University of Heidelberg

NAO Robot

Project Documentation

Robotics Practical Lecture

Winter Term 2020

Authors:
Klaus Breuer
Casimir Bokranz

Student IDs:
3549830
3542054

Supervisor:
Peter Wittlinger

Professor:
Prof. Dr. Katja Mombaur

June 10, 2020

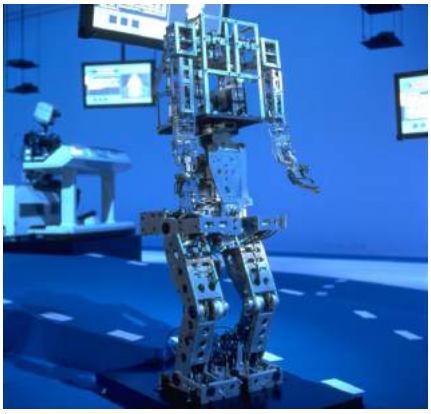
Abstract

This report contains the research and experiments carried out to make a NAO Robot climb a stair autonomously. This document presents the main target, the preparation of the necessary elements, the implemented routines, and finally, the outstanding conclusions of this work.

Contents

1 Introduction

"A robot is a programmable multi functional mechanism designed to move material, part, tools, or specialized devices through variable programmed motion for the performance of variety of tasks". This definition was created in 1979 by the Robotics Institute of America, and represents the very wide field of applications.
Robotics is not only about developing structures similar to human beings. There are other applications such as mobile robotics. However, humanoid robots were always been present in many systems. Already in the 70's Waseda University developed WABOT I, which would be one of the most outstanding humanoids of the decade. A picture of WABOT I is showed in figure 1(a). Likewise, some companies and universities developed robotic arms that, inspired by the mechanics of a human arm, were capable of fulfilling certain previously programmed tasks. An example of this last type of robots is Famulus from the company KUKA, which can be seen in figure 1(b). This is the worlds first industrial robot with six electronically driven axes.



((a)) WABOT I ,1972 [8]

((b)) Famulus by KUKA, 1973 [3]

Figure 1: example of first humanoid robots

The reason why humanoid robotics has stood out as an area of study in recent years is mainly due to its applications in biomechanics, rehabilitation, intelligent prostheses and exoskeletons. Today there are several humanoid robots available in the market for purchase. Examples of this are Spot and Atlas from Boston Dynamics, or Pepper from the company Softbank. SoftBank Robotics is a French company created in 2005 with several subsidiaries, including Aldebaran Robotics, creators of NAO Robot (2004), the robot that will be used in this project. A picture of each of the previous mentioned humanoid robots are shown in figure 2.

((a)) Atlas, 2013 and Spot, 2016 [1]

((b)) Pepper, 2014 [6]

((c)) NAO , 2004

Figure 2: robots at present

The humanoid robots are often seen as ideal assistants to humans. The design which is inspired by humans allow humanoid robots, like NAO, to perform complex motions, like balancing or waking. But the implementation of this tasks in reality are still not fully developed. One reason for limits in daily use, are the different kinds of obstacles in daily environments. One example is a stair.
The movement of stair climbing is very complex. Even for humans this is a challenging task. Children need a long learning and training period to gain this ability and also adults struggle sometimes if the step height or surface varies.
Therefore in this work the stair climbing with the humanoid robot NAO was explored.

1.1 Team and Background

Klaus Breuer. Master Student in Computer Engineering, Heidelberg University. He received his bachelor degree in Electronic Engineering from Federico Santa María Technical University, Chile in 2014. He is currently working as Research Assistant in the Optimization, Robotics Biomechanics research Group (ORB) at Heidelberg University.

Casimir Bokranz. Master Student in Computer Engineering, Heidelberg University. He received his bachelor degree in Mechatronics from Ulm University of Applied Sciences, Germany in 2019. He has experience in companies like Magirus and Kuka, where he did his internship, and in HD Vision Systems, where he worked as a Software Engineer.

2 Target

The main objective of this work was to implement a stair climbing routine for NAO.

This routine should be performed in biomechanical terms like a person would, that is, only using the legs to ascend, the arms accompanying the movement of the legs to maintain balance, and the speed should be constant throughout the routine.
The main objective will be implemented in two different ways, by means of two different control strategies, one open loop and the other one in close loop. The main idea behind both concepts are shown in figure 3 and described in the following sections.

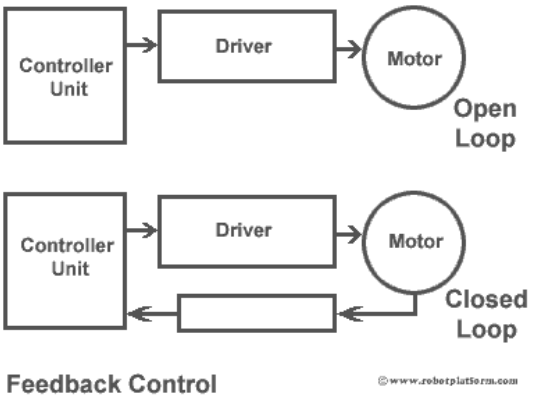


Figure 3: open and close loop control strategies [2]

2.1 Open Loop

An open loop control strategy is characterized by no feedback to the system input. In practical terms, there is no sensor used that allows us to know the effect of the controller units actuators. The main advantage of this type of control loop is that its implementation is relatively simple, since there is no need to implement the use of sensors. However, the disadvantage of this type of strategy is that there is no knowledge of the equality between the obtained and desired output, or the degree of difference.
In summary for the open loop control strategy it was our target to consider:

- Steps: 3
- Sensors: no
- Step height: fixed

2.2 Close Loop

Unlike its predecessor, a close loop control strategy is characterized by having output feedback, so that the performance of the controller unit can be graduated. There are different types of controllers for close loop strategies. For this work an on/off controller was chosen. The controller unit supplies an actuation until the sensor indicates that it is

not longer needed. Then the controller simply sends an actuation equal to zero.
In summary for the close loop control strategy it was our target to consider:

- Steps: 1
- Sensors: foot pressure sensor
- Step height: variable

3 Lab Time

Before starting work in the laboratory, an abstract was written, in which the required and optional objectives, the important dates and the work plan were specified. This abstract can be found in the appendix of this document.
While this plan underwent slight modifications due to internal issues such as scheduling lab appointments or the exams period, at the same time, the agenda was also affected by larger problems such as the current Covid-19 pandemic.

Despite all these complications, it's pleased to confirm that all the objectives set out at the outset were successfully achieved, and with the exception of a rescheduling, the work plan did not undergo any major modifications.

Date	Results	Observations
16.12.19-20.12.19	- Installation of Choreographe	- Familiarization with Nao and Software
14.01.20	- Literature on the subject was found	- Webots was downloaded
10.02.20	- Connection between Webots and Choreographe not implemented	- Literature was read
21.02.20	- Stair was rebuilt	
09.03.20-11.03.20	- First Steps in Routine	- Connection troubles
10.03.20	- Routine almost Complete	- Framework already understood
11.03.20	- Open loop routine successfully implemented (no Walking)	
16.03.20	- Open loop routine successfully implemented (with Walking)	
06.05.20	- First approaches with close loop strategy	
13.05.20-15.05.20	- Close loop routine successfully implemented	

Table 1: Robotics Lab's appointments

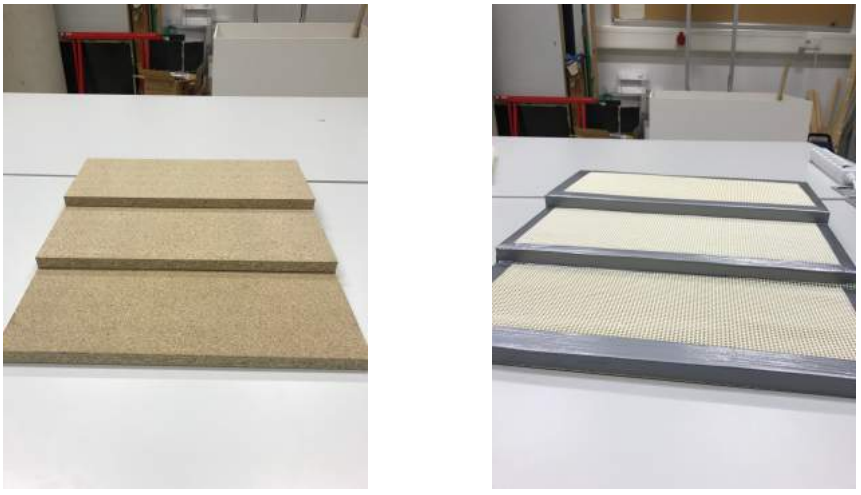
The following points can be commented from this table 1 , where the lab time is shown:

- It started with the basics, installing the necessary software and studying concepts related to NAO and the Choreographe Program.
- Literature related to similar works was studied.
- An attempt was made to achieve a connection between Webots (a Open source software for simulating robots in a 3D environment) and Choreographe, an idea that was later discarded because of connection problems. Moreover the simulation was not necessary, since we had the possibility to work directly with physical robot at the lab.
- The open loop strategy was implemented first because it was easier to implement, while the close loop strategy takes place at the end of the period in the lab.
- None of the control strategies were implemented in a single session.
- The team was constantly attending the lab, at least one week a month.

4 Preparations

4.1 Rebuild stair

The Robotics Lab had a staircase, however, it consisted of three pieces of wood that when joined together formed one. One of the first tasks in the lab was to reconstruct the staircase. For this work, the pieces of wood were glued together, the structure was covered with a non-slip material, and finally, plastic tape was used to define the contours of the structure. The previous and the final stair are shown in figure 4.



((a)) stair before ((b)) stair after

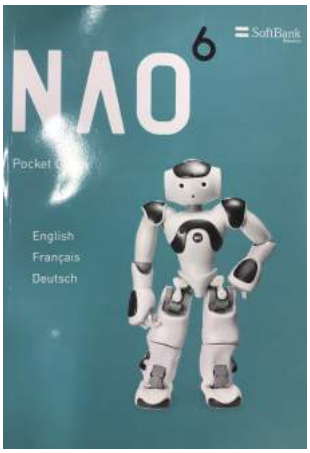
Figure 4: stair before and after reparation

4.2 Length measurement

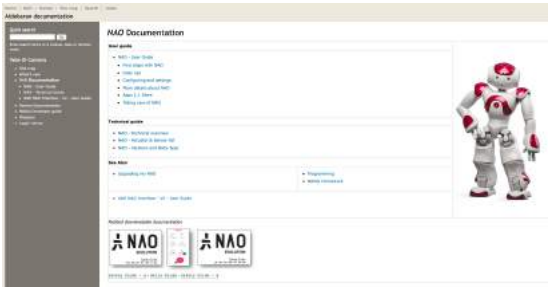
If the stair was too big for the robot, the robot would be doing some kind of tracking, rather than climbing the stair. That's why it was very important, before the experiments, to measure how long and high the stair was and how big NAO is. The measurements were conclusive, and indeed, it was validated that we were working with a scale model.

