



TECHNISCHE
UNIVERSITÄT
WIEN

MASTER-/DIPLOMARBEIT

Application of the humanoid robot Pepper as a mobile interactive architectural element: a study and a handbook for architectural designers and planners

ausgeführt zum Zwecke der Erlangung
des akademischen Grades eines
Diplom-Ingenieurs / Diplom-Ingenieurin

unter der Leitung von

Oliver Schürer

Senior Scientist Dipl.-Ing. Dr.techn.

E259 - Institut für Architekturwissenschaften
eingereicht an der **Technischen Universität Wien**
Fakultät für Architektur und Raumplanung

Vesela Danielova Petrova

Matr. Nr. 01128202

Wien, am 08.01.2017

eigenhändige Unterschrift

Acknowledgments

This work would not have been possible without the mentorship of Senior Scientist Dipl.-Ing. Dr. techn. Oliver Schürer, who guided me and helped me throughout the entire process.

I would also like to thank the staff of the Department for Architecture Theory and Philosophy of Technics of the Vienna University of Technology

Last but not least, I am grateful to my family and friends for their support, patience and love.

CONTENT

Abstract	1
Abstrakt	3
1. Scope and Objectives	5
1.1. Objectives and Tasks	5
1.2. Questions answered within this Diploma Thesis.....	7
2. Architecture: a Notion reflected in Software and Robotics.	8
2.1. Physical and Software Architecture	8
2.1.1. Origin and Evolution of the Term ‘Architecture’	8
2.1.2. What is ‘Software Architecture’?	13
2.1.3. Comparison between Physical and Software Architecture – common Features and Differences.	14
2.2. Robotics as the connecting Link in the Architectural Cycle.....	15
2.2.1. Robotics: a brief State – of – the – Art.....	15
2.2.2. Humanoid Robots as a suitable Choice for an Architectural Space.	17
3. Robots in continuous Interaction with Humans.....	22
3.1. Snackbot and Military Robots.	22
3.2. Robots as an integral Part of Architectural Space	25
3.3. Pepper integrated in a public Space	26
4. Architectural Space	30
5. Humanoid Robot Pepper vs Humans’ Navigating in Architectural Space.....	32

5.1. Pepper perception vs humans' perception of the surrounding environment	32
5.2. The Parallel between Pepper Perception of Space and a Nolli Map.	36
5.3. Pepper Implementation as an Office Concierge in the Department for Architecture Theory and Philosophy of Technics	38
6. Programing Pepper to be utilized as an Office Concierge in the Department for Architecture Theory and Philosophy of Technics.	44
6.1. Different Approaches on programing Pepper Speech Recognition.	44
6.2. Different Approaches on how to program the Conversational Capabilities of Pepper	48
6.3. Brief Manual on programing Pepper behavior for designers and planners in architecture	56
7. Programing Pepper Navigation.....	80
7.1. Exploration	80
7.2. Places and Patrol.....	84
7.3. How to Install an Application on a Physical Robot using Choreographe	85
8. Conclusions and Perspective for Future Work.....	92
8.1. Summary of Work Results and Achievements	92
8.2. Issues to be explored in Future.....	93
8.3. Conclusion	94
REFERENCES.....	95
TABLE OF FIGURES.....	99

TABLE OF TABLES.....	101
TABLE OF QR CODES.....	103
CURRICULUM VITAE.....	105

.

Abstract

This thesis convincingly proves the potential of adopting a robot as an architectural element, exploiting some of the most common scenarios that can occur. By discussing challenges, obstacles and viable solutions it results in guidelines for architectural designers and planners who intend to use a humanoid robot Pepper in their designs.

The work investigates and results in a design instigating and aiding the human – humanoid robot interaction in an office environment. The space utilization is implemented through specifically designed robot behaviors appropriate for the intended architecture function and representation.

The developed design is based on a concept of space as an expression of interrelated actions among artifacts and humans. It is realized without physically changing the existing built space. Instead it utilizes a humanoid robot as a mobile interactive element of office architecture. As such the robot also takes over several common concierge tasks. Thus it allows for higher utilization of the designed space through designing the interaction in it, while ensuring that qualified staff can concentrate and devote its efforts to more creativity demanding tasks, which will result in an increase of overall productivity in short-term prospective.

Additionally, several socio-psychological effects of long-lasting communication between people and robots in different type of working environment are addressed.

Through the assigned position and tasks of an office concierge, the humanoid robot serves as a conduit between the architectural environment and its users. In our case the synergy is achieved through programming a humanoid robot Pepper, of Aldebaran/SoftBank. The newly developed software architecture targets designing

different robot behaviors for routinely occurring scenarios in the office environment of the Department for Architecture Theory and Philosophy of Technics.

The presented results are systematized, analyzed and communicated in the form of a handbook for architectural designers and planners. It gives a basic overview of several relevant issues of robot behavior design. The adopted approaches comply with the expertise, experience and expectations of the target user group. Additionally, other optional more complex solutions are pointed out. Despite being mentioned, these are beyond the scope of this work as they require advanced background in robotic and programming rather than in architecture and design.

Abstrakt

Diese Masterarbeit beweist überzeugend das Potenzial, einen Roboter als architektonisches Element zu verwenden und untersucht einige der häufigsten Szenarien, welche möglich sind. Durch die Diskussion von Herausforderungen, Hindernissen und realisierbaren Lösungen ergeben sich Richtlinien für die Architekten und Planer, die beabsichtigen in ihren Entwürfen einen humanoiden Roboter Pepper zu einzubinden.

Die Masterarbeit ergibt ein Design, welches die Interaktion zwischen Mensch und humanoidem Roboter in einem Büroraum positiv anregt und unterstützt. Die Raumnutzung wird durch speziell entworfenes Roboterverhalten beeinflusst und hat auf die beabsichtigte Architekturfunktion, die Interaktion, einen positiven Einfluss.

Das entwickelte Design basiert auf einem Konzept von Raum, als Ausdruck aufeinander bezogener Handlungen zwischen Artefakten und Menschen. Es wird realisiert, ohne den vorhandenen Bauraum physisch zu verändern. Stattdessen wird ein humanoider Roboter als mobiles und interaktives Element der Büroarchitektur genutzt. Als solcher übernimmt der Roboter auch einige gewöhnliche Concierge-Aufgaben. Das ermöglicht eine bessere Raumnutzung durch die Gestaltung der Interaktion. Dieses Zusammenspiel sichert einerseits, dass sich qualifiziertes Personal auf kreativere Aufgaben den Fokus setzen kann, was andererseits zu einer Erhöhung der Gesamtproduktivität in kurzfristiger Perspektiven führt.

Darüber hinaus werden verschiedene sozio-psychologische Effekte einer lang andauernden Kommunikation zwischen Menschen und Robotern in unterschiedlichen Arbeitsumgebungen behandelt.

Durch die zugewiesenen Positionen und Aufgaben eines Büro-Concierge dient der humanoide Roboter als Verbindungsglied zwischen dem architektonischen Raum und seinen Nutzern. In unserem konkreten Forschungsfall wird die Synergie durch Programmierung eines humanoiden Roboters Pepper (entwickelt von Aldebaran / SoftBank) erreicht. Das Ziel der neu entwickelten Softwarearchitektur ist es, unterschiedliches Roboterverhalten für routinierte Szenarien in einem Büroraum zu entwerfen. Konkret soll dies in der Abteilung für Architekturtheorie und Technikphilosophie (der Technischen Universität Wien) stattfinden.

Die präsentierten Ergebnisse werden in Form eines Handbuchs für Architekten und Planner systematisiert, analysiert und kommuniziert. Es soll einen grundlegenden Überblick über einige relevante Aspekte des Roboterhaltens geben. Die verabschiedeten Ansätze entsprechen dem Fachwissen, den Erfahrungen und den Erwartungen der Zielgruppe. Zusätzlich werden andere komplexere Lösungen aufgezeigt. Diese werden im Rahmen dieser Arbeit jedoch nur erwähnt, da sie einen fortgeschrittenen Hintergrund in Robotik und Programmierung und nicht in Architektur und Design erfordern.

.

1. Scope and Objectives

1.1. Objectives and Tasks

This work investigates, analyses, systematizes, summarizes and produces robot behaviors that show how humanoid robots can fit in an environment designed for humans and what functions those machines can take over by means of interaction and mobility. It analyzes what functions a humanoid robot fulfills effectively in an office environment in order to improve the space functionality and proves its applicability in solving concrete tasks. The work studies, systematizes and summarizes the relation between physical architecture, software architecture, humanoid robots and humans (see Fig. 1. 1)

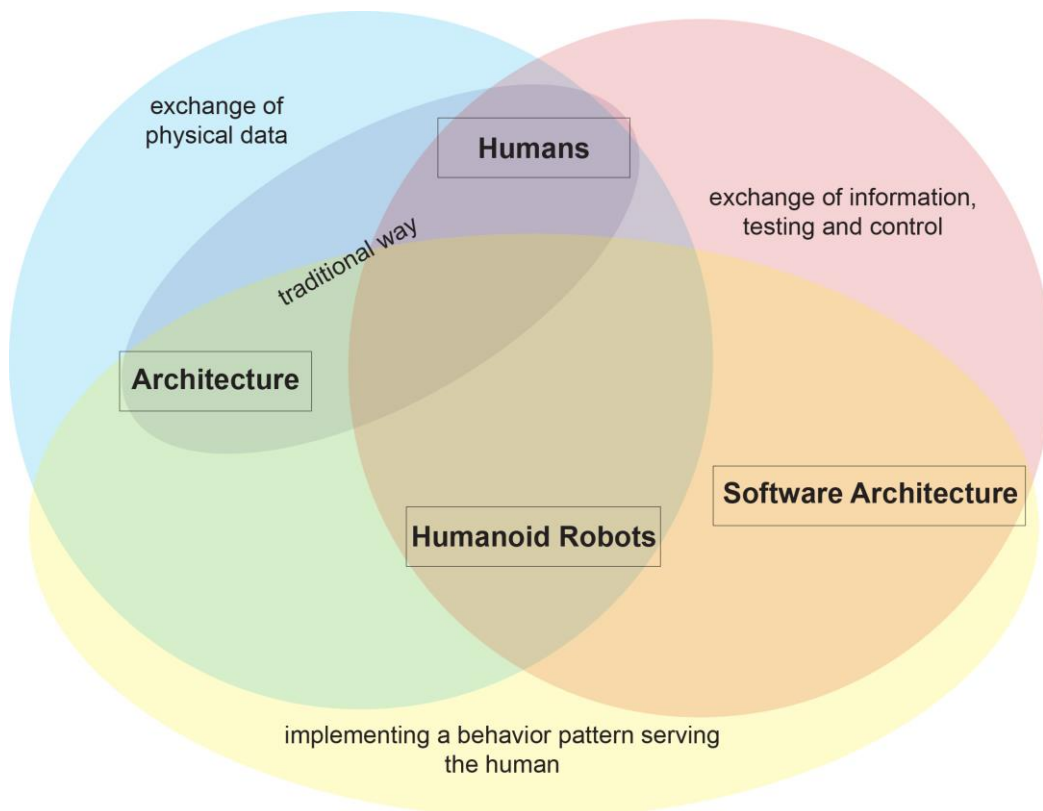


Fig. 1. 1 Interaction between tasks and involved players

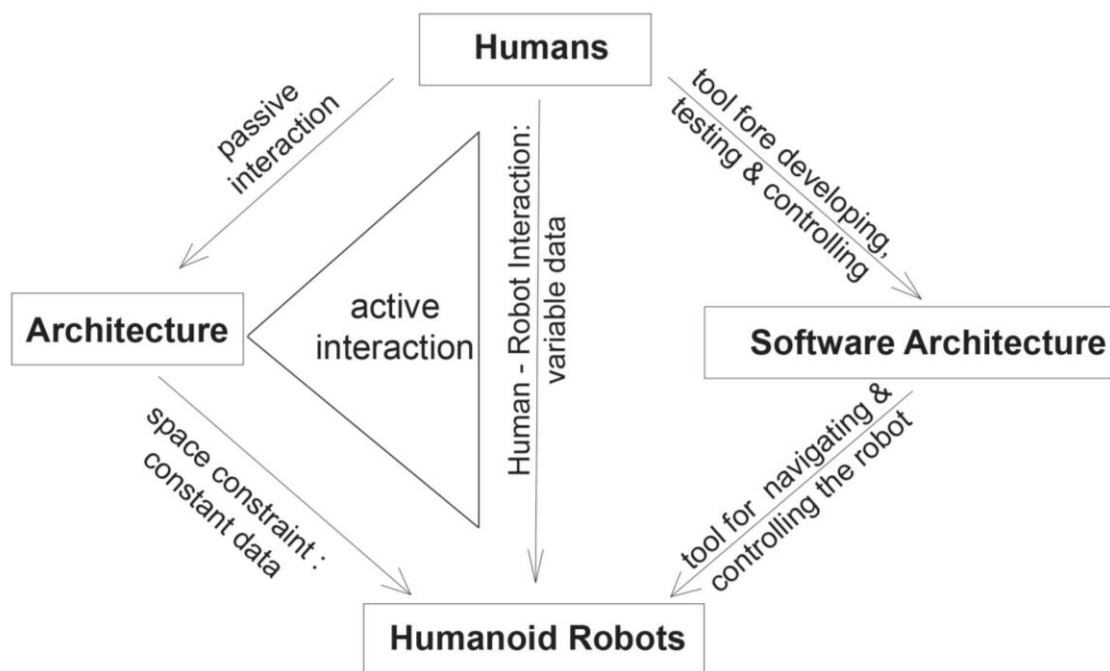


Fig. 1. 2. Interaction between Architecture, Software Architecture, Robots and Humans

The tools that maintain this relation and therefore allow for interaction between architecture, software architecture, robots and humans, as they are considered in this work, are shown in Fig. 1. 2.

Reflecting this interaction a behavior pattern for a humanoid robot Pepper serving as an office concierge in the context of a particular office environment is produced. An overview of the relevant programming tools and approaches is delivered through an explanation of the design process. An array of used tools and approaches, complemented by a comprehensive explanation of their application, is discussed. The most significant tools used for the robot behavior programming are speech, speech recognition, navigation and data communication via the onboard tablet. Most of the results, relevant to the described process, form the

basics of a handbook in Pepper programming for architectural designers and planners.

1.2. Questions answered within this Diploma Thesis

The presented diploma thesis profoundly studies some aspects of the communication between humans and the humanoid robot Pepper via answering the following questions:

1. Comparison between physical and software architecture. Can the robots be the connecting link between both types of architecture?
2. The robot influence on its interlocutors and their perception of the office space. How can we integrate and utilize a humanoid robot in an office space? What are the specificities of the interaction between different people and the robot? What are the impacts?
3. How is the understanding of the term 'space' evaluated?
4. Comparison between Pepper and humans navigating in space. How can the humanoid robot Pepper be efficiently implemented as an office concierge in the Department for Architecture Theory and Philosophy of Technics?

The work results in the development of guidelines for programming Pepper by non IT engineers/architects, who want to use Pepper efficiently in their works.

Thus, this work successfully answers the core question: How can the theoretical findings and outcomes be implemented in a technically sound project, so that people can benefit from the co-existence with the robot?

2. Architecture: a Notion reflected in Software and Robotics.

2.1. Physical and Software Architecture

2.1.1. Origin and Evolution of the Term 'Architecture'.

Architecture is a concept which meaning depends on the context of its use. In order to determine the most common of its meanings we can look up the topic 'Architecture' on one of the most popular websites worldwide - Wikipedia. The referred article provides information about the etymology of the word and its basic use (Wikipedia, 2017).

The term 'architecture' unites both, the process and the product of planning, designing, and constructing buildings and other physical structures. Its origin leads to:

- Latin - architectura,
- Greek - ἀρχιτέκτων arkhitekton 'architect', from ἀρχι- 'chief' and τέκτων 'builder'.

The given below pictures (Fig. 2. 1, Fig. 2. 2 and Fig. 2. 3) show the different steps in the development of a physical architectural object. In this particular case, this is the Guggenheim Museum in New York City by Frank Lloyd Wright.

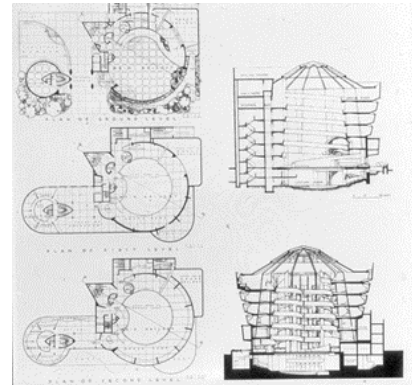


Fig. 2. 1 Architecture Guggenheim Museum, New York City, F. L. Wright:
Conceptual idea and building design (Breitling, 2003)



Fig. 2. 2 Architecture Guggenheim Museum, New York City, F. L. Wright:
Construction process (Gottscho-Schleisner, 12 November 1957)



Fig. 2. 3 Architecture Guggenheim Museum, New York City, F. L. Wright
(Onniboni, 2016)

While discussing the meaning of the term 'architecture' in the context of modern society we should differentiate between architecture in its original meaning, business architecture, cognitive architecture, computer architecture, enterprise architecture, interior architecture, landscape architecture, naval architecture, software architecture, system architecture, etc..

In the context of this thesis, architecture is considered in two of its meanings:

- its original meaning, as the art and craft of designing, planning and realizing cultural symbols by means of buildings, that will be referred to as traditional, physical or simply as architecture and
- software/non-physical architecture.

Both disciplines revolve around serving humans and making our life easier.

Traditional architecture and its designers are concerned with creating an appropriate, functional and aesthetically pleasing built environment. Software architecture and its developers focus on creating algorithms and algorithmic structures that, for example, can relieve people of time consuming tasks such as demanding calculations. Alongside this, both types of architecture also strive to deliver a functional and easy to understand and get used to visual design, a design that is of maximum use to the user because of its simplicity.

The following section delivers a comparison between physical and software architecture with respect to their speed of development and several inspirations regarding the final outlook of the resulting designs.

Physical architecture is one of the classical fields of human art and society. As such it has a long history and is founded on traditions and principles. Some of these principles are more than 2000 years old, such as Vitruvius' firmitas, utilitas,

venustas (see Fig. 2. 4.), (Vitruvius, n.d.). The basic principles have remained virtually the same throughout the centuries. They have only been adapted to the specific needs and beliefs of the era.

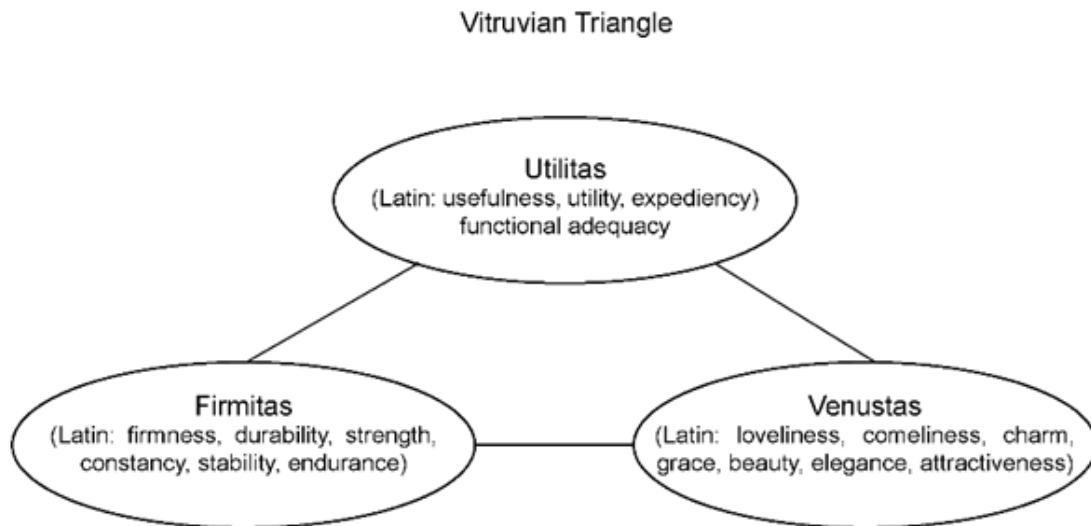


Fig. 2. 4 Vitruvius' principles of architecture

In recent years architectural design and its designing process have been greatly impacted by technology and its rapid development. CAD programs have become architectural designers' and planers' favored tool that allows them to design quicker, more precisely, more cost-effectively and sustainably. New building technologies have been developed and incorporated in architecture in order to improve its functionality and quality. The results reflect the architects' view of the world and aim to incorporate the aesthetical and functional needs architecture must satisfy into a comprehensive and exiting design.

In his article 'Six Themes for the next Millennium' (Pallasmaa, 1994) the Finnish architect Juhani Pallasmaa critiques on six features he considers essential for the architecture of the present millennium: *slowness, plasticity, sensuousness, authenticity* and *idealization*. He opposes the slowness and calmness the

architectural environment brings into the daily life to the stressful and hectic pace of our reality. In the same article Pallasmaa claims that 'Great architecture petrifies time' but more than that 'architectural works are museums of time and they also have the capacity of suspending time' (see Fig. 2. 5).



Fig. 2. 5 San Gimignano - a city built accretive over time accumulating a sense of continuity (Donati, 2016)

The importance of slowness is further stressed by Milan Kundera who relates it to memory in his novel 'Slowness' (Kundera, 1997). He writes that 'The degree of slowness is directionally proportional to the intensity of memory. The degree of speed is directionally proportional to the intensity of forgetting'. Both Pallasmaa's and Kundera's statements relate to architecture quality to serve as a source of preserved cultural and social history and the importance of slowness in that respect. Additionally, architecture and its design trends change in a pace far slower than that of software architecture because architecture and its building require a considerable amount of money and resources be invested.

2.1.2. What is 'Software Architecture'?

Software architecture (see Fig. 2. 6) can be seen and experienced in a different way than traditional architecture. It is defined and realized through its code and can be experienced through its visualized design.

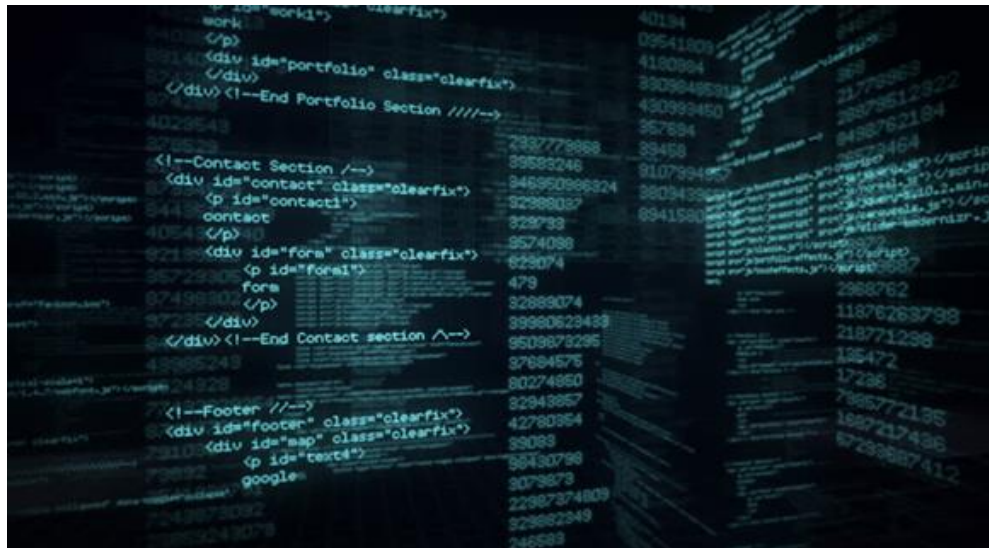


Fig. 2. 6 Artistic presentation space built by code. (studios, n.d.)

Because of this, the goals of its designers have a different focus. Software developers are concerned with creating interactive designs that use common concepts from the physical world. In terms of this work interaction design is considered from the perspective given in the book 'Interaction Design: beyond human - computer interaction' (Reogers, 2011). In it interaction design is said to be 'about creating user experiences than enhance and augment the way people work, communicate and interact'. Additionally, the same book describes Wingrad's comprehension on software architecture as 'designing spaces for human communication and interaction'. Since interaction design focuses on daily life and day to day activities, it is not surprising that its designers draws inspiration from the surrounding them architectural environment. Software architects aim for a user – friendly and subconscious semantic understanding of the software functions and

use. Therefore, they reflect the physical environment in their works. For example the 'Recycle Bin' in Windows OS serves the same purpose in the digital environment as it does in the physical one.

2.1.3. Comparison between Physical and Software

Architecture – common Features and Differences.

To sum it up both physical architecture and software architecture are inspired by the world. Both fields strive to engage the user. Traditional architecture delivers functional and aesthetic designs based on rationalizing nature, its processes or mathematical principles. Software architecture utilizes common terms and concepts that result in a design, which is easy to understand and interact with for the end user.

Another difference between the two types of architecture lies in their nature, i.e. physical architecture is slowly/passively interactive, immobile and with limited flexibility, while software designs are more rapidly/actively interactive, flexible and adaptable. Despite both architectures being impacted by one another, they don't actively interact. It is always a case of one serving the other in a certain capacity without a direct backward relation that leads to a closed loop.

