POLITECNICO DI TORINO

Bachelor of Science in Building Engineering

Master degree's thesis

Reinforcement learning applied to InfraBIM

Supervisor:
Ing. Anna Osello

Co-Supervisor:
Arch. Arianna Fonsati

Candidate:
Ing. Nicola Rimella

Academic Year: 2019/2020

Abstract

In the last decades, artificial intelligence and machine learning have become very popular showing incredible success in applications ranging from speech recognition to computational biology. However, few attempts have been done in the building or infrastructure field, where the potential of such methodologies is still unexplored.

The introduction of BIM (Building Information Modelling) and Infra-BIM (in the case of infrastructural projects) for the design, analysis and maintenance of projects makes available a big amount of data, along with a collection of modelling strategies that can be embedded in machine learning algorithms to solve complex problems.

The construction of the tunnels by tunnel boring machine (TBM) starts from the design of a theoretical alignment, which is used by the TBM as guidance to calculate at each step a new as-built alignment through a laser-based machine that deviates as little as possible from the theoretical one. However, there is a lack of information during the design phase because the locations of the rings are unknown before starting the cunstruction phase.

This thesis aims to use machine learning algorithms, from the reinforcement learning branch, to calculate the position of the rings defining the tunnel lining and figuring out the possible as-built alignement during the design phase.

This can be done by extrapolating the coordinates and altimetries of the theoretical alignment and combining them with the data relative to the geometry of the rings. After which a series of simulations are made out simulations that allow the algorithm to understand, according to structural connections, what ring's rotations are better, in order to minimize the distance between the as-built allignment and the theoretical alignment.
Subsequently, through the use of modelling software, the tunnel lining is modelled in order to enable future analyses.

Contents

Chapter 1

Introduction

The aim of this thesis is to be able to create a TIM (Tunnel Information Model) of a Tunnel's lining made with an automatic method.
This model should deviate as little as possible from the theoretical alignment, elaborated through geotechnical and topographical analysis, to allow, during the design phase, analysis, and studies to improve the construction processes, such as, for example, the optimization of the minimum radius of curvature.

At present, the operational alignment remains unknown until the moment of construction using a TBM (Tunnel Boring Machine) which chooses the best position of the rings to deviate as little as possible from the theoretical alignment.

To fulfill this aim it was decided to adopt an innovative approach based on automatic learning algorithms.

In the first part of the thesis, the applications of machine learning and reinforcement learning in the building and infra-structural field are analyzed, to better understand the potential of these methodologies.
These applications of ML (Machine Learning) and RL (Reinforcement Learning) range from construction management to the semantic enrichment of BIM

models, and show how the use of artificial intelligence helps to solve complex problems.

After which the problem to be solved is analyzed in detail, explaining how the unions between the different rings that make up the tunnel lining and how their union allows the tunnel to adapt to the different curvatures and altimetry of the alignment.
It will then be explained how all the data necessary to solve the problem can be extracted from the TIM model, to allow the development of the algorithm in a python environment.

Then it will be explained how the algorithm has been adapted to solve the problem, both at a geometric level and as learning parameters to be adopted.
The method that has been chosen is a Reinforcement learning algorithm. Reinforcement learning is a subcategory of machine learning based on the interaction of an agent with an environment through actions and rewards.
This type of algorithm is perfectly suited to solve the proposed problem because Reinforcement learning is based not on Dataset analysis but on simulations that reflect scenarios similar to reality.
The algorithm is based on Q-learning. This method allow to link possible ring rotations, with the operating alignment that will be created, to achieve the purpose. This connection is made through a table called "Q-table".
The compilation of this table takes place through simulations that, by testing different ring positions, associate to each position a reward based on the distance from the theoretical alignment.

At the end from the Q-table it's possible to extract the coordinates and rotations of the single rings to build a lining model.

Chapter 2

Literature review

In recent years there has been a great development in the technological field, which has allowed a rapid improvement in the computational power of the machines, and in digital methodologies applicable to the building and infrastructure field, such as Building Information Modelling.
Driven by the increasing overall amount of data a sharp increase in the successful deployment of techniques of ML has been seen for different tasks.

In this chapter is analized the applications of the BIM methodology applied to the infrastructure sector, afterwards applications of ML methodologies in tunnelling, building sector and direct applications with a BIM model,are shown.

2.1 BIM methodology in infrastructure sector - InfraBIM

"A BIM is a digital representation of physical and functional characteristics of a facility. As such it serves as a shared knowledge resource for information about a facility forming a reliable basis for decisions during its lifecycle from inception onward."[1]
Although BIM is now widely used in the construction sector, its development for infrastructure is slower, due to various problems.

These problems are mainly linked to the great complexity of the infrastructure works. In the construction sector it is relatively easier to plan and determine the elements and activities to be used in construction because there are typical design solutions and classification of technological elements.

Except in special cases, all the technical elements of a building, from bricks to MEP systems, are homologated and standardised by the various legislations for which they lend themselves well to a parametric formulation.

The digital representation of an infrastructure is a difficult operation, both for the extension and for the elements that make it up, which do not have defined shapes, think for example of earth movements for the formation of a embankment or excavations for a tunnel.

In tunnelling a tunnel model can be used for different purposes. Having a digital model of the work makes it possible to know the data necessary to be able to carry out more effective analyses on the terrain or to calculate more easily the materials and elements that will be used in the construction phase.
In addition, it is possible to use the models to perform 4D simulations to better understand the construction action that is processed by the TBM.

Ninić and Koch performed a careful analysis of how the development of a TIM model works is carried out, as shown in Figure 2.1[2].

It is also interesting to understand how the current trend, in InfraBIM modelling, is to combine classic design methods with computational design. In "On-demand generation of as-built infrastructure information models for mechanised Tunnelling from TBM data: A computational design approach" [3] Getulli, Capone Bruttini and Rahimian, show how the combination of visual programming with models can allow, for example, ondemand monitoring of the enclosure construction and automatic compilation of the data available from the TBM.

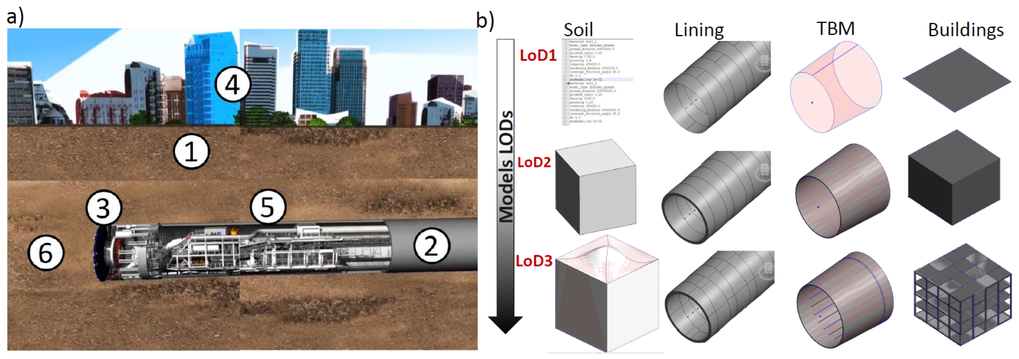


Figure 2.1: (Source: Ninić and Koch, 2017)
a) Main components of the urban tunnelling process:
1) soil, 2) lining, 3) TBM, 4)existing infrastructure, 5) grouting, 6) alignment;
b) tunnel information model with components modeled on different LoDs: soil, lining, TBM and buildings

2.2 Machine-Learning applications in tunnelling and building sector

Machine learning (ML) is the study of computer algorithms that improve automatically through experience[4]. The problems that can be tackled with machine learning can be divided into three main categories:

- **Supervised learning**: the machine is provided with a number of examples of inputs and their respective outputs and it must identify and learn the rules that link them in order to be able to provide outputs for new inputs. This category includes classification and regression problems, the aim of which is to assign new inputs to already known classes. The inputs are discrete in the case of classification and continuous in the case of regression.
- **Unsupervised learning**: only a series of inputs are provided to the machine from which it must extract patterns and structures that link them. This category includes clustering problems, whose goal is to group a series of inputs into classes not previously known: the algorithm must therefore discover the recurring patterns itself.

- **Reinforcement learning**: the machine interacts with an external, dynamic environment in which it must accomplish a given goal. For each action it carries out, the system returns a positive or negative feedback in relation to the objective to be carried out: from this feedback the machine learns. Learning by reinforcement finds applications in many branches of computer science and statistics, such as game theory or genetic algorithms, but also in other areas.

In the field of building and infrastructure only a few years ago it has begun to look for applications of machine learning algorithms. But they can be very useful in different areas.

In [5] Marcher, Erharter and Winkler carry out an analysis of the possible applications of ML in the tunnelling field is carried out:

- Autonomous support installation,
- Automatic rock (mass) classification,
- Geological prognosis updating ahead of the tunnel face,
- Overcoming of limitations in the definition of constitutive behaviour of soil and rock,
- Exploration of the applicability of reinforcement learning to fully automate different construction processes (self-driving TBMs).

The same article also describes how technological development will affect the world of construction and tunnelling in the coming years. Saying that great potential is seen in unsupervised Machine Learning approaches, where the final classification is not imposed upon the data but learned from it, also reiforcement learning seems to be trend-setting but has not been used for specific tunnel applications yet.

Finally, it describes the steps that should be taken in order to realize practical applications:

- Using ML for the optimization of design and construction phases as well as automation and operational guidance in all kinds of operations and maintenance of tunnel projects;
- Collecting as much high quality data as possible: all data from sensors on TBMs as well as jumbos, drilling rigs, excavators etc., scanning and photos from the tunnel face, all monitoring data;
- Combined use of BIM and ML;
- Standardization of input selection process and data partitioning methods;
- Standardization of ANN (Artificial Neural Network) model architectures and performance measures of ANNs for tunnel engineering.

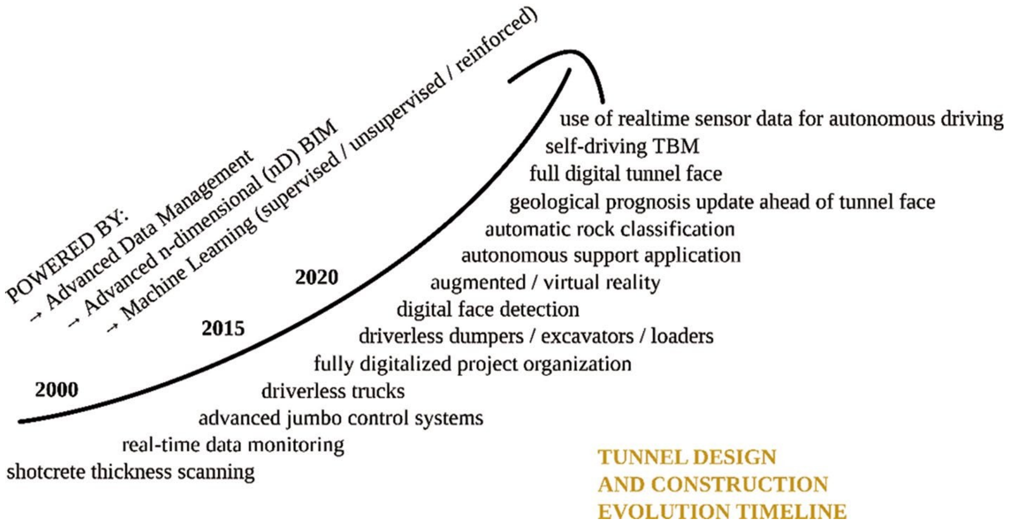

POWERED BY:
→ Advanced Data Management
→ Advanced n-dimensional (nD) BIM
→ Machine Learning (supervised / unsupervised / reinforced)
2000
2015
2020
use of realtime sensor data for autonomous driving
self-driving TBM
full digital tunnel face
geological prognosis update ahead of tunnel face
automatic rock classification
autonomous support application
augmented / virtual reality
digital face detection
driverless dumpers / excavators / loaders
fully digitalized project organization
driverless trucks
advanced jumbo control systems
real-time data monitoring
shotcrete thickness scanning
TUNNEL DESIGN AND CONSTRUCTION EVOLUTION TIMELINE

Figure 2.2: (Source: Marcher, Erharter, Winkler (2020)) Evolution timeline for digitalization in tunnelling

ML algorithms also find important applications in the energy field.
In [6] Seyedzadeh, Rahimian, Oliver, Rodriguez and Glesk described how to predict the energy performance of non-domestic buildings, this study is conducted in England where non-domestic buildings contribute 20% of the

UK's annual carbon emissions.

In the energy field we try to understand how to obtain better energy analyses than those achievable with classic software, so that we can make predictions on energy consumption trends.

The machine learning and deep learning algorithms also find a valid application to help in the decision making processes on site.

The adoption of machine learning algorithms allows, for example, to reconstruct point clouds from simple photographs taken on-site, from these models it is then possible to obtain the volumes of land to allow analysis on transport and land classification as shown in [7].

