Developing an autonomous hexapod robot for environmental exploration

Simon Cherlet, Pieter Maelegheer

Supervisors: Prof. dr. ir. Robain De Keyser, Dr. ir. Francis wyffels
Counsellors: Dr. Cosmin Copot, Jairo Andres Hernandez Naranjo

Master's dissertation submitted in order to obtain the academic degree of
Master of Science in Electromechanical Engineering

Department of Electrical Energy, Systems and Automation
Chairman: Prof. dr. ir. Jan Melkebeek

Department of Electronics and Information Systems
Chairman: Prof. dr. ir. Rik Van de Walle

Faculty of Engineering and Architecture
Academic year 2014-2015
Developing an autonomous hexapod robot for environmental exploration

Simon Cherlet, Pieter Maelegheer

Supervisors: Prof. dr. ir. Robain De Keyser, Dr. ir. Francis wyffels
Counsellors: Dr. Cosmin Copot, Jairo Andres Hernandez Naranjo

Master’s dissertation submitted in order to obtain the academic degree of Master of Science in Electromechanical Engineering

Department of Electrical Energy, Systems and Automation
Chairman: Prof. dr. ir. Jan Melkebeek

Department of Electronics and Information Systems
Chairman: Prof. dr. ir. Rik Van de Walle

Faculty of Engineering and Architecture
Academic year 2014-2015
Preface

Spurred by the enormous improvements in computer processing power, the field of robotics has flourished tremendously over the past few decades. Besides the traditional robot manipulator arms which have already reached a high state of maturity, considerate attention is being spent to the creation of mobile robots and gifting these with ever higher degrees of autonomy. Such autonomous mobile robots will relentlessly keep expanding their field of application and as such come in ever closer contact with our every day lives.

The mere thought of being able to grant a machine the possibility to sense and act in an environment on its own enticed us to work upon the masters’ project at hand. As it turns out this project has been an incredible enrichment yielding us multiple insights in the how and why of mobile robotics. What made the robotic experience particularly attracting for us is the fact that it is an inherently interdisciplinary research area involving mechanical engineering, electrical engineering, computer science, cognitive psychology, neuroscience...

It is our strongest hope that the amount of enthusiasm we’ve experienced throughout the development of the autonomous hexapod robot is in some way reflected in this written dissertation. Moreover we hope that the hexapod robot we’ve built can indeed serve as a steppingstone towards more involved research regarding mobile robots at the EESA department.

Before going any further we would like to express our sincerest gratitude towards anyone that made this masters’ project possible, in particular our counsellors dr. ir. C. Copot and ir. A. Hernández.

Simon Cherlet & Pieter Maelegheer,
May 2015
Permission for usage

“The authors give permission to make this masters’ dissertation available for consultation and to copy parts of this master dissertation for personal use. In the case of any other use, the copyright terms have to be respected, in particular with regard to the obligation to state expressly the source when quoting results from this master dissertation.”

May 2015
Developing an autonomous hexapod robot for environmental exploration

Masters’ dissertation submitted by Simon Cherlet & Pieter Maelegheer in order to obtain the degree of MSc in Electromechanical Engineering - Control Engineering & Automation.

Academic year 2014–2015. Faculty of Engineering & Architecture
Department of Electrical Energy, Systems & Automation

Supervisors: prof. dr. ir. R. De Keyser, dr. ir. F. Wyffels
Counsellors: dr. ir. C. Copot, ir. A. Hernández

Abstract

Autonomous six-legged robots show great promise for use in applications where rough, unstructured terrain has to be traversed such as search and rescue operations, volcanic exploration, mountain logging... The department wished to stimulate research upon the matter by reviving a 10-year-old hexapod robot from a former project with currently available actuators, sensors and computing power. The goal of this masters’ project is to provide the available hexapod robot with the fundamental building blocks to wander around certain predefined environments autonomously without putting itself into harm’s way. This could be seen as a first elementary step towards environmental exploration. Hereto, first a thorough literature study is performed on autonomous mobile robots, hexapods in particular and embedded programming. Next the components deemed necessary are purchased and put together. As such the robot’s sensory system consists of tactile sensors, infra-red range finders, a stereo camera and an inertial measurement unit (IMU). Subsequently a forward and inverse kinematics model is derived giving insight into the leg actuation part. At last all of the interfacing and programming is performed in Eclipse using mainly Python as programming language. The end result is the fully operational hexapod robot capable of walking straight with different gaits reaching top speeds of 10 cm/s, turning on the spot, walking in reverse, detecting ledges and obstacles and mounting slopes of up to 10° inclination with its body levelled out horizontally. The hexapod is thus endowed with the basic skill set to cope in predefined environments on its own. Now a next step could be enabling the robot to navigate through space in an intelligent, deliberate way.

Keywords

Hexapod robot, autonomous, forward & inverse kinematics, IMU
Developing an autonomous hexapod robot for environmental exploration

Simon Cherlet & Pieter Maelegheer

Supervisor(s): Prof. Dr. Ir. R. De Keyser, Dr. Ir. F. Wyffels, Dr. Ir. C. Copot, Ir. A. Hernández

Abstract—As a remnant of a robotics project carried out about then years ago the EESA department happened to have a rather basic hexapod robot at its disposal. In order to stimulate research upon six-legged robots and mobile robots in general the department expressed the wish to revive the old robot with currently available actuators, sensors and computing power. As such the goal was set for this masters’ dissertation to provide the available hexapod robot with the fundamental building blocks to wander around certain predefined environments autonomously without putting itself into harm’s way. This could be seen as a first elementary step towards environmental exploration. The extended abstract under current consideration tries to summarise the steps taken to bring this goal to completion and highlight the most important results.

Keywords—Hexapod robot, autonomous, forward & inverse kinematics, inertial measurement unit

I. INTRODUCTION

SPURRED by enormous improvements in actuation, sensing and computing capabilities, the field of robotics has flourished tremendously over the past few decades. Besides the traditional robot manipulator arms which have already reached a high state of maturity, considerate attention is being spent to the creation of mobile robots and gifting these with ever higher degrees of autonomy. One such type of mobile robots being the target of countless studies in research facilities and universities all over the world is the six-legged or hexapod robot. These insect-like machines possess great stability properties and as such show great promise for use in applications where rough, unstructured terrain has to be traversed such as search and rescue operations in the wake of natural disaster, volcanic and/or extraterrestrial exploration, mountain logging...

In what follows, first the steps taken and components bought to get to a working hexapod on which experiments could be performed are discussed. Next a closer look is given to the implementation of the walking gaits. Then two important capabilities for letting the robot cope on its own - being ledge and obstacle detection - are considered and subsequently an algorithm devised to let the hexapod mount slopes of up to 10° inclination while keeping its body levelled out horizontally is looked at. To finish off, a conclusion is given.

II. HEXAPOD ROBOT LIL’ HEX

In order to achieve the goal set out for this masters’ project (cfr. see abstract above) first an extensive literature study has been conducted on autonomous mobile robots, hexapods in particular and embedded programming. As stated by Bekey (2005) autonomous mobile robots are intelligent machines capable of performing tasks in the world by themselves, without explicit human control over their movements [1]. They are programmed to act and make decisions based on sensory feedback of their external and/or internal surroundings.

With this in mind necessary and reckoned useful components are purchased and mounted onto the available hexapod robot (stripped of the bulk of its former components). The central processing unit is a single-board embedded computer running Linux called BeagleBone Black. To operate the twelve hobbyist servos actuating the hexapod’s joints - i.e. two rotational joints per leg, one enabling leg swing, another leg lift - a separate Mini Maestro servo controller is used which is interfaced to the BBB via TTL serial communication. Its sensory system consists of six FSRs mounted onto the leg tips as tactile sensors, three IR PSDs for obstacle detection, an IMU to obtain info on the body’s orientation and a stereo camera. The latter could serve several purposes (e.g. obstacle detection and depth measurement, object recognition...) but is bought mainly in view of further research involving visual odometry and mapping. These sensors communicate to the BBB respectively via analog communication (FSRs and PSDs), I²C and USB. As for power supply, the twelve servos are fed separately using five rechargeable 1.2 V NiMH 2700 mAh batteries and the rest of the devices are fed by a 7.4 V Li-ion 2200 mAh rechargeable battery, using an additional DC/DC converter to supply the BBB with the necessary 5 V voltage level.

Once the hexapod is built into its final design it is given the name Lil’ Hex (figure 1). Looking at the rather basic mechanical design of the hexapod robot at hand (made out of plastic, simple hobbyist servos, only 2 DOFs per leg...) it is clear its application potential is confined to specific predefined environments posing not too large of a challenge (e.g. fairly flat terrains, only small rubble...).
III. WALKING GAITS

The initial experiments focused on enabling the limbed robot to walk and turn using different gaits and speeds. Hereto first a forward and inverse kinematic model of the legs was constructed to gain knowledge on the relationship between the servo-actuated joint angles and the leg displacement. Once this model was available three particularly interesting statically stable walking gaits have been implemented, namely the wave, ripple and tripod gait. It appeared letting the six-legged robot walk a straight line was hindered by influences such as slippage, slightly different step lengths etc. causing it to stray off over time. To keep the hexapod from drifting left or right the formerly open loop actuation was augmented with sensory feedback from its IMU (yaw data). It should be noted that although this implementation managed to let the hexapod run straight for far greater distances than before it could not prevent it from steadily moving laterally to its initial forward heading.

Next step was then to compare the speeds reached using different gaits and altering two parameters: the hip swing angle and speed. From this the most important insight is that Lil’ Hex reaches its top speed of 10 cm/s using the tripod gait making it about 100 times slower than Usain Bolt running the hundred meters. Furthermore whereas the tripod gait is only recommended for use on fairly flat, even terrains, the slowest most stable gait, i.e. the wave gait, is advised when the terrain is more rugged and slopes are to be overcome.

Finally to grant the legged robot complete manoeuvrability walking in reverse and turning on the spot have been added to its skill set. By using yaw information obtained from the IMU the latter could be done over an angle of its choosing.

IV. LEDGE & OBSTACLE DETECTION

In light of letting the hexapod cope autonomously two fundamental capabilities have been added to its repertoire: ledge and obstacle detection. The former is achieved by mounting FSRs on the leg tips of the robot and using these as boolean operators (contact/no contact), whereas the latter is done by means of infra-red range finders (i.e. IR PSDs) at the back and front and a binocular camera. If the legged robot for example detects a ledge or an obstacle, it could decide to reverse, turn and walk on in a different direction.

Contrary to the stereo camera, the IR PSDs have not been relied upon for precise distance measurement, they merely serve to detect whether or not an obstacle is nearby. Furthermore, as already mentioned before, the camera has mainly been purchased in light of further research involving visual odometry and mapping. Nonetheless experiments have been done using the stereo camera as well. To this end Brahmbhatt’s book on certain OpenCV functionalities proved particularly useful [2]. For instance after calibration, stereo rectification, disparity calculation and grayscale mapping the stereo camera was able to discern the closest objects in front of it, determine their dimensions, draw a contour around them and yield very precise estimates of the distance to the objects (figure 2). However letting the camera act upon the measured obstacles using stereo vision has not been implemented thus far, especially because of the fact that spurious measurements tend to occur as soon as the environment wasn’t well set-up as depicted in figure 2.

V. BODY LEVELLING

The foregoing sections already made mention of using orientation data obtained from the IMU. In fact a lot of study has gone in to how and why the accelerometer and gyroscope data (IMU) are fused together to yield information regarding the hexapod’s body orientation and how this could be retrieved from the IMU. As such another particularly interesting experiment was using roll, pitch and yaw data retrieved from this sensor in conjunction with leg actuation to let the hexapod robot level out its body horizontally at all times when standing on a slope of which the inclination could be adjusted. Basically the algorithm behind it involves using trigonometric relationships and the available kinematic models to calculate how the leg tips should be redirected in a plane able of counteracting the measured orientation angles. In a next step this body levelling ability was then used to let the limbed robot walk up slopes of up to 10° inclination using the aforementioned wave gait. This greatly benefits the way the robot absorbs the load posed on its rather frail mechanics. As a final remark it should be noted that due to the computation time needed to solve the non-linear equations involved, the levelling out does not happen instantaneously, but with a brief delay. Perhaps further optimisations could be made in this regard.

VI. CONCLUSION

The goal of this masters’ project has been successfully accomplished. The end result is the fully operational hexapod robot capable of walking straight with different gaits reaching top speeds of 10 cm/s, turning on the spot, walking in reverse, detecting ledges and obstacles and mounting slopes of up to 10° inclination with its body levelled out horizontally. The hexapod is thus endowed with the basic skill set to cope in predefined, not too challenging environments on its own (cfr. mechanical limitations). Now a next step could be enabling the robot to navigate through space in an intelligent, deliberate way.

REFERENCES

Contents

Preface iv
Permission for usage v
Abstract vi
Extended abstract vii
Table of contents ix
List of Figures xii
Used abbreviations xiv

1 Introduction 1
   1.1 Start of thesis 2
   1.2 Initial status 3
   1.3 Objective 4
      1.3.1 Main objective 4
      1.3.2 Subgoals 5
   1.4 Research plan 6
      1.4.1 Problem statement 6
      1.4.2 Research questions 6
      1.4.3 Relevance 6
      1.4.4 Research method 7
      1.4.5 Planning 8
2 Autonomous mobile robots 9
   2.1 Autonomy 10
   2.2 Mobile robots 11
      2.2.1 Locomotion 11
      2.2.2 Sensing 15
CONTENTS

2.2.3 Control .................................................. 17
2.3 Environmental exploration . ................................ 19
  2.3.1 Mapping ............................................. 19
  2.3.2 Localisation .......................................... 20
  2.3.3 Navigation ........................................... 21
2.4 Applications .............................................. 22
  2.4.1 Mobile robots in general ......................... 22
  2.4.2 Six-legged robots in particular .................. 23
2.5 Further evolutions .................................... 24

3 Lil’ Hex ....................................................... 25
  3.1 Hexapod frame ......................................... 26
  3.2 Actuators ................................................ 26
  3.3 Sensors .................................................. 28
    3.3.1 Force sensitive resistor ....................... 28
    3.3.2 Inertial measurement unit ..................... 30
    3.3.3 Infra-red position sensitive device .......... 32
    3.3.4 Stereo camera .................................. 33
  3.4 Processing unit ..................................... 35
  3.5 Power supply ........................................ 36
  3.6 Lil’ Hex ............................................... 37

4 Walking gaits ............................................... 39
  4.1 Kinematic model ...................................... 40
  4.2 Terminology and actuation ......................... 43
  4.3 Tripod Gait ........................................... 44
  4.4 Ripple Gait ............................................ 45
  4.5 Wave Gait ............................................. 45
  4.6 Run straight .......................................... 46
  4.7 Comparison ........................................... 47
  4.8 Turning and reverse .................................. 49

5 Ledge and obstacle detection ............................ 51
  5.1 Ledge detection ....................................... 52
  5.2 Obstacle detection ..................................... 54
    5.2.1 IR PSDs .......................................... 54
    5.2.2 Stereo vision ................................... 57
6 Body levelling 60
   6.1 IMU sensor fusion 61
   6.2 Level out on a changing slope 64
      6.2.1 Start and calibrate 64
      6.2.2 Levelling-out strategy 66
      6.2.3 Check constraints 70
   6.3 Mount a slope 71
      6.3.1 Results 72

