach[b], Christof Büskens[a] and Matthias Knauer[a]

[a]Centre for Industrial Mathematics, University of Bremen, Bremen, Germany; [b]Hydraulic Systems Performance & Integrity, Airbus, Bremen, Germany

ABSTRACT
Nowadays the routing of pipes inside or outside of structural components is usually done with CAD tools like CATIA. Up to now it is the task of design-engineers to manually find a connection between two points in $\mathbb{R}^3$ which on one hand respects all design directives and on the other hand minimises the weight of the pipe. As in general the number of feasible routings is very high, if not infinite, it is almost impossible to find an optimal solution without a powerful algorithm and an immense amount of development time. This paper presents a mathematical framework which allows the designer to automatically generate the optimal routing. To do so, the aforementioned problem is treated as an optimisation problem. With the mass of the pipe as the cost function to be minimised and the design directives handled as equality and inequality constraints, this leads to a nonlinear problem (NLP) which is solved by WORHP – a large-scale sparse NLP solver. A series of practically relevant scenarios like routings close to structural components, routings with standard bending angles as well as network routings for metallic and non-metallic pipes are investigated. First tests with real assemblies reveal that a significant reduction of the total mass is achieved by optimising the existing routing.

ARTICLE HISTORY
Received 11 June 2015
Accepted 13 February 2016

KEYWORDS
Optimal routing of pipes; piping; wiring; nonlinear programming; NLP solver WORHP; CATIA

1. Introduction

The development of modern aircrafts, ships and cars but also products from other disciplines, is very comprehensive and thus impossible without the support of proper software like Computer Aided Three-Dimensional Interactive Application (CATIA) or similar. The opportunity to model a product in a virtual environment with the help of useful tools, leads to numerous advantages. Nevertheless, the designing remains challenging when the degrees of freedom get large. While, for instance, the location of components like pumps, manifolds or actuators may be predefined by the system, the routing of pipes has to be adapted to structural components. In this context a routing has to be found by the designer that

- complies to the design directives,
- doesn't collide with any obstacles or other pipes,
- minimises the weight of the pipe.

These requirements form a typical optimisation problem which can be solved with numerical mathematics. Hence, the idea described in this paper is to translate the original problem into mathematics, i.e. into equations, to solve them with numerical algorithms and to transfer the results back to the original formulation.

In the past related problems have been observed in computer sciences, e.g. artificial intelligence, in navigation of vehicles, etc. Usually the well-known A*-algorithm was used which is based on a series of heuristics (Norvig and Russell 2009). For the abovementioned problem, however, this kind of solver approved to be too slow since the problem dimension becomes too large.

Within the scope of this article the approach to solve the problem is to use nonlinear programming. In this context a detailed derivation of the optimisation problem is given, followed by the investigation of a series of practically relevant routing scenarios (Kohlmai 2014). To additionally reduce the weight of pipes also the usage of non-metallic materials is considered. Those need to be treated differently than metallic ones, since non-metallic pipes are difficult to bend nowadays. As a consequence, a random shape of a pipe can only be created by splitting the geometry into straight and bended parts. Since these parts have to be connected with fittings or similar, additional masses have to be taken into account.

2. Nonlinear optimisation

As the theory of nonlinear optimisation is too extensive to be discussed here in detail, only the basic idea and the standard notation will be introduced. The interested readers may refer to (Avriel 2012) and (Geiger and Kanzow 2002).

CONTACT Friedrich Kohlmai kohlmai@math.uni-bremen.de

© 2016 Informa UK Limited, trading as Taylor & Francis Group

Minimising an unrestricted (nonlinear) function $f: \mathbb{R}^n \to \mathbb{R}$ is a typical task in many scientific fields. The standard approach is to find a solution $\boldsymbol{x} \in \mathbb{R}^n$ that fulfils the necessary condition

$$\nabla f(\boldsymbol{x}) = 0 \tag{1}$$

and the sufficient condition

$$\boldsymbol{z}^T \nabla^2 f(\boldsymbol{x}) \boldsymbol{z} > 0 \quad \forall \boldsymbol{z} \in \mathbb{R}^n, \boldsymbol{z} \neq 0. \tag{2}$$

In Equation (1) the expression ∇f represents the gradient of f and respectively in Equation (2) the statement $\nabla^2 f$ is the Hessian matrix of f. Since Equation (1) normally cannot be solved analytically, numerical algorithms like Newton's method are applied.

In general, the main idea in minimising restricted functions is similar to the unrestricted case, except that additional constraint functions must be considered. The standard formulation for a nonlinear problem (NLP) is stated below:

$$\begin{aligned} &\underset{\boldsymbol{x}\in\mathbb{R}^n}{\text{minimise}}\, f(\boldsymbol{x}) \\ &\text{subject to: } g_i(\boldsymbol{x}) \leq 0 \text{ with } i \in \{1, \ldots, m\} \\ &\qquad\qquad h_j(\boldsymbol{x}) = 0 \text{ with } j \in \{1, \ldots, p\} \end{aligned} \tag{NLP}$$

Since the (nonlinear) constraints g_i and h_j affect the search for a point $\boldsymbol{x}$ which minimises the cost function f, it is natural to observe a function that combines f with g_i and h_j.

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) := f(\boldsymbol{x}) + \sum_{i=1}^{m} \lambda_i g_i(\boldsymbol{x}) + \sum_{j=1}^{p} \mu_j h_j(\boldsymbol{x}) \tag{3}$$

Equation (3) defines the Lagrangian function where $\boldsymbol{\lambda} \in \mathbb{R}^m$ and $\boldsymbol{\mu} \in \mathbb{R}^p$ are known as Lagrangian multipliers. Hence, a triple $\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}$ has to be found that minimises Equation (3). Analogously to the unrestricted case the triple has to fulfil necessary conditions like the Karush–Kuhn–Tucker (KKT) conditions:

$$\nabla \mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) = 0 \tag{4a}$$

$$h_j(\boldsymbol{x}) = 0 \tag{4b}$$

$$g_i(\boldsymbol{x}) \leq 0 \tag{4c}$$

$$\lambda_i \leq 0 \tag{4d}$$

$$\lambda_i g_i(\boldsymbol{x}) = 0 \tag{4e}$$

These guarantee that both sums in Equation (3) vanish and thus do not change the cost function whereas at the same time the derivatives of g_i and h_j affect the derivative of the Lagrangian. In addition to Equations (4a)–(4e) it should be noted that the solution $\boldsymbol{x}$ also has to comply with some regularisations like the Linear Independence Constraints Qualification (LICQ) and the triple $\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}$ has to comply with sufficient conditions (Geiger and Kanzow 2002).

Solving Equations (4a)–(4e) is challenging since inequations appear which prohibit the direct application of Newton's method. To circumvent this problem and to achieve a good convergence behaviour a lot of numerical effort has to be made. In the scope of this paper the NLP solver WORHP (Büskens and Wassel 2012) was used which was mainly developed at the department 'Optimization and Optimal Control' at the University of Bremen. WORHP is a Sequential Quadratic Programming (SQP) method that solves a series of quadratic sub-problems which converge to the solution of the original problem. The gradient based algorithm is accelerated by using an Armijo step size control. Amongst other techniques WORHP offers the usage of finite differences or BFGS to compute derivatives and a globalisation strategy in case the given initial values are far from the optimum.

3. Pipe optimisation

The challenge besides the numerical problems mentioned above is to model the original problem with all its restrictions and to guarantee that the involved functions are continuously differentiable as this is necessary in Equation (4a).

3.1. Modelling of the cost function

As mentioned in the introduction the weight of the routing has to be minimised. Thus the mass m represents the cost function.

