Automated Greenhouse

ADIN SUTA AND KARL-FILIP SELANDER
Abstract

In agriculture, growing plants is usually done with soil as the growing medium. An alternative method for soil is using a hydroponic system with plants submerged in water. In this thesis, a hydroponic system was built with the purpose of finding out suitable constants and loop-time of a PID controller to reach optimal pH value for plant growth of a nutrient solution, without intoxicating it. An Arduino uno micro controller was used to turn on and off a water pump, oxygen pump and lights at certain intervals. A pH sensor was used to measure pH-levels in a tank, giving the output from the pH-sensor as the input for the PID-controller. Depending on the output of the PID, two servos opened valves to release either an alkaline or acidic solution into the water reservoir. The results of the tests conducted show that it is possible to find working constants and loop-time for the PID to successfully reach optimal pH of the water reservoir without intoxicating it.
Referat

Acknowledgements

We would like to thank Masoumeh Parseh and Nihad Subasic for the guidelines during the entirety of the project. We would also like to thank the Suta family and the Selander family for providing moral support and for storing plants and equipment which were needed to conduct the experiments. Special thanks goes to Tomas Östberg for giving us the opportunity to conduct the tests at KTH and for providing every necessary tool and equipment the students ask for.
Contents

1 Introduction ................................................. 1
   1.1 Background ........................................... 1
   1.2 Purpose ............................................ 1
   1.3 Scope ............................................. 1
   1.4 Method ............................................ 2

2 Theory ...................................................... 3
   2.1 Hydroponic system .................................... 3
       2.1.1 Nutrient film technique ......................... 4
   2.2 PID control theory for pH neutralization .......... 4
   2.3 pH-values for plants and cation exchange capacity 5
   2.4 Salinity and conductivity of the solution .......... 6

3 Demonstrator ............................................... 7
   3.1 Electronics ......................................... 7
       3.1.1 Microcontroller - Arduino Uno .................. 7
       3.1.2 Servo motors - SG90 MicroServo ............... 7
       3.1.3 pH-sensor - SEN0161 ................................ 7
       3.1.4 Voltmeter - Digital multimeter Fluke 170 ........ 8
       3.1.5 Relay - SainSmart 8 Channel DC 5V Relay ....... 8
       3.1.6 Power Supply Unit (PSU) .......................... 8
   3.2 Hardware ............................................. 8
       3.2.1 System .......................................... 8
       3.2.2 Relay box ....................................... 9
       3.2.3 Lighting ........................................ 10
       3.2.4 pH-buffer distribution ........................... 10
       3.2.5 pH value and measurement of EC ............... 12
   3.3 Software ............................................. 12
       3.3.1 pH-sensor ....................................... 13
       3.3.2 Servo motors .................................... 13
       3.3.3 PID ............................................... 13
   3.4 Test Results .......................................... 13
       3.4.1 Experimental Parameters .......................... 14
List of Figures

2.1 Nutrient film technique hydroponic system [17] ............................... 4

3.1 Complete system, drawn in Lucidchart ................................. 8
3.2 Concept of relay design, drawn in Lucidchart ...................... 9
3.3 Concept of valve design, drawn in Lucidchart ...................... 11
3.4 Flowchart of software process, drawn in Lucidchart ......... 12
3.5 pH-value and PID output over time in test 1 for kp = 20, ki = 2, kd = 0.2. ......................................................... 15
3.6 pH-value and PID output over time in test 2 for kp = 20, ki = 0.1, kd = 0.2. ......................................................... 16
3.7 pH-value and PID output over time in test 3 for kp = 20, ki = 0.1, kd = 0.2. ......................................................... 17
3.8 pH-value and PID output over time in test 4 for kp = 20, ki = 0.1, kd = 0.2. ......................................................... 17
3.9 pH-value and PID output over time in test 5 for kp = 20, ki = 0, kd 0.2. 18
3.10 pH-value and PID output over time in test 6 for kp = 20, ki = 0, kd 0.2. 18
3.11 pH-value and PID output over time in test 7 for kp = 15, ki = 1, kd 0.2. 19
3.12 pH-value and PID output over time in test 8 for kp = 15, ki = 0, kd 0.2. 20
List of Tables

3.1 EC values in the beginning and end of test 1 to 8. . . . . . . . . . . . . . 20
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pH</td>
<td>Potential hydrogen</td>
</tr>
<tr>
<td>PID</td>
<td>Proportional–integral–derivative</td>
</tr>
<tr>
<td>EC</td>
<td>Electrical conductivity</td>
</tr>
<tr>
<td>CEC</td>
<td>Cation exchange capacity</td>
</tr>
<tr>
<td>PSU</td>
<td>Power supply unit</td>
</tr>
<tr>
<td>AC</td>
<td>Alternating current</td>
</tr>
<tr>
<td>DC</td>
<td>Direct current</td>
</tr>
<tr>
<td>kBs</td>
<td>kiloByte per second</td>
</tr>
<tr>
<td>BNC</td>
<td>Bayonet Neill–Concelman</td>
</tr>
<tr>
<td>I/O</td>
<td>Input/Output</td>
</tr>
<tr>
<td>UV</td>
<td>Ultraviolet</td>
</tr>
<tr>
<td>CFL</td>
<td>Compact fluorescent light</td>
</tr>
<tr>
<td>Kp</td>
<td>Proportional coefficient</td>
</tr>
<tr>
<td>Ki</td>
<td>Integral coefficient</td>
</tr>
<tr>
<td>Kd</td>
<td>Derivative coefficient</td>
</tr>
</tbody>
</table>
Chapter 1

Introduction

1.1 Background

Farming has traditionally been done using soil and big acres of land for the crops to grow. Using a hydroponic system allows for space efficient solutions to grow plants along with efficient distribution of nutrients. The plants could be grown in levels on top of each other without soil, which could be an improvement to farming.

