Balancing-Table

Project Documentation

ME 405

Submission Date:
03/17/19

Christopher Moranda

Hanno Müller

1. Introduction

Many modern mechatronic applications will contain a motor, at least one sensor, and a closed loop controller. This kind of system is very popular in products ranging from printers and cars to quadcopters and segways. The design and implementation of such a controlled system should be in the skillset of every engineer who works in the mechatronics environment.

Balancing-Table Conceptualization

The concept for a self balancing table can be used in a variety of applications. An electronic camera gimbal, for example, uses a closed control loop with a gyroscope sensor to hold a film-camera in a steady position and provide smooth footage. In the field of medicine, surgical robots as well as support devices for people with diseases like Parkinson's employ a similar technology.

The system designed in this lab can be used as a prototype of a balancing table for general test and evaluation purposes. The balanced table on the top is moved by two DC-motors and an IMU-sensor is used to adjust the platform to a predefined angle.

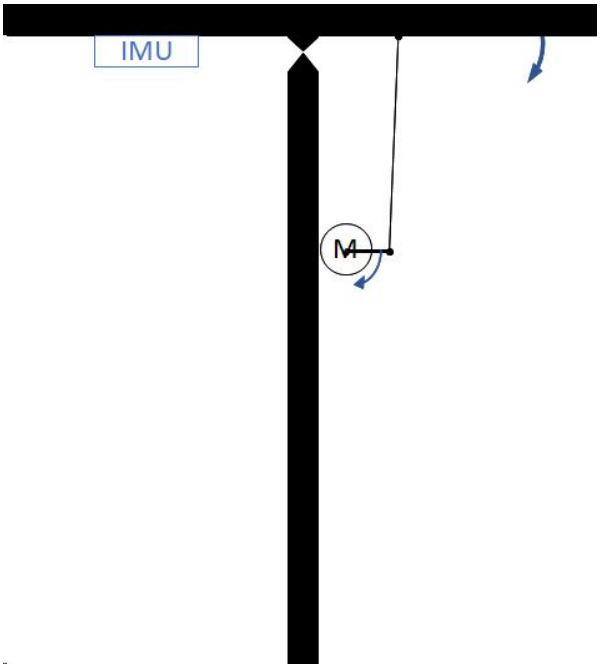

Figure 1: balancing-table-concept

As shown in Figure 1, the motor shaft is connected to the table with a lever and a conrod. Therefore, the change of the table-angle is not linear to the motor-shaft angle.

2. Specification

Due to the fact that this project was an attempt to improve a similar system designed in a previous quarter, many of the components have been reused. The following specification covers all parts to show their suitability for this project.

Actuator:

In order to move the platform to the correct angle, two motors are needed to move both axes. One motor controls the pitch and the other motor controls the roll movement. The requirements are the same for both motors because of a symmetric construction.
Two brushed DC-motors are used to control the motion of the platform. These motors are inexpensive and therefore used in many products.

Table 1: Motor requirements and specifications for chosen components

Motors	**requirement** pitch	roll	**chosen component** IG220019X00015R SHA YANG YE
Motion angle	180°	180°	360° (infinity)
Precision	<1°	<1°	-
Speed	1rad/s	1rad/s	15000rpm = 1571 rad/s
Cost	<30$	<30$	~20$ each

The selected model comes with an integrated gear. The gear has a ratio of 20:1.