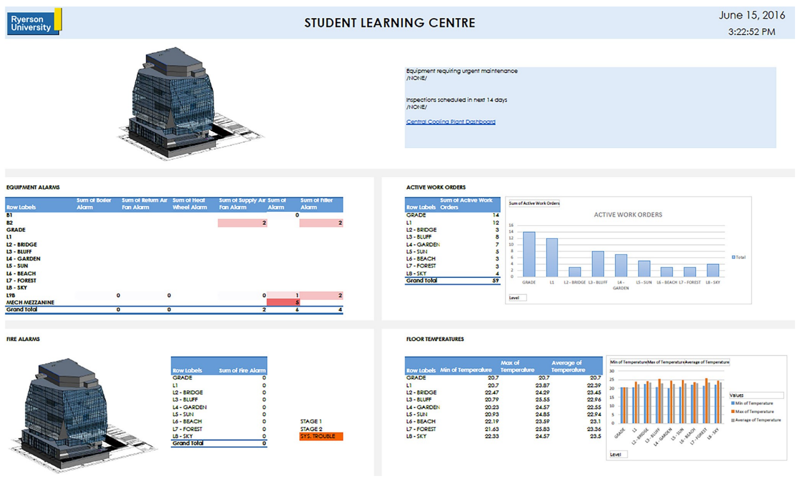
2.3 Machine-Learning and Building Information Model

Applications of machine learning algorithms become even more interesting when implemented with Building Information Models (BIM).

These methods can be used not only to read data from a BIM model, but also for parameter classification and database settings.
Several studies have shown how their application on coding and semantic enrichment gives excellent results to overcome obstacles such as code compliance checking, simulation, functional analysis and information exchange.

In [8] a method for code checking between the parameters of a BIM model and building codes is presented using a similarity-based method and a supervised learning-based method that can associate building classifications with IFC parameters.

Another area where code management and semantic enrichment is of great importance is facility management where, as shown by [9] the ML can be used to improve the management of Work orders (WO) and communicate efficiently with the facility team.



Ryerson University

STUDENT LEARNING CENTRE

June 15, 2016
3:22:52 PM

Equipment requiring urgent maintenance
/NONE/

Inspections scheduled in next 14 days
/NONE/

Central Cooling Plant Dashboard

EQUIPMENT ALARMS

Row Labels	Sum of Boiler Alarm	Sum of Return Air Fan Alarm	Sum of Heat Wheel Alarm	Sum of Supply Air Fan Alarm	Sum of Alarm	Sum of Filter Alarm
B1					0	
B2				2		2
GRADE						
L1						
L2 - BRIDGE						
L3 - BLUFF						
L4 - GARDEN						
L5 - SUN						
L6 - BEACH						
L7 - FOREST						
L8 - SKY						
L9B	0	0		0	1	2
MECH MEZZANINE					5	
Grand Total	0	0		2	6	4

ACTIVE WORK ORDERS

Row Labels	Sum of Active Work Orders
GRADE	14
L1	12
L2 - BRIDGE	3
L3 - BLUFF	8
L4 - GARDEN	7
L5 - SUN	5
L6 - BEACH	3
L7 - FOREST	3
L8 - SKY	4
Grand Total	59

FIRE ALARMS

Row Labels	Sum of Fire Alarm
GRADE	0
L1	0
L2 - BRIDGE	0
L3 - BLUFF	0
L4 - GARDEN	0
L5 - SUN	0
L6 - BEACH	0
L7 - FOREST	0
L8 - SKY	0
Grand Total	0

STAGE 1
STAGE 2
SYS. TROUBLE

FLOOR TEMPERATURES

Row Labels	Min of Temperature	Max of Temperature	Average of Temperature
GRADE	20.7	20.7	20.7
L1	20.7	23.87	22.39
L2 - BRIDGE	22.47	24.29	23.45
L3 - BLUFF	20.79	25.55	22.96
L4 - GARDEN	20.23	24.57	22.55
L5 - SUN	20.93	24.85	22.94
L6 - BEACH	22.19	23.59	23.1
L7 - FOREST	21.63	25.83	23.36
L8 - SKY	22.33	24.57	23.5

Figure 2.3: (Source: Zhang and El-Gohary(2019)) Sample dashboard (landing page with full building view).

Thanks to the data processed and associated with the model, it is possible to return dashboards summarising the status of the WO in the case study analyzed in the article.

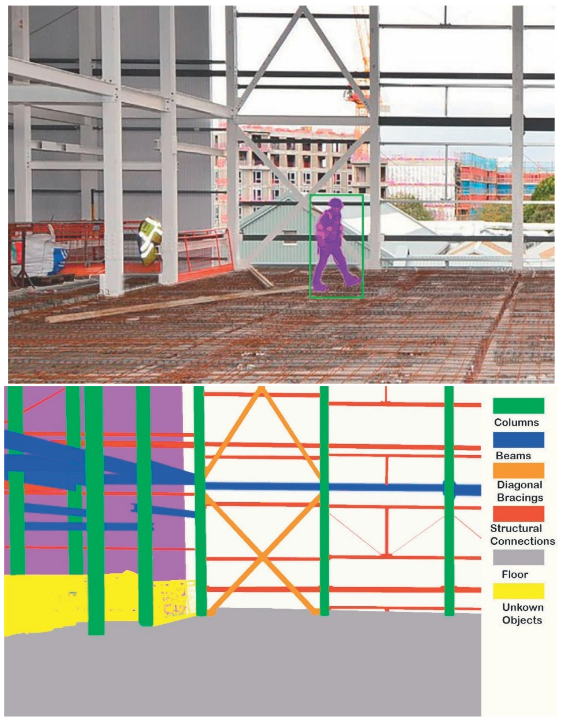


Figure 2.4: (Source: Rahimiana, Seyedzadeh, Oliver, Rodriguez and Dawood(2020)) Above creating an object mask with human recognition using CNN. Below output of semantic segmentation for identifying structural building parts).

The application of ML and DL algorithms in construction management also has great potential. The use of these methodologies together with a virtual reality application can allow on-demand monitoring of construction projects as shown by [10].

In general, in the application of machine learning methods in the construction sector, and in particular with the integration in BIM methodology, one of the major stumbling blocks is the lack of big data to use.

Chapter 3

Methodology

As illustrated above, the objective of this thesis work was to optimize, during the design phase, the process that enables to create an operational alignment for tunnels made using an automatic method.
To implement this process, it was designed an AI able to concerning the theoretical alignment under examination and the structural connections of the rings that make up the tunnel lining, to reconstruct an operational alignment as congruent as possible to the theoretical one.

In particular, the result obtained made it possible to develop a process that allow to create operational alignments with a good correspondence with the relative theoretical alignment, developing a maximum recorded deviation of 1.00 cm.
The application of this process in the design phase is very effective in predicting the model useful for the TBM (Tunnel Boring Machine) to place the structural rings making up the tunnel in the subsequent construction phase, facilitating to optimize, for example, the definition of the minimum radii of curvature.

In the first part of this chapter, the methodological approach developed for the construction of the optimization process is described in detail, with an overview of the operation of the TBMs and the structural rings that make up the tunnels.

Initially, the preparation and data collection phase necessary for the realization of the algorithm used by the artificial intelligence is described, i.e. the trend of the theoretical alignment and the rotation system of the structural rings, with possible real assembly positions.
In particular, the operations necessary to create a TIM (Tunnel Information Model) are described.

Through the use of Dynamo, visual programming platform, and Autodesk Revit plugin, it is shown how it was possible to calculate the possible positions of the ring axis examined. The rotations data were subsequently exported in Excel format, allowing to read this data in the Python programming environment.

The operation of the automatic learning system through which thecon cui artificial intelligence structured the algorithm is described below, which has enabled to obtain the most accurate operating alignment.
Through the adoption of the reinforcement learning method, is illustrated how it has been possible to structure the simulations, deepening the functioning of the algorithms based on Q-Learning and the functioning of a Q-Table, a table that describes the behavior learned by artificial intelligence.

In the sub-chapter method validation, the methodology through which the parameters enabling the definition of the maximum deviation from the theoretical alignment has been deepened. This has allowed demonstrating, moreover, how it was able to elaborate the operative alignment using the deterministic method. This will be followed by an in-depth examination of the system used to perform the simulations, to obtain the desired results.

Finally, an analysis of the results produced by the artificial intelligence is presented and, through the information contained in the Q-Table, a TIM model containing the sequence of structural rings in the determined positions is reconstructed.

3.1 Methodological approach

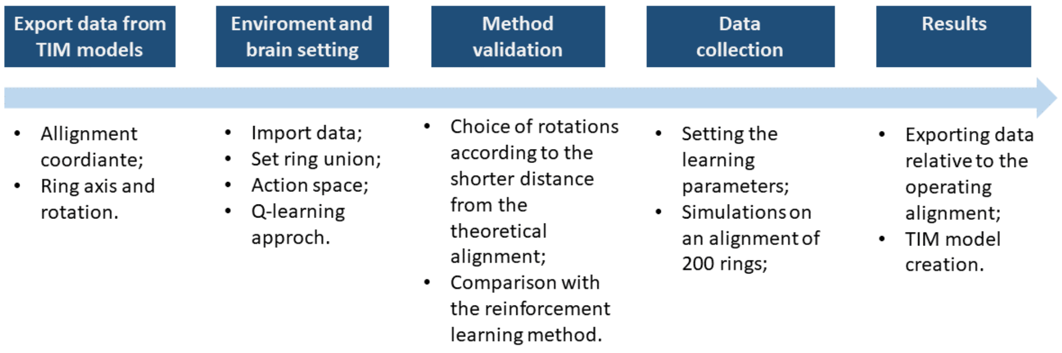

Export data from TIM models
Enviroment and brain setting
Method validation
Data collection
Results
• Allignment coordiante;
• Ring axis and rotation.
• Import data;
• Set ring union;
• Action space;
• Q-learning approch.
• Choice of rotations according to the shorter distance from the theoretical alignment;
• Comparison with the reinforcement learning method.
• Setting the learning parameters;
• Simulations on an alignment of 200 rings;
• Exporting data relative to the operating alignment;
• TIM model creation.

Figure 3.1: Methodological workflow.

The methodological approach adopted in this thesis work is based on 5 fundamental parts:

- Extraction of data from the TIM model.
  Having adopted the infraBIM methodology during the design phase, the TIM model of the tunnel under examination contained all the information and data necessary to solve the problem, create an operational alignment corresponding to the theoretical alignment. Through the TIM model, it was possible to analyze the structure of the structural rings and the positions of the relative axes on the alignment.

- Setting the Environment and Brain in Python.
  Once the necessary data had been extracted, the Environment and Brain were set up in Python language. These are fundamental parts for the construction and good functioning of a Reinforcement learning algorithm and will be better explained in subchapter 3.4.

- Definition of the operational alignment using a deterministic approach.
  Before starting with the simulations, the operating alignment for the tunnel under examination was calculated using the deterministic method. The objective was to obtain a comparison alignment, thus evaluating the effectiveness of the defined Reinforcement learning algorithm.
- Elaboration of the simulations.
  Once the phase of collecting the necessary data and creating the algorithm based on artificial intelligence functioning was completed, 18 different simulations were carried out, each with different parameters. This analysis has enabled to define the most effective settings for the resolution of the final operating alignment. Once I highlighted these parameters and set them correctly, a simulation was performed.
- Extracting the data produced by the simulation and reproducing a TIM operating alignment.
  From the simulation carried out, we extracted the relative Q-Table, a table that contains all the information produced by the Q-Learning algorithm. These data have allowed recreating the operating alignment in TIM format, with a maximum deviation from the theoretical alignment of 1.00 cm.

3.2 Mechanised tunnelling overview

When we talk about highly mechanized excavations we are referring to those tunnels that are made using TBM (Tunnel Boring Machine) machines.
This method of excavation and tunnel boring is particularly suitable for long tunnels, at least 2/3 km long, because, although the time savings in excavation and automatic construction of the structure is considerable, the cost of transport, assembly, and disassembly of the machine is significant, and not always convenient.

Excavation by TBM has carried out thanks to a cutter head that digs into the earth and rock crushing the material, the head is pushed by jacks

placed at the tail which are connected to the rings already placed [13].
Proceeding with the excavation, the TBM places the new segments to form the next rings. The excavation material produced during the operation of the machine is automatically transported outside the tunnel to be disposed of.

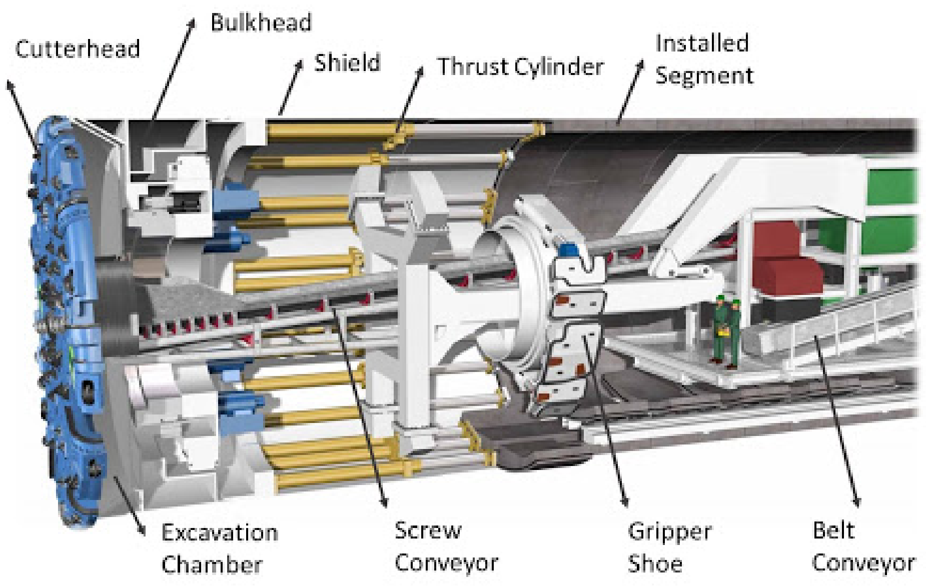


Figure 3.2: (Source: Engineering Design Director(2019)) TBM machine parts.