4.3 Read documentation

The documentation was extremely necessary for process a good project start. For the work with NAO, two main sources were used. The first was the NAO 6 user manual, while the second was NAOs online documentation [4]. The title page of the documentation and a screenshot of the homepage are shown in figure 5. Both sources were useful, however, the online documentation was where more answers were solved, since the web page has examples, codes, definitions, among others.



((a)) NAO Robot user manual



((b)) Aldebaran website

Figure 5: NAO robot official documentation

4.4 Choreographe

Choreographe is one of the most important elements of this project. This consists of a software that allows NAO to be programmed by means of data flow based programming. The signals are moved from block to block, usually from left to right and up to down. Finally it is the data flow that determines which functions are activated first and which are activated later.
Although the structure of the code will be reviewed in detail later, there are certain elements of Choreographe that are explained in the following section.

4.4.1 Block Diagram

Block diagram window is the main window of Choreographe. It contains all the blocks that will be part of the routine. One example of this block diagram window is shown is figure 6. Choreographe has different blocks to implement different ways of programming. However, for this project there were mainly used two: Timeline Box and Python Box.

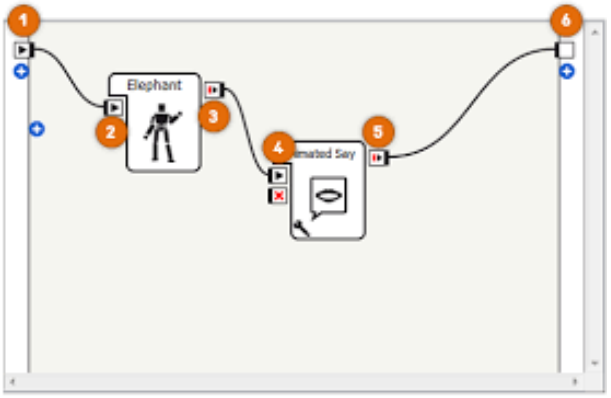


Figure 6: Generic Block Diagram [4]

4.4.2 Timeline Box

This block allows to save different joint positions and execute them when the program runs. To program a position, NAO relaxes his whole body, and that is the moment when moving NAO to the desired position (for example, lift a leg, move an arm, etc.). This particular block was very useful for the open loop control strategy. In this function, it possible to control the speed of movement too. It can see a section of such an timeline box in figure 7.

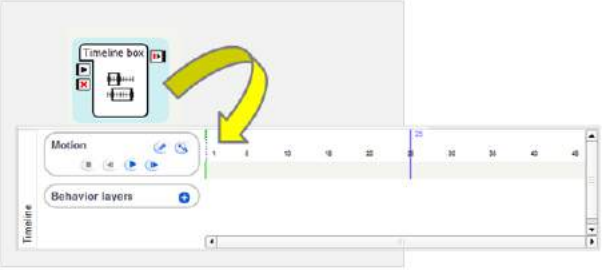


Figure 7: Generic Timeline Box window [4]

4.4.3 Python Box

This block allows through python programming to move to NAO and make use of its sensors. To program a position, a small routine must be written, which can be implemented in different ways. By means of this function it is possible to move NAO's extremities in a translational or rotational way, move a motor at a certain angle, measure from a sensor,

among others. This particular block was very useful for the close loop control strategy. The main structure of a python box is shown in figure 8.

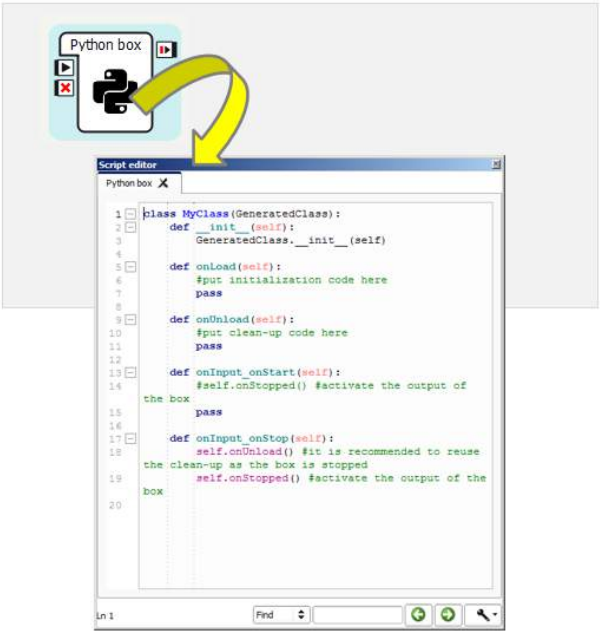


Figure 8: Generic Python Box window [4]

5 Routine Implementation

The tasks for both strategies of stair climbing are already described above. In this section the main idea, the approach and the implementation are described of both strategies.

5.1 Open Loop Strategy

The open loop strategy for stair climbing is inspired by the experience of the ORB research group and Lu Wei [5, 9]. In both projects an feedforward approach was implemented for the step motion. This means, no sensors were used for the motion itself. The positions of all joints, in specified time steps, are taught to NAO, to create the step motion. Such an feedforward approach just works with one defined and known step size.

