Aerial Manipulator Load Estimation

Tom Schmitz, *Student, UL,* Jan Eric Dentler, *UL,* and Dr.-Ing.Holger Voos, *Professor, UL*

***Abstract*—The recent development of transportation applications for Unmanned Aerial Vehicles (UAV) is facing the problem of moving unknown payloads. This becomes particularly challenging in aerial manipulator scenarios, where an unknown load is manipulated with a robotic arm which is attached to a UAV. The challenge tackled in this paper is the estimation of the UAV payload and its position with respect to the drone parameters. More precisely we will assess a specific setup of a drone carrying a two joint robotic manipulator equipped with a gripper as end-effector.**

***Index Terms*—Drone, Paper, Manipulator, Load, mass, Inertia, UAV.**

I. Introduction

MACHINES have been around to take over tasks which are not or hardly possible for humans. With each step up in technology machines were created that opened the doors to new possibilities. The latest big achievement being autonomous vehicles, UAVs in particular have seen their applications explode through the last years. The latest idea being an implementation in transportation of goods. The big change this Idea introduces, compared to simpler tasks like observation and aerial photography is that the payload is unknown and could take up a substantial amount of the overall weight of the UAV.

This project is a based on the work of Patrick Hoffmann, ”Development of a controller for lightweight manipulator” [2]. Mr. Hoffmann successfully planned and implemented the robotic arm which is used as aerial manipulator for all the manipulations of this project. This work focuses on estimating the weight and position of the payload gripped by this manipulator.

Quad-copters have 4 rotors mounted symmetrically around the body. The controller has the target to stabilize the UAV while flying and interpret the commands received like ”fly forward” by reacting and accelerating the necessary motors in order to tilt the Drone towards the desired direction of movement.
Since the drones are not always symmetric any asymmetry and change in the drone inertia e.g. by payload changes the way the drone behaves and needs to be compensated by the controller. If a change in load distribution is not compensated for, the Drone can experience a constant motion disturbance in form of change in movement direction relative to the desired direction. This can be prevented if the payload mass and position are known.

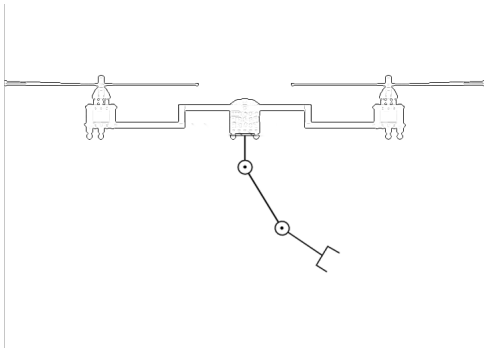

Fig. 1. Side-view of a drone carrying our robot manipulator.[1]

Figure 1 shows very the utilized aerial manipulator mounted on a quad-copter Drone.

II. Setup

In this section the experimental setup, control resources and tools will be described and explained.

A. Hardware

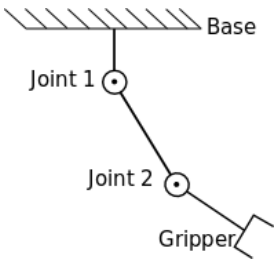

Fig. 2. Staticly mounted robotic arm.

Figure 2 shows the robotic arm manipulator, fixed on a rigid horizontal platform, here, the drone mounting point. The manipulator consists of two joints and a gripper at the end. The two joints are operated by Dynamixel servo motors. The first joint being a type MX28 while the second joint and the gripper feature an AX18A model. Due to its position under the drone, the operating space has to be limited to the points situated below the rotors and withing safe clearance of any other appliances featured by the specific UAV. This clearance must be defined for each scenario and setup. Figure 3 shows the workspace described above for a scenario where no other appliances are mounted below the UAV. The algorithms elaborated in this paper although apply to all possible orientations of the robotic arm.