The guidance system of the TBM constantly monitors the trend of the tunnel axis to correct it when it deviates from the designed one.
The trend of the machine depends on numerous factors: sudden variations in the torque of the cutting head, cutting head block, unexpected variations in soil density, insufficient pressure of the pumped mortar to fill the space between the extrados of the lining and the tunnel profile. Having to deal with all these possible problems in the course of work at the moment it is not possible a priori to know the exact position of the segments that will be placed by the TBM.

3.2.1 Lining component's geometry

The rings that make up the structure of the mechanized tunnels are prefabricated elements that, as mentioned above, are placed directly by the TBM machine during the excavation phases. The rings can have different shapes [3]:

- Segments with parallel faces form straight rings - limited to straight tunnel's sections;

- Segments tapered on one side form left/right tapered rings - curved tunnel's sections;

- Segments tapered on both sides form universal tapered rings - straight and curved tunnel's sections.

In the case in question, the ring is tapered on both sides, this allows the tunnel structure to adapt easily to both straight and curved tunnel's sections by simply changing its rotation concerning the previous one, as shown in Figure 3.3.

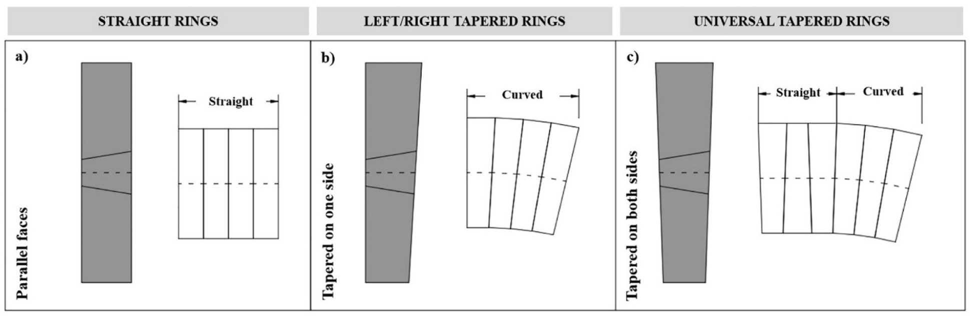
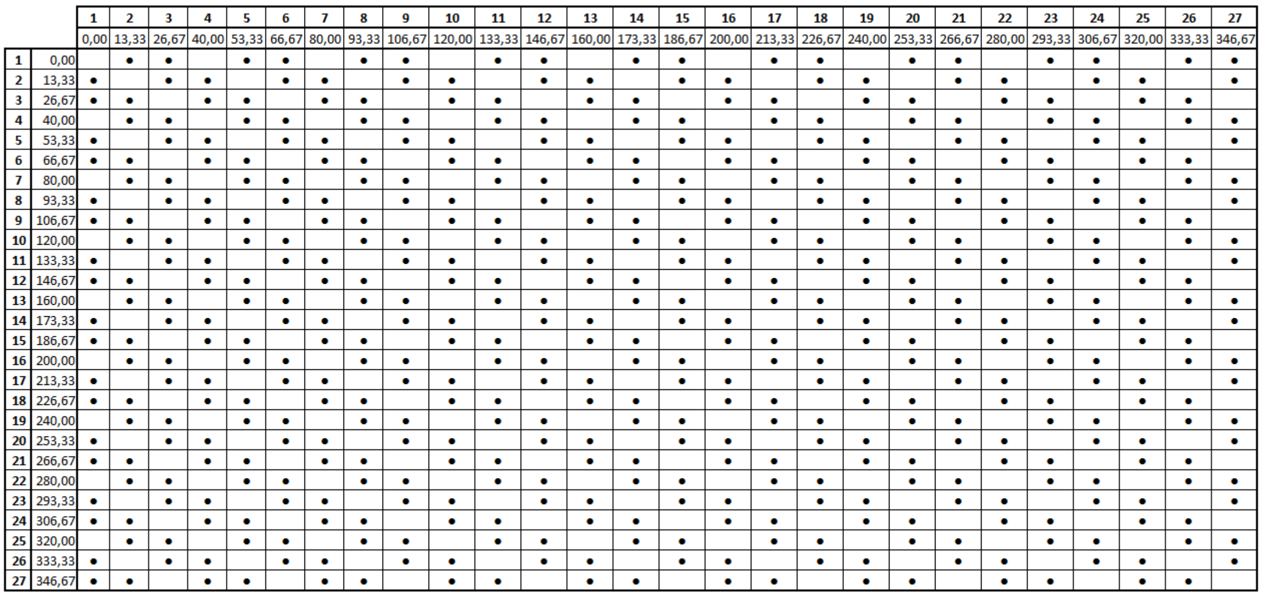


Figure 3.3: (Surce: Getuli, Capone, Bruttini and Rahimian(2020)) Segment and ring typologies: a) segments with parallel faces – straight rings; b) segments tapered on one side – left-right tapered rings; c) segments tapered on both sides – universal tapered rings.

The rings that form the structure of the tunnels are made up of several segments with radial joints, to join the segments together, and longitudinal joints, which allow the rings to be joined together.



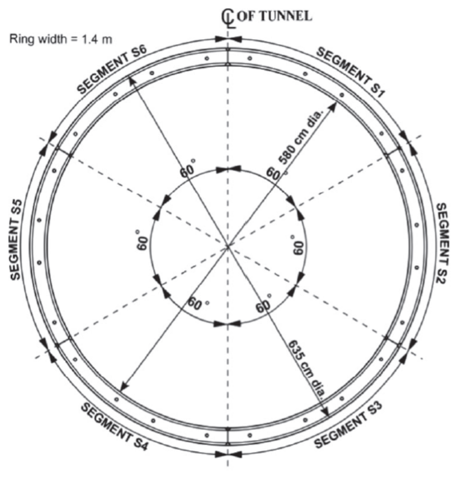
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27
		0,00	13,33	26,67	40,00	53,33	66,67	80,00	93,33	106,67	120,00	133,33	146,67	160,00	173,33	186,67	200,00	213,33	226,67	240,00	253,33	266,67	280,00	293,33	306,67	320,00	333,33	346,67
1	0,00		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●
2	13,33	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●
3	26,67	●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●	
4	40,00		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●
5	53,33	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●
6	66,67	●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●	
7	80,00		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●
8	93,33	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●
9	106,67	●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●	
10	120,00		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●
11	133,33	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●
12	146,67	●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●	
13	160,00		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●
14	173,33	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●
15	186,67	●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●	
16	200,00		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●
17	213,33	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●
18	226,67	●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●	
19	240,00		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●
20	253,33	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●
21	266,67	●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●	
22	280,00		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●
23	293,33	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●
24	306,67	●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●	
25	320,00		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●
26	333,33	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●
27	346,67	●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●		●	●	

Figure 3.4: Positioning matrix.

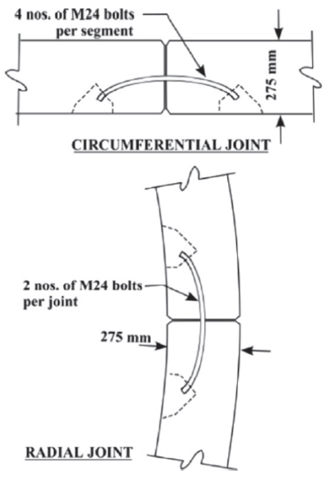
Depending on the number of longitudinal joints, the possible positions of the ring are defined it is necessary that, despite the rotations, the longitudinal joints are well aligned.
A positioning matrix is therefore defined which will be used by the TBM to allow the correct positioning of the segments.

This matrix is processed in such a way that it does not allow a new segment that must be positioned to connect only to the longitudinal joints of a single previously positioned segment. This matrix, therefore, depends on the type of segment chosen for the construction of the ring. Different types of segments can be used for mechanized tunnels: Rectangular lining, Rhomboidal lining, Trapezoidal lining.

In the case in question, a ring composed of 9 trapezoidal segments with 3 longitudinal joints each, 27 different positions are therefore possible Figure 3.5 and the positioning matrix will be as indicated in Figure 3.4.



(a) Tunnel cross section.


4 nos. of M24 bolts per segment
275 mm
CIRCUMFERENTIAL JOINT
2 nos. of M24 bolts per joint
275 mm
RADIAL JOINT

(b) Joint details.

Figure 3.5: (Source: Koneshwaran, Thambiratnam and Gallage (2015))Ring geometry example.

3.3 Tunnel Information Model (TIM)

Koch and Vonthron in [14] say that "all relevant data needed for the planning, construction and maintenance of tunnels is collected, classified, structured and linked into a holistic, object-oriented Tunnel Information Model" this model is called TIM.
A complete TIM model must consist of several parts:

- The model of the lining, even if the precise position of the segments up to the tunnel realization using the TBM is unknown, it is fundamental already during the design phase to have a model of the tunnel lining to optimize the analysis and execution of the subsequent phase of the excavation;
- Soil model, a complete TIM model must also contain information relating to the soil in which the tunnel will be built; this data is derived from geognostic and geological surveys.
- TBM model, all information concerning the TBM must also be included within the TIM model: thrust force, torque, excavation time, pressure

monitoring system, and the transport system of the excavated material. This model is particularly useful when excavation simulations are performed.

- A model of the territory, this model represents the environment built in the area where the tunnel will be located, it is, therefore, geo-referenced and is useful to evaluate possible subsidence due to the excavation.

Since the design of a tunnel is a very complex project, to create a complete and correct model it is necessary to use different software, in the case under consideration only the lining model has been analyzed.

For the realization of the lining model, it was decided to use the Autodesk Revit software, while Civil3D software was used for the design of the alignment. For the realization of a TIM model, however, it is necessary to make the two software communicate to have all the information in a single model. To allow this dialogue, the Dynamo visual programming plugin is used, through which you can export the coordinates of the alignment from Civil3D to a .xlsx file and, reading this file, recreate the alignment in Revit.

3.3.1 Ring modelling and data extraction

The modeling of the ring that forms the tunnel structure started from a .dxf file containing the geometries of the segments and the ring that was designed. This file was imported into a Revit family so that the geometries could be converted into a solid, the ring family thus produced was in turn loaded into a generic adaptive model so that the points needed for the location and an R rotation parameter could be linked.

After re-creating the geometry of the ring in the program, the necessary data was extrapolated to proceed with the application of the algorithm, these exports were made by converting the necessary data into Excel tables using the Dynamo plugin.

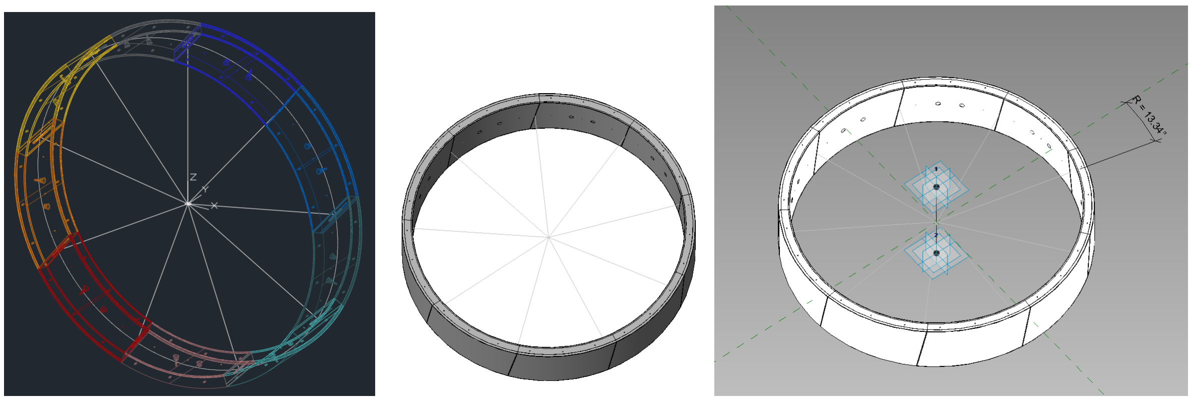


Figure 3.6: On the left, the ring in .dxf format, in the middle the model converted to Revit fault, on the right the adaptive model of the ring.

The alignment has been divided into points 2m apart and the coordinates (X, Y, Z) of these points have been exported in a file called "T-Alignment.xlsx", this file will allow to reconstruct the theoretical alignment through the python programming language.
The same alignment is also divided into 4 cm long parts and exported in a file called "R-Alignment.xlsx", this second file will be used to calculate the rewards during the following simulations.

The last data that is needed comes from the geometry of the ring and in particular concerns the change of the ring axis at different rotations.
To extrapolate this data, the ring was recreated in Dynamo's modeling environment and, by binding one of the two faces, the 27 possible rotations were performed. From this operation, axis position changes were recorded and this data was exported to a file called "Positions.xlsx".