7 Conclusion 73
   7.1 Achievements 73
   7.2 Further recommendations & research 74

A Manual 76
   A.1 Software on BeagleBone Black 77
   A.2 Software on PC 77

Bibliography 78
List of Figures

1.1 The hexapod robot as it was made available at the start of the masters’ project. ......................................................... 2
1.2 Right: servo enabling hip swing. Middle: servo enabling leg lift. ................................................................. 4
1.3 Time line displaying the necessary steps in the execution of the masters’ project. ......................................................... 8

2.1 AGVs produced by Egemin for material handling in warehouses [45]............................................................ 10
2.2 The quadruped robot called BigDog developed by Boston Dynamics [50]. ........................................................ 12
2.3 Insect leg anatomy and realisation in hexapod robot. ................................................................. 13
2.4 LAURON V hexapod walking up a slope. LAURON V was developed in 2013 at FZI in Karlsruhe, Germany [34] ................................................................. 14
2.5 The laser range finder, a commonly used sensor in mobile robotics. ........................................................ 15
2.6 The two distinct control architectures [31]. ................................................................. 17
2.7 The two distinct mapping approaches. ................................................................. 20
2.8 Large hexapod robots. ................................................................. 23

3.1 A servo and its working principle. The hexapod has got twelve servo-actuated joints. ................................................................. 27
3.2 FSR 400 [37]. ................................................................. 29
3.3 Sensitivity of FSR [20]. ................................................................. 29
3.4 SparkFun triple axis accelerometer and gyro breakout - MPU-6050 [38] ........................................................ 30
3.5 Influence of gravity in a two-axis accelerometer. ................................................................. 31
3.6 An IR PSD and its working principle. ................................................................. 33
3.7 A stereo camera and its working principle. ................................................................. 34
3.8 The hexapod’s processing unit (left) and its servo controller (right). ........................................................ 35
3.9 Power supply: Li-ion battery and DC/DC-converter [45, 46] ................................................................. 36
3.10 Full schematic of the various hexapod robot’s components and their inter-facing. ................................................................. 37
<table>
<thead>
<tr>
<th>Number</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.11</td>
<td>Pictures showing the transition from initial to final status</td>
<td>38</td>
</tr>
<tr>
<td>4.1</td>
<td>Rear view: obtaining rod mechanism of leg</td>
<td>40</td>
</tr>
<tr>
<td>4.2</td>
<td>Forward kinematics</td>
<td>41</td>
</tr>
<tr>
<td>4.3</td>
<td>Inverse kinematics</td>
<td>42</td>
</tr>
<tr>
<td>4.4</td>
<td>Leg indication</td>
<td>43</td>
</tr>
<tr>
<td>4.5</td>
<td>Relationship between pulse width and servo angle from left front knee</td>
<td>43</td>
</tr>
<tr>
<td>4.6</td>
<td>Tripod gait</td>
<td>44</td>
</tr>
<tr>
<td>4.7</td>
<td>Ripple gait sequence [II]</td>
<td>45</td>
</tr>
<tr>
<td>4.8</td>
<td>Wave gait sequence [II]</td>
<td>46</td>
</tr>
<tr>
<td>4.9</td>
<td>Flowchart: tripod straight walking with IMU</td>
<td>47</td>
</tr>
<tr>
<td>4.10</td>
<td>Influence of different gaits, swing size and hip velocity on the velocity of the hexapod</td>
<td>48</td>
</tr>
<tr>
<td>4.11</td>
<td>Turning right over 90° using the IMU</td>
<td>50</td>
</tr>
<tr>
<td>5.1</td>
<td>Flowchart: interchanging between forward backward and turning</td>
<td>53</td>
</tr>
<tr>
<td>5.2</td>
<td>Measured characteristics of IR PSDs</td>
<td>54</td>
</tr>
<tr>
<td>5.3</td>
<td>Obstacle detection and avoidance: test setup</td>
<td>56</td>
</tr>
<tr>
<td>5.4</td>
<td>Obstacle detection and avoidance: flowchart</td>
<td>56</td>
</tr>
<tr>
<td>5.5</td>
<td>Remapping the images</td>
<td>57</td>
</tr>
<tr>
<td>5.6</td>
<td>3D maps</td>
<td>58</td>
</tr>
<tr>
<td>5.7</td>
<td>Detecting the obstacles</td>
<td>59</td>
</tr>
<tr>
<td>6.1</td>
<td>Placement of MPU-6050 on Lil' Hex</td>
<td>61</td>
</tr>
<tr>
<td>6.2</td>
<td>Euler angles applied to the Hexapod and their mutual relations</td>
<td>64</td>
</tr>
<tr>
<td>6.3</td>
<td>Flowchart: Beginning of horizontal levelling-out</td>
<td>65</td>
</tr>
<tr>
<td>6.4</td>
<td>Front view: Level roll principle</td>
<td>68</td>
</tr>
<tr>
<td>6.5</td>
<td>Right side view: Level pitch principle</td>
<td>68</td>
</tr>
<tr>
<td>6.6</td>
<td>Flowchart: Levelling-out strategy</td>
<td>69</td>
</tr>
<tr>
<td>6.7</td>
<td>Flowchart: Check constraints and send to servos</td>
<td>70</td>
</tr>
<tr>
<td>6.8</td>
<td>Body levelling algorithm with Lil' Hex not yet in final design</td>
<td>71</td>
</tr>
<tr>
<td>6.9</td>
<td>Flowchart: Walk up a slope</td>
<td>72</td>
</tr>
</tbody>
</table>
### Used abbreviations

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABS</td>
<td>Acrylonitrile butadiene styrene</td>
</tr>
<tr>
<td>AGV</td>
<td>Automated guided vehicle</td>
</tr>
<tr>
<td>AI</td>
<td>Artificial intelligence</td>
</tr>
<tr>
<td>ANFIS</td>
<td>Adaptive neural fuzzy inference systems</td>
</tr>
<tr>
<td>AUV</td>
<td>Autonomous underwater vehicles</td>
</tr>
<tr>
<td>BBB</td>
<td>BeagleBone Black</td>
</tr>
<tr>
<td>CPG</td>
<td>Central pattern generator</td>
</tr>
<tr>
<td>DMP</td>
<td>Digital motion processor</td>
</tr>
<tr>
<td>DOF</td>
<td>Degree of freedom</td>
</tr>
<tr>
<td>FSR</td>
<td>Force sensitive resistor</td>
</tr>
<tr>
<td>GPS</td>
<td>Global positioning system</td>
</tr>
<tr>
<td>HCI</td>
<td>Human computer interaction</td>
</tr>
<tr>
<td>HSV</td>
<td>Hue-saturation-value</td>
</tr>
<tr>
<td>I2C</td>
<td>Inter-integrated circuit</td>
</tr>
<tr>
<td>IDE</td>
<td>Integrated development environment</td>
</tr>
<tr>
<td>INS</td>
<td>Inertial navigation system</td>
</tr>
<tr>
<td>IR PSD</td>
<td>Infra-red position sensitive device</td>
</tr>
<tr>
<td>IMU</td>
<td>Inertial measurement unit</td>
</tr>
<tr>
<td>MEMS</td>
<td>Micro electromechanical systems</td>
</tr>
<tr>
<td>MPU</td>
<td>Motion processing unit</td>
</tr>
<tr>
<td>NEMS</td>
<td>Nano electromechanical systems</td>
</tr>
<tr>
<td>OPAMP</td>
<td>Operational amplifier</td>
</tr>
<tr>
<td>Abbreviation</td>
<td>Description</td>
</tr>
<tr>
<td>--------------</td>
<td>--------------------------------------------------</td>
</tr>
<tr>
<td>PWM</td>
<td>Pulse width modulation</td>
</tr>
<tr>
<td>RGB</td>
<td>Red-green-blue</td>
</tr>
<tr>
<td>ROS</td>
<td>Robot operating system</td>
</tr>
<tr>
<td>SLAM</td>
<td>Simultaneous localisation and mapping</td>
</tr>
<tr>
<td>SoC</td>
<td>State of charge</td>
</tr>
<tr>
<td>TTL</td>
<td>Transistor-transistor logic</td>
</tr>
<tr>
<td>UART</td>
<td>Universal asynchronous receiver/transmitter</td>
</tr>
<tr>
<td>UAV</td>
<td>Unmanned aerial vehicle</td>
</tr>
<tr>
<td>VGV</td>
<td>Vision guided vehicle</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

The introductory chapter readily provides an overview of what is aimed at with this masters’ thesis and what the corresponding research plan is.

Chapter two takes on autonomous mobile robots in general, hexapods in particular, and the concept of environmental exploration. As such the reader can procure itself the necessary background to approach the rest of this masters’ dissertation.

In the next chapter the hexapod robot under current consideration, named Lil’ Hex, is discussed. An overview of all of its components and capabilities is given. This illustrates the design choices made throughout the realisation of the project.

Subsequently the forward and inverse kinematics of the hexapod are derived yielding insight into the relationship between the servo-actuated joint angles and the leg displacement. Here this knowledge is then used to implement different walking gaits. The actuation implemented throughout the rest of this masters’ project also relies heavily on the deduced kinematic model.

Chapter five then gives an overview of how the hexapod is made possible to deal with ledges and obstacles.

Hereafter an algorithm is derived in chapter six to enable the hexapod robot to level its body horizontally on a slope with varying inclination. This is then used to let it mount slopes with its body levelled out horizontally which greatly benefits the way the robot absorbs the load posed on its rather frail mechanics.

Finally a conclusion and future recommendations are given.
1.1 Start of thesis

The past few years a lot of research has been conducted on the use of multi-legged mobile robots instead of their wheeled counterpart to perform particular operations. Notwithstanding both types of locomotion have their own pros and cons the former is potentially well-suited for traversing rough, irregular terrain. One particularly interesting limbed vehicle especially with regard to stability is a six-legged robot or so-called hexapod robot. Possible applications of such a robot might be the exploration of extraterrestrial and volcanic landscapes, underground mining, logging...

As a remnant of a robotics project carried out about then years ago the department of Electrical Energy, Systems & Automation happens to have a rather basic hexapod at its disposal (figure 1.1). Given the field of mechatronics has continued to evolve significantly in the past decade it seems worthwhile to revive the old hexapod using currently available computing power, sensors and actuators. As such the department expresses the desire to bring the six-legged robot to life and provide it with a certain degree of autonomy excluding any form of teleoperation. This would pave the way for further research on the available hexapod.

Figure 1.1: The hexapod robot as it was made available at the start of the masters’ project.
1.2 Initial status

To furnish the reader with the necessary background on the starting point of this masters’ project a brief description of the initial situation is given:

The frame of the hexapod, measuring 300 by 200 by 50 mm, is fabricated out of some sort of plastic, possibly ABS. They have chosen a traditional inline hexapod design\(^1\) (figure 1.1). Each of the six hexapod legs is equipped with two Standard Futaba S-3111 servos enabling a forward/backward hip swing and an up/down leg lift (2 DOFs/leg). The latter movement is realised through the mechanism depicted in the picture given on the next page (figure 1.2). The available servos need to be tested for proper working and if necessary replacements will be made.

A so-called EyeBot-Controller M5 was provided as the brains of the hexapod. This single-board embedded computer produced by Joker Robotics back in 2005 has now - ten years later - become completely obsolete. Therefore the need for a new embedded computer board is irrefutable. Nowadays several of such boards are available on the market (Raspberry Pi, Arduino, Dwengo, Beaglebone...). Furthermore some expansion board may be required to be able to control all of the twelve servos using pulse width modulation.

As to sensors, the only ones made available at the time were three Sharp GP2D02 IR PSDs for measuring distances within a range of 10 to 90 cm and a medium resolution EyeCam C2 colour camera. To obtain increased sensing capabilities one might consider replacing the single EyeCam C2 with stereo camera and adding some other sensors such as an IMU for acquiring body orientation and FSRs to detect foot-ground contact.

Finally to be able to power all of the aforementioned equipment suitable batteries will need to be purchased and of course all of the components need to be wired properly.

\(^1\)Unfortunately no technical drawings or documentation whatsoever are available.
1.3 Objective

1.3.1 Main objective

As mentioned in section 1.1 the aim of this masters’ project is to revive the given hexapod using currently available computing power, sensors and actuators and to provide it with a certain amount of autonomy. More specifically the robot should be given the fundamental building blocks to roam around certain predefined environments on its own without putting itself into harm’s way. This involves for instance the ability to overcome small rubble, detect and avoid larger obstacles and ledges, walk up a slope... The terms ‘on its own’ refer to the fact that no explicit human control over its movements (like remote control or teleoperation) is allowed. Consequently the limbed craft should be able to interact with its environment just by evaluating its own on-board sensors. This implies a core focus on sensing and acting within a given environment.

\[2\text{Given the far from sturdy design of the available hexapod robot (i.e. fabricated out of plastic, the use of simple hobbyist servos for the leg actuation and the limited number of DOFs per leg) the environments the hexapod can deal with are of course rather restricted.}\]
1.3 Objective

The realisation of this masters’ project can then serve as a platform for further research at the department of Electrical Energy, Systems & Automation concerning hexapods and mobile robots in general. For instance an obvious next step in the pursuit of environmental exploration would be granting the robot the capability to navigate through space in an intelligent, deliberate way. This is what is referred to in literature as the issue of localisation, mapping and path planning (as discussed in chapter 2, section 2.3). Other possible research topics could be the implementation of CPGs for locomotion, energy-efficient walking, or the use of robot learning techniques such as neural networks, reinforcement learning and/or evolutionary algorithms to perform certain tasks.

1.3.2 Subgoals

Clearly the objective described above remains rather broad. Therefore to avoid any ambiguity the following subgoals have been posed:

1. **Obtain a fully operational hexapod robot.** This comes down to determining all of the hardware components that need to be purchased (such as a single-board embedded computer, possible expansion boards, servos, desired sensors, batteries...), exploring their proper connection and working, and putting it all together into a fully equipped limbed robot.

2. **Get acquainted with embedded programming.** This may require installing a Linux distribution and/or an IDE such as Eclipse or Microsoft Visual Studios.

3. **Study six-legged locomotion.** This implies implementing some possible walking gaits and investigating their properties in terms of stability and speed. Doing so insight can be gathered as to which gaits and speed settings are advised in different types of terrains (flat, inclined, rugged, slippery...).

4. **Add sensor evaluation to the mix.** This is the final step where all of the acquired sensors are brought to use enabling the hexapod robot to walk around autonomously without putting itself into harm’s way. In particular the robot should be able to deal with obstacles, ledges and slopes.
1.4 Research plan

1.4.1 Problem statement

‘Which steps are required to make the given hexapod robot fully operational and able to wander certain predefined environments autonomously?’

1.4.2 Research questions

To support the problem statement one can derive a number of research questions. From these questions it becomes clear that besides some design choices a mainly descriptive investigation will be conducted:

- Which steps and components are generally required to grant the hexapod the ability of locomotion and what type of gaits are there? (Chapter 2 & 4)

- What type of sensors are available and which are commonly used in mobile robot applications? (Chapter 2 & 3)

- Given its design and functionalities which environments should the limbed robot be capable of dealing with? (Chapter 3, 4, 5 & 6)

- What are the traditional approaches to converting a robot’s sensory percepts to well-chosen actions within its environment? (Chapter 2)

- What levels of autonomy are currently employed in real-life mobile robots? (Chapter 2)

1.4.3 Relevance

The above clearly shows the problem statement is in complete support of the desired, rather practical objective described in section 1.3. Consequently the conducted research can go hand in hand with the practical achievement of the masters’ project. By immersing itself in the field of mobile robotics a continuously improving image is acquired of the different actions this assignment actually demands.

\[^{3}\text{As mentioned before the far from sturdy design of the available hexapod robot (i.e. fabricated out of plastic, the use of simple hobbyist servos for the leg actuation and the limited number of DOFs per leg) greatly restricts the environments the hexapod can deal with.}\]
1.4 Research plan

1.4.4 Research method

To start off a thorough study of mobile robots and hexapods in particular is indispensable. Moreover, lacking a computer science background, learning how to program an embedded computer is an absolute prerequisite. For this preliminary study several sources can be consulted: the internet, existing master’s dissertations and scientific papers, books and possibly even specialists in the field of legged robots.