5.1.1 Approach

Walaa Gouda and Walida Gomaa are describing in their work 5 phases of a step motion [10]. First the robot has to place itself in front of the stair. The next phase is to balance the whole weight on one foot. After this the robot can place the other foot on top of the next step. The end of the third phase is to displace the center of mass on this foot. The fourth movement is to place the lower leg next to the higher one. So the robot gets to the fifth phase, where he is balancing on both legs [10].
The same phases can be seen in Lu Wei's work . He uses timeline commands to implement

the climbing motion [9]. With these timeline commands the robot reaches a specified configuration of all joints after a given time. One timeline command can store multiple joint configurations in different points in time [4].

In this work the motion is implemented with this timeline commands. With this the step motion can be achieved.
The goal of the open loop strategy is to climb 3 steps with defined and known dimensions. NAO starts this motion in front of the stair and has not to place itself in front of the stair. So the first phase of Walaa Gouda and Waliad Gomaa must not be executed [10]. The following 4 phases are executed. The realized motion of this work should just be inspired by both referenced works and not be as near as possible on there motions [10, 9].
After the first step, NAO starts the second step with the other foot. The third step starts with the same foot as the first step. With this the robot will do an alternating motion to climb the three steps of the stair.

5.1.2 Implementation

This implementation step of the project is inspired by the second to fifth phases of Walaa Gouda and Walida Gomaa [10]. In the following just the first step is described, because the second step has mirrored poses and the third step similar one. Here it's important that the poses are similar, but not exactly the same. The reason for this is, that each step of the stair has different dimensions and NAO is not similar oriented to each step. The block diagram of the Choreographe project is showed in figure 9. Here it can see an initial stand up command and after this the three timeline boxes.

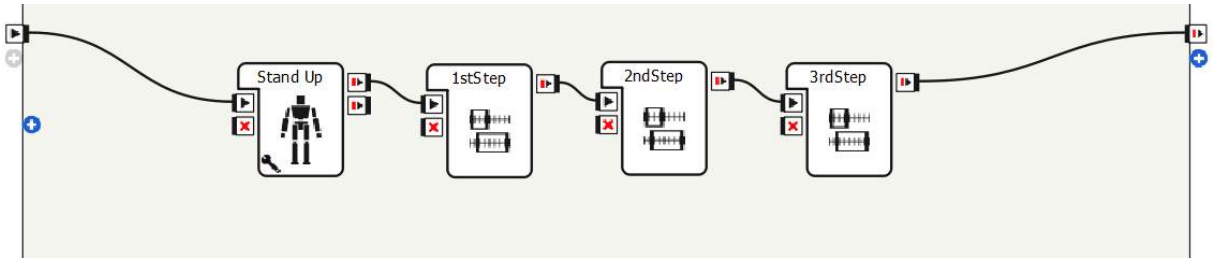


Figure 9: block diagram of open loop

To describe the first implemented step motion more in detail, in this section 7 phases are described. At the end of each phase NAO is standing in a stable position. The motion in each phase needs some additional fix joint poses, to prevent a crash of the robot.

The initial pose (Subfigure 10(a)) of the robot is the build in "Stand Up" posture in Choreographe. In this pose NAO is standing up straight. The weight is equally distributed on both feet. From this pose the robot moves in pose 2 (Subfigure 10(b)). With this motion NAO shifts his center of pressure to the left. In this pose the left foot holds all the weight of the robot. So from now the right foot can be moved freely through the space.

((a)) Pose 1

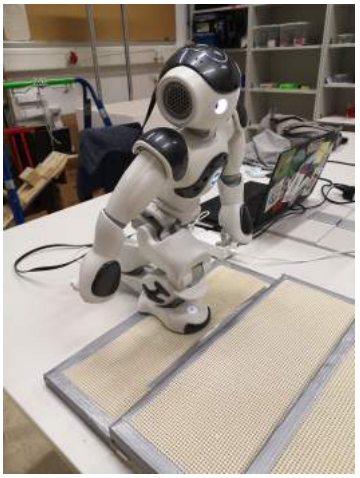
((b)) Pose 2

Figure 10: Feed forward poses 1 and 2

With the next motion the right leg is moved up and forward. It has to be taken into account that the toes of the foot don't touch the edge of the step. At the end of this motion NAO stands in pose 5 (Subfigure 11(a)). In the next motion the robot places his right foot on the top of the next step. In this motion the robot loses his balance for a very short moment and ends in pose 6 (Subfigure 11(b)). From this pose NAO brings his weight over the right foot. So he can move in the next step his left foot free in space. In pose 9 (Subfigure 11(c)) his right knee is bent.

((a)) Pose 5

((b)) Pose 6



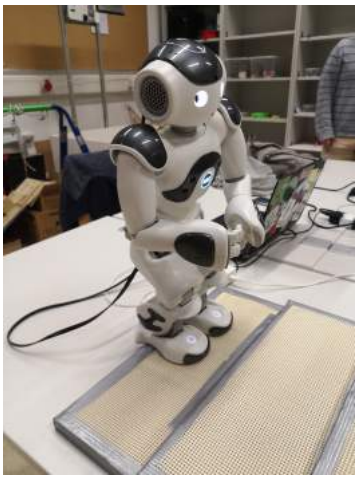
((c)) Pose 9

Figure 11: Feed forward poses 5, 6 and 9

In the next motion NAO gets the height of the step. For this he pushes himself up with the right leg. To improve stability, NAO moves both arms to the side. So his inertia is

smaller and its easier to maintain stability. The final pose after this motion is shown in subfigure 12(a). From this pose NAO places his left foot next to his right foot on the step and brings his arms together, because the critical balance part is over. Now he stands in pose 17 (Subfigure 12(b)) stable on the step. With the last motion the robot is standing up straight. He ends in the final pose 19 (Subfigure 12(c)).