Breaching the gap between the two results in a cohesive relationship where architecture goes from slowly and passively interactive (through building orientation based on natural elements like the sun and the wind) to rapidly and actively interactive (through immediate interaction response upon pushing a button or voice command). This can be achieved through the addition of interactive mobile architectural elements such as doors, windows, ventilation system etc. Software

architecture will, in turn, have a new role in this context - to power the architectural elements and thus to guarantee a fine control, implemented by hardware elements such as motors, sensors, controls etc.

The presented project utilizes a humanoid robot as an element embodying software architecture. Its interactive nature is the basis for the human-humanoid robot interaction that will actively engage the users of the design in a different way than so far.

2.2. Robotics as the connecting Link in the Architectural Cycle

2.2.1. Robotics: a brief State – of – the – Art

Further, several issues of robotics are discussed, its state – of – the – art and the precise reason how and why a robot can serve as a mobile interactive architectural element and a conduit between physical and software architecture is explained.

In order to determine that an overview of our present is needed. We live in the boom of the Digital Revolution that has started in the 1980s. It has since then greatly changed the way we perceive and understand the world. Now it is at the point of evolving further.

This is a prerequisite for traditional architecture to be taken outside its present boundaries and to be turned into more than a functional aesthetic object. Architects are now able to realize designs that were impossible to build in the past. This is due to the development of new technologies, the introduction of new materials and the greater machine power. An impressive example of their impact on architecture

is La Sagrada Familia which is still 'a constructional challenge' and 'one of the largest testing grounds for construction methods in the world' with its revolutionary design, despite having been under construction for almost 150 years (Família, n.d.).

A next stage in architecture evolution is the incorporation of cutting-edge technologies that enable it to adapt to the changing users' needs and make it actively interactive. This kind of interaction has an immediate and direct result while being under the arbitrary control of the user. The new trend reflects the marriage between physical and software architecture. It also ensures the improved functionality of the resulting designs through their interactivity and adaptability by reflecting the latest technological developments. Robotics is a technology, allowing architects to add mobile architectural elements to the built environment.

Consequently, this trend brings up the question of robots suitability for these tasks. Robotics is a field strongly impacted by science fiction and pop-culture. As David Hanson says 'Robots can be useful in many shapes and forms, and the field is young - with so much room left for innovation and diversification in design' (Hanson, 2011). Shuji Hashimoto (Hashimoto, 2004) states 'Robot is one of the most suitable information terminals with multi-modal communication channel to realize a new type of 'man – machine' interface with Kansei '¹. He also argues the need for a robot to have a body and behavioral pattern similar to that of a human when it is supposed to occupy the same space and interact with humans (see Fig. 2. 7).

¹ Kansei – the word is used in Japanese for the subjective concepts sensibility, feeling, emotion or intuitiveness



Fig. 2. 7 Humanoid robot Pepper in an office environment (Robotics, 2017)

Considering this, it is very important to define the precise function and tasks which will be allocated to the robot serving as an architectural element.

Conclusion:

In terms of this work the robot serves as an office concierge taking over routine tasks in order to relive qualified staff from them.

2.2.2. Humanoid Robots as a suitable Choice for an Architectural Space.

In this section the suitability of a humanoid robot for an architectural space is discussed. I believe a humanoid robot is the option that would best fit office space

based on Hashimoto's paper (Hashimoto, 2004) defining humanoid robots as a suitable choice for sharing a residential space with humans. While both, residential and office, spaces are functionally different what unites them is the presence of humans with which a robot is to interact. The defining factor for the suitable robot design is the effectiveness of this interaction. Additionally, a humanoid robot can resemble the human not only in its shape but also in its size. Having the same or similar size ensures the humanoid robot can be programmed to tackle tasks based on established by human behavior patterns. This is practical because the built architectural environment is designed to accommodate the needs and physiology of its human users (see Fig. 2. 8).



Fig. 2. 8 Different humanoid robots in an architectural environment

Furthermore, a humanoid robot gives the user the sense of something uncanny² but also of something they can communicate with. We are social creatures and

² Uncanny valley- The concept of the uncanny valley, in HRI, regards the hypothesized relationship between the degree of an object's resemblance to a human being and the human's emotional response to such an object. In his article 'The Uncanny Valley' Masahiro Mori 'hypothesized that a person's response to a humanlike robot would abruptly shift from empathy to revulsion as it approached, but failed to attain, a lifelike appearance. This descent into eeriness is known as the uncanny valley' (Mori, 2012) (Wikipedia, 2017).

interact daily with one another. At the same time, technology importance in everyday life rises because it allows us to stay in constant contact. Because of that we use it more and more. Thus our interaction with technology increases and we become more accustomed to it. Regarding this a humanoid robot addresses both the aspect of human – human interaction and the human – technology interaction.

Another advantage of humanoid robot versus other forms of modern technology is that it can initiate interaction upon detecting the proximity and location of a possible interlocutor. It can also influence the atmosphere in a space through its designed behavior. Additionally, it can ensure the satisfaction of the user by increasing the functionality and even the aesthetic appearance of a space, thus making it more engaging.

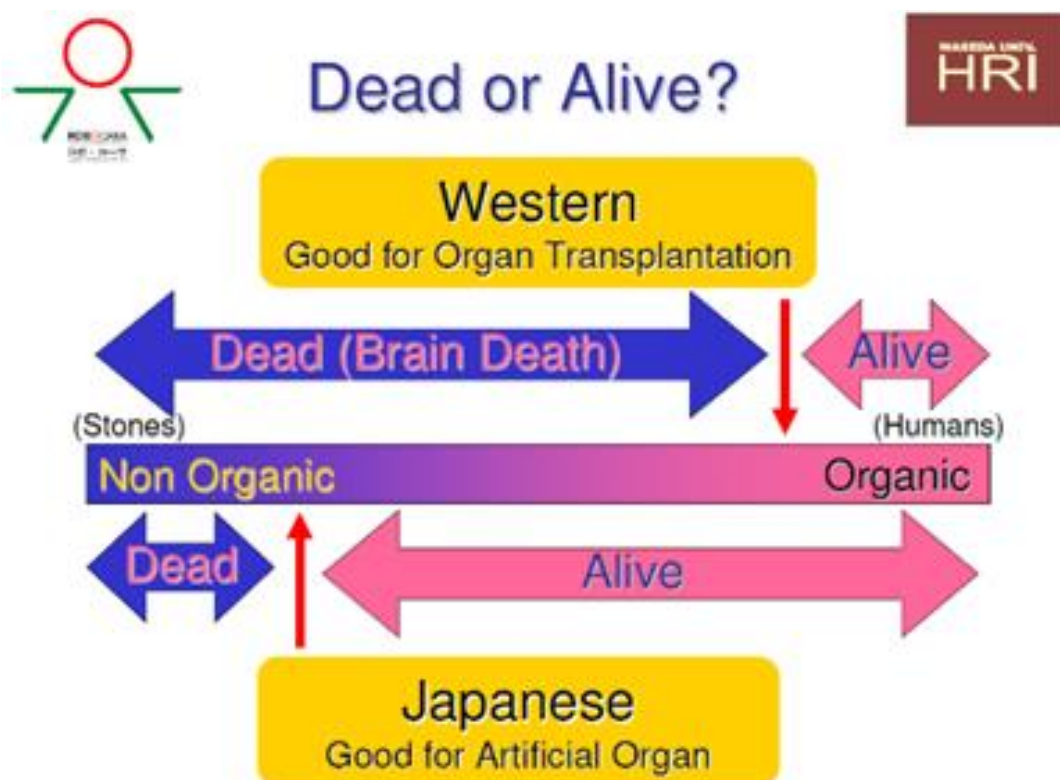


Fig. 2. 9 Cultural Differences in the Perception (Takanishi, 2005)

However, there is a big difference in how people from various cultures perceive robots, which affects how they feel about them. Unsurprisingly new technology prompts conflicting reactions. Humanoid robots are different from everything used on a commercial scale so far. This makes their implementation exciting for some, while others are rather cautious. Furthermore, there is a vast difference in what various cultures consider dead or alive; animate or inanimate (see Fig. 2. 9).

In Western culture the reanimation of the dead, respectively giving life to a soulless object, is a recurring topic that inevitably leads to the destruction of the creator. One of the many examples of such a scenario is Mary Shelley's Frankenstein (Shelley, 1818). Nowadays the renowned scientist Professor Stephen Hawking in several of his interviews predicts the negative effect of artificial intelligence (AI) and therefore of robotics. His views on the issue are extensively covered by the media which magnifies their impact on public opinion (SULLEYMAN, 2017) (Smith, 2017). A contrasting image of AI, robots and cyborgs can be found in Japanese art. An example for this is the original manga 'Ghost in the Shell' (Shirow, 1989). The Hollywood movie 'Ghost in the Shell' (Ghost in the Shell , 2017) amongst other works is based on it. This brings up the importance of pop culture in influencing how people perceive technology and robotics specifically. The Western movie industry often showcases intelligent humanoid robots and androids as a humanity's enemy in a despotic world.

In respect to this cultural obstacle, it is important for the western world to differentiate between the actual state-of-the-art and science fiction. Artificial intelligence is still in its infancy which allows the users to feel more in control. Humanoid robots are nowhere near as intelligent as their organic counterparts. At the moment, people are ready to overlook the small flaws humanoid robots might

have due to imperfect software and hardware in favor of feeling superior to them, especially when humanoid robots use is intuitive and easy. This point is investigated in the next chapter.

Conclusion:

The presented state-of-the-art convincingly proves that a humanoid robot is more acceptable than any other modern technology. The humanoid robot could be easier to handle and requires less time to master.

3. Robots in continuous Interaction with Humans.

3.1. Snackbot and Military Robots.

Continuous human – robot interaction and its impact on people is a topic of high interest to scholars from different fields, i.e. computer science, robotics, sociology, philosophy etc.

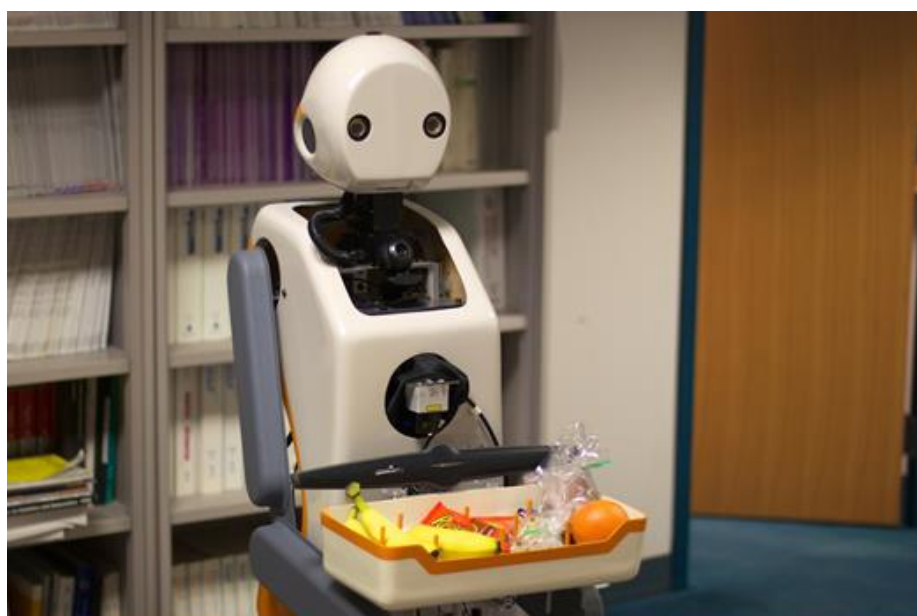


Fig. 3. 1 Snackbot

A study relevant to the topic is the introduction of the humanoid robot Snackbot (see Fig. 3. 1) to an office environment, (Jordan, 2016). While minimal interaction was foreseen and supposed to occur between the robot and the office workers, the study showed quite the opposite – a growing amount of interaction. Furthermore, communication rules were developed regarding how people were to interact with Snackbot within the relatively short period of two weeks. It was also recorded that the office workers had developed emotional attachment towards Snackbot. They had attributed to it human-like qualities such as ‘having a crush’ on some of their

co-workers, based on its speech and movement patterns. As a result the addition of Snackbot to this office environment changed the dynamic as well as the general working atmosphere.



Fig. 3. 2 Military robot: iRobot PackBot (iRobot, 2014)

Another indicative recorded human-robot interaction scenario that investigates the effect interacting with robots has on people, is the utilization of robots by military personnel on the battle field, (see Fig. 3. 2), (Jordan, 2016). In this case, the studied robots are not humanoid but ones whose design is based on the tasks they are supposed to accomplish. However, it is an indisputable how people see and feel about them. A statement on the topic says 'When robots are damaged, some troops insist that they get the same robot back – not a replacement unit'. This showed clearly the emotional attachment and sentimentality if not even camaraderie that troops developed towards robots that work alongside them. Based on this how troops felt about the units that don't even resemble them it could only be assumed how strong the impact would be if they had to work alongside humanoid robots.

Furthermore, both examples show that the acceptance of robots integrated in the human environment is only a matter of time. Once a period in which people get used to the presence of the robot passes and this is no longer a novelty – the robot becomes a part of the team. Moreover, people start seeking out suitable ways in which to interact with robots instead of expecting the robot to take the initiative exclusively. This shows that in general people are not inherently opposed to the human – robot interaction. Even though some people are rather conservative and might need longer to adapt, humans are creatures of contact and communication, they won't only accept the presence of a robot introduced to their environment but will also reach out to it themselves. This is particularly valid in the case of humanoid robots because they resemble humans, which makes them more approachable.

Aside from that a humanoid robot as an interactive architectural element and therefore part of the built environment can serve as a link between it and its users. Through specifically designed patterns of robot behaviors, the robot can be the 'voice' of the architectural environment it belongs to. Additionally, the presence of a robot, especially of a humanoid one, in a space can clearly influence how people see, understand and comprehend that space by changing the dynamic of interaction in it and provoking emotional reactions through robot's behaviors.

A reason to assume that a humanoid robot starting out as servicing the architectural space it inhabits will grow to be an integral part of it are the documented desire of the troops to receive the same unit after repairs rather than a new one and the fact that the Snackbot users were upset when the robot broke down (Jordan, 2016). A change or removal of the humanoid robot in the future would affect the users, especially the ones used to interact with it on a daily basis, the same way a physical renovation and augmentation of the space would.

3.2. Robots as an integral Part of Architectural Space

Mark Weiser claims in his essay *The Computer for the 21st Century* (Weiser, 1991) that 'Most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it.'

The Snackbot experiment gives a glimpse at the multi-layered nature and the purpose that a humanoid robot can have in an architectural space. In this case it is both a social companion and a vending machine. This makes it interesting because it means that the humanoid robot has to multitask which requires a 'smarter' and more autonomous robot. This provokes the question how many functions a robot can have and how many of them it can carry out simultaneously. It also showcases how social functions can be combined with commercial ones.

Multitasking is generally a common human quality, yet, not all humans can excel in it. Therefore, it is important to define if and to what extent a robot can and should master it. Even more interesting is to determine what kind of functions a robot can have. Following the goal to build a robot after the image of a human, future engineers can design a robot to do everything or most of the things a human can do. However, they don't have to restrict themselves to these boundaries. No matter how realistic or not a humanoid robot looks, it is a robot - a machine, not a human being. It should not be held to the same standards and expectations as its organic counterpart. It doesn't have to be limited to what a human can do. Instead, a humanoid robot should be fitted out to do things a human can't do. Thus, it is able to form and expand the architectural environment it belongs to. A humanoid robot could be used as a social companion, a concierge, a clerk or it could potentially be transformed into an extension of an off-site archive, library or storage by providing

relevant information to the items in the allocated space. In this way it will enlarge its habitat through the constant connection with another physical space.

Thus, the multiple functions a robot can accomplish at a time will become even more important in the future. As time goes by, people change their needs when it comes to spaces, both in terms of size and functionality. While new designs obey the trends of the current era and are somewhat adapted to the needs that are foreseen for the near future, old and historic buildings can't be adapted so easily. For one reason or another, historic architecture can't always be completely renovated to fit and accommodate the new needs. In Europe, this is an issue because many cities strive to preserve their historic centers and are proud of having centuries-old buildings, standing and being used. In order to preserve such buildings and their historic spirit and value, architects are restricted in what they can do in order to adapt them to the new needs and standards. In some cases restrictions limit or forbid structural changes. Therefore, architects need to find innovative ways to satisfy the functional demands so that the buildings can still be actively used. Such a way can be the utilization of humanoid robots.

3.3. Pepper integrated in a public Space

This chapter is focused on two examples showing how the humanoid robot Pepper can be implemented in public spaces in Europe. However, in both cases Pepper is separated from the general flow of people. It does not roam its surroundings freely and does not hold a prolonged conversation with the passing-by people.

The humanoid robot Pepper is already popular in Japan (Robotics, n.d.). However, it is a rather novel experience for Europa. This is due to the difference in culture which has been discussed in Chapter 2.2.2 .



Fig. 3. 3 A humanoid robot Pepper at Brussels Airport, Belgium (Petrova, 2017)

The first example shows Pepper welcoming travelers at the Brussels Airport, Belgium. Pepper is positioned on a red carpet to separate it from the people. Pepper greets travelers and provides them with information in three languages: English, French and Dutch (see Fig. 3. 3). Approaching the robot is supervised by a human responsible for the control and observation of the robot.

The second example shows Pepper in the office of DSK Bank in the shopping mall Serdika in Sofia, Bulgaria (see Fig. 3. 4 and Fig. 3. 5). The robot is programmed to understand several words and phrases in Bulgarian. It provides information about the bank services and the bank portfolio which are projected on its on-board tablet. Pepper is positioned on a platform surrounded with vertical barriers on two sides.

Information on what the robot can do and what it understands is projected on a fixed tablet on the third side (see Fig. 3. 6).



Fig. 3. 4 Pepper being charged, DSK bank, Serdika mall, Sofia, Bulgaria
(Petrova, 2017)

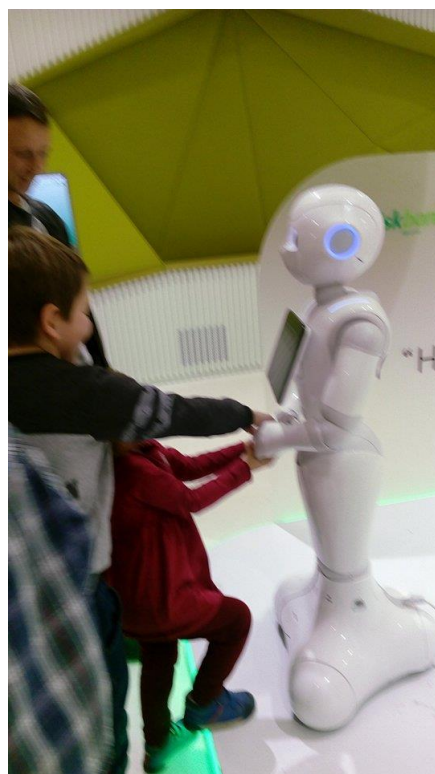


Fig. 3. 5 Pepper interacting with people, DSK bank, Serdika mall, Sofia, Bulgaria
(Petrova, 2017)

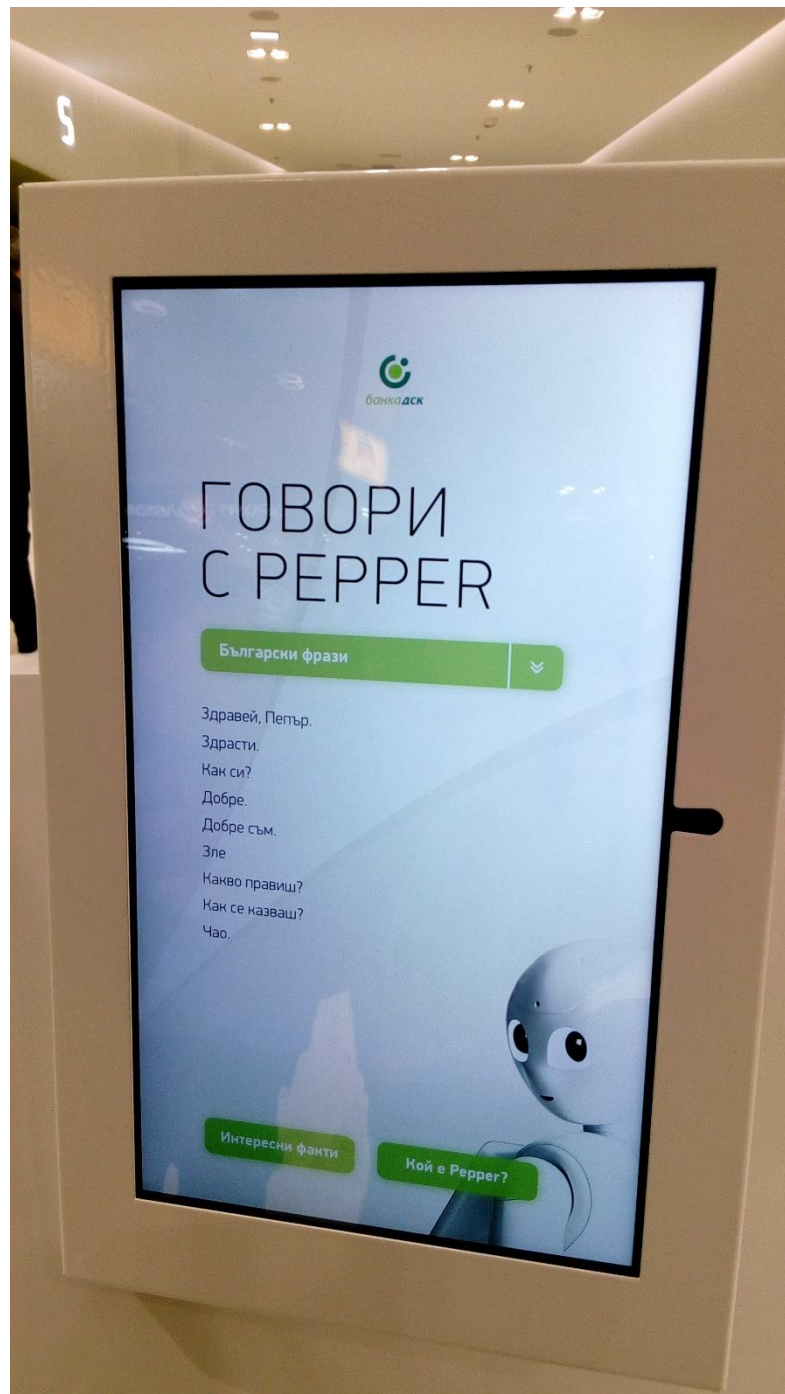


Fig. 3. 6 Tablet providing information on Pepper functions, DSK bank, Serdika mall, Sofia, Bulgaria (Petrova, 2017)

4. Architectural Space

Within the scope of this work it is essential to define what the term 'space' means. In general 'space' can mean different things to various groups of people depending on the context it is used in. Even when a person consults a dictionary (Press, n.d.), the term does not have an unequivocal meaning. The term 'space' can be used to describe:

- an empty area that is available to be used; the area around everything that exists, continuing in all directions: a gap; an empty or uncovered place;
- (time) an amount of time;
- (beyond Earth) the region outside the Earth's atmosphere, in which all stars and other planets, etc. are situated;
- (Business English/Marketing) 'space' is a section of a newspaper, magazine, or website that is sold to companies who want to advertise.

This shows how starkly the meaning of space varies depending on the context. In terms of this work, it is important to determine how architects define space. An architect might define space through volumes, masses, buildings and the relation between them.

The definition reflected in this work is given by Pierre Von Meisse, who states that: 'Architectural space is born from the relationships between objects or boundaries and from the plains which do not themselves have the character form of objects, but which define limits.' (Meiss, 1990). Another concept of space is given by Mauricio Mondragon and Luis Lopeza (Mauricio Mondragon, 2012). They define space as a container of bodies and objects. This clearly proves the multi-layered

character which the meaning of words has. It is a matter of context, interpretation and perspective. In this sense, an architects' design of space is a reflection of their interpretation of the context (Fig. 4. 1). How architects do it and what tools do they use in order to realize their design is a matter of their personality, preference and views. This being said, architects always seek new and interpret the exciting tools to realize their visions, to create unique designs and to think outside the box. Based on this using the behavior of a humanoid robot positioned in a space can be considered an unconventional design tool. In spite of space being usually defined through its physical shape and the physical borders that encompass it, this is neither always the case, nor does it have to be.

Therefore, space can also be an abstract unification or division of spaces based on a custom made system of rules.