Once sufficient background information has been gathered a verification of the status quo at the start of the project is in place (cfr. section 1.2). Hereby it is important to note that certain data and dimensions may be difficult to acquire (e.g. no technical drawings available, hard or non-measurable dimensions). As a consequence judicious estimations may have to be made.

Next, strengthened by the conducted literature study, one can decide on what components to order to get from this initial situation to the desired goal. So at this point all of the required hardware components are purchased such as the single-board embedded computer, the expansion boards, the sensors, the batteries, extra utilities...

After consulting the different components’ data sheets and getting acquainted with a suitable programming environment (IDE and/or Linux distribution) a clear view emerges on what type of connections should be made and how possible data/actuation is obtained. Subsequently the realisation of the main objective can be addressed. As discussed in section 1.3.2 this will be done in steps: starting with a focus on pure actuation and gradually bringing sensors to the mix.
1.4.5 Planning

Now it is possible to construct a time line displaying the various steps mentioned in the previous subsection in an orderly fashion (figure 1.3). It is important to note however that a strict, chronological demarcation of the different steps would not be in correspondence with the way the research process actually evolves. Namely it might happen that certain aspects are treated somewhat simultaneously or get revisited in a later stage. With this in mind the time line merely serves as a guidance.

As spoken of, providing the department with a hexapod robot capable of interacting with its environment (both sensing and acting) is the goal for this academic year. Accomplishing this elementary first step will facilitate any further research concerning hexapods and mobile robots in general. As such the scope of this project can reach far beyond this year alone.

![Figure 1.3: Time line displaying the necessary steps in the execution of the masters’ project.](image-url)
Chapter 2

 Autonomous mobile robots

Spurred by the enormous improvements in computer processing power, the field of robotics has flourished tremendously over the past few decades. Besides the traditional robot manipulator arms which have already reached a high state of maturity, considerate attention is being spent to the creation of mobile robots and endowing these with ever higher degrees of autonomy. From a largely dominant industrial focus, robotics is rapidly expanding into human environments and vigorously engaged in its new challenges. Interacting with, assisting, serving, and exploring with humans, the emerging robots will increasingly touch people and their lives (Kaneko et al., 2010) [2]. As Dudek and Jenkin (2010) pointed out, for a mobile robot to behave intelligently in large-scale environments not only implies dealing with the incremental acquisition of knowledge, the estimation of positional error, the ability to recognize important or familiar objects or places, and real-time response, but it also requires that all of these functionalities be exhibited in concert [4]. What’s more the nature of the environment in which the mobile robot is to be used highly influences the exactingness placed on its design. Clearly letting a mobile robot cope in an unknown and possibly dynamic outdoor environment (i.e. with moving objects and possible interaction with living beings) poses significantly greater challenge than applying it in a well-known rather static indoor environment.

This chapter should be regarded as a means of procuring the reader with the necessary background to approach the masters’ dissertation at hand. To avoid overloading the overview many topics and terms are merely touched upon leaving those who are interested free to further investigation.
2.1 Autonomy

As mentioned before contemporary robotics engineers are relentlessly aiming at increasing the amount of autonomy or self-willingness displayed by the mobile robot. As stated by Bekey (2005) autonomous mobile robots are intelligent machines capable of performing tasks in the world by themselves, without explicit human control over their movements [3]. They are programmed to act and make decisions based on sensory feedback of their external and/or internal surroundings. Additionally, they should be reliable, tolerant to various system malfunctions and have the ability to adjust dynamically to changes in their environment (Jakimovski, 2011) [1]. The first mobile robots exhibiting a fair amount of autonomy were the so-called AGVs, already introduced for the first time back in 1953. These mobile vehicles are most often used in industrial applications to move materials around a manufacturing facility or warehouse as shown by figure 2.1 [47, 48]. They follow markers or wires in the floor or use vision, magnets or lasers for navigation. Given the fact they require specific modifications to the environment or infrastructure in which they operate, AGVs are actually only autonomous within the strict confines of their predefined operating environment. Today highly autonomous robots exist (both for in as for outdoor applications) relinquishing the need for any structural changes to their operating space (section 2.4). These possess a wide range of sophisticated sensors for measuring both external as internal state and are capable of exploring possibly unknown environments by their selves. To conclude this section one should note that several mobile robot manufacturers offer their clients the ability to manually override the mobile robot control system enabling remote control or teleoperation whenever this is wished for. This is referred to as so-called sliding autonomy.

![AGVs produced by Egemin for material handling in warehouses](image)

**Figure 2.1:** AGVs produced by Egemin for material handling in warehouses [48].
2.2 Mobile robots

A mobile robot generally comprises four main capabilities: sensing, reasoning, locomotion and communication. The robot’s sensory system is needed to gain information about its environment (both internal as external) whereas its actuators are used to exert forces upon the environment generally resulting in motion of the entire robot or some of its parts (wheels, limbs...). Furthermore embedded computers serve as the brain of the robot enabling the processing of sensor outputs, reasoning, localisation, navigation, obstacle avoidance, task performance... Finally it is highly desirable to have the ability to communicate with a base station, with one another and/or with humans. For this purpose earlier robot communication systems mainly employed radio frequency transmitters and receivers, while nowadays most mobile robots working in both indoor and outdoor environments use wireless Ethernet communication (Wi-Fi) [3].

Of course bestowing the robot with such capabilities also requires some sort of on-board power supply. Nowadays the vast majority of autonomous systems rely on (rechargeable) batteries, although in some cases alternators driven by internal combustion engines are used too.

2.2.1 Locomotion

Over the years scientists and engineers all over the world have procured mobile robots with the most diverse motive systems, some of which strongly inspired by nature itself. Besides the classical approach of using wheels and/or tracks, robots have been designed to walk, run, crawl, roll, slither, jump, climb, swim or even fly (cfr. quad copters [58], UAVs...). The choice of which locomotion method to use is mainly driven by the application and the domain in which the mobile robot should operate. Other important factors are of course energy consumption, complexity and cost.

Limbed robots

In regard of the subject of this masters’ thesis some special attention to limbed mobile robots is in place. As pointed out by Bekey (2005) what’s most interesting about the locomotion used by animals and humans is their astounding adaptability and versatility to overcome any kind of terrain. Contrary to their wheeled counterpart legged machines do not require a continuous, unbroken support path. Legs make it possible to move on smooth or rough terrain, to climb stairs, to avoid or step over obstacles, and to move at
various speeds. Consequently already from the early days of robotics walking machines with four, six or eight legs have been proposed for operations on rough terrain or in hostile environment. However despite of the plethora of multi-legged vehicles in research and even hobbyist environments, up until today very few legged robots have actually found permanent application in that specific field (cfr. Bigdog shown in figure 2.2, RHex). This is largely due to the fact that in the vast majority of practical applications, wheeled or tracked vehicles still have competitive advantage over legged ones in terms of power requirements and complexity and provide lower-cost solutions to the need for locomotion, even in difficult environments like for instance the lunar surface (Bekey, 2005) [3]. Moreover reaching the same flexibility and adaptability in leg placement like living beings do seemingly effortlessly remains a serious hurdle to overcome.

![BigDog climbing rubble.](image1)
![BigDog walking at the beach.](image2)

**Figure 2.2:** The quadruped robot called BigDog developed by Boston Dynamics [50].

As already indicated above one of the choices the robotics engineer has to make is the number of legs to provide the limbed robot with. Clearly this choice greatly influences the stability properties displayed by the robot. Biped walking machines are only dynamically stable (requiring the implementation of balancing control techniques) whereas crafts with more than two legs (i.e. four, six, eight) have the ability of moving in a statically stable manner [16]. Consequently keeping a moving robot from falling over is far less demanding when having more than two legs.
Six-legged robots

As it happens the leading actor of this thesis project is a six-legged or so-called hexapod robot. What makes these insect-like machines so appealing is the fact that they possess a multitude of walking gaits in which static stability is maintained. This greatly eases the task of keeping the robot in the upright position at all times.

![Insect leg anatomy. Hexapod robot leg with 3DOF.](image)

Figure 2.3: Insect leg anatomy and realisation in hexapod robot.

Figure 2.3a shows the anatomy of an insect’s leg. The general approach used in hexapod robots is providing the leg with two to three servo-actuated rotational joints (figure 2.3b). Using three instead of two servos yields much closer imitation of the real insect’s leg and results in higher manoeuvrability (e.g. enable sideways walking, negotiating stairs...) albeit at the expense of a larger energy consumption and control complexity.

By careful study of insects’ walking patterns the designers of six-legged walking machines are then able to program the succession of leg placements needed to achieve a certain motion. Typically this requires an inverse kinematics model to be devised first such that the programmer knows what type of servo inputs furnish the desired leg displacements as output. Another approach to the generation of the motor patterns required for different kinds of gaits is trying to emulate the neural-control mechanisms or so-called CPGs that are actually involved in animal locomotion. Having been the subject of extensive research over the past few years, a lot has been written regarding CPGs and their implementation in legged robots. For an in-depth review of CPG models for locomotion control in animals
and robots one can consult [11]. On fairly flat terrain the aforementioned approaches can be applied without further ado. But in order to overcome irregular, rocky terrain or obstacles in general, legs have to be placed in an adaptive aperiodic manner. This requires integrating certain sensors that give the robot an idea on how and where to place its legs next.

To finish this subsection one should emphasise that the discussed six-legged robots have been the target of countless studies in research facilities and universities all over the world. An illustrative listing of the numerous hexapod robots around (up until 2005) and their distinct research purposes is given by Bekey (2005) [3]. Particularly interesting is the use of hexapod robots as platforms for robot learning techniques such as learning using evolutionary algorithms, reinforcement learning and neural-network-based learning (e.g. Rodney, LAURON...).

Figure 2.4: LAURON V hexapod walking up a slope. LAURON V was developed in 2013 at FZI in Karlsruhe, Germany [34].


2.2 Mobile robots

2.2.2 Sensing

Sensors in robotic applications

As stated in the introductory paragraph of section 2.2 robots need the ability to gain information about their environment. The most fundamental classification of sensor types is into proprioceptive (internal) and exteroceptive (external) sensing. Internal-state sensors provide information on the internal parameters of a robotic system such as the battery level (measured via SoC methods), the number of wheel rotations (via synchros, resolvers or encoders), the joint angles (via potentiometers or encoders), the payload, the internal temperature... External-state sensors on the other hand are used to monitor the world outside the robot itself (e.g. vision, audition, touch...).

Commonly used sensors in mobile robotics are: tactile sensors (e.g. limit switches, FSRs), proximity sensors, distance sensors (infra-red, ultrasonic or laser range finders), inertial sensors (accelerometers and gyroscopes possibly fused in IMUs), tilt sensors (inclinometers and compasses), positioning sensors (e.g. GPS for outdoor robot applications) and cameras. As a result of the enormous research in computer vision (cfr. the OpenCV library), cameras are steadily offering the mobile robot a true extension of its sensing capabilities. Besides detecting shapes or colours they enable depth perception (stereo vision), visual odometry, facial recognition, gesture recognition, object identification, motion tracking, HCI etc. Of course depending on the application of the mobile robot other sensors may be added too such as light sensors, sound sensors, temperature sensors, current sensors...

(a) SICK laser range finder [33].

(b) Principle of operation.

Figure 2.5: The laser range finder, a commonly used sensor in mobile robotics.
2.2 Mobile robots

Odometry

What’s specifically important for mobile robots is the ability to estimate their travelled distance within a given environment. This so-called odometry can be performed in several possible ways, some more favourable than others. For instance the traditional approach for wheeled robots is counting the number of wheel revolutions and multiplying by the wheels’ diameter. Other less advocated methods one could think of is integrating velocity or acceleration info received by IMUs\(^1\) or, more specific for limbed robots, counting the number of steps knowing the step length. Unfortunately all of these techniques, and especially the latter two, are prone to errors (slippage, unequal wheel diameters or step lengths, integration of high-frequency noise...) and unavoidably lead to imprecise position estimates drifting off over time. For this reason odometry information is only useful when the robot is capable of periodically updating its position by means of some external reference (exteroceptive sensing).

Over recent years developments in robotics and computer vision have led to another form of odometry known as visual odometry. Hereby sequential camera images are processed to estimate the distance travelled by the robot. This camera-based technique allows for improved navigational accuracy in robots or vehicles using any type of locomotion on any surface. As such it provides a valuable odometry alternative for legged mobile robots which obviously can’t make use of the traditional approach used by wheeled ones.

Data fusion

As described by Dudek and Jenkin (2010) the question of how to combine measurement data from different sensors, positions and/or times is a major and extensively examined research issue. In essence fusing data from different sensors, positions and/or times yields enhanced measurement estimates. One example in mobile robotics where data fusion techniques prove particularly useful is the process of maintaining an ongoing estimate of the position of a robot relative to some external set of landmarks or features (see further in section 2.3.2). Although various approaches have been developed to guide the process of data fusion, two distinct but related techniques have become pre-eminent in mobile robotics, namely approaches based on the classic least squares combination of estimates (Kalman filter, extended Kalman filter) and approaches that utilize a less strong error model (Markov localisation, discrete grid representations, Monte Carlo and condensation techniques). For a comprehensive treatment of these methods a referral to \[4\] is in place.

\(^1\)Only done when using a very costly so-called INS e.g. in robot aircrafts and underwater vehicles \[3\].
2.2 Mobile robots

2.2.3 Control

Control architectures

From a hardware point of view, contemporary mobile robots use computationally powerful, on-board processors to process sensory inputs, to generate commands to the actuators, and to perform such cognitive functions as reasoning and planning. On the software level a lot of attention has been given to the specific (software) architecture used to transform a robot’s sensory percepts to well-chosen actions within its environment. Over the years two fundamentally distinct architectures have been coined. On the one hand there is the hierarchical, deliberative architecture (also referred to as the ‘Sense-(Model)-Plan-Act’ paradigm) and on the other there is the reactive, behaviour-based architecture proposed by Brooks in 1986. The former approach is characterised by a hierarchically layered structure in which each layer provides subgoals (or explicit instructions) to the next layer (figure 2.6a). Perception is used to modify and update the world model, so that action is produced by planning and reasoning from the model rather than directly from the perception. In other words, the robot senses and then thinks before it can move [3].

Unfortunately this approach is less adequate in dealing with rapidly changing environments. In order to address this issue Brooks proposed a reactive, behaviour-based architecture characterised by a close coupling of perception and action, without an intermediate

![Diagram](image-url)
cognitive layer. Rather than using the ‘horizontal’ structure of the Sense-Plan-Act model, he suggested a vertical decomposition in terms of separate behaviours which could be initiated in parallel dependent on the specific sensory inputs (figure 2.6b). The higher, more complex behaviours subsume those beneath them (hence subsumption architecture sometimes coined) [3]. Alas purely reactive architectures also have some shortcomings especially when it comes to more complex tasks. As a result contemporary mobile robots commonly use hybrid versions of the two types of architecture in an attempt to combine the best of both worlds. See [3, 4] for a more elaborate discussion on control architectures for autonomous robots.