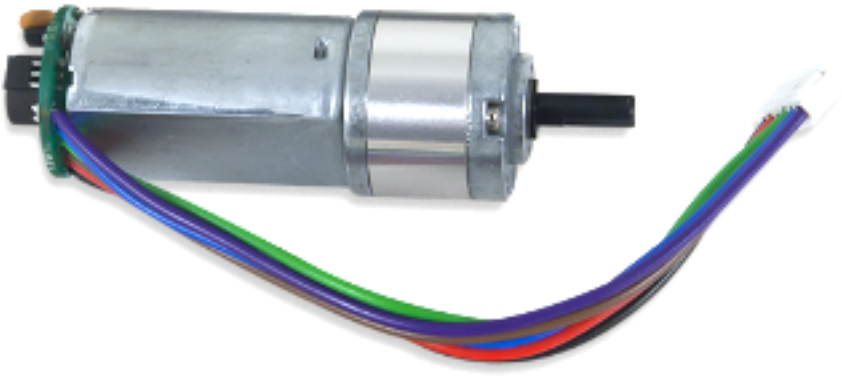

Figure 2: Selected Brushed DC Motor
https://cdn10.bigcommerce.com/s-7gavg/products/472/images/2724/Motor_Gearbox_obl_600__77128.1448061633.1280.1280.png?c=2

Sensors:

Two kinds of sensors are used in the designed system. An IMU is used to read the angle of the platform. An encoder is used in conjunction with each motor to track the motor-shaft position.

IMU:

To detect the current position of the platform, the gyroscope sensor inside of an IMU is used. The BNO055 Sensor from Bosch was chosen because of its high degree of integration. This IMU uses an integrated controller to process the sensor-fusion and to filter this pre-processed data.

Table 2: Gyroscope requirements and BNO055 specifications

gyroscope	**requirement**	**chosen component** BNO055
Frequency	<1Hz	4kHz
Resolution	0.25°	16bit/360°=0.006° (1dps)
Voltage/Power	3-5V	3.3V
Interface	I2C,SPI	I2C, UART
Cost	<40$	~25$

Encoder

The chosen DC-motors come with an integrated encoder. It is a magnetic encoder that is attached to the shaft at the back of the motor. Table 3 shows the suitability of this encoder to meet the requirements. The integrated gearbox of the motor multiplies the resolution of the encoder per revolution by 19.

Table 3: Encoder requirements and specifications for chosen device

Encoder	**requirement**	**chosen component** integrated in chosen motor
Resolution	10°	12 steps/revolution * 19 = 228➞1.6°
Voltage/Power	3.3 - 5V	3.5 - 20 V
Interface	2 signal-wires	2 signal-wires
Cost	<20$	-

Microcontroller:

Nucleo 476RB
This microcontroller is the preferred board to run Micropython on. Due to the nature of our project, the use of tasks was necessary. Python contains the yield function which makes task implementation easy. Therefore, this board was ideal to implement the system.

Motor-Driver

X-NUCLEO-IHM04A1
The motor driver chosen was compatible with our microcontroller. It can also be used to control two Brushed DC motors, which our project required.

Joint

A major improvement is the joint between handle and platform, which was changed from a ball-joint to a rotatable universal joint. The issue of the ball joint was due to the joint allowing for three degrees of freedom instead of two, resulting in unintended spinning of the platform. The new coupling shaft joint has only two degrees of freedom for the roll and pitch movement between the handle and platform. A rotational movement of the handle will be transmitted directly.

