UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería Geodésica, Cartográfica y Topográfica

Metodología y herramientas software basadas en SIG para la planificación de rutas para oleoductos. Un caso de estudio sobre una hipotética conexión entre Valencia y Alicante.

Trabajo Fin de Máster

Máster Universitario en Ingeniería Geomática y Geoinformación

AUTOR/A: Miñambres Vidal, Manuel

Tutor/a: Coll Aliaga, Peregrina Eloína

Cotutor/a externo: GUNTHER-DIRINGER, DETLEF

CURSO ACADÉMICO: 2022/2023

GIS-based Methodology and Software Tools for Oil Pipeline Routing Planning

A case study of an hypothetical connection between Valencia and Alicante

by

Manuel Miñambres Vidal

A Thesis Submitted to the Department of Cartographic Engineering, Geodesy and Photogrammetry in partial fulfillment of the requirements for the Degrees of

Geomatics Engineering and Geoinformation

and

Geomatics Science

at the

Polytechnic University of Valencia and Karlsruhe University of Applied Sciences

September 2023

Advisor: Dr. Salvador Bayarri Romar
Mentor: Dr. Eloina Coll Aliaga
Co-Mentor: Dr. Detlef Günther-Diringer

"The mystery of life isn't a problem to solve but a reality to experience."

–Frank Herbert, Dune

Acknowledgments

Me gustaría comenzar dando las gracias a las personas involucradas en el desarrollo de esta tesis. En primer lugar, gracias a mi tutora Eloína, por haberse ofrecido a tutorizarla. También agradecer a Ángel Marqués que, aunque no haya estado relacionado con la realización de esta tesis (pero sí con las prácticas en las que esta se enmarca), junto con Eloína, han supuesto un apoyo importante en mi desarrollo como profesional, y sobre todo como persona. Siguiendo en esta línea, pero cambiando de tercio, quiero agradecer con mención especial a Salvador por la ayuda brindada durante el desarrollo de esta tesis, así como por haberme dado la oportunidad de trabajar en Gilytics AG, empresa de la que sólo tengo buenas palabras. A su vez, agradecer a mis compañeros de dicha empresa. Gracias a Elena R, Elena C, Sonia, Jordi, Ramón, Alberto; y a los jefes Philippe y Stefano, por haberme ayudado, cada uno en mayor o menor medida, (pero todos aportando) a crecer como profesional. A su vez, saber que mi relación con ellos no acaba, sino que continuaré formando parte de Gilytics AG, es un hecho que me alegra enormemente.

Now, I want to thank the other responsible for this thesis, Prof. Günther-Diringer, who, in addition to being the co-mentor of this thesis, was in charge of the coordination for the Double Degree Program from the HKA side, one of the most transforming experiences I ever had. Taking advantage of writing in English now, I would like to mention some people who made this experience like I described before. Firstly, I would like to thank my classmates Lars and Felix, with whom I worked on many projects. Special mention to my floor mates in Hadiko, Benjamin, and Zihan. I won't forget our endless conversations about life, philosophy, physics, politics, or even aliens. Two truly friends I am proud to conserve despite the distance. Looking forward to see you again in our monthly call. Por último, gracias a Javi y Ana, compañeros con los que partí de Valencia al comienzo de esta aventura. Danke schön!

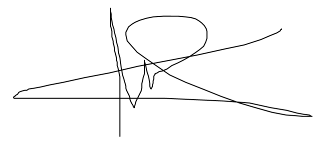
Dicho esto, ahora me gustaría mencionar a todas aquellas personas que, aun no habiendo estado relacionadas con la realización de esta tesis, o ni siquiera con los estudios que la motivan, sí que han sufrido de lleno sus consecuencias. Muchas horas de dedicación y estancias en el extranjero hacen que los que te quieren cerca no lo pasen especialmente bien. Estos son, en primer lugar, mi familia, de la que estoy profundamente orgulloso, y a la que agradezco todos los consejos dados que me han llevado a estar donde estoy. Y por otro lado, María, a la que doy las gracias por tener esa facilidad para despejar de mi cabeza toda preocupación, y así poder disfrutar plenamente del poco tiempo libre que tengo.

Para finalizar, me gustaría lanzar un gracias general a todas las personas que me rodean, las cuales me abstengo de mencionar por una cuestión práctica, pero que saben perfectamente que estas palabras son para ellos. Sois gente maravillosa.

- GRACIAS -

State of commitment

"The present document has been produced entirely by the undersigned; it has not been submitted as another previous academic work and all material taken from other sources has been appropriately quoted in quotation marks and cited in the text, as well as referenced in the bibliography".



Manuel Miñambres Vidal

Abstract

Among the initiatives shown in the presentation of the REPowerEU (EU proposed actions to rapidly reduce dependence on Russian fossil fuels and speed up the green transition), it is possible to see strong support for hydrogen as both a green and reliable alternative for Europe energy production. This idea is materialized in some points of the document, such as setting a target of 10 million tonnes of domestic renewable hydrogen production or a €200 million fund to accelerate hydrogen projects. This clarifies the necessity of planning, constructing, and exploiting new pipelines to supply the EU energy consumption demand.

Thus, this project aims to develop a GIS-based methodology and software tools to be integrated into Pathfinder (software for the optimization of infrastructures of the company Gilytics AG) for oil pipeline routing. This methodology covers data needed, pre-process and process of the data, optimal route calculation with Pathfinder, implementation and integration of additional geoprocesses for more realistic pipeline design and finally cost calculation. The methodology and developed tools are applied to a hypothetical connection between Alicante and Valencia.

Resumen

Entre las iniciativas mostradas en la presentación del REPowerEU (acciones propuestas por la UE para reducir rápidamente la dependencia de los combustibles fósiles rusos y acelerar la transición verde), es posible ver un fuerte apoyo al hidrógeno como alternativa ecológica y fiable para la producción de energía en Europa. Esta idea se materializa en algunos puntos del documento, como el establecimiento de un objetivo de 10 millones de toneladas de producción de hidrógeno renovable o un fondo de 200 millones de euros para acelerar los proyectos de hidrógeno. Esto pone de manifiesto la necesidad de planificar, construir y explotar nuevos conductos para abastecer la demanda de consumo energético de la UE.

Así pues, este proyecto pretende desarrollar una metodología y herramientas software basadas en SIG que se integrarán en Pathfinder (software para optimización de infraestructuras de la empresa Gilytics AG) para el trazado de oleoductos. Esta metodología abarcará los datos necesarios, el tratamiento previo y el proceso de los datos, el cálculo de la ruta óptima con Pathfinder, implementación e integración de geoprocesos adicionales para un diseño más realista de los oleoductos y, por último, el cálculo de los costes. La metodología y herramientas desarrolladas se aplicarán a una hipotética conexión entre Alicante y Valencia.

Contents

List of Figures

List of Tables

Listings

Acronyms

AHP Analytical Hierarchy Process.

BDN Banco de Datos de la Naturaleza.

BTN Base Topográfica Nacional.

DEM Digital Elevation Model.

EU European Union.

GA Genetic Algorithm.

GDAL Geospatial Data Abstraction Library.

GEOS Geometry Engine Open Source.

GIS Geographic Information System.

GLPK GNU Linear Programming Kit.

HILUCS Hierarchical INSPIRE Land Use Classification System.

ICV Instituto Cartográfico Valenciano.

IGN Instituto Geográfico Nacional.

LCP Least Cost Path Analysis.

LiDAR Light Detection and Ranging.

LP Linear Programming.

MCDA Multi-Criteria Decision Analysis.

MILP Mixed-Integer Linear Programming.

OSM Open Street Map.

PyPSA Python for Power System Analysis.

SA Simulated Annealing.

SIOSE Sistema de Información de Ocupación del Suelo de España.

SW Software.

TCMS Test Case Management System.

WFS Web Feature Service.

1 Introduction

The current document presents the memorandum of the Final Master's Degree Thesis belonging to the student Manuel Miñambres Vidal in partial fulfillment of the Dual Master's Degree Program in Geomatics Engineering and Geo-information offered by the Superior Technical School in Geodesy, Cartography and Land Surveying Engineering of the Polytechnic University of Valencia (UPV), and Geomatics Science offered by the Faculty of Information Management and Media (IMM) of the Karlsruhe University of Applied Sciences (HKA).

This thesis has been mentored by Professor Eloina Coll Aliaga, belonging to the Department of Cartography, Geodesy and Photogrammetry of the UPV, co-mentored by Professor Detlef Günther-Diringer, belonging to the IMM, and advised by Salvador Bayarri Romar, software team lead at Gilytics AG.

1.1 Thesis Scope

The scope of this thesis is, by its nature, the Geomatics Engineering and Geo-information, as its submission is required for the partial fulfillment of these studies. Among the different disciplines which compose this academic field, the master thesis will focus on GIS and Geoinformatics. The former consists of integrated computer hardware and software that store, manage, analyze, edit, output, and visualize geographic data, while the latter is the science and technology which develops and uses information science to address the problems of geography, geoscience, and more related with the purpose of the thesis, engineering.

GIS is widely used for urban planning, cartography, or natural resources management applications. In engineering, it is mainly used for finding optimal allocations for new infrastructures. Nowadays, the most popular GIS solutions are desktop applications such as QGIS or ArcGIS. This software allows the classic operations of the vector and raster GIS paradigm to be performed easily. Buffers and spatial differences can be examples of these operations.

However, these desktop GIS applications are general purpose, lacking specific industry requirements. Thus, engineering companies demand specific tools that could reduce costs by automatizing time-consuming tasks based on geographic information.

Therefore, under the pragmatic nature of the pipelines planning state of the art, which will be examined in a comprehensive way in State of the Art, this thesis will be focused on combining GIS and optimization methods to produce a methodology and software tools capable of solving and fulfilling the issues as mentioned earlier and necessities in the field of pipeline planning.

These optimization methods belong to Mathematics's branch known as Mathematical optimization or mathematical programming, which consists of selecting the best element regarding some criterion from some set of available alternatives.

1.2 Thesis Purpose

Having introduced the scope of this work, its purpose can be defined accordingly in a sequence of main objectives as follows:

- Establish the data required for new pipeline projects based on a comprehensive literature review.
- Address the process to obtain and pre-process such information and its sources.
- Determine the adequate criteria for such infrastructures based on a comprehensive literature review and the input of engineers working in the industry.
- Calculate the optimal route for the hypothetical connection using Pathfinder, comparing different algorithms and their configurations.
- Develop a geoprocess to determine the most optimal pipeline configuration. The aim is to calculate the optimal location for pumps and pressure reduction stations, pipeline sizes to be used in the pipeline, and a realistic construction cost.
- Demonstrate that the knowledge and skills gained during the master's degree are appropriately applied.

1.3 Document Outline

The present thesis is divided into nine sections as follows:

In section State of the Art, it is shown the current state of the specific matter of this thesis. A literature review has analyzed different approaches to optimal routing, specifically, optimal routing for fluid transportation pipelines.

Following, section Data lists the data used in the project, its sources, the collection process followed, and if needed, any edition process made.

Methodology section thoroughly explains the project's development, from its design to implementation and final validation.

After the explanation of the methodology, section Research results describes and shows the final approach achieved.

Later, in Case Study section, the tool will be used for a specific purpose, applying it in a simulated real need.

Budget section estimates the total cost for implementing the project, itemizing the different costs along with its current market prices.

Lastly, in Conclusions, a discussion takes place to analyze the extent to which the project objectives are achieved, as well as propose enhancements and ways to extend the project in the future.

The Bibliography section includes all the articles, books, and media links that encompass the references used in this work.

Appendix section holds the annexes of the project. More specifically, the cartography and code produced.

2 State of the Art

2.1 Geopolitical Context

In recent years, the European Union (EU) has been dedicated to a profound energy transition, signifying a pivotal shift from conventional fossil fuel-based energy systems toward sustainable and renewable energy sources (European Commission (2022)). However, a significant challenge arises from the intermittent nature of renewable energy sources, as their operational efficiency is contingent upon prevailing weather conditions. In response to this intermittency, two principal avenues have emerged as essential complements to bolster green energy solutions: nuclear energy and gas/oil.

Nuclear energy, generally regarded as an eco-friendly energy source, found limited favor in Germany's strategic energy plan, with the nation instead opting to emphasize gas-based alternatives. Historically, Germany had been sourcing gas from Russia (Oltermann (2022)). However, this energy arrangement encountered a seismic shift on the 24^{th} of February 2022, when Russia initiated its invasion of Ukraine. This geopolitical development prompted Germany and numerous other central EU nations that relied on Russian gas imports to seek alternative energy supply routes and sources promptly.

It is worth noting that implementing most of these energy alternatives necessitates meticulous planning and extensive infrastructure development, particularly involving establishing new grid systems and pipeline connections. At this juncture, the present thesis assumes significance and relevance.

2.2 Conventional vs. Innovative: Energy Infrastructure Design Paradigms

Traditionally, the approach to designing novel linear infrastructure within the energy sector has revolved around on-site exploration and the expertise of seasoned engineers. However, this conventional modus operandi is suboptimal, characterized by financial and temporal inefficiencies. The requirement for skilled professionals and on-site assessments from the project's inception contributes to elevated costs and sluggish progress. Furthermore, under the findings of Wang et al. (2019b), the conventional engineering paradigms have involved a stepwise development of oil pipelines, involving iterative attempts. Nevertheless, this incremental approach fails to assure the attainment of a global optimum and fails to account for the operational expenses linked to varying flow rates.