**Low-level vs high-level control**

Approaching the subject from a more control engineering perspective, a mobile robot comprises different levels of control. At the lowest level (i.e. executonal control, e.g. controlling desired leg or wheel movement, balancing...) the fundamental concepts of linear and/or non-linear control theory can be applied. For instance a (linear) feedback controller commonly used in robotics is the PID-controller, but depending on the system to be controlled other more advanced control principles such as adaptive, nonlinear and/or optimal control techniques may be required. A prerequisite for the foregoing control methods is the availability of mathematical models of the system being controlled.

The upper layer of control implemented in mobile robots is then concerned with task planning (i.e. strategic or task control). Up until today a variety of high-level control formalisms have been devised. The most important ones currently applied are artificial neural networks, fuzzy logic and hybrid approaches like ANFIS. The use of artificial neural networks is a result of ongoing research in machine learning and cognitive science attempting to adopt human-like cognitive skills (like the ability to learn from experience). Fuzzy logic is essentially a rule-based control implementation that approaches the way us people make decisions (e.g. If... Then...). A large research literature exists on these control formalisms and their application to autonomous vehicle control [7, 8, 12, 13, 14, 15].
2.3 Environmental exploration

As discussed in the introductory chapter of this masters’ dissertation, a first step to environmental exploration is gifting the mobile robot the skill set to wander about in space without putting itself into harm’s way (i.e. detect and avoid obstacles, ledges, deal with slopes...). Once these fundamental capabilities have been provided one can focus on enabling the robot to move in an intelligent, well-considered way throughout its environment. This requires the robot to have a sense of what its environment looks like, where the robot itself is situated within this environment and how to traverse it in the best possible way (according to a particular criterion). These issues are commonly referred to as mapping, localisation and navigation or path planning. As a result of a vast literature study initiated already before the millennium transition and improvements in sensing and computing capabilities, the three mentioned exploration pillars have reached a more than acceptable performance level in most of the currently employed real-life applications (section 2.4).

2.3.1 Mapping

To render environmental exploration possible the availability of an explicit representation of the environment being traversed - be it in or outdoor - is indispensable. In a priori unknown environments it is highly desirable the mobile robot can create such so called mapping by itself using exteroceptive sensors (such as computer vision, sonars and/or laser range finders) in conjunction with the earlier discussed odometry information obtained via proprioceptive sensing (section 2.2.2). This topic, referred to as SLAM, has been studied extensively the past two decades and has now reached a high state of maturity as is demonstrated by the ROS open-source community providing off-the-shelf SLAM libraries [29, 30, 32]. That being said performing outdoor mappings still poses significantly greater challenge than its indoor counterpart. Finally, when the environment is known (e.g. a hospital, warehouse, industrial grounds...) the robot could be relieved of this mapping task by storing a map of the terrain in its memory beforehand.

To complete this concise discussion on mappings it remains to be noted there are two traditional approaches to representing a robot’s configuration space: metric mapping (cfr. 2 or 3D occupancy grids, geometric maps) and topological mapping (figure 2.7). [4] provides an elaborate discussion on these two ways of portraying space.
2.3 Environmental exploration

(a) A 2D occupancy grid [29].

(b) A topological map [28].

Figure 2.7: The two distinct mapping approaches.

2.3.2 Localisation

Localisation, pose estimation or positioning is concerned with letting the robot answer the question ‘Where am I?’ Perhaps hearing this, one immediately comes to think of using GPS to furnish the robot with very precise positional information. But unfortunately global positioning techniques can only be applied in outdoor Earthly environments. Typically performing localisation urges the robot to have a map of the environment at its disposal (see 2.3.1). Again by using the earlier outlined odometry information (section 2.2.2) in combination with exteroceptive sensors for landmark detection the robot is then able to maintain an ongoing estimate of its pose within this environment. The keyword here is estimate. Mapping, localisation and navigation are probabilistic problems. It is impossible to know a robot’s pose exactly, given imperfect sensors and imperfect knowledge of the robot’s environment. Therefore one can only determine the probability that the robot is in a location given a set of sensor readings. For this reason mapping, localisation and navigation rely on statistical procedures like the ones mentioned in the paragraph on data fusion (section 2.2.2) to get enhanced estimation accuracy when combining info from internal and external sensors. It should be brought to attention that such procedures generally require good models of the sensors and their uncertainty.

\[^2\]Sensors such as sonars, stereo cameras and laser range finders yield distance to the perceived landmarks and by applying triangulation and/or trilateration techniques the robot can then pinpoint its exact position.
2.3.3 Navigation

For a robot to navigate throughout its environment purposefully it should be granted the explicit ability to determine and maintain a path or trajectory to a desired destination. Finding this best path (according to a particular criterion) from start to goal while avoiding obstacles and bearing in mind the kinematic constraints of the vehicle is termed path or trajectory planning. This actively studied robotics topic starts from the premise that a map of the environment is available and the robot is able to localise itself within this map (see the above paragraphs). In literature many path planning methods have been proposed. The main demarcation is between discrete (relying on graph-based, topological maps e.g. Dijkstra’s algorithm, A and A*, D*, dynamic programming…) and continuous methods (e.g. potential fields, bug algorithms…). For the interested reader [4] presents an excellent overview of many such path planning algorithms.
2.4 Applications

What follows is a brief overview of some of today’s mobile robot manufacturers and applications. The given overview is far from complete and merely serves as a reference of contemporary state-of-the-art.

2.4.1 Mobile robots in general

Autonomous mobile robots are especially well suited for tasks that exhibit one or more of the following characteristics: dirty, dull, distant, dangerous or repetitive. That being said, there are numerous classes of mobile robots currently in operation with varying degrees of autonomy. Although many of them are fundamentally research vehicles and thus experimental in nature (universities, space agencies...), a substantial number are deployed in other contexts as well. For instance one of the world’s leading manufacturers of intelligent mobile platforms for education, research and industry is Adept MobileRobots [49]. Their focus is on wheeled mobile robots used for both in and outdoor exploration of possibly unknown environments, material transport and security and surveillance. Next particularly interesting firm is Bossa Nova Robotics [23], developer of a ballbot personal robot named mObi to assist humans in daily, dull tasks. Another company producing advanced dynamic robots with remarkable behavior regarding mobility, agility, dexterity and speed is Boston Dynamics [50]. They offer high-end humanoid, quadruped and hexapod robots for research and military operations (cfr. reconnaissance and material transportation through hostile, possibly inhospitable terrain). On the other hand several firms like Honda and Pal Robotics create humanoid robots with high entertainment value (cfr. ASIMO, REEM-C) [54, 57]. For an overview of ASIMO’s skill set one can consult the Honda Robotics website [54]. It should be noted however that today’s anthropomorphic robots, although already exhibiting quite compelling behaviour, are still nowhere near the capabilities as portrayed in contemporary science fiction movies.

Continuing the overview, AsiRobots [51] is a world leader in vehicle automation delivering mobile robots with varying degrees of autonomy for mining, agriculture, automotive and research applications. Furthermore Seegrid [24] delivers so called VGVs, the next generation of AGVs for material handling throughout manufacturing and distribution facilities, whereas Harvest Automation [25] makes autonomous mobile robots for material handling tasks in unstructured, challenging environments such as those found in agriculture and construction. Next, Kiva Systems, a subsidiary of Amazon, offers the Kiva Mobile Robotic Warehouse Automation System for stock transportation [52] and Aethon
for instance produces the TUG mobile robot to perform delivery and transportation tasks in hospitals (freeing up clinical and service staff to focus on patient care) \[9\] [53]. Other currently deployed mobile robots are autonomous cleaning machines both in domestic (e.g. \textit{iRobot}'s Roomba, Scooba, Braava and Looj [55]) and industrial settings (e.g. \textit{Intelli-BotRobotics}'s Hydrobot, Aerobot and Duobot [56]).

Finally, autonomous mobile robots have also found application in sky and water. 3DR [58] for instance delivers cutting-edge UAVs to overfly and map any area using on-board cameras. The \textit{ECAgroup} [27] and \textit{Bluefin Robotics} [26] on the other hand manufacture AUVs for defense, commercial and scientific customers worldwide.

### 2.4.2 Six-legged robots in particular

As explained in section 2.2.1 legged mobile robots haven’t really found permanent applications yet. There was the impressive \textit{Plusjack Walking Harvester} (figure 2.8a) developed in 1999 by Plustech Oy, the R&D unit of the Finnish company Timberjack (now a subsidiary of John Deere). This prototype hexapod robot was designed for logging on rough mountain terrain and showed great promise [59]. Nevertheless it never made it into production and was donated to the Finnish Forest Museum Lusto in Punkaharju Finland in 2011 [60]. Besides this there is the noteworthy \textit{Mantis} robot build by Micromagic Systems in 2012 (figure 2.8b) [61, 62]. The Mantis is the biggest, all-terrain hexapod robot currently operational in the world. However it merely serves an entertaining purpose being available for private hire, custom commissions, events and sponsorship.

(a) The Plusjack Walking Harvester [59]. (b) The Mantis hexapod robot [61, 62].

\textbf{Figure 2.8:} Large hexapod robots.
Furthermore over the past few years a small market has arisen in which some manufacturers now offer turnkey hexapod robots for hobbyist purposes. These hexapods come with different designs and options (inline or circular body design, two or three DOFs per leg...) but quite often only involve teleoperation or open loop actuation.

2.5 Further evolutions

As portrayed in the previous section autonomous mobile robots will relentlessly keep expanding their field of application and as such come in ever closer contact with our every day lives. Further research and developments can be expected in all important aspects of mobile crafts (locomotion, sensing, cognition, navigation...). Advanced sensor systems will significantly increase the capabilities of autonomous vehicles and will enlarge their application potential. The pursuit of continually higher degrees of autonomy for increased reliability and robustness will lead to developments linked to buzz words such as self-monitoring, self-maintenance and repair, adaptability, reconfigurability, compliance...

Another important field of continued study will be the concept of multiple robots working together alongside humans to achieve certain tasks (cfr. swarms, colonies...), sharing information with each other and their supervisor. Furthermore as MEMS and NEMS technology is reaching new heights it can be expected that this will lead to astounding miniature mobile robotic applications.

More specifically focusing on those mobile robots inspired by nature, major research will go on into two distinct fields: approaching the same rich, dexterous motor skills as displayed by humans and animals (i.e. locomotion) and trying to imitate human cognition and intelligence (i.e. learning, planning, reasoning...). The latter is driven by developments in AI (cfr. machine learning, robot learning, deep learning...). As such (legged) mobile robots will become more suitable to perform a large number of tasks, able to adapt to new and unstructured environments and also have the ability to learn or gain new knowledge like adjusting for new methods of accomplishing its tasks. Be it as it may, the dawn of a new era in mobile robotics is near and biological inspiration will continue to be one of the main driving forces.
Chapter 3

Lil’ Hex

Once a thorough literature study has been performed one is able to determine necessary and useful components in order to achieve this project’s goal (section 1.3). This chapter provides an overview of all of the components the available hexapod robot has been equipped with. First a discussion on the hexapod frame and the actuators is given. Next a closer look is given to the purchased sensors, being force sensitive resistors, an inertial measurement unit, infra-red position sensitive devices and a stereo camera. Lastly the obtained processing unit and the utterly important power supplies are discussed.

After stripping the available hexapod robot of his former components, leaving only the naked frame, these newly bought parts can then be mounted. When everything is finally put together into its final design and the limbed robot is fully operational, it deserves a name of its own. It is named Lil’ Hex.
3.1 Hexapod frame

First off, to have an idea of what the given hexapod robot and its leg mechanisms look like a referral to section 1.2 of the introductory chapter is in place. Basically the only part that is retained from the original hexapod is its plastic body frame and leg mechanisms. This implies the number of DOFs per leg remains fixed to two. From the discussion in subsection 2.2.1 it is evident that this poses constraints on the maximum manoeuvrability that can be attained (i.e. no sideways walking possible, less flexibility in overcoming certain obstacles...). Nonetheless expanding from two to three servo-actuated rotational joints like shown in figure 2.3b would come at the cost of higher energy consumption and increased leg control complexity. Obviously this implies a trade-off that has to be made.

Another important issue is the fact that the available hexapod frame and leg mechanisms are fabricated out of plastic (possibly ABS) and that the leg actuation is realised by simple hobbyist servos (see next section). This far from sturdy design of course greatly confines the environments the hexapod robot can actually deal with. Anyhow it is clear that the aforementioned mechanical limitations need to be kept in mind when performing real-life experiments.

3.2 Actuators

Based on the mechanical design a total of twelve servos are needed in order to move the robot with two DOFs per leg. Originally, these servos were already on-board and only the ones broken needed to be replaced.

To be precise the original actuators the robot was equipped with are Standard Futaba S-3111 servos which have the following specifications: a setting torque of 37 Ncm and a positioning rate of 0,15s/60° at an operating voltage of 6V [19]. The choice of the new actuators is based on these properties and are the Futaba 1-S3151 servos (figure 3.1a) which have a setting torque of 39 Ncm and a positioning rate of 0,17s/60° at the same operating voltage of 6V [35].

Concerning the inequality of the positioning rates a replacement of all the servos could be a valuable adaptation so that a more symmetric walking gait could be obtained. However as will be depicted in one of the following sections (section 4.7) the speed will also be managed on a software level, thus making these changes unnecessary.
As stated by Braunl [5], there is a significant difference when talking about servos instead of servomotors. A servomotor is defined as a high-quality DC motor that allows for precise control of angular position, velocity and acceleration, i.e. in a closed loop. It consists of a suitable motor coupled to a sensor for position feedback. A servo, like the Futaba S-3111, is a DC motor with some simple internal electronics for pulse width control. These motors only use position sensing via a potentiometer as an elementary part of their working principle (as described in the following subsection); the position measurement is not made available to the user, ruling out any form of feedback control.

![Futaba S3151 servo](image1.png)

**Figure 3.1:** A servo and its working principle. The hexapod has got twelve servo-actuated joints.

The servo has three wires: Vcc, ground and a control wire. A PWM signal which specifies the desired shaft position needs to be sent through the control wire of the motor. Most commonly, this is done with a frequency of 50 Hz (thus a period of 20 ms) and a pulse width between 0.5 ms and 1.5 ms, where the shaft’s angular position is based on the width of this electric pulse. Figure 3.1b for instance shows a pulse width ranging from 1 ms to 2 ms and the corresponding shaft positions. Internally, the PWM signal is decoded into a reference voltage, where after it is compared with the potentiometer’s voltage which depends on the shaft position. The direction of the rotation is chosen based on the sign of the voltage difference. If no signal is read-in no comparison will be made and thus no torque will be applied, even though the motor is supplied with power.

These servos are a good and simple solution for the tasks the robot should be able to do for now. Although, they do have the drawback that they don’t allow the user to know whether the servo actually reached the preferred position. The ability to obtain information on the
real servo angle is an absolute prerequisite to be able to apply some form of feedback control (like a PID-controller) on the leg placements.

In some of the applications of the hexapod robot, multiple motors could be controlled by the same PWM signal (e.g. the tripod gait), but in order to remain as flexible as possible a unique control signal for each servomotor is required. Moreover all servos should be supplied with a control signal at all times, so that these are always provided with a holding torque. Based on this, a processing unit which is capable of providing twelve independent PWM signals is desired (see section 3.4).