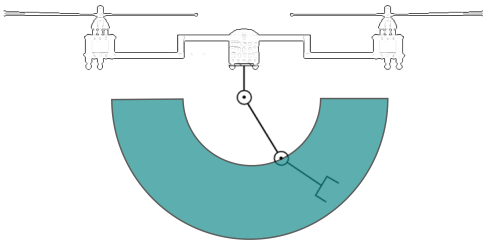

Fig. 3. Workspace of the manipulator.

B. Software

The UAV as well as our robotic arm are controlled via ROS(Robot operating system) [3]. The core of ROS runs under Linux either on a remote pc or on the drone computing unit itself. In ROS executable files are called ”Nodes” and programmed in C++. A node can for example have the functionality to control the drone position or function as a driver to our component, the two-joint Dynamixel manipulator. Communication between ROS and the hardware is done is implemented via channels which are called ”Topics”. If a Node wants to provide data to another one it will write or ”publish” this data stream through a ROS::Publisher into a Topic. In order to receive information, a reader called ROS::Subscriber will listen to the data stream published in a topic. Each topic can hold different data. One topic used extensively in order to gain real-time knowledge about the current situation is ”motor_states”. It contains all the data acquired and published by the Dynamixel motors/drivers like their current joint position, load or time-stamp.

III. METHODS

This section will bring the reader the analytic view to the main problem closer. Since the UAV behavior will depend on the position and mass of the payload, our most important variables to be considered are the angles of the manipulator joints and the torque applied by the Dynamixel servos to reach the goal orientation. Since UAVs rely on models to adapt their flight behavior, the weight and position of the payload can be used to adapt and update the inertia model of the UAV as precise as possible. Based on the new model the UAV can be stabilized during operation. The key here is to process the data in real-time and commit to the balancing of the UAV as possible. The faster and more accurate the estimation of the payload parameters, the more stable the drone will be during operation. The first step is to determine the mass and next try to derive the inertial parameters of the payload such as its center of gravity. The center of gravity of the payload would be important especially when lifting larger objects which feature a center of gravity further away from the gripper.

A. Schematic view

To better understand the manipulator and its potential Figure 4 shows the fixed parameters L1 and L2, denoting the lengths

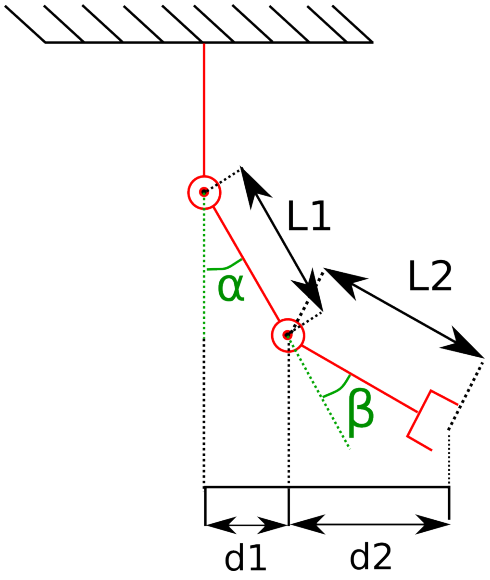

Fig. 4. annotation of the geometrical parameters.

of the actuator ligaments. The influencing factors here are the angles alpha and beta which determine d1 and d2. The latter express the horizontal distance of the joint 2 and payload from the center of gravity of the drone and will be essential later on. The more eccentric the payload is applied from the mounting point horizontally, the higher the disturbance to the system.

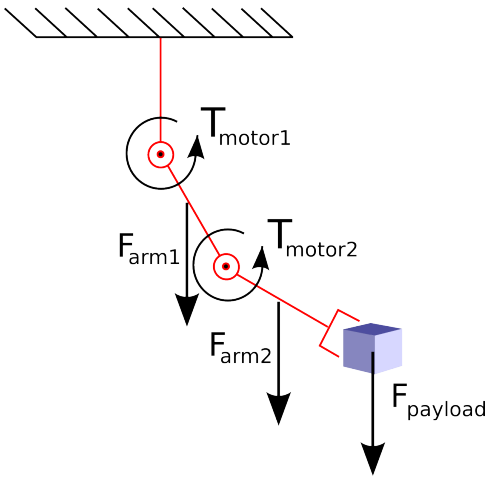

Fig. 5. Force analysis of the elements.