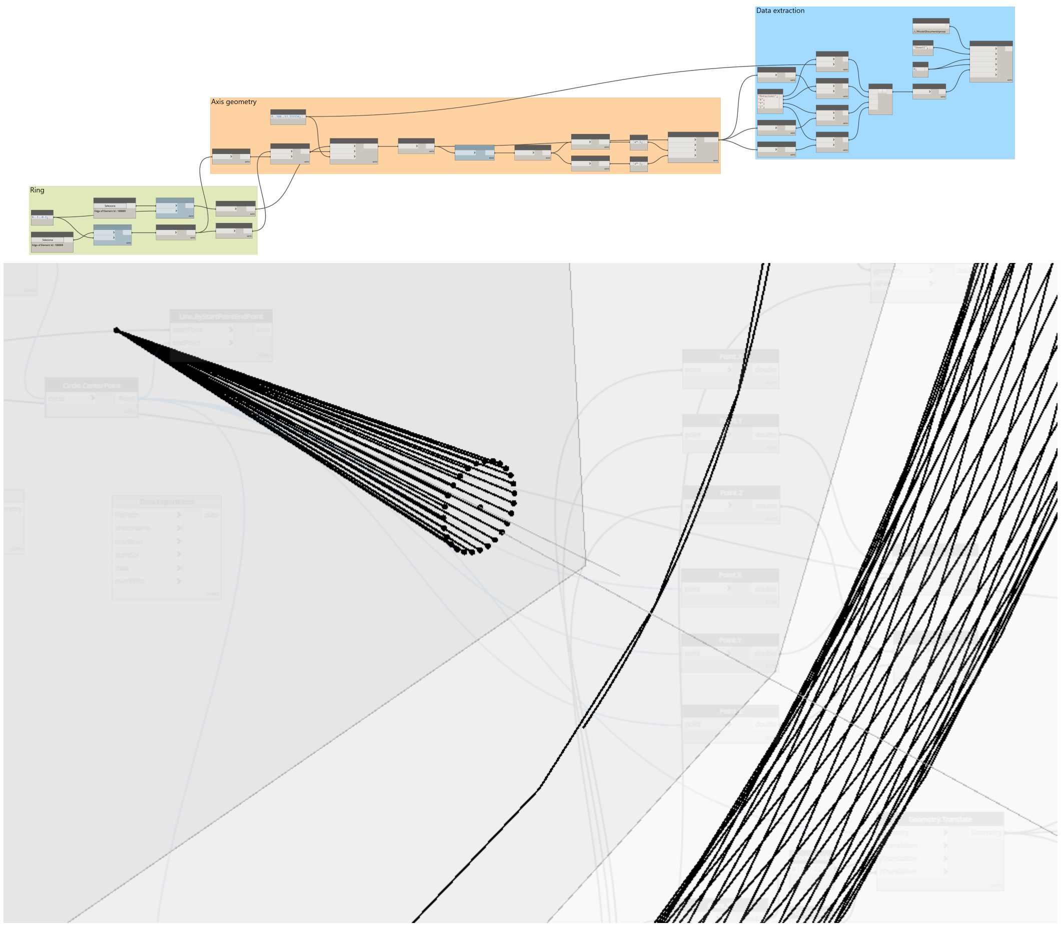


Figure 3.7: Above the dynamo script.
Below the geometry of the axis at the different rotations.

3.4 Reinforcement learning

Abhishek Nandy and Manisha Biswas in [15] say that "Reinforcement Learning (RL) is an approach through which intelligent programs, known as agents, workin a known or unknown environment to constantly adapt and learn based on giving points."

Reinforcement learning implements the most natural concept of learning: interact with an environment and react according to a feedback.
An RL algorithm can be thought as infant who is completely unaware of what

surround it and slowly start eating, waving is arms, playing, etc. without an explicit teacher, but through a sensorimotor connection with its environment.
This connection produces a flow of information enabling the infant to learn cause-effect relationships, consequences of actions and how to achieve goals, which are then used in the future when similar conditions appear. RL builds this procedure with four elements: agent, environment, action space and rewards.

By keeping the infant example, the infant is the agent, the environment could be the house in which he is living, the action space is composed by its possible behaviour and the rewards are the outcomes of its actions. For instance, the infant see a lemon (environment), decides to eat the lemon (action) and then it feels the sourness (reward). Clearly, an algorithm is way less complicated then a sentence being, but interestingly it can learn how to perform complicated tasks if guided in the right direction.

In practice RL problems involve learning a map environment-action, called policy, which tells what action to perform according to the state of the environment in order to maximize a numerical rewards. In a single step the agent observes the environment state, takes an action according to the current policy, observe a reward and change the policy according to what has been observed. In complicated application, this could be extended also to future rewards, indeed current actions could affect not only the next state, and so the next reward, but also future states and so future rewards.

3.4.1 Q-Learning approach

The mathematics behind reinforcement learning is based on Markov decision processes (MDP) [16]. This models are used to model decision making where on the one hand a part of the outcome is random and on the other hand it is guided by a decision maker. An MDP consists of:

- a set of states $\mathcal{S}$;

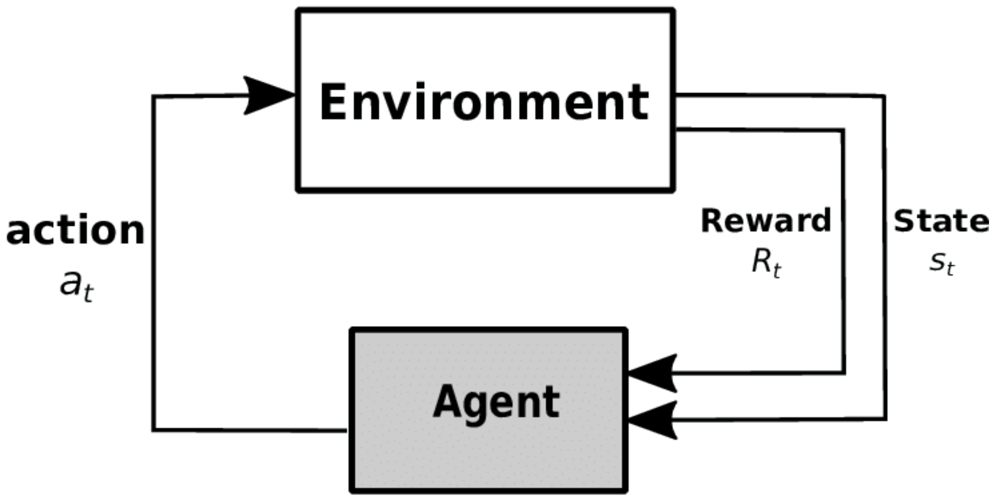


Figure 3.8: Reinforcement lerning mechanism.

- a set of actions $\mathcal{A}$;
- a transition probability $p(s'|s, a)$ with $s, s' \in \mathcal{S}$ and $a \in \mathcal{A}$ (or transition function in deterministic cases);
- a reward function $r(s, a)$ with $s \in \mathcal{S}$ and $a \in \mathcal{A}$.
- a discount factor $\gamma \in (0, 1)$.

The MDP is then represented by the realization of states, actions and rewards, meaning that over time there is an evolving state S_t, an action that is taken A_t and a reward that is outputted R_t. This formalizes mathematically what people call environment, and precisely the agent-environment interaction:

1. the agent observes state s_t;
2. the agent chooses action a_t;
3. the environment output the reward r_t according to $r(s_t, a_t)$ and the new state s_{t+1} according to $p(\cdot|s_t, a_t)$.

Remark that capital letters are used for random variables, before actually observing the values, while lower case is used for realizations, what is actually observed.

In general, the agent would like to optimize not only the current reward, but also the future ones by using the discount factor γ (discounted reward). This optimization is done according to a policy $\pi(s)$ which tells how the agent chooses the action to take given the state $s \in \mathcal{S}$. The combination of the discounted reward and policy creates the well-known quality function (Q-function):

$$Q^{\pi}(s,a) = \mathbb{E}\left[\sum_{\tau=t}^{\infty} \gamma^{\tau-t} R_{\tau} \middle| S_t = s, A_t = a\right], \tag{3.1}$$

where $\mathbb{E}$ represent the expectation of the random variable $\sum_{\tau=t}^{\infty} \gamma^{\tau-t} R_{\tau}$, with $R_{\tau} = r(S_{\tau}, A_{\tau})$ and $A_{\tau} = \pi(S_{\tau})$. The intuition behind this quantity is that the agent wants to quantify the future rewards and she/he gives them more or less importance to future depending on the chosen discount factor γ.

However, future rewards have a stochastic nature which requires the use of the expectation to quantify in mean how they behaves. As it is going to be explained later, the stochastic element could be introduced by the transition p or the policy π or both or not present at all, but this $\mathbb{E}$-notation keeps the range of applications as general as possible. Ideally, the policy π should be chosen in such a way that maximizes Q^{π}, or better it should output the sequence of actions that maximizes $Q^{\pi}(s,a)$ given the current state s and the current action a. The maximum of Q^{π} is called $Q^{\star}$ and defined as follows:

$$Q^{\star}(s,a) := \max_{\pi} Q^{\pi}(s,a). \tag{3.2}$$

Under the deterministic policy $\pi^{\star}(s) := \arg\max_{a \in \mathcal{A}} Q^{\star}(s,a)$ the optimal Q-function $Q^{\star}$ satisfies the Bellman equation:

$$Q^{\star}(s,a) = \mathbb{E}\left[R_t + \gamma \max_{a' \in \mathcal{A}} Q^{\star}(S_{t+1}, a') \middle| S_t = s, A_t = a\right]. \tag{3.3}$$

Intuitively, this equation represents the recursive nature of the Q-function. If at time $t+1$ we are able to maximize the Q-function, then surely we are able to express the Q-function at time t by using the maximized Q-funtion at $t+1$ discounted by γ and the current reward R_t.

The next step is to learn $Q^\star(s,a)$, which can be seen as a table with the states as rows and the actions as columns. This is possible only if $\mathcal{S}, \mathcal{A}$ are finite and relatively low-dimensional. In the case of continuous and/or high-dimensional state space and action space $Q^\star(s,a)$ can be approximated with deep neural networks [17], but this is behind the scope of this work. Consider the case of $Q^\star(s,a)$ as a table, this is known in the literature as Q-table. Define then $Q^\star$ as a matrix:

$$Q^\star = (q_{s,a})_{s\in\mathcal{S},a\in\mathcal{A}}, \tag{3.4}$$

then from the Bellman equation (3.3) we obtain:

$$q_{s,a} = \mathbb{E}\left[R_t + \gamma \max_{a'\in\mathcal{A}} q_{S_{t+1},a'} \middle| S_t = s, A_t = a\right]. \tag{3.5}$$

It can be then observed that there are two important definition of the elements in the Q-table:

- the target value $q_{s,a}$;
- the Bellman equation value:

$$\mathbb{E}\left[R_t + \gamma \max_{a'\in\mathcal{A}} q_{S_{t+1},a'} \middle| S_t = s, A_t = a\right] \approx r_t + \gamma \max_{a'\in\mathcal{A}} q_{s_{t+1},a'}.$$

Note that the approximation of the Bellman equation value becomes an equal when the overall procedure is deterministic, which is indeed the scenario treated in this thesis. It is known from (3.3) that those two elements should be equal. Hence the distance between target value $q_{s,a}$ and the Bellman equation value should be quantified, this can be done with:

$$L(q_{s,a}) = \left(q_{s,a} - r_t + \gamma \max_{a'\in\mathcal{A}} q_{s_{t+1},a'}\right)^2, \tag{3.6}$$

which should be obviously minimized with respect to the target value $q_{s,a}$. To do so a common approach is gradient descent [19] which consists of minimizing a function $L(x)$ by moving on the direction of the gradient $\nabla L(x)$ with a step λ (learning rate), mathematically:

$$x = x - \lambda \nabla L(x). \tag{3.7}$$

Algorithm 1 Q-table learning

```
1:  Input the number of episodes $E$, the horizon $T$, the learning rate $\lambda$
2:  Initialize $q_{s,a} = 0$ for all $a \in \mathcal{A}, s \in \mathcal{S}$
3:  for $e = 1, \ldots, E$ do
4:      Set the initial state $s_0$
5:      for $t = 1, \ldots, T$ do
6:          Choose action $a_{t-1}$
7:          Input $s_{t-1}, a_{t-1}$ in the environment
8:          Observe the new state $s_t$ and the previous reward $r_{t-1}$
9:          Update the Q-table:
10:             $q_{s_{t-1},a_{t-1}} = q_{s_{t-1},a_{t-1}} - \lambda \left( q_{s_{t-1},a_{t-1}} - r_{t-1} + \gamma \max_{a' \in \mathcal{A}} q_{s_t,a'} \right).$
```

In the case of Q-learning $\nabla L(q_{s,a}) = 2(q_{s,a} - r_t + \gamma \max_{a' \in \mathcal{A}} q_{s_{t+1},a'})$ then the gradient descent update with learning rate λ is going to be:

$$q_{s,a} = q_{s,a} - 2\lambda \left(q_{s,a} - r_t + \gamma \max_{a' \in \mathcal{A}} q_{s_{t+1},a'} \right), \tag{3.8}$$

where without loss of generality we can exclude the 2 from the equation (choose $\lambda = \lambda/2$). Given this learning rule, the only things that remains to be set are the horizon T and the number of episodes E.
The horizon T represents how long the experiment is going to be, how many states, rewards and actions are observed. The number of episodes tells how many times we repeat the experiments, how many experiments of horizon T are run. The overall procedure is synthetized in Algorithm 1.