In contrast, the incorporation of remote sensing and Geographic Information Systems (GIS) offers a transformative avenue to reduce expenditures and expedite the entire process, obviating the need for physical inspections during the initial design phases and the presence of specialists. Adopting these technological tools affords decision-makers a rapid and enhanced comprehension of projects, supplemented by the assimilation of copious data, scenario simulations, and realistic financial estimations predicated on these simulations.

The inefficiencies intrinsic to the conventional methodologies for optimal pipeline routing predominantly stem from resource-intensive and protracted procedures. These approaches rely on unwieldy paper maps that lack precision and fail to accommodate many influential factors dictating pipeline routing comprehensively. Consequently, technical, economic, and environmental considerations must be more adequately addressed in delineating pathways, a deficiency attributed to the antiquated techniques employed. In this context, GIS tools introduce novel methodologies for routing, facilitating the holistic consideration and equitable weighting of all pertinent variables.

In alignment with the preceding, the principal objective of this thesis is to enhance the efficacy and performance of the approach mentioned above to calculating and devising fresh linear infrastructures, particularly in the context of gas/oil transportation pipelines.

2.3 Literature review

This section will show the reader the current state of the art of the main areas of knowledge on which the thesis relies. In general, the papers and scientific documentation consulted can be grouped into three: the ones that are focused on routing algorithms, the ones that talk about the typical constraints of pipeline projects for oil/gas transportation and the data to be taken into account, and the ones that define and solve optimization problems in the field of oil/gas transportation.

2.3.1 Data and Criteria

In Abudu and Williams (2015), the data is grouped in environment, construction, and security. In this case, the environmental criteria address minimizing the risks of groundwater contamination and maintaining a minor degrading effect on the environment, such as the effects on land cover, land uses, habitats, and sensitive areas, regarding construction criteria, maximizing the use of existing rights of way around roads and utility lines and maintaining routing within areas of low terrain costs. Finally, security criteria discuss the necessity to ensure access to the pipeline for maintenance and protection against vandalism. In this case, the weighting of the layers was made based on questionnaires to collate responses from experts. The experts considered land cover information, protected sites, geology, streams, and linear features more critical.

Moreover, Macharia and Mundia (2014) describes a model which incorporates several variables such as pipeline length, topography, geology, soil types, populated areas, game parks, forests, rivers, wetlands, roads, groundwater points, rail-line, and roads to identify an optimal route. All these variables are weighted using an AHP. It shares, too, a list of crucial routing factors such as keeping the pipeline away from the populated and settlement areas, minimizing crossing water bodies, utilizing existing linear disturbances, etc. More similar criteria can be found in Moreno-Bernal and Nesmachnow (2020).

In a similar way, Cruz-Chávez et al. (2020) makes use of the same layers as the ones estimated by Abudu and Williams (2015). Its weighting is similar too, so at this point, we can already determine, based on the literature, the layers to use in the present project and the key routing factors.

Apart from the aforementioned layers and criteria, the terrain plays an important role when planning the optimal route for a pipeline. Durmaz et al. (2019) shares a methodology to obtain four different categories of terrain based on a DEM: ridges, streams, flat terrain, and steep terrain. These topological classes are derived from slope and water accumulation maps, and the raster algebraic rules are shown in Table 1.

Table 1: Topographic classes

Steep terrain	Slope equal or greater than 20%
Ridges	Slope less tan 20% and water accumulation equal to 0
Flat terrain	Slope less than 20% and water accumulation between 0 to 100
Stream Channels	Slope less than 20% and water accumulation greater than 100

In addition to that, an AHP was run in order to determine the weight and priority of these classes, which result can be show in Table 2.

Table 2: Topographic weights

Category	**Priority**
Steep terrain	41.8%
Ridges	12.0%
Flat terrain	19.1%
Stream Channels	27.1%

2.3.2 Algorithms

Abudu and Williams (2015) investigated the practical implementation of the LCP (Least Cost Path) algorithm, emphasizing its relevance in terrain analysis. The LCP algorithm's ability to consider diverse impedance factors, such as elevation and slope, makes it an effective tool for identifying paths of minimal resistance. Their work underscores the algorithm's computational efficiency and accuracy, rendering it well-suited for navigating complex terrains. The use of this algorithm is done by Durmaz et al. (2019) as well. It is a graph-based algorithm that can be applied to raster data. Its simplest version is usually implemented by default in many GIS desktop applications.

Cruz-Chávez et al. (2020) contributed to the discourse by evaluating the utility of Simulated Annealing for route optimization. Their study highlighted SA's proficiency in traversing intricate solution spaces, leading to nearly optimal solutions. A key takeaway from their research is SA's adaptability across varied routing scenarios, endowing it with the versatility required to tackle real-world routing complexities.

Similar to the previous one, the Genetic Algorithm is an evolutionary optimization technique that explores a solution space by mimicking the principles of natural selection (woo Kim et al. (2022)). It is suitable for solving complex routing problems with dynamic constraints while handling diverse objective functions and constraints, which provides flexibility for adapting to changing conditions.

Moreno-Bernal and Nesmachnow (2020) undertook a comprehensive exploration of the A* algorithm's application in network routing. Their research underscored A*'s operational efficiency in identifying the shortest paths, combining the principles of Dijkstra's algorithm and greedy best-first search. The algorithm's real-time suitability and ability to uncover optimal routes are central elements, positioning it as a valuable asset in scenarios demanding swift decision-making.

In conclusion, the choice of algorithm for routing purposes depends on the specific characteristics of the problem, such as the nature of the network, the presence of constraints, and the desired level of optimality. The LCP is suitable for terrain-based analysis, and the GA and SA are effective for complex and dynamic scenarios. At the same time, the A* algorithm is well-suited for network-based routing tasks.

2.3.3 Optimization

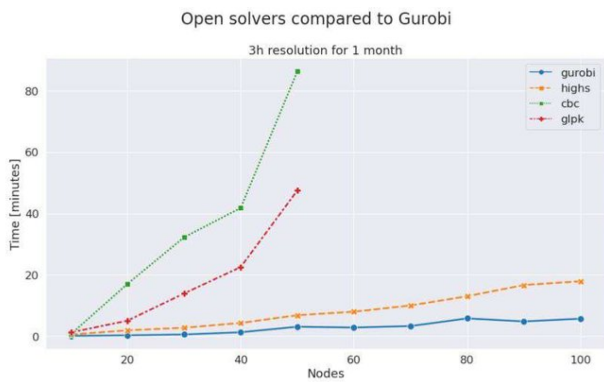
Regarding optimization problems, the research task of this thesis is to study the parameters that define a pipeline and, mainly, its construction cost.

In Wang et al. (2019b) is presented a method to determine the location, the operational plan of pump stations, and the location of pressure reduction stations, based on a stochastic MILP model, where the pressure control is taken into account. The problem is described as a sequence of nodes and arcs. There are four types of arcs: pipeline arc, pump station arc, fictitious arc, and pressure reduction station arc. This configuration is also shown in Wang et al. (2019c), but adding a heating station arc that controls the temperature of the fluid to be transported.

Both publications show an objective function representing the total construction cost, the different constraints to solve the optimization problem and ensure safe fluid transportation, and the variables that represent the final pipeline configuration once the problem is solved.

These two publications are essential for the thesis because, based on them, a cost function and MILP model are derived. The problem statement along with the MILP model finally developed for the sake of this thesis will be explained in the section Problem description and Mathematical model respectively.

Figure 1: Solvers comparison PyPSA



However, these researches do not consider any geo-information as input for the model; both consider the path as an already pre-calculated sequence of nodes and the distances that separates them, so it is part of the SW tool developed to integrate the MILP model in its GIS environment.

In addition, the two publications embrace the use of the well-known solver software Gurobi Optimization (2023). This software supports different programming languages and performs well in time and resource consumption. However, Gurobi is a commercial tool, and therefore it represents a cost for developing this tool, not for prototyping, as Gurobi offers a free trial, but for production. Thus, from an economic perspective, it is worth determining the feasibility of using a free open-source alternative.

Several comparisons have been made so far to determine which is the fastest solver in the market. One of the most relevant is the work conducted for several years by professor Mittelmann (2023) at the Arizona State University. He run different test to benchmark the most used solvers in the industry. The results of the last run can be shown in Table 3.

Table 3: MILP Benchmark

	CBC	**Gurobi**	**COPT**	**SCIP**	**SCIPC**	**HiGHS**	**Matlab**
unscaled	1328	81.5	164	888	727	715	2715
scaled	16.3	1	2.01	10.9	8.92	8.77	33.3
solved	107	227	204	137	152	158	72

In this case, Table 3 shows the unscaled and scaled shifted geometric means of run times and the number of problems solved. Apart from Gurobi and COPT (both commercial solutions), the most fast and consistent open source solver is HiGHS. This was also substantiated in Parzen et al. (2022), based on the benchmarks run by the PyPSA-Eur (2021) community. Figure 1 illustrates the performance of HiHGS relatively to GLPK and CBC (both open-source solution) and the aforementioned Gurobi.

It is possible to conclude that HiGHS represents the best alternative among the open-source model solvers, and that HiGHS's performance is comparable with that of Gurobi too. But out of the scientific research, when dealing with real problems (were the number of variables is quite larger), the gap, in terms of time, between the commercial and the current open-source solutions increase exponentially with the number of variables.

To conclude, open source solvers can do the job, and among them, HiGHS seems to be the most promising, but it is worth it to develop the SW tool in a way that a possible change of solver does not require much effort.

3 Data

In this section, a closer look at the different layers incorporated into the project will take place. These encompass the data gathered and the code, libraries, and commands used to edit them. This exploration is divided into two main subsections: Data Sources, where the sources from which the information is gathered are explained; and Dataset, where a detailed list of the various layers used, along with the specific commands that contributed to their creation are presented. This dual examination clearly explains how data and technology converge in the project's development.

3.1 Data Sources

3.1.1 IGN

The Instituto Geográfico Nacional of Spain is the national geographical institute responsible for cartographic and geodetic activities in the country. Established in 1870, its main duties involve the production, maintenance, and dissemination of accurate and up-to-date geographic information and topographic maps of Spain and its territories. As a key player in Spain's geospatial ecosystem, the CNIG provides access to its data and cartographic products to government entities, businesses, academia, and the public.

BTN

The Base Topográfica Nacional serves as a fundamental cartographic dataset for the country. It provides a comprehensive and detailed representation of Spain's topographic features, including terrain, roads, rivers, settlements, and administrative boundaries.

It is known for its high level of accuracy, with various scales available, ranging from 1:2,000 to 1:25,000. As consulted in IGN (2023a), the BTN's data acquisition involves a combination of field surveys, remote sensing, and data contributions from authoritative sources. Field surveys are conducted to capture specific features, while remote sensing techniques, such as satellite imagery and airborne LiDAR, are utilized to cover large areas and collect elevation data. Additionally, the IGN relies on data contributions from regional authorities and municipalities, fostering a collaborative approach to keep the dataset up-to-date.

SIOSE

The Sistema de Información de Ocupación del Suelo de España (SIOSE) is a comprehensive geographic information system managed by the Instituto Geográfico Nacional (IGN) of Spain, offering detailed and accurate data on land use and land cover across the country. Through advanced remote sensing technologies and expert interpretation, SIOSE classifies diverse land features such as urban areas, agricultural zones, forests, and water bodies. The system's rich and up-to-date data aids in informed decision-making for urban planning, environmental management, natural resource assessment, and disaster response, contributing to sustainable development and effective policy formulation throughout Spain.

3.1.2 BDN

The Banco de Datos de la Naturaleza (BDN) (Nature Database) is a repository of comprehensive and diverse ecological and environmental data related to Spain. It serves as a centralized information hub for various aspects of the natural world, including flora, fauna, ecosystems, and land use. Managed by relevant scientific and governmental institutions, the database supports research, conservation efforts, policy-making, and informed decision-making in areas such as biodiversity preservation, ecological studies, and sustainable resource management.

MFE

The Mapa Forestal de España at a 1:50,000 scale (MFE50) is a cartographic representation of the status of forest masses, created by the Banco de Datos de la Naturaleza (Nature Database), based on a hierarchical conceptual model of land uses, particularly focused on forested areas.

The database consists of a range of descriptive fields regarding the ecology and structure of these forest masses. Within the forested land use, up to three different tree species are considered, each with its developmental stage (reforestation, wild growth, under-story, and canopy), occupancy (percentage of the species in relation to total trees), and the covered capacity fraction for the entire forested area (percentage of ground covered by the horizontal projection of tree canopies).

3.1.3 OSM

The Open Street Map is a collaborative and crowd-sourced mapping project that aims to provide free, open, and detailed geographic data for the entire world. Founded in 2004, OSM relies on contributions from millions of volunteers worldwide who use GPS devices, aerial imagery, and other sources to map roads, buildings, landmarks, and various geographical features.

3.1.4 ICV

The Instituto Cartográfico Valenciano (ICV), also known as the Valencian Cartographic Institute, is an organization based in the Valencian Community of Spain. Its primary purpose is to produce and manage cartographic information and geospatial data related to the Valencian region. This includes creating maps, geographical databases, and other spatial information resources.

IDEV

It refers to a framework or system that facilitates the discovery, access, sharing, and use of geospatial data across various organizations and sectors within the Valencian Community of Spain.