3.3 Sensors

An autonomous mobile robot needs senses of its own to be able to gain information about its environment (both internal as external). In regard of what is aimed at with this masters’ project (section 1.3) the following sensors have been allotted to the hexapod robot: FSRs, an IMU, IR PSDs and a stereo camera. These are discussed successively in the following subsections. Important aspects one should bear in mind when working with real-life sensors for mobile robot applications is the fact that they are noisy, return an incomplete description of the environment and cannot usually be modelled completely. In the knowledge that performing actions upon a faulty sensory read-in could have fatal consequences, it is absolutely imperative that sufficient attention is being spent to proper sensor evaluation.

3.3.1 Force sensitive resistor

Throughout the exploration of an unknown environment, one of the unwanted events that could arise is to place a leg without making ground contact when needed. This situation could lead to a stumble with undesirable consequences. To avoid such a possible disastrous event, a tactile sensor is attached at the bottom of each leg.

The applied tactile sensors are the FSRs 400 of Interlink Electronics (figure 3.2) which are often used in robotic applications. The dimensions of the sensors are chosen so that these can be attached cleanly onto the legs. The force sensitivity range is between 0.2 N and 20 N [20] and the weight of the robot is about 2 kg (= 20 N). Clearly, these sensors will suffice knowing that one leg is never carrying all of the weight (there are minimally three legs on the ground).
The working principle of these small tactile sensors is quite straightforward. Essentially these polymer thick film devices exhibit a decrease in resistance with increase in force applied to the surface of the sensor. By integrating the FSR in a voltage divider (with optional OPAMP) one obtains an analog output voltage which changes according to the applied force (figure 3.3). This analog signal can then be read in by the available on-board computer unit (section 3.4). The sensitivity of the measurement can be adapted by adjusting the resistor RM in the voltage divider as depicted in figure 3.3. In this project the FSRs will serve solely as boolean operators (contact or no contact), thus a relatively high resistor is chosen.
that a leg is only allowed to move if three legs, i.e. two exterior legs on one side and the middle leg on the other side, have an equal pressure distribution first. However it is highly questionable that accurate and reliable force readings can be obtained given the far from robust way these sensors have been mounted onto the leg tips. Perhaps another sensory method and especially leg design would be more appropriate for such applications e.g. the 3D force sensors and spring-dampers which have been mechanically integrated into the legs of LAURON V (figure 2.4) [34].

3.3.2 Inertial measurement unit

To grant Lil’ Hex with a certain degree of autonomy, it should at least have a sense of how its body is orientated in the environment. In this regard an IMU is attached to the frame of the hexapod robot. This device can communicate its data to the on-board processing unit (section 3.4) over I²C.

An inertial measurement unit is basically a combination of accelerometers (yielding acceleration along a given axis), gyroscopes (giving the angular speed about a certain axis) and optionally digital compasses (furnishing orientation data). In most applications (like UAVs, gaming, smartphones, limbed robots etc.) all of this data is fused together to get highly accurate estimates on the body’s orientation. As such an IMU is the embodiment of the concept of data fusion discussed earlier in section 2.2.2. The IMU attached onto Lil’ Hex is the MPU-6050 of Invensense which combines a three-axis accelerometer and a three-axis gyroscope together with an on-board digital motion processor capable of processing complex nine-axis fusion algorithms [38]. They state nine-axis instead of just six because there’s the ability to connect an extra triple axis digital compass to the MPU-6050 by means of I²C-communication.

Figure 3.4: SparkFun triple axis accelerometer and gyro breakout - MPU-6050 [38].
3.3 Sensors

An accelerometer on its own yields the current rate of acceleration along a certain axis (x, y and/or z) by using the principle of spring-mounted masses whose displacement can be measured during acceleration. This simply follows from Newton’s second law and the spring-mass relation. As depicted in figure 3.5a, the mass not only experiences linear accelerations due to possible movement but also gravitational acceleration. In order to obtain pure linear acceleration without the influence of gravity (in dynamic applications), one should know the local gravity vector both in direction (e.g. using an inclinometer) and magnitude such that it can be factored out of the accelerometer readings. On the other hand if one is interested in orientation data (like pitch and roll, not yaw as is explained in section 6.1) in fairly static applications, this can be found just by looking at the decomposition of the gravity vector along the three axes (x, y, z) using basic trigonometry (cfr. section 6.1). What’s more based on the above one can conclude that an accelerometer in free fall will actually read out zero acceleration as there is no relative movement between the mass and springs 3.5b.

![Figure 3.5: Influence of gravity in a two-axis accelerometer.](image)

A gyroscope on the other hand gives the angular velocity about a particular axis (x, y and/or z). Most commonly this information is then integrated over time to find out the changes in rotational attributes of a body like pitch, roll and yaw. The gyroscope basically makes use of the Coriolis effect which creates a fictitious force on a mass that is moving in a rotating reference frame. In MEMS technology movement of the mass is caused by a high frequent oscillator, and the Coriolis effect is picked up by capacitive sensing cones.

As becomes clear from the discussion above both accelerometers and gyroscopes can, after some processing, grant information on the orientation of the robot’s body. As it happens...
both sensors have got their own distinct drawbacks and advantages making them perfectly suited for applying the concept of sensor fusion. On the one hand accelerometers are very noisy (e.g. most of the time a low-pass filter is applied causing a delayed response) but remain stable over time. Gyroscopes on the other hand are very smooth (as a result of the single integration of the angular velocity over time) with a fast response, but unfortunately the integration also instigates an inevitable drift over the long run. Moreover the gyro doesn’t measure the gravity vector while it is desired to perceive the tilt w.r.t. this vector as is luckily done by the accelerometers. Consequently the accelero and gyro data are indeed often combined to get enhanced results. The accelerometer corrects the gyro for drift and maintains the orientation w.r.t. gravity and the gyro ensures a fast, more smoothed response.

3.3.3 Infra-red position sensitive device

After implementing the ability to detect ledges and to know the orientation of the hexapod’s body, another wanted property of the robot should be to detect and avoid obstacles. One way of measuring the distance to an object is by making use of relatively cheap IR PSDs. It should be mentioned however that in real-life mobile robot applications mostly other more robust distance measurement sensors are used such as ultrasonic sensors, laser rangefinders (cfr. section 2.2.2) and/or stereo camera.

The IR PSDs which are implemented here are three Sharp GP2Y0A21YK0Fs (figure 3.6a). These small and rather inexpensive devices are often used in touchless switches, robot cleaners, as sensor for energy saving and amusement equipment. The distance measuring range is about 10 to 80 cm and the data has to be read-in using an analog input port [21].

As can be seen in figure 3.6a the Sharp sensor has two lenses. One of these lenses is used to send a signal and another one to capture that same signal. In the case of an infra-red sensor, this is a pulsed infra-red LED which emits a light that is invisible to the human eye. The angle under which the reflected light is captured again depends on the distance from the detected object as shown in figure 3.6b. Utilising this relationship then leads to the eventual distance measurement. This data is translated internally to an analog voltage signal such that it can easily be read-in by the processing unit [5]. When making use of these inexpensive sensors one should bear in mind that they generally provide far from consistent or robust measurements since they experience considerate influence from different possibly changing environmental aspects (e.g. sunlight, reflective properties of different surfaces). As such it is advised not to use the IR PSDs for precise distance
measurements, but merely to get an indication on whether or not there is an obstacle ahead.

(a) Sharp GP2Y0A21YK0F IR PSD.   (b) Working principle [42].

Figure 3.6: An IR PSD and its working principle.

Besides the advantage of the easy implementation, the read-out voltage has a rather inconvenient non-linear relation with the real distance to the object measured. These non-linear relationships will have to be measured for each of the three available IR PSDs (section 5.2). As depicted earlier, obstacles closer than 10 cm or further than 80 cm will lead to incorrect readings. Not seeing the far objects isn’t really a concern for the moment, but being incapable of detecting closer objects could be fatal. Clever mechanical mounting of the IR PSDs could be a way to resolve this issue.

3.3.4 Stereo camera

As said in the previous section, walking through an environment could be potentially dangerous without the ability to detect and avoid obstacles. The IR PSDs are suitable to measure the distance to an object, but aren’t capable of differentiating a wall from an obstacle (that could possibly be overcome). That is why another range measurement device having the potential of being a very rich source of information is added to the sensory system of Lil’ Hex: the 3D camera (also known as binocular or stereo camera). Besides obtaining depth a lot of other data can be deduced from a camera as well (section 2.2.2). For instance in a later stage one may consider using the camera to infer odometric information (i.e. visual odometry as outlined in section 2.2.2 [30] or perform such things as object recognition.
In this masters’ project the robot will utilise the Minoru 3D webcam so that depth information can be retrieved from the environment. The Minoru is a low-cost commercial 3D webcam which has two build-in cameras. Communication with the hexapod’s processing unit (section 3.4) and power supply is done over a single USB cable [39]. There are better single cameras available, but this one is good enough to experiment with binocular vision.

Stereo vision actually refers to the Greek word “stereopsis”, where “stereo” stands for solid and “opsis” for appearance or sight. The depth perception created with stereo vision simply exploits the triangulation principle. Basically, two frames are captured from the environment with two cameras which are horizontally shifted from each other. These two images are scanned for equal points and based on these positions depth information of the object can be retrieved (figure 3.7b). The resulting data obtained with stereo vision is explicitly explained in section 5.2.2.
3.4 Processing unit

Now that the hexapod is able to retrieve information from the environment and it has the means to move around, it also needs to think and make proper decisions. One could say that after implementing senses and muscles it still needs the brains, a device which is capable of transforming the retrieved information (inputs) into the proper action (outputs). Preferably, this on-board computer should be light, low in energy consumption and still remain powerful enough to do all of the necessary computations.

Since the department happened to have a BeagleBone Black (figure 3.8a) at its disposal this was picked to be the computational unit of the hexapod robot [43]. This low-power, single-board computer produced by Texas Instruments is capable of setting up all of the necessary communications with the earlier chosen devices and acting as the brain of the robot. One has the possibility to install the operating system (e.g. Angstrom, Debian, Ubuntu, Android...) of his or her choosing. Anyhow in this master project the BBB will run under the Linux distribution called Debian. To facilitate the debugging process it is chosen to write the programs remotely in an IDE called Eclipse, using mainly Python as a programming language. For each experiment the necessary program can then easily be loaded into the memory of the embedded computer.

Alas the BeagleBone Black isn’t capable of delivering twelve PWM signals at a time. This implies that the servo-actuated leg joints cannot be controlled individually which would decrease the degree of flexibility. Luckily the Mini Maestro 12-channel USB servo controller of Pololu provides the ideal solution. The Mini Maestro can send out the necessary amount of PWM signals and can be controlled by the BBB via TTL serial communication [65].

![Figure 3.8: The hexapod’s processing unit (left) and its servo controller (right).](image)
3.5 Power supply

The last item which has not yet been described, is the quite important power supply of the robot. A lot of the initial problems during preliminary experiments were entirely due to insufficient power resources. Consequently the importance of adequate power supply and its influence on the robot’s overall performance cannot be overstressed. If one takes a quick peek at the list of components by now; twelve servos, FSRs, the IMU, IR PSDs, two cameras, the BeagleBone Black and the Mini Maestro, then it can be concluded that the power consumption will add up if everything is fully operational.

Knowing that the twelve servos require the most power and that they should be able to demand variable peaks of current, i.e. depending on the load the robot should overcome, it is advised to power the actuators from a separate battery pack. Hereto five rechargeable 1.2 V NiMH 2700 mAh batteries are acquired which deliver the necessary 6 V for both types of servos [19, 35]. The rest of the equipment is fed by a Li-ion rechargeable battery with two cells (figure 3.9a), thus providing a voltage of 7.4 V and a capacity of 2.25 Ah [45]. In fact, the BeagleBone Black should be fed with 5 V, so an additional DC-DC converter is of paramount importance (figure 3.9b) [46].

Of course the available battery supply greatly influences the amount of time the hexapod robot can actually work autonomously. Operating with everything at its maximum isn’t always necessary and could lead to a fast power shortage. In order to have a handle on this power consumption, one could for instance adapt the actuation method (e.g. the walking gait or speed) or power off some less needed sensory equipment based on the measured battery level (SoC measurements). This topic of energy efficiency hasn’t been investigated in this master dissertation but could form interesting further research.

(a) Li-ion rechargeable battery. (b) DC/DC-converter.

Figure 3.9: Power supply: Li-ion battery and DC/DC-converter [45, 46].
Once all of the aforementioned equipment has been purchased and their proper working has been explored, the old hexapod robot is first stripped of all his former components, leaving only the naked frame (figure 3.11a). Next, with a proper weight balancing in mind, the newly bought components are then mounted and wired meticulously ending up with the robot’s final design (figure 3.11b). Through this process the legged vehicle has gathered a sense of personality, and as such it deserves a name of its own. As mentioned before it was given the name Lil’ Hex.

For the sake of completeness figure 3.10 shows a full schematic of the external devices attached onto the hexapod i.e. the actuators, the sensors, the processing unit and the power supply. Also the interfacing between the different devices and the necessary voltage levels are depicted.

**Figure 3.10:** Full schematic of the various hexapod robot’s components and their interfacing.
3.6 Lil’ Hex

(a) The stripped, naked hexapod frame.

(b) The final design of the hexapod robot having earned the name Lil’ Hex.

Figure 3.11: Pictures showing the transition from initial to final status.
Chapter 4

Walking gaits

Now that Lil’ Hex is fully equipped with senses, brains and muscles it is time to let it take its first steps. Six-legged locomotion is particularly interesting because of the multitude of possible walking gaits in which static stability is maintained (section 2.2.1). The robot can move in the environment while lifting one, two or three legs at a time and still remain stable. However, these walking gaits all have their own advantages and disadvantages (and still remain statically stable).

In this chapter first the forward and inverse kinematics of the hexapod are derived yielding insight into the relationship between the servo-actuated joint angles and the leg displacement. Then using this knowledge three different gaits are implemented namely the tripod, ripple and wave gait. It appears letting the robot walk straight is less straightforward than expected and requires some sensory feedback from the IMU (yaw data). Once all gaits have been explored a comparison is made. Finally Lil’ Hex is programmed to walk backward and turn on the spot over a desired angle (using IMU data).
4.1 Kinematic model

Towards programming different leg displacements it will come in handy to have the kinematics available of Lil’ Hex. Therefore, before discussing the implemented walking gaits, the obtained kinematic model of the six-legged robot is described in this section.

Figure 4.1 shows the rod mechanism of the legs onto a rough sketch of the rear side of the hexapod robot. This simplified leg model (of the left rear leg) is also depicted in figure 4.2 where it will serve as a guidance for the derivation of the forward and inverse kinematics. The origin of coordinate frame (v,w) is placed at the shaft of the knee servo and the origin of coordinate frame (x,y,z) is located on top of the frame at the hip joint.