3.1.1. Metallic pipes

To begin with, the cost function for metallic pipes is modelled. In this regard the fluid that flows through the pipe is considered although this is not absolutely necessary in this case.

$$\begin{aligned} m &= m_p + m_{fl} = \rho_p V_p + \rho_{fl} V_{fl} \\ &= \rho_p \frac{\pi}{4}\left(d_o^2 - d_i^2\right) l + \rho_{fl} \frac{\pi}{4} d_i^2 l \\ &= \underbrace{\left(\rho_p \frac{\pi}{4}\left(d_o^2 - d_i^2\right) + \rho_{fl} \frac{\pi}{4} d_i^2\right)}_{\text{const.}} l \end{aligned} \tag{5}$$

In Equation (5) the subscripts p and fl stand for pipe and fluid, whereas the variables V, ρ, d_o, d_i, l represent the volume, density, outer diameter, inner diameter and length respectively. It becomes clear that the mass m is a function of the length l since all the other expressions are constant.

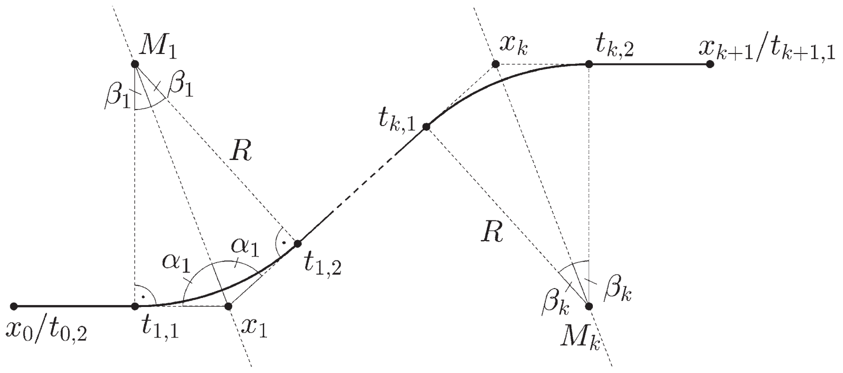


Figure 1. Centre line of a random pipe with nodes $x_0, \dots, x_{k+1}$, tangent points $t_{0,2}, \dots, t_{k+1,1}$ and constant bending radius R.

In Figure 1 the centre line of a pipe with $k \in \mathbb{N}$ bends and a constant bending Radius R is sketched. The determination of its length l can either be done with the tangent points $\boldsymbol{t} \in \mathbb{R}^3$ or with the nodes $\boldsymbol{x} \in \mathbb{R}^3$. Due to the fact that the number of nodes is half the number of tangent points, the aim is to describe the length as a function of $\boldsymbol{x}$.

$$l(\boldsymbol{t}) = \sum_{i=0}^{k} \|\boldsymbol{t}_{i,2} - \boldsymbol{t}_{i+1,1}\| + \sum_{j=1}^{k} b_j \tag{6}$$

Equation (6) defines the length as the sum of straight parts $\|\boldsymbol{t}_{i,2} - \boldsymbol{t}_{i+1,1}\|$, where $\|\cdot\|$ denotes the Euclidian norm, and the sum of arc lengths b_j. In order to obtain $l = l(\boldsymbol{x})$ the terms have to be rewritten.

3.1.1.1. Arc length. The definition of the scalar product of two vectors $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^n$ that enclose an angle γ is given by $\boldsymbol{a} \cdot \boldsymbol{b} = \|\boldsymbol{a}\|\|\boldsymbol{b}\|\cos(\gamma)$. From Figure 1, $\boldsymbol{a}$ can be identified as $(\boldsymbol{x}_j - \boldsymbol{x}_{j-1})$, $\boldsymbol{b}$ as $(\boldsymbol{x}_j - \boldsymbol{x}_{j+1})$ and γ as $2\alpha_j$. Thus the bending angle is given by:

$$2\alpha_j = \arccos\left(\frac{\left(\boldsymbol{x}_j - \boldsymbol{x}_{j-1}\right) \cdot \left(\boldsymbol{x}_j - \boldsymbol{x}_{j+1}\right)}{\|\boldsymbol{x}_j - \boldsymbol{x}_{j-1}\|\|\boldsymbol{x}_j - \boldsymbol{x}_{j+1}\|}\right) \tag{7}$$

Moreover Figure 1 reveals the dependency between the angles α_j and β_j. By the angular sum in a quadrilateral, it follows:

$$2\beta_j = \pi - 2\alpha_j \tag{8}$$

With Equations (7) and (8) finally the arc length is computed:

$$\begin{aligned} b_j = 2\beta_j R &= \left(\pi - 2\alpha_j\right) R \\ &= \arccos\left(-\frac{\left(\boldsymbol{x}_j - \boldsymbol{x}_{j-1}\right) \cdot \left(\boldsymbol{x}_j - \boldsymbol{x}_{j+1}\right)}{\|\boldsymbol{x}_j - \boldsymbol{x}_{j-1}\|\|\boldsymbol{x}_j - \boldsymbol{x}_{j+1}\|}\right) R \end{aligned} \tag{9}$$

3.1.1.2. Length of the straight parts. As shown in Figure 1, on one hand it is

$$\begin{aligned} \|\boldsymbol{x}_i - \boldsymbol{x}_{i+1}\| = \|\boldsymbol{x}_i - \boldsymbol{t}_{i,2}\| + \|\boldsymbol{t}_{i,2} - \boldsymbol{t}_{i+1,1}\| \\ + \|\boldsymbol{t}_{i+1,1} - \boldsymbol{x}_{i+1}\| \end{aligned} \tag{10}$$

and on the other hand

$$\|\boldsymbol{x}_i - \boldsymbol{t}_{i,2}\| = R\tan\left(\beta_i\right) \tag{11}$$

$$\|\boldsymbol{t}_{i+1,1} - \boldsymbol{x}_{i+1}\| = R\tan\left(\beta_{i+1}\right). \tag{12}$$

Thus it follows from Equations (10)–(12):

$$\begin{aligned} \|\boldsymbol{t}_{i,2} - \boldsymbol{t}_{i+1,1}\| =& \|\boldsymbol{x}_i - \boldsymbol{x}_{i+1}\| \\ &- R\left(\tan\left(\beta_i\right) + \tan\left(\beta_{i+1}\right)\right) \end{aligned} \tag{13}$$

3.1.1.3. Summary. The length l can be described as a function of nodes $\boldsymbol{x}$:

$$l(\boldsymbol{x}) = \sum_{i=0}^{k} \|\boldsymbol{x}_i - \boldsymbol{x}_{i+1}\| - 2R\sum_{j=1}^{k}\left(\tan\left(\beta_j\right) - \beta_j\right) \tag{14}$$

$$\beta_j = \frac{1}{2}\arccos\left(-\frac{\left(\boldsymbol{x}_j - \boldsymbol{x}_{j-1}\right) \cdot \left(\boldsymbol{x}_j - \boldsymbol{x}_{j+1}\right)}{\|\boldsymbol{x}_j - \boldsymbol{x}_{j-1}\|\|\boldsymbol{x}_j - \boldsymbol{x}_{j+1}\|}\right). \tag{15}$$

As it can be seen from Equations (14) and (15) the length is a nonlinear function. For the fact that the optimal number of bends is discrete and unknown, the overall optimisation problem gets also discrete and therefore even more complicated.

3.1.2. Non-metallic pipes

In the case of non-metallic pipes, it is assumed that the pipes are not bendable. As a consequence of this, a non-metallic pipe has to be composed by bends and straights which have to be connected somehow in order to get the desired shape, see Figure 2.