The primary goal of IDEV is to promote the integration and interoperability of geospatial information from different sources, such as government agencies, local authorities, research institutions, and private companies. By establishing standardized protocols, metadata, and data sharing mechanisms, IDEV aims to make geospatial data more accessible and usable for decision-making, planning, analysis, and research purposes.

3.2 Datasets

All the layers used in the project are retrieved from one of the aforementioned data sources. The layers obtained will be part of the datasets, which are divided accordingly to what has been exposed in Data and Criteria section. Some data sources offer the information in a suitable way for the project and any other process than the download itself is needed, for instance, this is the case of the OSM information, which is fully integrated in Pathfinder. However in the rest of the cases there is a need for a previous reorganization of the data in order to integrate it in Pathfinder.

The BTN information is served through the downloads center of the CNIG (2023). The information in the BTN is organized in provinces, and to each province corresponds a *ESRI Shapefile* for each feature present in that province.

This information will be used later in Pathfinder through a connection to a WFS service. Because of this, we are not going to simply select the two provinces object of the case study, but the entire extension available, so the data can be reused in other projects inside Pathfinder. Thus, in order to obtain a unique layer for each feature (road, rivers, etc), a first process has to be done. In this case, we can use the GDAL library and its command *ogrmerge.py*.

```bash
#!/bin/bash

LAYERS=("RIO" "CARRETERA")

for f in *.zip
  do unzip "$f" -d "${f%.zip}"
done

for LAYER_NAME in "${LAYERS[@]}"
  do ogrmerge.py -single -o merged_$LAYER_NAME.json ../BTN/**/*_$LAYER_NAME.shp
done

```

Listing 1: From BTN to merged layers example

In Listing 1 there is an example of how to use this command along with some other actions to, from the list of ZIP files, directly downloaded, create a new GeoJSON file that contains all the information available in each of the provinces for the desired layers. The relation between the table name and the feature is described in IGN (2023a).

To finalize, all the layers created, and the ones directly downloaded, have to be uploaded to the Pathfinder GeoNode server, in order to make use of a WFS. In the upload process, the mandatory metadata is properly set to comply with the use conditions of the data providers. This is, *Work derived from BTN 2022 CC-BY* for the layers derived from the BTN. In addition to that, the layers are named using the corresponding ISO 3166-2 code of the county, autonomous community or province along with a short name that summarizes the content of that layer.

3.2.1 Environment

Forest Layer of Forest in the Valencian Community derived from the MFE database. Layer generated filtering the features which definition is Forest and merging the information of the three VC's provinces.

Humid Areas Layer of Humid areas in Spain derived from the BTN database from the IGN. Layer corresponding to Table 0319S_Humedal.

Natural parks Layer of Natural parks in Spain derived from the BTN database from the IGN. Layer generated filtering the features of Table 0107S_ZON_PRO which type is National Park or Park.

Protected sites Layer of Protected sites in Spain derived from the BTN database from the IGN. Layer corresponding to Table 0107S_ZON_PRO.

Protected landscapes Layer of Protected landscapes in Spain derived from the BTN database from the IGN. Layer corresponding to Table 0107S_ZON_PRO, generated filtering the features which type is Protected sites.

3.2.2 Hydrology

Lakes Layer of Lakes in Spain derived from the BTN database from the IGN. Layer generated combining Lagoon (Table 0316S_Laguna) and Reservoir (Table 0325S_Embalse).

Rivers Layer of Rivers in Spain derived from the BTN database from the IGN. Layer corresponding to Table 0302L_Río

Sea Layer of Sea in Spain derived from the BTN database from the IGN. Layer corresponding to Table 0306S_Aguas_marinas.

3.2.3 Infrastructures

Airports Layer corresponding to the OSM features with the tag aeroway=aerodrome and aeroway=terminal.

Helipads and Heliports Layer corresponding to the OSM features with the tag aeroway=helipad and aeroway=heliport.

Oil and gas pipelines Layer of Oil and Gas Pipelines in Spain derived from the BTN database from the IGN. Layer corresponding to Table 0701L_CON_COMB.

Power lines Layer of Power lines in Spain derived from the BTN database from the IGN. Layer generated filtering the features of Table 0710L_LIN_ELEC which electric tension is between 100kV and 150kV.

Railway Layer of Railway High Speed in Spain derived from the BTN database from the IGN. Layer corresponding to Table 0641L_FC_CONV.

Railway High Speed Layer of Railway High Speed in Spain derived from the BTN database from the IGN. Layer corresponding to Table 0638L_FC_ALT_VEL.

Roads Layer of Railway High Speed in Spain derived from the BTN database from the IGN. Layer corresponding to Table 0605L_CARRETERA.

3.2.4 Settlements

Archaeological sites Layer of Railway High Speed in Spain derived from the BTN database from the IGN. Layer corresponding to Table 0558P_YAC_ARQ.

High cultural value sites Layer of High cultural value sites in the Valencian Community. Layer corresponding to the layer Bienes de Interés Cultural de la Comunitat Valenciana from the IDEV.

Urban areas Layer of Urban areas in the Valencian Community. Layer derived from the urban planning layer from the IDEV. Layer generated filtering the features with classification equal to urban or for development.

3.2.5 Terrain

Agricultural Land Layer of Agricultural Land Use areas in Spain derived from the SIOSE database from the IGN. Layer generated filtering the features with a HILUCS code equal to 110 (1_1_Agriculture)

Clay Layer of Clay soils presence in the Valencian Community. Layer derived from the Mapa Geológico de España a escala 1:50.000 (3ª Serie) del IGME filtering the features which description contained 'clay'.

DEM Digital Elevation Model (DEM) derived from the MDT05 of the Instituto Geográfico Nacional (IGN). The DEM was obtained after a resampling to 10 meters resolution of the original.

Slope Slope map derived from the aforementioned DEM. Generated using the following GDAL command:

```
gdaldem slope MDT_malla_10m_etrs89h30.ers slope_map.tif

```

Listing 2: Command to generate a slope map from a DEM

Flow Accumulation Map Map derived from the generated DEM with the following command of the SAGA library:

```
saga_cmd ta_hydrology 0 -ELEVATION ../DEM/DEM_Clipped.tif -FLOW
    flow_accumulation_cells.tif -FLOW_UNIT 0

```

Listing 3: Command to generate a flow accumulation map

Flat terrain Map derived from the generated Slope map with the following GDAL command:

```
gdal_calc.py -A Slope_VC_area.tif -B flow_accumulation_cells.tif --calc="
    logical_and(A<20,logical_and(B>0,B<=100))" --NoDataValue 0 --extent="
    intersect" --overwrite --outfile Flat_terrain.tif

```

Listing 4: Command to generate a flat terrain map

Ridges Map derived from the generated Slope map with the following GDAL command:

```
gdal_calc.py -A Slope_VC_area.tif -B flow_accumulation_cells.tif --calc="
    logical_and(A<20,B==0)" --NoDataValue 0 --extent="intersect" --
    overwrite --outfile Ridges.tif

```

Listing 5: Command to generate a ridges map

Stream channels Map derived from the generated Slope map with the following GDAL command:

```
gdal_calc.py -A Slope_VC_area.tif -B flow_accumulation_cells.tif --calc="
    logical_and(A<20,B>100)" --NoDataValue 0 --extent="intersect" --
    overwrite --outfile Stream_channels.tif

```

Listing 6: Command to generate a stream channels map

4 Methodology

This section describes the methodology followed during the realization of this project. Firstly, the mathematical model for the optimization problem will be explained. Later, it is going to be described the SW development steps followed in order to implement, integrate the model into Pathfinder, and test it.

4.1 Problem description

As stated by Wang et al. (2019b), it is possible to simplify the problem accordingly to the following schema shown in Figure 2, where these nodes are used to check the pressure of the whole pipeline system to ensure safe transportation. The pipeline arc only permits the pipeline's construction, and the station arc permits one of the three types of station to be constructed. In this project, a different definition of the problem will be done, where fictitious stations are omitted, and some constraints are reformulated.

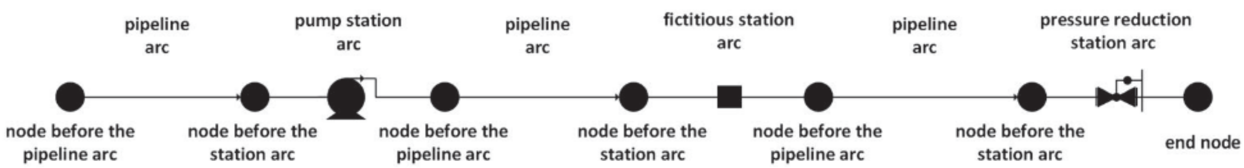


Figure 2: Nodes and arcs schema

Thus, the problem described in this study is stated as follows.
Given:

- The elevation along the pipeline route
- The available size, price per unit length, and pipeline design pressure
- The available pump operational plan
- The construction cost of the pump station and the pressure reduction station
- The flow rate of the pipeline

Determine:

- The inner diameter of the pipeline in each segment
- The locations of the pump stations and the pressure reduction stations
- The pump operational plan of the pump stations

To effectively build and solve the model, the following assumptions are made:

- The model focuses on a single pipeline, which only has starting and end nodes.
- The friction item for the pressure drop formula is pre-calculated according to the fluid type and available diameters of the pipeline.

4.2 Mathematical model

The problem presented is a constrained optimization problem. Therefore, an objective function to be maximized or minimized has to be defined, as well as some constraints to limit the possible optimal values.

4.2.1 Objective function

The objective function (Eq. 1) proposed represents the total construction cost of a pipeline. Thus, the objective is to minimize it. This construction cost is composed of the construction cost based on the length of the pipeline and its unitary price per length, a second component based on the number of pump stations and its unitary construction cost, and a third component based on the number of pressure reduction stations and its unitary construction cost. The objective function presented is based on the one used by Wang et al. (2019b), but with minor modifications.

$$\min f = \sum_{j \in \mathrm{L}_a} \sum_{g \in \mathrm{G}} \phi_j T_{\mathrm{P}_{j,g}} C_{\mathrm{UP}g} + \sum_{j \in \mathrm{L}_a} T_{\mathrm{D}j} C_{\mathrm{UD}} + \sum_{j \in \mathrm{L}_a} \sum_{y \in \mathrm{Y}} T_{\mathrm{Z}j,y} C_{\mathrm{UZ}} \tag{1}$$

In order to know the meaning of the symbols used in the aforementioned equation and the following ones, see section Nomenclature.

4.2.2 Constraints

Now, with the objective function defined, some constraints will be added to the model in order to ensure safe fluid transportation along the pipeline. Note that the way used to introduce these constraints corresponds to the advanced modeling technique known as the big M constraints method, which is a technique to incorporate logical conditions or restrictions into a linear programming problem. It involves introducing a large positive constant (often denoted as "M") to represent the cost or penalty of violating a particular constraint. The method effectively enforces the desired logical conditions by formulating an auxiliary variable that becomes active only when the original constraint is violated (Rardin (1998)).

Pipeline hydraulic constraints

Pipeline pressure drop

$$p_i - p_{i+1} - C_{\mathrm{D}_g} Q_{\mathrm{A}} \phi_j \geq -M((1 - T_{\mathrm{P}_{j,g}}) + \sum_{y \in \mathrm{Y}} T_{\mathrm{Z}j,y} + T_{\mathrm{D}j}), \forall i \in N_a, j \in L_a, i = j, g \in G \tag{2}$$

$$p_i - p_{i+1} - C_{\mathrm{D}_g} Q_{\mathrm{A}} \phi_j \leq M((1 - T_{\mathrm{P}_{j,g}}) + \sum_{y \in \mathrm{Y}} T_{\mathrm{Z}j,y} + T_{\mathrm{D}j}), \forall i \in N_a, j \in L_a, i = j, g \in G \tag{3}$$

Equations 2 and 3 determine the pressure drop between two consecutive nodes. This pressure drop is computed with a flow-related equation using the pressure between the two nodes, the horizontal distance between them, and the pressure drop coefficient: $C_{\mathrm{D}_g} = \varepsilon(\nu^m/d^{5-m})\rho g$. Note that these expressions 2 and 3 differ from the ones used by Wang et al. (2019b). The reason is that the original constrain formulation was in conflict with some other constrains, making the model infeasible by nature. Adding the other two node types into the equation solved the issue.

Pressure reduction station

$$p_i - p_{i+1} - C_{\mathrm{F}_{j,a}} \geq M(T_{\mathrm{D}_j} - 1), \forall i \in N_a, j \in L_a, i = j \tag{4}$$

$$p_i - p_{i+1} - C_{\mathrm{F}_{j,a}} \leq M(T_{\mathrm{D}_j} - 1), \forall i \in N_a, j \in L_a, i = j \tag{5}$$

In this case, equations 4 and 5 compute the continuous variable pressure value reduced by a pressure relief valve placed between the given nodes.

Pump station

$$p_i - p_{i+1} + \alpha_y \geq M(T_{\mathrm{Z}_{j,y}} - 1), \forall i \in N_a, j \in L_a, i = j, y \in Y \tag{6}$$

$$p_i - p_{i+1} + \alpha_y \leq M(T_{\mathrm{Z}_{j,y}} - 1), \forall i \in N_a, j \in L_a, i = j, y \in Y \tag{7}$$

The limitations on pressure augmentation within the pump stations are indicated in equations 6 and 7, where the pressure undergoes an increment from node i to node i + 1. This augmentation is formulated in terms of the pump characteristics.