In a hydroponic system, the plants are grown with water as a growing medium instead of soil. The nutrition is concentrated minerals diluted in the water. This way of growing plants opens up possibilities for skyscraper greenhouses, which could be built in areas of the world where farmable lands are scarce or even nonexistent.

1.2 Purpose

A miniature greenhouse will be built and automated to explore the possibility of creating an alternative to farming crops. The research question of the project is to find what are the values for the loop-time and constants of a proportional-integral-derivative controller (PID controller) to neutralize the pH value of a nutrient solution, used in a hydroponic system, without intoxicating the environment. The electrical conductivity (EC) of the solution is an indicator of the toxicity and will be measured before and after the PID controller has neutralized the pH value.

1.3 Scope

In a hydroponic system there are several parameters to be controlled and regulated such as: temperature of nutrient solution and air, pH and EC of nutrient solution, nutrient pump, oxygen pump and lighting. The focus of the project is on the axiomatization of the hydroponic system and controlling the pH-value and monitoring the EC-values in the nutrient solution. In order to meet the purpose and answer the research question of this work an automated hydroponic system was built, along with automatic pH solution-release.
CHAPTER 1. INTRODUCTION

1.4 Method

The hydroponic system contains lighting, nutrient and oxygen pumps, pH-buffer, nutrient distribution, and utilizes the nutrient film technique. One basic solution and one acidic solution were released automatically via a PID-controlled algorithm. The pH is measured by using a pH-meter connected to an Arduino, and the EC is measured by manually checking the conductivity of the water with a multimeter.

The lighting, water pump and oxygen pump can all be controlled via the Arduino. A box containing the Arduino and all the electrical components was built to simplify the programming, so that the focus can lie on optimizing the controller.
Chapter 2

Theory

2.1 Hydroponic system

Growing plants has traditionally been done with soil as the growing medium. An alternative growing medium is using a nutrient solution mainly composed of water, with minerals diluted in the water. The roots of the plants pick up nutrients from the solution to grow and develop. This kind of system is called a hydroponic system. There are several different basic types of hydroponic systems which all share the same basic principle described before. The systems can differ in some aspects such as some use either moving parts or no moving parts, recovery or non-recovery, but overall the ingoing components are the same. A reservoir containing nutrient solution which is either pumped with a water pump to the roots (moving) or the roots are submerged in the reservoir (no moving). The excess water from the moving system can either be recollected in the reservoir (recovery) or not (non-recovery). Apart from nutrients the plants require light and air for the photosynthesis.

The hydroponic system which is going to be utilized in this report is the nutrient film technique.
2.1.1 Nutrient film technique

The basic concept of NFT is to place plants in a tilted container of a certain length as to create a gentle tilt, where a recirculated (with a water pump nutrient solution) streams over the exposed roots, from the higher end to the lower end of the container. From the container the nutrient solution drains into a reservoir, where it is aerated with a air pump and the pH-value, electrical conductivity and temperature monitored. From the reservoir the solution is pumped into the container[1]. Concept model of NFT is illustrated in Fig. 2.1.

2.2 PID control theory for pH neutralization

The pH-value and the EC is going to be controlled using a PID controller coupled with the Arduino. A PID controller is usually used for linear systems since the linear changes in the system easily can be controlled with linear parameters[7]. Since the pH-value is not expected to behave linearly, it poses a problem to control the system with a linear controller.

Although the pH-gauge is directly connected to the Arduino, the pH-value in the solution does not stabilize immediately. Therefore the loop-time of the PID and size of the output (amount of pH-buffer or nutrient solution released) needs to be adjusted according to how fast the system changes[8]. This is done by creating an algorithm which checks how much the last change affected the system. If the
2.3 PH-VALUES FOR PLANTS AND CATION EXCHANGE CAPACITY

The pH-value only changed by a small amount, the amount of fluid released at each cycle is increased and vice versa.

The system will also change over time which creates problems because of the changes of the characteristics of the system. Since the plants are continuously growing, the hydrogen ion release from the plants is expected to increase overtime. Therefore the nutrient solution’s pH might drop below what is acceptable for plant growth, making a pH stabilizing algorithm necessary. The EC is also used in the system as a feedback value to more describe the systems properties regarding the ion concentration of the water. It is also giving out a warning when the water in the solution needs to be swapped, since the salinity is too high, thus creating a toxic environment for the plants.