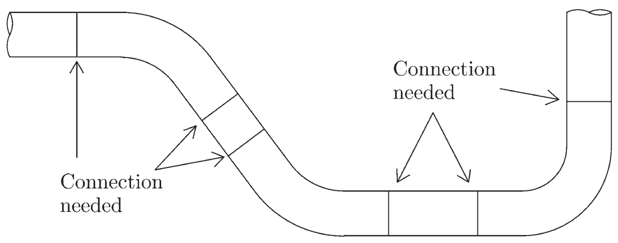


Figure 2. A non-metallic pipe has to be composed by bends and straight parts which are manufactured separately. Fittings or similar are needed to hold these parts together.

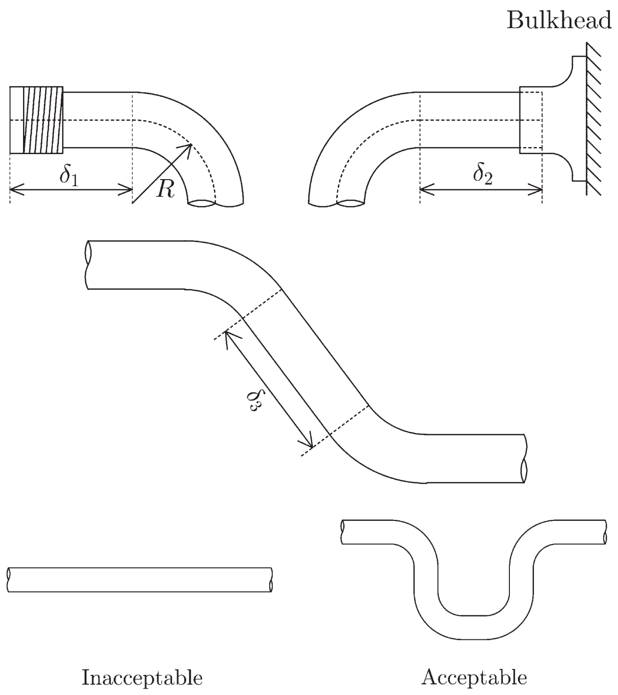


Figure 3. In the top row the minimum distance between an end of a pipe and the following bending is illustrated. In the middle row a minimum distance between two bends is shown. The bottom row indicates that straight connections between two points shall be avoided since thermal expansion or deflections might be hindered.

Since connection parts like fittings or sleeves add further masses to the pipe, the cost function m has to be extended by m_{Con} which represents the weight of the connectors.

$$m = \left(\rho_p \frac{\pi}{4}\left(d_o^2 - d_i^2\right) + \rho_{fl}\frac{\pi}{4}d_i^2\right)l + 2k\, m_{\text{Con}} \quad (16)$$

The term $2k$ arises from the fact that for each bending two connectors are needed. As it can be seen this time, the consideration of the fluids mass is obligatory to preserve the right ratio between the weight of the pipe and the weight of the connectors.

3.2. Modelling of design constraints

In the following a few selected design restrictions shall be considered. While some of them result due to manufacturing limitations, others are needed for technical reasons.

In fabrication it is common to form long straight pipes which get cut into the desired lengths and then get shaped by bending machines. Usually these machines use one bending radius only to accelerate the bending process. This fact was already considered within the derivation of the length l since R was assumed to be constant. Another constraint that has an impact on the geometry of a pipe is the fact that between the ends of a pipe and the neighboured bends a minimum distance δ must be given. This is illustrated at the top row of Figure 3 which indicates that the value of δ depends on the way the pipe endings looks like. In mathematical expression this can be written as:

$$\delta_1 \leq \|\boldsymbol{t}_{0,2} - \boldsymbol{t}_{1,1}\| \quad (16)$$

$$\delta_2 \leq \|\boldsymbol{t}_{k,2} - \boldsymbol{t}_{k+1,1}\| \quad (17)$$

Equation (16) describes the minimum distance for the left end of the pipe (see Figure 1) which at least has the value of δ_1. The same goes for the right end which is described by Equation (17). For the sake of completeness Equations (16) and (17) must be reformulated into:

$$g_1(\boldsymbol{t}) := \delta_1 - \|\boldsymbol{t}_{0,2} - \boldsymbol{t}_{1,1}\| \leq 0 \quad (18)$$

$$g_2(\boldsymbol{t}) := \delta_2 - \|\boldsymbol{t}_{k,2} - \boldsymbol{t}_{k+1,1}\| \leq 0 \quad (19)$$

It has to be noted that these constraints must also be rewritten in dependency to the nodes $\boldsymbol{x}$ which in this case is easily done by applying Equation (13) with $i = 0$ and $i = k$.

A similar constraint is valid between two neighboured bends which can be seen in the middle row of Figure 3. Equation (20) describes the inequality constraints that must be considered within the NLP.

$$\delta_3 \leq \|\boldsymbol{t}_{i,2} - \boldsymbol{t}_{i+1,1}\| \quad (20)$$

The next constraint deals with the bending angles of the pipe. It is assumed that the range of bending angles is limited due to manufactural reasons. Therefore a lower and upper bound δ_4 and δ_5 are defined:

$$\delta_4 \leq 2\beta_j \leq \delta_5 \quad (21)$$

Equation (21) represents two inequalities which can be separated in order to comply with the standard NLP notation.

As some pipes may be routed along areas where the temperature is very high or where the pipe bends due to external forces, it might be useful to demand higher bending angles since these can better compensate expansions and deflections (this is indicated in Figure 3 at the bottom). In this case a suitable constraint can be of the form:

$$\delta_6 \leq \sum_{j=1}^{k} 2\beta_j \quad (22)$$

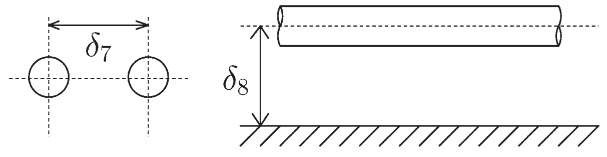


Figure 4. The collision between two pipes (left) and between a pipe and an obstacle (right) must be avoided.

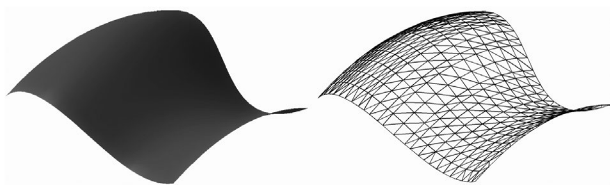


Figure 5. A sample surface (left) was created in CATIA and exported via STL to MATLAB® where it was reconstructed (right).

Equation (22) states that the sum of all bending angles must not deceed a predefined value δ_6. An alternative possibility would be to set the lower bound δ_4 in Equation (21) to a higher value.

Another typical constraint for the bending angles is the limitation to a discrete set of standard angles like the ones in Equation (23) or any other set.

$$2\beta_j \in \{10^\circ, 20^\circ, 30^\circ, \ldots, 170^\circ\} \tag{23}$$

This limitation is of particular interest when non-metallic pipes are used. A discrete set of standard bends is cheaper in production and reduces the quantity of different bends.

3.3. Collision avoidance

This section sets the focal point to the definition of distance functions which guarantee that collisions between pipes and other objects are avoided. In this context it is necessary to distinguish between the distance of two pipes and between the distance of a pipe and another object, see Figure 4.

Therefore the aim is to obtain further constraints like

$$\delta_7 \le \mathcal{D}\left(g_x, h_y\right) \tag{24}$$

$$\delta_8 \le \mathbb{D}\left(g_x, E_{tr}\right) \tag{25}$$

where the distance functions $\mathcal{D}$ and $\mathbb{D}$ ensure that a minimal distance of δ_7 and δ_8 are kept.

In order to choose a meaningful definition for the distance functions, the following assumptions are made:

- pipes are treated like polygonal lines,
- obstacles are treated like a composition of triangles.