Arc constraints

$$\sum_{g \in \mathrm{G}} T_{\mathrm{P}_{j,g}} = 1, \forall j \in L_a \tag{8}$$

Equation 8 ensures that at least one pipeline size is chosen for each arc of the pipeline.

Node pressure constraints

Design pressure constraints

$$p_i - z_i \leq W \sum_{g \in \mathrm{G}} T_{\mathrm{P}j,g} P_{\mathrm{G}g}, \forall i \in N_a, j \in L_a, i = j \tag{9}$$

Equation 9 ensures that the difference between a node height and its pressure is always within the chosen pipeline segment pressure design. With variable W it is possible to add some extra safety margin.

$$p_i - z_i \geq P_{\mathrm{D\ min}}, \forall i \in N_a \tag{10}$$

Equation 10 ensures a minimum pressure along the pipeline.

Maximum allowable elevation difference constraints

$$C_{\mathrm{F}j} \leq M \times T_{\mathrm{D}j}, \forall j \in L_a \tag{11}$$

This constrains ensures that if a pressure drop exists in arc j, then a pressure drop station must be built.

Apart from the constraints shown above, some other constraints are added to the model, although they will not always be applied in the solver, as they correspond to constraints that the user may or may not choose to use. These constraints are the initial pressure at the first node of the pipe and the maximum pressure at the last node.

$$p_0 = P_s + z_0 \tag{12}$$

$$p_n - z_n \leq P_e \tag{13}$$

In equation 12, p_0 and z_0 represent the pressure and elevation of the starting node respectively. In equation 13, p_n and z_n represent the pressure and elevation of the last node in the pipeline respectively.

4.3 Software Tool development

The tool has not been developed - at least in strict terms - under any specific SW development methodology. However, the stages defined have resemblances with the Waterfall method. This method consists of a linear and sequential methodology with distinct phases that follow a strict order. Each phase must be completed before the next one begins (Royce (1970)). The student, aware of the weaknesses of this methodology (rigidity and inflexibility, limited user involvement or late detection, among others), has chosen it because of its simple structured approach, ease of management, and well-suited for small projects, and, last but not least, because he is fully aware that the more significant SW project for which this tool is intended (Pathfinder), already has a better methodology to which it will adhere once the tool is fully integrated.

With all the above, the sequence of phases is, firstly, a requirements gathering and system design. This part corresponds to what is exposed in section 4.1—secondly, a prototyping phase focus on creating a preliminary version of the software. The Product Manager and the developer discuss this prototype, and any necessary changes are made to refine it. Thirdly, the integration phase, where the individual components or modules developed by different teams or developers are integrated to form the complete software system. This involves making sure that the different parts of the software work together seamlessly and that interfaces between components are well-defined and functional. Last but not least, a testing phase aims to identify and rectify any defects or issues before deployment takes place.

4.3.1 Prototyping

In this phase, the main aim was to obtain a standalone script capable of achieving the most basic aspirations of the final tool. The main features were: a model-solving section, input and output for the geospatial data, and some visual outputs.

This standalone script was coded in Python, as this is the language used by Django, the framework used for the development of Pathfinder. There are many libraries for defining and solving optimization problems. In this case, PuLP was the library chosen based on its ease of use, open source and free nature, flexible problem formulation, constraints expressiveness, and mainly its support for multiple solvers, as already discussed in Optimization section.

During this stage of the project, it was crucial to determine the feasibility of applying a MILP solver to actual spatial data. Some tests were performed, and the conclusion was that the default PuLP solver (COIN-OR Foundation (2016)) was not as optimal as needed. This was unsurprising, as it was commented in chapter Optimization. So the default PuLP solver was substituted by HiGHS. This solver has the possibility to tune the solver's behavior for the sake of performance. Very often, the solvers spend too much time trying to achieve a solution for the most optimal value when actually, the current solution is not that far from the optimal. In practice, finding the exact optimal solution for complex MILP problems can be computationally infeasible due to the discrete nature of the integer variables. The optimality gap allows users to make informed decisions about whether to continue the optimization process or stop it based on the quality of the current solution. It's particularly useful when solving large-scale MILP problems, where finding the exact solution might be time-consuming or even impossible within a reasonable time frame. The optimality gap is the difference between the objective value of the best solution found by the solver and the objective value of the true optimal solution. In other words, it measures how far the current solution is from being the best possible. The gap is usually expressed as a percentage or an absolute value.

However, in order to add this gap parameter to the solver, a modification in PuLP source code was needed. The PuLP contributor community later added this modification in commit #641 - HiGHS API interface improvement including time_limit in PuLP public repository `https://github.com/coin-or/pulp`, but at the moment of writing this memorandum, the changes are not published in a release, thus this has to be taken into account when integrating the tool in Pathfinder, which installation of the library has to point to the specific commit hash.

For managing the spatial data, the Python binding of GDAL was used. The spatial data in this phase came from a *ESRI Shapefile* containing a testing path calculated previously in Pathfinder and exported. Two point layers are created: one for the pump location and the other for the pressure reduction stations.

In addition to the aforementioned libraries, Matplotlib was used to visualize the results in a way that is understandable at first glance. The plots resulting are similar to Figure 3, where it is possible to see how the solver adjusts the pressure to be close to the path elevation (always between the safety intervals), the selection of two different pumps with different power capacity and the pressure drop along the pipeline.

Apart from the visual results, the pipeline sizes chosen for each pipeline segment are exported in a formatted table created with the well-known Python library for data manipulation and analysis named Pandas.

Once the basic functionalities were proved to be feasible, the next step was to integrate it into Pathfinder.

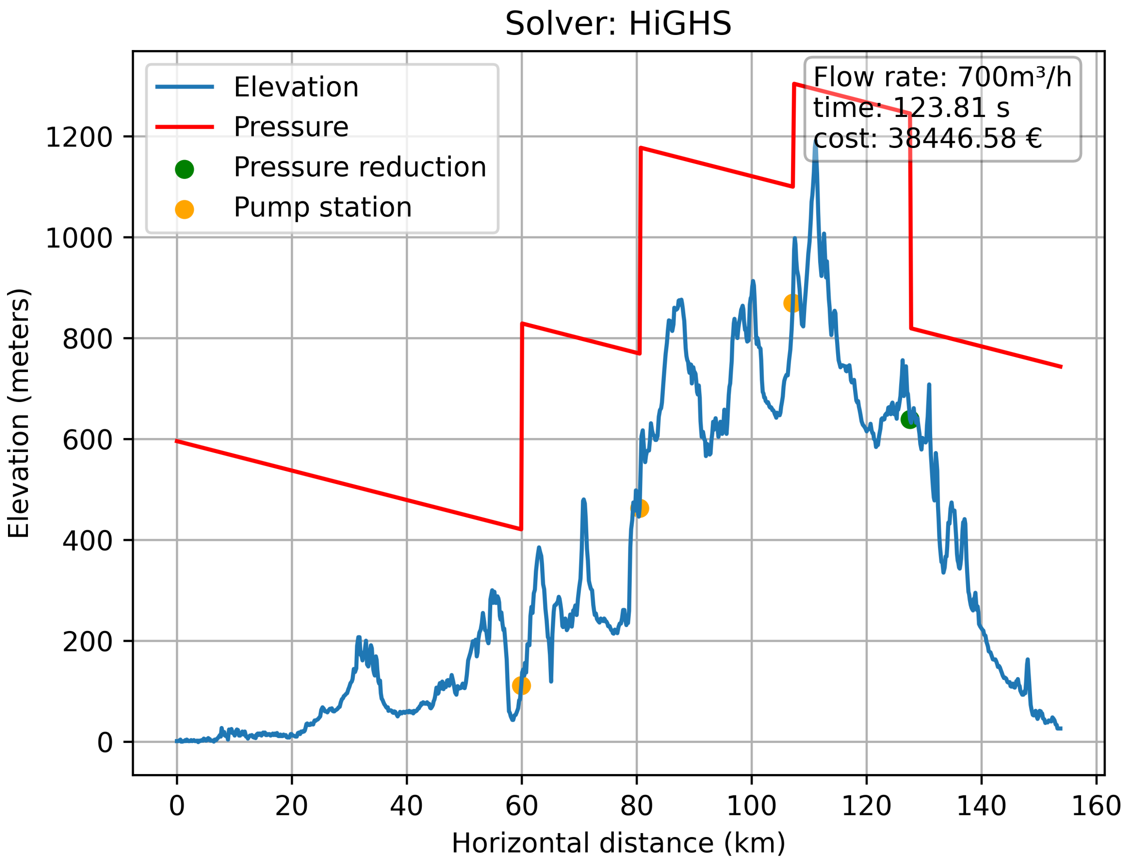
4.3.2 Integration

The first step for integrating the tool in Pathfinder is to install all the libraries needed - PuLP and HiGHS - in the server that runs the application. The first library can be installed by adding it to the requirements.txt file that stores all the Python libraries needed to run the application. As mentioned before, the release of that library did not contain the last modifications needed for this project. Thus it was needed to point to the specific git hash of the aforementioned commit in the requirements.txt. In the case of HiGHS, its installation was made using Make.

Once the libraries are installed, some modifications to the prototype have to be made. Pathfinder Back End relies on Django. This Python framework has a library called GEOS, an open-source software library for performing geometric operations on spatial data. Even though this library is similar to GDAL, or even relies on GDAL, the geometry treatment is different.

The other significant modification is the input and output process. The integration of the tools is going to be made using the geoprocessing part of Pathfinder. A geoprocess consists of some input data, usually from the current project where the user is working, some parameters to configure the behavior of the geoprocess, and the output, which is automatically integrated into the project. Thus, the input parameters are no longer hardcoded but retrieved as Parameter objects. The same happens for the output, which is no longer an ESRI Shapefile but a Vector Parameter object into which the data will be set.

Figure 3: Prototyping results: Plot



The work was made creating a new branch of the main project. All the code was written in this branch, along with some modifications to the existing infrastructure. After that, a Pull Request (PR) was made to the main branch, and a review from the repository's maintainers took place.

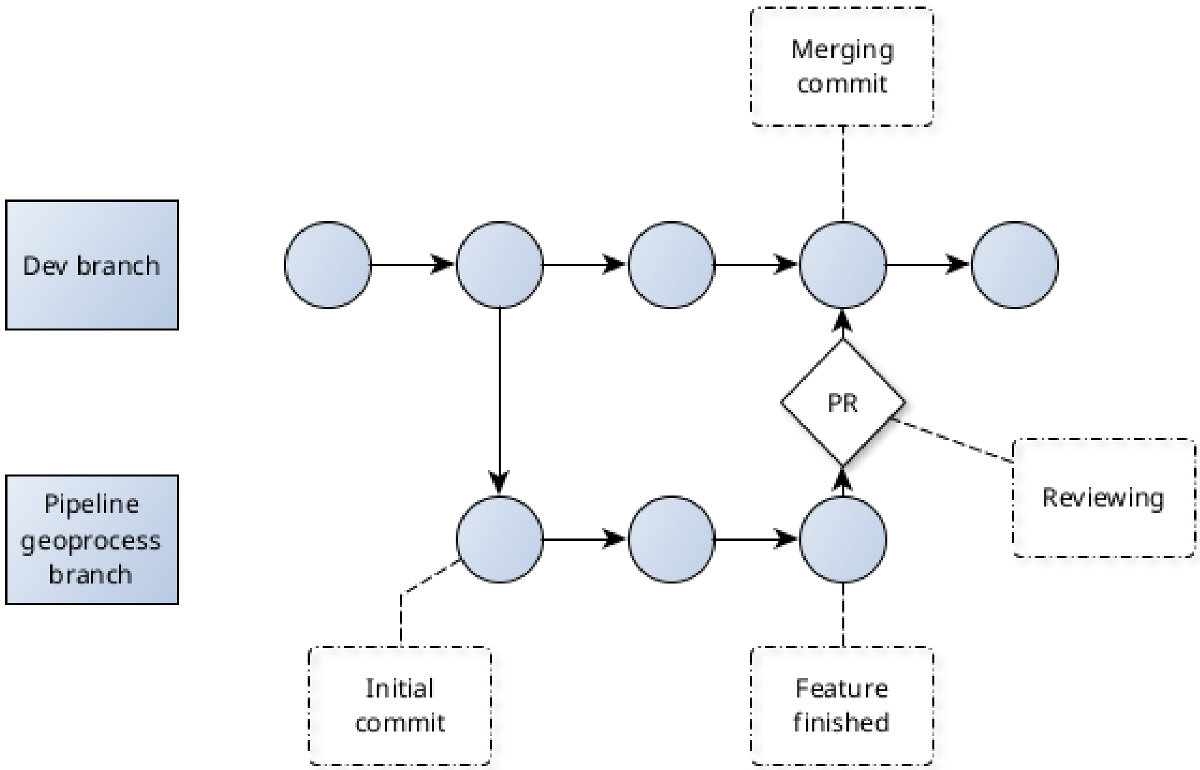
The purpose of that PR is to initiate a discussion, review, and eventual incorporate those changes into the main codebase. This review is made by the maintainers of the repository, mainly, Senior Software Engineers.

Once the PR was approved by a maintainer of the project, the tool was available in staging environment, where the next testing step take place.

4.3.3 Testing

For testing, the Kiwi Test Case Management System (TCMS) software was used. There a test case was design to ensure the proper function of the tool periodically and check that

Figure 4: Git workflow schema



the new features added to Pathfinder do not affect this part of the code.