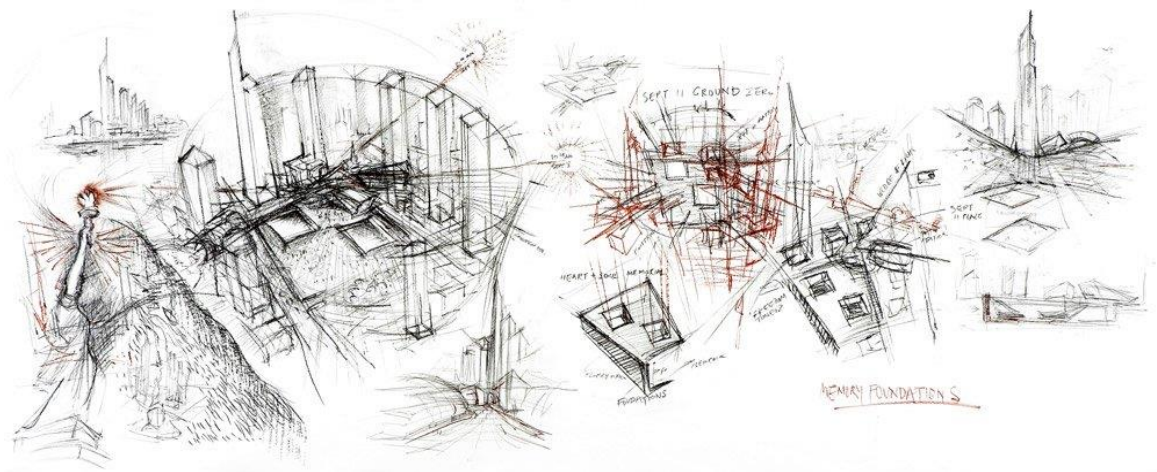


Fig. 4. 1 Daniel Libeskind sketch depicting multiple interpretations of the concept of space. This is a huge scroll depicting the master plan for the Ground Zero site in New York (Libeskind, 2013)

5. Humanoid Robot Pepper vs Humans' Navigating in Architectural Space

5.1. Pepper perception vs humans' perception of the surrounding environment

It is without a doubt that robots and humans see, perceive and understand the surrounding environment and the pertaining space differently. Among the major reasons for this are the different tools they engage in accomplishing this task. The robot 'sees' with its sensors and 'perceives' the environment based on its algorithms. Its 'understanding' is limited to the database it is connected to. Humans, on the other hand, rely on a combination of their senses: sight, hearing, touch, smell, as well as, the knowledge and experience they have accumulated over time. Their senses are far more complex, adaptable and experienced than any sensors present robots have.

As the presented work deals with a humanoid robot called Pepper from here after Pepper behavior will be studied, explained, and utilized in a design. As already said a human's perception is more accurate than that of Pepper. A big contrast is observed in comparing Pepper space mobility to that of a human. People do not need to keep as large safe distances as the robot does in order not to bump into obstacles. For example, it is more than natural for humans to walk up to a wall with little to no distance between themselves and the wall, while Pepper sensors cause the robot to keep a minimum safe distance of 30 cm from any detected frontal obstacles when Pepper moves with 0.35 m/s (default speed). The required security distance to frontal objects, based on Pepper speed is given in Table 1.

Speed (m/s)	Frontal security distance (m)
0,1 (minimal speed)	0,12
0,2	0,17
0,3	0,25
0,35 (default speed)	0,30
0,4	0,35
0,5	0,40
0,55 (maximal speed)	0,40

Table 1. Frontal security distance (Aldebaran/SoftBank, n.d.)

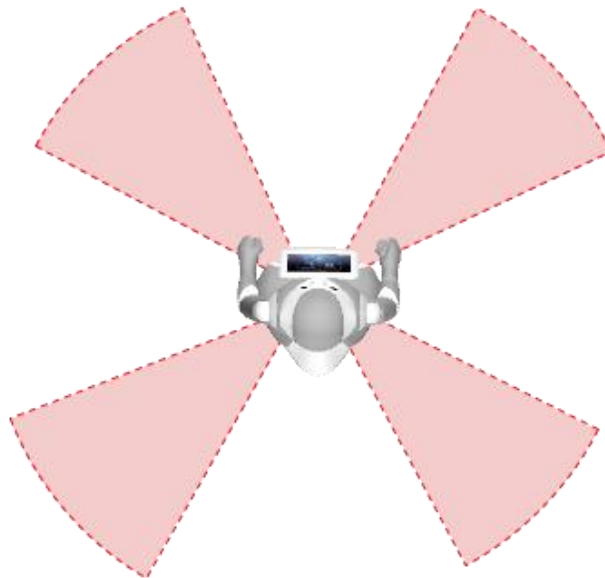


Fig. 5. 1 Pepper: Four main Blind Zones (Aldebaran/SoftBank, n.d.)

Furthermore, while humans rely on their periphery vision alongside their frontal one, for Pepper there are blind zones which its sensors do not cover (see Fig. 5. 1). In order for the humanoid robot to detect objects in these zones, Pepper needs to rotate either its head or its base.

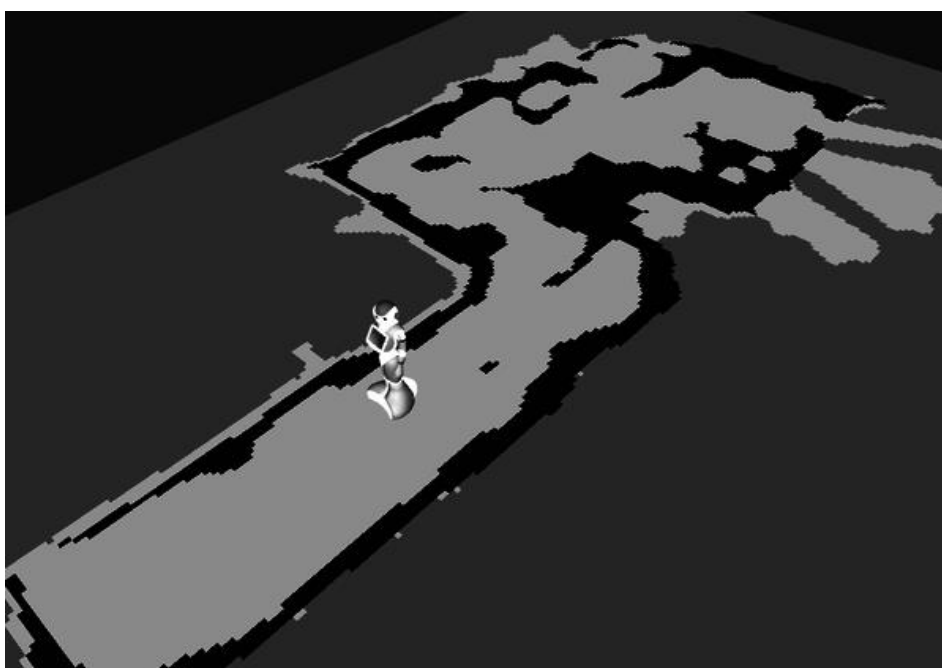


Fig. 5. 2 Pepper gray - scale visualization of a pace exploration, (SoftBanks, 2017)

This strongly influences the way Pepper navigates in its environment. When it explore its surroundings Pepper generates a gray-scale plan (see Fig. 5. 2). It depicts what the robot sensors determine as accessible and inaccessible space within the surrounding environment. In the plan pale gray is used for the space in which the humanoid robot can maneuver and black for any physical obstacles in its path and space boundaries. Based on this exploration in the gray-scale plan, existing furniture is categorized as the same kind of obstacle as a wall. This means

that for Pepper a room is not defined solely by its walls but also by the specific arrangement of the objects in it.

In contrast, humans' perception of space is different than the one described. On one hand, people clearly distinguish between walls, other unmovable boundaries and furniture, which is movable and they can rearrange themselves. On the other hand, it is in humans' nature to maneuver in close proximity to walls, tables, chairs, etc. or to rearrange mobile obstacles in a safe way with respect to the planned route.

In this sense, Pepper interpretation of its surroundings is more imprecise than that of a human, fixed and generic, but still accurate enough and sufficient for the robot need to navigate in them. Furthermore, when the robot calculates and selects a path, it selects the one that is secure for it without disturbing anything in the space, while humans can consider the most optimal or the shortest of all routes. This reinforces John Jordan's statement that '...robots don't need to replicate human performance, they just need to be good enough' (Jordan, 2016).

Following this line of thoughts, the contrast between Pepper and humans' perception can be considered from a new perspective. Pepper capabilities and autonomy are defined by what it is programmed to do. Even then it may need the assistance and guidance of a human in order to accomplish successfully some of the tasks. The guidance can be something as minor as requesting confirmation that it has understood a posed question correctly or something as complex as requesting help for localizing and positioning itself in the space.

Consequently, Pepper can neither accomplish all actions completely autonomously, nor at this point can it complete them simultaneously. In other

words, it is not just the humans that need the robot assistance but also the robot that needs the help of humans in order to accomplish its tasks successfully.

5.2. The Parallel between Pepper Perception of Space and a Nolli Map.

Space is boundless. In urban architecture it is depicted in figure-ground diagrams that categorize it either as built or as unbuilt in the context of the urban environment. Regarding the way in which Pepper depicts its surroundings it is necessary to say that the created by it two dimensional diagrams of its surroundings strongly resemble the idea of a Nolli plan, however, applied on a much smaller scale than the initial concept.

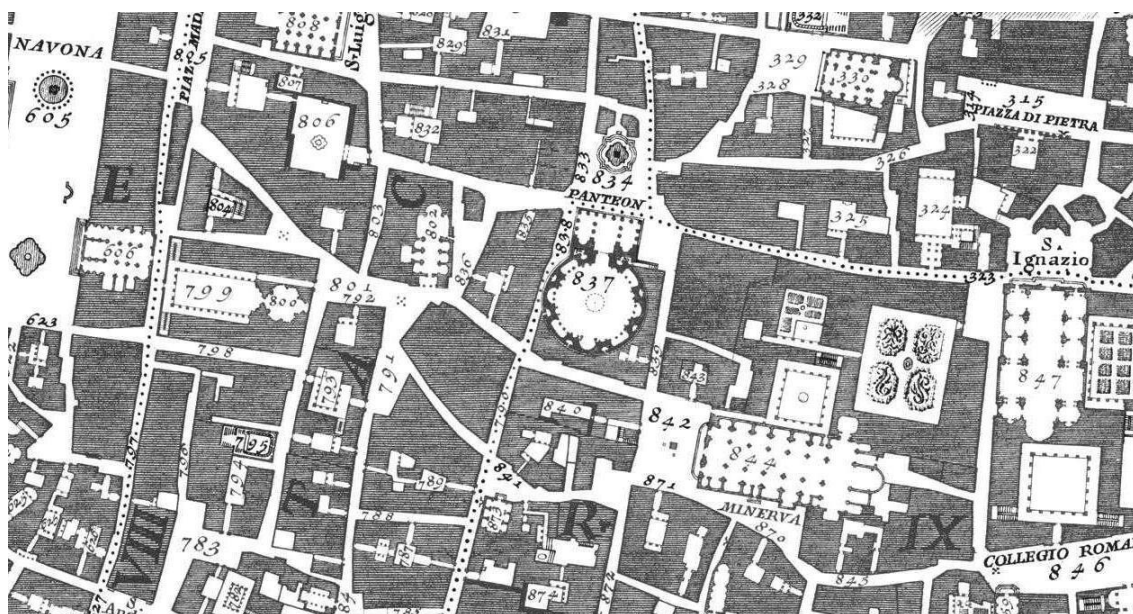


Fig. 5. 3 Nolli map: La Nuov a Topografia di Roma, 1748, (Ng, 2013)

For example, the Nolli master plan of Rome (see Fig. 5. 3), dating from the first half of the 18th century merges the existing enclosed public spaces to the open civic spaces when dividing the space into sub-spaces. The enclosed public spaces of the colonnades in St. Peter's Square and the Pantheon are depicted as open civic space (Tice, et al., n.d.).

Pepper generates a similar gray-scale plan of the encompassed space after every reading of its surroundings (see Fig. 5. 2). Unlike the Nolli master plan, the Pepper graphic does not have sharp edges and smooth lines. It resembles an irregularly shaped bubble, where not only the walls but also every other object placed in the space is considered a space building element. In other words, Pepper interpretation of the unbuilt or built environment is defined by the space in which it can maneuver. This is the negative space between all obstacles and boundaries. When moving Pepper software only regards the space the robot can negative in and the set of routes it can undertake within it to reach the desired destination.

The bubble of space, depicted in the Pepper generated gray-scale plan, can be seen as a parallel to the open civic spaces in the urban environment. At the same time this is the same space within which a human can safely maneuver, while sharing a room with Pepper.

Humans are much more agile and see the space differently, but if they have to define a detail free space they can move in, they draw a graphic similar to a Pepper generated one. The human's graphic incorporates the robot one but it is somewhat bigger and edgier.

5.3. Pepper Implementation as an Office Concierge in the Department for Architecture Theory and Philosophy of Technics

The goal of this project is to implement a humanoid robot Pepper as an office concierge in the Department for Architecture Theory and Philosophy of Technics at the Vienna University of Technology. In this context the robot comes in contact with two major groups of people - the workers (the people who work in the department) and the visitors. That means that there are two target groups of users that need to be considered:

- people who interact with Pepper on a daily basis;
- people who see and meet the robot for the first time.

Chapter 3 'Robots in continuous Interaction with Humans.' gives a brief overview of two documented scenarios where people come into continuous contact with robots in their work environment. The Snackbot experiment is extremely interesting for this work. On one hand the robot, although different from Pepper, is humanoid. On the other, it documents the time needed and the stages the participants go through from not knowing the robot to getting attached to it and accepting it as a part of their environment.

The cited experiment also provides an insight on how the people working for the Department for Architecture Theory and Philosophy of Technics might feel about having Pepper nearby. The fact that they will be in contact with it far more often than occasional visitors raises the question in what way their interaction with the robot will be different. As most users would expect Pepper will recognize the people

it comes into contact with on a daily basis by learning their faces. Based on that Pepper can inquire how their day is going, how they feel or initiate any other sort of small talk that will give the communication a more personal touch. Additionally, this can be mirrored in its gestures, through physical contact between the robot and the people such as shaking hands or hugging. All of these tasks can be technically achieved based on what the robot is able to do.

However, different technical obstacles prevent the system from working properly in such a scenario. For example, Pepper can learn to recognize someone's face and the recognition results can further be improved by repeating the process over time. Despite of that, with the present inbuilt sensors if a person wears glasses, when the robot is taught to recognize their face, the learning process will fail. This is a technical problem that will likely be solved with future updates. As hardware and software for face recognition provide better recognition and develop quicker for other machines, e.g. cellphones, it can be deduced that soon solutions for robots will be implemented, as well.

For overcoming the technical issue with glasses two scenarios have been investigated.

The first scenario requires that people wear contact lenses or take their glasses off. However, that is not a viable solution since contact lenses are not used by everyone and following the other path people will have to take their glasses off every time the robot has to recognize them. It can be entertaining to do once or twice but it is impractical to do all the time.

The other more practical solution in terms of the Department is to create an identification badge or card that Pepper can detect and recognize. Due to its

complexity this solution has been considered outside the scope of this work. Furthermore, using the robot for issues of security or surveillance is not part of the social role attributed to the office concierge functions.

As a result, I have selected not to use the face recognition function within this project because a significant amount of people wear glasses.

Another point concerning this sort of interaction is the body language of the robot and more specifically the physical contact between Pepper and the human. When people are in constant contact with the robot they quickly develop an understanding regarding its handling, what they can and cannot do and the reaction their actions can trigger in the robot. In this aspect, people who interact with it regularly can come into casual physical contact with it (e.g. pat it on the head, shake its hand, etc.) without the concern that this might result in a problem or an inadvertent damage of Pepper. Furthermore, the physical contact within the interaction strengthens the personal aspect of the developed communication process.

The most important part of the interaction with Pepper is the verbal exchange. The people who talk with it can quickly develop a good understanding as to how they need to speak to the robot in terms of voice strength, clarity and speed of pronunciation, maximum distance from Pepper in which it can understand them, the best way to position themselves when talking to it. They will also be more familiar with the mistakes the robot makes.

The interaction between Pepper and the second main group of users, the visitors, has an entirely different character. That is due to the fact that they do not come often if at all into contact with humanoid robots. Therefore, initially the visitors are not familiar with the position and the functions of the robot within the Department.

That is why it is important to design accurately as many scenarios as possible in order to limit the misunderstandings, mistakes or incidents that occur because of the visitors' unfamiliarity with situation.

For students or any other visitors to the Department, Pepper is meant to serve as an aid. However, they are unfamiliar with the way they have to communicate with the robot, or even unsure if they are supposed to interact with it. This imposes the need for a different approach to be adopted. The pattern of speech, gestures and signaling also needs to be adapted to the peculiarities of this kind of interaction. In this scenario Pepper speech has to be welcoming and engaging without being too personal. This approach can be achieved by Pepper greeting and communicating with people in a friendly way without trying to initiate any physical contact through its gestures. This restriction is due to several reasons:

- lack of initial information on how people feel towards Pepper; despite being charming and friendly as it is intended to be, this does not necessarily mean that it would be how first - time visitors see it.

- entering the personal space of somebody or provoking contact could result in negative reaction and diminish the desire of that person to keep on interacting with Pepper.

- contact between Pepper and people unfamiliar with it could lead to an unintentional incident or damaging of the machine due to lack of caution, unfamiliarity and/or overexcitement from the side of the human (e.g. shaking hands is too risky as Pepper's hand can easily be damaged).

In terms of speech it is important to ensure that Pepper understands its interlocutor correctly and therefore provides accurate information and answers posed

questions. It is important that people speak clearly, loudly and close enough to Pepper, so that it can understand them. In order to ensure the communication quality if the robot doesn't understand the human, it provides several solutions: it suggests the human to speak louder, or slower, or towards the robot head where the receivers are situated, etc.

Pepper also indicates through its speech which language(s) it speaks and understands at the given time. While it can in general interact in 21 languages, the currently active language of communication is specified. This means that if Pepper speech recognition is set to English, this is the only language it will process in the scope of the conversation. It will not understand any other language.

The built-in tablet on its body can be utilized as a means to convey information through visuals which ensures an easier and more engaging conversation. If Pepper needs to provide some information to a visitor about the location of a room or an object within the Department, it can show them a plan graphic as a reference to spoken directions. Another considered scenario is when Pepper is asked about a person or about providing information on who visitors need to contact in a given situation. Then Pepper can use the tablet to show a photo of the person's face and visual information will serve as a reference to oral one.

In conclusion, this chapter summarizes the two target groups who will benefit from Pepper as an office concierge and identifies the approach and main and tasks that Pepper is to fulfil on a daily basis. Further, Pepper behaviors are programmed and tested in the Department in order to fulfill all of the discussed tasks which comprises the core of the presented work.

Additional materials including the behaviors and scripts from the following chapters are provided online. (see Fig. 5. 4)



Fig. 5. 4 QR Code1: QR code of the developed domain

6. Programing Pepper to be utilized as an Office Concierge in the Department for Architecture Theory and Philosophy of Technics.

6.1. Different Approaches on programing Pepper Speech Recognition.

One of the most important aspects of human – Pepper interaction is conversation. It provokes immediate reactions from both sides. Based on that either side can adapt to a given situation. This helps minimize the misunderstandings.

In order to have a conversation Pepper needs to be able to understand what people are saying and it needs to be able to speak. This part of the work focusses prevailingly on Pepper ability to understand while the next one gives an insight into its ability to speak.

As evident from Fig. 6. 1, Pepper can be programmed to speak and understand 21 languages using Choreographe³. The figure gives a brief overview of the extent to which each language is supported by the software. The best supported languages are English and Japanese. If a project is to be executed using any of the other languages, for example German, possible challenges in terms of speech recognition and pronunciation of the robot must be taken into account.

³ Choreographe - a multi-platform desktop application that allows you to create very powerful behaviors (e.g. for interacting with people, dancing, sending e-mails, etc...), without writing a single line of code. In addition, it allows adding your own Python code to a behavior, (Robotics, n.d.).

6. Programing Pepper to be utilized as an Office Concierge in the Department for Architecture Theory and Philosophy of Technics

Locale		Language	Dialog code	Tools	Content and Samples	Notifications	ASR Engine	TTS Engine	Activity launcher	Conversation	Remote ASR Engine
Codification			Choregraphe		NAOqi API		Basic Channel		Remote		
ja_JP	Japanese	jpj	✓	✓	✓	✓	AiTalk	✓	✓	✓	✓
en_US	English	enu	✓	✓	✓	✓	Nuance	✓	✓	✓	✓
fr_FR	French	frf	✓	✓	✓	✓	Nuance	✓	✓	⊗	⊗
it_IT	Italian	iti	✓	✓	✓	✓	Nuance	✓	⊗	⊗	⊗
de_DE	German	ded	✓	✓	✓	✓	Nuance	✓	⊗	⊗	⊗
es_ES	Spanish	spe	✓	✓	✓	✓	Nuance	✓	⊗	⊗	⊗
zh_CN	Chinese	mnc	✓	✓	•	✓	Nuance	✓	⊗	⊗	⊗
nl_NL	Dutch	dun	✓	✓	•	✓	Nuance	•	⊗	⊗	⊗
ar_SA	Arabic	arw	✓	✓	•	✓	Nuance	•	⊗	⊗	⊗
ko_KR	Korean	kok	✓	✓	•	⊗	⊗	•	⊗	⊗	⊗
pl_PL	Polish	plp	✓	✓	•	⊗	⊗	•	⊗	⊗	⊗
pt_BR	Brazilian	ptb	✓	✓	•	⊗	⊗	•	⊗	⊗	⊗
pt_PT	Portuguese	ptp	✓	✓	•	⊗	⊗	•	⊗	⊗	⊗
cs_CZ	Czech	czc	✓	✓	•	⊗	⊗	•	⊗	⊗	⊗
da_DK	Danish	dad	✓	✓	•	⊗	⊗	•	⊗	⊗	⊗
fi_FI	Finnish	fif	✓	✓	•	⊗	⊗	•	⊗	⊗	⊗
sv_SE	Swedish	sws	✓	✓	•	⊗	⊗	•	⊗	⊗	⊗
ru_RU	Russian	rur	✓	✓	•	⊗	⊗	•	⊗	⊗	⊗
tr_TR	Turkish	trt	✓	✓	•	⊗	⊗	⊗	⊗	⊗	⊗
nn_NO	Norwegian	nor	✓	⊗	•	⊗	⊗	⊗	⊗	⊗	⊗
el_GR	Greek	grg	✓	⊗	•	⊗	⊗	⊗	⊗	⊗	⊗

✓	OK
•	Partial / Not tested
⊗	Not Supported Yet

Fig. 6. 1 Pepper language table, (Robotica, n.d.)

For this project three approaches towards a robust speech recognition are considered (see Table 2). They are based on using the following software:

- Google Speech Recognition;
- IBM Watson;
- Nuance.

Software	Internet Connection	Type of recognition
Google Speech Recognition	needed	Sentence based
IBM Watson	needed	Sentence based
Nuance	Not needed	Keywords based

Table 2 Speech Recognition software with respect to internet connection and recognition typology

The most important factors when choosing an appropriate software are: the percentage of recognized speech (measured in keywords or sentences) and the need for a fast internet connection so that the speech recognition works properly.

Both software, Google Speech Recognition and IBM Watson, function on sentence based speech recognition, which allows for more freedom and flexibility in a conversation. They ensure a good robot speech understanding because they connect Pepper to a vast data pool that gets constantly updated.

However, Google Speech Recognition and IBM Watson have two significant disadvantages. The first disadvantage is the need for a constant and fast internet connection. Their usefulness is determined by the time needed for a data transfer to and from the respective Google/IBM servers in order to have a successful human - Pepper communication. This renders both engines ineffective should the internet be slow or non-existent. The second disadvantage is related to the speech recognition being sentence based. Sentences are a more complex language unit than words. While using sentences means flexibility of the speech, it also increases

the chances for the robot to recognize something wrongly. Aside from that both engines require experience and a good understanding of how the system works. The only way to use Google Speech Recognition or IBM Watson is for the person programming the robot to set up the connection between the robot and the servers through a customized script. Another relevant factor that needs to be considered when using IBM Watson is that charges also apply.

Nuance engine is the default TTS (Text to Speech) and SR (Speech Recognition) Engine, incorporated in Choreographe, for English. The speech recognition used by Nuance is word based. This means that it works best with single-word commands and keywords. The robot can still understand sentences, however with a lower success rate than when using the other two options, Google Speech Recognition or IBM Watson. Additionally, the robot speech recognition functions offline. It does not need an internet connection which also means that it doesn't have a constant access to an online dynamic data base like the other two. A drawback of the software is that this limits it to a scripted by the designer data base incorporated in each project.

As a result, Pepper vocabulary and understanding are only as vast and extensive as it is scripted. Once scripted it is only active within the relevant project. It is not automatically available for other Choreographe projects. Pepper vocabulary using Nuance is determined by a static data base which can only be updated and enriched manually. A manual restart of the Choreographe robot behavior is needed for any changes to take effect.

It is important to consider that when using a word based speech recognition software, single word commands and short sentences have a higher recognition rate than long and complex ones. A way around this is to use sentences where a

specific word is programmed to be a keyword. This is a crucial point regarded in Pepper programming in terms of this work. A more extensive explanation about how it works within the script is given in the following sub-chapters.