Firstly, the forward kinematics starting from the known servo angles, i.e. the knee servo ($\theta$) and the hip servo ($\phi$), will be calculated in order to find out the end effector position (H). Let’s take the left rear leg as an example for the convenience of explaining. To start, the end effector position will be calculated w.r.t. the (v,w)-coordinate frame. Starting from the servo motor a basic rotational movement determines the coordinates of point B (step 1). Hereafter follows the hardest part in solving these geometric equations namely to obtain the $(v_C, w_C)$-coordinates together with the angles $\psi$ and $\alpha$ out of the nonlinear set of equations in step 2. Once this is found, step 3, 4 and 5 are rather straightforward geometric equations. In step 6, the end effector position is determined w.r.t. the knee reference frame and in order to know the 3D position of the foot, the hip swing effect needs to be taken into account as well. Therefore in step 7, the $(v_H, w_H)$-coordinates are described w.r.t. the
4.1 Kinematic model

(x,y,z)-coordinate frame. This merely implies a translation of the fixed distances (30 mm in x-direction and 40 mm in y-direction) between the two coordinate frames and a rotation depending on the hip angle \( \phi \) as in figure 4.2b).

\[
\begin{align*}
\mathbf{v}_B &= |AB| \cos(\theta) \\
\mathbf{w}_B &= |AB| \sin(\theta)
\end{align*}
\]

\[
\begin{align*}
\mathbf{v}_C &= |AB| \cos(\theta) - |BC| \cos(\alpha) \\
\mathbf{w}_C &= |AB| \sin(\theta) - |BC| \sin(\alpha) \\
\mathbf{v}_C &= \mathbf{v}_D + |DC| \cos(\psi) \\
\mathbf{w}_C &= \mathbf{w}_D + |DC| \sin(\psi)
\end{align*}
\]

\[
\beta = \psi - \tan\left(\frac{|IC|}{|ID|}\right)
\]

\[
\begin{align*}
\mathbf{v}_F &= \mathbf{v}_D + |FD| \cos(\beta) \\
\mathbf{w}_F &= \mathbf{w}_D + |FD| \sin(\beta)
\end{align*}
\]

\[
\begin{align*}
\mathbf{v}_G &= \mathbf{v}_E + |EG| \cos(\beta) \\
\mathbf{w}_G &= \mathbf{w}_E + |EG| \sin(\beta)
\end{align*}
\]

\[
\begin{align*}
\mathbf{v}_H &= \mathbf{v}_G - |GH| \sin(5^\circ) \\
\mathbf{w}_H &= \mathbf{w}_G - |GH| \cos(5^\circ)
\end{align*}
\]

\[
\begin{align*}
x_H &= (v_H - 30 \text{ mm}) \cos(\phi) \\
y_H &= w_H - 40 \text{ mm} \\
z_H &= x_H \tan(-\phi)
\end{align*}
\]

After obtaining the forward kinematics, the inverse kinematics are determined on the next page such that a wanted end effector position can be translated into actual actuator angles (knee angle \( \theta \) and hip angle \( \phi \)). Regard that there are only two DOFs which means that

Figure 4.2: Forward kinematics.
only two coordinates can be freely chosen. Consequently the (y,z)-coordinates will be given as an input in the inverse kinematics and will be transformed into the proper servo angles. The forward and inverse kinematics of the right legs can be obtained by similar reasoning.

\[
\begin{align*}
\text{1} \quad w_H &= y_H + 40 \text{ mm} \\
\text{2} \quad w_G &= w_H + |GH| \cos(5^\circ) \\
\beta &= \arcsin \left( \frac{w_G - w_E}{|EG|} \right) \\
\text{3} \quad v_H &= y_E + |EG| \cos(\beta) - |GH| \sin(5^\circ) \\
\psi &= \beta + \arctan\left( \frac{|IC|}{|ID|} \right) \\
\text{4} \quad v_C &= v_D + |DC| \cos(\psi) \\
w_C &= w_D + |DC| \sin(\psi) \\
\text{5} \quad |AC| &= \sqrt{v_C^2 + w_C^2} \\
\delta &= \arctan \left( \frac{w_C}{v_C} \right) \\
\gamma &= \arccos \left( \frac{|BC|^2 - |AB|^2 - |AC|^2}{-2|AB||AC|} \right) \\
\text{6} \quad L &= v_H - 30 \text{ mm} \\
\phi &= -\arcsin \left( \frac{z_H}{L} \right) \\
\text{7} \quad \theta &= 180^\circ + \gamma + \delta \\
x_H &= L \cos(\phi)
\end{align*}
\]

(a) Rod mechanism of a left leg.

(b) Top view: hip movements.

**Figure 4.3:** Inverse kinematics.
4.2 Terminology and actuation

Some clarification of the common terminology might be convenient before explaining the implemented walking gaits. As represented in figure 4.4, LF, LM and LR are respectively the left front, middle and rear leg. This also applies for the right legs being RF, RM and RR. Other terms often used are the swing and stance phase. The swing phase indicates when the leg is in the air, and logically, the stance phase points to the moment when the leg is on the ground, e.g. as shown in figure 4.6a of section 4.3 the thick lines indicate the swing phase, whereas the dashed line represents the stance phase.

Knowing the way the servos should be controlled from section 3.2, it would be preferable to set the servo position with comprehensible angles instead of PWM signals. That is why, in advance of the actual programming, the relationship between the period of the pulse and the angle of the servo is measured as in figure 4.5. These obtained characteristics will contribute to a more genuine and definitely more intelligible programming of the walking gaits.

Figure 4.4: Leg indication.

Figure 4.5: Relationship between pulse width and servo angle from left front knee.
4.3 Tripod Gait

The most popular walking gait applied with six-legged robots is the tripod gait. This walking gait uses three legs at a time and is considered as the fastest statically stable walking gait.

The gait can be divided into four main steps. A first step is lifting (thus moving the knee servos) the exterior legs at the right side, being RF and RR, together with the middle leg at the left side (LM). Once these legs are going into their swing phase, the robot is supported by RM, LF and LR. After the lifting, the hip servos move RF, RR and LM forward, while the legs on the ground push the robot’s body forward by moving the servos in the opposite direction of those in the air. When this swing phase is done, the legs are set back onto the ground and the second step can begin, thus swinging the other three legs in the air now.

(a) Tripod gait sequence [1].

(b) Programming of the tripod gait.

Figure 4.6: Tripod gait.
4.4 Ripple Gait

A second well-known gait with six-legged locomotion is the ripple gait. This one is considered as an intermediate gait between the wave and the tripod, where two legs are in the air at the same time.

The ripple gait works with two independent gaits at each side of the hexapod. The stance phase usually takes twice as long as the swing phase and the opposite legs are 180 degrees out of phase. In a first step RR and (slightly later) LF are brought into their swing phase while the other legs are dragging the hexapod forward in their stance phase. After this, first RR and then LF are set onto the ground where after RM and LR are lifted. Once RM is brought back down RF goes into the air and an instance later while LR is set onto the ground LM goes up as well. The same sequences are repeated as depicted in figure 4.7.

![Ripple gait sequence](image)

**Figure 4.7:** Ripple gait sequence [1].

4.5 Wave Gait

The final gait that has been implemented is the wave gait. This is the most stable gait, but also the slowest way to move in the environment. There is only one leg in its swing phase at a time, while the others remain on the ground.

This gait is characterised by the “wave” that is going through the legs of the hexapod, hence the name. The wave starts with lifting one leg of the ground, swinging it to the front and placing it back down. During this swing the other legs drag the body forward and as such place the next leg ready for the swing. A sequence for example is RR-RM-RF-LR-LM-LF as illustrated in [1.8]
4.6 Run straight

Prior to comparing the previously described walking gaits one should try to obtain a straight marching such that the obtained measurements are relevant. While implementing the walking gaits it was noticed that the direction of walking is highly influenced by the ground surface. In order to walk as straight as possible independent of the surface walked upon, the hexapod will use the IMU\textsuperscript{1} to keep its body into the desired orientation.

The flowchart of the implemented program for the tripod gait is depicted in \ref{fig:flowchart}. As in the previous section, it starts by initialising the servo speeds and angles together with a boolean which indicates when a drift has occurred. Hereafter the hexapod is set in the horizontal position and the calibration procedure of the IMU can begin. Once this is finished, the actual program kicks off by reading in the current yaw value of the robot’s body. Hereby a drift is detected when the yaw goes outside the predefined boundaries (in this case $1^\circ$). When a drift is detected, the “Drift” boolean is set to “True” and the sign of the measured yaw determines how the gait should be adjusted. If the angle is smaller than zero it implies that Lil’ Hex’s body is oriented to the right. Subsequently the body is set back to the desired direction by increasing the right hip angles while the left ones remain the same. The opposite will be done when a positive yaw value is measured. Once the body is oriented back again into the correct direction and the “Drift” boolean was previously set to “True”, it will do a couple more steps of the tripod gait with the previous angle settings. The number of extra steps could be chosen freely via the adaptable “Counter” value.

\begin{figure}[h]
\centering
\includegraphics[width=\textwidth]{wave_gait_sequence}
\caption{Wave gait sequence \[1\].}
\end{figure}

\footnote{1The data acquisition done using the IMU is extensively covered in section \ref{sec:imu_data}.}
4.7 Comparison

Now that all of these gaits are fully implemented (i.e. programmed and running quite straight) it is useful to study the main differences such that the most suitable gait could be picked depending on the situation at hand (e.g. ground surface, slope, battery level...). Not only are the gaits compared w.r.t one another but also the influence of the velocity of the hip servos and the size of the swing phase are examined.
The measurements have been done on a plain table where the hexapod needed to cross a distance of one meter while the time was being measured. The results are plotted in figure 4.10. It is no real surprise that the tripod gait came out as a winner with a top speed of 0.10 m/s which is about 100 times slower than the fastest man alive running the 100 m, being Usain Bolt. Regarding the influence of the swing size and the velocity of the hip, one clearly sees a high impact in the beginning but going towards the highest swing angles and motor speeds these improvements become rather insignificant.

To clarify, it has never been our intention to create world’s fastest hexapod robot. These measurements solely serve as a guidance to program, e.g. when a fast gait is demanded, a tripod gait with a swing size of at least 15 degrees and a hip servo velocity above 100 degrees per second should be implemented, else it would be unjustified to walk with this more unstable gait. Integrating this knowledge into Lil’ Hex could also lead to an algorithm where it is able to determine the proper gait at the correct time and situation (based on some sensory input).

Figure 4.10: Influence of different gaits, swing size and hip velocity on the velocity of the hexapod.
4.8 Turning and reverse

After implementing all of these walking gaits, it is relatively straightforward to create a turning or a reverse gait. Every gait can be adjusted in such a way that another direction is obtained. For instance a difference in hip swing size on both sides leads to a turning gait. Even further, moving the left and right legs in the completely opposite direction with respect to one another creates a turning on the spot and naturally, programming both gaits as the forward gait but switching the directions of the hip movement results into the backward gaits. The program structures for turning and running backwards can be found in the next chapter (section 5.1) as they were used in conjunction with the robot’s ability to sense ledges.

Another useful building block that has been implemented is letting the robot turn over a precisely specified angle by making use of the IMU’s yaw data. For instance while traversing an environment it could be desirable to let the hexapod turn 90 degrees or another angle of its choosing when an obstacle is detected. Turning over a specified angle (here 90°) to the right is done by using the program depicted in figure 4.11. Since the measured yaw values lie within ]-180°,180°] (cfr. see section 6.1 on the use of the arctan2 function) a shift of 360° is performed when the yaw value is smaller than -90° to avoid complications in comparing values when turning right across the transition from -180° to 180°. When Lil’ Hex has almost reached the desired angle (here 12° away from it), a transition is made from the fast, hefty tripod to the gentile wave gait such that the desired angle can be reached precisely.
4.8 Turning and reverse

\[
\psi_d = 90^\circ
\]

Calibrate IMU

\[
\psi_i = \text{initial yaw}
\]

\[
\psi_i < -90^\circ
\]

Shift = 360°

\[
\psi_i + \text{Shift}
\]

\[
|\psi_m - \psi_i| > \psi_d - 12^\circ
\]

Turning tripod gait

\[
\psi_m = \text{current yaw}
\]

***Turning wave gait***

\[
\psi_m < -90^\circ
\]

\[
\psi_m + \text{Shift}
\]

\[
|\psi_m - \psi_i| > \psi_d
\]

End

Figure 4.11: Turning right over 90° using the IMU.
Chapter 5

Ledge and obstacle detection

To let the multi-legged robot cope on its own while roaming around space two fundamental capabilities have been added to its repertoire: ledge and obstacle detection. As one may recall from section 3.3 the former is achieved by mounting FSRs on the leg tips of the robot and using these as boolean operators, whereas the latter is done by means of infra-red position sensitive devices at the back and front and a binocular camera.

In what follows first an experiment is discussed where Lil' Hex walks around on the table without falling off. This is done to get a first experience on using the tactile sensors for ledge detection. Secondly a closer look is taken at obstacle detection by means of the IR PSDs and the stereo camera.
5.1 Ledge detection

A first undesired event that could lead to a damaged robot is that of walking over a ledge. To avoid this, the FSRs, mounted onto the flattened surface of the feet, have been put to use. As illustrated in figure 3.3 in section 3.3.1 one can also influence the sensitivity of the sensor by changing the applied resistor in the voltage divider. In this case a resistor of 10 kΩ satisfied the desired sensitivity as boolean operator. Mounting the devices onto the legs and placing an extra safety cap creates a kind of pretension which affects the sensor readings even without an external force applied onto it. Therefore every FSR characteristic should be measured only when fully attached onto the hexapod’s legs. However a complete calibration is not necessary, since the FSRs solely serve as a boolean operator. The only thing that needs to be specified is an appropriate threshold value.

In figure 5.1 the program structure is shown for an experiment in which Lil’ Hex is capable of walking around on the table using the tripod gait without falling off. The reaction to a ledge detection is, in this case, simply changing the direction of walking, i.e. going from a forward gait to a backward gait followed by a turn (e.g. here 90° by making use of the building block described in section 4.8). The measurements happen when the hexapod switches the three legs that are in the air with those onto the ground. First, the legs that end their swing phase will be put onto the ground, where after the tactile sensor at the front (i.e. with respect to the direction of walking) will check whether it is making ground contact or not. If a measurement above the threshold value is detected (thus a force is applied) the gait can continue. If not, the hexapod should react and change its direction, in this case, go to the backward gait. From the current interrupted position it is relatively easy to begin the backward gait. The legs that should have made ground contact, but didn’t, are lifted back up into the air and simply do the opposite hip swings compared to the forward gait. After a predefined number of steps, Lil’ Hex starts turning over the earlier mentioned 90°. Once it is finished turning it goes back forward with a normal tripod gait. As such Lil’ Hex stays on the table at all times.

Although the implemented program is rather basic, it gives a good first experience on the use of the tactile sensors for detecting ledges. Problems encountered during the developing of this program were especially sensor-related. For instance a bug in the BeagleBone Black always gives the previous measurement instead of the current one when reading the analog input ports and at times measurements are corrupted causing a read-in failure. To deal with this, when measuring, multiple read-ins are done in quick succession and only the last successful reading is kept to act upon.
5.1 Ledge detection

Start

Forward gait
RF-RR-LM up
RF-RR-LM swing to front
RM-LF-LR move to front
RF-RR-LM down
RF on the ground?
False
Steps = 0
True
RM-LF-LR up
RF-RR-LM move to back
RM-LF-LR swing to front
RM-LF-LR down
RF-RR-LM move to back
RM-LF-LR swing to front
RM-LF-LR down
LF on the ground?
False
Steps = 0
True
Turn 90°
Steps = 3?
False
Steps += 1
False
False
False
False

Backward gait
RF-RR-LM up
RF-RR-LM swing to back
RM-LF-LR move to front
RF-RR-LM down
RR on the ground?
False
Steps = 0
True
RM-LF-LR up
RF-RR-LM move to front
RM-LF-LR swing to back
RM-LF-LR down
RF-RR-LM move to front
RM-LF-LR swing to back
RM-LF-LR down
LR on the ground?
False
Steps = 3?
False
True

Figure 5.1: Flowchart: interchanging between forward backward and turning.
5.2 Obstacle detection

Another needed property in order to create an autonomous hexapod robot which is capable of walking around on its own, is the ability to detect and react to obstacles on its way. This obstacle detection and avoidance has been implemented in a first stage solely using the distance sensors. Contrary to the stereo camera, the IR PSDs have not been relied upon for precise distance measurement, they merely serve to detect whether or not an obstacle is nearby.