While the first assumption is made to simplify the problem, the second one is motivated by software tools like CATIA, AutoCAD or similar. These offer export options such as Surface Tessellation Language (STL) which saves any geometrical structure as a set of triangles, see also Figure 5.

Hence, the definition of the distance functions can be reduced to compute the distance between line segments and triangles.

3.3.1. Distance between line segments and points

To begin with, the distance between a line segment g_x that starts at the node $\boldsymbol{x}_i$ and ends at $\boldsymbol{x}_{i+1}$ and a point $\boldsymbol{P}$ is observed, as it is indicated in Figure 6 on the left.

The line segment is given below in parametrical form where the parameter r is limited to [0, 1].

$$g_x(\boldsymbol{x}): \boldsymbol{x} = \boldsymbol{x}_i + r\,\underbrace{(\boldsymbol{x}_{i+1} - \boldsymbol{x}_i)}_{:=\boldsymbol{u}} = \boldsymbol{x}_i + r\boldsymbol{u} \tag{26}$$

If the projection point $\boldsymbol{F_P}$ of $\boldsymbol{P}$ on the line through g_x is part of g_x then the distance is easily computed since it is the distance between $\boldsymbol{P}$ and $\boldsymbol{F_P}$. Otherwise it is either the distance between $\boldsymbol{P}$ and $\boldsymbol{x}_i$ or between $\boldsymbol{P}$ and $\boldsymbol{x}_{i+1}$. To decide which one is the case, it is useful to observe the location of $\boldsymbol{F_P}$ or more precise the parameter r of$\boldsymbol{F_P}$. Thus the plane E through $\boldsymbol{P}$, which is normal to g_x, must be computed.

$$E(\boldsymbol{x}): \boldsymbol{x} \cdot \boldsymbol{u}^T = c \tag{27}$$

The constant c is obtained by evaluating Equation (27) for the point $\boldsymbol{P}$. Bringing Equations (26) and (27) together then leads to the parameter r:

$$r = \frac{c - \boldsymbol{x}_i \cdot \boldsymbol{u}^T}{\boldsymbol{u} \cdot \boldsymbol{u}^T} \tag{28}$$

This finally leads to the first distance function $d(g_x, \boldsymbol{P})$:

$$d\left(g_x, \boldsymbol{P}\right) := \begin{cases} \|\boldsymbol{P} - \boldsymbol{x}_i\| & \text{if} \quad r < 0 \\ \|\boldsymbol{P} - \boldsymbol{F_P}\| & \text{if} \quad r \in [0, 1] \\ \|\boldsymbol{P} - \boldsymbol{x}_{i+1}\| & \text{if} \quad r > 1 \end{cases} \tag{29}$$

3.3.2. Distance between two line segments

Determining the distance between two line segments g_x and h_y in $\mathbb{R}^3$ gets more complicated as more points are involved. The right illustration in Figure 6 indicates that two parameters r and s must be observed this time in order to be able to compute the minimal distance. The line segment h_y is defined below.

$$h_y(\boldsymbol{x}): \boldsymbol{x} = \boldsymbol{y}_j + s\,\underbrace{(\boldsymbol{y}_{j+1} - \boldsymbol{y}_j)}_{:=\boldsymbol{v}} = \boldsymbol{y}_j + s\boldsymbol{v} \tag{30}$$

This time the plane E contains h_y and is spanned by the vectors $\boldsymbol{v}$ and $\boldsymbol{w} := \boldsymbol{u} \times \boldsymbol{v}$ where $\times$ denotes the cross product of two vectors.

$$E(\boldsymbol{x}): \boldsymbol{x} = \boldsymbol{y}_j + s\boldsymbol{v} + t\boldsymbol{w} \tag{31}$$

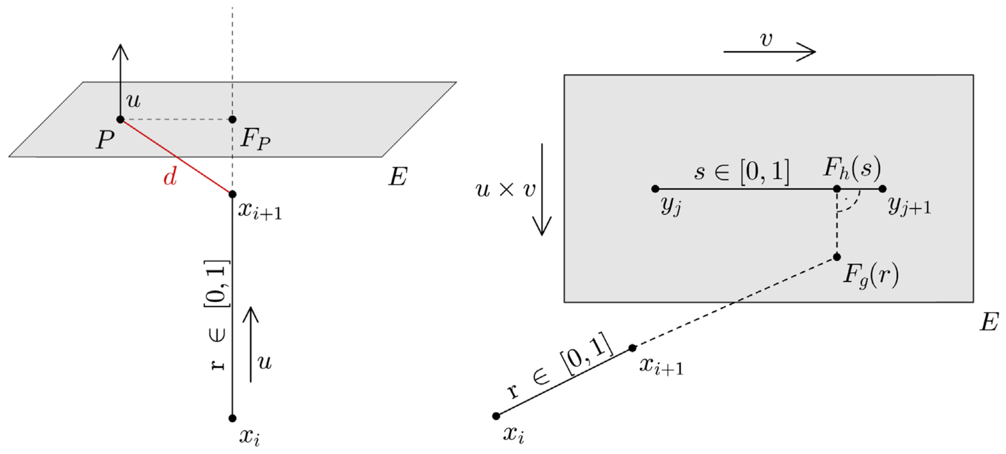


Figure 6. The left illustration shows that the shortest distance between a point *P* and a line segment g_x is given by *d*. On the right picture the parameters *r* and *s* of the two points F_g and F_h have to be computed first to get the minimal distance between two line segments g_x and h_y.

Intersecting E with g_x and solving the resulting linear system of equations in Equation (32) then leads to the intersection point $\boldsymbol{F}_g$ and automatically to the parameters r, s, t.

$$\begin{aligned} \boldsymbol{x}_i + r\boldsymbol{u} &= \boldsymbol{y}_j + s\boldsymbol{v} + t\boldsymbol{w} \\ \Leftrightarrow \ [\mathbf{u}, -\mathbf{v}, -\boldsymbol{w}] \cdot (r, s, t)^T &= \boldsymbol{y}_j - \boldsymbol{x}_i \end{aligned} \tag{32}$$

With r and s the second distance function is defined:

$$\mathcal{D}\left(g_x, h_y\right) := \min \begin{cases} d\left(g_x, \boldsymbol{y}_j\right) & \\ d\left(g_x, \boldsymbol{y}_{j+1}\right) & \\ d\left(h_y, \boldsymbol{x}_i\right) & \\ d\left(h_y, \boldsymbol{x}_{i+1}\right) & \\ ||\boldsymbol{F}_g - \boldsymbol{F}_h|| & \text{if} \quad r, s \in [0, 1] \\ ||\boldsymbol{F}_g - \boldsymbol{y}_j|| & \text{if} \quad r \in [0, 1], s < 0 \\ ||\boldsymbol{F}_g - \boldsymbol{y}_{j+1}|| & \text{if} \quad r \in [0, 1], s < 1 \end{cases} \tag{33}$$

In case the line segments g_x and h_y are parallel, the plane E cannot be spanned. Then the distance function $\mathcal{D}$ reduces to the first four entries.

3.3.3. *Distance between line segments and triangles*

It is reasonable that deriving the distance between a line segment and a triangle is even more complicated since a triangle consists of three points $\boldsymbol{y}_j, \boldsymbol{y}_{j+1}, \boldsymbol{y}_{j+2}$ as well as of three line segments h_{y1}, h_{y2}, h_{y3} and a plane E_{tr} which is bounded by them, see Figure 7.