3. Design Development

3.1. Hardware Design

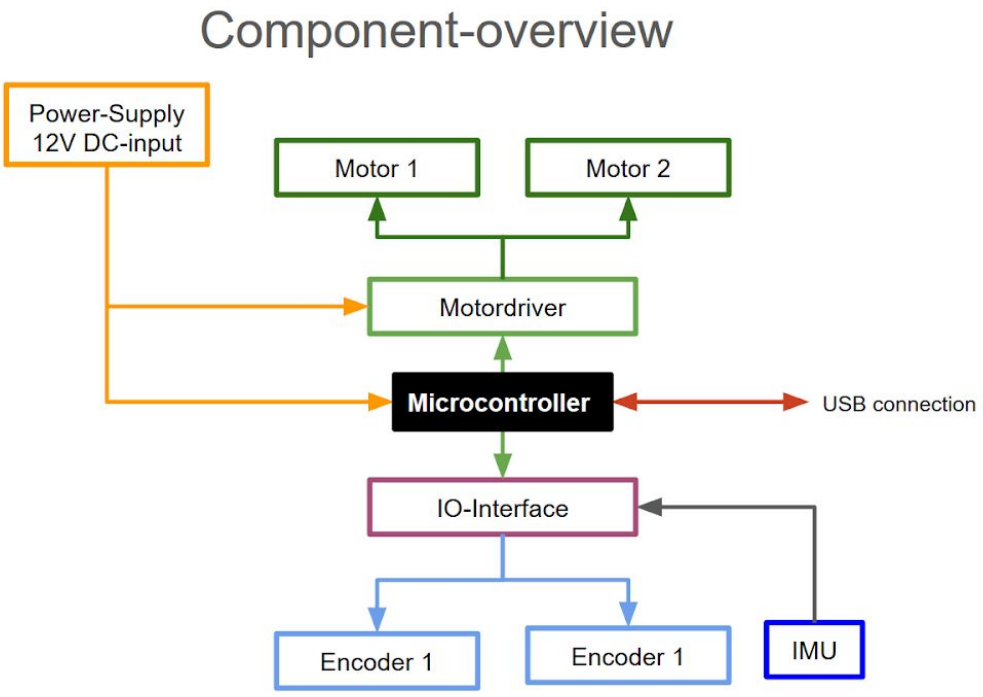

Figure 3: System Overview of Hardware Design

Electronic

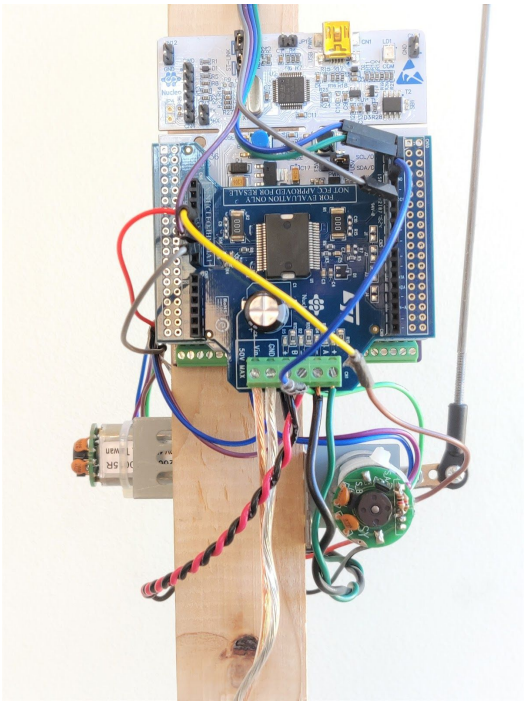

Figure 4: System Electronics

Power supply

The power supply was used to supply a voltage differential across the motor-driver. Using pulse width modulation of a 12V signal provided the torque necessary to adjust the table top with the motors.

PWM Motor-Control

The motor was controlled using pulse width modulation to provide necessary torques of the motor. The motor shield we used allowed easy switching of current direction. The H-Bridge design allowed for seamless direction switching of the motor.

I2C connection

The microcontroller reads from the BNO055 sensor using its I2C bus. The memory address used for the bus is 0x29. Using this, the microcontroller can write to a specified register on the sensor, and read data back on the bus. Most of the valuable data for our purposes was stored in two bytes in two registers. This required reading 2 bytes at once over I2C and bit shifting the most significant byte to combine with the least significant.

IO-Interface (Shoe of Brian)

By connecting our microcontroller to a shield dubbed “the Shoe of Brian”, we were able to easily access the necessary GPIO pins. The shoe allowed for the pins from our sensors to be connected to the pins via screw terminals.

Encoder

The Encoder has to be supplied with 5V from the microcontroller and a Pullup resistor has to be added to both signal outputs. The 1KΩ pullup-resistors are added to the solder contacts of each signal-contact and VCC on the encoder board.