((a)) Pose 13



((b)) Pose 17

((c)) Pose 19

Figure 12: Feed forward poses 13, 17 and 19

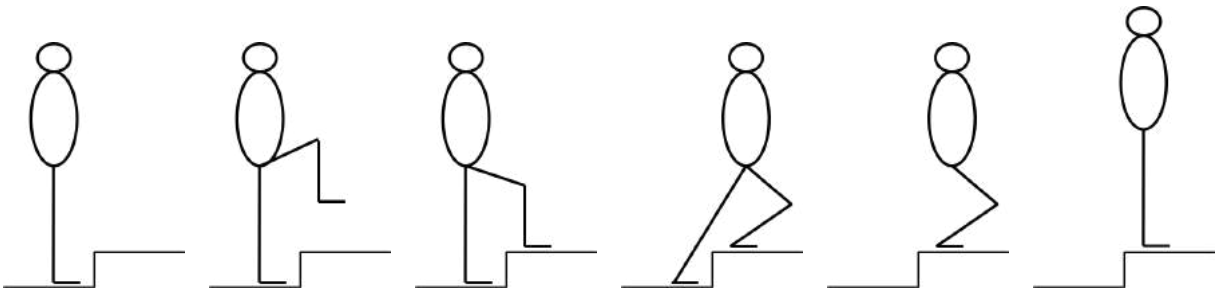
In the case of open loop control strategy two routines were implemented. In the first one, NAO begins to climb the stair immediately. In the second one, NAO walks in search of the stair and when he detects it, by means of the proximity sensors in his feet, he starts to climb it.

5.2 Close Loop Strategy

In contrast to the open loop strategy, the close loop strategy has the target to climb a stair with varying step height. To reach this goal NAO uses the 4 force sensitive resistors in each foot to recognise the step [4]. In this work it is focused in the close loop strategy on one step of a stair. It is no whole stair implemented or tested, but the procedure of one step can be reiterated for the next steps of the stair. For this it has to be ensured, that the robot is positioned as near as possible in front of each step.

5.2.1 Approach

From the first pose (Subfigure 13(a)), the robot moves the left foot up. NAO ends the first movement in the second pose (Subfigure 13(b)), independent of the step height.



((a)) Pose 1 ((b)) Pose 2 ((c)) Pose 3 ((d)) Pose 4 ((e)) Pose 5 ((f)) Pose 6

Figure 13: Feed back poses 1 to 6

To place the left foot on the step, a loop is implemented. In each iteration NAO moves his left foot a small space down. Then he reads out the 4 force sensitive resistors and adds the values up. If the value is under 3 no stair is detected and the robot hands on with the next iteration. If the value is even or more than 3, it will be interpreted as a contact and NAO has reached pose 3 (Subfigure 13(c)). The value of 3 is used, because each sensor measure sometimes errors between 1 and 2. In practice it showed, that the value of 3 performs well to make a decision.
After this the weight of NAO is shifted over the left foot. Now the robot can go from pose 4 (Subfigure 13(d)) to pose 5 (Subfigure 13(e)). In this movement NAO places the right leg next to the left one. To achieve this, the same approach is used as for the first foot. So first the right foot is placed over the step and then moves iterative down since there is a contact.
At the end NAO has to stand up and ends in pose 6 (Subfigure 13(f)).

5.2.2 Implementation

To implement the movements, described in the previous chapter python blocks in Choreograph were used. In these python blocks, it is possible to use cartesian commands. With these commands the points of NAO shown in figure 14, can be moved in cartesian spaces. This makes it uncomplicated to move for example the leg up. So nearly all movements were implemented with these commands, while manipulating the position of both legs and the torso. The points of both arms and the head were not used.

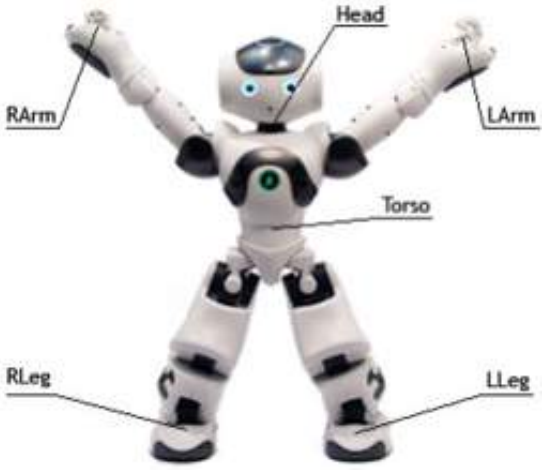


Figure 14: origin of cartesian spaces for cartesian commands [7]

Just for shifting the weight it was necessary to move the left hip motor directly. Also therefor the python framework gives an method. Also the readout of the pressure sensors can be handled in python. For moving the arms of the robot two timeline commands are used, where just the position of the arms are stored. In this way the arms could be manipulated at the same time independent from the feet position.
The resulting block diagram in Choreograph is shown in figure 15. In the first line you can see the python blocks for placing the left foot down. In the second line the weight is shifted over the left foot. In the last line right foot is placed on the step and NAO is standing up.