The test case consists in reproduce some steps and check that the results are what is expected and no error ocurred during the evaluation. The checks proposed are:

- The geoprocess shows the proper error message for the following situations:

  The provided number of diameters, design pressures and prices do not match

  The starting pressure is not greater than the minimum pressure

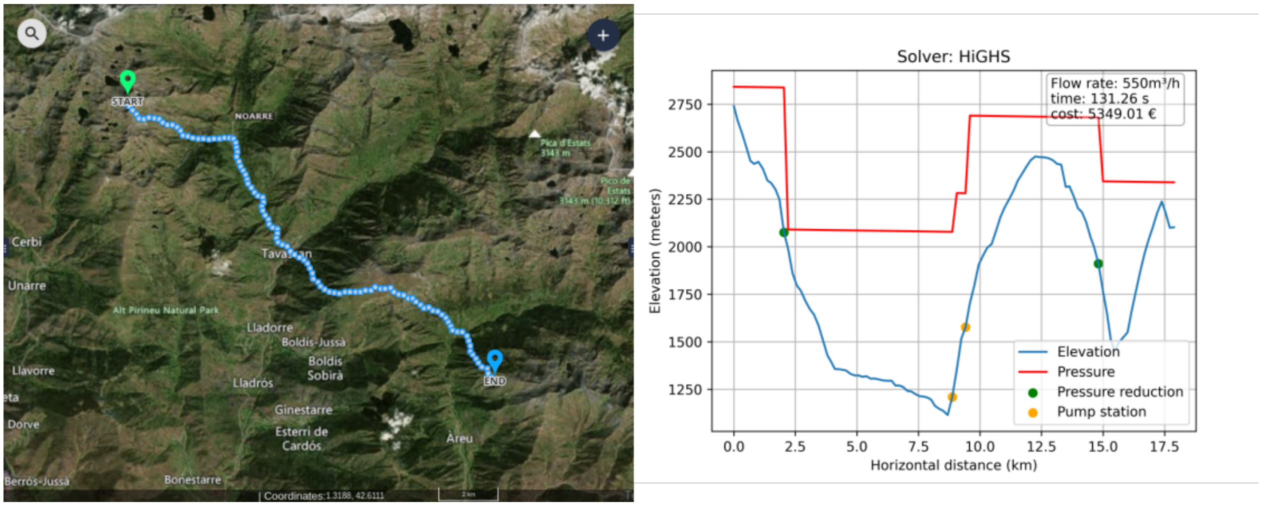
  The maximum pressure at the end of the pipeline is not greater than the minimum pressure

  The minimum pressure provided is greater than the design pressures provided

- The time limit works

- The results are properly calculated

Figure 5: Integrated results: Path and Plot



As mentioned above, Kiwi TCMS was used to manage the manual tests. This software allows the creation of test cases. A test case is a small setup and a few steps devoted to evaluating if a component of a SW project works as expected. Several test cases can be added to a test plan. Later, derived from a test plan, a test run executes each of the test cases of the test plan. In this case, just one test case was created, and its result was successful.

5 Research results

The main outcome of this thesis is the geoprocess for Pipeline optimization developed. Thus, describing it and showing its functionalities will be the focus of this section.

5.1 Pipeline optimization geoprocess

The geoprocess is configured as follows:

Inputs:

- Optimal path: the base 3D path to use in the geoprocess
- Density: the density of the fluid to be transported in kg/m^3
- Viscosity: the kinematic viscosity of the fluid to be transported in mm^2/s
- Cost pump station: the construction cost of a pump station in currency units
- Cost pressure reduction station: the construction cost of a pressure reduction station in currency units
- Design factor: a safety factor to manage how near can be the difference between the pressure of each node minus the elevation of each node and the bearing capacity of the corresponding segment. From 0 to 1 (no separation at all)
- Minimum pressure: the minimum pressure along the pipeline in meters
- Initial pressure: optional parameter which if set, will make the initial node have a determined pressure in meters
- Maximum final pressure: optional parameter which if set, will make the pipeline pressure at the end of the pipeline meet the introduced pressure value in meters
- Flow rate: flow rate of the pipeline in m^3/h. In conjunction with the viscosity and the density plays a key role in the determination of the pressure drop and the allocation of the pump stations.
- Pipeline sizes: a list of the diameter of each available pipeline in mm
- Design pressures: a list of the design pressure of each available pipeline in meters
- Prices: a list of available pipelines prices in currency unit/1000 meters
- Pump power: a list of power capacity of each available pump in MPa
- Relative gap: the relative gap tolerance for the solver to stop. In percentage. (See Prototyping)
- Absolute gap: the absolute gap tolerance for the solver to stop. In currency units. (See Prototyping)
- Time limit: a time limiter to stop the solver. In seconds

Table 4: Pressure table sample

Horizontal distance (Km)	**Pressure (m)**
0	201.0
0.141	200.6
0.283	200.1
0.424	199.7
0.587	199.1
0.707	198.8
0.849	198.3
...	...

Constants:

- Gravitational acceleration: 9.80665 m/s^2
- M: 1000
- m: 0.123
- e: 100

Outputs:

- Pump stations: A geospatial format vector layer, containing Point features corresponding to the optimal location for the pump stations needed
- Pressure reduction stations: A geospatial format vector layer containing Point features corresponding to the optimal location of the pressure reduction stations needed
- Pressure table: A table containing the calculated pressure at each node of the pipeline. Tabular data. (See Table 4).
- Sizes table: Optimal pipeline size chosen for each node of the pipeline. Tabular data. (See Table 5).
- Pipeline cost: The resulting cost. A floating point number retrieved from applying the optimal solution to the optimizing function. (See Mathematical model)

The visual appearance of the tool once integrated in Pathfinder can be seen in Figure 6, where the input parameters have been grouped in thematic sections for a better organization and understanding of the tool.

The geoprocess lets the user to introduce the fluid type the pipeline is meant to transport. This is made by the configuration of the viscosity and density of the fluid. Apart from that, a key factor of the fluid's transportation is the estimated flow rate, which can also be added in the geoprocess.

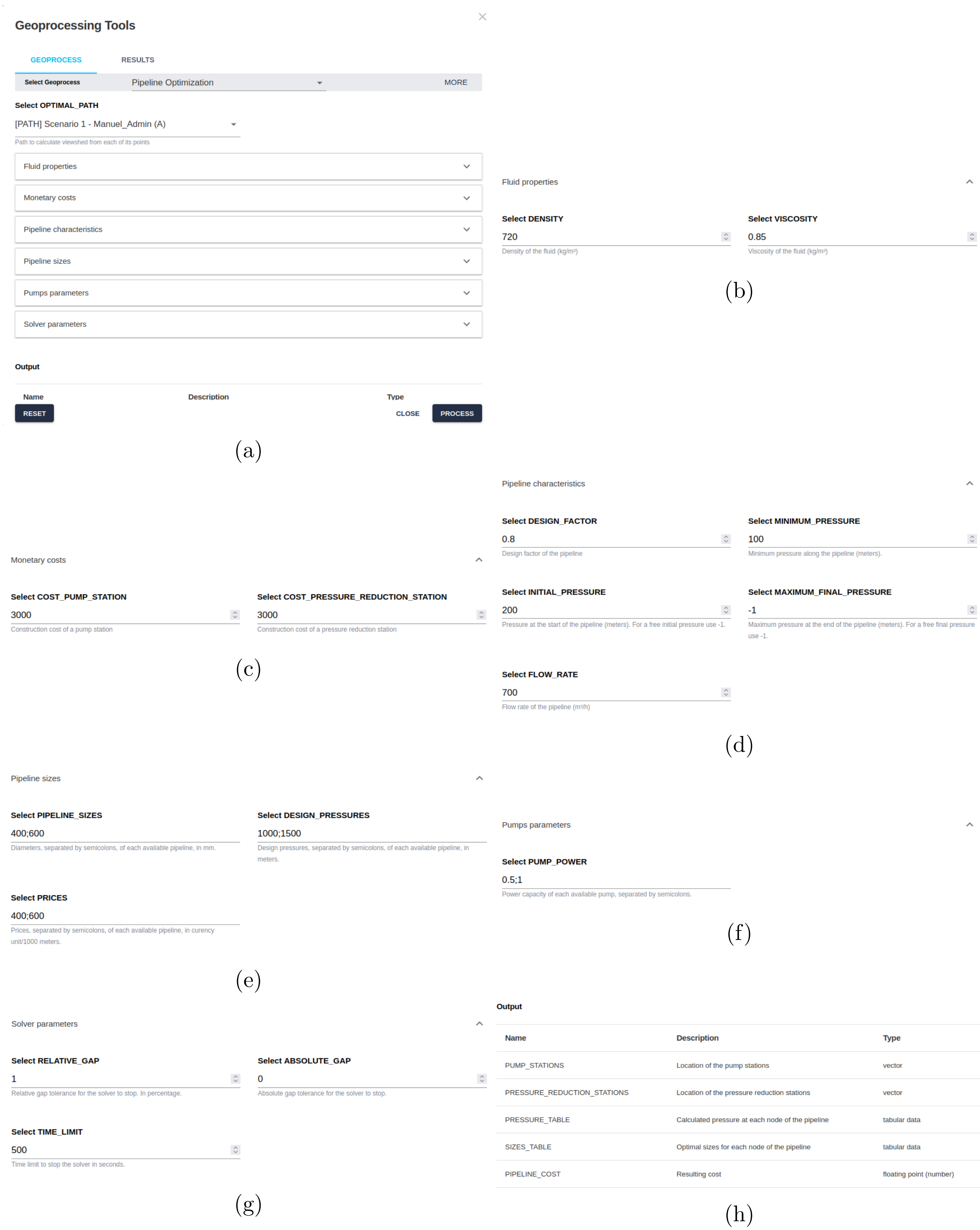


Figure 6: Geoprocess visual aspect in Pathfinder

Table 5: Sizes table sample

Diameter (mm)	Design pressure (m)	Cost
400	600	400
400	600	400
400	600	400
600	800	600
600	800	600
600	800	600
400	600	400
...	...	...

Once that is determined, the user can introduce the technology of his availability. This refers to the different pipeline segments that will be used for constructing the pipeline: its diameter, bearable pressure, and price. In addition to that, in order to move the fluid, pump stations are needed. The characteristic information of these pump stations can be modeled using the following two parameters: the power of each available pump and its construction cost. In order to ensure safe fluid transportation conditions, pressure reduction stations are needed as well. These stations are valves, so their construction cost can be added to the model. The model is in charge of locating these two components to comply with each pipeline segment's pressure design. However, this limit can be reduced using the design factor for more secure conditions.

The fluid starts from a factory with a specific pressure and ends in another whose pressure may vary from the initial one. Thus, adding a starting and maximum ending pressure allows the model to be aware of the actual conditions of the problem to be solved.

Finally, the solver parameters give the user more governance over the execution, limiting the computation time consumed or the solution's optimality in relative or absolute terms.

For a more visual way of understanding the main result of this thesis, a video demonstration using the geoprocess in the Pathfinder project have been made. This project already contains all the layers commented in section Data, but will be explained in a more detailed way in Case Study section. In order to view it, please, scan the following QR code:

Finally, its codebase is available in this document's Code section for a more detailed analysis of the SW tool.

Figure 7: QR Code to demo

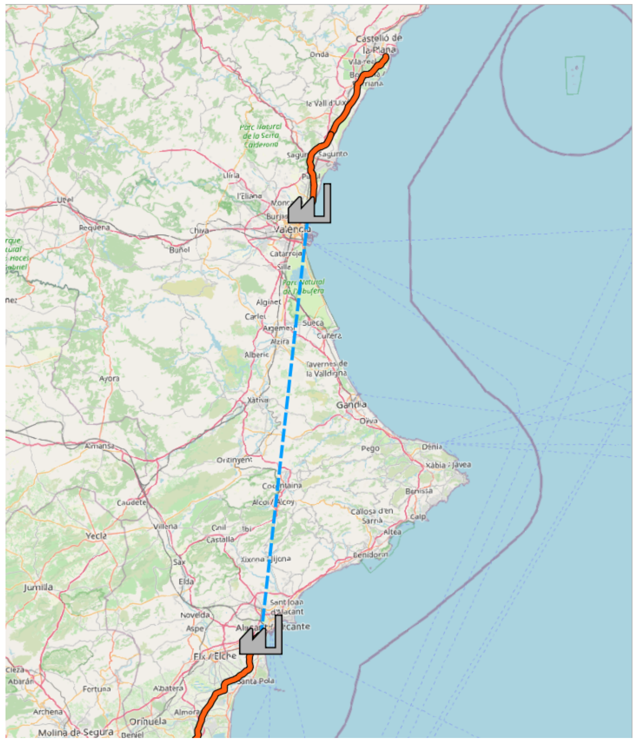
6 Case Study

With the tool integrated, it is the moment to apply it to a case study. Although the scenario recreated here is not based on a real need, the intention was to make it as plausible as possible. Thus, the start and end points needed to calculate an optimal path for a pipeline correspond to two real oil stations belonging to the CLH Group, the leading storage and transportation company for petroleum products in Spain, as well as one of the largest private companies in its sector at international level. The starting point corresponds to a station located a few kilometers northeast of Valencia city (39° 32' 13" N, 0° 18' 46" E). The ending point corresponds to another station, in this case, located 4 kilometers to the southwest of Alicante city (38° 19' 10" N, 0° 31' 56" E). For a more referenced location of the two stations to be connected, see section Cartography and Figure 8. Apart from the points, in order to start working on Pathfinder, a working area is needed, too. That working area corresponds to the extent Pathfinder will use for searching the most optimal path. This area is defined as a polygon, either drawn directly inside Pathfinder or uploaded as a geospatial format file. In this case, it is going to be drawn in Pathfinder itself.