The way to access the mass of the payload will be through the torque applied at the joint actuators of the aerial manipulator. Those correspond to the load parameters from the motor_states topic in ROS. The load is expressed in % of torque applied relative to the maximum stall torque. The stall torque is dependent on the voltage on the motor which is more or less 12 Volts constantly in our case. The applied torque can be read from motor1 and motor2 which gives two sources of data and can be used for better approximation. Contributing to the torque on the motors are the weight of the components of the robotic arm and the payload. To recap,

the torque is the cross-product of a force and its distance to the center of the joint in question which is denoted as "lever" here depicted in equation (1). Practically this cross-product will be implemented with the knowledge of the angle θbetween the force and the arm as shown in equation (2).

$$Torque = lever \times force \tag{1}$$

$$Torque = ||lever|| \cdot ||force|| \cdot sin(\theta) \tag{2}$$

The following equations (3) and (4) based on Figure 5 show how the torques exerted on Joint1 and Joint2 are composed in the given system. For simplification of the algorithms the distance from the gripper to the payload center of gravity, $L_{payload}$ is assumed to be negligible and therefore assumed zero in the following applications.

$$\begin{aligned} T_{Motor1} =& F_{payload} \times L_{arm1,arm2,payload} \\ &+ F_{arm1} \times L_{arm1} \\ &+ F_{arm2} \times L_{arm1,arm2} \end{aligned} \tag{3}$$

$$\begin{aligned} T_{Motor2} =& F_{payload} \times L_{arm2,payload} \\ &+ F_{arm2} \times L_{arm2} \end{aligned} \tag{4}$$

where

$$F = force, T = torque, L = lever.$$

$$\vec{L}_{arm1,arm2,payload} = \vec{L}_{arm1} + \vec{L}_{arm2} + \vec{L}_{payload}$$

$$\vec{L}_{arm2,payload} = \vec{L}_{arm2} + \vec{L}_{payload}$$

By solving the equations (3) and (4) to $F_{payload}$ and valid theoretical formulation is obtained for the force exerted by a payload applied closely to the end effector. $\vec{L}_{payload}$ being assumed $\vec{0}$, appears no longer in the following formulation. One more assumption which is acceptable for explaining the principle here is that the center of gravity is located at the end of each of arm. Since the main weight is emerging from the servo-motors, which are mounted at the joints, this is a realistic assumption but will be slightly corrected to achieve a higher quality of the results in the algorithms. Equations (5) and(6) show the final theoretical solution for the payload calculation.

$$F_{payload} = -\frac{F_{arm1} \times L_{arm1} + F_{arm2} \times L_{arm1,arm2} - T_{Motor1}}{L_{arm1,arm2}} \tag{5}$$

$$F_{payload} = \frac{-F_{arm2} \times L_{arm2} + T_{Motor2}}{L_{arm2}} \tag{6}$$

Last the weight of the payload can be calculated by dividing $F_{payload}$ by the acceleration of earth which is 9.81 m/s. All further optimizations of the and correction factors are introduced and taken account for in the numerical part which follows.

B. numerical view

The code snipped presented below in Listing 1 covers the core of the node which estimates the weight of the payload. As mentioned before first the torque solely exerted by the manipulator components is estimated (lines 38-39). Those equations relate strongly to equations (3) and (4) but differ by the fact that there has been introduced two corrective factors which are meant to better evaluate the position of the center of gravity for the servos (which are part of each arm) relative to the total length of the arm. So the center of gravity of the $arm1 = lever1 * 0.9$ while it is for $arm2 = lever2 * 0.5$. This deducted from the measured torque reveals the torque exerted by the payload (lines 46-47). Having two values for the payload mass is very helpful in order to converge faster to plausible values and double check for possible erroneous measurements. This provides for example the possibility to ignore spins(=iterations) where the masses determined from joint 1 and joint 2 defer too heavily. For though the algorithm simply calculates their average value. This unconditional feedback is useful to better understand how the servos react in different situations and to get more simple output for experiment analysis.