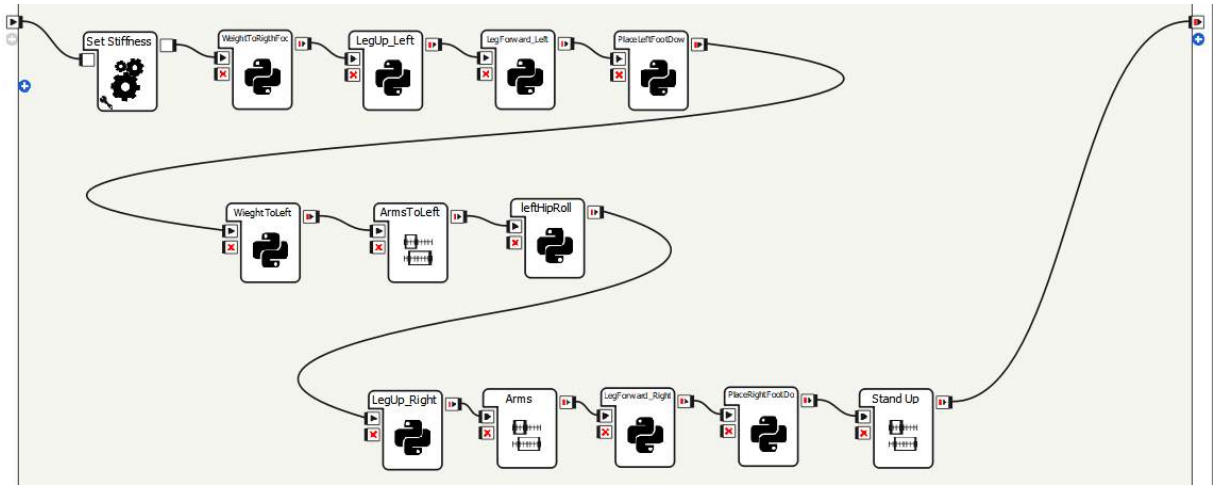
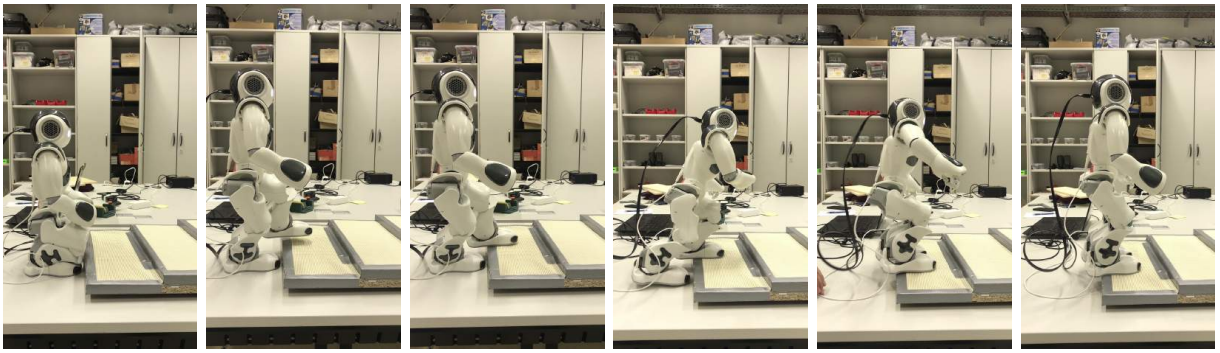


Figure 15: block diagram of closed loop

In figure 16 one stair climbing motion is shown. Here it can be seen the main positions, mentioned in the approach. In the appendix pictures of three motions with different step height are depicted. In this movements it is good to see, that they differ from each other

after pose 2. To achieve pose 2 the step height has no influence to the movement. And at the end of the movement NAO reaches, independent of the height of the step, the same position (pose 6). Between those poses the joint configurations of NAO differ from each other.



((a)) Pose 1 ((b)) Pose 2 ((c)) Pose 3 ((d)) Pose 4 ((e)) Pose 5 ((f)) Pose 6

Figure 16: Feed back poses of NAO

6 Summary

Two different approaches for stair climbing were implemented in this work. In the following sections problems, the experience with these approaches and possible future work are described. At the end we will come to the conclusion of the work.

6.1 Problems

At the work with NAO and the implementation of both routines some problems occurred. Some of them could be solved and others stayed. The unsolved problems must be considered, if future projects are planned.
The first problematic behavior, which was observed, are connection errors between NAO and the computer. This occurred not often and could be handled, by restarting NAO and reconnect afterwards to the robot again. These disconnects cost time and are therefore a little bit annoying, but they did not influence the movement of NAO.
The second observation is not allocated to the robot. It is caused from the rubber mats, which were mounted on the stair. On the one side they increased the friction of the surface so the foots of NAO aren't sliding anymore. On the other side they reduced the stability of the robot. The reason for this is the softer underground of the rubber mat. So it is not clear if the increase of friction is better, than the loss of stability.
The next issue which was handled is the Fall Manager of NAO [4]. This is an behavior of NAO, which is running per default in all applications of NAO. The idea is to protect the robot if he is falling. Unfortunately this Fall Manager is very sensitive. At some point of the stair climb movements it is wanted to 'fall' just a little bit on the other foot. If the Fall Manager detects this fall, NAO reacts by rising up his hands very fast to protect

his head. This fast behavior leads to a fall in any case. So in the open loop approach we designed the movement in a way, that the Fall Manager never detects a falling. In the close loop approach it was not possible, because the size of the stair and so the movement itself was varying for every step height. So we turned the Fall Manager off. So it will not interrupt the movement of climbing the step. An instruction, in which way you can turn of the Fall Manager you can find in the online documentation of NAO [4].
The biggest problem of NAO in stair climbing is the limitation of the hardware. These limitations caused in a very high motor heat after a short working period with the robot. If the motors of NAO are to hot, he will shut down and needs to rest to cool down. This behavior made it time intensive to work. Out of 4 hours of work it was just possible to move NAO 1,5 hours in the laboratory. The even worse problem of the hardware limitations are some unexpected behaviors of NAO in some situations. For example if the robot should lift his left leg up in z-direction, he will do this in the most cases. But in some other cases he will move in addition also in x- and y-direction, which makes the movement unpredictable and uncontrollable. The reason for this could not be found.