To summarize the overall message of the section.

- Given an environment, the interaction agent-environment can be modelled with a Markov decision process.

- The agent wants to maximize the discounted future reward given the current action and the current state, which is called Q-function and denoted with Q^{π}. This means that per each combination of action and state the maximum value of Q^{π} has to be considered, which is denoted with $Q^{\star}$.

- Given $Q^{\star}$ the agent can choose action according to the maximum score given the current state. Under this deterministic policy $Q^{\star}$ satisfies the Bellman equation.

- $Q^{\star}$ is unknown, but it can be represented, under certain conditions, as a table called Q-table. This Q-table has to satisfies the Bellman equation.

- Given the constraint of the Bellman equation optimize the Q-table to minimize the distance between the elements of the Q-table and the respective Bellman equation formulation.

In practice, given the state s choosing the action as $\arg\max_{a' \in \mathcal{A}} q_{s,a'}$ during training raises exploration problems. This means that the gradient descent is not able to explore all the possible trajectories, but on contrast it remains stuck on local optima. Such an issue is called exploration-exploitation dilemma[18], meaning that there is a trade-off between exploring and finding the actual optima. Precisely, on one hand we can exploit the maxima straight away (exploitation), but this lacks in exploring all the possible solutions; on the other hand we can explore all the possible solutions (exploration), but this going to be slow and it does not ensure to go in the right direction.
To overcome this problem a balance between exploration and exploitation has to be found, a good compromise is ϵ-Greedy [20]. ϵ-Greedy simply chooses with probability ϵ the $\arg\max_{a' \in \mathcal{A}} q_{s,a'}$ and with probability $1 - \epsilon$ a random action in the action space. Algorithm 1 is then reformulated as follow.

Algorithm 2 Q-table learning: ϵ-Greedy

```
1:  Input the number of episodes $E$, the horizon $T$, the learning rate $\lambda$
2:  Initialize $q_{s,a} = 0$ for all $a \in \mathcal{A}, s \in \mathcal{S}$
3:  for $e = 1, \dots, E$ do
4:      Set the initial state $s_0$
5:      for $t = 1, \dots, T$ do
6:          Choose action:
7:              $a_{t-1} = \begin{cases} \arg\max_{a' \in \mathcal{A}} q_{s_{t-1},a'} & \text{with prob. } \epsilon \\ \text{sample uniformly from } \mathcal{A} & \text{with prob. } 1-\epsilon \end{cases}$
8:          Input $s_{t-1}, a_{t-1}$ in the environment
9:          Observe the new state $s_t$ and the previous reward $r_{t-1}$
10:         Update the Q-table:
11:             $q_{s_{t-1},a_{t-1}} = q_{s_{t-1},a_{t-1}} - \lambda \left( q_{s_{t-1},a_{t-1}} - r_{t-1} + \gamma \max_{a' \in \mathcal{A}} q_{s_t,a'} \right).$
```

3.4.2 Enviroment and agent setup

To use the methodology described above, it is, therefore, necessary to set the environment and the agent that will interact with it correctly.

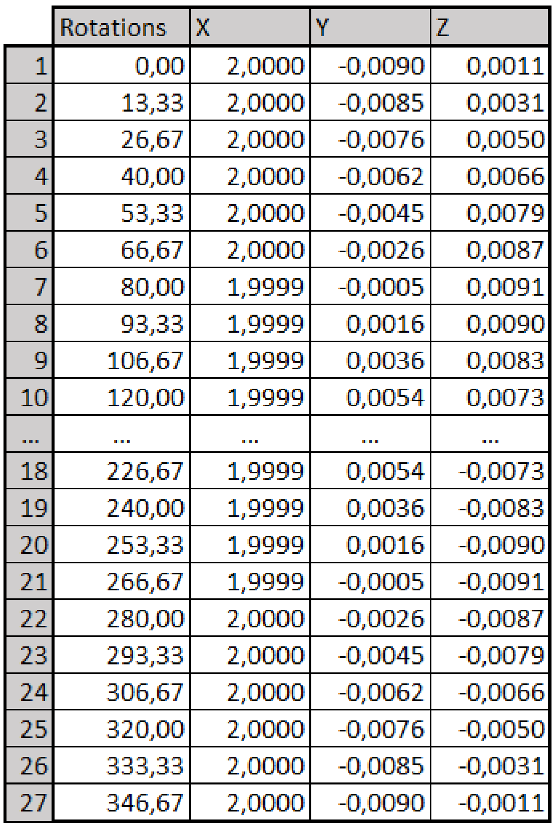
These settings start with reading the data extracted from the TIM model, as described in 3.3.1, reading the "T-Alignment.xlsx" file produces the axis of the theoretical alignment, which is used as a graphic comparison for setting the operating alignment that will be built as a concatenation of the axes of the following rings.

The "Positions.xlsx" Figure 3.9 file is used to set the actions space, $\mathcal{A}$, this file consists of a first column with the ID of the different actions, i, a column with the associated rotations, and finally 3 columns with the a_x, a_y, a_z increments of the ring axis to the different rotations.

So, the action space, is defined as follows:

$$\mathcal{A}_{i=[1,27]} = \{a_x^i, a_y^i, a_z^i\} \tag{3.9}$$

The positioning of the axes of the rings that make up the structure of the



	Rotations	X	Y	Z
1	0,00	2,0000	-0,0090	0,0011
2	13,33	2,0000	-0,0085	0,0031
3	26,67	2,0000	-0,0076	0,0050
4	40,00	2,0000	-0,0062	0,0066
5	53,33	2,0000	-0,0045	0,0079
6	66,67	2,0000	-0,0026	0,0087
7	80,00	1,9999	-0,0005	0,0091
8	93,33	1,9999	0,0016	0,0090
9	106,67	1,9999	0,0036	0,0083
10	120,00	1,9999	0,0054	0,0073
...	...	...	...	...
18	226,67	1,9999	0,0054	-0,0073
19	240,00	1,9999	0,0036	-0,0083
20	253,33	1,9999	0,0016	-0,0090
21	266,67	1,9999	-0,0005	-0,0091
22	280,00	2,0000	-0,0026	-0,0087
23	293,33	2,0000	-0,0045	-0,0079
24	306,67	2,0000	-0,0062	-0,0066
25	320,00	2,0000	-0,0076	-0,0050
26	333,33	2,0000	-0,0085	-0,0031
27	346,67	2,0000	-0,0090	-0,0011

Figure 3.9: "Position.xlsx" value.

tunnel begins with the positioning of the first ring with the axis coinciding with the theoretical alignment, then the axes of the following rings will be positioned starting from the endpoint of the previous axis.

To carry out this operation, it was necessary to apply rigid translation and rotation operations on the reference system of the new axis that is placed.

Starting from the first point of the placed axis the first operation that is performed is the rotation on the XY plane and around the z-axis of an α quantity equal to the angle between the x-axis and the x'-axis Figure 3.10.

$$\begin{cases} x = x' cos(-\alpha) + y' sin(-\alpha) \\ y = -x' sin(-\alpha) + y' cos(-\alpha) \end{cases} \tag{3.10}$$

The second operation to be performed is the rotation on the XZ plane around the y-axis of a *beta* quantity equal to the angle between the x-axis and the x' axis to align the two reference systems perfectly.

$$\begin{cases} x = x' cos(-\beta) + z' sin(-\beta) \\ z = -x' sin(-\beta) + z' cos(-\beta) \end{cases} \tag{3.11}$$

Finally, since the systems are aligned, it is necessary to apply a rigid translation of the new system of a length equal to the axis previously placed; with the new reference system thus placed, it is possible to proceed with the positioning of the new axis, with the parameters relative to the rotation chosen in the action space Figure 3.10.

$$\begin{cases} x = x + a^i_x \\ y = y + a^i_y \\ z = z + a^i_z \end{cases} \tag{3.12}$$

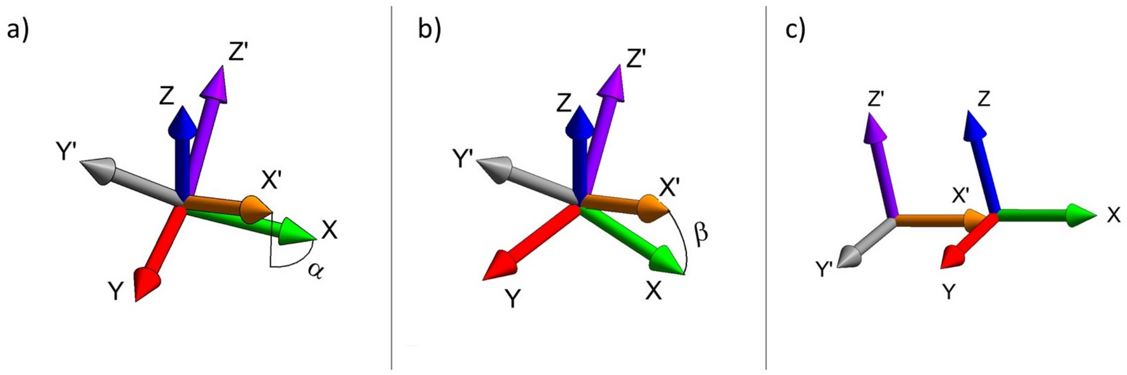


Figure 3.10: a) xy rotation, b) xz rotation, c) Traslation

Repeating these operations for each loop that needs to be placed will then obtain the operating alignment Figure 3.11.

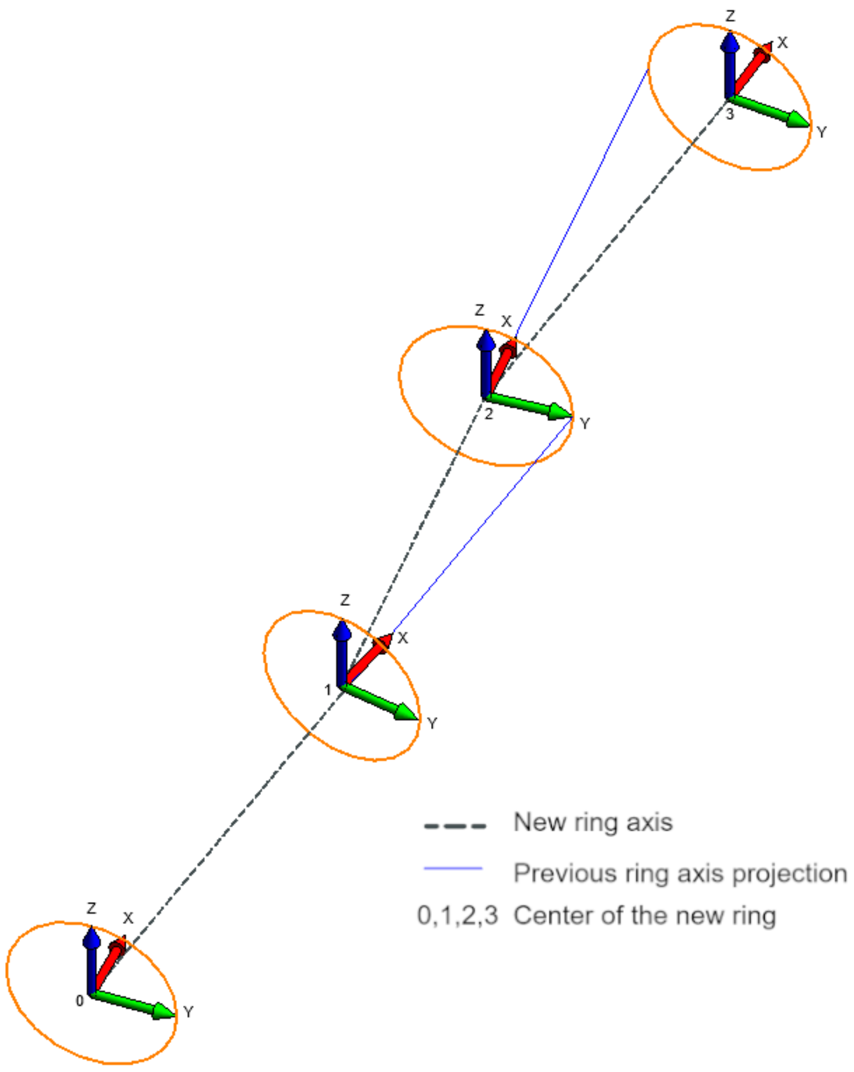


Figure 3.11: assembly of ring axes

Each action is then associated with a state described by the coordinates of the end of the axis.

After setting the problem from a geometric point of view, we move on to the definition of the mechanism for calculating the rewards.

The rewards are calculated based on the distance between the end of the axis is placed (i.e. the new state), and the points in the "R-Alignment.xlsx" file.

Since the rewards that are assigned to the different states must be positive, and higher when the state being analyzed is closer to the theoretical alignment, the following form has been used for their calculation:

$$r = e^{-\sqrt{(s_x - x^\star)^2 + (s_y - y^\star)^2 + (s_z - z^\star)^2}} \tag{3.13}$$

Putting the negative distance as the exponent of the exponential what happens is that for a distance equal to 0, and therefore with an operative alignment coinciding with the theoretical alignment at that point, the reward will be 1 while for a distance equal to 1 m the reward will be 0.37 and so on.