All Pathfinder projects have a resolution parameter, configuring all other spatial information to use that resolution. Because of that, it is important to have an as high as possible resolution. This parameter works in conjunction with the working area, as the combination of both is used to estimate the "weight" of the project. Because of that, it is worth it to generate a polygon that will cover the most probable areas in which the corridor will pass while keeping its area as small as possible to set a small resolution in the project. In this case, the resolution was set at 10 meters with the area drawn.

With all the above, the Pathfinder project is created. The next step will consist of loading all the geographic information layers needed and assigning a resistance for each one, which will be explained in the following section.

Figure 8: Hypothetical connection location. Basemap: OSM



6.1 Layers weighting

According to what was discussed in Data and Criteria section, the main key routing factors are:

- Maximize the use of existing rights of way
- Maximize bare land
- Maintain the route within low-cost terrain and stable soil areas
- Guarantee good access for maintenance, emergency and protection
- Minimize rights of way crossing
- Minimize water body crossing
- Minimize forest areas crossing
- Minimize agricultural lands crossing
- Avoid water accumulation areas, ridges and streams
- Avoid steer and large-slope terrain
- Avoid hard rocks and clay soils
- Avoid built up areas

(a) Aerial view of the starting point (b) Aerial view of the ending point

Figure 9: Aerial view of the stations to be connected

Thus, the layers loaded in the project are weighted, setting a layer resistance interval from -3 (Min) to 3 (Max), as shown in tables 6, 7, 8, 9, 10 and 11.

Table 6: Environment layers weight

Layer	**Weight**
Forest	1
Humid areas	1
Landscape interest	1
Natural park	1
Natural parks and micro-reserves	FB
Protected sites	FB
Protected landscapes	FB

Table 7: Hydrology layers weight

Layer	**Weight**
Lakes	FB
Rivers	1
Sea	FB

Table 8: Infrastructures layers weight

Layer	Weight	
	Ring 1	Ring 2
Airports	FB	
Helipads, Heliports	FB	
Oil Gas Pipelines	-1	
Power lines (<100kV-150kV)	-1	
Railway	FB	-1
Railway High Speed	FB	-1
Roads	FB	-1

Table 9: Settlements layers weight

Layer	Weight
Archaeological sites	3
High cultural value	3
Urban areas	3

Table 10: Terrain layers weight

Layer	Weight
Agricultural land use	2
Clay	2
Flat terrain	1
Ridges	2
Stream channels	2
Urban areas	3
Urban areas	3
Urban areas	3
DEM	0
Slope	RN

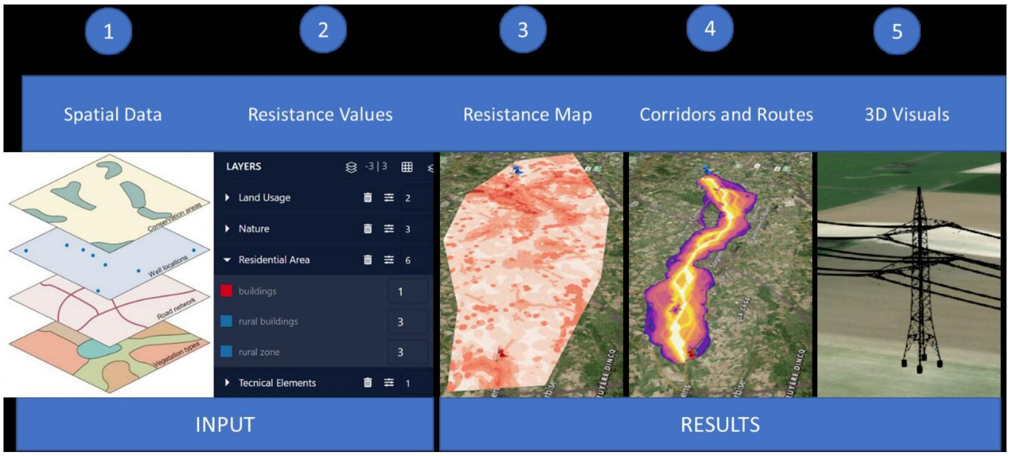
Table 11: Categories layers weight

Category	Weight
Environment	1
Hydrology	3
Infrastructures	1
Settlements	1
Terrain	1

6.2 Scenario configuration

Pathfinder constitutes a powerful tool for optimal routing and sitting. Its capabilities are vast and many parameters can be configured. This section will give a brief explanation of the main key points of the configuration followed in the project.

Figure 10: Pathfinder methodology



6.2.1 MCDA

Pathfinder runs a multi-criteria decision analysis (MCDA) to calculate the Resistance Map for the entire planning area. The Multi Criteria Decision Analysis (MCDA) performed by Pathfinder calculates a final resistance value for each location in the project area. To do so, it combines the layer resistances and category weights according to a formula which can be customized for specific clients or regional standards. In this case, the default MCDA operates by adding all category contributions to calculate the final value.

6.2.2 Algorithm

Among the algorithms available, the Pathfinder Explorer Algorithm is chosen. This algorithm is a multipath algorithm which can be used like the standard routing algorithms, but with some advantages as great performance with large areas, generation of many alternative routes as well as capable of being expanded to incorporate intricate geometric and multi-map limitations. In this case, 4 routes are going to be calculated.

With the scenario configured, the next step is to generate the resistance map, the corridor map and the paths.

The Resistance Map consists of a raster map of 10 meters resolution whose cells store the sum of the resistance of the layers in that specific pixel. In the case of a forbidden area, the pixel will have no value and be visualized in transparency regardless of the value of other layers in those locations. It occupies the full extension of the working area and represents the base for further calculations.

Figure 11: Resulting RM, CM and Paths

The Corridor Map displays the regions through which an ideal path is most probable to traverse, offering a swift summary of available routing choices. In this case, as the algorithm elected is the Explore routing algorithm, the corridor width is used to control the spatial influence of the routers combined to make the corridor.

The paths are the fourth most optimal routes, in terms of cost, based on the Resistance Map information from the starting to the end point.

All the aforementioned maps and paths can be seen inside Pathfinder in Figure 11.

6.3 Geoprocess use

With the paths calculated, it is the moment to apply the geoprocess developed to the most optimal one.

6.3.1 Configuration

As explained in Research results section, the tool developed has many parameters to be configured. In this example, as the case study is focused on oil transportation, the density and kinematic viscosity will be set to 720.3 kg/m3 and 0.85 mm2 /s, respectively. Following, a pump station construction cost of 3,000 € will be estimated. The exact amount is set for the pressure reduction stations.

For the pipeline characteristics, it is going to be supposed that the company has one type of pipeline whose diameter, pressure design, and price are 400 mm, 600 meters, and 400€, respectively. On the other hand, the pump power available is 0.5, 1, and 2 MPa.

Moreover, a minimum pressure of 100 meters is set. An initial pressure for the starting station of 200 meters and maximum final pressure at the destination station of 400 meters are supposed. The flow rate estimated is 700 m^3/h, and for more secure flow transportation conditions, a design factor of 0.8 is set.

Finally, in order to obtain a solution in a reasonable amount of time, the solver timer is set to 500 seconds and a relative gap of 10%.

6.3.2 Results

After the calculation, the geoprocess shows its results to the user, as it is possible to see in figure 12. From that view, it is possible to either show the results in the map inside Pathfinder or export them to work with them in a GIS desktop application such as QGIS or ArcGIS.

Figure 12: Geoprocessing results

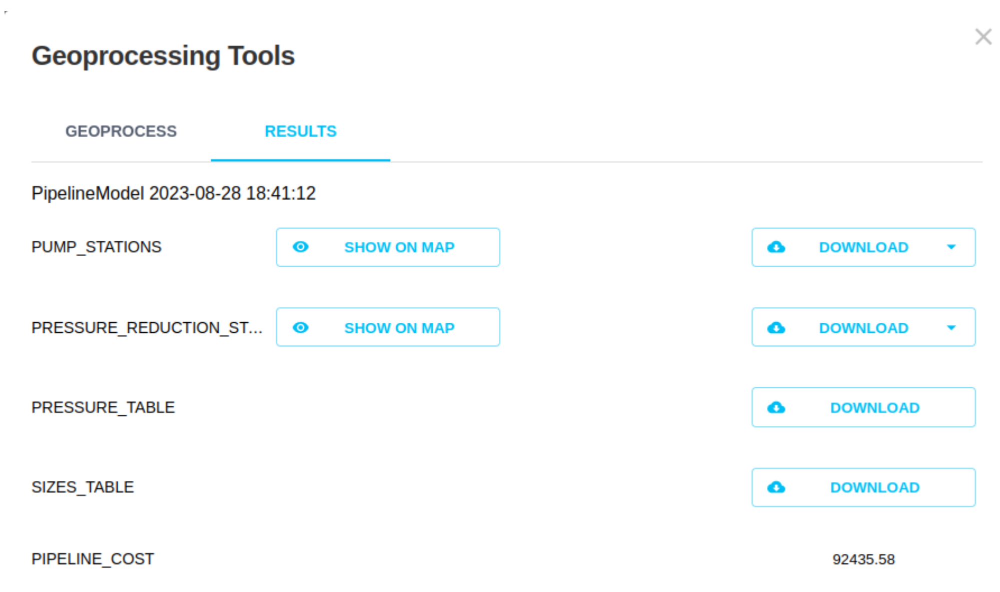


Table 12: Pump stations coordinates in WGS84

Longitude (Degrees)	**Latitude (Degrees)**
-0.421842	39.437160
-0.471365	39.368792
-0.602971	39.158249
-0.673553	39.070579
-0.730048	38.879703
-0.726604	38.853959
-0.681736	38.699012
-0.664522	38.680380

Table 13: Pressure reduction stations coordinates in WGS84

Longitude (Degrees)	**Latitude (Degrees)**
-0.642295	38.636146
-0.588585	38.456117

To summarize, the optimal pipeline calculated for this case study is 156 Km long and needs eight pump stations (see Figure 12) and two pressure reduction stations (see Figure 13). The cost estimated is 92,435 €.

In order to see the distribution of the pump and pressure reduction stations along the pipeline, see section Cartography.

7 Budget

In this section, an estimation of costs and budget emulated under real commercial situations has been carried out. The budget of this project is simple as there are virtually no costs for its development apart from the dedicated human resources. Note that the student has been working at Gilytics AG during the development of the tool and that it was Gilytics itself the promoter of adding this new tool to their SW.

However, for the sake of this memorandum, it is going to be supposed that a customer had requested it. Under this circumstance, the client has to purchase Pathfinder and pay for one of its licenses, and the development cost will also be charged to him.

Firstly, the human resources cost will be estimated. In this case, the project has been developed by the student. However, following the emulated situation, a worker who has already fulfilled the studies this thesis is submitted for will take the student's place. In that situation, the people implied in the thesis development are an Engineer with the role of developer, two Senior profile workers, one with the role of Manager or Team Lead, and the other with the role of Senior Back-End Engineer.

To be as accurate as possible in this task, the National Collective Bargaining Agreement for Engineering; technical survey offices; inspection, supervision, and technical and quality control BOE (2023), was consulted. From that document, it has been retrieved the base salary for Engineers with Master's Degree studies, such as the case of the student. All the itemization is presented 14. Note that the Contract Rewards have been estimated and do not correspond to any official source nor personal experience from the student.

As stated before, in addition to the human resources cost, the customer requesting this tool has to purchase a Pathfinder license. There are different options, but a Pathfinder Basic license has been chosen for the present case. This license includes access to Pathfinder geoprocesses and up to one administrator user account. The price for this kind of license is 25,000€.

With all the above, it is possible to calculate the total cost of the project by adding these two costs up and applying an increment in the concept of industrial benefit. To finish, the total cost is calculated and shown in table 15.

Table 14: Human Resources Cost

	Manager	**Senior Engineer**	**Junior Developer**
Salary			
Base Salary	27,113.82 €	27,114.82 €	27,113.82 €
Contract Rewards*	15,000 €	10,000 €	0 €
Annual salary	42,113.82 €	37,114.82 €	27,113.82 €
Salary/Hour			
Annual work hours	1,800	1,800	1,800
Salary/Labor hour	23 €/h	21 €/h	15 €/h
Project Dedication			
Required months	4		
Required hours	960		
Work Assignment			
Assigned work	5.00%	5.00%	70.00%
Hours dedication	48	48	672
Work Cost			
Num. Workers	1	1	1
Total cost per Team Player	1,123 €	990 €	10,122 €
Total human resources cost	**12,235 €**		

Table 15: Cost Production

Total human resources cost	12,235 €
Pathfinder license	25,000 €
Total cost production	37,235 €
Total cost production + Industrial Benefit (15%)	**42,821 €**

8 Conclusions

This study presents a comprehensive approach to pipeline routing and optimization, using standard Pathfinder features for the routing part and a custom geoprocessing tool for the optimization part. This methodology successfully integrated various spatial layers, multi-criteria decision analysis (MCDA), and advanced algorithms to determine optimal paths for pipeline installation. The results from the case study demonstrated how the combination of Pathfinder and the developed geoprocess provides feasible and efficient pipeline routes while considering multiple factors such as environmental, infrastructural, and terrain considerations.