6.2. Different Approaches on how to program the Conversational Capabilities of Pepper

A crucial factor for the successful interaction between a human and a robot is the robot ability to understand its interlocutor and to deliver appropriate replies to given questions and requests. There are several ways in which to initiate and simulate conversation when using the software Choreographe in the context of Aldebaran humanoid robot Pepper. This sub-chapter provides an overview of the different tested approaches, while comparing them and explaining their advantages and disadvantages. Based on that, 'Dialog' is chosen as the most appropriate tool for this project.

The first point to be covered in every verbal conversation is to ensure that the parties involved are capable to speak. While speaking is an inherent ability for most people it is not nearly as inherent for a robot. Nevertheless, making Pepper say something is a fairly easy task. There are several ways to do that using Choreographe. A short overview of the software interface is given in Fig. 6. 15 and its description.

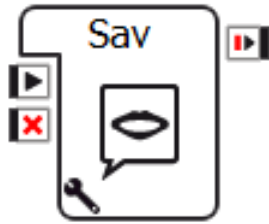


Fig. 6. 2 Say

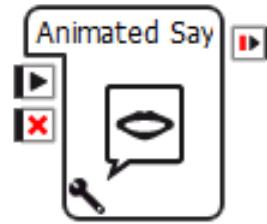



Fig. 6. 3 Animated Say

Speech can be programmed via *Say* (see Fig. 6. 2) or *Animated Say* (see Fig. 6. 3) boxes. All the user needs to do is to type what Pepper needs to say in the Text field that can be opened by clicking on the parameter button . The process is explained in detail in sub-chapter 6.3. Brief Manual on programming Pepper. The only difference between the two boxes is that the *Animated Say* includes an 'Animation' parameter that allows for movements simultaneous with the speech. The animations can either be programmed by the user and added to the module or loaded from the library available in Choregraphe. The 'Speaking movement mode' parameter can be set to contextual or random, or be disabled. This affects how well the robot movements fit the words it says. When using *Say* box any movements need to be added externally by using a separate movement command box. In a simple definition it does not make a significant difference if movements are programmed separately from speech but it makes the project significantly harder to handle and process in a complex situation.

Both boxes (*Say/Animated Say*) are a sufficient tool for a monologue, however, their application is limited to programming robot speech. They do not ensure that Pepper understands when spoken to. This makes them unsuitable as a primary tool for developing an interactive project, i.e. a project where the robot communicates with the interlocutor.

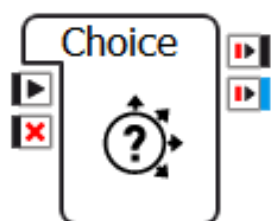


Fig. 6. 4 Choice



Fig. 6. 5 Choice (light)

A slightly more complex but also more useful option are *Choice* (see Fig. 6. 4) or *Choice (light)* (see Fig. 6. 5) boxes⁴. They work in combination with *Set Reco. Lang.* (see Fig. 6. 6), which allows the designer to set a conversation language. If the box is not present and/or a language is not set, the *Choice/Choice (light)* will be useless as Pepper will be unable to process anything said to it.



Fig. 6. 6 Set Reco. Lang.


The *Choice/Choice (light)* allows the robot to ask a preprogrammed question and to understand preprogrammed answers to this question. An overview of how to set this up in a definition is given in Fig. 6. 22, Fig. 6. 23, Fig. 6. 24 and the provided explanations.

One of the most significant drawbacks of this option is that when answering a Pepper question the interlocutor needs to formulate their answer in the same way it has been phrased in *Choice/Choice (light)*; otherwise the robot will not recognize

⁴ No difference has been determined between *Choice* and *Choice (light)* while developing this project.

and understand the answer. Should the word order be different or the answer be more detailed, it will not be recognized by the robot as one of the listed options.

Another drawback of the *Choice/Choice (light)* is that it can only be used for a one predefined course in a conversation where deviations are not possible. Thus, the communication can be described as a linear definition where the human is limited to answering the Pepper questions without being free to go outside of the scripted conversation. This strongly limits the usefulness of these command boxes. Despite of this, *Choice* and *Choice (light)* have the advantage of limited understanding (the robot demonstrates an ability to communicate by asking and interpreting the answers to a limited number of pre-programmed couples 'question-answer').

The *Choice/Choice (light)* can be customized to have multiple possible answers leading to a number of outcomes. Fig. 6. 24 and its description give more details about how to customize them. The shown view is opened once the user clicks twice on the command box. Additionally, by clicking on the parameter button  the user is given access and can alter the box parameters. While tuning the parameters in the 'Set parameters of Choice' panel (see Fig. 6. 7) the user should bear in mind that the aim is to develop an optimally functioning solution. For example, lowering the value of 'Minimum threshold to understand' will result in a higher number of recognized answers but the chance for falsely 'understanding' also grows. The default threshold of this characteristic in our case is 30%, i.e. the input value is 0.3. Another parameter that should be considered is 'Repeat validated choice', which limits misunderstanding because the robot repeats any answer it recognizes and thus it can easily be verified by the user. This is especially beneficial if the 'Minimum threshold to understand' is lowered. In this case, it can be used as an extra reference to check if the speech recognition functions correctly.

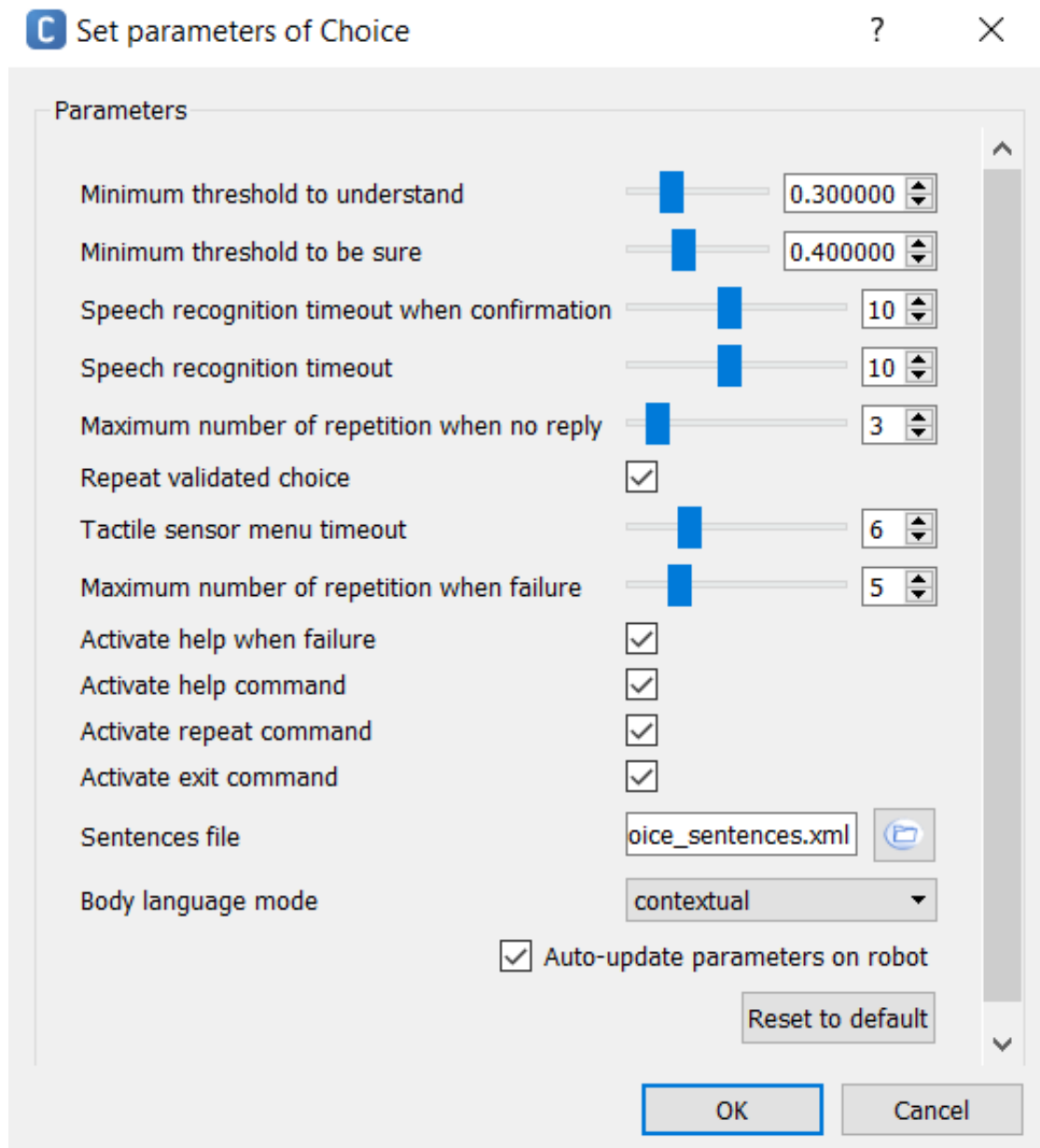


Fig. 6. 7 Choice box Parameters

Despite all pointed out so far advantages the *Choice/Choice (light)* does not give the feeling of natural interaction between a human and Pepper. It is strictly limited to the scripted conversation flow. It does not allow for questions to be posed in a random order or to be skipped. They are always posed in the preprogrammed in the definition (robot behavior) order. Hence, it does not allow the interlocutor to go off script.

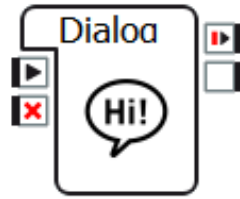


Fig. 6. 8 Dialog

According to me, the best tool to simulate natural interaction, using Choreographe, is *Dialog* (see Fig. 6. 8). However, its use is a bit more complex than that of the options presented so far.

Dialog allows the designer to script a variety of scenarios. Multiple topics, where each one is seen as a different scenario can be scripted and organized based on predefined rules. It can either be based on a hierarchy or on an equality between the script lines. Within an internal hierarchy each scenario can be triggered by a word or a sentence, using a tree branch structure. Additionally, this approach allows for additional functions executed by the robot to be incorporated (e.g. gestures, movements or visual data) that complement the speech. This increases the interaction complexity.

As already stated *Dialog* is a more complex tool than the ones discussed so far. Everything it contains needs to be accurately and precisely scripted in advance by the designer; otherwise the command box does not function properly. The syntax of the scripts introduced is very close to the common language (in this case English) which is a strong advantage. The punctuation plays an important role and strongly impacts the content of the script. This topic will be discussed in further detail in the following paragraphs and in sub-chapter 6.3 Brief Manual on programing Pepper.

Sometimes a project contains more than one *Dialog* script. In order to distinguish between different *Dialog* scripts, they can be named after the dialog topics they contain. A step by step explanation of how to set up a *Dialog* box and an allocated to it script is given in, Fig. 6. 25, Fig. 6. 26, Fig. 6. 27 and their descriptions.

When speaking about 'dialog' generally we understand a back and forth communication between two parties. In our case, one of the parties is the humanoid robot and the other - the human. The information fed to Pepper by its interlocutor is defined by the function 'u: (.....)' in the *Dialog* script. The information contained in the brackets can refer to words, sentences, concepts, events, etc. In return the robot response can also be combined with pre-installed behaviors.

The *Dialog* script syntax includes some highly useful features that are used to structure the dialog. Below some of the features, I have used in my *Dialog* scripts, are listed:

- concept (~.....)
- event (e:Variable);
- random (^rand);
- adding a behavior (^start);

An explanation of how to use them is provided in the three tutorials cited below (see Fig. 6. 9, Fig. 6. 10 and Fig. 6. 11).



Fig. 6. 9 QR Code 2:
Dialog Tutorial 1
(HEIRlab, 2015)



Fig. 6. 10 QR Code 3:
Dialog Tutorial 2
(HEIRlab, 2015)



Fig. 6. 11 QR Code 4:
Dialog Tutorial 3
(HEIRlab, 2015)

A full overview of the tool and its features is also available on the website doc.aldebaran.com under the QiChat section. The 'Syntax' sub-section (see Fig. 6. 12) gives a detailed overview, while the 'Cheat Sheet' sub-section (see Fig. 6. 13) is well suited for a quick overview of the most important features.



Fig. 6. 12 QR Code 5: *Dialog Syntax Rules* (Robotics, n.d.)



Fig. 6. 13 QR Code 4: *Dialog Syntax Cheat Sheet* (Robotics, n.d.)

Dialog(Fig. 6. 8) needs to be combined with *Set Language*(Fig. 6. 14) analogous to the way *Choice*(Fig. 6. 4) needs to be combined with *Set Reco.Lang*(.Fig. 6. 6).

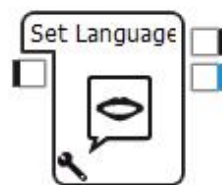


Fig. 6. 14 *Set Language*

6.3. Brief Manual on programing Pepper behavior for designers and planners in architecture

This sub-chapter provides practical information on how to program a humanoid robot Pepper. It is based on the functions needed so the robot can be successfully utilized as an office concierge in the Department for Architecture Theory and Philosophy of Technics. The workflow is presented through a series of screenshots. They all follow the same pattern. The relevant fields in each screenshot are framed and numbered in red. A short text explanation is provided following the frame numbering used in the screenshots.

1. *Choreographe Overview (see Fig. 6. 15)*

The screenshot shows the software interface as it looks upon starting the program. It gives a basic overview of several crucial for the program running characteristics.

- 1) Work space - the field in which command boxes are organized and interconnected in order to create a robot behavior.
- 2) Project name and Properties menu – The Properties menu allows for the project name to be changed and for project language(s) to be set.
- 3) (Active) Behavior(s) – A project can contain more than one behavior, in which case it is important to pay attention which one is currently active.
- 4) Connect/Disconnect buttons – The green button opens up a connection box from where the user can select by name their physical robot (see Fig. 6. 16), while the red one disconnects the robot. Connection to the virtual robot is set up automatically.
- 5) The framed brackets show if Choreography is connected to a virtual or a physical robot.
- 6) Main categories of available pre-programed commands.
- 7) Show Filter (Search) – A tool for quicker browsing through the available commands.
- 8) Robot Applications – A list of available applications installed on the presently connected robot. This list may vary from robot to robot.

Robots will refine behavioral space in architecture

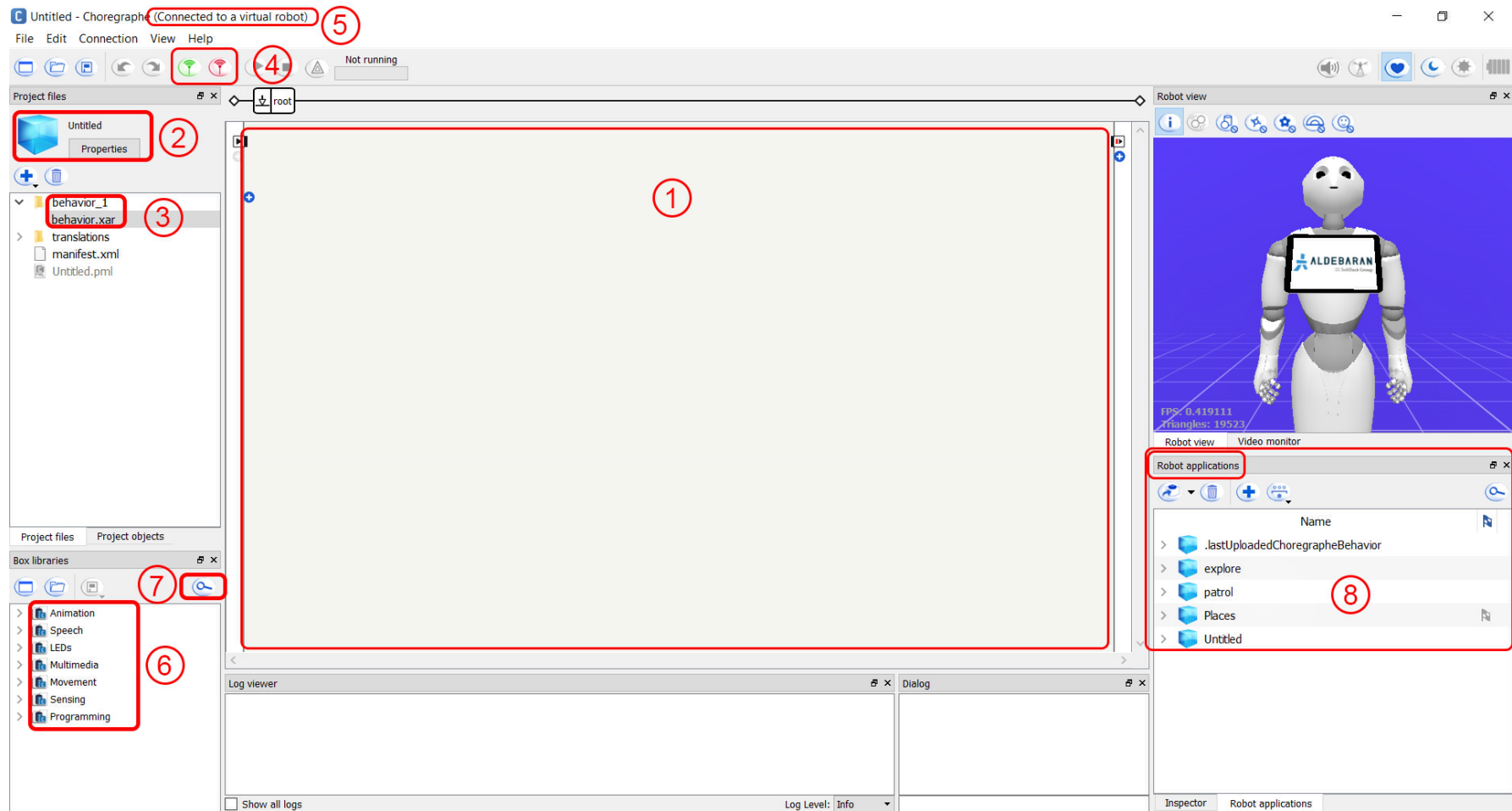


Fig. 6. 15 Choregraphe Overview

2. *Connecting to the physical robot (see Fig. 6. 16 and Fig. 6. 17)*

Fig. 6. 16 and Fig. 6. 17 show the process of establishing a connection to a physical robot. This is important because all functions applied throughout the project have to be tested and run on the physical robot.

In both screenshots frames 1) and 4) show the changes in interface upon switching from a virtual to a physical robot. Fig. 6. 16 shows the two steps in establishing the connection. They can be implemented via buttons 2) and 3)

- 1) The framed text in brackets shows if Choreographe is currently connected to a virtual or a physical robot.
- 2) Connect button.
- 3) The 'Connect to...' window shows all available robots Choreographe can connect to.
- 4) Robot Applications – The list varies between a virtual and a physical robot and can be different for different physical robots, too.

3. *Programming the language and the speech of the robot (see Fig. 6. 18
and Fig. 6. 19)*

Fig. 6. 18 and Fig. 6. 19 show how to program the robot language and speech.

- 1) Show Filter (Search), pointed in Fig. 6. 18 opens up a search field. By introducing the correct key word we can access the required application (see Fig. 6. 19).
- 2) Search for an application.

Robots will refine behavioral space in architecture

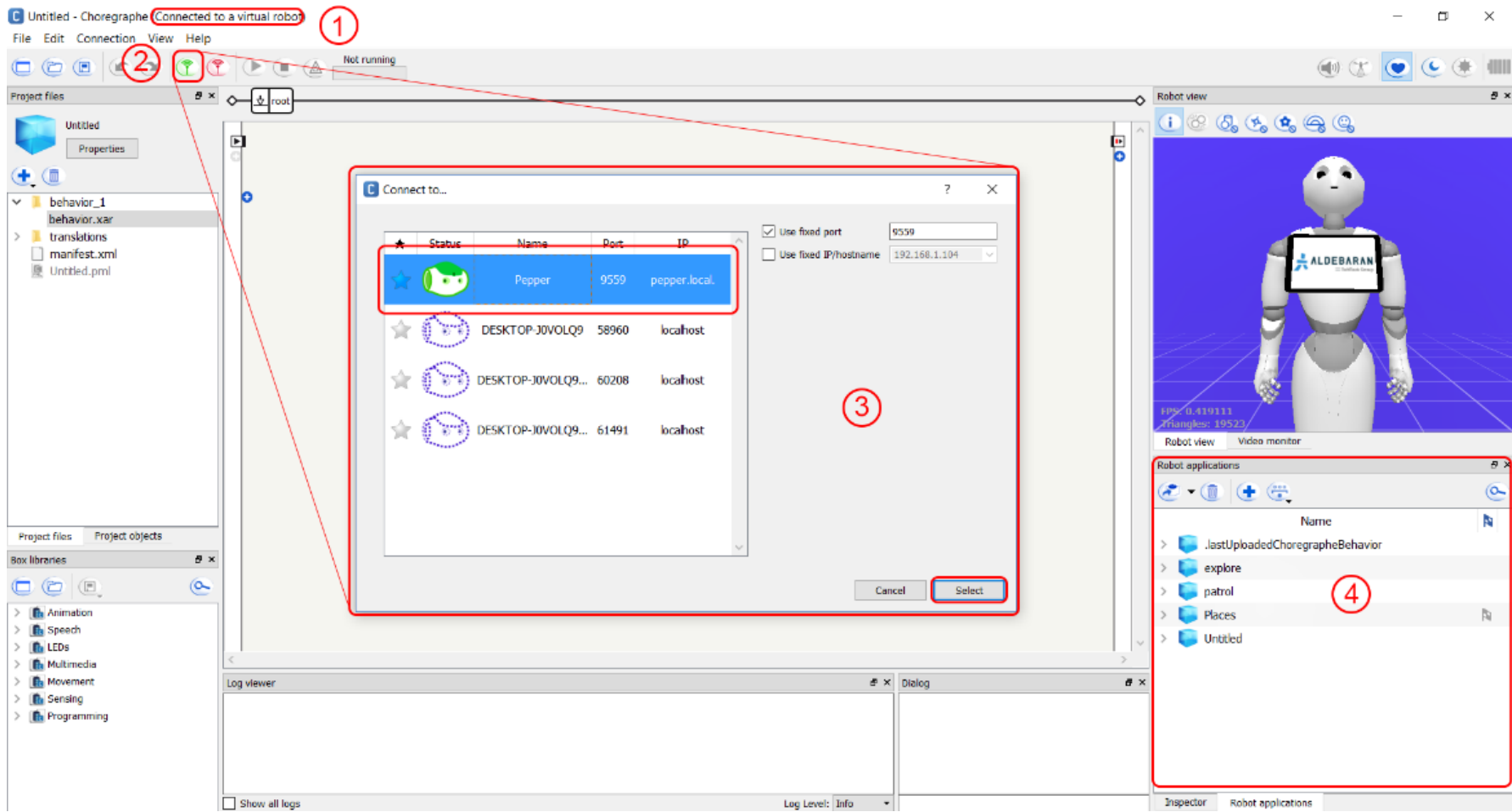


Fig. 6. 16 Choregraphe: Connecting to a physical robot

6. Programming Pepper to be utilized as an Office Concierge in the Department for Architecture Theory and Philosophy of Technics