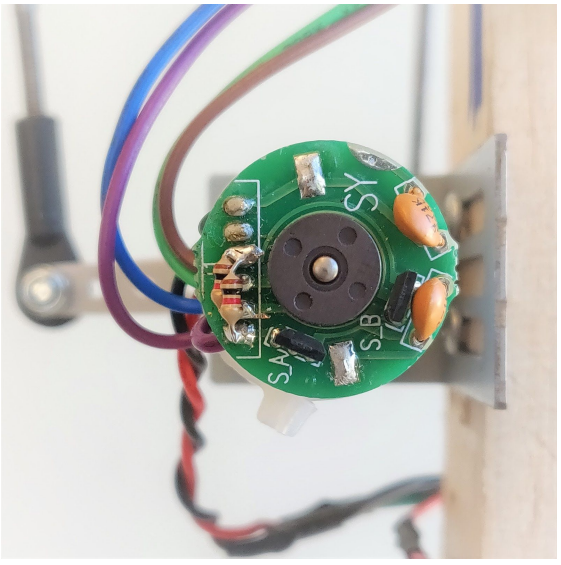

Figure 5: 1KΩ pullup resistors added to Hall Sensors

Mechanic:

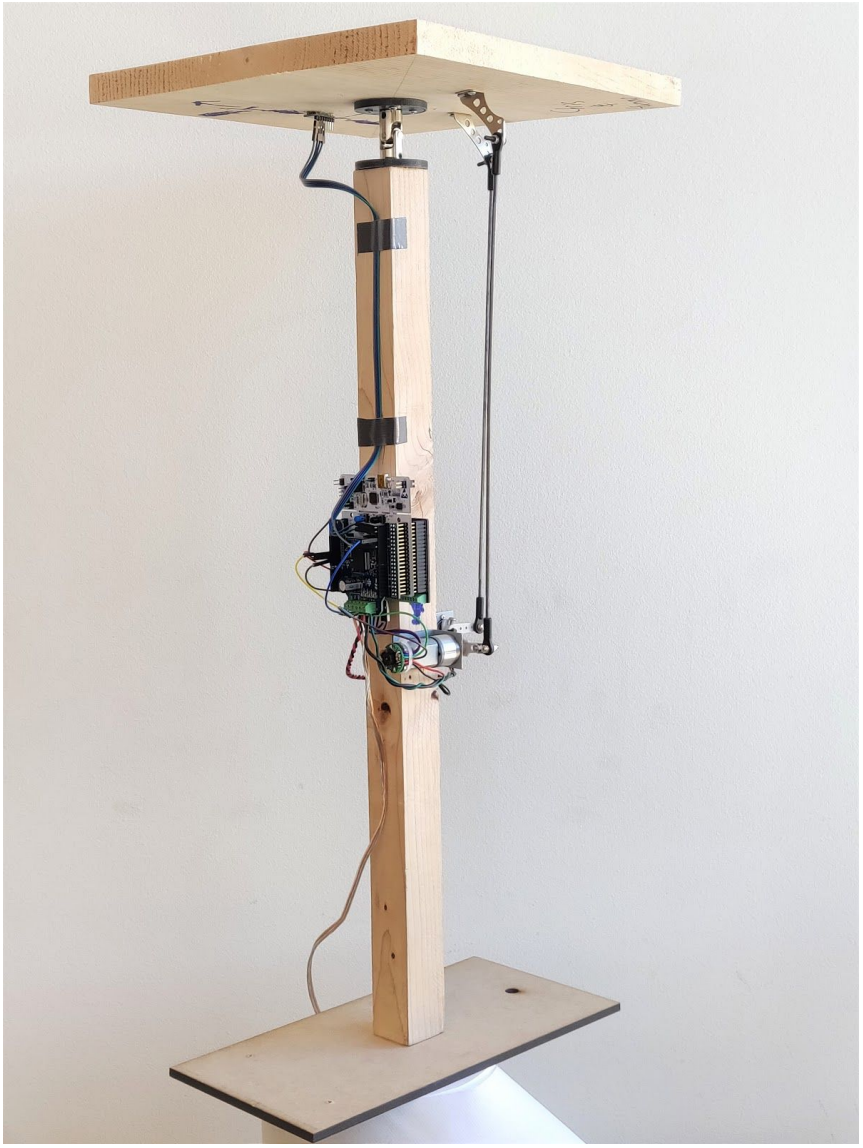

Figure 6: HC Woody in his full glory (Mechanical System Overview)

Joint/mounting

To attach the joint to the grip-handle and the platform, a 3D-model of a flange was designed and produced by 3D printing. The joint can be attached to the flange and secured by two headless screws. The flange can be connected to the handle or the platform with up to 4 screws. *Figure 7* shows the 3D-model that was printed twice, one for every side of the yoint.

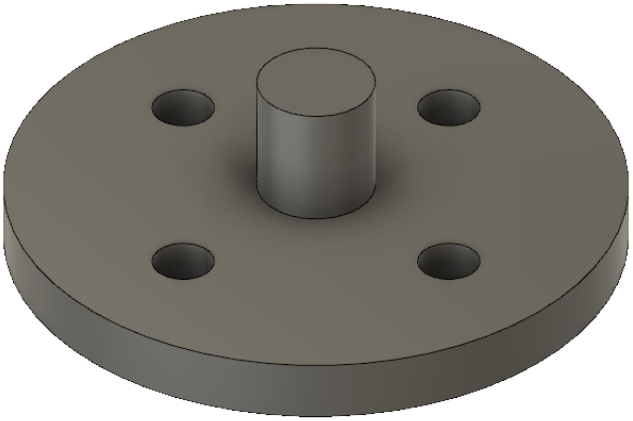

Figure 7: 3D-Model of Joint Adapter Flange

Base-Construction

The basic construction of the balancing-table-prototype contains of a hand-grip and a squared plattform. Both elements are taken over from the previous group and are built out of wood because it is easy to attach components to it and modify the shape.

3.2. Software Design

Motor

For motor control, a general motor class had been written in lab. This class allowed for initialization of two different motors using two different sets of GPIO pins. The class also contained a function for setting the pulse width of a motor, by giving an integer that was interpreted by the function as a percentage of the duty cycle.

Sensor

A specific IMU class was written for the BNO055 sensor. By reading through the data sheet, we were able to determine the registers necessary for our project. The register numbers were included as constants in our code, to make the function calls more general. Our code initialized a BNO055 object to the I2C bus on the microcontroller. The general read data function allowed for I2C data transfer from whichever register was required. Most of the data was stored in two different registers as 2 byte signed values. These values had to be placed into a 16 bit value by bit shifting the most significant byte and using a bitwise or to combine the values. The number is interpreted as a 2's complement number to account for negative values.

To take readings from the encoders, a class was written to allow for 2 encoder objects to be created and used at once, one for each motor. The encoders each have a function to set the current reading to 0, for calibration purposes. The encoder values can then be read using a hardware counter that sums the quadrature signals to keep track of current position of the motor shaft.