2.3 pH-values for plants and cation exchange capacity

The cation exchange capacity (CEC) of a soil is often measured as an indicator of a soils fertilization capacity. Higher values of CEC means that the soils number of exchangeable cations per dry weight is higher. This indicator can also be described as the number of hydrogen ions necessary to fill the soil cation holding sites per 100 grams of dry soil. Since plants release cations (hydrogen ions) to exchange them for cations in the soil, the pH-value naturally increases in the soil and is therefore also an indicator of how the soil consequently should be approached. Using pH-buffers, nutrient solutions or exchanging the water should all be possibilities to consider to optimize the fertility of the growing environment for the plants.
Most plants thrive in pH-values ranging from 5 to 7 since the general maximum ion uptake occurs between these values. There is however a difference between growing plants in soilless culture versus growing them in soil or peat. The CEC is very restricted in soilless cultures and can therefore pose a problem in growing the plants, making the system more vulnerable to changes in pH-values or nutrient concentration. The roots release hydrogen ions when cations are taken up more rapidly than anions (the solution pH falls); conversely, the roots release bicarbonate and hydroxyl ions when anions are taken up more rapidly than cations (the solution pH rises)\cite{2}. In practice it seems that pH normally increases with time. However there are often observations in an initial drop in the pH of nutrient solution in contact to plants. This happens due to the uptake of NH$_4^+$ ions in preference to NO$_3^-$ ions, exchanging H$^+$ ions in the solution from the plant. A 10\% solution of KOH is suggested to raise the pH value and have a pH controller linked to the nutrient to maintain the pH between 5.5-6.5\cite{1}. There is also the possibility to use a second pH controller to avoid over dosage of acid and ruining the crops. The second pH sensor is then used as an alarm which triggers when the values are extreme (outside of 5-7 for example).

The properties of the water added also influences the pH in the solution. Hard water (pH greater than 7) increases the rapidity with which the changes in pH occur making the need of adding acids to increase the pH. This increased supplementation of acid results in steady accumulation of nitrate, phosphate or sulfate depending on the acid used. As a result, the water needs to be leaked or replaced on a regular basis to avoid the augmentation of salinity in the water\cite{1}.

### 2.4 Salinity and conductivity of the solution

Conductivity is a measurement of the solution's ability to pass current. The concentration of ions in the water is directly related to the conductivity, thus making it a relevant factor to take into consideration when establishing the properties of the solution in order to control it correctly. The dissolved ions come from the release of ions from the plants, the ions from the added nutrients and the ions of the added acid. The latter is related to the salinity of the water and is an indicator that the water should be replaced before any plants take damage.

Since a complete chemical analysis for every measurement of salinity is not practical in this study, the conductivity should be a good practical indicator of the salinity in the solution. This may indicate that water should be leaked or replaced. However, the conductivity of the solution provides no information on how the concentration of ions is distributed (what type of ions are present and the concentrations of these) which is a drawback of this method. There are however methods to finding the relationship between conductivity, temperature, nutrient deficiencies and how fertilizers contribute to the change in electrical conductivity. This should give a good enough general estimate of these properties of the solution\cite{4}. Therefore a voltmeter measuring the conductivity at regular intervals\cite{3}.
Chapter 3

Demonstrator

3.1 Electronics

The system in regards to electronics consist of a microcontroller, two servo motors, one pH-sensor, one volt-meter, a 8-channel relay and a power supply unit (PSU).

3.1.1 Microcontroller - Arduino Uno

The Arduino Uno is a microcontroller[11] with digital input/output pins and analog input pins. It can be powered by a computer through USB or with a battery/alternating current (AC) to direct current (DC) adapter. It’s operating voltage is 5V and can run programs of up to 32 kilobyte per second (kBs).

3.1.2 Servo motors - SG90 MicroServo

The servo motor SG90[12] weighs 9g and can pull up to 800g. It is suited for a Arduino or Raspberry pi. It requires 5V to operate and has a circular movement range of 180°. It takes input signals from one of the digital pins of the Arduino.

3.1.3 pH-sensor - SEN0161

The SEN0161 [13] is specifically designed for Arduino controllers. It has a Bayonet Neill-Concelman (BNC) connector and pH2.0 sensor interface. By connecting the pH-sensor to the BNC connector, and plugging in the pH2.0 interface to an analog input on the Arduino board it will provide pH-level readings. Its operating voltage is 5V, with a pH measuring range of 0-14. It will provide an accuracy of +0.1 pH at 25° celsius. The output of the pH-electrode is millivolts and the relation between voltage and pH which differs for different temperatures is available in a chart provided by the manufacturer [13].
3.1.4 Voltmeter - Digital multimeter Fluke 170

The voltmeter Fluke 170\cite{14} when connected to two probes can be used to measure the resistance in water. It can measure up to 50 M with an accuracy of $\pm (0.9\% + 1)$.

3.1.5 Relay - SainSmart 8 Channel DC 5V Relay

The SainSmart relay is able to handle high-current AC250V 10A or DC30V 10A\cite{15}. It turns on and off the relays when a current of 5V runs through it and so it’s well suited for an Arduino.

3.1.6 Power Supply Unit (PSU)

To decrease the amount of current the Arduino needs to provide for the equipment connected to it a regular PSU is used. The PSU can supply 5V and 2.1A and is connected to all components which require power.

3.2 Hardware

3.2.1 System

The complete greenhouse will facilitate light, nutrients, water and air for the plants to grow.