The idea in determining the distance is to project the ending points of the line segments onto the plane that contains E_{tr}, since this helps to decide which further action has to be taken. As an example the right illustration in Figure 7 can be observed. The projections lay in the areas 2 and 3 which means that the minimal distance is either $\mathcal{D}\left(g_x, h_{y2}\right)$ or $\mathcal{D}\left(g_x, h_{y3}\right)$. Thus the areas must be defined first.

$$\begin{array}{llll} A_1: & s \in [0, 1] & , t \in [0, 1] & , s + t \le 1 \\ A_2: & s \ge 0 & , t \ge 0] & , s + t < 1 \\ A_3: & s > 0 & , t \in [0, 1] & \\ A_4: & s \in [0, 1] & , t < 0 & \\ A_5: & s < 0 & , t \ge 1 & \\ A_6: & s < 0 & , t < 0 & \\ A_7: & s > 1 & , t < 0 & \end{array} \tag{34}$$

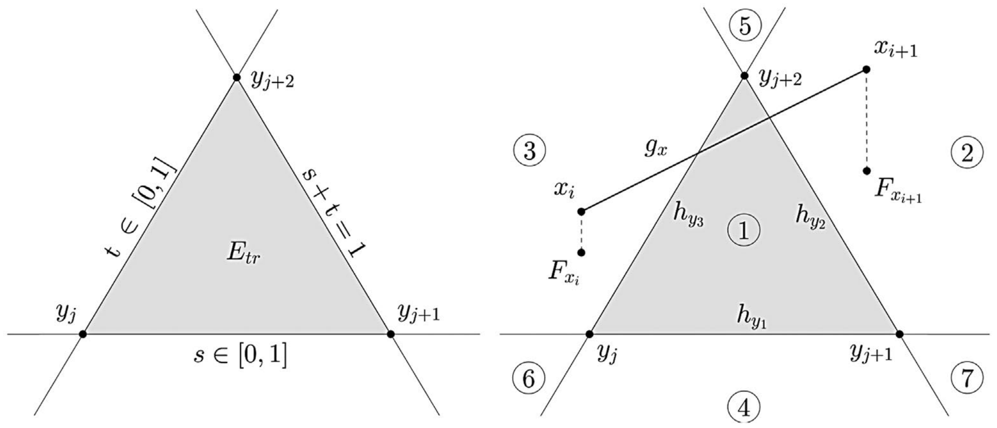


Figure 7. A triangle E_{tr} is defined by limiting the parameters s and t of h_{y1} and h_{y3} to [0, 1] and by restricting the sum of them to $s + t \leq 1$. This also defines the area 1 in the right illustration. Changing the values of r, s, t define the areas 2–7.

In the second step the projections $\boldsymbol{F}_{\boldsymbol{x}_i}$ and $\boldsymbol{F}_{\boldsymbol{x}_{i+1}}$ of the nodes $\boldsymbol{x}_i$ and $\boldsymbol{x}_{i+1}$ on the plane must be computed which can be done with analytical geometry. Then finally the distance function $\mathbb{D}(g_x, E_{tr})$ can be defined by considering every possible combination of the projection points $\boldsymbol{F}_{\boldsymbol{x}_i}$ and $\boldsymbol{F}_{\boldsymbol{x}_{i+1}}$. As 49 different cases exist, only an excerpt of the distance function $\mathbb{D}$ is shown in Equation (35).

$$\mathbb{D}(g_x, E_{tr}) := \left\{ \begin{array}{ccc} & \text{if } \boldsymbol{F}_{\boldsymbol{x}_i} \in & \boldsymbol{F}_{\boldsymbol{x}_{i+1}} \in \\ \min\left(||\boldsymbol{x}_i - \boldsymbol{F}_{\boldsymbol{x}_i}||, ||\boldsymbol{x}_{i+1} - \boldsymbol{F}_{\boldsymbol{x}_{i+1}}||\right) & A_1 & A_1 \\ \min\left(||\boldsymbol{x}_i - \boldsymbol{F}_{\boldsymbol{x}_i}||, \mathcal{D}\left(g_x, h_{y2}\right)\right) & A_1 & A_2 \\ \min\left(||\boldsymbol{x}_i - \boldsymbol{F}_{\boldsymbol{x}_i+1}||, \mathcal{D}\left(g_x, h_{y2}\right)\right) & A_2 & A_1 \\ \vdots & \vdots & \vdots \\ \min\left(\mathcal{D}\left(g_x, h_{y1}\right), \mathcal{D}\left(g_x, h_{y3}\right)\right) & A_3 & A_4 \\ \min\left(\mathcal{D}\left(g_x, h_{y1}\right), \mathcal{D}\left(g_x, h_{y3}\right)\right) & A_4 & A_3 \\ \vdots & \vdots & \vdots \\ \mathcal{D}\left(g_x, h_{y1}\right) & A_6 & A_7 \\ \mathcal{D}\left(g_x, h_{y1}\right) & A_7 & A_6 \\ d\left(g_x, \boldsymbol{y}_{j+1}\right) & A_7 & A_7 \end{array} \right. \tag{35}$$

For the sake of completeness it has to be pointed out that prior to evaluating Equation (35), it has to be investigated whether a line segment g_x crosses the triangle E_{tr}. This is done by setting Equations (26) and (36) equal. If the parameters s and t comply with the setting of A_1 then the distance $\mathbb{D}(g_x, E_{tr})$ is zero.

$$E_{tr}(\boldsymbol{x}) \colon \boldsymbol{x} = \boldsymbol{y}_j + s(\boldsymbol{y}_{j+1} - \boldsymbol{y}_j) + t(\boldsymbol{y}_{j+2} - \boldsymbol{y}_j) \tag{36}$$

One should be reminded that in order to stick to a distance δ, the distances between every line segments and every triangles have to be computed. Thus, if the number of triangles becomes large, the computational time increases significantly.

As a final note to this section it has to be mentioned that Equations (29), (33) and (35) are differentiable and therefore comply with the KKT conditions. This can be shown with analytical geometry but shall be skipped here due to its extent.

4. Numerical results

In this chapter some capabilities of the solving algorithm will be shown by different problem scenarios. Most of the tests are performed on the surface of an elliptic hull which was designed in CATIA, transferred to MATLAB® and solved by WORHP (see Figures 8–12).

In order to perform the tests the settings from Table 1 are chosen. ρ_{Sky} represents the density of the oil Skydrol LD-4, ρ_{Ti} the density of Ti-3Al-2.5V and ρ_{CFRP} the density of CFRP (Carbon Fibre Reinforced Plastic). All values in Table 1 are typical in aircraft engineering with the only exception m_{Con}. Its value might vary and thus was set arbitrarily.

4.1. Metallic vs. non-metallic pipes

Using non-metallic materials to save weight is standing to reason as its density is lower than the one of metallic materials. Comparing ρ_{CFRP} with ρ_{Ti} brings a reduction of around 64%. However, since additional components are needed to connect straight parts with bends, the advantage can get lost due to the fact that every bending adds $2m_{\text{Con}}$ to the total weight of the pipe.

Regardless the choice of material, the optimal amount of bends needs to be found in order to optimise the routing. To do so the optimisation has to be repeated for increasing k, starting with $k = 1$. Once the total weight stops to decrease the optimal number is determined. However, for this simple method many iterations might be needed.

Figure 8 shows the optimal routings for both metallic (bottom) and non-metallic pipes (top). As expected, the routing of the metallic pipe adopts the shape of the hull and thus has the minimal length. Choosing more bends worsens the result as the nodes start to affect each other (minimal distance and bending angle must be kept!). For non-metallic pipes one bending minimises its mass although it extends its length. Table 2 reveals that around 27.5% can be saved in this test by using a non-metallic pipe.