A tolerance parameter is also set, i.e. a maximum distance that if exceeded by a negative reward equal to -5 Figure 3.12.

When the new operating alignment exceeds the tolerance value the simulation ends and it is possible to analyze the data obtained.

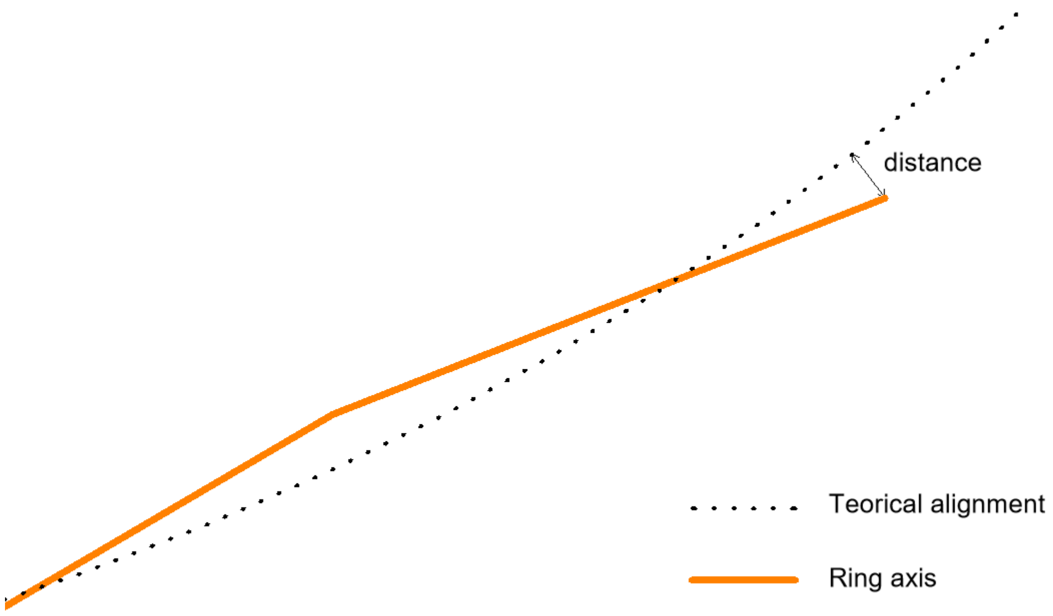


Figure 3.12: Calculation of rewards

3.4.3 Reinforcement Learning brain

The reinforcement learning brain is part of the algorithm where decisions are made to successfully achieve the goal.

The environment sets the first state with the position of the axis coinciding with the theoretical alignment, then the next a_t action is chosen in the brain.

The actions, as previously described, are chosen within the action space, to guarantee the correct choice of possible actions, however, the action space is put in relation with the positioning matrix described in 3.2.1; in this way the action space also varies according to the action chosen previously, preventing the concatenation of actions that cannot be in sequence. With the choice of the action a_t new state s_{t+1} is therefore also determined, and from s_{t+1} a reward r_t is calculated, which is used to update the $q_{s,a}$ of the a_t

that led to the state s_t.

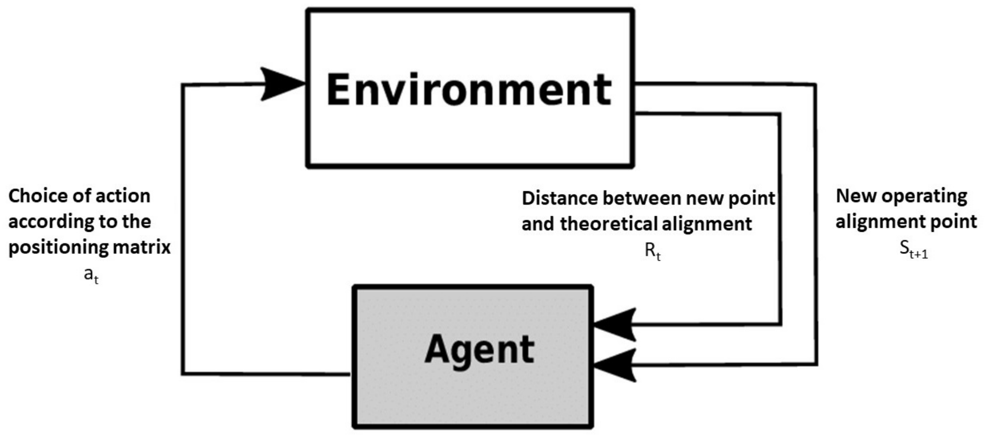

Environment
Agent
Choice of action according to the positioning matrix
a_t
Distance between new point and theoretical alignment
R_t
New operating alignment point
S_t+1

Figure 3.13: RL workflow

3.4.4 Baseline

After setting the algorithm's environment, agent, and brain, before proceeding with the simulations, and then with its training, an operational alignment is calculated using a deterministic method.

This method uses the same functions that have been described previously but, instead of using the distance to calculate rewards, it identifies among the possible actions the one that minimizes the distance between the axis of the new ring and the theoretical alignment as shown in 3.

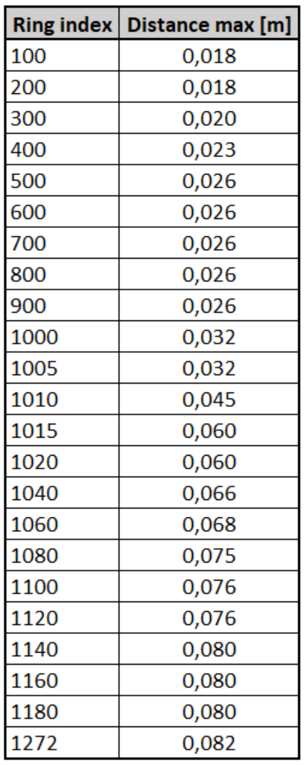
Algorithm 3 Baseline

```
1: Input the number of horizon $T$
2: Set the initial state $s_0$
3: for $t = 1, \dots, T$ do
4:     Choose action:
5:         $a$ = action that minimize distanze from theoretical alignment
6:     Input $s_{t-1}, a_{t-1}$ in the environment
7:     Observe the new state $s_t$
```

The calculation of the baseline has been performed both on a alignment of about 400 m, composed of 200 rings, (which will be the number of hives that will also be used for the application of the RL algorithm) and on a alignment longer than about 2500 m.

On the first alignment, a maximum deviation of about 1.8 cm from the theoretical alignment was obtained, while on the longest alignment the deviation that is recorded reaches 8.8 cm. By carrying out intermediate tests, it can be seen that by adopting a deterministic approach, increasing the length of the theoretical alignment also increases the deviation of the operational alignment from it Figure 3.14

Furthermore, it can be noted that, between ring 1005 and ring 1040, the deviation between the alignments increases considerably, due to the presence of a curve, and the impossibility of a deterministic approach to predict it.
Since on the first 200 rings the baseline has obtained a maximum deviation from the theoretical aligmnet of 1.8 cm, it was decided to set the tolerance parameter, describe in 3.4.2 to 1 cm.



Ring index	Distance max [m]
100	0,018
200	0,018
300	0,020
400	0,023
500	0,026
600	0,026
700	0,026
800	0,026
900	0,026
1000	0,032
1005	0,032
1010	0,045
1015	0,060
1020	0,060
1040	0,066
1060	0,068
1080	0,075
1100	0,076
1120	0,076
1140	0,080
1160	0,080
1180	0,080
1272	0,082

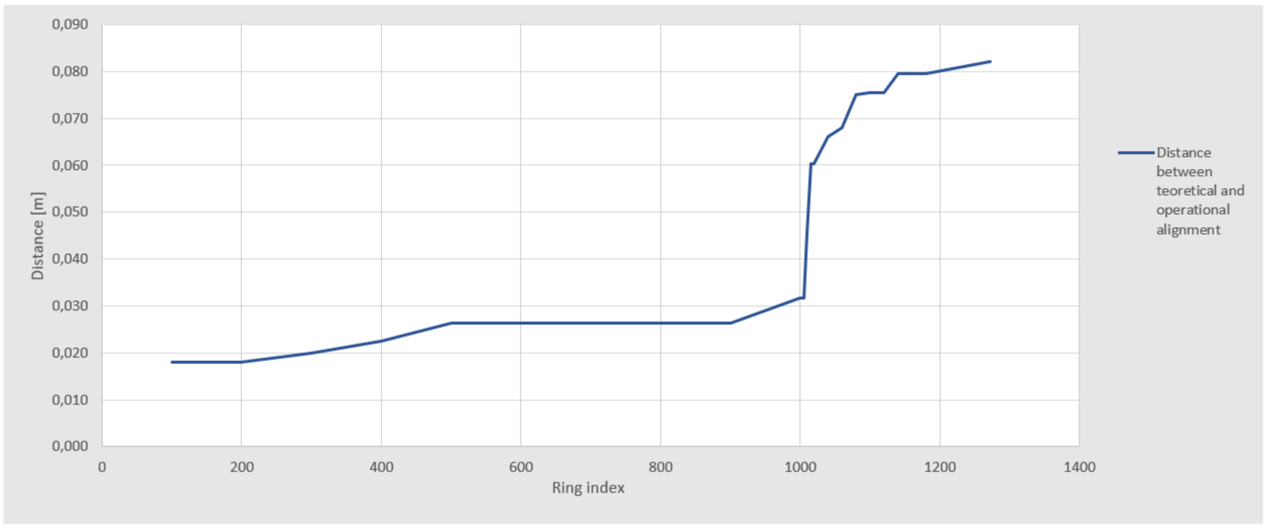


Figure 3.14: Above the Baseline table.
Below a graph showing the trend of the deviation from the theoretical alignment as the number of rings increases.

3.4.5 Method validation

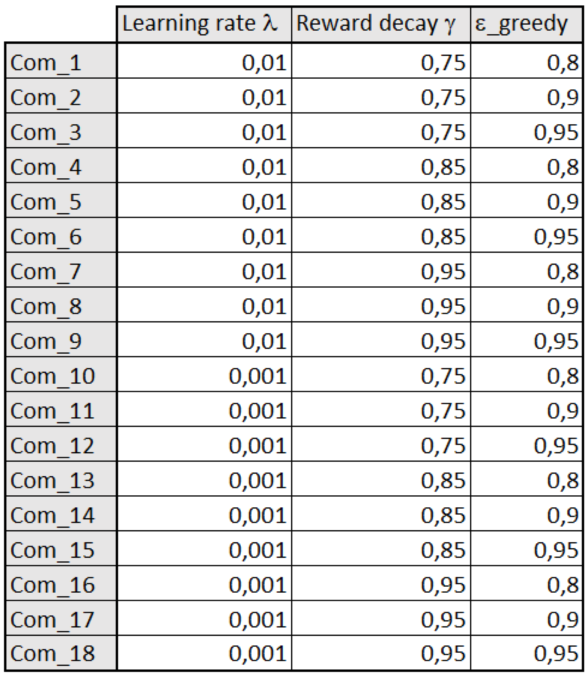
As explained in 3.4.1 the Q-learning algorithm used depends on 3 main learning factors:

- discount factor, or Reward decay *gamma*;
- the learning rate *lambda*; item *epsilon*-Greedy.

The choice of these parameters can improve, or worsen, the learning capacity of the algorithm, measurable through the function $L(q_{sa})$ minimized through gradient descent.
The lower the value of the function, the better the result will be since it means that a good approximation of the Bellman equation has been obtained.
To set the parameters correctly and therefore understand which combination is better to guarantee good learning of the algorithm it is necessary to make some simulations and evaluate the results obtained.

18 simulations of 10000 episodes were carried out to set the learning parameters.

This number of episodes is not enough to solve the whole alignment but it allows to evaluate the learning parameters without losing too much time in the computation.
The purpose of this operation is to obtain the best parameters to perform a longer simulation that manages to place 200 loops correctly.
The table Figure 3.15 shows the values of the parameters used in the different combinations.



	Learning rate λ	Reward decay γ	ε_greedy
Com_1	0,01	0,75	0,8
Com_2	0,01	0,75	0,9
Com_3	0,01	0,75	0,95
Com_4	0,01	0,85	0,8
Com_5	0,01	0,85	0,9
Com_6	0,01	0,85	0,95
Com_7	0,01	0,95	0,8
Com_8	0,01	0,95	0,9
Com_9	0,01	0,95	0,95
Com_10	0,001	0,75	0,8
Com_11	0,001	0,75	0,9
Com_12	0,001	0,75	0,95
Com_13	0,001	0,85	0,8
Com_14	0,001	0,85	0,9
Com_15	0,001	0,85	0,95
Com_16	0,001	0,95	0,8
Com_17	0,001	0,95	0,9
Com_18	0,001	0,95	0,95

Figure 3.15: Value of differente combinations.

In the first 9 simulations that are carried out using a learning rate of 0.01, as explained above we can say that this parameter indicates how much the agent is willing to overwrite the old rewards with the new ones.

For low values of the learning rate the algorithm takes more time to learn and overwrite the information, as can be seen by comparing the simulations from 1 to 9 with the simulations from 10 to 18, with learning rate 0.01, the first ones already after about 5000 episodes change the inclination of the curve, indicating that the learning is proceeding faster Figure 3.16.