![Figure 3.1. Complete system, drawn in Lucidchart](image)

Everything needed for the greenhouse is numbered in the conceptual illustration of the greenhouse in Fig. 3.1 from 1 to 12. The Arduino (1) is the foundation of this setup since it will control every aspect of the greenhouse. The lights (9) to provide light for the plants at an interval, air pump (4) to aerate the nutrient
3.2. HARDWARE

solution, water pump (3) to provide the plants with nutrients, are all controlled by the Arduino (1), separately through the relay box (5). The container (7) where the plants are set is stationed on the table (6) at an angle, alfa (8), to create a slope, where the water will pour downstream from the left end to the right end, into the reservoir (2). In the reservoir the nutrient solution is contained, the pH-value and EC of the solution is measured, and monitored with a pH-gauge (12), and voltmeter (x). The measured values are then fed back to the Arduino (1), which in return controls a valve construction (11), which dispenses either a diluted nutrient solution or pH-buffer from the nutrient solution container (10).

3.2.2 Relay box

Lights, water and air pump will be turned on and off at a predetermined interval from the relay box. Everything needed for the relay box is numbered in Figure 3.2.

![Figure 3.2. Concept of relay design, drawn in Lucidchart](image)

An extension cord (1) is wired into the box (7) where the cord is divided into its three electrical conductors, the red wire(phase) going to the switch (2), and then going to the flat pin (3), where also the brown (neutral) and black (zero) is wired. The three wires are then divided into several sets of wires (red, brown, black). One red wire from every set goes to a separate channel on the relay (6), from the relay (6) every wire is connected to one outlet each. From the flat pin the corresponding brown and black wire to the red wire is connected to the same outlet. The Arduino
(5) is connected to the relays general purpose input/output (I/O) pins and can turn on and off every channel and in turn each outlet by itself. The pH sensor (8) is outside of the box and is connected to the pH circuit board with a BNC connector, the circuit board in turn is connected to the Arduinos analog input pin. The servo motors (9) are outside of the box and are connected to the Arduinos general purpose I/O pins.

3.2.3 Lighting

Plants respond to a wide spectrum of light, ranging from ultraviolet to far-red. The regions of light, which plants require to grow and develop, can be divided into UV (400 nm), the visible light (400-700 nm) and far-red (700-800) [5]. A study published in Botanical Gazette they’ve found that plants grown under light of 6500 lux for 16 hours a day emitted by compact fluorescent (CFL) lights produced plants quicker, with sturdier stems, largest and most numerous leaves of the deepest green color, compared to their control group [6]. These lights were used in the greenhouse.

3.2.4 pH-buffer distribution

For the regulation of pH-value there are two containers stationed above the reservoir, see component 10, 11 and 2 in 3.1. One container for a solution to raise the pH-level (pH-up) and one for a solution to decrease it (pH-down). The concept for the distribution is illustrated in Fig. 3.3.
3.2. HARDWARE

The pH-up container will be stationed above the left pipe and the pH-down above the right pipe. Two valves are connected to each pipe with a tube with a 9 mm diameter and 80 mm long. The tube will have a hole in it where a straw is placed to release air. One servomotor is placed between one valve from each pipe. When the servo turns $\pm90^\circ$ one valve opens, the fluid from the container will be forced down by gravity into the space between the two valves of one pipe and replace the air contained within. The space will contain the desired control volume required for the present system. When the servo returns to its original position the opened valve will close shut. A second servomotor is placed between the lower valve of each pipe. When it turns $\pm90^\circ$ one valve will open and the tube between the two valves of the same pipe will release its control volume and the liquid will be forced out into the reservoir stationed below.

Figure 3.3. Concept of valve design, drawn in Lucidchart
3.2.5 pH value and measurement of EC

The pH-value in the nutrient solution should constantly be monitored. The pH-sensor submerged in the reservoir sends measurements of the pH-levels to the Arduino. Depending on the input from the pH-meter the Arduino will decide whether to increase, decrease or do nothing to change the pH-level of the reservoir.

Once the pH reference value is reached the EC-value will be measured. This is done with a voltmeter with high values of EC corresponding to high salinity according to CWT 2004[9]. If the EC gets too high it indicates the salinity is too high and this could potentially harm the plants. This means that the reservoir water needs to be leaked or replaced.

3.3 Software

The software acts upon one input, which is the pH-level of the tank measured by the pH-gauge. It implements a PID controller which calculates a suitable output, which is either increasing or decreasing the pH-level by moving two servos in succession, letting pH-up or pH-down fluid flow into the tank. The flow chart of the process is illustrated in Fig.3.4. The code for the software can be found in appendix A.

![Flowchart of software process, drawn in Lucidchart](image-url)
3.4. TEST RESULTS

3.3.1 pH-sensor

The output of the pH-sensor is in millivolts and so a conversion from millivolts to pH is required. The correct values of pH can be found from either the charts on the manufacturers website [13], or from the code that they provide found in appendix A. The pH-sensor needs to be calibrated and it is done by measuring two fluids with known pH-levels and the error which is calculated by subtracting the known pH by the measured pH. The numerical value will be added to the code to account for the offset. For our specific pH-gauge it had an error of 0.1. The code for the pH-sensor can be found in appendix A lines 73 to 97. On line 95 the error 0.1 is added to the final phValue.

3.3.2 Servo motors

The rotation angles of the servo motors were measured by allowing the servos rotate to increasingly larger angles until it opened the valves. The delays in the code appendix A lines 36 to 71, allow enough time for the fluids to fill the control volume completely, and for the servos to close before the other opens. The for loops condition variable times depends on the output of the PID controller, with a maximum value of 5 and a minimum value of 1. The control volume will be released a number of times within the interval of 1-5 times.