Due to heavy variations within the measured torque during initial tests a running average is introduced with the motivation to smoothen out possible fluctuations of the dynamixels controller. This behavior is expectable since the servos use position-controlled PID controllers which constantly correct the applied torque in order to achieve the goal-positions .

The angular positions of the joints can be read from the ROS::publishers of motors 1 and 2. Since the MX28 and AX18A position values are 10bit encoded and describe the currently internally targeted servo-step this value needs to be converted to radians (lines 24-25). The straight downwards pointing position is declared as POS(0,0) for the Joint space. This actuator position is to be avoided during operation since there is no torque to be applied in the static mounting scenario shown in Figure 2.

```
1  //getting status of the motors
2  void angleCallback(const dynamixel_msgs::
       MotorStateList::ConstPtr& msg)
3  {
4   //physics constants
5   double gravity = 9.81 //[m/s  ]
6
7      //setup
8   dynamixel_msgs::MotorState motor1state=msg->
       motor_states[0];//loading stats of first motor
9   dynamixel_msgs::MotorState motor2state=msg->
       motor_states[1];//loading stats of second
       motor
10  dynamixel_msgs::MotorState motor3state=msg->
       motor_states[2];//loading ststs of the gripper
        motor
11
12  // possible input from UAV controller:
13  double droneAngle = 0; //[  ]inclination of the
       drone relative to the gravitational direction.
14  double droneVertAcc = 0; //[m/s  ] vertical
```

```
    acceleration of the drone;

double weightDynamixel = 0.06*(droneVertAcc+
    gravity); //54.5g -> estimate to 60g with
    accessories
double L1 = 0.0733; // length of arm Join1 -
    Join2
double L2 = 0.1055; // length of arm Join2 -
    gripper
double maxTorque_MX28 = 2.5; // stall torque, max
     torque at 12V
double maxTorque_AX18A = 1.8; // idem for AX18A
    motor


//calculate the real joint agles from the
    motor_states.positon
double alpha = droneAngle+(motor1state.position
    -2048)*0.088;
double beta = droneAngle+(motor2state.position
    -512)*0.29;


//real positions relative to base (for cross
    product)
double lever1 = L1*sin((alpha)*PI/180); //lever
    of L1 depending on alpha
double lever2 = L2*sin((alpha+beta)*PI/180); //
    lever of L2 depending on beta
double lever12=lever1+lever2;//total lever:
    joint1 to gripper

//transform the maxLoad/measuredLoad back to real
     Torque in Nm
double T1= motor1state.load*maxTorque_MX28;
double T2= motor2state.load*maxTorque_AX18A;

//torque produced by the components without
    payload
double noloadT1= (lever1*0.9+lever12)*
    weightDynamixel; //gravity point at 90% of
    lever length
double noloadT2 = (lever2*0.5)*weightDynamixel;//
    gravity point at 50% of lever length

//calculate the mass from current value
double corrMX28=2.4; //compensate loss in gears
double corrAX18=1.65;// and internal motor
    resistances.

//calculate the mass from current value
double massJ1=(T1+noloadT1)/(lever12)/-9.81*
    corrMX28;
double massJ2=(T2+noloadT2)/(lever2)/-9.81*
    corrAX18;
double Mass=(massfromM1+massfromM2 )/2;

// calculate the running average over the 20 past
     values
double avgMass20=(avgMass20*20+Mass)/21;

//write the real-time payload weight into a ros::
    publisher topic
publishtemp.data=avgMass20;
avgMass20_pub.publish(publishtemp);

}
```

Listing 1. main algorythm

Lines 42-43 introduce the corrective factors corrAX18 and corrMX28. Those are meant to compensate for any loss in the gearing and other mechanical elements of the motors or internal losses of the motors. Those two parameters are experimentally determined and can hardly be determined without manufacturer knowledge of the servo motors. In order to adequately estimate those, different scenarios have been tested and the corrective values adapted independently until the results matched the expectation. For the usage in this project, a linear mapping of the torque losses for the joint actuators showed to be sufficient since the errors detected during the setup tests showed to be mostly proportional to the applied payload. The final values for the torque correction were estimated to x2.4 for the MX28 servo and x1.65 for the AX18A servo