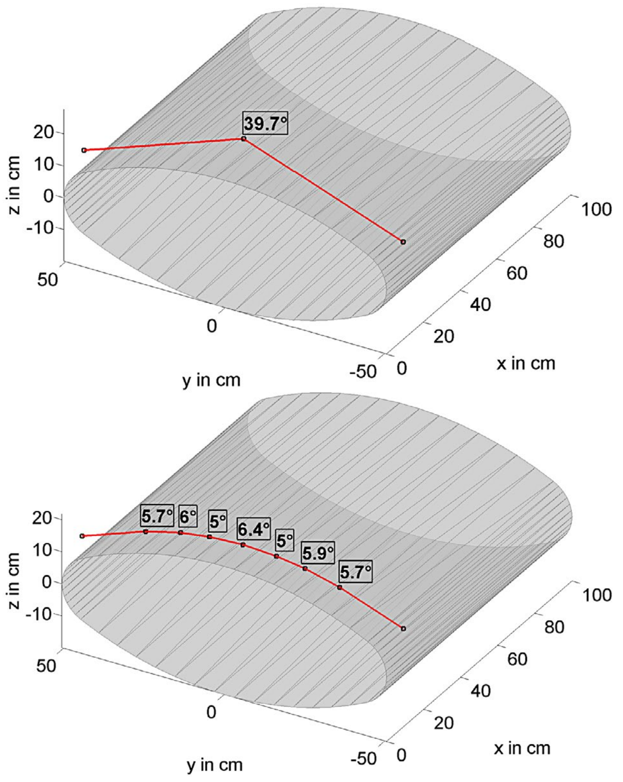


Figure 8. Top: Optimal routing for a non-metallic pipe. Bottom: Optimal routing for a metallic pipe.

4.2. Variation of the bending angles

As non-metallic pipes are assumed to be a composition of straight parts and bends, it is preferable to restrict the bending angles to a set of standard angles, as stated in Equation (23). Thus the bending angle of 39.7° in Figure 8 is not acceptable. However, rounding it up to the next value of the given set of angles, setting this angle as an equality constraint and then repeating the optimisation process, leads to the desired result which can be seen in Figure 9 at the top.

If a pipe is routed along hot areas or structures that tend to deflect, metallic materials are preferred due to their elastic behaviour. To improve their flexibility the total bending angle $\sum_j 2\beta_j$ can be increased. This can be seen in Figure 9 at the bottom where the value was set to 120°. While the routing becomes more cornered, the total mass barely increases, see Table 3. As only one additional constraint has to be defined, this way of reducing the stiffness is remarkable.

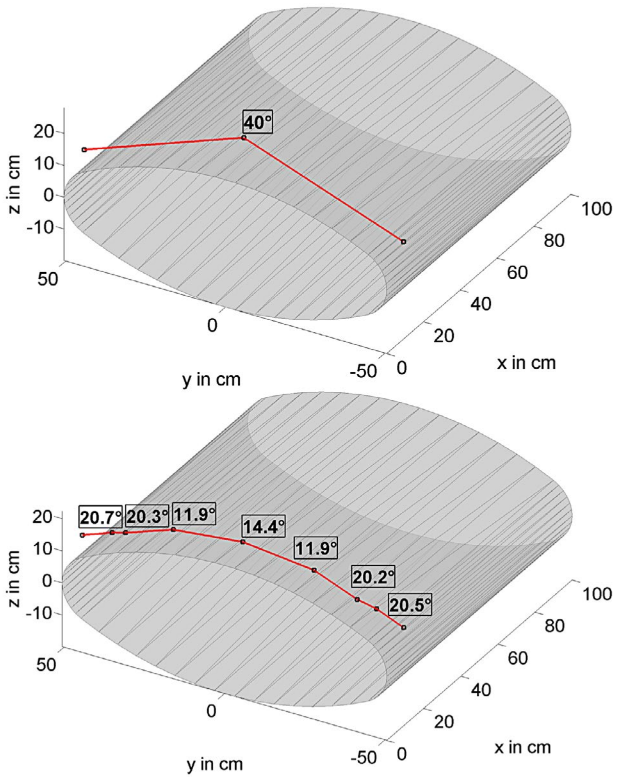


Figure 9. Top: Optimal routing for a non-metallic pipe with a standard bending angle of 40°. Bottom: A total bending angle of 120° increases the flexibility of the pipe.

4.3. Routing close to an obstacle

So far routings have been considered that pass along a convex hull causing the shape of the pipe to adopt the appearance of the hull. As it is illustrated in Figure 10 at the top, the routing behaves differently for the case of a concave hull. The optimal routing connects the starting and ending point with the shortest path possible causing the routing to depart from the hull.

Since pipes have to be fixed to the hull to avoid oscillations, it is necessary to push the routing to a surface. A

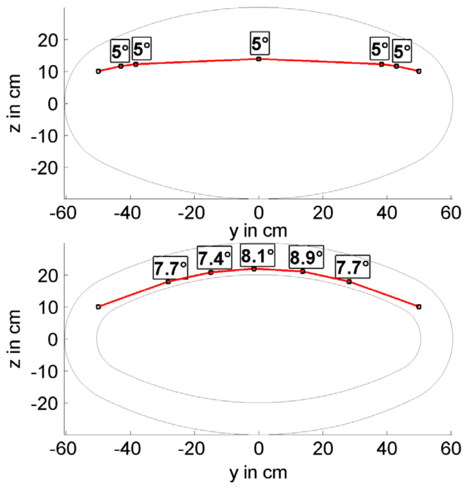


Figure 10. Top: Routing inside the hull leads to an almost direct connection. Bottom: An extra obstacle forces the routing to get closer to the outer hull.

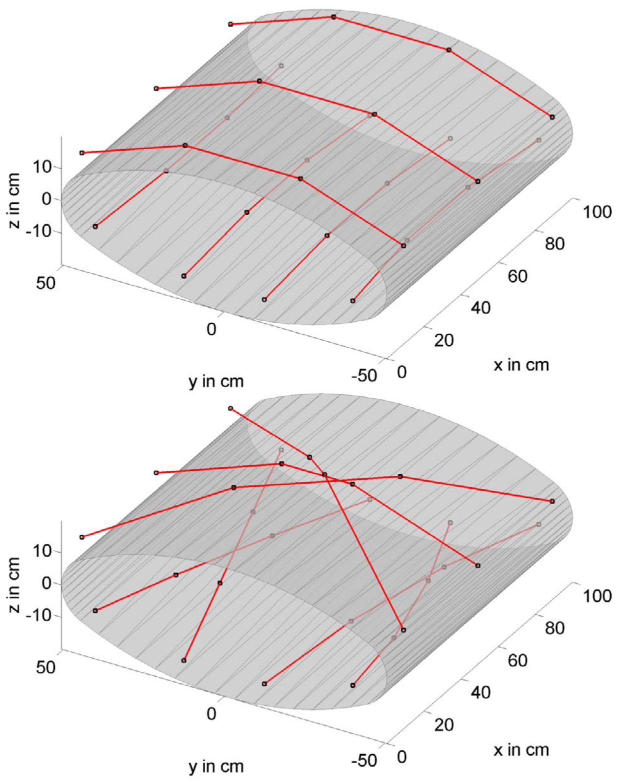


Figure 11. Top: Multiple routing of seven pipes that do not cross each other. Bottom: Multiple routing of seven pipes that cross each other.

possibility is to create additional obstacles. In the figure above a second elliptic hull was placed inside the original one. This obstacle prevents the direct connection which is the reason why the routing surrounds it and therefore comes closer to the original hull.

It should be mentioned that copying the original hull almost doubles the amount of constraints which increases the computational time. It should be questioned whether a few additional plates or other simple obstacles are sufficient to solve the problem.