3.3.3 PID

The code for the PID used in this project is a library provided by the Arduino Playground called Arduino PID Library written by Brett Beauregard, appendix B. A few changes has been made to the original code to be more suitable for the system evaluated in this report.

In appendix A lines 136 to 155 there have been added to the original Compute() function found in appendix B lines 16 to 39. The if statement at line 136 is a workaround of the problem of integer overflow when trying to have a sampling time of 1 minute or more. Variables tnow and previoustime uses the built in function millis() which calculates the time since the program started running. There are two intervals in the Compute() function, one at line 119 and one at 136. The interval 119 decides how often the PID controller computes an output as a function of the input but the interval on line 136 allows it to act upon it. The final if-statement at line 141 inhibits the PID controller to do anything when the pH-level is within an acceptable range. The input, output and elapsed time is printed in the compute function, to gather data for the tests.

3.4 Test Results

The conducted tests are meant to represent a problem where the pH-value rises or dips to levels that prevent optimal ion uptake from plants in a hydroponic system.
The pH-sensor mentioned above has been used to measure the pH-value of the solution. The output values of the sensor have been saved from the Arduino and then plotted in a graph over time in MATLAB. Different starting points for the pH-value have been used which show different problems that occur in hydroponic systems.

The EC of the water is calculated by pouring water into a glass of known height (h), length (l) and depth (d) of water and measure the resistance(Ω) of the water with a voltmeter. The following equation 3.1 calculates the EC\[10\].

\[
EC = \frac{l}{\Omega \times h \times d}
\]  

(3.1)

if all lengths are in cm and resistance in Ω multiply it by 1000 to get mS/Cm which is a common way of denoting it.

3.4.1 Experimental Parameters

Changes in values of the parameters Kp, Ki and Kd in the controller show how the characteristics of the control changes when these parameters change. The coefficient Kp determines how aggressively the PID reacts to the current error, Ki determines how aggressively the PID reacts to error over time and Kd determines how aggressively the PID reacts to changes in error. By changing the parameters one by one and testing it in different conditions (pH values in our system’s case), the tuning of the controller becomes evidently clearer. With these experiments, both the parameters and start pH-values were changed. The changes in the starting values in the tank shows that the system can handle both high and low pH-values approximately equally.

It should also be mentioned that the initial values of parameters Ki, Kp and Kd were not chosen arbitrarily. A short list in the code was made with some constant values of pH and used as an input. Using trial and error and the understanding of how the individual constants of the PID-controller change the properties of the controller, the initial values of Kp = 20, Ki = 2 and Kd = 0.2 were used.
3.4. TEST RESULTS

3.4.2 Results

The test results are illustrated in Fig. 3.5 to Fig. 3.12, in which the change in pH-value and the output of the PID represent the y-axis which are plotted over the x-axis time. The graphs were plotted in MATLAB version 2016a.

The software and system was initially tested in a base solution of pH-level around 9 for some set values of Kp, Ki and Kd. The results are presented in Fig. 3.5. The output is at its max value constantly without a significant change in pH-level. The pH-down solution was too weak to change the pH-value of the tank.

![Graph showing pH-value and PID output over time in test 1 for kp = 20, ki = 2, kd = 0.2.](image)

The **Compute()** in the software has a pH-value interval from 5.8 to 6.2 where the PID is inactive. The output within that interval should be low and the pH-level should be unchanged over time. This was tested at a pH-level of just below 6 in the tank. The results can be found in Fig. 3.6. The output of the PID is less than 10 and positive, which is correct since the pH-value is just below the reference value. The pH-level is not changing so the software is working as it should.
Fig. 3.6. pH-value and PID output over time in test 2 for $k_p = 20$, $k_i = 0.1$, $k_d = 0.2$.

Fig. 3.7, 3.8, 3.9 and 3.10 are the results of tests for a set concentration of pH-up (1 litre water and 30ml Potassium hydroxide 50%) and -down (1 litre water, 30ml phosphoric acid 81%). The first three were tested for acidic levels in the tank and the last one for a base level in the tank. Parameters $K_p$ and $K_d$ were the same for all the tests, $K_i$ was changed from 0 in test one and two to 0.1 in tests illustrated in Fig. 3.9 and 3.10. The constants used for results in Fig. 3.9 had the fastest settling time. The constants used for results in Fig. 3.10 show some oscillation but the neutralization did not settle due to the pH-up solution leaked in the test and so the pH-level couldn’t change because no pH-up solution went into the tank.
3.4. TEST RESULTS

**Figure 3.7.** pH-value and PID output over time in test 3 for $kp = 20$, $ki = 0.1$, $kd = 0.2$.

**Figure 3.8.** pH-value and PID output over time in test 4 for $kp = 20$, $ki = 0.1$, $kd = 0.2$. 
CHAPTER 3. DEMONSTRATOR

Figure 3.9. pH-value and PID output over time in test 5 for $kp = 20$, $ki = 0$, $kd = 0.2$.