C. getting more out of it: center of gravity

Now that the weight of the payload has been estimated when it is held closely within the gripper, the next step is to try to find a good estimate for the center of gravity of the object. This seems to be a hard problem but just from looking at the resources and the previous equations (3) and (4) used to determine the mass of the payload there is a clear opportunity. Since there are 2 inputs(loads on each motor) in use to find 1 unknown (payload mass), there is still room to determine a second unknown. This leaves room to find one additionally introduced parameter, $L_{payload}$ if the equations are solved adequately.

Ignore the weight of the motors and gripper itself there would be a possibility to easily solve this set of equations and implement it which is definitely possible.
The results then solve like this:
From the equations of T_{Motor1} and T_{Motor2} (3) and (4):

$$F_{payload} = (T_{Motor1} - T_{Motor2})/L_{arm1}$$

and finally

$$L_{payload} = (T_{Motor2}/F_{Payload}) - L_{arm2}$$

The second unknown of $L_{payload}$ can be calculated from this system fo equations. The second unknown, $F_{payload}$ of course staying as determined earlier.

The weight of F_{motor2} and $F_{gripper}$ is neglected since this would lead to increasingly complicated equations and get out of the scope of this project which deals with the detection of the mass of the payload only.

IV. RESULTS

The firsts tests which are not evaluated qualitatively here were done to determine the corrective parameters corrMX28 and corrAX18 mentioned earlier in IV.B.
The next series of tests deals with the accuracy of the methods and a final test run simulates the manipulator on a moving drone.

A. series: Overall method accuracy

This series of tests is designed to provide reliable data for realistic loading cases of the manipulator and determine the reliability and error of the presented methods.

1) experimental setup: The Robotic arm as well as the drone could take any angular value but for the following experiments the orientation of the arm and gripper is limited to three exemplar cases which are rated the most natural while gripping and holding an Object in flight. Each test series starts from an empty manipulator which is gradually loaded with three objects of increasing weight. The joint angles for the three manipulator orientations are:

$$POS(Joint1[^\circ], Joint2[^\circ])$$
$$1.POS(0, 90),$$
$$2.POS(45, 45),$$
$$3.POS(90, 0).$$

The weights of the objects are 138g, 273g and 360g. Object number one is a screw, object number two a cylindric metal rod and object number three a steel plate. All three objects are comparable in size and have their center of gravity vertically in-line with the gripper.

2) experimental outcome: For illustrative purposes the graph in Figure 6 is shown, representing the first test case only since the 2nd and 3rd are very similar in shape and do not provide any closer insight. This plotted data is in fact composed of a value for the direct mass and one for the running average mass over the 10, 20 and 30 past iterations. In order to better see the difference, the running average of 20 is omitted but was conducted and recorded.
As for the further statistical processing of this data the direct mass and the running average over 20 iterations, which proved to be the most responsive and yet reliable, will be taken into account.

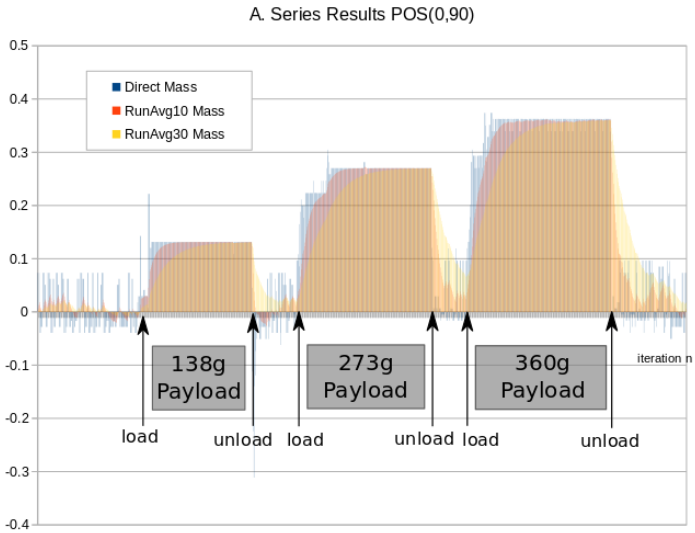

Fig. 6. Graphical results of test series A0-90.