6.2 Best Practices

While reflecting both strategies it is clear to say, that the close loop approach is more flexible and reliable compared to the open loop approach. With the open loop implementation NAO climbs the specific stair 1 out of 2 times without falling. If the dimensions of the stair changed, NAO is not able to climb without falling. Already with removing the rubber mats, a fall was caused. For an application in daily life or as an reliable process this is not acceptable.
The tests with the close loop approach are more reliable. NAO can walk up steps with a step height between 1.2cm and 3.5cm reliable and in some times also up to 3.8cm. He also climbs the stair successfully with and without the rubber mat. So the close loop approach is reliable and more flexible. On the other side, there are also some disadvantages of the close loop routine to mention. The climbing of one step needs up to 1 minute, what is a lot of time, if you think about a whole stair. Also in some special cases the earlier mentioned unpredictable behaviours of NAO causes to an crash of NAO.

6.3 Future Work

It is necessary to develop the field of stair climbing with humanoid robots further. For this it can be interesting to work with the close loop approach. On one side it is interesting to improve the step motion it self. It is important to make it more reliable and faster for a real application. For this it can be good to follow a more human like movement. Also the use of other sensors, like for example vision, can improve the motion itself.
Another interesting topic would be the implementation for a whole stair climbing. This would begin with the recognition of the stair itself, then positioning in front of the stair and afterwards climbing the whole stair.
Maybe it is also necessary to switch to another hardware for stair climbing, if you want to develop a system which can reliable walk up stairs in daily life and in normal environments. The reason for this are the earlier mentioned hardware restrictions of the NAO.

6.4 Conclusion

At the end of this work we can say, that with this project the complexity of a stair climbing motion is shown. Even more complex is the implementation of such a motion with an humanoid robot. To rise to this challenge, NAO in combination with Choreographe is a good choice to make first experiences. Because of the given interface of NAO, it is convenient to implement stair climbing. For example you have not to care about the forward kinematics. But also a more complex handling of the robot is possible with the python blocks of Choreographe, where you can apply functionalities like cartesian movement of specified parts of NAO.
With these possibilities in this work two approaches were implemented to achieve the goal of stair climbing. As a result the open loop implementation is less reliable and less flexible as the close loop approach, where the pressure sensor were used. With this implementation NAO can climb stairs between 1.2cm and 3.5cm reliable. The work also showed the restrictions hardware causes in implementation. So it is important to chose the right hardware for a certain task.

References

[1] Atlas and Spot. https://www.i-programmer.info/news/169-robotics/13127-atlas-the-gymnast-spot-the-dog.html. Accessed: 2020-05-28.

[2] control. http://www.robotplatform.com/knowledge/motion_control/feedback_control.html. Accessed: 2020-05-28.

[3] KUKA Famulus. https://automationspraxis.industrie.de/industrierobotik/nggallery/image/kuka-famulus/page/2/. Accessed: 2020-05-28.

[4] Nao Software Documentation. http://doc.aldebaran.com/1-14/contents.html. Accessed: 2020-04-17.

[5] Optimization, Robotics and Biomechanics (Rubrecht-Karls-Universität Heidelberg). https://typo.iwr.uni-heidelberg.de/groups/orb/home/. Accessed: 2020-04-01.

[6] Pepper. https://www.therobotreport.com/trust-key-service-robot-design-softbank-pepper/. Accessed: 2020-05-28.

[7] Reference Documentation. http://www.cs.cmu.edu/~cga/nao/doc/reference-documentation/naoqi/motion/control-cartesian.html. Accessed: 2020-05-28.

[8] WABOT I. https://www.newsviewsnetwork.com/humanoid-robots-part-1-the-journey/. Accessed: 2020-05-28.

[9] Wei Lu. Design and implementation of autonomous stair climbing with NAO humanoid robot. http://urn.fi/URN:NBN:fi:amk-201505188582, 2015.

[10] Walid Gomaa Walaa Gouda. Complex motion planning for nao humanoid robot. 2014.

A Previous documentation

1. Abstract
2. NAO Project Apply

Nao Robot Project

Klaus Breuer, Casimir Bokranz
Master Computer Engineering Students, Second Semester.
Robotic Practicals
Robotics Teaching Lab, Heidelberg University
Wintersemester 2019/2020

To climb a stair. The idea is to implement a routine, where the robot can climb a stair. To do that the robot is going to use its different sensors.

Targets

required

- ✓ The robot can climb a stair .
- ✓ The robot can climb different kind of stairs with different step height.

optional

- ✓ The robot recognize the size of the stair and recognize the height.
- ✓ The robot should decide if it's possible to climb the stair.