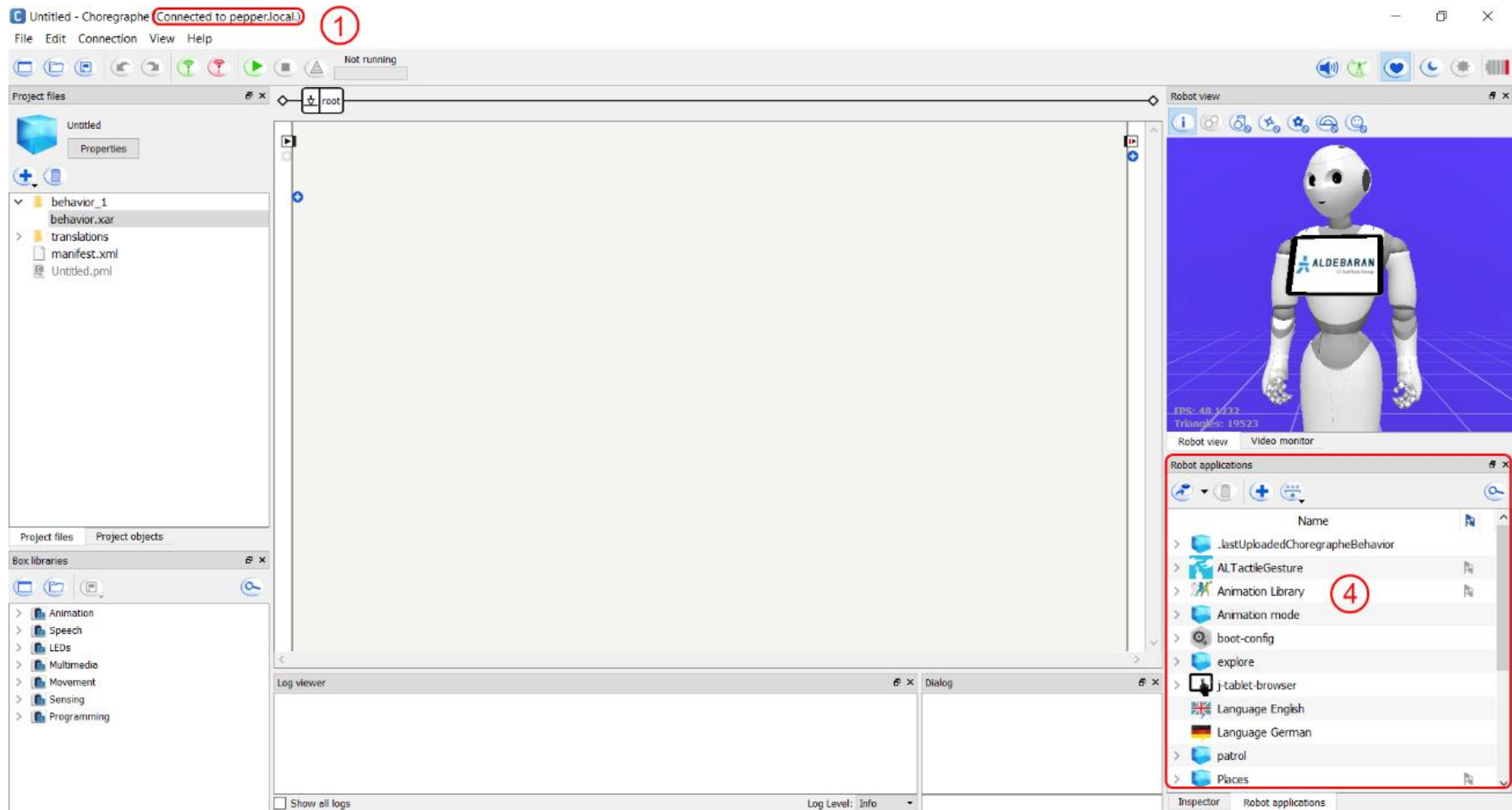


Fig. 6. 17 Choregraphe: Connected to the physical robot

Robots will refine behavioral space in architecture

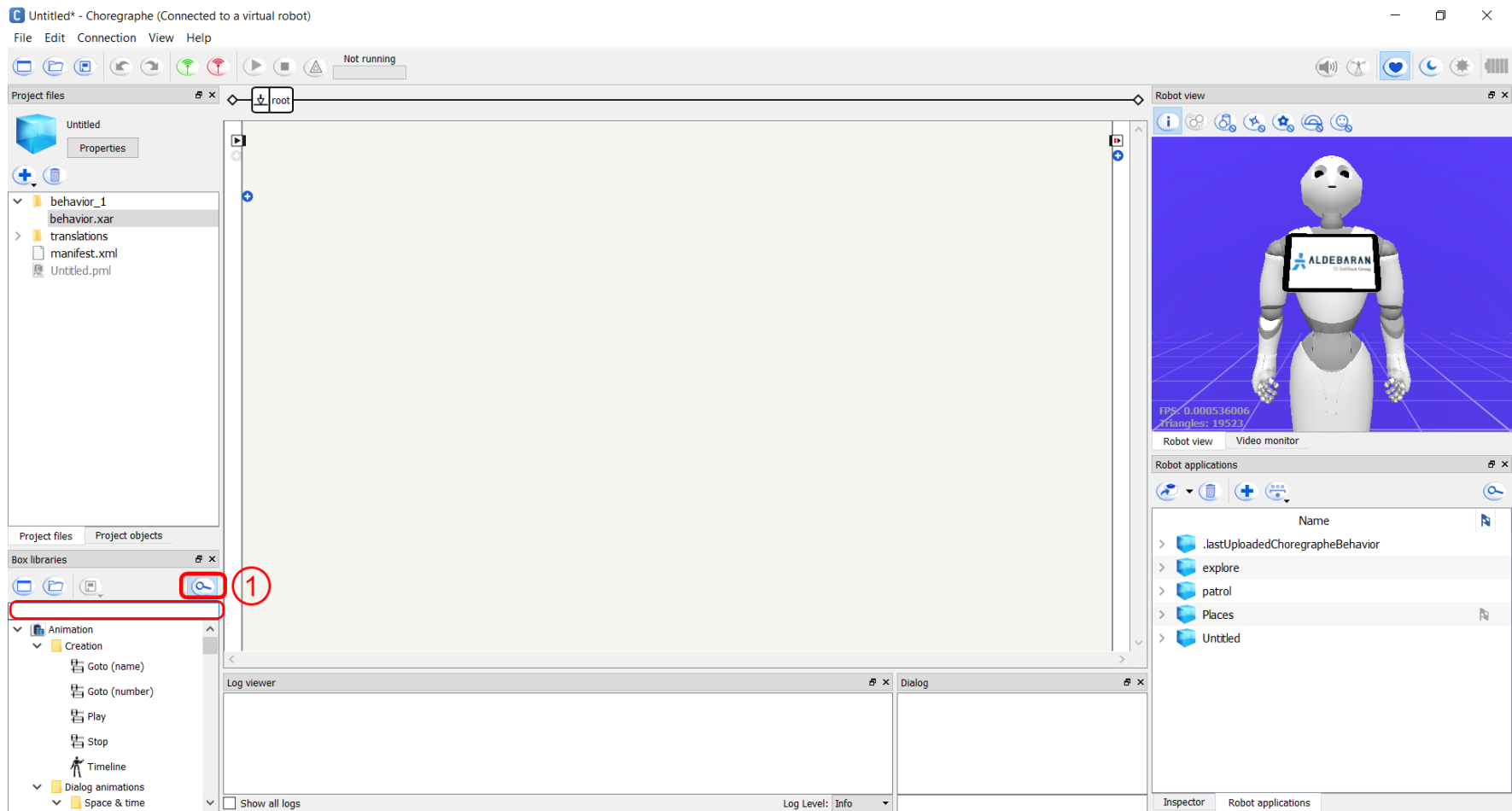


Fig. 6. 18 Choregraphe Command Search

6. Programming Pepper to be utilized as an Office Concierge in the Department for Architecture Theory and Philosophy of Technics

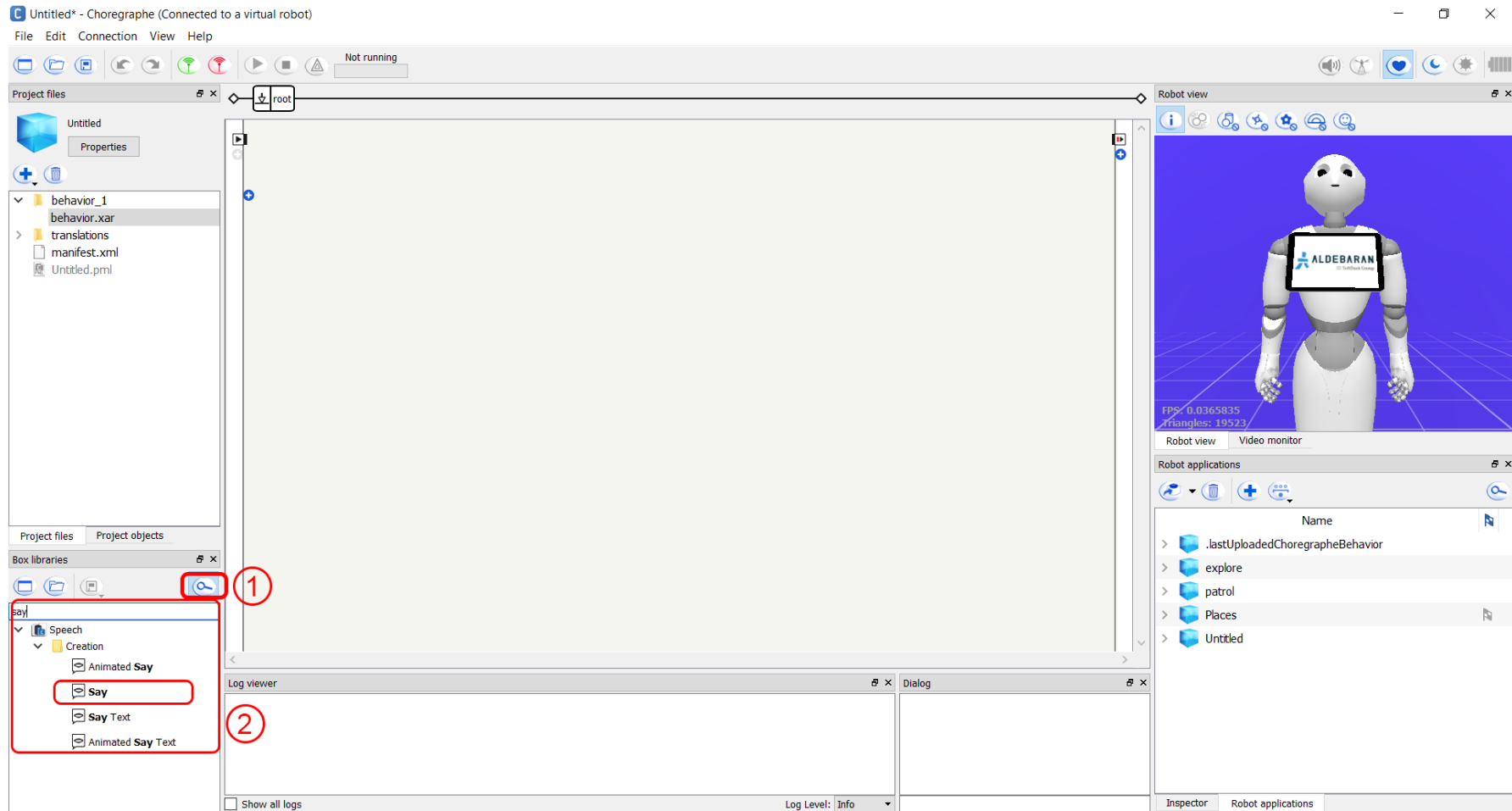



Fig. 6. 19 Choregraphe Speech: Say

4. *Choreographe Speech: Say Customization (see Fig. 6. 20)*

This screenshot shows step by step how to find, use and customize the Say in Choreographe

- 1) Show Filter – Allows a command box to be found quicker. This step can be omitted. The alternative is to find the command manually by browsing through all available commands.
- 2) Find the necessary command.
- 3) Drag and Drop – Drag and drop the command from the list of commands to the Work space (see Fig. 6. 15).
- 4) Open the parameters window by clicking on the parameter button  at the bottom left corner of the box.
- 5) Adjust the parameters - Any desired text can be filled in the Text field.

5. *Choreographe Speech: Animated Say (see Fig. 6. 21)*

The screenshot shows how to find, use and adjust *Animated Say* in Choreographe step by step. The first four stages coincide with those relevant for *Say* (see Fig. 6. 20). Thus, their detailed explanation is intentionally omitted.

- 5) Adjust the parameters - Any desired text can be filled in the Text field. Additionally, a movement, which complements the text, can be added in the 'Animation' field. The movement can be custom developed but then it needs to be installed on the robot as a Robot Application in order to use it (application installation is covered in sub-chapter 7.3). The 'Speaking Movement Mode' can be set to contextual, random or disabled

6. Programming Pepper to be utilized as an Office Concierge in the Department for Architecture Theory and Philosophy of Technics

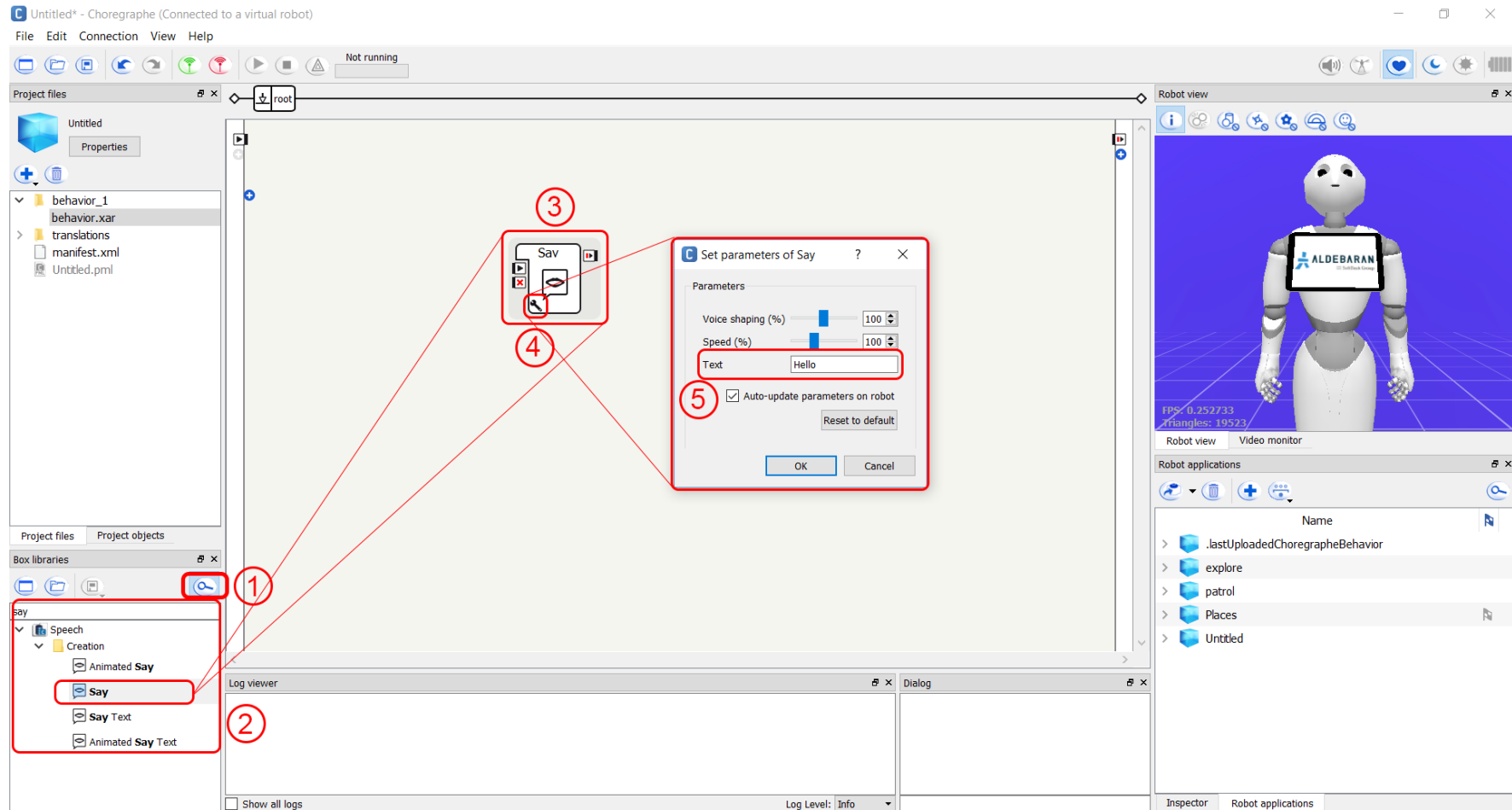


Fig. 6. 20 Choregraphe Speech: Say Customization

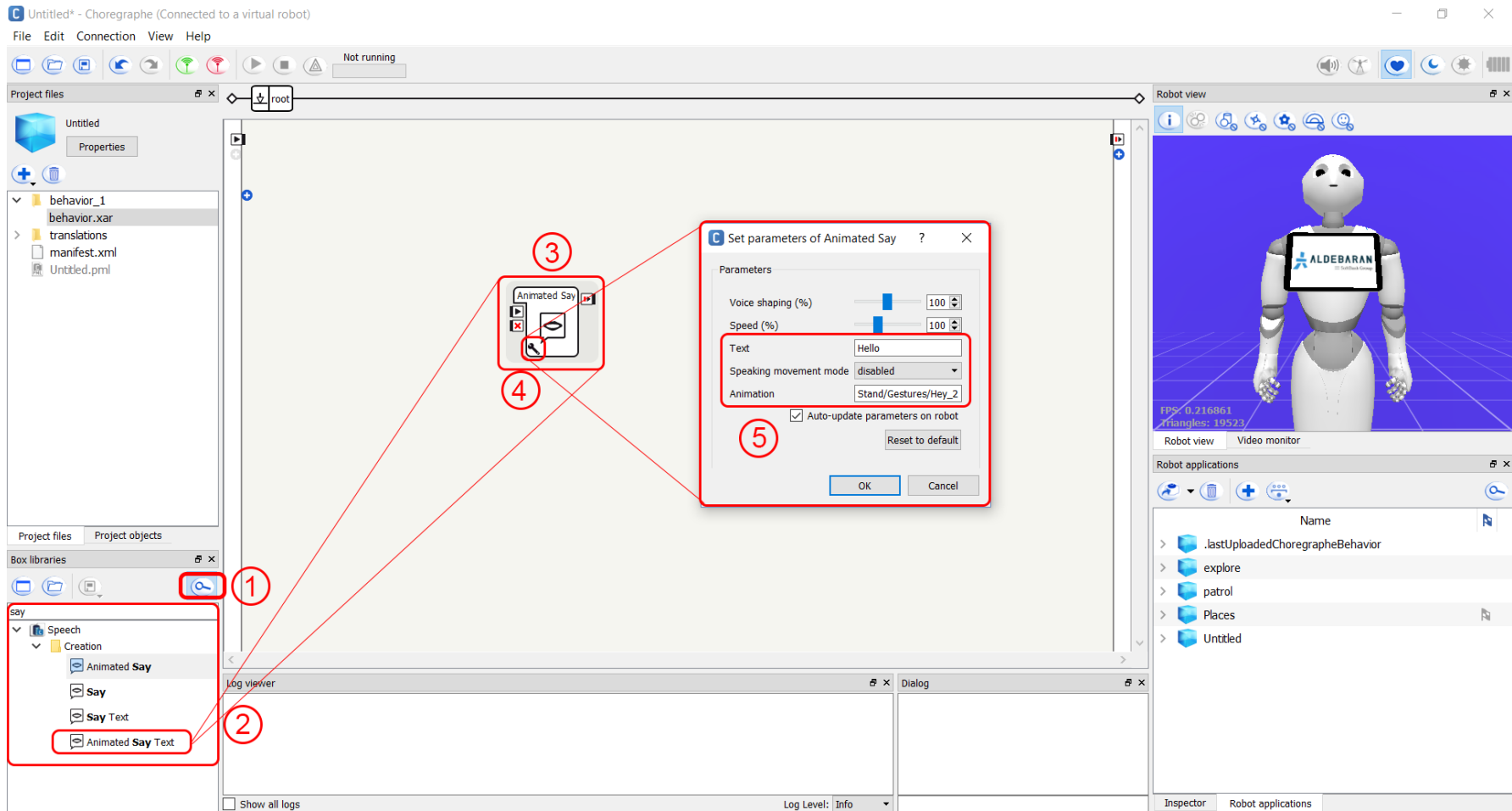




Fig. 6. 21 Choregraphe Speech: Animated Say

6. *Choreographe Set Reco. Lang.* (see Fig. 6. 22)

The screenshot gives a quick step by step overview on how to set up a language recognition.

- 1) Find the *Set Reco. Lang.* command box - Do this either by typing the name or by browsing through the available command boxes.
- 2) Drag & Drop and connect to the start – Connecting the command box to the start ensures that language recognition is started automatically as soon as the behavior starts and that it is active while the behavior is running. If *Set Reco. Lang.* is not connected to the start of the behavior, it can still be run manually by clicking on the left side square .

Set parameters – The box allows you to select what language is to be recognized from a list of available languages.

- 3) The field shows when a behavior is running. The behavior can be Run from the play button . After the start the bar is colored in green and stays green while a behavior is running

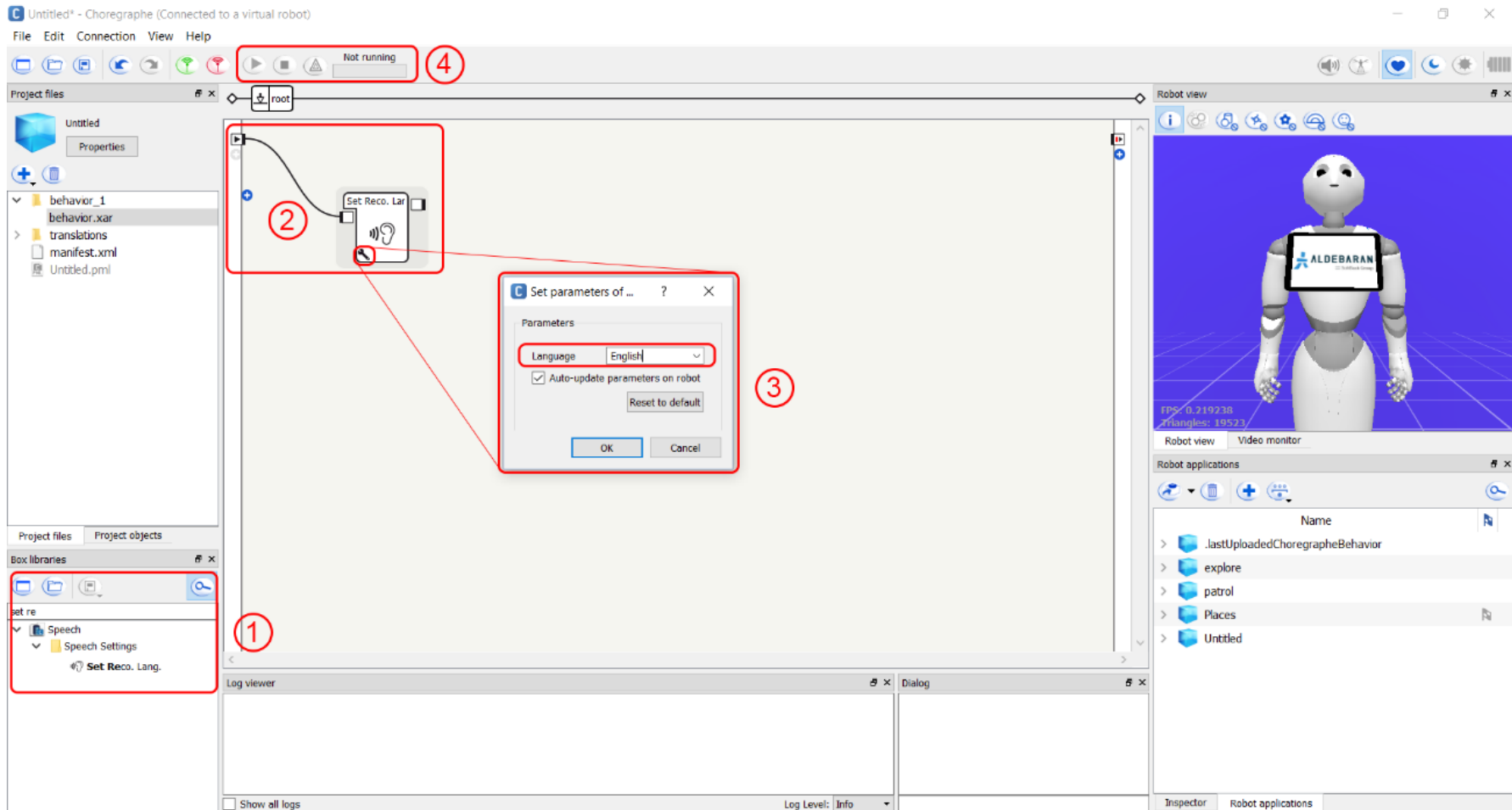


Fig. 6. 22 Choregraphe: Set Reco. Lang.

7. *Choreographe: Choice*

7.1. *Choreographe: Choice in a definition (see Fig. 6. 23)*

The screenshot gives a quick step by step overview on how to set up a *Choice* definition.

- 1) Add *Set Reco. Lang.* and *Choice* to the Work space (see Fig. 6. 15) and connect them to each other.
- 2) Use Ctrl+E or right mouse click on *Choice* in order to open 'Edit Box'.
- 3) From 'Edit box' additional outputs (namely answers to questions) can be added to *Choice* in order to customize it. The new outputs are distinguished by their name and type⁵. It is important to choose the correct type because that would ensure successful connection and data transfer to following command boxes.
- 4) Any new outputs that are added on the right side of *Choice*.

7.2. *Customized Choice (see Fig. 6. 24)*

The *Choice* customization panel/view can be opened by clicking twice on the *Choice* command box. The field 'Localized Text' contains what the robot says. The field 'Choice' contains the optional answers the user must choose from. Each answer has to be connected to an output. An output can be connected to more than one answers.