However, there are several areas where enhancements could be made to further improve the software performance and utility, as well as new areas to further develop them.

8.1 Enhancements

While HiGHS is a solid solver, it is worth noting that commercial solver options often offer more robust optimization capabilities. Exploring integration with well-established commercial solvers could potentially improve the quality and efficiency of the optimization process. This will be the case when the demand for this tool by Gilytics' customers increases.

Additionally, the representation of the pipeline in the three-dimensional view can be improved by adding a 3D pipeline model. Pathfinder only uses 3D pylon models to represent the pipeline in the 3D view. Instead, some pipeline segment types could be modeled in 3D and added to the 3D map for a more realistic pipeline representation.

8.2 Future work

Expanding the tool's capabilities to handle network optimization scenarios, not limited to point-to-point paths, would be a valuable extension. This could accommodate complex pipeline networks with multiple source and destination nodes.

In conclusion, this study demonstrated the potential of the developed geoprocessing tool for pipeline optimization and routing. By addressing the mentioned enhancements and exploring the suggested future work areas, the tool can evolve into a valuable resource for the energy industry, contributing to the efficient, sustainable, and resilient transportation of resources.

Nomenclature

Decision varibles

$C_{\mathrm{F}j}$	Pressure drop of the pressure reduction station (MPa)
p_i	Pressure of node i (MPa)
$T_{\mathrm{D}j}$	0-1 variable, equal to 1 if a pressure reduction station is built in arc j, and equal to 0 otherwise
$T_{\mathrm{P}_{j,g}}$	0-1 variable, equal to 1 if a pipeline sized g is built in arc j, and equal to 0 otherwise
$T_{\mathrm{Z}_{j,y}}$	0-1 variable, equal to 1 if a pump station is built under y pump operational plan

Indices and sets

G	Set of pipeline sizes, denoted by index g
L_a	Set of all the nodes
N_a	Set of all the nodes
Y	Set of pump operational plans, denoted by index y

Parameters

ν	Kinematic viscosity of the fluid (Pa $\cdot$ s)
ϕ_j	Distance of pipeline arc j (km)
ε	Factor of the pressure-flow equation
C_{D_g}	Flow coefficient of pipeline sized g in the mass balance equations
C_{UD}	Construction cost of pressure reduction station
$C_{\mathrm{UP}g}$	Construction cost of pipeline sized g per unit length
C_{UZ}	Construction cost of pump station
m	A parameter related to the flow state
P_e	Maximum pressure at end of the pipeline (MPa)
P_s	Initial pressure of the pipeline (MPa)
$P_{\mathrm{G}g}$	Design pressure for pipeline sized g (MPa)
Q	Flow rate of the pipeline (m^3/h)
z_i	Elevation of node i (MPa)
M	A sufficiently large number
W	Deisgn factor of the pipeline

References

Abudu, D. and Williams, M. (2015). Gis-based optimal route selection for oil and gas pipelines in uganda.

Akshay, G., Shabbir, A., Seok, C. M., and Santanu, D. (2013). Solving mixed integer bilinear problems using milp formulations. *SIAM Journal on Optimization*, 23(2):721–744.

Almedallah, M. K., Branch, G., and Walsh, S. D. (2020). Combined well path, submarine pipeline network, route and flow rate optimization for shallow-water offshore fields.

BOE (2023). Xx convenio colectivo nacional de empresas de ingeniería; oficinas de estudios técnicos; inspección, supervisión control técnico y de calidad.

CNIG (2023). Centro de descargas del centro nacional de información geográfica.

COIN-OR Foundation (2016). Coin-or web site.

contributors, O. S. M. (2023). Open stree map.

Cruz-Chávez, M. A., Moreno-Bernal, P., Rivera-López, R., Ávila-Melgar, E. Y., Martínez-Bahena, B., and Cruz-Rosales, M. H. (2020). Gis spatial optimization for corridor alignment using simulated annealing.

Durmaz, A. I., Ünal, E. Ö., and Aydin, C. C. (2019). Automatic pipeline route design with multi-criteria evaluation based on least-cost path analysis and line-based cartographic simplification: A case study of the mus project in turkey.

European Commission (2022). Energy and the green deal.

Gilytics (2023). Pathfinder user manual.

Guardia, M. L., D'Ippolito, F., and Cellura, M. (2022). A gis-based optimization model finalized to the localization of new power-to-gas plants: The case study of sicily (italy).

Gurobi Optimization, I. (2023). Gurobi optimizer reference manual.

Gyabeng, B. A. (2020). Selection of optimum petroleum pipeline routes using a multi-criteria decision analysis and gis least-cost path approach.

Hall, J., editor (2022). *The 9th annual JuliaCon.*

Hamid-Mosaku, I. A., Oguntade, O. F., Ifeanyi, V. I., Balogun, A.-L., and Jimoh, O. A. (2020). Evolving a comprehensive geomatics multi-criteria evaluation index model for optimal pipeline route selection.

Huangfu, Q. and Hall, J. A. J. (2017). Parallelizing the dual revised simplex method.

IGN (2023a). *Especificaciones técnicas Base Topográfica Nacional.*

IGN (2023b). Instituto geográfico nacional.

ISO (2020). Iso 3166-2:2020.

Khani, A. and Amini, A. S. (2022). Optimum route election for oil and gas transmission lines using remote sensing and gis integration.

Li, Z. (2020). *Pipeline Spatial Data Modeling and Pipeline WebGIS*.

Macharia, P. M. and Mundia, C. N. (2014). Gis analysis and spatial modelling for optimal oil pipeline route location. a case study of proposed isiolo nakuru pipeline route.

Medvedeva, O. N. and Penenko, V. D. (2021). Routes of laying gas supply system pipeline.

Mittelmann, H. D. (2023). Decision tree for optimization software.

Moreno-Bernal, P. and Nesmachnow, S. (2020). A* algorithm for gis-based pipeline route selection in veracruz, mexico.

Oltermann, P. (2022). How reliant is germany – and the rest of europe – on russian gas?

Parzen, M., Hall, J., Jenkins, J., and Brown, T. (2022). Optimization solvers: the missing link for a fully open- source energy system modelling ecosystem.

PyPSA-Eur (2021). Open optimisation model of the european transmission system.

Rardin, R. L. (1998). *Optimization in operations research*, volume 166. Prentice Hall Upper Saddle River, NJ.

Royce, W. W. (1970). Managing the development of large software systems.

Seli, V. A., Zardasti, L. B., and Noor, N. B. M. (2022). Risk assessment of underground gas pipeline leakage incorporating geographical information system (gis).

Sivakumar, V. L., Nallanathel, M., M., R., and Veeraswamy, G. (2020). Optimal route selection for the transmission of natural gos through pipelines in tiruchengode taluk using gis - a preliminary study.

Wang, B., Liang, Y., Zheng, T., Yuan, M., and Zhang, H. (2019a). Optimisation of a downstream oil supply chain with new pipeline route planning.

Wang, B., Yuan, M., Yan, Y., Yang, K., Zhang, H., and Liang, Y. (2019b). Optimal design of an oil pipeline with a large-slope section.

Wang, B., Zhang, H., Yuan, M., Wang, Y., Menezes, B. C., Li, Z., and Liang, Y. (2019c). Sustainable crude oil transportation: design optimization for pipelines considering thermal and hydraulic energy consumption.

Warmerdam, F., Rouault, E., et al. (2023). Gdal documentation.

Wen, K., Lu, Y., Lu, M., Zhang, W., Zhu, M., Qiao, D., Meng, F., Zhang, J., Gong, J., and Hong, B. (2022). Multi-period optimal infrastructure planning of natural gas pipeline network system integrating flowrate allocation.

woo Kim, S., Piao, Z., and Lee, D.-K. (2022). Optimal road design using genetic algorithm to improve biodiversity and risk of soil loss. *Geocarto International*, 37(27):14811–14827.

A Appendices

A.1 Code

```python
# This geoprocess is an implementation of the MIP model shown in the scientific publication
# "Optimal design of an oil pipeline with a large-slope section" by Bohong Wang et al.
# In the present document, most of the variables are named acording
# to the nomenclature of the aforementioned paper,
# which meaning is available at its "Nomenclature" section:
# https://gilydrive.awsapps.com/workdocs/index.html#/document/1d80ccca9426adccac3ca0b8c4354199fc62e4492b2915487a98c0d6832866db

import sys

from django.contrib.gis.geos import MultiPoint, Point
from django.utils.translation import gettext_lazy as _

from pulp import (
    HiGHS_CMD, LpBinary, LpContinuous, LpMinimize,
    LpProblem, LpStatus, LpVariable, lpSum, re, value
)

from gilytics.utils import get_metric_srid_from_wgs84_point
from gilytics.utils_parameters import Parameter, VectorParameter
from modules.geoprocessing.base import GeoProcess, geoprocessDocURL
from modules.geoprocessing.exceptions import GeoProcessingInvalidParameterValue
from modules.mcda.api_v1.utils import SRID_WGS84


class PipelineModel(GeoProcess):

    _URLDOC = geoprocessDocURL() + "#pipeline-optimization"
    __doc__ = _(f"""
        Calculates the construction cost of a pipeline and the optimal location for
        its pump and pressure reduction stations given an optimal path and the
        pipeline characteristics.

        Generates two vector files, a table containing the optimal pipeline for each node
        and the total construction cost.

        <a href="{_URLDOC}" target="_blank">More information in Pathfinder's documentation.</a>
    """)
    title = _("Pipeline Optimization")

    def __init__(self, *args, **kwargs):
        super().__init__(*args, **kwargs)
```

```

        self.add_vector_parameter(
            "OPTIMAL_PATH",
            _("Path to calculate viewshed from each of its points"),
            Parameter.PARAM_INPUT,
            ds_type=VectorParameter.SOURCE_SCENARIO_PATH_3D,
            required=True
        )

        self.add_float_parameter(
            "COST_PUMP_STATION",
            _("Construction cost of a pump station"),
            Parameter.PARAM_INPUT,
            required=True,
            category=_("Monetary costs")
        )

        self.add_float_parameter(
            "COST_PRESSURE_REDUCTION_STATION",
            _("Construction cost of a pressure reduction station"),
            Parameter.PARAM_INPUT,
            required=True,
            category=_("Monetary costs")
        )

        self.add_float_parameter(
            "DENSITY",
            _("Density of the fluid (kg/m^3)"),
            Parameter.PARAM_INPUT,
            required=True,
            default=720,
            category=_("Fluid properties")
        )

        self.add_float_parameter(
            "VISCOSITY",
            _("Viscosity of the fluid (kg/m^3)"),
            Parameter.PARAM_INPUT,
            required=True,
            default=0.85,
            category=_("Fluid properties")
        )

        self.add_float_parameter(
            "DESIGN_FACTOR",
            _("Design factor of the pipeline"),
            Parameter.PARAM_INPUT,
            required=True,
            default=0.8,
            category=_("Pipeline characteristics")
        )

        self.add_float_parameter(
            "MINIMUM_PRESSURE",
```

```
            _("Minimum pressure along the pipeline (meters)."),
            Parameter.PARAM_INPUT,
            required=True,
            category=_("Pipeline characteristics"),
            default=100
        )

        self.add_float_parameter(
            "INITIAL_PRESSURE",
            _("""Pressure at the start of the pipeline (meters).
            For a free initial pressure use -1."""),
            Parameter.PARAM_INPUT,
            required=False,
            default=-1,
            category=_("Pipeline characteristics")
        )

        self.add_float_parameter(
            "MAXIMUM_FINAL_PRESSURE",
            _("""Maximum pressure at the end of the pipeline (meters).
            For a free final pressure use -1."""),
            Parameter.PARAM_INPUT,
            required=False,
            default=-1,
            category=_("Pipeline characteristics")
        )

        self.add_integer_parameter(
            "FLOW_RATE",
            _("Flow rate of the pipeline (m^3/h)"),
            Parameter.PARAM_INPUT,
            required=True,
            default=700,
            category=_("Pipeline characteristics")
        )

        self.add_string_parameter(
            "PIPELINE_SIZES",
            _("Diameters, separated by semicolons, of each available pipeline, in mm."),
            Parameter.PARAM_INPUT,
            required=True,
            default="400",
            category=_("Pipeline sizes")
        )

        self.add_string_parameter(
            "DESIGN_PRESSURES",
            _("Design pressures, separated by semicolons, of each available pipeline, in meters."),
            Parameter.PARAM_INPUT,
            required=True,
            category=_("Pipeline sizes"),
            default="1000"
```

```
        )

        self.add_string_parameter(
            "PRICES",
            _("""Prices, separated by semicolons, of each available pipeline,
            in curency unit/1000 meters."""),
            Parameter.PARAM_INPUT,
            required=True,
            category=_("Pipeline sizes"),
            default="400"
        )

        self.add_string_parameter(
            "PUMP_POWER",
            _("Power capacity of each available pump, separated by semicolons."),
            Parameter.PARAM_INPUT,
            required=True,
            category=_("Pumps parameters"),
            default="1"
        )

        self.add_integer_parameter(
            "RELATIVE_GAP",
            _("Relative gap tolerance for the solver to stop. In percentage."),
            Parameter.PARAM_INPUT,
            required=True,
            category=_("Solver parameters"),
            default=0
        )

        self.add_float_parameter(
            "ABSOLUTE_GAP",
            _("Absolute gap tolerance for the solver to stop."),
            Parameter.PARAM_INPUT,
            required=True,
            category=_("Solver parameters"),
            default=0
        )

        self.add_integer_parameter(
            "TIME_LIMIT",
            _("Time limit to stop the solver in seconds."),
            Parameter.PARAM_INPUT,
            required=True,
            category=_("Solver parameters"),
            default=500
        )

        self.add_vector_parameter(
            "PUMP_STATIONS",
            _("Location of the pump stations"),
```

```
            Parameter.PARAM_OUTPUT,
            ds_type=VectorParameter.SOURCE_GEOSGEOM
        )

        self.add_vector_parameter(
            "PRESSURE_REDUCTION_STATIONS",
            _("Location of the pressure reduction stations"),
            Parameter.PARAM_OUTPUT,
            ds_type=VectorParameter.SOURCE_GEOSGEOM
        )

        self.add_tabulardata_parameter(
            "PRESSURE_TABLE",
            _("Calculated pressure at each node of the pipeline"),
            Parameter.PARAM_OUTPUT
        )

        self.add_tabulardata_parameter(
            "SIZES_TABLE",
            _("Optimal sizes for each node of the pipeline"),
            Parameter.PARAM_OUTPUT
        )

        self.add_float_parameter(
            "PIPELINE_COST",
            _("Resulting cost"),
            Parameter.PARAM_OUTPUT
        )

    def RunGeoprocess(self):

        # Model constants
        ga = 9.80665
        M = 1000
        m = 0.123
        e = 100

        # Retrieving 3D path and tranforming to metric reference system
        opath = self.get_parameter("OPTIMAL_PATH").data()
        metric_srs = get_metric_srid_from_wgs84_point(opath.centroid)
        opath.transform(metric_srs)

        # Retrieve model parameters
        W = self.get_parameter("DESIGN_FACTOR").data()

        # Costs
        Cuz = self.get_parameter("COST_PUMP_STATION").data()
        Cud = self.get_parameter("COST_PRESSURE_REDUCTION_STATION").data()

        # Flow-pressure equation
        density = self.get_parameter("DENSITY").data()
        viscosity = self.get_parameter("VISCOSITY").data()
        flow_rate = self.get_parameter("FLOW_RATE").data()