Tasks

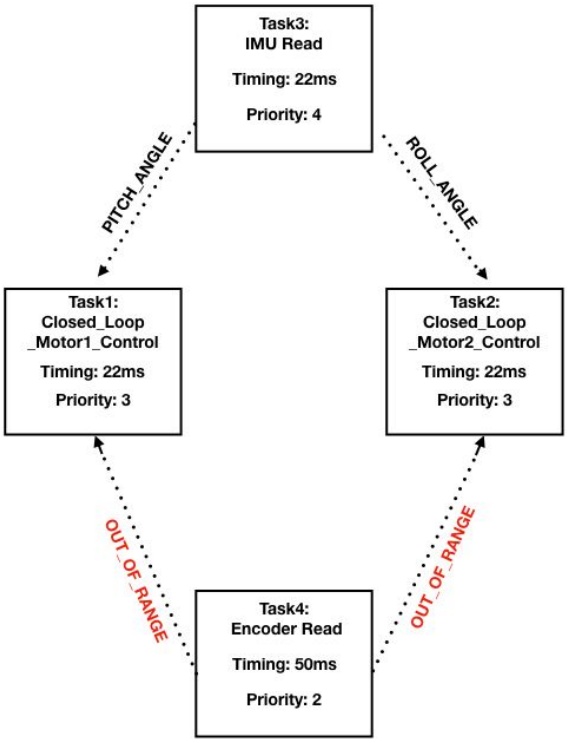

Figure 8: Task Diagram of Software System

Our task diagram for this system is relatively simple and shown in figure 8. We essentially take 4 different readings from 3 different sensors. Two of the readings are just encoder readings from each of our motors, and the other 2 readings both come from the BNO055 IMU sensor. We only use 4 tasks implemented using yield functionality in Micropython code. Task 1 controls the motor used to adjust pitch, Task 2 controls the motor used to adjust roll. Both of these tasks utilize closed loop control using both proportional and integral gain. The roll motor reads a roll angle from the IMU while the pitch motor reads a pitch angle from the same IMU. Both of these readings are done in Task 3, which takes the reading and utilizes shared variables to share with each of these motors. The motors take these angles to adjust the gain values within the closed loop control. Finally, the encoders on each motor are used to specify a physical range that the motors should spin within. This is to avoid physical damage caused by the levers on the motors from pushing against the base of the system. If the encoders pass a specified range, an error message is sent to stop the motors from spinning.

The timing and priority of the tasks was a bit counterintuitive to figure out. Normally a fail safe feature, such as the motor stop we implemented with encoders, would be high priority or even set to an interrupt to ensure a way to stop the system in an emergency case. However, because of the low danger involved in our system and the relatively slow physical movement, we prioritized responsiveness. Our cutoff point for motor range is about 10

encoder steps greater than the physical limit. Because of this, it is nearly impossible that the motor could have spun outside of the physical range before the encoder task gets called. The highest priority task in our system is the reading of the IMU, as the angles can change very quickly and the system aims to account for real time stabilization. The next highest priority, naturally, is the motor control loop. This allows us to take readings, adjust the motors, and stop the motors all in seamless action.

Control-Algorithm

In order to control the angle of the platform, two independent closed-loop controls are used. One motor is used to respond to one axis of the sensor independently. It became apparent, that a proportional-integral controller like the one shown in *figure 9* was needed, because a proportional-only-controller resulted in a steady angle error. Because of the usage of an integral-part in the control loop, an anti-windup case is added to prevent this variable from summing up infinitely. The factors for proportional and integral gain are tuned manually by increasing each until the system's reaction resulted in overshooting or unstable behaviour like oscillation. We then decreased the values a small amount to get an optimum performance.