In Figure 6 one can see slightly the behavior of the PID controller which can be observed in BLUE, causing an overshoot of the ”direct mass” value when the arm is loaded initially with an object. Also the fluctuations observed in no-load operation are quite significant.

For each of the three scenarios mentioned earlier the absolute average error and average relative error to the was determined, as well as the standard deviation of the measured values.

Object [Kg]	no-load	0.138	0.273	0.360
RunAvg20	0.0059	0.1308	0.2681	0.3557
RelErrRA20	n/a	0.0528	0.0179	0.0120
StdDevAD	0.0472	0.0019	0.0006	0.0080
StdDevRA20	0.0096	0.0006	0.0125	0.0015

TABLE I
STATISTICAL RESULTS OF TEST SERIES A0-90.
RUNAVG20: RUNNING AVERAGE OVER LAST 20 VALUES
RELERRRA20: RELATIVE ERROR OF THE RUNNING AVERAGE TO THE REAL VALUE
STDDEVAD: STANDARD DEVIATION OF THE DIRECT AVERAGE RANGE CONCERNED
STDDEVRA20: STANDARD DEVIATION OF THE RUNNING AVERAGE VALUES AFTER SETTLEMENT.

Again as the values from the other two test cases of Series A are very similar the outcome of the first test case will be considered here to base our assumptions and conclusions on. What jumps to the eye first when looking at Table I is the standard deviation which drops drastically when comparing the measurements for the direct mass and the one that comes out of the running average of 20. This shows us that the method of averaging out the fluctuations works indeed and gives for a more reliable value at any given time. The only drawback from this method is the need to wait 20 iterations before the value has settled to a new payload. At the current loop rate of 50hz this corresponds to about half a second which is acceptable. The settling time can be reduced up to 0.1 second if loop rate was increased to 100hz and a running average of only 10 values was used. The average of 10 is also reliable enough to make a first approximation or detect that the mass of the payload has indeed changed and does not emerge from a PID controller fluctuation.
Something worth pointing out here is also the relative error of the running average which is at 5.3 % for the lowest measured object. Interesting also here is to see that the relative error decreases with increasing payload weight. The maximum load of the manipulator is about 400-450 grams which is sufficient. From examining the relative error trend it can be said that the measurements will stay usable until the limit of the arm is reached. When going towards smaller loads difficulties in maintaining precision could be encountered due to the increasing relative error. This specific case will be dealt with in the experimental series C.

B. series: Accuracy during movement

The second series of tests was performed with the goal to simulate the behavior of the mass calculation while the arm is mounted on a moving UAV. The experimental oscillating acceleration is induced manually while controlled with a timer. The excitation values of 2Hz and 0.5Hz are just guidelines and can therefore slightly vary. As the latter are meant to represent a disturbance in movement this is accepted.

1) experimental setup: There were 2 scenarios, one where the UAV is accelerated horizontally in the plane of the arm joint DOFs and a second moving the UAV vertically in the direction of the gravitational force of earth. Each scenario is performed at joint position POS(45,45) and with the payload mass of 273 grammes. Both scenarios have three stages depicted on Figure 7 and Figure 8.