⁵ A detailed information of the Box Input/ Outputs can be found using this QR code

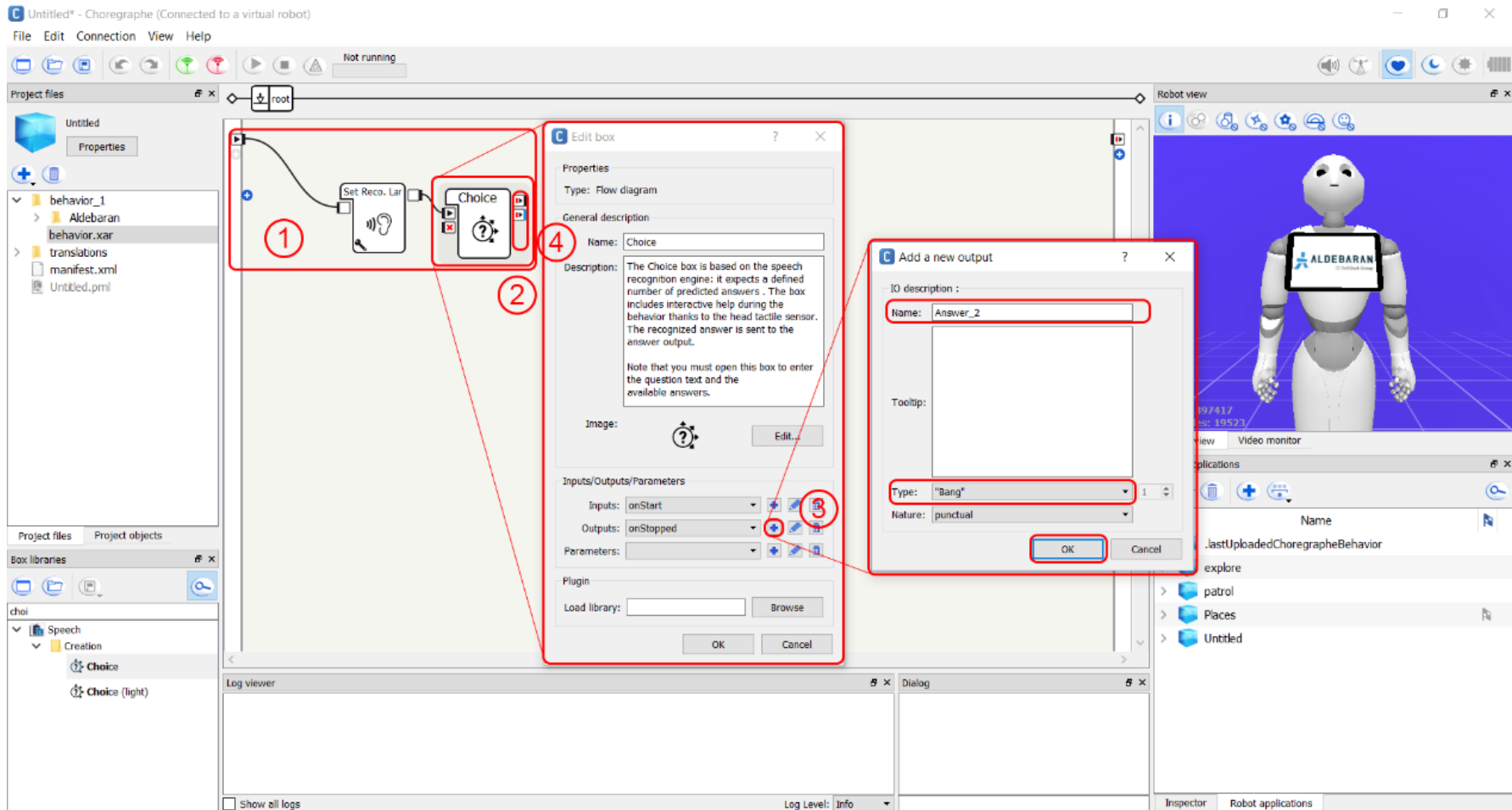


Fig. 6. 23 Choregraphe: Choice in a definition

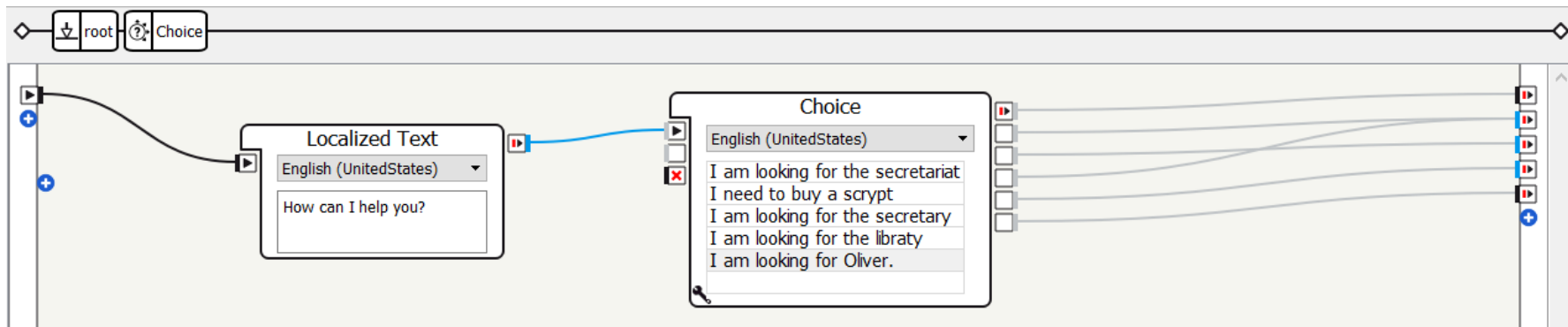


Fig. 6. 24 Customized Choice

8. Adding Choreographe Dialog

8.1. Choreographe: Add a Dialog (see Fig. 6. 25)

Together with Fig. 6. 26 and Fig. 6. 27 this screenshot gives a step by step overview on setting up a *Dialog*.

- 1) Find the *Dialog* command box
- 2) Drag & Drop in the Work space
- 3) A sample folder 'ExampleDialog' is added to the active behavior folder. It contains the files you need in order to script your own *Dialog*. The *.top file in this folder contains a sample *Dialog* script.

8.2. Choreographe: Dialog Customization (see Fig. 6. 26)

After following step 1) related to Fig. 6. 25 Choreographe Add a Dialog, the *Dialog* command box can be customized.

- 2) This frame shows the field containing Dialog command box name. Once you change it there, it will automatically update the command box name as well. It is useful to name your command box after the dialog topic it contains
- 3) The two frames depict how and where to add a dialog topic. The 'Add new dialog topic' window requires a topic name and make sure at least one language is selected.

6. Programming Pepper to be utilized as an Office Concierge in the Department for Architecture Theory and Philosophy of Technics

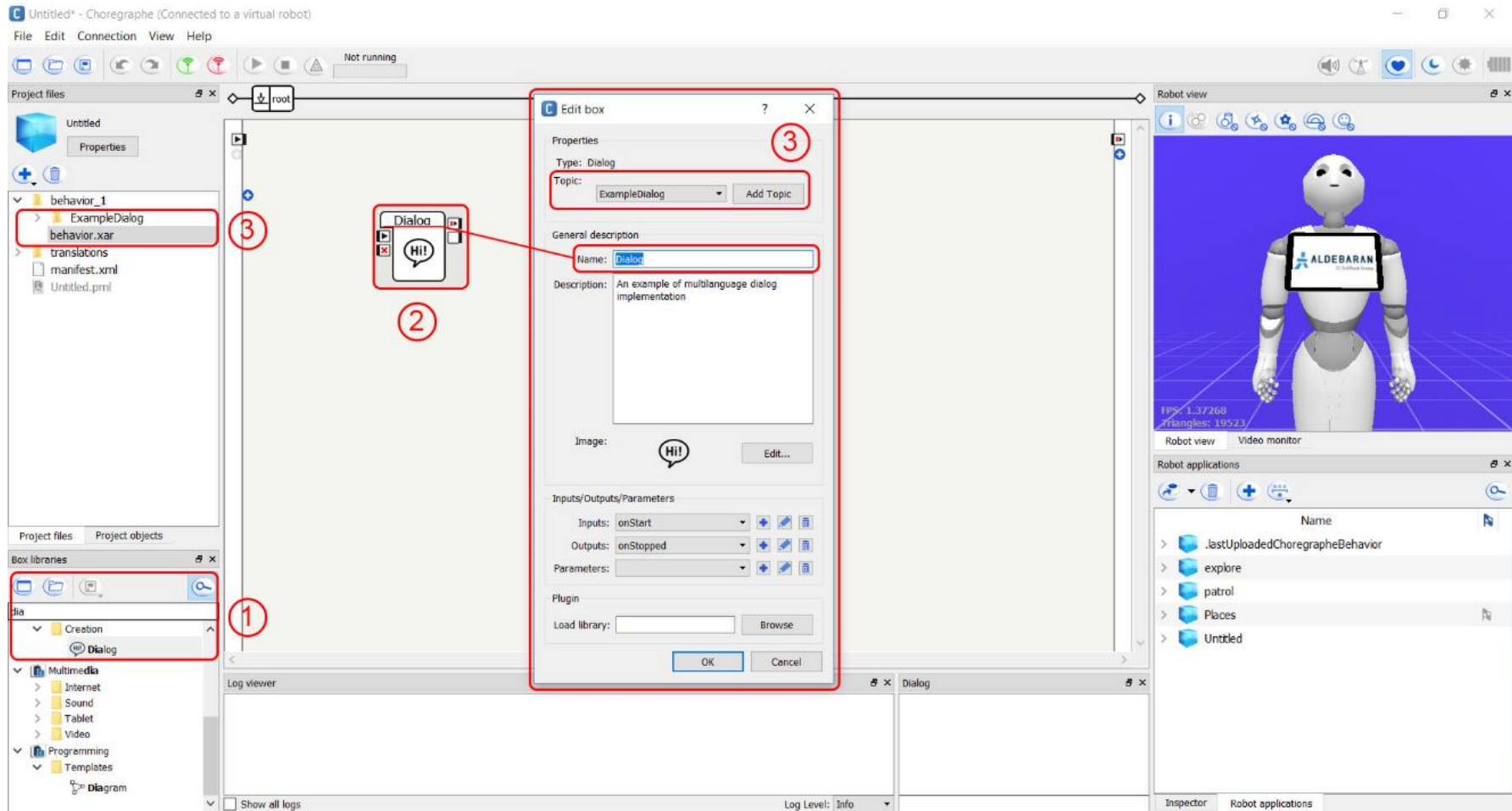


Fig. 6. 25 Choregraphe: Add a Dialog

Robots will refine behavioral space in architecture

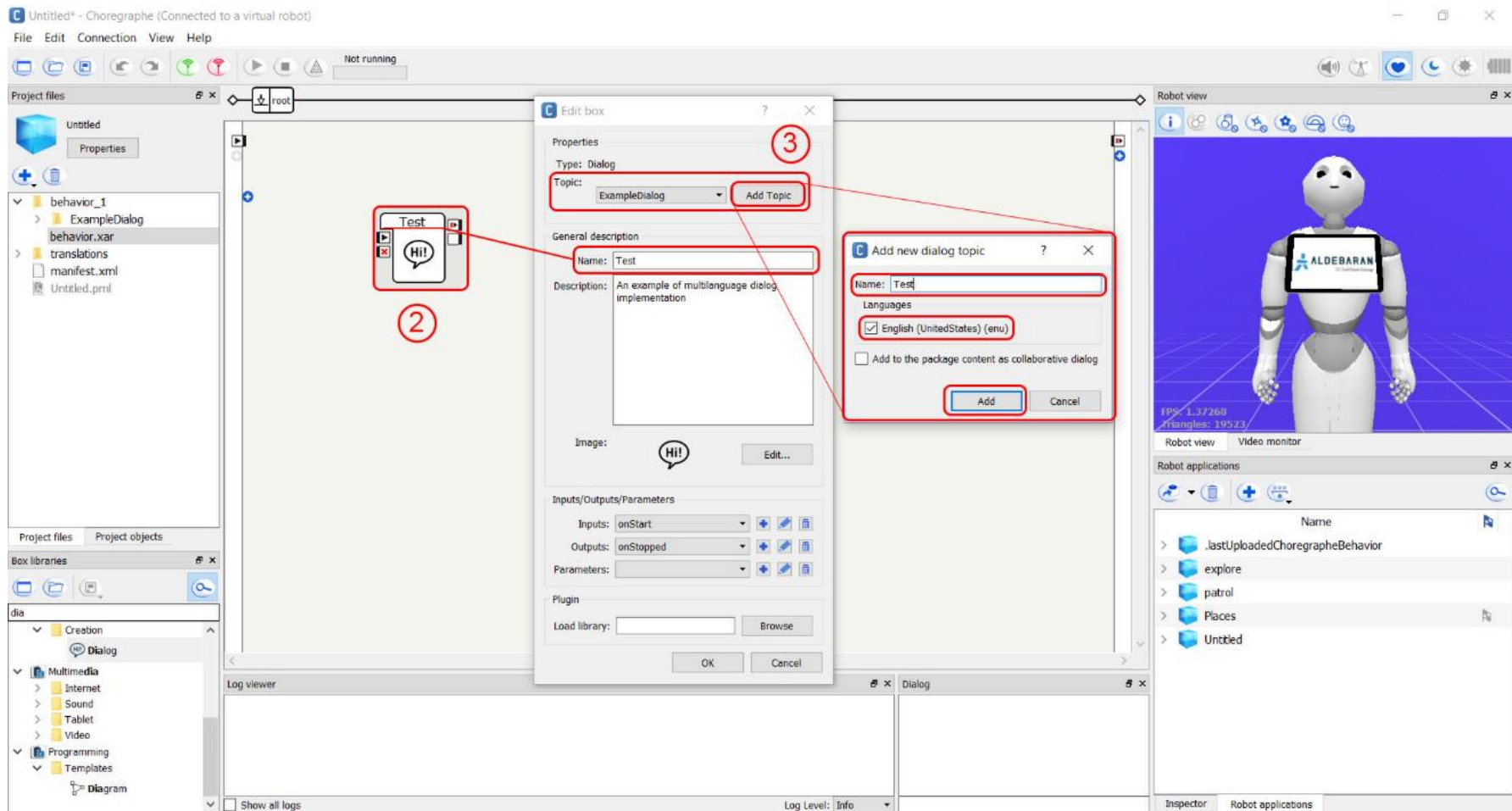


Fig. 6. 26 Choregraphe: Dialog Customization

9. *Choreographe: Dialog Scripting (see Fig. 6. 27)*

In order to start scripting a dialog you need to first complete all steps discussed in Fig. 6. 25 and Fig. 6. 26.

- 1) Adding a new dialog topic automatically sets up a new dialog folder in the behavior. It has the same name as the new topic (in this case Test).
- 2) The folder Test contains a Test_enu.top file. This is the file that contains the dialog script. Clicking on it opens the 'Script editor' window. You can start scripting.
- 3) The Test_enu.top file contains the topic name (topic: ~Test()) and the dialog language (language: enu⁶).

⁶ enu is the abbreviation used for English

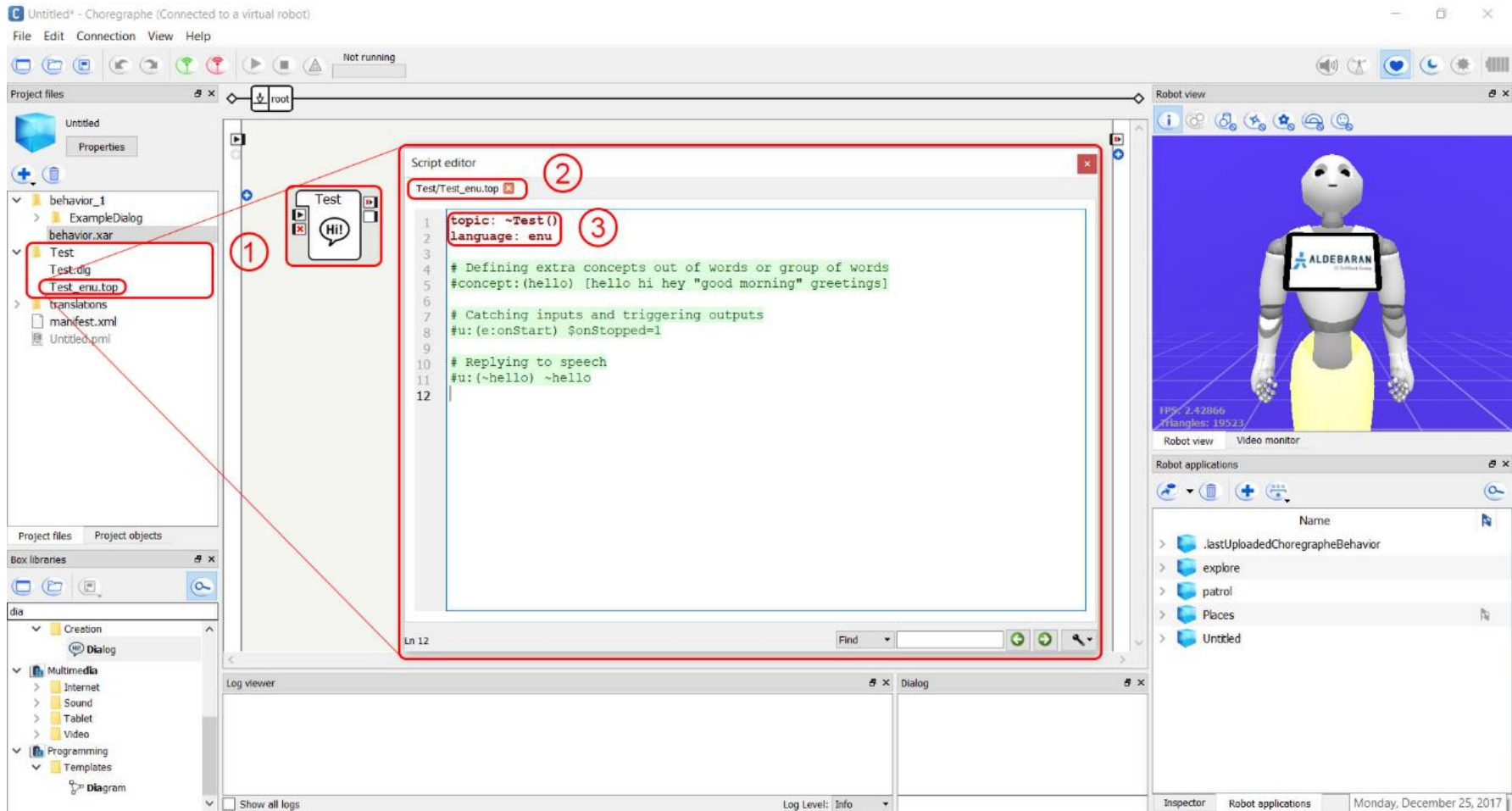


Fig. 6. 27 Choregraphe: Dialog Scripting

In Fig. 6. 28 and Fig. 6. 29 two examples of Dialog scripts are shown. They are compared and discussed in the following paragraphs.

The script shown in Fig. 6. 28 has the simpler syntax out of the two. It is sentence based which makes it result in more errors than the keyword based script shown in Fig. 6. 29. The given sentence based example has no internal hierarchy. This means that all lines are of equal significance and can be triggered at any given time.

In general, Pepper inbuilt speech recognition is word based. This is taken into account in the second example (see Fig. 6. 29). Key words are used in order to improve the robot understanding. The embedded hierarchy boosts the results even further.

For example line 19 in Fig. 6. 29 is:

```
u:({"Where is the"} secretary {"'s office"}?) "Are you looking for the secretary?"
```

This provokes the following algorithm: A human asks where the Department secretary is. The script line 19 is triggered by the word *secretary*. The scripted human input data is `u:({"Where is the"} secretary {"'s office"}?)`. By writing part of the question in brackets, e.g. {...}, we define it as optional. Leaving the word *secretary* outside these brackets defines it as a keyword for the question. This means that even if the human interlocutor uses only the word *secretary* or uses it in differently phrased questions e.g. 'Where is the secretary?', 'I am looking for the secretary?', etc., Pepper registers the word *secretary* and based on it asks for a confirmation if the person is looking for the secretary. It asks '*Are you looking for the secretary?*'.

Script editor

Hello/Hello_enu.top

```

1  topic: ~Hello()
2  language: enu
3
4  concept:(greetings){Hi Hello Hey "Hey there" "Hi there"}
5  concept:(ask_name){"Who are you" "what is your name" "do you hae a name" "do you
know who you are"}
6  concept:(name){"my name is Machina""I am Machina"}
7  concept:(ask_day){"How is your day going?" "How are you?" "How are you doing?" "How
do you do?"}
8  concept:(answer_day){"I'm good""Awesome,thanks" Great}
9  concept:(offer_help){" Can I do something for you?" "Is there something I can do
for you?" "Can I be of service?" "How can I be of service?" "How can I help you?" "Is
there something I can do for you?" "Are you looking for someone?" "Are you looking
for something?"}
10 concept:(secretariat){"i am looking for the secretary""I am looking for the
secretariat""I am looking for the secretary's office""I am looking for the
administration's desk"}
11 concept:(susane){"It is the third door along the corridor"}
12 concept:(script){script"I need a script""I need to buy a script""I nee to get a
script""I am looking for a script"}
13 concept:(library)(library "I am looking for the library?" "where is the library?" "I
wish to get a book from the librar?!" "I need a book from the library?" "Can I get a
book from the library?")
14 concept:(Rafaella){"The library is in the end of the corridor but if you need
something from it you need to ask Rafaella for help"}
15 concept:(robot)[Pepper Pepsi robot Machinal]
16 concept:(sound detection){Hello "is somebody there"}
17 concept:(why){"Why are you not talking""why are not answering me""why are you just
nodding""What is wrong with you"}
18
19 u:(~robot)^rand{Yes"That is me""Yes, i am listening to you""I'm listening"}
20 u:(~are you stupid)"a little bit, yes"
21 u:(~why)^rand{"because I don't understand you""becuse I can't hear you well
enough""it is hard for me to understand you, can you try one more time please"}
22 u:({"turn around""behind you"})^start(turn_around-a675fc/behavior_1)
23 u:(~greetings)^start(my_wave-47f7ba/behavior_1)^rand ~greetings
24 ^rand~offer_help
25 u:(~ask_name)^rand ~name
26 u:(~ask_day)^rand ~answer_day
27 u:(~script)^rand~script{"you can buy it from the secretry ""then you need to go to
te secretary"}
28 u:(~secretariat)~susane
29 u:(~library)~Rafaella
30 u:(Where is the WC,restroom,lavatory,toilet?)It is the first door to your right.
31 u:(talk)What should I say?

```

Fig. 6. 28 Dialog script example: sentence based with no hierarchy.

6. Programing Pepper to be utilized as an Office Concierge in the Department for Architecture Theory and Philosophy of Technics

Script editor

Test/Test_enu.top

```

1  topic: ~Test()
2  language: enu
3
4  concept:(hello) [hello hi hey "good morning" greetings]
5  concept:(yes) [yes "all right" sure "why not" ok certainly "very well" yep yea yeah
   definitely amen]
6  concept:(no) [no nope "don't want" "no way" never "not at all""no need"]
7  concept:(explore) [explore exploration "{Make an} exploration"]
8  concept:(self) [Pepper Pepsi Machina]
9  concept:(answer_self) [yes "I am listening""I'm here"]
10 concept:(restroom) [WC restroom bathroom toilet]
11 concept:(idk) ["not sure""I dont know""I think so""I guess"]
12
13 u:(e:FaceDetected^rand~hello)^start(my_wave-9d7401/
   my_wave)^rand~hello^wait(my_wave-9d7401/my_wave)
14 u:(e:Dialog/NotUnderstood) I do not understand what you are saying
15 u:(e:Dialog/NotUnderstood2) it is the second time that I can't understand what you say
16 u:(e:Dialog/NotUnderstood3) are you sure we speak the same language? $outCheckLanguage=1
17
18 u:(~self)^rand~answer self
19 u:({"Where is the"} secretary {"'s office"}?) "Are you looking for the secretary?" $susane
   = 1
20   u1:(^rand~yes) "Let me show you where it is on my tablet!" $map = 1
21   u1:(^rand~idk) "If you haave some general question regarding organization or need to buy
   a script then this is the person you are looking for.Would you like me to show you where
   her office is?"
22     u2:(^rand~yes) "It is the third door along the corridor.Sekretariat is written on
   the door.Shall I accompany you there?"
23     u3:(^rand~yes) "Please follow me"
24     u3:(^rand~no) "Ok. If you need help with something else, please feel free to
   ask."
25     u2:(^rand~no)
26     u1:(^rand~no) "Are you just looking for her office then?"
27
28 u:({"Where is the""I am looking for the"} library{?})^rand["You are looking for the
   library?""Are you looking for the library?""You want to know where the library is?"]
29   u1:(^rand~yes) "Let me show you where it is on my tablet!" $map = 1 "Shall I accompany
   you there?"
30     u2:(^rand~yes) "Please, follow me"
31     u1:(^rand~no) "ok"
32 u:({"I am looking for""Do you know where"} "Oliver Schurer" Oliver Schurer("is?")) "Are you
   looking for Oliver Schurer?"
33   u1:(^rand~yes) "Have you checked in his office?"
34     u2:(^rand~yes) "You can ask Susane, in the secretary office."
35     u1:(^rand~no) "ok"
36 u:({"I am looking for""Is there"} ^rand~restroom {here?})^rand["You are looking for the
   ^rand~restroom ?""Are you looking for the ^rand~restroom ?""You want to know where the
   ^rand~restroom is?""You need the ^rand~restroom ?" ]
37 u:(explore)I will explore the space ^start(exploration-c8782b/all)
38 u:(place)let me show you the map ^start(exploration-manager/display_places)
39 u:(patrol)Where would you like to go? ^switchFocus(PatrolApplication/PatrolBehavior)
40 # Defining extra concepts out of words or group of words
41 #concept:(hello) [hello hi hey "good morning" greetings]
42
43 # Catching inputs and triggering outputs
44 #u:(e:onStart) $onStopped=1
45
46 # Replying to speech
47 #u:(~hello) ~hello
48

```

Fig. 6. 29 Dialog script example: key words based with hierarchy.