4.4. Multiple routing

Sometimes a few pipes have to be routed simultaneously. In this case the cost functions Equations (5) and (16) have to be generalised such that they cover multiple pipes. This is described by Equation (37) for metallic pipes and by Equation (38) for non-metallic pipes.

$$m = \sum_{n=1}^{n_p} \left(\rho_{p,n} \frac{\pi}{4} \left(d_{o,n}^2 - d_{i,n}^2 \right) + \rho_{fl,n} \frac{\pi}{4} d_{i,n}^2 \right) l_n \quad (37)$$

$$m = \sum_{n=1}^{n_p} \left(\rho_{p,n} \frac{\pi}{4} \left(d_{o,n}^2 - d_{i,n}^2 \right) + \rho_{fl,n} \frac{\pi}{4} d_{i,n}^2 \right) l_n + 2k_n\, m_{\text{Con}} \quad (38)$$

In both equations the subscript n indicates that for each of n_p pipes the densities, the diameters, etc. might be different. However, in the frame of the following tests all parameters are set accordingly to Table 1.

In Figure 11 at the top a routing of seven pipes with two bends each is shown. As it can be seen the pipes are routed parallel as the starting and ending points are shifted equidistantly. At the bottom of Figure 11 the situation is changed. The starting and ending points are swapped such that a cross routing is created. The purpose of this is to illustrate that the collision avoidance between pipes is working.

Since the number of involved variables and constraints has been increased, this has a significant influence on the computational time. Both the system of equations and inequations in Equations (4a)–(4e) and the differentiation of the Lagrangian function in Equation (4a) become more extensive.

4.5. Network routing

Another import scenario is the optimal routing of networks which leads to a further reduction of the weight of pipes. From a mathematical point of view, networks represent a special case of multiple routings. The only difference to be considered is the fact that two line segments must touch each other which can be described with Equation (39). Here the distance between two chosen line segments g_{x1} and h_{y1} is zero.

$$\mathcal{D}\left(g_{x1}, h_{y1}\right) = 0 \quad (39)$$

The top illustration in Figure 12 shows a network routing with one branch (left pipe). Since the ending points are distributed equally the appearance of the routing is almost symmetric. In the bottom illustration of Figure 12 two branches are considered. As the touching points of the branches with the main route (the one in the centre) are at different line segments, the shape of the routing is not symmetric anymore. A further reason for this is the fact that additional constraints have been considered.

$$\sphericalangle\left(g_{x1}, h_{y1}\right) = 135^\circ \quad (40)$$

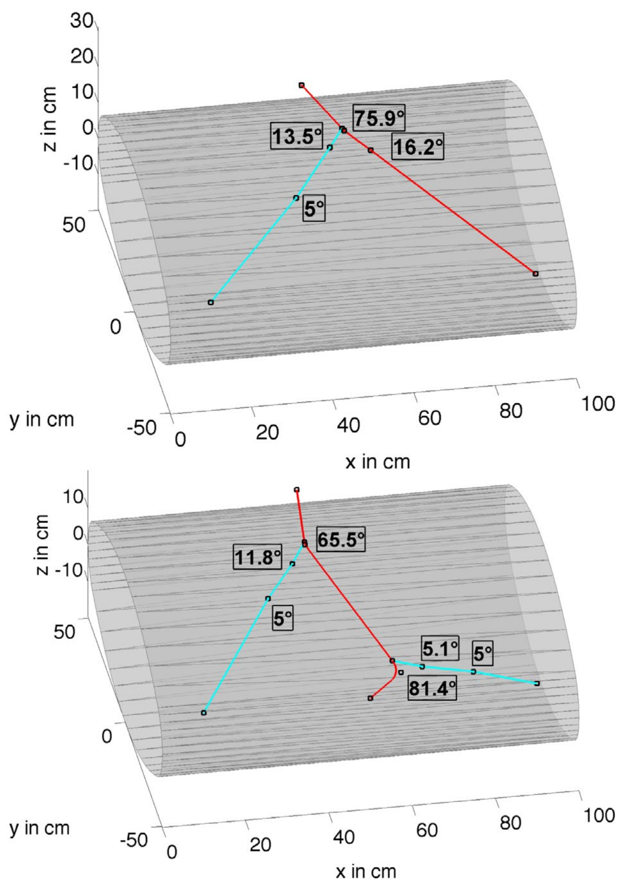


Figure 12. Top: Network routing with one branch. Bottom: Network routing with two branches and *Y*-junctions.

Table 1. General settings.

Parameter	Unit	Constraint	Bound
d_i	1.14 cm	δ_1	7.0 cm
d_o	1.27 cm	δ_2	7.0 cm
R	3.81 cm	δ_3	4.4 cm
ρ_{Sky}	0.99 g/cm^3	δ_4	5°
ρ_{Ti}	4.48 g/cm^3	δ_5	160°
ρ_{CFRP}	1.6 g/cm^3	δ_6	5°
m_{con}	5 g	δ_7	2.5 cm
		δ_8	1.7 cm

Table 2. Weight savings for non-metallic pipes.

Scenario	Bends	Weight (g)	Length (cm)
Non-metallic pipe	1	159.38	106.21
Metallic pipe	7	219.81	103.40

Table 3. Influence of the bending angles.

Scenario	Bends	Weight (g)	Length (cm)
Non-metallic pipe with standard bend	1	159.51	106.30
Metallic pipe with $\sum_j 2\beta_j = 120°$	7	221.28	104.09

$$\sphericalangle\left(g_{x2}, h_{y2}\right) = 135° \tag{41}$$

The equations above describe the angle between the touching line segments which is set to 135°. This is motivated by the fact that in practical standard connections like *T*-junctions (90°) or *Y*-junctions (135°) are preferred.

4.6. Routing along real structures

This section deals with the improvement of an existing routing by applying the abovementioned methods. As a testing assembly a metal sheet with eight ribs was chosen. In order to get rid of unneeded information like drill holes, screws, etc. only the silhouettes of the main structures were used which lowered the number of triangles significantly. The latter one is shown in Figure 13 at the top where also three metallic pipes (Ti-3Al-2.5V) with different diameters and bending radii are routed from left to right.

For the optimisation process the starting and ending points were adopted, however, the material was changed to CFRP. To keep the pipes close to the metal sheet two additional plates were created which can be seen in Figure 13 at the bottom. All in all the optimised route looks similar to the original one but saves around 126 cm of length, see Table 4. In combination with CFRP a weight reduction of 25% is achieved.

The optimisation was also performed for other assemblies where up to 14% of the total length and up to 30% of the weight was saved.

4.7. Computational time

Within this section the range of computational time varied along with the complexity of the problems. While for the simpler cases (up to six variables and 500 constraints) the computation took up to one minute, the time increased considerably for more complicated scenarios (30 variables and 3000 constraints, 20 min). The main reason for this are the computation and evaluation of Equations (4a)–(4e).

In order to accelerate the process some improvements may be developed. One of the most promising is to develop a strategy that distinguishes between necessary and unnecessary triangles (e.g. triangles that cannot collide with the pipe due to the distance or obstacles in between may be disregarded). This way the amount of triangles and constraints can be decreased significantly which will allow the routing along structures with several millions of triangles (and many more constraints). Another promising outlook is to transfer the computation of the structures to a GPU instead of a CPU.

5. Conclusions

This paper followed the aim to optimise the route of metallic and non-metallic pipes in order to minimise their weight. To do so the mathematical theory of non-linear optimisation was introduced and the NLP solver WORHP was used to solve the problem.