First stage: waiting for the mass to settle at running average of 20.
Second stage: slow ocillating movement along axis 10 cm at 0.5Hz freq
Third stage: fast ocillating movement along axis 10 cm at 2Hz freq

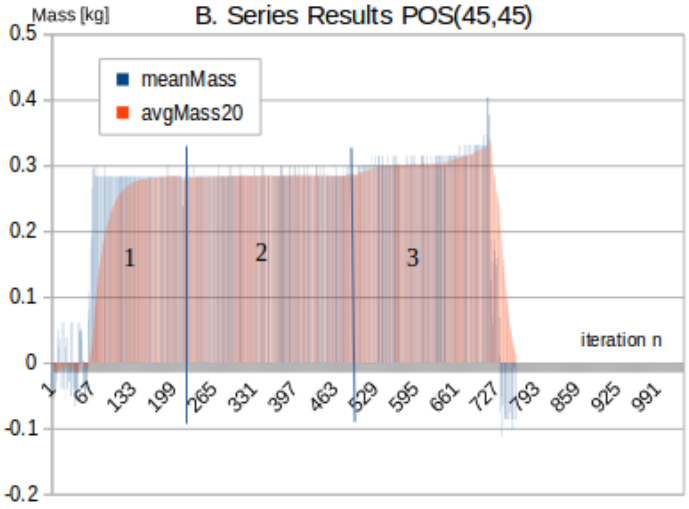

Fig. 7. B Series Horizontal movement

2) experimental outcome: Since during the first stage there is no movement at all, behavior is as expected. During the second stage some measurement spikes of 0.3kg arise within the direct mass values which can be seen in Figure 7. Those arise due to the movement and are compensated effectively by the running average. Passing on to the third stage with the faster movement and thus faster acceleration a stepped behavior is observed for the mass which does not recover again when moved in the opposite direction. This can be explained by the servo motors internal controller as well as the losses within the joint. What the PID controller is trying to accomplish is keeping the goal angular position. While the force on the motors grows due to the external acceleration, the controller increases the applied torque and keeps a slightly erroneous position and therefore falsely outputs a higher mass value.

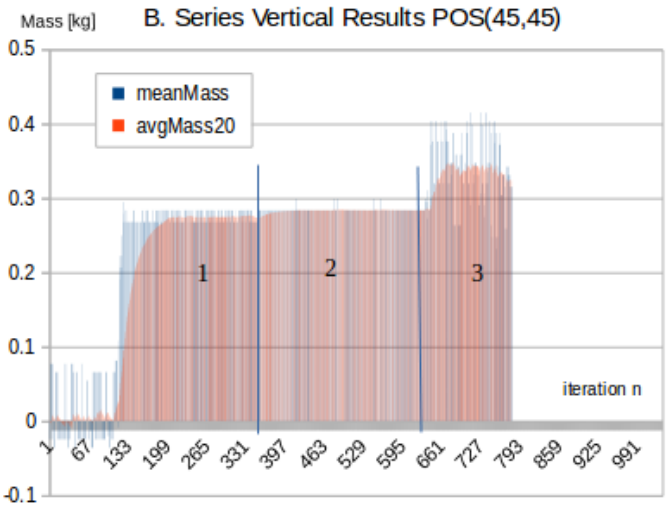

Fig. 8. B Series Vertical movement

The results shown in Figure 8 represent a similar behavior to the previous scenario. This time the movement/acceleration is applied vertically and the reason why the error in stage 3 grows so large is because of the accumulated gravity of earth. Each period of movement the robot manipulator is further loaded without time to realign to the goal position. The load of the payload plus the positive acceleration is not correctly compensated by the servos in time. Surprisingly though second stage 2 seems to be mostly un-affected by the movement fluctuations. This leads to conclude that for such scenarios there is no need to account for extra accelerations even if one is are aware or in control of such. The servo-controller reacts mostly on impacts, and does accept a certain false-position of the joints. When reaching the point of reverse acceleration the motor torque does not recover completely since for this to happen a larger amount of the load would need to be reversed. Here the losses in the gearing of the joints play a substantial role. The error is cumulated throughout each movement and will eventually reach a maximum value. To regain a correct measurement value the payload needs to be released and re-applied ultimately. An other possibility could be to slightly move the robotic arm up and down to the desired position again in order to regain the normal measurement state.

C. series: Low Mass Payload experiment

This series is the last of this project and was performed to assess the potential to determine low-weight payloads. In order to be able to correctly implement this method of load estimation, its limitations must be absolutely clear and defined. This chapter tries to elaborate the limitations of the presented methods.