The simulations 3, 6, 9, 12, 15, 18 are performed with a value of *epsilon*-Greedy equal to 0.95 the fact that this value is high and close to 1 indicates that the exploration of the algorithm is limited and not all states will be examined. From the comparison of these graphs with the respective simulations with *epsilon*-Greedy more bases it can be noticed how the above mentioned combinations always reach lower values, and therefore a better

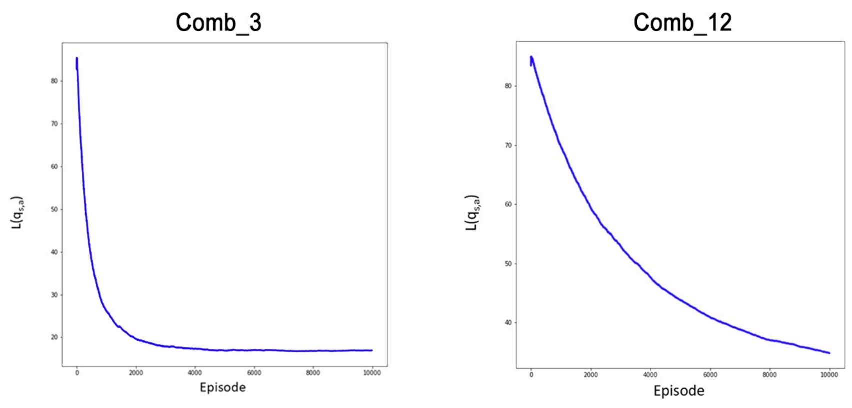


Figure 3.16: Comparison between Combination 3 and 12.

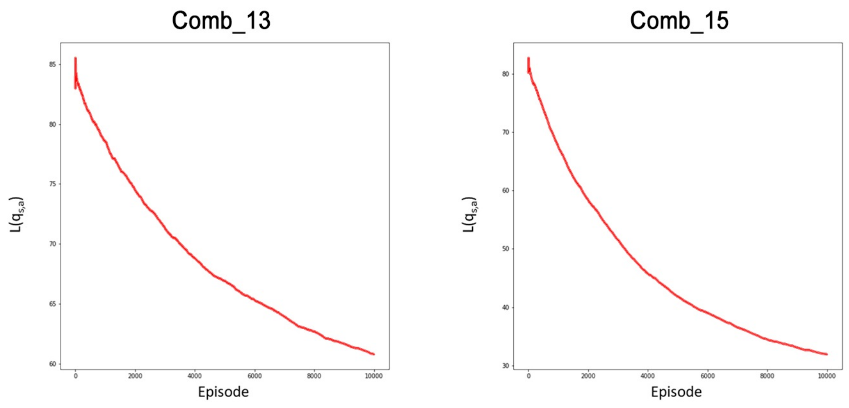


Figure 3.17: Comparison between Combination 3 and 12.

approximation with the Bellman equation Figure 3.17.

Finally, comparing the simulations 3, 6, 9, 12, 15, 18 we can see how the influence of the reward decay parameter, linked to the importance of future rewards, acts both on the inclination of the curve and on the precision of the algorithm, measurable as said by the lowest value of the $L(q_{sa})$ function Figure 3.18.

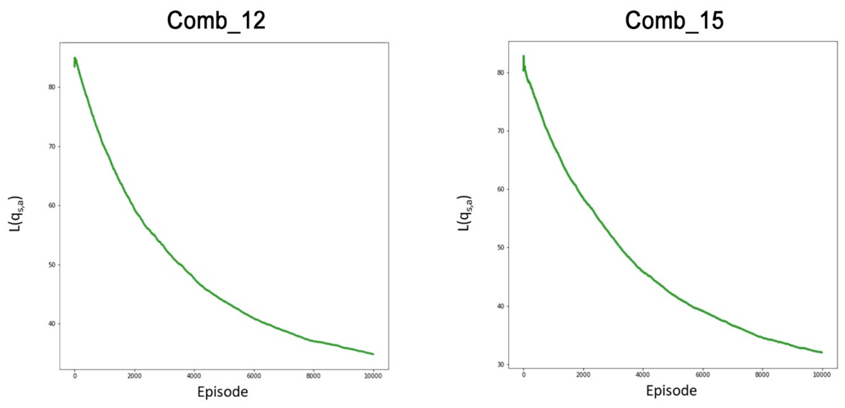


Figure 3.18: $L(q_{sa})$ results in combinations 10-18.

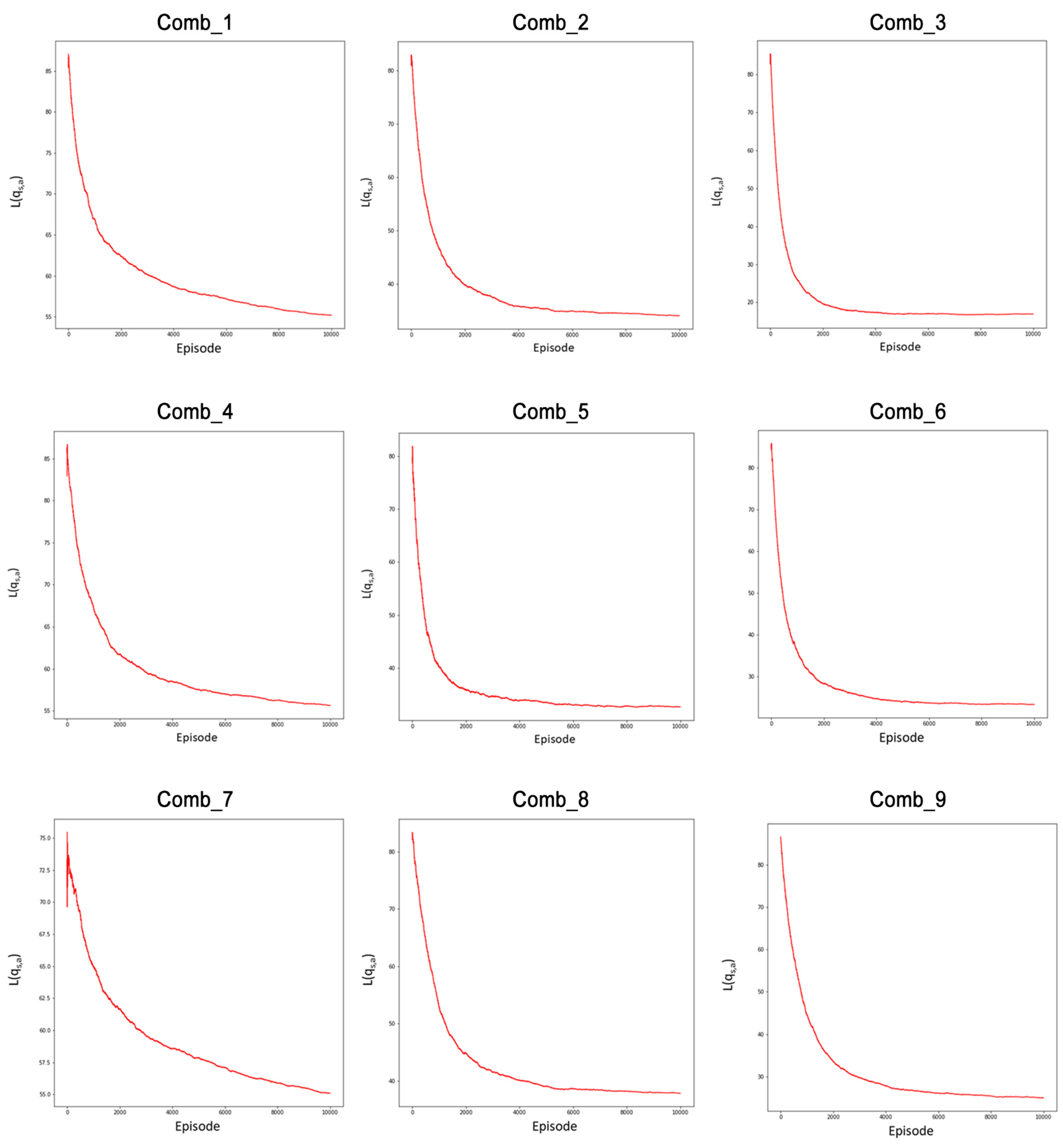


Figure 3.19: $L(q_{sa})$ results in combinations 1-9.

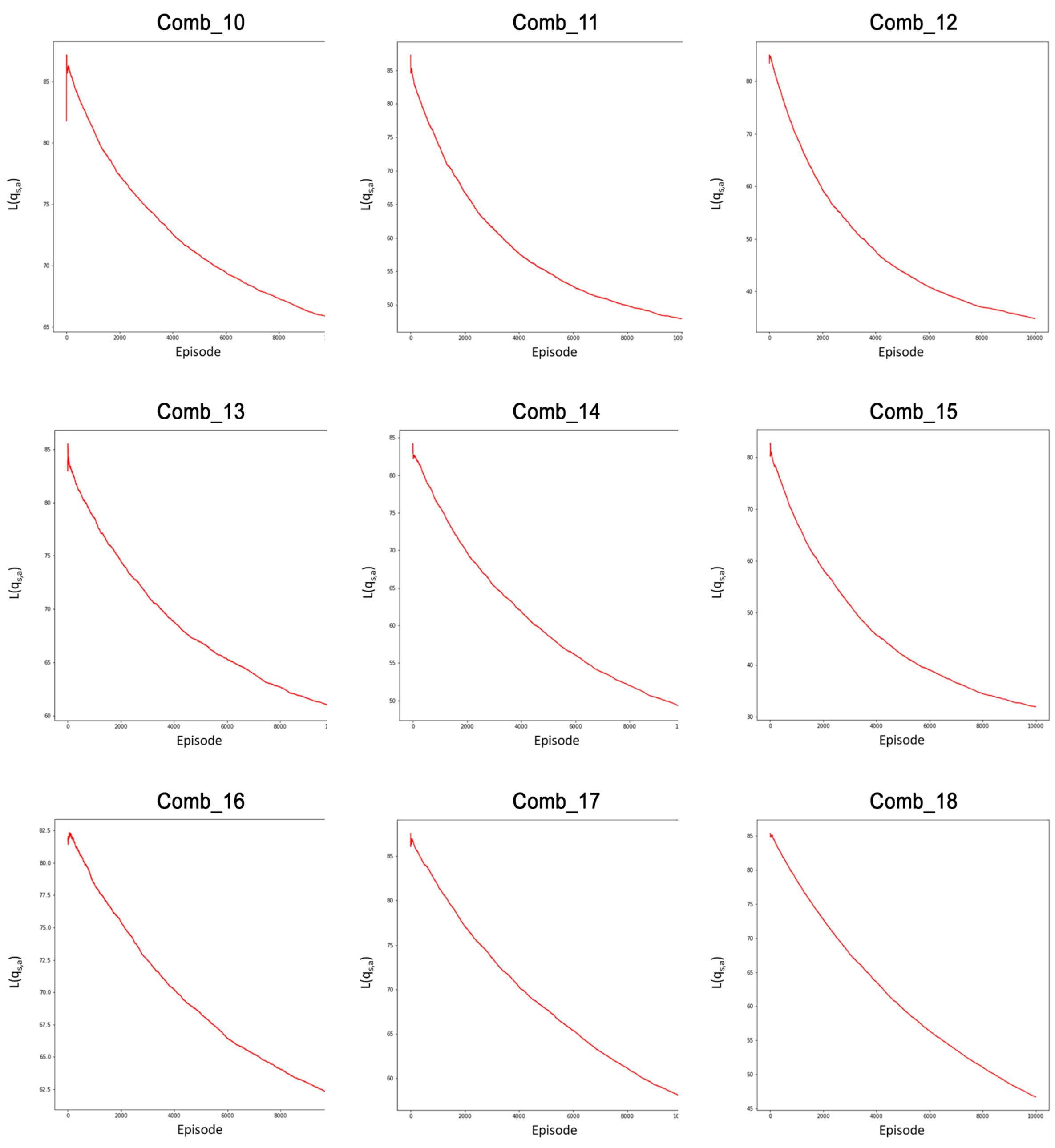


Figure 3.20: $L(q_{sa})$ results in combinations 10-18.

Chapter 4

Results

As explained, the objective of this thesis is to be able to reconstruct an operational alignment of the tunnel so that it deviates as little as possible from the theoretical alignment that has been designed. Subsequently, using this data, place the rings that make up the structure of the tunnel with the correct rotations, to guarantee its correctness.

4.1 Data collection

To obtain a Q-table able to represent exactly which are the best positions that the rings must assume along the alignment, simulations are performed on an alignment with a length of about 400m.

For these simulations the parameters of combination 15 described in 3.4.1 have been adopted:

- discount factor *gamma*=0.85;
- learning rate *lambda*=0.001;
- *epsilon* Greedy=0.95.

To obtain the desired results, it was necessary to carry out 1,800,000 simulations. The large number of simulations required is due to the high state

space dimension nature, which is difficult to manage by a method based on simple Q-learning.
To speed up the construction process of the Q-table it was decided to divide the simulations into 6 parts of 300'000 simulations each.
After 300,000 simulations, the algorithm can exactly place about 40 rings, so the approach adopted for the placement of the 200 rings is sequential.
After having carried out the first part of the simulations, the Q-table will contain the necessary information to be able to place the first 40 rings with a maximum deviation from the theoretical path of 1 cm, therefore the following simulations can be made starting from ring 40 instead of ring 1 and will enrich the Q-table with the new information of the following rings.