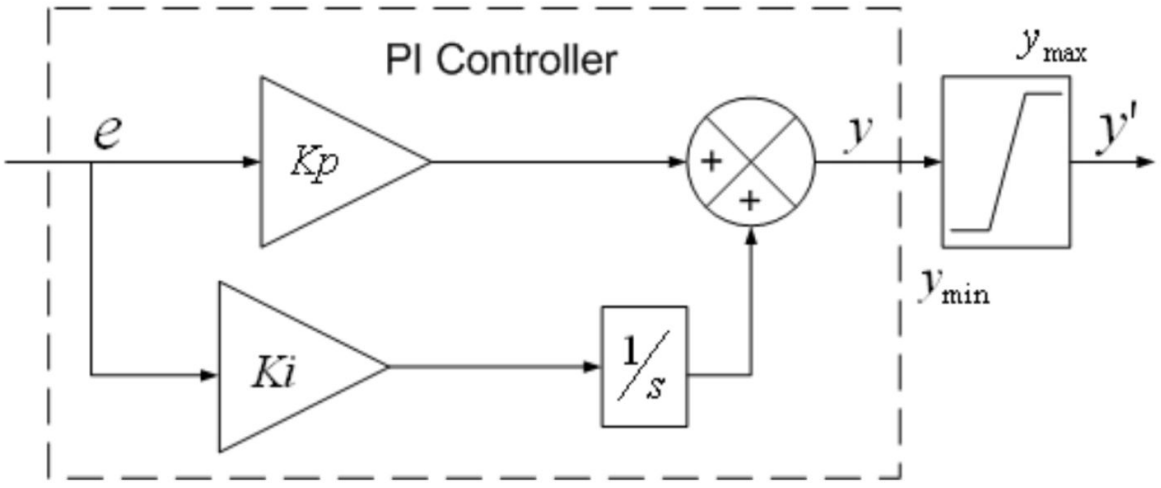

Figure 9: Diagram of a Proportional Integral Controller
https://pdfs.semanticscholar.org/9774/270fa07e4be5dbe77dc7bf2a285167b82b68.pdf

Proportional

The proportional part multiplies the difference between the setpoint and the actual platform-angle with the constant proportional gain factor Kp to generate the motor voltage output. The best result for Kp was achieved with the value 3.7 .

Integral

The integral part multiplies the error with the constant integration gain factor Ki and sums up all previous errors to generate the output voltage in sum with the result of the proportional part. This method is used to erase a steady angle error, because the output voltage will be increased the longer an error exists. The best result for Ki was achieved with the value 0.07 .

Anti-Windup

To prevent the integral part from generating a huge output value, caused for example by a mechanical disturbance that can't be corrected by the system, an anti-windup feature was added. If the variable for integration of the error reaches ∓400, this feature stops the variable from growing any larger. The values ymin = -400 and ymax = 400 were generated empirically.

Results

Overview

The system functioned very well overall. This was demonstrated through several tests that showed physical success as well as the output of satisfying data results.

Mechanics

Different features were optimized using different methods, and this led to a high performance prototype. Possible improvements are mentioned below.

Joint

The selected rotatable joint prevents the table from spinning horizontally and provides a good bearing for pitch and roll movements. The only thing that could be improved is the bearing-clearance of the joint that allows a rotation of the table in the horizontal direction to a certain degree. With a higher quality joint, this disadvantage can be fixed easily.

Limitation of angle correction

The angle-correction between handle and table functions up to 40 encoder steps in every direction. This is due to a relatively small lever that is attached to the motor-shaft. A longer lever would increase the maximum angle that can be corrected, but would also increase the requirements for the motor related to a higher torque and a higher resolution. This could be achieved with a higher gearbox ratio for the same motor but it would decrease the maximum speed of the output-shaft, which has to be checked to still be efficient.

Electronics

Electronics have been primarily assembled with pre-built components. As soon as debugging and optimizing is done, power can be supplied only from a single power source such as a 3S Lithium-Ion battery to make the system portable.

IMU

The IMU we used had some problems with angle values drifting, even when the system was not moving. The readings may have been affected by a possible faulty sensor or a software error. We programmed the IMU driver from scratch so it is possible we missed a step in the setup procedure. Trying another sensor or debugging the code could solve the drift problem.

Motor

The motor was sufficient for our system. The torques provided were mechanically capable of controlling the table top and the resolution of the encoders was adequate. If we change the lever to account for physical limitations of the system, we would need a motor with a higher gearbox ratio to achieve a higher torque.