1) experimental setup: This test is done at stand-still to guarantee the measurements are as clear as possible. As before the join orientation was POS(45,45). The applied weight was 36.3 Grammes only which is about a third of the previously smallest value tested for. The reason why smaller specimen have not been tested yet is because of the low their impact

to the larger UAVs in question. Light payloads usually don't feature an important inertia so the lower bound detection limit was set to 100 g previously.

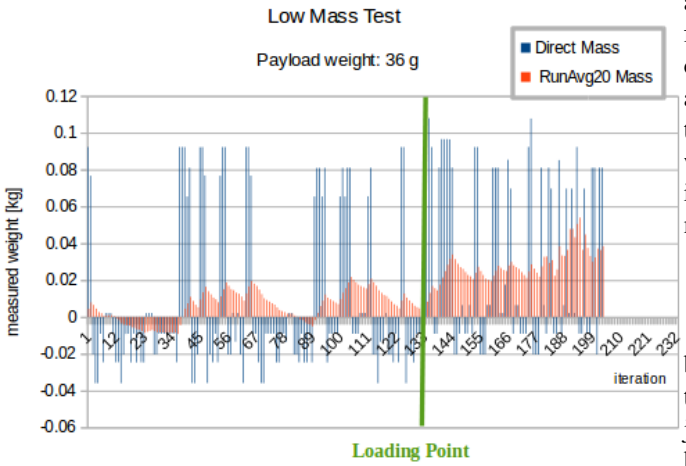

Fig. 9. B Series Vertical movement

2) experimental outcome: Figure 9 shows the no-load state in comparison to the state after the green line which denotes the section while the manipulator carries 36 g of payload. Even tough the weight values are fluctuating between -0.01 Kg and +0.02 Kg before the application of the load, a steady value increase clearly passing above 0.02 Kg can be noticed upon loading. The total average of the values before the loading point (at no-load) is 0.0098 Kg while the standard deviation is 0.01 which is satisfying. The total average of the values after the loading is 0.0273 Kg which again can keep up with the small but constant error experienced during the medium mass experiments. Although when looking at the the high relative error which is now at 25 % for the running average value this is at the limit of which is acceptable for load estimation and for smaller UAV such a tolerance could become problematic.

D. further implementation

Left open for discussion and research is yet the idea to further advance by determining the center of gravity and inertia of the payload in question and transfer this information to the UAV model. The algorithms provided in this paper are meant to be simple and lightweight to implement while delivering reliable results usable in UAV control for the future. One suggestion is to further analyze the behavior of the reaction to external accelerations emerging from a moving drone and the manipulator-UAV implementation combined with the control algorithms developed in an earlier thesis by Patrick Hoffmann. A point to definitely have a look into if further developed would be to better determine the no-load times since there are some spikes there that an be filtered out from the set if done with care.

V. Conclusion

This Project has been a good step into the right direction while providing the essentials required to estimate a load applied to the given aerial manipulator. A plus is definitely the modular code design which gives many opportunities to integrate further knowledge or adapt the corrective factors to the needs of the Tasks. It has shown in the experiments that adjusting the two previously introduced "correcting factors" for the servos leads to a shift in the reliable operating range of the estimation algorithm. As is, the parameters are set to achieve a maximum span of reliable values from 0.1 kg up to 0.4 kg and less precise but still generally usable values for weights going down to 0.05kg. Which is still a topic of interest is how to take the most advantage out of the two independent measuring points for the torque.

VI. Discussion

Servo-motors which are torque controlled are mostly black-box systems and the applied torque does not map directly the compensated torque due to the numerous losses in the joints and control algorithms. One idea here which could yield huge success is to use force/pressure sensors additionally to determine the exact torque applied. Those could be as interface between the actuator and the UAV while featuring only a minimal movement in order to not disturb the interaction of the aerial manipulator with its environment.

References

[1] $https : //www.simpleplanes.com/a/0C9513/Huge - Quadcopter$
[2] Patrick Hoffmann, "Development of a controller for lightweight manipulator", Bachelors Thesis, University of Luxembourg 2016.
[3] ROS web-page: www.ros.org/ .