Although this sequential approach is not optimal for the Q-table, it was decided to use it because, given the high dimensional nature of the state space, it would be too expensive to solve the problem classically.
At the end of the simulations, the Q-table has a size of 27 columns, equal to the number of available actions, and 162'544 rows, equal to the number of states that have been taken into account by the algorithm.
Below are the graphs related to the placement of the rings in the simulations.

As you can see, as the number of simulations increases, the trend of the rings that are placed increases.

4.2 TIM modelling

From the Q-table it is, therefore, possible to extract the positions of the rings in terms of coordinates (state) and rotations (action).
With the help of the dynamo visual programming plugin, it is possible to reconstruct the model of the tunnel lining.

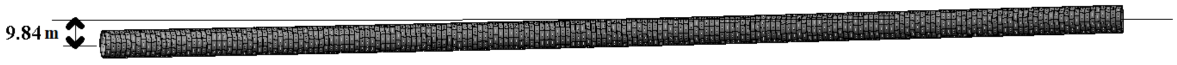


Figure 4.1: Alignment altimetric trend.

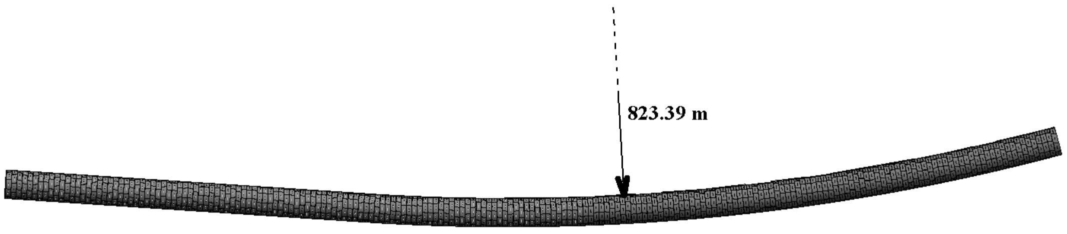


Figure 4.2: planimetric course of the tunell.

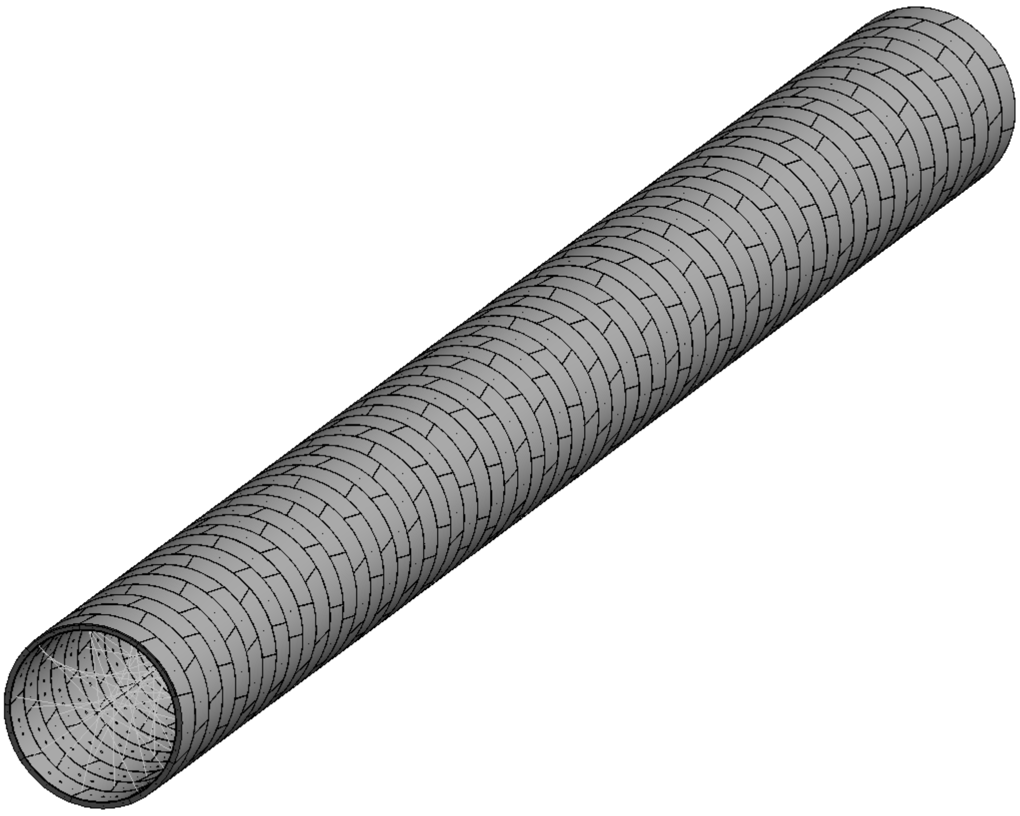


Figure 4.3: Detail showing 60 rings that make up the tunnel.

Chapter 5

Conclusion

In conclusion, it can be said that the algorithm adopted was able to reconstruct a aligment composed of 200 rings with a maximum deviation from the theoretical alignment of 1 cm.
The construction of this alignment, however, was expensive from a computational point of view because the number of states to explore to obtain the result is difficult to manage through a simple Q-learning method.

To obtain better results from the computational speed point of view it is necessary, as explained in 3.4.1, to adopt algorithms using Artificial Neural Network (ANN). Such a development, however, requires strong expertise in Data Science.

It can also be said that thanks to the ease of extraction of the data necessary for the algorithm to operate, i.e. coordinates of the theoretical alignment and geometry of the ring that is used, the method developed and easily replicable.

Bibliography

[1] National Institute of BUILDING SCIENCES(2007). *NATIONAL BUILDING INFORMATION MODELING STANDARD.*
Publication: Version 1: - Part1: Overview,Principles, and Methodologies
Publisher: buildingSMARTalliance - National BIM Standard

[2] Jelena Ninić and Christian Koch(2017). *Parametric multi-level tunnel modelling for design support and numerical analysis.*
Publication: EURO:TUN 2017, Innsbruck University, Austria
Publisher: ResearchGate

[3] Vito Getuli, Pietro Capone, Alessandro Bruttini, Farzad Pour Rahimian(2020). *On-demand generation of as-built infrastructure information models for mechanised Tunnelling from TBM data: A computational design approach.*
Publication: Automation in Construction Vol 121
Publisher: Elsevier

[4] Tom Mitchell(1997). *Machine Learning.*
Publisher: McGraw Hill

[5] Thomas Marcher, Georg H. Erharter, Manuel Winkler(2020). *Machine Learning in tunnelling – Capabilities and challenges.*
Publication: Geomechanics and Tunnelling Vol 13
Publisher: Wiley

[6] Saleh Seyedzadeh, Farzad Pour Rahimian, Stephen Oliver, Sergio Rodriguez, Ivan Glesk(2020). *Machine learning modelling for predicting non-domestic buildings energy performance: A model to support deep energy retrofit decision-making.*
Publication: Applied Energy Vol 279
Publisher: Elsevier

[7] Mirsalar Kamari, Youngjib Ham(2020). *Vision-based volumetric measurements via deep learning-based point cloud segmentation for material management in jobsites.*
Publication: Automation in Construction Vol 121
Publisher: Elsevier

[8] Ruichuan Zhang, Nora El-Gohary(2019). *A Machine-Learning Approach for Semantic Matching of Building Codes and Building Information Models (BIMs) for Supporting Automated Code Checking.*
Publication: Recent Research in Sustainable Structures
Publisher: Springer

[9] J.J. McArthur, Nima Shahbazi, Ricky Fok, Christopher Raghubar, Brandon Bortoluzzi, Aijun An(2015). *Machine learning and BIM visualization for maintenance issue classification and enhanced data collection.*
Publication: Advanced Engineering Informatics
Publisher: Elsevier

[10] Farzad Pour Rahimiana, Saleh Seyedzadeh, Stephen Oliver, Sergio Rodriguez, Nashwan Dawood(2020). *On-demand monitoring of construction projects through a game-like hybrid application of BIM and machine learning.*
Publication: Automation in Construction Vol 110
Publisher: Elsevier

[11] Ying Hong, Ahmed W.A. Hammadb, Ali Akbarnezhad, Mehrdad Arashpour(2020). *A neural network approach to predict-*

ing the net costs associated with BIM adoption.
Publication: Automation in Construction Vol 119
Publisher: Elsevier

[12] Qianli Zhang, Weifei Hu, Zhenyu Liu, Jianrong Tan (2020)Paris Buttfield-Addison(2017). *Tunnelling and Underground Space Technology.*
Publication: Tunnelling and Underground Space Technology
Publisher: Elsevier

[13] Ulrike Pelz, Engineering Design Director, Lendlease, NorthConnex Project(2019). *Tunnel Construction Overview.*
Publisher: Australia Tunnelling Socety

[14] Christian Koch, Andre Vonthron(2017). *A tunnel information modelling framework to support management, simulations and visualisations in mechanised tunnelling projects.*
Publication: Automation in Construction
Publisher: ResearchGate

[15] Abhishek Nandy, Manisha Biswas, Avirup Basu(2018). *Reinforcement Learning.*
Publication: Book
Publisher: Apress

[16] Puterman, Martin L(2014). *Markov decision processes: discrete stochastic dynamic programming.*
Publisher: John Wiley & Sons

[17] Wang, Ziyu and Schaul, Tom and Hessel, Matteo and Hasselt, Hado and Lanctot, Marc and Freitas, Nando(2016). *Dueling network architectures for deep reinforcement learning.*
Publisher: International conference on machine learning

[18] Laureiro-Martínez, Daniella and Brusoni, Stefano and Canessa, Nicola and Zollo, Maurizio(2015). *Understanding the*

exploration–exploitation dilemma: An fMRI study of attention control and decision-making performance.
Publication: Strategic Management Journal Vol 36
Publisher: Wiley Online Library

[19] Ruder, Sebastian(2016). *An overview of gradient descent optimization algorithms.*
Publisher: arXiv preprint arXiv:1609.04747

[20] Stadie, Bradly C and Levine, Sergey and Abbeel, Pieter(2015). *Incentivizing exploration in reinforcement learning with deep predictive models.*
Publication: arXiv preprint arXiv:1507.00814

[21] Tanya Bloch, Rafael Sacks(2020). *Comparing machine learning and rule-based inferencing for semantic enrichment of BIM models.*
Publication: Automation in Construction Vol 91
Publisher: Elsevier

[22] Saleh Seyedzadeha, Farzad Pour Rahimian, Parag Rastogi, Ivan Glesk(2019). *Tuning machine learning models for prediction of building energy loads.*
Publication: Sustainable Cities and Society Vol 47
Publisher: Elsevier

[23] Somin Park, Francis Baek, Jiu Sohn, Hyoungkwan Kim(2019). *Deep Learning-based Vehicle Image Matching for Flooding Damage Estimation.*
Conference: Creative Construction Conference

[24] Wolfgang Eber(2019). *Artificial Intelligence in Construction Management – a Perspective.*
Conference: Creative Construction Conference

[25] Qianli Zhang, Weifei Hu, Zhenyu Liu, Jianrong Tan(2020). *TBM performance prediction with Bayesian optimization and automated machine learning.*
Publication: Tunnelling and Underground Space Technology Vol 103
Publisher: Elsevier

[26] Pin Zhang, Heng Li, Q.P. Ha, Zhen-Yu Yin, Ren-Peng Chend(2020). *Reinforcement learning based optimizer for improvement of predicting tunneling-induced ground responses.*
Publication: Advanced Engineering Informatics Vol 45
Publisher: Elsevier

[27] Pin Zhang, Huai-Na Wu, Ren-Peng Chen, Tommy H.T. Chan(2020). *Hybrid meta-heuristic and machine learning algorithms for tunnelinginduced settlement prediction: A comparative study.*
Publication: Tunnelling and Underground Space Technology Vol 99
Publisher: Elsevier

[28] ZongWei Yao, Qiuping Huang, Ze Ji, XueFei Li, Qiushi Bi(2020). *Deep learning-based prediction of piled-up status and payload distributionof bulk material.*
Publication: Automation in Construction Vol 121
Publisher: Elsevier

[29] Sivalingam Koneshwaran, David P. Thambiratnam, Chaminda Gallage(2015). *Response of segmented bored transit tunnels to surface blast.*
Publication: Advances in Engineering Software
Publisher: Elsevier

Acknowledgements

Ringrazio la mia famiglia per avermi supportato e insegnato a credere nelle mie capacitá, in particolare mio fratello per avermi trasmesso la passione per la ricerca e per il supporto che mi ha dato durante questo lavoro di tesi.
Ringrazio il Politecnico, il DISEG, la professoressa Osello, Arianna Fonsati e tutti i professori che ho avuto il piacere di incontrare in questo percorso formativo per le conoscenze che mi hanno trasmesso.
Ringrazio Sara i miei compagni di universitá per aver condiviso con me il periodo universitario, tra divertimenti e studio.
Ringrazio Andrea, Martina e Elisa per essere stati sempre presenti e disponibili ad aiutarmi.