Figure 3.10. pH-value and PID output over time in test 6 for $kp = 20$, $ki = 0$, $kd = 0.2$. 
3.4. TEST RESULTS

The results presented in Fig. 3.11 and 3.12 were conducted with a different concentration in pH-up (1 litre water, 45ml potassium hydroxide 50%) and pH-down (1 litre water, 45 ml phosphoric acid 81%). The tests were conducted with a base ph-level as starting point in the tank. Parameters Kp and Kd were the same for both the tests, Ki was changed from 1 in test one to to 0 in test two. In Fig. 3.11 the pH-value settles within an hour with some overshoot. In Fig. 3.12 there is no overshoot and the pH-value settles within 30 minutes. Overall the controller seems to perform better without the integral part, which is also the case for Fig. 3.9 and 3.10 compared to Fig. 3.9 and 3.8.

Figure 3.11. pH-value and PID output over time in test 7 for kp = 15, ki = 1, kd 0.2.
CHAPTER 3. DEMONSTRATOR

Figure 3.12. pH-value and PID output over time in test 8 for kp = 15, ki = 0, kd = 0.2.

The resistance was measured and the EC-value calculated with equation 3.1 as explained in 3.4 after each test and compared to toxic EC-values for plants. The values from the tests in 3.1 compared to the toxic values for plants presented in the table found at [16] indicate that the nutritional value was low but the environment was not toxic. Any of these neutralization processes could probably be done several times before the environment becomes too toxic for the plants to grow in.

<table>
<thead>
<tr>
<th>Test</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>EC Start [$\mu$S/cm]</td>
<td>0.0010</td>
<td>0.0010</td>
<td>0.0010</td>
<td>0.0011</td>
<td>0.0010</td>
<td>0.0010</td>
<td>0.0010</td>
<td>0.0010</td>
</tr>
<tr>
<td>EC End [$\mu$S/cm]</td>
<td>0.0013</td>
<td>0.0010</td>
<td>0.0021</td>
<td>0.0017</td>
<td>0.0017</td>
<td>0.0022</td>
<td>0.0028</td>
<td>0.0023</td>
</tr>
</tbody>
</table>

Table 3.1. EC values in the beginning and end of test 1 to 8.

Different loop-times and sampling times were tested for the PID controller. A loop-time of 30 minutes was tested but it took too long for the pH-value to change. A loop-time of less than 1 minute was also tested but it did not allow enough time for the servo motors to release the equivalent amount of control volume to the PID controllers output. All the tests and corresponding graphs have been done with the loop-time of 2 minutes.
Chapter 4

Discussion and conclusion

4.1 Discussion

The plants impact on the pH-level of the water tank were simulated in order to test the pH-neutralization process. In the theory 2.3 it was stated that the pH-level would eventually drop due to the uptake and release of ions when the plants grow. The change in pH would take several days to drop to the level where the PID controller was tested at. The pH scale is a 10 base logarithmic scale and so it is hard to predict intuitively how it behaves. In the research question it was stated the process would be tested with different loop times, Kp, Ki and Kd values to find the fastest way possible to neutralize the pH-level.

There is a similarity between the physical properties of the system and the parameters of the PID controller used in this report. For example, if the pH-solution released from the servos is to be more concentrated, the system will be reaching the set point faster but will increase the overshoot as well. This is similar to increasing the Kp constant in the PID controller. Therefore, it is vital to know the physical properties of each system before making any substantial changes to the parameters.

The integral part of the controller has a purpose of making the output larger if the system has had a large error for a longer period of time, which is beneficiary. It also has a property of looking at the system’s total error and can therefore overshoot the pH-value once it has reached the set point. Since the goal of reaching a set point in a hydroponic system is to make the environment as endurable for the plants as possible, the integral part can many times be neglected fully. This creates a PD-controller which makes the pH-value go to a set point as fast as possible, without creating unnecessary overshoot in the system. One drawback is, as mentioned above, that the output doesn’t change because of an error stretching over a large time.
4.2 Conclusion

The parameters which were arrived at in 3.4 reached the set point of pH = 6 as fast as possible in the system used in this report. It should be noted however, that the physical properties of the system need to be counted in when conducting tests which determine Ki, Kp and Kd. These parameters affect the pH value along with the system’s amount of water in the tank, the concentration of the pH-down and -up solutions, the size of control volume, the range of the output of the controller and the conversion to how many times the control volume can be released. To properly tune all of these parameters for a specific system requires extensive testing and should be regarded as individual, unless the system’s properties are similar to the ones tested. The tests conducted are elementary, but can serve as a guideline to how a large offset from the set point of pH = 6 can be handled. The parameters listed in 1.2 were tested and a PID controller which arrived at pH = 6 fast and reliably was constructed from the tests conducted.

Since the EC and toxicity of the solution is directly affected by release of pH-solution in a hydroponic system, it should be used only when it is necessary. Using a PID-controlled pH-equalizer as described in this report should primarily be used if the pH-values become extreme (pH = 1-4 or pH = 9-12 for example) and can be used as a pis aller in an automated hydroponic system, if supervision of the system should fail.
Bibliography