5.2.1 IR PSDs

First off, the IR PSDs as described in section 3.3.3 are implemented onto the robot. Before one can go over to implementing any sensor-actuator interaction, one should obtain the relationship between the distance measured and the read-out voltage. In order to retrieve a very accurate characteristic swiftly, a small program has been written in Python. At every distance going from 100 cm to 2.5 cm in steps of 2.5 cm 500 voltage measurements are read-in and each time only their mean value is saved to be the most likely voltage value related to that distance. This led to the graph depicted in figure 5.2.

![Figure 5.2: Measured characteristics of IR PSDs.](image)
Now that the read-in voltages can be linked with the appropriate distances it is possible to inform these to Lil’ Hex. A program has been developed which uses the PSDs much like a boolean to see whether or not an obstacle is standing in the way. Once an object is detected in a range of 25 cm the hexapod needs to change its direction. In this program the afore explained running straight and turning a desired angle applications (section 4.6 and 4.8) are used as well to let Lil’ Hex run a square course with four obstacles in every corner (figure 5.3).

The flowchart of the developed application is illustrated in figure 5.4. Just like every program that makes use of the IMU (here to run straight and to turn a desired angle), the robot starts with the calibration procedure. Thereafter the number of turns are set to zero (with variable “Turns”) and the program begins with a wanted yaw value ($\psi_w$) of zero degrees in the running straight algorithm (as the read-in yaw of figure 4.9). After every leg has done its swing phase during a wave gait (thus six steps) the left an right PSDs are checked. This is a rather slow process because a lot of measurements are taken into account in order to get a reliable distance measurement. If one of the two PSDs detects an obstacle within the range of 25 cm, the program jumps to the turning a correct angle application (as in 4.8) with a desired angle of 90 degrees. Else it would continue its wave gait. Once the turning is done, the number of turns are incremented and the program continues in the second case where the $\psi_w$ now becomes -90 degrees so that it detects drifting w.r.t. this yaw value from this point on. This process continues, letting the robot walk in a square.

As a further experiment it came to mind to use the IR PSDs to detect obstacles and as such trigger the stereo camera to obtain more precise insight on the robot’s surroundings. This way it could be avoided that the time and energy consuming camera processing is continuously running. Consequently the hexapod could then make decisions based on its stereo vision instead of its PSDs. Nonetheless this has not been implemented thus far.
5.2 Obstacle detection

Figure 5.3: Obstacle detection and avoidance: test setup.

Figure 5.4: Obstacle detection and avoidance: flowchart.
5.2.2 Stereo vision

As already mentioned before, the camera has mainly been purchased in light of further research involving visual odometry and mapping and for being a rich source of potential information in general. Nonetheless experiments have been done using the stereo camera as well. However letting the camera act upon the measured obstacles using stereo vision has not been implemented thus far. In what follows the steps taken to get to a working obstacle detection program using stereo vision are described. Hereby Brahmbhatt’s book on OpenCV proved particularly useful [6].

Calibration & stereo rectification

First of all, like every sensor both cameras should be individually calibrated. If this isn’t done the robot would only know about objects in their image coordinates which is not useful to interact with the real world. After the single camera calibration one should also do a stereo calibration in order to find the baseline (horizontal distance) between the two cameras and the rotation matrix which will be used to align the two imaging planes. This latter mentioned aligning procedure is necessary because due to manufacturing errors misalignment is often the case for two cameras in a stereo set-up. If an exact alignment is obtained it is easier to find the matching points in both images cause the search is restricted to the same row of captured frames. This process, of aligning both images is often referred to as stereo rectification. An example of this process is shown in figure 5.5 where the red lines serve as an indication of the obtained alignment.

![Figure 5.5](image_url)

(a) Capture left and right images.  
(b) Rectified left and right images.

**Figure 5.5:** Remapping the images.
5.2 Obstacle detection

Disparity calculation

Eventually the disparity algorithm which makes use of the above mentioned row searching method (to find equal points) can be applied in order to retrieve the depth map of the captured environment. The OpenCV disparity functionalities produce a grayscale image from the captured frames where an increase in darkness implies a greater depth. This disparity map can also be translated to the real depth information, i.e. the depth of the pixels in metres. Figure 5.6 shows the obtained 3D maps from the captured frames in figure 5.5. The black and white image in 5.6a illustrates the 2D depth map and the point cloud depicts the points of these images in 3D (figure 5.6b).

![Disparity map](image1)
![Pointcloud](image2)

(a) Disparity map. (b) Pointcloud.

Figure 5.6: 3D maps.

Obstacle detection

Now that the depth of the environment in front of the hexapod is obtained, it should also be able to differentiate obstacles from each other so that it can plan on how to avoid it. Obstacle detection is often done based on the color of an object. An image is in fact a color map where mostly every pixel has three values and the combination then indicates a color. Different color maps can be applied, e.g. the most common ones are the RGB and HSV color maps. The algorithm then searches the color to be found, based on predefined threshold values and the applied color map. Basically, that is how the hexapod will also be able to distinguish several objects from each other. As mentioned before the disparity map generates a grayscale image or in numbers an image with pixels between 0 (black)

---

1 Disparity refers to the difference in location of an object in corresponding left and right images as seen by the left and right eye which is created due to parallax. The brain uses this disparity to calculate depth information from the two dimensional images.
and 255 (white). Using this image the largest and closest areas are determined by the help of some OpenCV functionalities and the so called threshold image is obtained (black and white). This image is shown in figure 5.7a where the two objects are clearly distinguished from the rest of the environment. The contour of the object is found, the area is calculated and finally the central point is determined \((x,y,z)\), where the depth information \((z)\) was already retrieved earlier (see 5.6). The results of the initially captured frames (figure 5.5) are represented in image 5.7b where the recognised objects are indicated by a blue contour and the depth position is plotted into the calculated central point.

![Threshold map.](image1)

(a) Threshold map. ![Detected obstacles.](image2)

(b) Detected obstacles.

**Figure 5.7**: Detecting the obstacles.

To conclude, this first experience with a stereo vision camera already gives an idea of its potential. In the quest of developing an autonomous hexapod robot which is able to do environmental exploration this device could be used for more than just obstacle detection. As mentioned in section 3.3.4, it is most likely one of the best ways to get odometry information with a multi-legged robotic system. Furthermore, while obtaining its own location it is also possible to capture a map of the environment, also known as SLAM. Stereo SLAM has already been applied in several real-life applications and some open-source libraries are available as referenced in 2.3.1. Although in the future, it remains to be seen what level of accuracy and performance can be achieved with the purchased low-cost Minoru 3D webcam.
Chapter 6

Body levelling

Contrary to their wheeled counterpart, multi-legged robots have the desirable ability to adjust their body configuration while moving (e.g. reposition their centre of gravity, change body orientation...). Just as the legendary mountain goat-like animal “dahu” which has legs of different lengths to fit the mountain side Lil’ Hex should also be able to adjust its legs in order to mount slopes more instinctively. Although it is a fictional fairytale it can be easily shown profitable from a mechanical point of view since levelling-out the hexapod horizontally on the slope avoids the occurrence of an extra moment originating from the gravity force onto the body of the bug-like creature.

This chapter will describe all of the steps that have been taken in order to eventually let Lil’ Hex mount plain slopes. It will start by creating a deeper insight into the sensor fusion that can be done by the IMU which was already touched upon in section 3.3.2. Hereafter a levelling-out procedure for remaining in a horizontal body position on a slope with adjustable inclination will be implemented by making use of the kinematic model described in section 4.1. Finally this quite extensive procedure is combined with the wave gait and the running straight algorithm (section 4.5 and 4.6 respectively) to let the limbed robot mount slopes with inclinations of up to 10°.
6.1 IMU sensor fusion

As discussed earlier in section 2.2.2 the IMU could in theory serve as a position estimator, though due to the necessity of integrating the accelerations twice it is strongly discouraged. Nonetheless it still provides useful orientation data which have been applied in several applications within this master dissertation (sections 4.6, 4.8, 6.2 and 6.3). Before actually implementing the different IMU applications, it is desirable to get undisturbed data out of the MPU-6050. As one may recall from section 3.3.2 the accelerometer and the gyroscope are actually quite complementary, hence they are often used together in sensor fusion. The question then arises how to do so?

Figure 6.1 represents the orientation of the IMU onto the hexapod robot. This implies that the shown coordinate frame will also apply for Lil’Hex. The MPU-6050 is attached onto the midpoint of the body frame. By placing the IMU as such the roll or longitudinal axis will be equal to the y-axis, the pitch or transverse axis will correspond to the x-axis and the yaw or normal axis will be equivalent to the z-axis.

Figure 6.1: Placement of MPU-6050 on Lil’ Hex.
Section 3.3.2 already explained that the gyroscope delivers angular velocity and thus needs to be integrated once to deliver the orientations, which causes the aforementioned long term drift. Practically to compute this orientation, the IMU should be initialised at a known sensor position (the often referred to calibration procedure in the various IMU applications of the current and previous chapters) and from there on it can update its orientation using measured time intervals (e.g. \( \omega \cdot (t_{n+1} - t_n) + \text{pitch}_{t_n} = \text{pitch}_{t_{n+1}} \)).

Also mentioned in section 3.3.2 is the possibility to use the accelero data to calculate the Euler angles, pitch \( \phi \) and roll \( \rho \) in rather static applications by looking at the decomposition of the gravity vector along the three axes (x,y,z) and applying the trigonometric relations of (6.1) The yaw on the other hand, can’t be computed using the accelerometer data due to the fact that turning around the yaw axis won’t cause any change in the accelerations, e.g. a strict rotation around the yaw axis will cause accelerations \( a_x \) and \( a_y \) to remain zero, while \( a_z \) stays equal to the gravitational acceleration \( g \).

\[
\phi = \arctan \left( \frac{a_y}{\sqrt{a_z^2 + a_x^2}} \right) \quad \text{and} \quad \rho = \arctan \left( \frac{a_x}{\sqrt{a_y^2 + a_z^2}} \right) \quad (6.1)
\]

This finally leads to the sensor fusion where the accelerometer (noisy but good on the long term) will correct the acquired orientations of the gyroscope (fast, quite undisturbed response but drift on the long term). This fusion is often done by a Kalman or a complementary filter\(^1\). However, as an attentive reader would already have noticed, not being able to calculate the yaw with the accelerometers makes the drift correction for this angle not as well as with the pitch and roll values.

As one may recall the MPU-6050 has an on-board DMP which is able to do a nine-axis sensor fusion between three-axis gyroscopes, accelerometers and optionally also compasses. In theory, if one really wants to be able to correct the long term drift of the yaw as well, this latter three-axis compass (also described in section 3.3.2) should be able to provide the solution by delivering the magnetic north to the gyroscopes. Although even without the implementation of the compasses, the data processed by the DMP appears to give quite accurate yaw values already. One may wonder on how this data fusion has been done, but Invensense doesn’t provide a lot of information about this on-board DMP. Most likely it uses the Invensense motion determination [22]. This algorithm is used to determine

\(^1\)Is actually a steady-state Kalman filter which doesn’t consider any statical description of the noise corrupting the signals. This filter is obtained by a simple analysis in the frequency domain while the Kalman filtering approach works in the time domain. [18]
whether a sensor, that can identify motion (e.g. gyroscopes and accelerometers), has moved or not. Basically when a signal is received from the sensor, the moments of this signal are calculated and compared to determine whether the signal is Gaussian. If so it is recognised as coming from a non-moving sensor while a non-Gaussian one implies an event of motion. The DMP eventually delivers the quaternions, which one could immediately use in his/her applications. However due to little experience with this rotation representation system, in this master dissertation the read-in quaternions are translated into well-known Euler angles using the set of equations\(^2\) in 6.2. By applying this conversion it implies that the order of rotation is considered as first yaw \(\gamma\), then pitch \(\phi\) and finally roll \(\rho\) which is also illustrated in figure 6.2 where roll is the inner circle, yaw the outer and pitch lies in between. To be clear, the hexapod in figure 6.2a is oriented in the same way as the one in figure 6.1 where the Minoru webcam indicates the front of the hexapod. Moreover the direction of the arrows indicate the positive turning angles. Concerning the occurrence of a Gimbal lock\(^3\) one shouldn’t be worried because Lil’ Hex’s body will most likely never be orientated with a pitch of 90°. Although if it would it won’t have any dramatic consequences cause the estimator isn’t using the Euler angles. They are merely obtained from the quaternions which have no gimbal lock issues. The quite satisfying read-out yaw value has also been illustrated by the obtained results of the running straight and turning a desired angle applications, described in section 4.6 and 4.8 respectively.

\[
\begin{bmatrix}
\rho \\
\phi \\
\gamma
\end{bmatrix}
= \begin{bmatrix}
\arctan \left( \frac{2(q_0q_1 + q_2q_3)}{1 - 2(q_1^2 + q_2^2)} \right) \\
\arcsin \left( (2(q_0q_2 - q_3q_1)) \right) \\
\arctan \left( \frac{2(q_0q_3 + q_1q_2)}{1 - 2(q_2^2 + q_3^2)} \right)
\end{bmatrix}
\] (6.2)

\(^2\)In the programs the quaternions were converted using the \texttt{arctan2} function instead of the \texttt{arctan} so that the output range is correct (being \([-180^\circ, 180^\circ]\)).

\(^3\)Gimbal lock arises from the indistinguishability of changes in the first and third Euler angles (here yaw and roll) when the second Euler angle is at some critical value (being pitch at 90°).
6.2 Level out on a changing slope

Let’s start by developing a horizontal levelling-out strategy while standing on a slope of which the inclination could be adjusted. As mentioned earlier, the eventual algorithm makes use of the kinematic model (section 4.1) and in addition it also falls back on other mathematically supported concepts such as basic geometry, the normal vector and the Cartesian plane equation.

6.2.1 Start and calibrate

Representing the complete program in one flowchart would be too overwhelming at first so different parts will be described one by one. Before starting the loop to level out the hexapod’s body, it is set in a configuring position which is considered as horizontal at sight. Hereafter the IMU starts with the calibration procedure while the robot stands still. When the calibration is ended, the initial position of the IMU must be considered as the world coordinate frame, i.e. every movement of the robot will be measured by the gyroscopes w.r.t. this frame (as explained in section 6.1). In order to level-out the body
of the hexapod, these coordinates should be transformed into their body coordinates, i.e. w.r.t. a reference frame fixed to the body. By making use of the yaw angle, pitch and roll w.r.t. to the world reference frame can be easily transformed into the pitch and roll angle w.r.t. the body frame (equation 6.3). This will allow Lil’ Hex to turn around the yaw axis on a slope compared to its initial calibration position and still be able to level-out horizontally. The program structure of these first steps is depicted in figure 6.3. From now on roll, pitch and yaw will also be referred to as $\phi$, $\theta$ and $\psi$. 

$$
\begin{bmatrix}
\phi_{body} \\
\theta_{body}
\end{bmatrix} =
\begin{bmatrix}
\cos(\psi_{world}) & -\sin(\psi_{world}) \\
\sin(\psi_{world}) & \cos(\psi_{world})
\end{bmatrix}
\begin{bmatrix}
\phi_{world} \\
\theta_{world}
\end{bmatrix}
$$

(6.3)

According to figure 6.3, the program will continue when a roll or pitch outside of the threshold values is measured by adding the current measured angle to the previously levelled-out angle. This is important because it prevents a hysteresis between two different angles. For example, if a measured angle of 5 degrees leads to a compensated angle of -0.6 degrees (due to errors) the next loop would send the hexapod back to an angle of 5 degrees because it thinks it’s on a slope of -0.6 degrees. By taking into account this previously measured values, the next loop would try to compensate an angle of 4.4 instead of 5 degrees, which eventually leads to a measuring between the desired boundaries.