7. Programing Pepper Navigation

7.1. Exploration

Exploration is an application that allows Pepper to 'explore' its surroundings in a 'user defined' radius.

There are several ways in which the exploration can be executed, for example by installing an explore application on the physical robot and running it using Choreographe, by using Python 2.7⁷ or by using ROS⁸.

In the following paragraphs, executing exploration with Choreographe and executing it by running a Python script using IDLE (Window)/Terminal (Mac)⁹ for Python 2.7 will be compared. The final results are the same in both cases. However, one method or the other one can be more beneficial in a given situation, based on the foreseen use of the exploration results. For example, a Python script is the better option for big spaces¹⁰ and for people with Python experience. It is a single script that can be customized and/or combined with other scripts. The explore application installed on Pepper and run using Choreographe is the better option for small spaces¹¹ and when the designer has limited scripting experience.

The exploration application consists of a five-script definition (see Fig. 7. 3). It requires no to little scripting experience. It can be customized but it is not obligatory.

⁷ Python 2.7 – software used for executing Python scripts written with the Python 2.7 syntax.

⁸ ROS (Robot Operating System) provides libraries and tools to help software developers create robot applications. It provides hardware abstraction, device drivers, libraries, visualizers, message-passing, package management, and more.. (HoangGiang88, 2017)

⁹ Development Environment for Python scripts. IDLE is Python's built-in Integrated Development Environment, which is installed by default upon installation of Python. Terminal is OS X's standard console used as Python development environment.

¹⁰ big space – in this context, as big are regarded spaces for which the needed exploration radius is greater than 10 m.

¹¹ small space – in this context, as small are regarded spaces for which an exploration radius of 10 m is enough.


```

#!/usr/bin/env python
# -*- encoding: UTF-8 -*-

"""Example: Use explore method."""

import qi
import argparse
import sys
import numpy
import Image

def main(session):
    """
    This example uses the explore method.
    """
    # Get the services ALNavigation and ALMotion.
    navigation_service = session.service("ALNavigation")
    motion_service = session.service("ALMotion")

    # Wake up robot
    motion_service.wakeUp()

    # Explore the environment, in a radius of 2 m.
    radius = 2.0
    error_code = navigation_service.explore(radius)
    if error_code != 0:
        print "Exploration failed."
        return
    # Saves the exploration on disk
    path = navigation_service.saveExploration()
    print "Exploration saved at path: \" + path + "\""
    # Start localization to navigate in map
    navigation_service.startLocalization()
    # Come back to initial position
    navigation_service.navigateToInMap([0., 0., 0.])
    # Stop localization
    navigation_service.stopLocalization()
    # Retrieve and display the map built by the robot
    result_map = navigation_service.getMetricalMap()
    map_width = result_map[1]
    map_height = result_map[2]
    img = numpy.array(result_map[4]).reshape(map_width, map_height)
    img = (100 - img) * 2.55 # from 0..100 to 255..0
    img = numpy.array(img, numpy.uint8)
    Image.frombuffer('L', (map_width, map_height), img, 'raw', 'L', 0, 1).show()

if __name__ == "__main__":
    parser = argparse.ArgumentParser()
    parser.add_argument("--ip", type=str, default="127.0.0.1",
                        help="Robot IP address. On robot or Local Naoqi: use 127.0.0.1.")
    parser.add_argument("--port", type=int, default=9559,
                        help="Naoqi port number")

    args = parser.parse_args()
    session = qi.Session()
    try:
        session.connect("tcp://" + args.ip + ":" + str(args.port))
    except RuntimeError:
        print ("Can't connect to Naoqi at ip \" + args.ip + "\" on port " + str(args.port) + ".\n"
              "Please check your script arguments. Run with -h option for help.")
        sys.exit(1)
    main(session)

```

Fig. 7. 1 The script of the executable file 'explore.py'

Running a Python script (see Fig. 7. 1)

A basic Python script for exploration called 'explore.py' (see Fig. 7. 2)¹² is available online. It can be run using IDLE or Terminal based on the OS (which needs to be installed following the Aldebaran instructions, see Fig. 7. 3). To ensure proper operation a path to Pepper needs to be set.



Fig. 7. 2 QR Code 7: The script of the executable file explore.py'



Fig. 7. 3 QR Code 8: Python SDK Installation Guide

The second essential step before utilizing the script is to ensure that all required libraries/packages (i.e. numpy, Image, etc.), which are to be imported and used are already installed and functioning

Finally, the script can be customized. The most important sections of the given sample script are framed in red (see Fig. 7. 1).

- 1) The value 9559 is the default port value. It is always needed. All Aldebaran Robots have the same port value, equal to 9559. Do not change it!

¹² explore.py – Python script file containing the script for exploration

- 2) The IP address – it is important that it is the same as the one of the robot currently being used.
- 3) Radius value – in this script the exploration radius is given as a parameter, so that its value can easily be changed.

1. Using Choreographe

For this project I have chosen to use the explore application executed using Choreographe. For this I use the following Navigation bundle is used (see Fig. 7. 4).



Fig. 7. 4 QR Code 9: Navigation bundle for Choreographe, (aldebaran, 2016)

After starting Choreographe and connecting it to the physical robot (Fig. 6. 16 and Fig. 6. 17) the explore.pml¹³ file needs to be opened. Then, it should be run once in order to ensure it is working properly. After that the application needs to be installed on the robot. This way it can be used in different projects and it can be incorporated in them. The application makes the robot explore autonomously its

¹³ explore.pml – a Choreographe file containing the definition for the explore application

surroundings in a 10 m radius. Hence, it is not suitable for bigger spaces without being modified.

Despite the above stated shortcoming, this approach has been selected for this project because it fits with the set goals. The project is developed using the Choreographe environment because this is a more user-friendly option for people who have less experience in scripting and in using Python 2.7.

While working in Choreographe, the user doesn't need to customize the scripts, explicitly provide a port value or IP data. This is automatically delivered through the connection between Choreographe and the robot. It minimizes the chance for errors occurring due to customization. Furthermore, as the application is installed on the machine it is easy to combine with other functions. For example it can be triggered by a vocal command in a *Dialog*. In sub-chapter 7.3 it is shown step by step how to install the explore application on the robot.

7.2. Places and Patrol

As I have decided to go with Choreographe, the applications *explore*, *places* and *patrol* had been installed. They are used to navigate the robot in the physical environment.

All three application have been installed following the same steps. The screenshots presenting the sequence of installation steps of the 'explore' application are shown in Fig. 7. 5, Fig. 7. 6, Fig. 7. 7, Fig. 7. 8 and Fig. 7. 9. The three applications have to be installed in the given below order:

1. Exploration;
2. Places
3. Patrol.

When the exploration is run multiple times it is likely to show similar, yet, slightly different results. This is due to the precision in the readings from Pepper sensors.

The application '*places*' can be run only after '*explore*' has been executed and saved. The visual panel of '*places*' allows the user to pin – point a location or a place on the Pepper generated gray-scale plan and to name and save the locations/places(e.g library, secretary's office,..etc.). Only after that, the '*patrol*' application can be run. When patrolling Pepper moves from its present location to a selected by the user point on the space gray-scale plan.


7.3. How to Install an Application on a Physical Robot using Choreographe

Download the bundle and extract it into a folder on your computer. The new folder should contain the three empty file folders named 'explore', 'places' and 'patrol' as shown in

1. Fig. 7. 5. Each of them contains one of the applications used for robot navigation.

Open the first folder, named 'explore' and start explore.pml (

- 1) Fig. 7. 6). The definition of the '*explore*' application is contained in *explore.pml*.

- 2) Make sure you are connected to the physical robot. It is not important whether you will connect to the robot before or after starting the file *explore.pml* but you need to be connected to it before installing the application.
2. Make sure the *.pml file works properly by running the definition it contains. This will ensure the correct installation of an application on the robot. (Fig. 7. 9).
 - 1) The behavior is running
 - 2) Text will appear in the '*Dialog window*' as Pepper executes the behavior
 3. Install the application on your physical robot.
 - 1) Click on the  button to start the installation.
 - 2) A message will appear in the '*Log Viewer*' upon the completion of the installation.
 - 3) The application will appear in the '*Robot Applications*' menu. On this robot the applications '*places*' and '*patrol*' are already installed as well. .

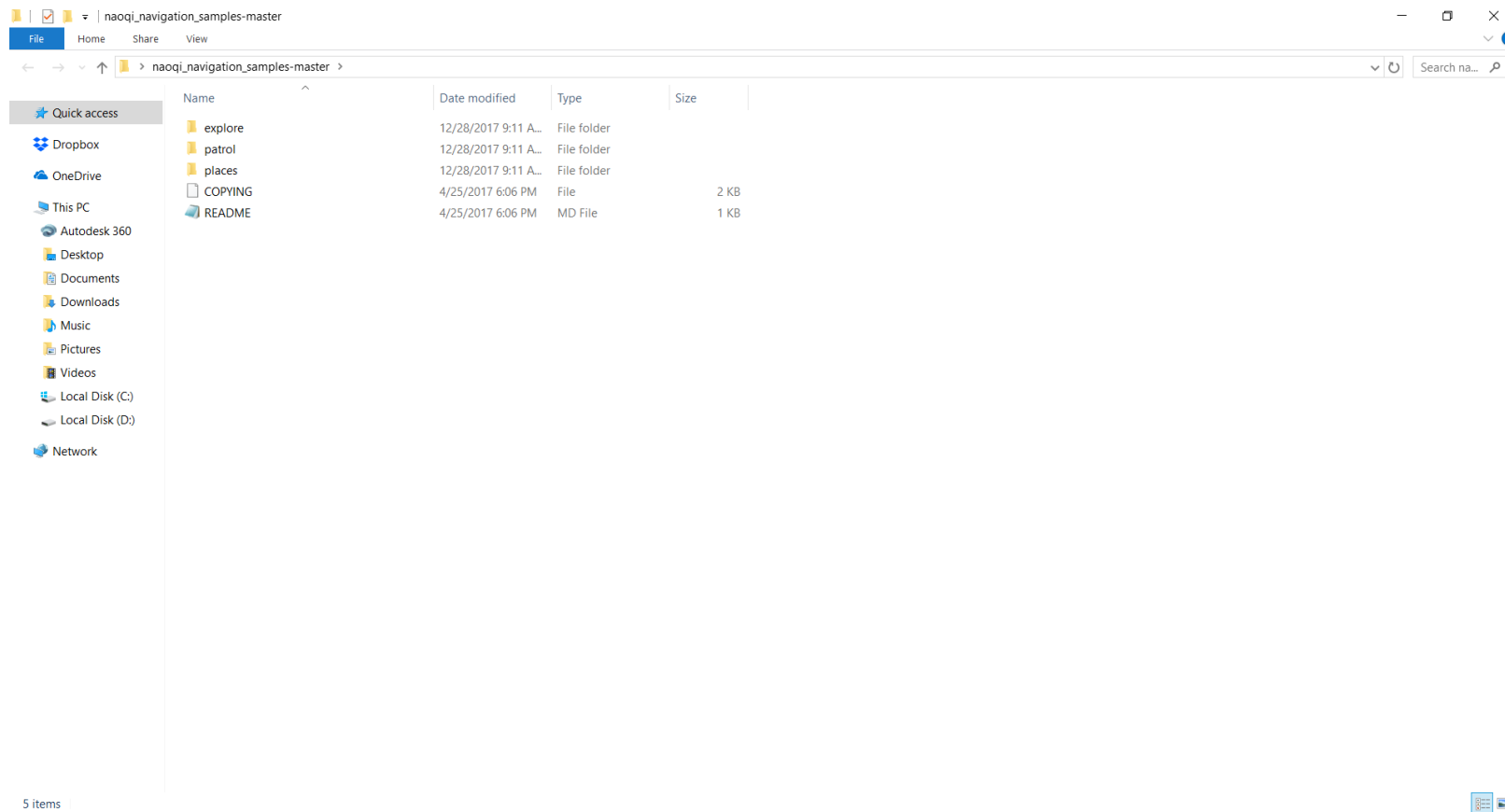


Fig. 7. 5 Bundle folder

Robots will refine behavioral space in architecture

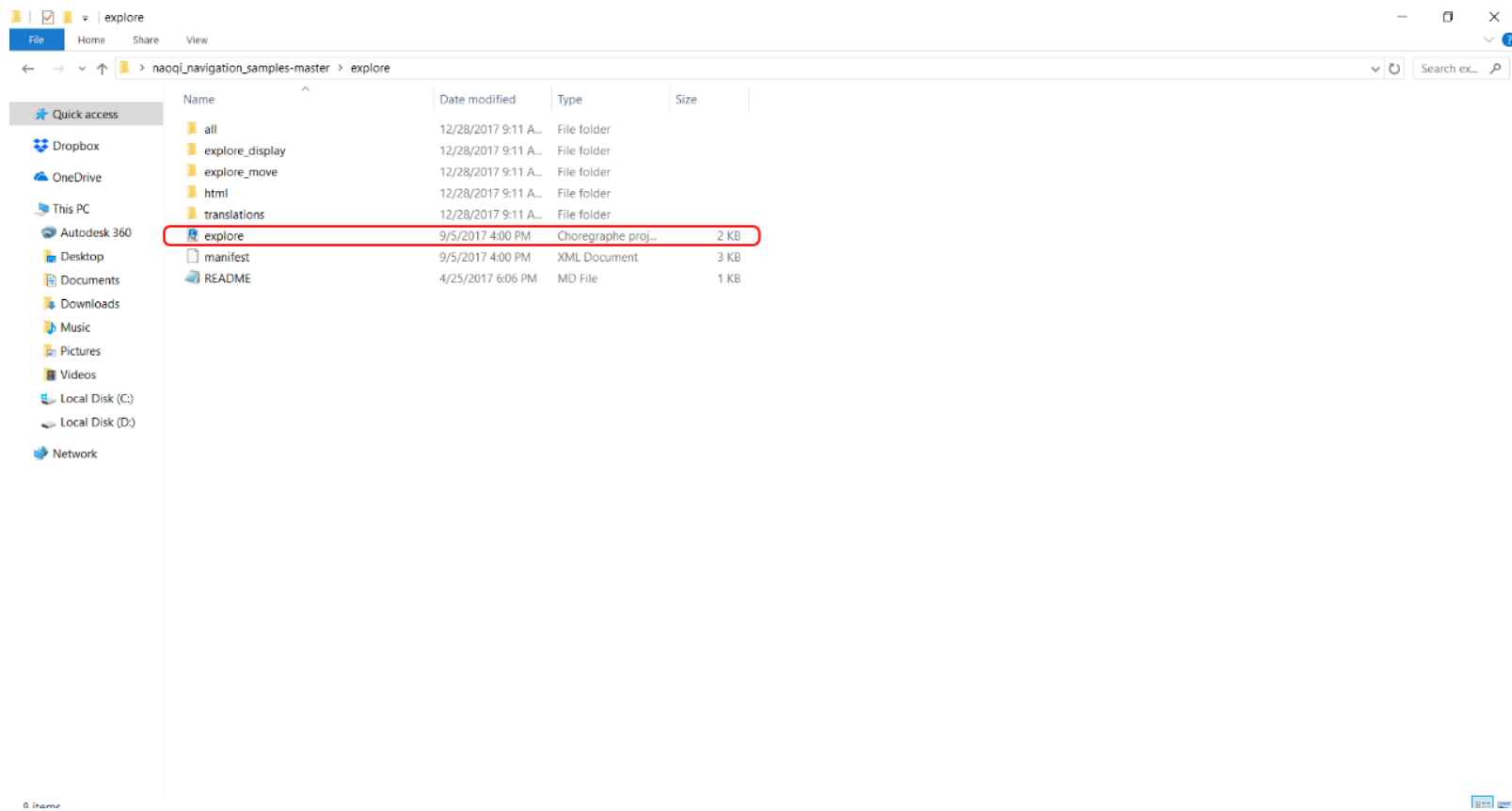


Fig. 7. 6 explore.pml

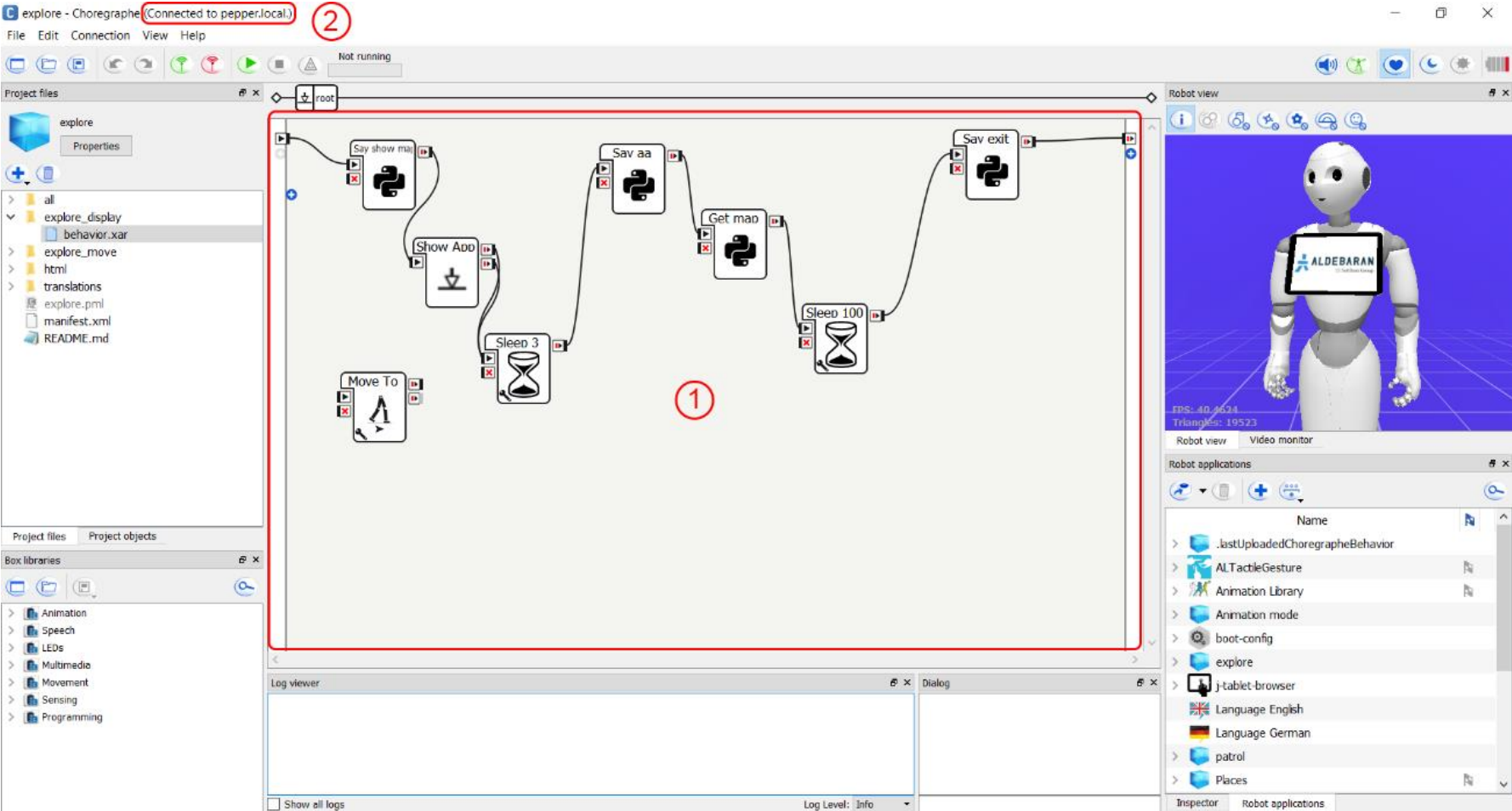


Fig. 7. 7 Choregraphe: explore.pml

Robots will refine behavioral space in architecture

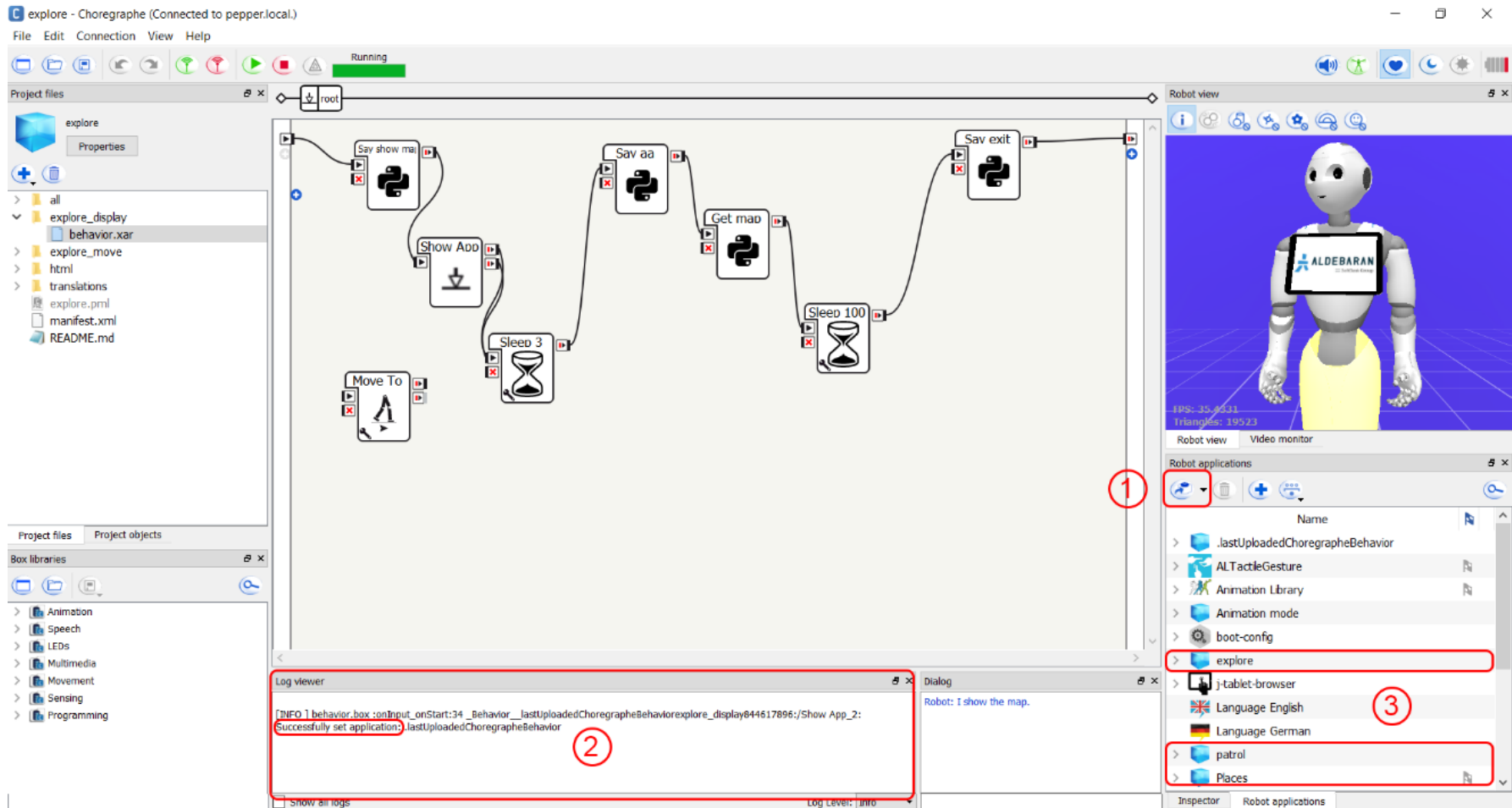


Fig. 7. 8 Choregraphe: Install an application on a physical robot

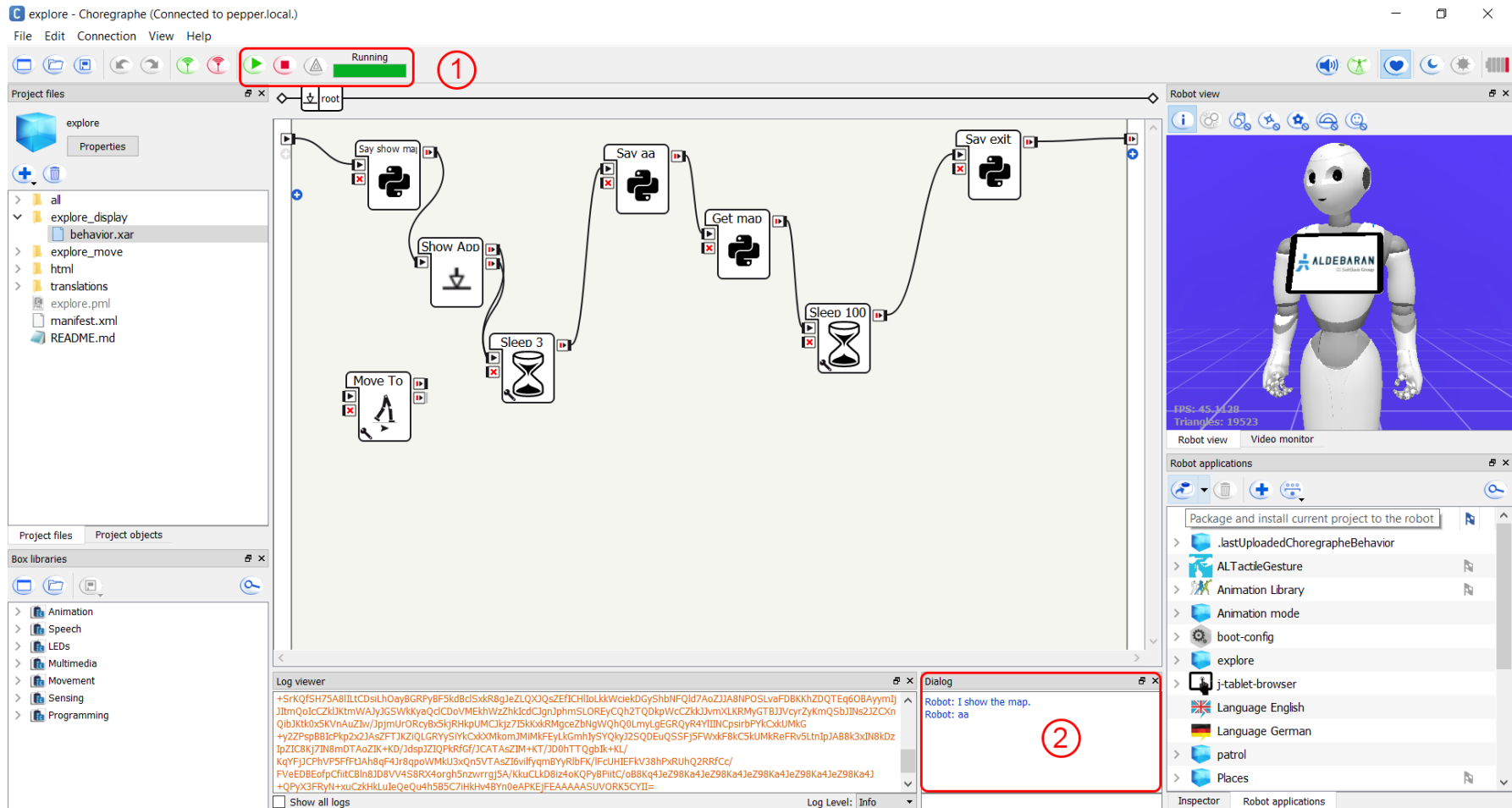


Fig. 7. 9 Choregraphe: run explore.pml behavior

8. Conclusions and Perspective for Future Work

8.1. Summary of Work Results and Achievements

The work investigates the introduction of humanoid robots to the architectural environment. It proposes an approach for transforming the humanoid robot Pepper from an object in an office space into a mobile interactive architectural element which provides directions and information to its users.