Appendix A

Software code

Code for the Arduino to regulate pH-levels, including code for pH-sensor, servo motors and PID-controller.

```c
#include <Servo.h>

#define RELAY1 6
#define RELAY2 7
#define RELAY3 8
#define SensorPin 0 //pH meter Analog output to Arduino

unsigned long int avgValue; //Store the average value of the sensor feedback

unsigned long hour = 3600000, previoustime = 0, previoustime2 = 0, previoustime3 = 0; //hr in milliseconds, time for relays 1 and 2

int buf[10], temp, relayState1 = LOW, relayState2 = LOW, servoOutput;

Servo top, bottom; //Define our Servo

double pHtest[10] = {7.2, 7.21, 7.1, 6.98, 6.7, 6.9, 6.7, 6.6, 7.9};

int b= 0;

double lista_pH [100];

double lista_output [100];

int raknare = 0;

unsigned long lastTime;

double Input, Output;

double Setpoint = 6; //referens

double Iterm, lastInput;

double kp = 20, ki = 0.1, kd = 0.2;

unsigned long SampleTime; //1 sec

double outMin, outMax;

bool inAuto = false;
```
APPENDIX A. SOFTWARE CODE

30#define MANUAL 0
31#define AUTOMATIC 1
32#define DIRECT 0
33#define REVERSE 1
34
35int controllerDirection = DIRECT;
36
37void pH_up(int times) { // Right behind
38    times = times * -1;
39    for (int i = 0; i < times; i++) {
40        // top servo open and shut with 10s delay for liquid to fill
41        top.write(113);
42        delay(1000 * 4); //
43        top.write(90);
44        delay(1000 * 2);
45    }
46    // w8 10s
47    delay(1000 * 2);
48    // bottom servo open and shut with 10s delay for liquid to flow out
49    bottom.write(50);
50    delay(1000 * 4); //
51    bottom.write(90);
52    delay(1000 * 2);
53}
54
55void pH_down(int times) { // Left behind
56    for (int i = 0; i < times; i++) {
57        // top servo open and shut with 10s delay for liquid to fill
58        top.write(55);
59        delay(1000 * 4);
60        top.write(90);
61        // w8 10s
62        delay(1000 * 2);
63        // bottom servo open and shut with 10s delay for liquid to flow out
64        bottom.write(122);
65        delay(1000 * 4);
66        bottom.write(90);
67        delay(1000 * 2);
68    }
69}
70
71float pHmeasure() {
72    for (int i = 0; i < 10; i++) // Get 10 sample value from the sensor
73        for smooth the value
74        { buf[i] = analogRead(SensorPin);
75            delay(10);
76        }
77    for (int i = 0; i < 9; i++) // sort the analog from small to large
78    { 
79
for (int j = i + 1; j < 10; j++)
{
    if (buf[i] > buf[j])
    {
        temp = buf[i];
        buf[i] = buf[j];
        buf[j] = temp;
    }
}

avgValue = 0;
for (int i = 2; i < 8; i++) // take the average value of 6 center sample
    avgValue += buf[i];
float phValue = (float)avgValue * 5.0 / 1024 / 6; // convert the analog into millivolt
phValue = 3.5 * phValue + 0.1; // convert the millivolt into pH value and add 0.1 from calibration
return phValue;

void Motorer( double output) {
    double roundoutput = round(output / 10);
    int integer_output = int(roundoutput);
    
    if (integer_output > 0) {
        pH_down(integer_output);
    }
    else {
        pH_up(integer_output);
    }
}

void Compute() {
    if (!inAuto) return;
    unsigned long now = millis();
    int timeChange = (now - lastTime);
    if (timeChange >= SampleTime)
    {
        /* Compute all the working error variables*/
        double error = Setpoint - Input;
        ITerm += (ki * error);
        if (ITerm > outMax) ITerm = outMax;
        else if (ITerm < outMin) ITerm = outMin;
        double dInput = (Input - lastInput);
        /* Compute PID Output*/
        Output = kp * error + ITerm - kd * dInput;
        if (Output > outMax) Output = outMax;
        else if (Output < outMin) Output = outMin;
    }
}
/*Remember some variables for next time*/
lastInput = Input;
lastTime = now;
if (tnow - previousTime > hour / 30) {
    previousTime = tnow;
    lista_pH[raknare] = Input;
    lista_output[raknare] = Output;
    raknare = raknare + 1;
    if (Input < 5.8 or Input > 6.2) {
        Motorer(Output);
    }
}
for (int i = 0; i < raknare; i++) {
    Serial.print("pH: ");
    Serial.print(lista_pH[i]);
    Serial.println("Output ");
    Serial.print(lista_output[i]);
    Serial.println(" ");
}
Serial.println("raknare");
Serial.println(raknare);
}

void SetTunings(double Kp, double Ki, double Kd)
{
    if (Kp < 0 || Ki < 0 || Kd < 0) return;
    double SampleTimeInSec = ((double)SampleTime) / 1000;
    kp = Kp;
    ki = Ki * SampleTimeInSec;
    kd = Kd / SampleTimeInSec;
    if (controllerDirection == REVERSE) {
        kp = (0 - kp);
        ki = (0 - ki);
        kd = (0 - kd);
    }
}

void SetSampleTime(int NewSampleTime)
{
    if (NewSampleTime > 0)
    {
        double ratio = (double)NewSampleTime / (double)SampleTime;
        ki *= ratio;
        kd /= ratio;
        SampleTime = (unsigned long)NewSampleTime;
    }
void SetOutputLimits(double Min, double Max)
{
    if (Min > Max) return;
    outMin = Min;
    outMax = Max;
    if (Output > outMax) Output = outMax;
    else if (Output < outMin) Output = outMin;
    else if (ITerm > outMax) ITerm = outMax;
    else if (ITerm < outMin) ITerm = outMin;
}