![Flowchart: Beginning of horizontal levelling-out.](image)

**Figure 6.3:** Flowchart: Beginning of horizontal levelling-out.
6.2 Level out on a changing slope

6.2.2 Levelling-out strategy

After the calibration procedure and checking the measured roll and pitch of Lil’ Hex’s body, the real levelling-out strategy (depicted in figure 6.6) can start. It should be noted however that most of the times there are multiple leg configurations which are able to level out the six-legged robot, therefore some constraints will need to be predefined.

First of all the levelling-out procedure will start with the current hip angles as a fixed given. This may have the flaw of decreasing the flexibility of overcoming a steeper slope, but it reduces the levelling-out options by a certain amount along with the complexity of the algorithm. Besides, in a first step of climbing the slope (section 6.3) it is desirable that the hips are fixed while levelling in order to smoothly integrate the wanted walking gait.

Secondly the algorithm begins by checking which leg is positioned lowest onto the slope. This can be done by simply looking at the signs from the measured pitch and roll and from dividing Lil’ Hex’s body into quadrants. As shown in figure 6.1 a negative roll implies that the left side of the hexapod’s body is lower than the right side, whereas a positive roll implies the opposite. The same applies for the pitch angle where a negative value indicates that the rear of the robot is standing higher than the front and vice-versa for a positive pitch. This creates four different situations where each time another leg is considered as being at the lowest point onto the slope, a first with LF (pitch < 0, roll < 0), a second with LR (pitch > 0, roll < 0), a third with RF (pitch < 0, roll > 0) and finally with RR (pitch > 0, roll > 0).

Now that the hexapod has determined which leg is situated most downhill, it starts with extending this leg to its maximum limit, thus choosing the minimal reachable y-coordinate known from the kinematic study in section 4.1. This precaution is taken because it increases the chances of completely levelling-out horizontally onto the slope. Together with the constraints of the hip angles mentioned earlier the downhill leg’s actuators are completely determined. For the further explanation of the algorithm let us consider that the hexapod is in its first situation, so that LF is fully extended.

The next steps in the algorithm will define the plane where the feet of the hexapod should be put in order to counteract the measured pitch and roll. Knowing that a plane can be defined with only three points, the procedure still needs to determine two other legs besides

---

4 The same notations will be used in this section as applied in chapter 4. To refresh: LF indicates left front leg, LM the left middle leg and LR the left rear one. Same applies for RF, RM and RR.
6.2 Level out on a changing slope

LF. An inclination about the longitudinal axis (y-axis) can be compensated by creating the same angle as this measured roll value between the right and left side of the hexapod. Accordingly the second leg which will be constrained is, in this case, RF. The height difference between the RF and LF leg tips should be such that it compensates the roll. Following this philosophy, the \((x_{RF,H}, y_{RF,H})\)-coordinates should satisfy the relationship (6.4) so that RF compensates the roll as in figure 6.4. From the kinematics (section 4.1) it is known that both coordinates depend on each other. Starting from (6.5) where \(\alpha\) represents the hip angle of RF and \(v_{RF,H}\) the end effector position of the same leg but w.r.t. to the coordinate frame in the knee servo, one can continue the inverse kinematics and obtain the function \(x_{RF,H} = f(y_{RF,H})\). Hereafter the set of equations (6.4) and (6.5) can be solved in order to find the desired end effector position of RF.

\[
\tan \phi = \frac{y_{LF,H} - y_{RF,H}}{x_{RF,H} + 177 \text{ mm} - x_{LF,H}} \tag{6.4}
\]

\[
x_{RF,H} = (v_{RF,H} + 30 \text{ mm}) \cos(\alpha) = ... = f(y_{RF,H}) \tag{6.5}
\]

Subsequently a third leg location should be determined in order to define a plane. Following the same ideas as with the roll angle but now applied for the pitch, the rear and front of the hexapod robot should compensate this measured pitch value. Therefore in this case, the x- and y-coordinates of LR (opposite to LF w.r.t. transverse axis) need to be determined by satisfying equation (6.6). Like with the roll compensation this will be solved by using the inverse kinematics but now to find the relationship \(z_{LR,H} = f(y_{LR,H})\) (using known hip angle).

\[
\tan(\theta) = \frac{y_{LR,H} - y_{LF,H}}{-z_{LF,H} + 272 \text{ mm} + z_{LR,H}} \tag{6.6}
\]

\[
= \frac{y_{LR,H} - y_{LF,H}}{-z_{LF,H} + 272 \text{ mm} + f(y_{LR,H})}
\]

\(^5\)Subscript RF implies the coordinate frame at the hip servo as represented in figure 6.4 and H refers to the end effector position as applied in section 4.1.
6.2 Level out on a changing slope

Figure 6.4: Front view: Level roll principle.

Figure 6.5: Right side view: Level pitch principle.
Finally, the three points are derived to define a plane which now enables Lil’ Hex to calculate how the remaining legs should be redirected. However the coordinates of the three legs should be transformed to one common coordinate frame before actually setting up the Cartesian equation \((ax+by+cz=d)\) which will characterise the plane. In every case, the chosen common frame is the one at the hip servo of the fully extended leg, thus in this example the hip servo of LF. By choosing a position on the rigid body frame, the coordinates of the other two legs can then easily be translated using the fixed dimensions of the body. Hereafter, using these three points, two directional vectors can be computed and by applying the cross product the normal vector of the searched for plane is obtained. This last vector contains the \(a\), \(b\) and \(c\) parameters of the Cartesian equation, filling in one of the coordinates of one of the legs delivers also the constant \(d\). Hereby, the plane is fully defined and thus the rest of the legs can be brought onto the surface. For example, for RM this is done by solving the Cartesian equation together with the functions \(x_{RM,H} = f(z_{RM,H})\) and \(x_{RM,H} = f(y_{RM,H})\) obtained from the inverse kinematics. The same method can be applied for the remaining legs (RR, LM).

![Figure 6.6: Flowchart: Levelling-out strategy.](image)
6.2.3 Check constraints

Eventually, the retrieved heights (y-coordinates) of the legs should be translated to applicable servo commands. The required servo angles are calculated using the inverse kinematics as depicted in section 4.1, however the mechanical design restricts the possibilities of levelling-out very steep slopes and so the inverse kinematics will not be able to come up with actual servo angles when an unreachable height is used as input. Therefore the unrealistic heights will be restricted to the maximal obtainable y-coordinate. Hereafter, the wanted angles are transformed to the proper PWM signals and send to the actuators (figure 6.7). Once this final step is completed the program goes back to the beginning of the loop by reading in a new set of yaw, pitch and roll (figure 6.3) where after the whole procedure is repeated.

As such Lil’ Hex is now able to level out its body horizontally when placed on a slope of which the inclination can be adjusted. This experiment is portrayed in figure 6.8. Note that Lil’ Hex was still in the development phase (not yet final design).
6.3 Mount a slope

The previous section described the applied levelling-out principles while just standing still on a slope with adjustable inclination. Bringing this together with the running straight application a program has been developed to walk up a slope with its body levelled out horizontally. Hereby the most stable gait, namely the wave gait, has been utilized.

The basic program structure is shown in figure 6.9. After reading in the yaw, pitch and roll value at the beginning of the loop, the level-out algorithm depicted in section 6.2 is applied. This determines the ground plane of the legs, and thus the knee servo angles. After that, the yaw angle is checked and the running straight application kicks in (section 4.6) which adjusts the hip angles. Next, six steps will be taken (thus each leg will move once) using these adjusted settings of the knee and hip angles. After that, new IMU values are read in and the whole process starts over.
6.3 Mount a slope

6.3.1 Results

All of the aforementioned has been tested by letting the six-legged robot mount a slippery glass slope of which the inclination could be readily adjusted. Due to its mechanical limitations the maximum inclination the hexapod can deal with is about 10°. By adding the running straight application every drift due to slippage is compensated immediately and as such Lil’ Hex manages to walk straight to the top of the slope with its body levelled out horizontally. Once at the top Lil’ Hex uses its built-in tactile sensing (FSRs) to detect the ledge and halt. Due to the presence of the non-linear solvers and the processing limitations of the BBB, the body levelling does not occur instantaneously, but with a brief delay. Consequently it takes quite some time in between each six leg movements. However once the transition has been made from standing on the floor to being entirely on the slope, the inclination does not change any more, and Lil’ Hex can continue its path more swiftly without the need for readjusting (and thus computing) its knee angles any longer.

Figure 6.9: Flowchart: Walk up a slope.
Chapter 7

Conclusion

7.1 Achievements

The preconceived goals of making the available six-legged robot fully operational and providing it with the fundamental skill set to wander around certain predefined environments autonomously without putting itself into harm’s way have been accomplished successfully.

Hereto first a thorough literature study has been performed on autonomous mobile robots, hexapods in particular and embedded programming (chapter 2). Based on the acquired knowledge the required components could then be purchased and put together (chapter 3). Once a fully working hexapod was available the gathered insight on programming an embedded computer running Linux could be put to use. To facilitate the debugging process it was chosen to write the programs remotely in an integrated development environment called Eclipse, using mainly Python as a programming language. For each experiment the necessary program could then easily be loaded into the memory of the embedded computer.

Second, after deducing the forward and inverse kinematic model of the legs, the initial experiments focused on enabling the limbed robot to walk and turn using different gaits and speeds (chapter 4). In this regard three particularly interesting statically stable walking gaits have been studied. From this study it could be seen that the fastest one, being the tripod gait, reaching speeds of up to 10 cm/s, is only recommended for use on fairly flat, even terrains, whereas the slowest one, i.e. the wave gait, is advisable when the terrain is more rugged and slopes are to be overcome. What’s more, letting the robot walk straight ahead appeared to be less straightforward than expected (cfr. slippage) and was only made possible by using data obtained from the IMU to adapt the gaits when straying.
Thirdly, once the locomotion part had been investigated, the focus was directed entirely on letting the hexapod robot detect and avoid ledges and obstacles (chapter 5). FSRs were mounted on the leg tips to detect foot-ground contact and obstacle detection is done by means of IR PSDs and a stereo camera. Contrary to the stereo camera, the IR PSDs have not been relied upon for precise distance measurement, they merely serve to detect whether or not an obstacle is present. Moreover it should be noted that the stereo camera has far richer potential than is exploited in this masters’ dissertation thus far. Particularly interesting further research could involve using the camera for visual odometry and mapping (e.g. SLAM as discussed in section 2.2.2 and 2.3.1).

Fourthly an algorithm has been devised to let the six-legged robot level out his body horizontally when placed on a slope of which the inclination can be adjusted (chapter 6). For this purpose fused data of the IMU is used to deliver information on the robot’s body orientation. This capability is then utilised to let the hexapod robot mount slopes of up to 10° inclination while keeping its body in the horizontal position, and thus aiding at a better weight distribution. The latter is particularly important given the far from robust design of the legged robot at hand (i.e. fabricated out of plastic and actuated by means of simple hobbyist servos).

Finally it is appropriate to note all of these capabilities have been brought together using a mainly reactive, behaviour-based control architecture (cfr. section 2.2.3). In particular the control formalism implemented throughout is rule-based, i.e. a set of rules is constructed, and based on which rule is satisfied, the hexapod robot behaves differently.

In summary, the six-legged craft has indeed been given the fundamental building blocks to cope in certain predefined environments. It is able to walk straight with different gaits, turn on the spot, walk in reverse, detect ledges and obstacles and mount slopes of up to 10° inclination with its body levelled out horizontally.

7.2 Further recommendations & research

First of all it is important to note that the furnished building blocks can of course be further optimised and explored. For instance one could aim at improving the speed of the levelling process while mounting a slope, by taking more into account the processing limitations of the Beaglebone Black when programming. Besides this the rich potential of the stereo camera on getting info on the robot’s surroundings is worthwhile to be examined up closer
(hereby considering the Minoru 3D camera is a low-cost device which limits its capabilities somewhat). Next it could also be nice to provide the hexapod the possibility of wireless communication with a remote pc. Lastly instead of the IR PSDs which accompanied the original limbed robot, the use of more commonly applied range sensors in mobile robots such as ultrasonic ones could be considered.

As outlined in the introductory chapter (section 1.3) the realisation of this masters’ project can now serve as a platform for further research concerning hexapods and mobile robots in general. Given the limbed robot is provided with the basic tool set to wander around autonomously without endangering itself, the next obvious step would be enabling the legged robot to navigate through an environment in an intelligent, deliberate way. This is what is referred to in literature as the issue of localisation, mapping and path planning (section 2.3). One interesting route to be considered in this regard is the use of ROS packages implementing visual odometry and mapping using a stereo camera [30, 31].

Other appealing further research topics that have come to mind while executing the masters’ project at hand are the implementation of CPGs for locomotion, energy-efficient walking, or the use of robot learning techniques such as neural networks, reinforcement learning and/or evolutionary algorithms to perform certain tasks.

To conclude, as mentioned explicitly several times throughout, the rather basic mechanical design of the hexapod robot at hand limits its actual application potential to specific predefined environments posing not too large of a challenge. To be able to investigate the true potential of exploration using six-legged robot’s (e.g. dealing with rocks, large rubble, mountain slopes...) one of the recommendations would be building an entirely new hexapod robot with a robust mechanical design (e.g. aluminium frame), dedicated hardware, fully integrated sensors and a larger number of DOFs per leg. Perhaps inspiration could be drawn from the six-legged robot LAURON series being researched upon since 1994 at the FZI in Karlsruhe, Germany (cfr. figure 2.4) [34]. This would of course take research upon the matter to a whole new level.

\footnote{Made out of plastic, simple hobbyist servos, only 2 DOFs per leg...}
Appendix A

Manual

In this appendix an enumeration of the software used throughout is given. First listed are the libraries installed on the BeagleBone Black and thereafter the software used on the PC (running Linux distribution Ubuntu as operating system). Every information regarding software can also be found on the attached CD (i.e. source code or text file with useful sites).

Installing Ubuntu on your PC is certainly advised in order to get acquainted with basic Linux commands which can also be applied onto the BeagleBone Black. Furthermore making connection with the BBB is fairly easy by using the terminal in Linux. In Windows this would require installing Putty.

As already mentioned to facilitate the debugging process we’ve made use of an IDE called Eclipse, but other IDEs are possible as well. Another option is to program the applications directly onto the BBB using the ‘vi’ editor. However, using the latter, is rather inconvenient and doesn’t provide the useful tools to debug like an IDE does.

The attached disk at the end of the masters’ dissertation contains some of the developed programs, some source code, interesting sites, datasheets of several components, measured characteristics and a pdf-file of the final masters’ dissertation.
A.1 Software on BeagleBone Black

- Adafruit-BeagleBone-IO-Python: python library for GPIO control
- pySerial: python library for serial communication
- SymPy: python library used for nonlinear solvers
- DropBoneIMU: C libraries to help with sensor fusion on the IMU
- OpenCV: C++ and python library for image processing

A.2 Software on PC

- Ubuntu: to get acquainted with linux environment and easier interaction with BBB
- Eclipse Kepler: IDE easy to cross-develop (using arm-linux-gnueabi) applications for the ARM architecture.
- Pydev: python libraries to include in Eclipse environment
- Maestro control center: To control Mini Maestro from Pololu and thus servos from the PC (without BBB)
Bibliography


BIBLIOGRAPHY