The work merges immobile and mobile building elements into a single efficiently operated synergic environment. It uses robot behaviors to design an efficient and beneficial for people interaction in space. This improves the office architecture functionality in different aspects. The humanoid robot takes over routine tasks, i.e. guides newcomers, provides information or answers to interlocutors' question in the context of the office environment of the Department for Architecture Theory and Philosophy of Technics. Pepper relieves employees from some routine and boring tasks while allowing them to concentrate on more productive, creative and challenging tasks. Thus, as a whole the architectural space becomes more functional and its inhabitants - more productive. At the same time, the humanoid robot Pepper successfully acts as a mobile interactive architecture element and people benefit from it.

Assigning concierge functions to a humanoid robot is among the routine scenarios for its utilization. This makes it not only an active architectural element but also a tool that can influence the team spirit and the economic efficiency in a long-term prospective.

Following the developed for this diploma thesis guidelines for programming Pepper to fulfil concierge tasks does not require any specific IT or software expertise. This

considerably broadens the scope of its applications, including future application in bigger office or public spaces. Hence, Pepper can easily be adapted to serve any physical space and to fulfil a majority of routine everyday tasks.

8.2. Issues to be explored in Future

This diploma thesis sets the beginning of a series of future works that will focus of studying Pepper applications on the Department for Architecture Theory and Philosophy of Technics. The additional functions that can be included in scenarios, in which Pepper is utilized, are not limited to the ones proposed bellow. Considering Pepper technical parameters the future works can go in the following directions:

- connecting the Pepper to the existing university platform TISS, so the robot can provide information about the schedule for the courses being thought, and to the online calendar of department employees;
- providing Pepper with access to the Department library database. Thus, the robot can give information on volume availability and location;
- programing Pepper face recognition for several of the employees (let's say 5) who use it most often.

All these scenarios will significantly improve its applicability and flexibility in serving the Department employees.

The thesis can serve as a foundation for implementing Pepper into bigger office spaces and public spaces.

8.3. Conclusion

The presented in this work state-of-the-art convincingly proves that a humanoid robot is more acceptable than any other modern technology to the human users. The humanoid robot is easier to communicate with and controlling it requires less time to master.

In terms of this work the robot serves as an office concierge taking over routine tasks in order to relieve qualified staff from them. Based on this thesis I conclude that it is possible and practical to utilize humanoid robots as architectural elements. They can fulfill some of the functions of an office concierge such as providing directions and information to the office visitors and employees. As technologies advance people will benefit from humanoid robots being able to take on more complex tasks. The work demonstrates in a user-friendly and adaptable for architects and planners form how a humanoid robot Pepper can be implemented in an office environment.

REFERENCES

Aldebaran/SoftBank, n.d. *Aldebaran/SoftBank: Reflexes External Collision*. [Online]

Available at: <http://doc.aldebaran.com/2-5/naoqi/motion/reflexes-external-collision.html>

[Accessed 17.12.2017].

aldebaran, 2016. *Github*. [Online]

Available at: https://github.com/aldebaran/naoqi_navigation_samples

[Accessed 17.08.2017].

Breitling, D.-I. d. S., 2003. *Baugeschichte TU Berlin*. [Online]

Available at: [http://baugeschichte.a.tu-](http://baugeschichte.a.tu-berlin.de/bg/lehre/veranstaltung_dokumentation.php?det_id=154&veranst_id=48)

[berlin.de/bg/lehre/veranstaltung_dokumentation.php?det_id=154&veranst_id=48](http://baugeschichte.a.tu-berlin.de/bg/lehre/veranstaltung_dokumentation.php?det_id=154&veranst_id=48)

[&veranstaltung=vorlesung&semester=](http://baugeschichte.a.tu-berlin.de/bg/lehre/veranstaltung_dokumentation.php?det_id=154&veranst_id=48)

[Accessed 11.12.2017].

Donati, S., 2016. *Italy Magazine*. [Online]

Available at: <http://www.italymagazine.com/news/newly-opened-torre-campatelli-offers-depth-look-history-san-gimignano>

[Accessed 5.12.2017].

Família, F. J. C. d. T. E. d. I. S., n.d. *Sagrada Familia : History and Architecture*. [Online]

Available at: <http://www.sagradafamilia.org/en/geometry/>

[Accessed 11.12.2017].

Ghost in the Shell. 2017. [Film] Directed by Rupert Sanders. USA: DreamWorks Pictures, Reliance Entertainment, Arad Productions.

Gottscho-Schleisner, I., 12 November 1957. *Wikipedia*. [Online]

Available at:

https://upload.wikimedia.org/wikipedia/commons/3/3e/Guggenheim_Museum_construction_LOC_gsc.5a25494.jpg

[Accessed 11.12.2017].

Hanson, D., 2011. *IEEE Spectrum*. [Online]

Available at: <https://spectrum.ieee.org/automaton/robotics/humanoids/why-we-should-build-humanlike-robots>

[Accessed 11.12.2017].

Hashimoto, S., 2004. Humanoid robot for Kansei communication - computer must have body . In: J. V. ,. K. H. P Sinčák, ed. *Machine Intelligence: Quo Vadis?*. Singapore: World Scientific Pub Co Inc, pp. 357-370.

HEIRlab, M., 2015. *YouTube*. [Online]

Available at:

https://www.youtube.com/watch?v=VDj4sILw0jM&list=PLqTE73vFeu2oOFiqoY_b

DdK7utlT7uxx0&index=6

[Accessed 10.08.2017].

HEIRlab, M., 2015. *YouTube*. [Online]

Available at:

https://www.youtube.com/watch?v=2chk74mCDms&index=5&list=PLqTE73vFeu2oOFiqoY_bDdK7utlT7uxx0

[Accessed 10.08.2017].

HEIRlab, M., 2015. *YouTube*. [Online]

Available at:

https://www.youtube.com/watch?v=xj9Pal3WKYQ&list=PLqTE73vFeu2oOFiqoY_bDdK7utlT7uxx0&index=4

[Accessed 10.08.2017].

HoangGiang88, 2017. *Wiki ROS*. [Online]

Available at: <http://wiki.ros.org/>

[Accessed 20.12.2017].

iRobot, 2014. *www.auvsi.org*. [Online]

Available at: <http://www.auvsi.org/irobot-deliver-20-hazmat-packbots-canada-0>

[Accessed 27.12.2017].

Jordan, J. M., 2016. *Robots*. Cambridge(Massachusetts): The MIT Press.

Kundera, M., 1997. *Slowness: A Novel*. New York: Harper Perennial.

Liebeskind, D., 2013. *a huge scroll depicting the masterplan for the Ground Zero site in New York*. s.l.:<https://www.dezeen.com/2013/02/28/architectural-drawings-by-daniel-libeskind-at-ermanno-teseschi-gallery/>.

Ltd., S., Original June 2 2016. *Top Websites May 2017 All Traffic*. [Online]

Available at:

<https://pro.similarweb.com/#/industry/topsites/All/999/1m?webSource=Total>

[Accessed 30.06.2017].

Mauricio Mondragon, L. L., 2012. *Space and Time as Containers of the "Physical Material World" with some Conceptual and Epistemological Consequences in Modern Physics*. s.l.:s.n.

Meiss, P. v., 1990. *Elements of Architecture, from form to place*. 1st ed. London & New York: E.F.Spon.

Mori, M., 2012. The Uncanny Valley. *IEEE Robotics & Automation Magazine* , 06 June, pp. 98 - 100.

Ng, A., 2013. *BlockUrbanism*. [Online]

Available at: <http://www.blockurbanism.andrew-ng.com/nolli-rome/>

[Accessed 15.12.2017].

Onniboni, L., 2016. *Archiobjects*. [Online]

Available at: <https://archiobjects.org/wp-content/uploads/2014/07/Guggenheim->

Museum-New-York-FLW.jpg

[Accessed 11.12.2017].

Pallasmaa, J., 1994. Six Themes for the next Millenium. *The Architectural Review*, 1 July, pp. 74-79.

Petrova, V. D., 2017. *Humanoid Robot Pepper on Charliroi Airport*. [Art] (Personal Archive).

Petrova, V. D., 2017. *Pepper in a DSK bank office, Serdika mall, Sofia, Bulgaria*. [Art] (Personal Archive).

Press, C. U., n.d. *Cambridge Online Dictionary*. [Online]

Available at: <https://dictionary.cambridge.org/dictionary/english/space>

[Accessed 2017].

Reogers, Y., 2011. *Interaction Design : beyond human - computer interaction*. 3rd ed. s.l.:John Wiley & Sons.

Robotica, A., n.d. *Aldebaran Pepper Languages*. [Online]

Available at: http://doc.aldebaran.com/2-4/_images/languages_pep.png

[Accessed 15.12.2017].

Robotics, A., n.d. *Aldebaran Choreographe Overview*. [Online]

Available at: [http://doc.aldebaran.com/2-](http://doc.aldebaran.com/2-1/software/choreographe/choreographe_overview.html)

[1/software/choreographe/choreographe_overview.html](http://doc.aldebaran.com/2-1/software/choreographe/choreographe_overview.html)

[Accessed 15.08.2017].

Robotics, A., n.d. *Aldebaran Dialog Cheat Sheet*. [Online]

Available at: [http://doc.aldebaran.com/2-](http://doc.aldebaran.com/2-4/naoqi/interaction/dialog/aldialog_syntax_cheat_sheet.html#dialog-cheat-sheet)

[4/naoqi/interaction/dialog/aldialog_syntax_cheat_sheet.html#dialog-cheat-sheet](http://doc.aldebaran.com/2-4/naoqi/interaction/dialog/aldialog_syntax_cheat_sheet.html#dialog-cheat-sheet)

[Accessed 20.12.2017].

Robotics, A., n.d. *Aldebaran Dialog Syntax*. [Online]

Available at: [http://doc.aldebaran.com/2-4/naoqi/interaction/dialog/dialog-](http://doc.aldebaran.com/2-4/naoqi/interaction/dialog/dialog-syntax_full.html#dialog-rules)

[syntax_full.html#dialog-rules](http://doc.aldebaran.com/2-4/naoqi/interaction/dialog/dialog-syntax_full.html#dialog-rules)

[Accessed 20.12.2017].

Robotics, S., 2017. *SoftBank Robotics*. [Online]

Available at: <https://www.ald.softbankrobotics.com/en/robots>

[Accessed 11.12.2017].

Robotics, S., n.d. *SoftBank Robotics*. [Online]

Available at: <https://www.ald.softbankrobotics.com/en/robots/pepper>

[Accessed 18.12.2017].

Shelley, M., 1818. *Frankenstein*. 1st ed. London: Lackington, Hughes, Harding, Mavor & Jones.

Shirow, M., 1989. *Kōkaku Kidōtai (Ghost in the Shell)*. s.l.:Kodansha.

Smith, N. R., 2017. *USA Today*. [Online]

Available at: <https://www.usatoday.com/videos/tech/2017/11/03/stephen->

[hawking-robots-replace-humans-completely/107294198/](#)

[Accessed 10.12.2017].

SoftBanks, A., 2017. *Aldebaran ALNaviation*. [Online]

Available at: <http://doc.aldebaran.com/2-5/naoqi/motion/alnavigation.html>

[Accessed 15.12.2017].

studios, z., n.d. *Shutterstock*. [Online]

Available at: <https://ak3.picdn.net/shutterstock/videos/9274148/thumb/11.jpg>

[Accessed 5.12.2017].

SULLEYMAN, A., 2017. *Stephen Hawking warns "Artificial Intelligence may replace Humans altogether"*. [Online]

Available at: <http://www.independent.co.uk/life-style/gadgets-and-tech/news/stephen-hawking-artificial-intelligence-fears-ai-will-replace-humans-virus-life-a8034341.html>

[Accessed 10.12.2017].

Takanishi, A., 2005. *Cultural Differences in the Perception*. Barcelona, ICRA 2005, IEEE International Conference on Robotics and Automation.

Tice, J., Steiner, E. & Oregon, 2.-2. U. o., n.d. *The Nolli Map Website*. [Online]

Available at: <http://nolli.uoregon.edu/nuovaPianta.html>

Vitruvius, n.d. *De architectura*. s.l.:s.n.

Weiser, M., 1991. The Computer for the 21st Century. *Scientific American*, Issue Communications, Computers, and Networks, pp. 94-104.

Wikipedia, 2017. *Wikipedia*. [Online]

Available at: <https://en.wikipedia.org/wiki/Architecture>

.

TABLE OF FIGURES

FIG. 1. 1 INTERACTION BETWEEN TASKS AND INVOLVED PLAYERS	5
FIG. 1. 2. INTERACTION BETWEEN ARCHITECTURE, SOFTWARE ARCHITECTURE, ROBOTS AND HUMANS.....	6
FIG. 2. 1 ARCHITECTURE GUGGENHEIM MUSEUM, NEW YORK CITY, F. L. WRIGHT: CONCEPTUAL IDEA AND BUILDING DESIGN (BREITLING, 2003)	9
FIG. 2. 2 ARCHITECTURE GUGGENHEIM MUSEUM, NEW YORK CITY, F. L. WRIGHT: CONSTRUCTION PROCESS (GOTTSCHO-SCHLEISNER, 12 NOVEMBER 1957).....	9
FIG. 2. 3 ARCHITECTURE GUGGENHEIM MUSEUM, NEW YORK CITY, F. L. WRIGHT (ONNIBONI, 2016).....	9
FIG. 2. 4 VITRUVIUS' PRINCIPLES OF ARCHITECTURE	11
FIG. 2. 5 SAN GIMIGNANO - A CITY BUILT ACCRETIVE OVER TIME ACCUMULATING A SENSE OF CONTINUITY (DONATI, 2016)	12
FIG. 2. 6 ARTISTIC PRESENTATION SPACE BUILT BY CODE. (STUDIOS, N.D.)	13
FIG. 2. 7 HUMANOID ROBOT PEPPER IN AN OFFICE ENVIRONMENT (ROBOTICS, 2017) .	17
FIG. 2. 8 DIFFERENT HUMANOID ROBOTS IN AN ARCHITECTURAL ENVIROMENT.....	18
FIG. 2. 9 CULTURAL DIFFERENCES IN THE PERCEPTION (TAKANISHI, 2005).....	19
FIG. 3. 1 SNACKBOT	22
FIG. 3. 2 MILITARY ROBOT: IROBOT PACKBOT (IROBOT, 2014)	23
FIG. 3. 3 A HUMANOID ROBOT PEPPER AT BRUSSELS AIRPORT, BELGIUM (PETROVA, 2017)	27
FIG. 3. 4 PEPPER BEING CHARGED, DSK BANK, SERDIKA MALL, SOFIA, BULGARIA (PETROVA, 2017)	28
FIG. 3. 5 PEPPER INTERACTING WITH PEOPLE, DSK BANK, SERDIKA MALL, SOFIA, BULGARIA (PETROVA, 2017).....	28
FIG. 3. 6 TABLET PROVIDING INFORMATION ON PEPPER FUNCTIONS, DSK BANK, SERDIKA MALL, SOFIA, BULGARIA (PETROVA, 2017)	29
FIG. 4. 1 DANIEL LIEBESKIND SKETCH DEPICTING MULTIPLE INTERPRETATIONS OF THE CONCEPT OF SPACE. THIS IS A HUGE SCROLL DEPICTING THE MASTER PLAN FOR THE GROUND ZERO SITE IN NEW YORK (LIEBESKIND, 2013)	31
FIG. 5. 1 PEPPER: FOUR MAIN BLIND ZONES (ALDEBARAN/SOFTBANK, N.D.).....	33
FIG. 5. 2 PEPPER GRAY - SCALE VISUALIZATION OF A PACE EXPLORATION, (SOFTBANKS, 2017)	34
FIG. 5. 3 NOLLI MAP: LA NUOVA TOPOGRAFIA DI ROMA, 1748, (NG, 2013)	36
FIG. 5. 4 QR CODE1: QR CODE OF THE DEVELOPED DOMAIN	43
FIG. 6. 1 PEPPER LANGUAGE TABLE, (ROBOTICA, N.D.).....	45
FIG. 6. 2 SAY	49
FIG. 6. 3 ANIMATED SAY	49
FIG. 6. 4 CHOICE.....	50
FIG. 6. 5 CHOICE (LIGHT)	50
FIG. 6. 6 SET RECO. LANG.	50
FIG. 6. 7 CHOICE BOX PARAMETERS	52
FIG. 6. 8 DIALOG	53
FIG. 6. 9 QR CODE 2: DIALOG TUTORIAL 1 (HEIRLAB, 2015).....	55

FIG. 6. 10 QR CODE 3: DIALOG TUTORIAL 2 (HEIRLAB, 2015)	55
FIG. 6. 11 QR CODE 4: DIALOG TUTORIAL 3 (HEIRLAB, 2015)	55
FIG. 6. 12 QR CODE 5: DIALOG SYNTAX RULES (ROBOTICS, N.D.)	55
FIG. 6. 13 QR CODE 4: DIALOG SYNTAX CHEAT SHEET (ROBOTICS, N.D.)	55
FIG. 6. 14 SET LANGUAGE.....	55
FIG. 6. 15 CHOREOGAPHE OVERVIEW	58
FIG. 6. 16 CHOREOGAPHE: CONNECTING TO A PHYSICAL ROBOT	60
FIG. 6. 17 CHOREOGAPHE: CONNECTED TO THE PHYSICAL ROBOT.....	61
FIG. 6. 18 CHOREOGAPHE COMMAND SEARCH	62
FIG. 6. 19 CHOREOGAPHE SPEECH: SAY	63
FIG. 6. 20 CHOREOGAPHE SPEECH: SAY CUSTOMIZATION	65
FIG. 6. 21 CHOREOGAPHE SPEECH: ANIMATED SAY	66
FIG. 6. 22 CHOREOGAPHE: SET RECO. LANG.....	68
FIG. 6. 23 CHOREOGAPHE: CHOICE IN A DEFINITION	70
FIG. 6. 24 CUSTOMIZED CHOICE	71
FIG. 6. 25 CHOREOGAPHE: ADD A DIALOG.....	73
FIG. 6. 26 CHOREOGAPHE: DIALOG CUSTOMIZATION	74
FIG. 6. 27 CHOREOGAPHE: DIALOG SCRIPTING	76
FIG. 6. 28 DIALOG SCRIPT EXAMPLE: SENTENCE BASED WITH NO HIERARCHY.	78
FIG. 6. 29 DIALOG SCRIPT EXAMPLE: KEY WORDS BASED WITH HIERARCHY.	79
FIG. 7. 1 THE SCRIPT OF THE EXECUTABLE FILE 'EXPLORE.PY'	81
FIG. 7. 2 QR CODE 7: THE SCRIPT OF THE EXECUTABLE FILE EXPLORE.PY'	82
FIG. 7. 3 QR CODE 8: PYTHON SDK INSTALLATION GUIDE	82
FIG. 7. 4 QR CODE 9: NAVIGATION BUNDLE FOR CHOREOGAPHE, (ALDEBARAN, 2016)	83
FIG. 7. 5 BUNDLE FOLDER	87
FIG. 7. 6 EXPLORE.PML	88
FIG. 7. 7 CHOREOGAPHE: EXPLORE.PML.....	89
FIG. 7. 8 CHOREOGAPHE: INSTALL AN APPLICATION ON A PHYSICAL ROBOT.....	90
FIG. 7. 9 CHOREOGAPHE: RUN EXPLORE.PML BEHAVIOR	91

TABLE OF TABLES

TABLE 1. FRONTAL SECURITY DISTANCE (ALDEBARAN/SOFTBANK, N.D.)	33
TABLE 2 SPEECH RECOGNITION SOFTWARE WITH RESPECT TO INTERNET CONNECTION AND RECOGNITION TYPOLOGY	46

Robots will refine behavioral space in architecture

.

TABLE OF QR CODES



Fig. 5. 4 QR Code1: QR code of the developed domain



Fig. 6. 9 QR Code 2: Dialog Tutorial 1 (HEIRlab, 2015)



Fig. 6. 10 QR Code 3: Dialog Tutorial 2 (HEIRlab, 2015)



Fig. 6. 11 QR Code 4: Dialog Tutorial 3 (HEIRlab, 2015)



Fig. 6. 12 QR Code 5: Dialog Syntax Rules (Robotics, n.d.)



Fig. 6. 13 QR Code 6: Dialog Syntax Cheat Sheet (Robotics, n.d.)



A detailed information of the Box Input/ Outputs can be found using this QR code



Fig. 7. 2 QR Code 7: The script of the executable file explore.py'



Fig. 7. 3 QR Code 8: Python SDK Installation Guide



Fig. 7. 4 QR Code 9: Navigation bundle for Choreographe, (aldebaran, 2016)

Vesela Danielova Petrova

Curriculum Vitae

Work experience:

since 11.2017
Architectural Assistant
Karl Langer Architekt
Köstlergasse 1, Vienna, Austria

1.03.2016-31.07.2016
Tutor
University of Technology - Vienna

1.07.2014-31.08.2014
Architecture Intern
Vladimirova Architectural Studio
Tsar Osvoboditel Str.10,
Sofia, Bulgaria

15.07.2012 – 05.09.1012
Architecture Intern
Architectural Studio 2000
Hadzhi Dimitar Str.24,
Sliven, Bulgaria

Education and training:

Since 1.10.2015
Master Programme : Architecture
University of Technology – Vienna

1.10.2011-1.10.2015
Bachelor Programme : Architecture
University of Technology – Vienna
Diploma: Architect, B.Sc.

09.2010 – 06.2011
Master Programme : Architecture
University of Structural Engineering
and Architecture “Lyuben Karavelov”
Sofia, Bulgaria
(6-year programme, transferred to
University of Technology – Vienna
after the first year)

09.2005 - 06.2010
“Zahari Stoyanov” Language High
School – Sliven, Bulgaria
Profile: German and English
Diploma: General Secondary School