void SetMode(int Mode)
{
    bool newAuto = (Mode == AUTOMATIC);
    if (newAuto == !inAuto)
    { /*we just went from manual to auto*/
        Initialize();
    }
    inAuto = newAuto;
}

void Initialize()
{
    lastInput = Input;
    ITerm = Output;
    if (ITerm > outMax) ITerm = outMax;
    else if (ITerm < outMin) ITerm = outMin;
}

void SetControllerDirection(int Direction)
{
    controllerDirection = Direction;
}

void setup()
{
    Serial.begin(9600);
    top.attach(9);  // servo on dp 9
    bottom.attach(10);  // servo on dp 10
    pinMode(RELAY1, OUTPUT);  // oxygenpump
    pinMode(RELAY2, OUTPUT);  // lights
    pinMode(RELAY3, OUTPUT);  // waterpump
    pinMode(13, OUTPUT);
    digitalWrite(RELAY3, LOW);
    digitalWrite(RELAY1, LOW);  // Turns ON Relays 3 8 for waterpump
//PID
//SetTunings(10, 1, 0.1);
SetControllerDirection(DIRECT);
SetMode(AUTOMATIC);
SetOutputLimits(-50, 50);
Setpoint = 6;
SampleTime = 1000 * 30;

void loop()
{
    //RELAY
    tnow = millis(); // time after program started running
    //interval fpr relay 1 Blink Without Delay method for oxygenpump
    if (tnow - previoustime > hour * 2)
    {
        previoustime = tnow;
        Serial.println(Output);
        if (relayState1 == LOW)
        {
            relayState1 = HIGH;
        }
        else
        {
            relayState1 = LOW;
            digitalWrite(RELAY1, relayState1); // turn on or off oxygen pump
        }
    }
    else
    {
        relayState1 = LOW;
        digitalWrite(RELAY2, relayState2); // turn on or off light
    }
    //PID, input/output och Print ln till serial Monitor
    Input = (double)pHmeasure();
    Compute();
}
Appendix B

Original PID code

Original code of PID-controller, name Arduino PID Library written by Brett Beau-regard.

```c
/*working variables*/
unsigned long lastTime;
double Input, Output, Setpoint;
double ITerm, lastInput;
double kp, ki, kd;
int SampleTime = 1000; //1 sec
double outMin, outMax;
bool inAuto = false;

#define MANUAL 0
#define AUTOMATIC 1
#define DIRECT 0
#define REVERSE 1

int controllerDirection = DIRECT;

void Compute()
{
  if(!inAuto) return;
  unsigned long now = millis();
  int timeChange = (now - lastTime);
  if(timeChange>=SampleTime)
  {
    /*Compute all the working error variables*/
    double error = Setpoint - Input;
    ITerm += (ki * error);
    if(ITerm > outMax) ITerm = outMax;
    else if(ITerm < outMin) ITerm = outMin;
    double dInput = (Input - lastInput);
    /*Compute PID Output*/
    Output = kp * error + ITerm - kd * dInput;
    if(Output > outMax) Output = outMax;
    else if(Output < outMin) Output = outMin;
    /*Remember some variables for next time*/
  }
}
```
36    lastInput = Input;
37    lastTime = now;
38 }
39}
40
41 void SetTunings(double Kp, double Ki, double Kd)
42{
43    if (Kp<0 || Ki<0|| Kd<0) return;
44    double SampleTimeInSec = ((double)SampleTime)/1000;
45    kp = Kp;
46    ki = Ki * SampleTimeInSec;
47    kd = Kd / SampleTimeInSec;
48    if (controllerDirection == REVERSE)
49    {
50        kp = (0 - kp);
51        ki = (0 - ki);
52        kd = (0 - kd);
53    }
54}
55
56 void SetSampleTime(int NewSampleTime)
57{
58    if (NewSampleTime > 0)
59    {
60        double ratio = (double)NewSampleTime
61          / (double)SampleTime;
62        ki *= ratio;
63        kd /= ratio;
64        SampleTime = (unsigned long)NewSampleTime;
65    }
66}
67
68 void SetOutputLimits(double Min, double Max)
69{
70    if (Min > Max) return;
71    outMin = Min;
72    outMax = Max;
73    if (Output > outMax) Output = outMax;
74    else if (Output < outMin) Output = outMin;
75    if (ITerm > outMax) ITerm = outMax;
76    else if (ITerm < outMin) ITerm = outMin;
77}
78
79 void SetMode(int Mode)
80{
81    bool newAuto = (Mode == AUTOMATIC);
82    if (newAuto != !inAuto)
83    { /* we just went from manual to auto */
84        Initialize();
85    }
```cpp
inAuto = newAuto;

void Initialize()
{
lastInput = Input;
ITerm = Output;
if(ITerm > outMax) ITerm = outMax;
else if(ITerm < outMin) ITerm = outMin;
}

void SetControllerDirection(int Direction)
{
controllerDirection = Direction;
}

void setup()
{
// put your setup code here, to run once:
kp = 1;
ki = 1;
kd = 1;
SetSampleTime(1000*60*30);
SetControllerDirection(DIRECT);
SetMode(AUTOMATIC);
}

void loop()
{
// put your main code here, to run repeatedly:
Input = 1;
Compute();
}
```