Control System

Step-Response

A common way to test the performance of a system is to graph the system response of a setpoint-change. The closed-loop-system will change the output-parameter to match the setpoint. Therefore we changed the setpoint once during runtime and recorded the system response. We changed the Setpoint of the pitch axis from 0° to 10°.

Figure 10: Step response of pitch axis

The result of this step-response is shown in figure 10. After the setpoint changed from 0° to 10° the motor-voltage jumps to a high level to turn the shaft and change the table-angle. The encoder-steps change parallel to the platform angle but result in a stepped signal due to the low resolution.

Motor Angle Limit

To demonstrate the motor-shaft angle limit, the handle was tilted until the maximum value of 40 encoder steps from neutral position was reached on one axis. After the motor stopped, the handle was tilted backwards until the motor was enabled again and the angle-compensation restarts.

Figure 11: Demonstration of Motor Limit

As soon an the encoder-signal reached the limit of -40 steps, the motor was disabled. This resulted in a steady encoder signal and a disabled angle compensation showed by a fast changing angle. When the handle is turned to the opposite direction and an angle of -10° was reached, the control-loop started working again and the platform was leveled again.

4. Appendices

Doxygen available on:

Mercurial: http://wind.calpoly.edu/hg/mecha02

Github: https://github.com/Cmoranda1/ME405
https://github.com/Cmoranda1/ME405/tree/master/ME-405_Final_Project

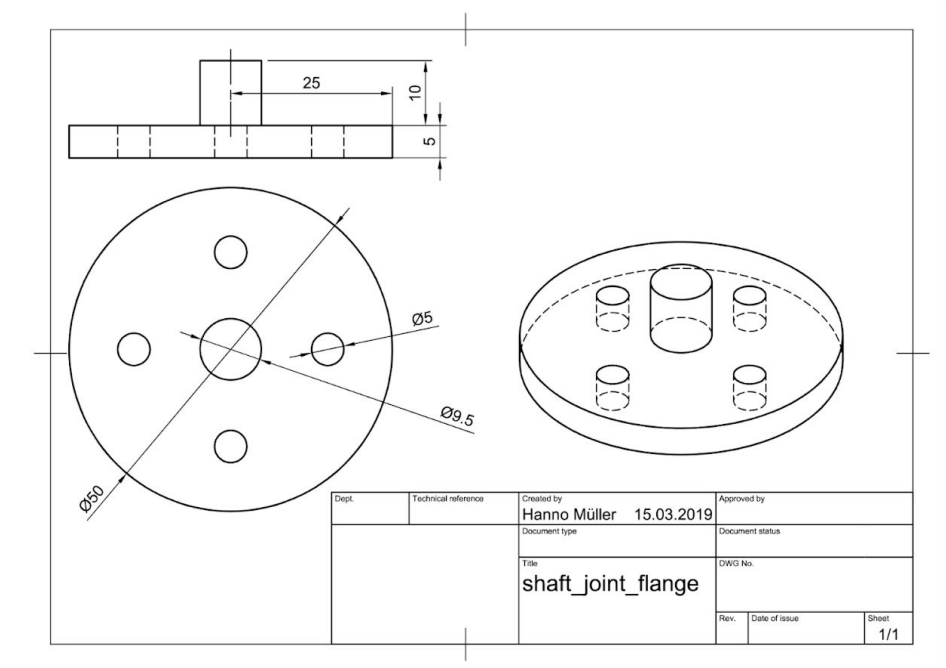

Figure 12: CAD drawing of Flange

5. References

BNO055 datasheet
https://ae-bst.resource.bosch.com/media/_tech/media/datasheets/BST-BNO055-DS000.pdf

Encoder datasheet
https://reference.digilentinc.com/_media/motor_gearbox/magnetic-encoders.pdf

Motor datasheet
https://reference.digilentinc.com/_media/motor_gearbox/290-006_ig220019x00015r_ds.pdf

Shoe-of-Brian reference
http://wind.calpoly.edu/ME405/doc/index.html