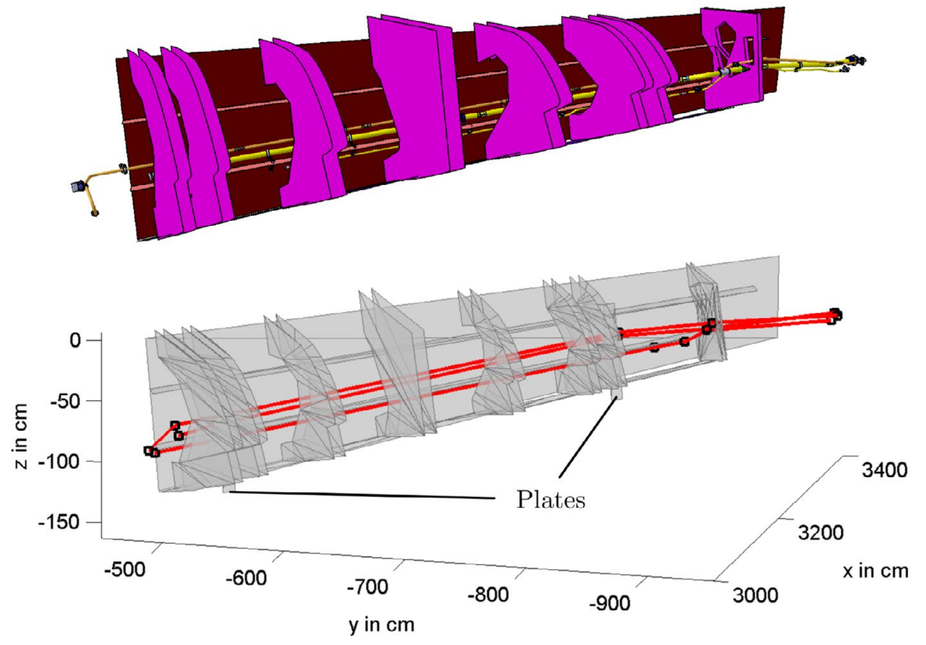


Figure 13. Top: Original routing of three pipes along simplified ribs. Bottom: Optimised routing which is kept close to the surface with the help of two additional plates.

Table 4. Original vs. optimised routing.

Scenario	Weight (g)	Length (cm)
Optimised – non-metallic	9174	1714.9
Original – metallic	12,097	1841.1

Modelling the cost functions as well as the equality and inequality constraints were in the focus of this paper since they represent the problem formulation in nonlinear optimisation. Of special importance was the development of distance functions which were needed to avoid collisions between pipes and obstacles.

In a series of practically motivated tests the functionality of the mathematical framework was proven. Routings

- for metallic and non-metallic pipes,
- with different bending angles,
- close to obstacles,
- with multiple pipes and
- with networks

revealed the capability of nonlinear optimisation and WORHP. Tests with real structure files demonstrated the advantages of the described method since existing routings where significantly improved.

The suggested method to minimise the weight of pipes allows a fully automated routing and is also usable for wiring or any other path finding problems in $\mathbb{R}^n$ with $n \geq 2$. In the future the acceleration of the computation will become of more interest. A promising approach is to bypass the usage of MATLAB® and to transfer the computation of the 3D structures to a GPU instead of a CPU.

Nomenclature

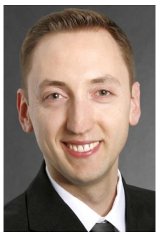
$\boldsymbol{a}$, $\boldsymbol{b}$	Random vectors in $\mathbb{R}^n$
b_j	Length of an arc (cm)
c	Constant
$\mathbb{D}(g_x, E_{tr})$	Function that describes the distance between g_x and E_{tr}
$\mathcal{D}(g_x, h_y)$	Function that describes the distance between g_x and h_y
$d(g_x, \boldsymbol{P})$	Function that describes the distance between g_x and $\boldsymbol{P}$
d_i, d_o	Inner and outer diameter (cm)
$E(\boldsymbol{x})$	Plane
E_{tr}	Triangle through $\boldsymbol{y}_j$, $\boldsymbol{y}_{j+1}$, $\boldsymbol{y}_{j+2}$
$f(\boldsymbol{x})$	Cost function
$g_i(\boldsymbol{x})$	Inequality constraint
g_x, g_{x0}, g_{x1}	Line segments through $\boldsymbol{x}_i$ and $\boldsymbol{x}_{i+1}$
$h_j(\boldsymbol{x})$	Equality constraint
h_y, h_{y0}, h_{y1}	Line segments through $\boldsymbol{y}_j$ and $\boldsymbol{y}_{j+1}$
k	Number of bends
$\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu})$	Lagrangian function
$l(\boldsymbol{x})$	Length of the pipe (cm)
m	Mass of the pipe (g)

m_{Con}	Mass of the connector (g)
n_p	Number of pipes
$\boldsymbol{P}$	Random point in $\mathbb{R}^3$
R	Bending radius (cm)
r, s, t	Parameters
$\boldsymbol{t}_{0,2}, \dots, \boldsymbol{t}_{k+1,2}$	Tangent points in $\mathbb{R}^3$
$\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{w}$	Direction vectors in $\mathbb{R}^3$
X	Set of feasible solutions
$\boldsymbol{x}$	Optimisation variable in $\mathbb{R}^n$
$\boldsymbol{x}_i, \boldsymbol{x}_{i+1}, \boldsymbol{y}_j, \boldsymbol{y}_{j+1}, \boldsymbol{y}_{j+2}$	Points of the line segments and triangles in $\mathbb{R}^3$
α, β, γ	Angles (°)
δ	Constraint value
$\boldsymbol{\lambda}$	Lagrangian multiplier in $\mathbb{R}^m$
$\boldsymbol{\mu}$	Lagrangian multiplier in $\mathbb{R}^p$
ρ_p	Density of the pipe's material (g/cm^3)
ρ_{CFRP}	Density of CFRP (g/cm^3)
ρ_{Sky}	Density of Skydrol LD-4 (g/cm^3)
ρ_{Ti}	Density of Ti-3Al-2.5V (g/cm^3)

Disclosure statement

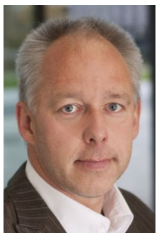
No potential conflict of interest was reported by the authors.

Notes on contributors

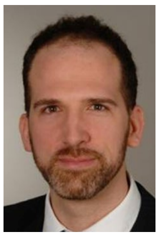


Friedrich Kohlmai received his Bachelor Degree in Mechanical Engineering at the University of Siegen in 2009 and his Master Degree in Industrial Mathematics at the University of Bremen in 2014. He was working on optimal routings within his Master Thesis and as a working student at Airbus Operations GmbH. His current research deals with the modelling, parameter identification and optimal control of ship engines.

Volker Baumbach received his PhD in Fluid Mechanics at the Drop Tower of the University of Bremen. He worked as post-doc at LEGI of the University of Grenoble and is currently working for Airbus as head of 'Hydraulic Performance and Integrity' group.



Christof Büskens studied Mathematics at the University of Münster where he specialised in optimal control theory. Since 2004 he is the head of 'Optimisation and Optimal Control' at the Centre for Industrial Mathematics (ZeTeM) at the University of Bremen. Amongst others he is supervising the development of the NLP solver WORHP.



Matthias Knauer studied Mathematics at the University of Bayreuth. Since 2004 he is working in the working group of 'Optimisation and Optimal Control' at the University of Bremen. His tasks cover optimal control problems which are usually connected with industrial applications.

References

Avriel, M., 2012. *Nonlinear programming: analysis and methods*. Mineola, NY: Dover.

Büskens, C. and Wassel, D., 2012. The ESA NLP solver WORHP. *Springer optimization and its applications*, 73 (4), 85–110.

Geiger, C. and Kanzow, C., 2002. *Theorie und Numerik restringierter Optimierungsaufgaben* [Theory and Numerics of restricted Optimisation Problems]. Berlin: Springer.

Kohlmai, F., 2014. *Optimal routing of pipes of hydraulic systems*. Unpublished thesis. University of Bremen.

Norvig, P. and Russell, S., 2009. *Artificial intelligence: a modern approach*. 3rd ed. Upper Saddle River, NJ: Prentice Hall Press.