Important Dates

07.02 First Milestone

23.03 Deadline Functionality (All have to work!)

06.04 Project Presentation and send Report (2 Weeks later)

Work Plan

20.12 Familiarize with Nao (Hardrware) and decide the programming Strategy (Software)

20.01 Normal walking (without Stair)

05.02 Familiarize with sensors for recognition and Implementation Step motion

20.02 Climb a Stair with know parameters

05.03 Implement Recognize-decide code

20.03 Routine works "Recognize-decide-walk"

December 4, 2019

Application to Nao Robot project

Klaus Breuer, Casimir Bokranz
Robotic Practicals
Robotics Teaching Lab, Heidelberg University
Wintersemester 2019/2020

Major / minor subject and academic semester

- **Breuer:** Master in Computer Engineering, Second Semester.
- **Bokranz:** Master in Computer Engineering, Second Semester.

Relevant credits

- **Breuer:**
  - 6 Credits. Currently taking Robotics I at Heidelberg University
  - 6 Credits. Biomechanics at Heidelberg University
  - 6 Credits. Microcontrolles at Heidelberg University
- **Bokranz:**
  - 5 Credits. Robotics at Ulm University of Applied Sciences
  - 6 Credits. Biomechanics at Heidelberg University
  - 6 Credits. Design of Autonomous Systems
  - 6 Credits. Currently taking Robotics I at Heidelberg University

Relevant prior knowledge or experiences

- **Breuer:**
  - Python course at Heidelberg University (Physics)
  - Biomechanics Project
- **Bokranz:**
  - Experience in Python while Bachelor thesis

Motivation

Humanoid robots are a fascinating topic in science, but potential customers of such robots have not much trust in this technology yet. Better humanlike motions and behaviour could increase this trust in robots. We would like to implement different behaviours to the robot to learn more about the human motion and what are the important or less important characteristics of humanlike motions. We look forward to work with different sensors and actuators.

Priorities State: 1. Nao Robot - 2. Delft Hand - 3. KUKA Arm

October 30, 2019

B Poses of the open loop strategy

((a)) Pose 1

((b)) Pose 2

((c)) Pose 3

((d)) Pose 4

((e)) Pose 5

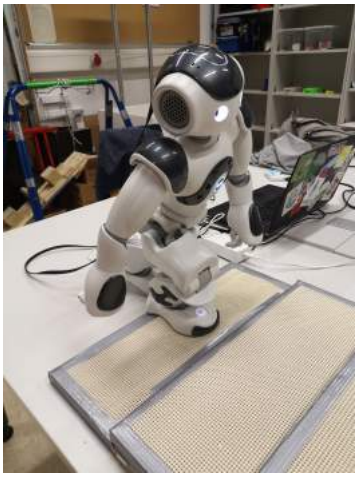
((f)) Pose 6

Figure 17: Feed Forward Step Pose 1-6

((a)) Pose 7

((b)) Pose 8

((c)) Pose 9



((d)) Pose 10

((e)) Pose 11

((f)) Pose 12

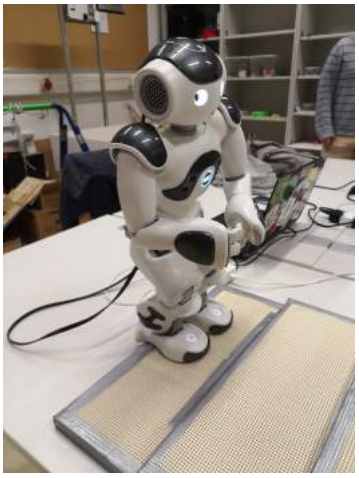
((g)) Pose 13

((h)) Pose 14

((i)) Pose 15

Figure 18: Feed Forward Step Pose 7-15

((a)) Pose 16



((b)) Pose17

((c)) Pose 18

((d)) Pose 19

Figure 19: Feed Forward Step Pose 16-19

C Close Loop Tests with different step heights

((a)) walk 1 pose 1

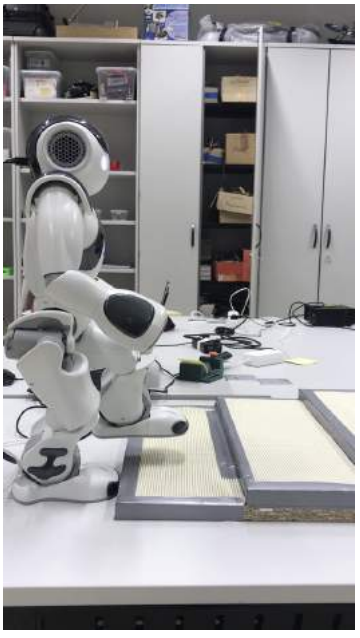
((b)) walk 2 pose 1

((c)) walk 3 pose 1

Figure 20: Close Loop Step 1

((a)) walk 1 pose 2

((b)) walk 2 pose 2



((c)) walk 3 pose 2

Figure 21: Close Loop Step 2

((a)) walk 1 pose 3

((b)) walk 2 pose 3

((c)) walk 3 pose 3

Figure 22: Close Loop Step 3

((a)) walk 1 pose 4

((b)) walk 2 pose 4

((c)) walk 3 pose 4

Figure 23: Close Loop Step 4

((a)) walk 1 pose 5

((b)) walk 2 pose 5

((c)) walk 3 pose 5

Figure 24: Close Loop Step 5

((a)) walk 1 pose 6

((b)) walk 2 pose 6

((c)) walk 3 pose 6

Figure 25: Close Loop Step 6