```

```
        # Pressure options
        Pd_min = _meters_to_mpa(self.get_parameter("MINIMUM_PRESSURE").data())

        Ps = self.get_parameter("INITIAL_PRESSURE").data()
        if (Ps != -1):
            Ps = _meters_to_mpa(Ps)

        Pe = self.get_parameter("MAXIMUM_FINAL_PRESSURE").data()
        if (Pe != -1):
            Pe = _meters_to_mpa(Pe)

        # Retrieve the pipeline parameters
        diameter_list = [
            float(diameter)
            for diameter in self.get_parameter("PIPELINE_SIZES").data().split(';')]

        design_pressure_list = [
            _meters_to_mpa(float(pressure))
            for pressure in self.get_parameter("DESIGN_PRESSURES").data().split(';')]

        prices_list = [
            float(price) for price in self.get_parameter("PRICES").data().split(';')]

        power_list = [float(power) for power in self.get_parameter("PUMP_POWER").data().split(';')]

        # Initial checks

        if (not (len(diameter_list) == len(design_pressure_list) == len(prices_list))):
            raise GeoProcessingInvalidParameterValue(
                _("""The provided number of diameters, design pressures
            and prices do not match"""))

        if (Ps < Pd_min and Ps != -1):
            raise GeoProcessingInvalidParameterValue(
                _("""The problem is infeasible: The starting pressure has to be greater
                than the minimum pressure."""))

        if (Pe < Pd_min and Pe != -1):
            raise GeoProcessingInvalidParameterValue(
                _("""The problem is infeasible: The maximum pressure at the end of the
                pipeline has to be greater than the minimum pressure."""))

        if (Pd_min > max(design_pressure_list)):
            raise GeoProcessingInvalidParameterValue(
                _("""The problem is infeasible: The minimum pressure provided is greater than
```

```
        the design pressures provided. Try to decrease it or increase the
        design factor of the pipeline."""))

    # Generate data model

    Y = {0: {'a': 0, 'b': 0}}

    for i, power in enumerate(power_list):
        Y[i] = {'a': power, 'b': 0}

    G = {
        i: {"diameter": d, "design_pressure": p, "cost": c}
        for i, (d, p, c) in enumerate(zip(diameter_list, design_pressure_list, prices_list))
    }

    A = [flow_rate]

    # Generate the nodes set
    Na = {0: {'HD': 0, 'height': Point(opath[0]).z}}
    path_enumerator = enumerate(opath)
    next(path_enumerator)

    for i, pnt in path_enumerator:
        Na[i] = {
            'HD': (Point(pnt).distance(Point(opath[i-1]))) / 1000 + Na[i-1]['HD'],
            'height': Point(pnt).z
        }

    # Generate the arcs set
    La = {i: Point(opath[i]).distance(Point(opath[i+1])) / 1000
          for i in range(opath.num_points-1)}

    # Declare the problem varibale
    prob = LpProblem("Optimal_design_pipeline_cost", LpMinimize)

    # Model variables
    Tp = {}  # Size
    Td = {}  # Pressure reduction
    Tz = {}  # Pump station
    p = {}  # Pressure
    Cf = {}  # Pressure reduced

    for j in La:

        Tp[j] = {}
        Tz[j] = {}

        for g in G:

            Tp[j][g] = LpVariable(f'Tp_{j}_{g}', cat=LpBinary)

```

```python
        Td[j] = LpVariable(f'Td_{j}', cat=LpBinary)

        for y in Y:
            Tz[j][y] = {}
            for a in A:
                Tz[j][y][a] = LpVariable(f'Tz_{j}_{y}_{a}',  cat=LpBinary)

        Cf[j] = {}

        for a in A:
            Cf[j][a] = LpVariable(f'Cf_{j}_{a}', cat=LpContinuous, lowBound=0, upBound=M)

    for i, pnt in enumerate(opath):
        p[i] = {}
        for a in A:
            p[i][a] = LpVariable(f'p_{i}_{a}', cat=LpContinuous, lowBound=-M, upBound=M)

    # Set the problem function to be optimized
    prob += (
        # pipeline cost
        lpSum([La[j] * Tp[j][g] * G[g]['cost'] for j in La for g in G]) +
        # pressure reduction station cost
        lpSum([Td[j] * Cud for j in La]) +
        # pump station cost
        lpSum([Tz[j][y][a] * Cuz for j in La for y in Y for a in A])
    )

    # Add the constraints to the model

    # - Arc constraints
    for j in La:
        # One size of pipeline has to be chosen for every pipeline arc
        prob += lpSum(Tp[j][g] for g in G) == 1

        # If a pressure drop exists in arc j, then a pressure drop station must be built
        prob += lpSum(Cf[j][a] for a in A) <= M * Td[j]

    for i, pnt in enumerate(opath):
        for a in A:
            # The pressure of each node minus the elevation of each node cannot be lower than
            # the minimum pressure limit
            # Elevation comes in meters but has to be in MPa
            prob += p[i][a] - _meters_to_mpa(Point(pnt).z) >= Pd_min

        for j in La:
            if (i == j):

                for a in A:
```

```
                    prob += p[i][a] - _meters_to_mpa(Point(pnt).z) <= \
                        W * lpSum(Tp[j][g] * G[g]['design_pressure'] for g in G)

                    # Pressure reduction station
                    prob += p[i][a] - p[i+1][a] - Cf[j][a] >= M * (Td[j] - 1)
                    prob += p[i][a] - p[i+1][a] - Cf[j][a] <= M * (1 - Td[j])

                    Qa = a**1.877
                    for g in G:

                        Cd = e * (viscosity**m / G[g]['diameter']**(5-m)) * density * ga

                        # Pressure drop
                        prob += p[i][a] - p[i+1][a] - Cd*Qa*La[j] >= \
                            -M * ((1 - Tp[j][g]) + lpSum(Tz[j][y][a] for y in Y) + Td[j])

                        prob += p[i][a] - p[i+1][a] - Cd*Qa*La[j] <= \
                            M * ((1 - Tp[j][g]) + lpSum(Tz[j][y][a] for y in Y) + Td[j])

                    for y in Y:
                        # Pump station
                        prob += p[i][a] - p[i+1][a] + Y[y]['a'] - Y[y]['b'] * Qa >= \
                            M * (Tz[j][y][a] - 1)

                        prob += p[i][a] - p[i+1][a] + Y[y]['a'] - Y[y]['b']*Qa <= \
                            M * (1 - Tz[j][y][a])

    # Setting the pressure at the starting node (if added)
    if (Ps != -1):
        prob += p[0][A[0]] == Ps + _meters_to_mpa(Point(opath[0]).z)

    # Setting the maximum pressure at the end of the pipeline (if added)
    if (Pe != -1):
        prob += p[opath.num_points - 1][A[-1]] - _meters_to_mpa(Point(pnt).z) <= Pe

    # Solve the model

    # Retrieving solver parameters
    gap_rel = (self.get_parameter("RELATIVE_GAP").data())/100
    gap_abs = self.get_parameter("ABSOLUTE_GAP").data()
    time_limit = self.get_parameter("TIME_LIMIT").data()
```

```python
    try:
        old_stdout, old_stderr = _setup_sys_for_pulp_patch()
        status = prob.solve(HiGHS_CMD(gapRel=gap_rel, gapAbs=gap_abs, timeLimit=time_limit))
    finally:
        _recover_sys_for_pulp_patch(old_stdout, old_stderr)

    if (status == 0):
        raise GeoProcessingInvalidParameterValue(
            _(f'The problem is {LpStatus[status]}. More execution time is needed.'))

    elif (status == -1):
        raise GeoProcessingInvalidParameterValue(
            _(f'The problem is {LpStatus[status]}. Try to relax the parameters.'))

    # Process the results

    p_result = {}
    sizes = {}
    pump_nodes = []
    reduction_nodes = []

    for v in prob.variables():

        # Pressure of each node (from MPa to meters)
        if (re.match('p_', v.name)):
            p_result[Na[int(v.name.split('_')[1])]['HD']] = _mpa_to_meters(v.varValue)

        # Size chosen in each node
        if (re.match('Tp', v.name)):
            if (v.varValue == 1):
                sizes[int(v.name.split('_')[1])] = G[int(v.name.split('_')[2])]

        # Pump stations
        if (re.match('Tz', v.name)):
            if (v.varValue == 1):
                pump_nodes.append(int(v.name.split('_')[1]))

        # Pressure reduction stations
        if (re.match('Td', v.name)):
            if (v.varValue == 1):
                reduction_nodes.append(int(v.name.split('_')[1]))

    # Outputs

    # Estimated cost

    self.get_parameter("PIPELINE_COST").set_data(round(value(prob.objective), 2))

```

```
        # Sizes

        formatted_data = []

        formatted_data.append(["Diameter (mm)", "Design pressure (m)", "Cost"])

        for size in sorted(sizes.keys()):
            formatted_data.append([
                sizes[size]["diameter"],
                _mpa_to_meters(sizes[size]["design_pressure"]),
                sizes[size]["cost"]
            ])

        self.get_parameter("SIZES_TABLE").set_data(formatted_data)

        # Pressure

        formatted_data = []

        formatted_data.append(['Horizontal distance (Km)', 'Pressure (m)'])

        for distance in sorted(p_result.keys()):
            formatted_data.append([distance, p_result[distance]])

        self.get_parameter("PRESSURE_TABLE").set_data(formatted_data)

        # Layers

        # Pumps
        pumps = MultiPoint([Point(opath[i]) for i in pump_nodes], srid=metric_srs)
        pumps.transform(SRID_WGS84)
        self.get_parameter("PUMP_STATIONS").set_data(pumps)

        # Pressure reduction stations
        reductions = MultiPoint([Point(opath[i]) for i in reduction_nodes], srid=metric_srs)
        reductions.transform(SRID_WGS84)
        self.get_parameter("PRESSURE_REDUCTION_STATIONS").set_data(reductions)


def _meters_to_mpa(height):
    """
        This function transforms height in meters of head to pressure in MPa
    """
    return height * 0.009804139432


def _mpa_to_meters(pressure):
    """
```

```
        This functions transforms pressure in MPa to meters of head
    """
    return pressure * 101.99773339984


def _setup_sys_for_pulp_patch():
    """
        Function to avoid the error realted with the redirection of the
    output.
        PuLP expects sys.stdout to be an instance of a TextIOWrapper
    object,
        but Celery sets this variable to be a LoggingProxy.
        This generates an error when PuLP tries to call fileno(),
        a function that the object LoggingProxy do not have.
        If sys.stdout is None, PuLP just ignores the previous mentioned
    step.
    """

    old_stdout = sys.stdout
    old_stderr = sys.stderr

    sys.stdout = None
    sys.stderr = None

    return old_stdout, old_stderr


def _recover_sys_for_pulp_patch(stdout, stderr):
    """
        Function to recover the previous value of the system's output
    """

    sys.stdout = stdout
    sys.stderr = stderr
```

Listing 7: Geoprocess code

A.2 Cartography


General situation of the hypothetical connection proposed
Legend
Factory
Existing Oil Pipelines
Hyphotetical connection
20
40 km
València
Gandia


Map of the calculated pipeline and the position of the pump and pressure reduction stations
Legend
Existing Oil Pipelines
Hyphotetical connection
Calculated pipeline
Factory
Pump station
Pressure reduction station
20
40 km