Torque Control for Schunk Modules

Tobias Kunz

***Abstract*— This paper describes the steps taken this semester towards a torque controller for Schunk modules. A large part of the work consists of identifying the motor variables and estimating the current based on the measurable motor variables. We show results of a controller that is able to achieve a desired current. In connection with the presented results on the current-torque relationship, this yields a torque controller.**

I. Introduction

In order to be able to implement a more advanced balancing controller on Golem Krang, e.g. the Stilman controller, we need to be able to apply a given torque to the Schunk modules. Enabling us to do that is the goal of the work described in this paper.

There are several steps we need to achieve in order to be able to to control the torque. First, we need to find out which variables we can read from the motors and which ones we can govern. This is covered in section II. As described in section II, current cannot be measured directly. So, we have to estimate it based on other variables. This is described in section III. Building on top of this, we design a current controller in section IV. As our final goal is to govern torque and not current, we need to find out the relationship between current and torque. This is done in section V. Section VI discusses the relaxation of motor limits, which is necessary to achieve higher torques.

II. Variable Identification

Before we can do anything else, we need to find out which information we can read out from the modules and how. This might seem trivial as there is documentation available, which lists all the parameters that can be read out. However, experiments showed quickly that we cannot trust the documentation because what was supposed to be current was not.

To experimentally show that what is supposed to be current is not, we set this pseudo-current to a constant value, which made the module turn. We then tried to resist the motion of the arm by pushing against it. This should require a constant force at all speeds but that was not the case.

A short note on nomenclature: On one side of the motor controller we have the DC bus, which supplies the motor controller with an almost constant voltage of about 24 V from the power supply. The current on the DC bus varies. We call these two the *bus voltage* and the *bus current*. On the other side of the motor controller we have the actual motor. The voltage applied to and the current running through the motor both vary. We call these two just *voltage* and *current*.

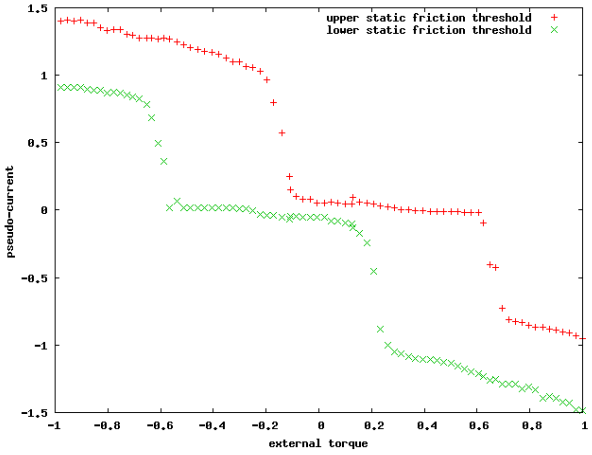

Fig. 1. Voltages at which the arm starts to move upwards (against gravity) or downwards at different angles.

Table I lists motor variables, showing both the description from the documentation and the name we gave them based on our experiments.

One experiment we ran for identifing the variables was that of gravity compensation. We wanted to figure out how much voltage is needed for gravity compensation. We measured two different voltages for each angle: The one when the arm at rest started to move upwards and the one when it started to move downwards. We tried this for different angles of the arm, i. e. for differernt external torques acting on the module due to gravity. Figure 1 shows the two voltages for different angles. We believe the difference between the two values is due to static friction. But we do not have an explanation for why this difference is not constant or the results overall.

Another experiment we ran is that of pushing against a fixed object. We set up the robot arm such that it was pushing against an object without moving. We applied

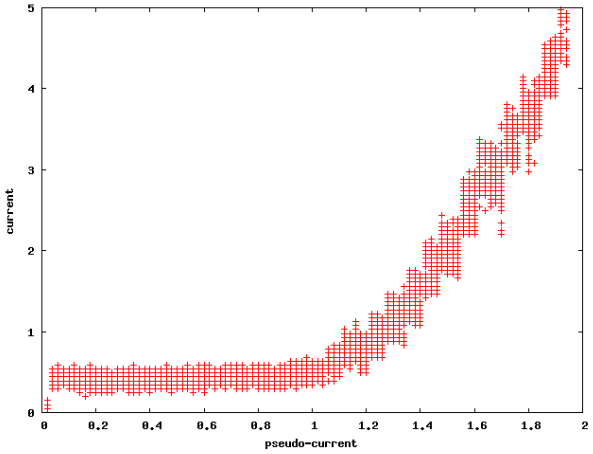

Fig. 2. Voltage (x-axis) vs. bus current (y-axis) while pushing against a fixed object. Labeling on the graph is outdated.

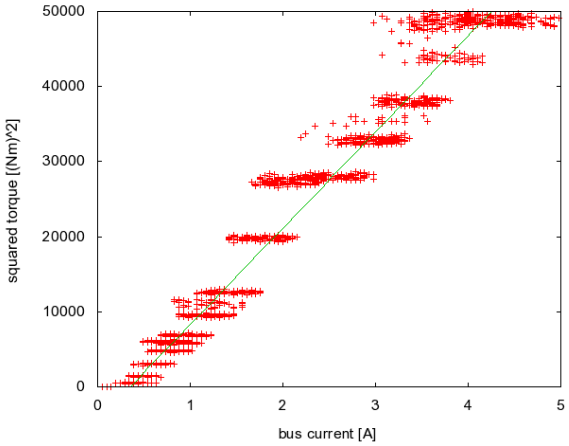

Fig. 3. Bus current vs. squared torque at rest.

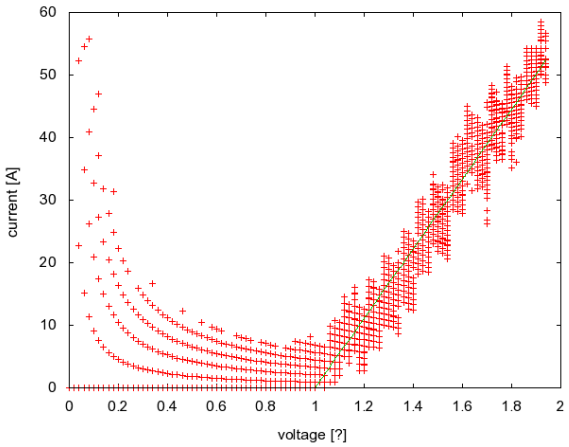

Fig. 4. Voltage vs. current while pushing against a fixed object. Current has been calculated as described in section III.

different voltages to the motor and measured motor variables and the force applied using the force/torque sensor.

Figure 2 shows the relationship between voltage and bus current. Between 0 and 1, changing the voltage does not seem to have any effect on anything. The reason for that is still unknown. Bus current should be quadratic in voltage. If it was the motor current, it would be linear. It is hard to tell whether it is linear or quadratic from figure 2.

A better indication that bus current is actually bus current is figure 3. In the static case, when torque grows, both voltage and current grow linearly. Assuming that the power on the bus and at the motor is equal and given the fact that bus voltage is constant, bus current has to grow quadratically with torque. This is what figure 3 clearly shows.

III. Estimating Current

We can meassure the bus voltage and bus current as well as the voltage applied to the motor. What we cannot meassure is the actual current running through the motor. However, the current can be estimated based on the three other values.

First, we based our calculation on the assumption that the power drawn from the power supply is the same as the power applied to the motor, i.e. there is no power loss in the motor controller. This gave us the following equation:

$$I = \frac{I_b U_b}{U}$$

Unfortunately, this assumption does not hold. When applying zero voltage, the average bus current is 0.21 A. This is a serious problem because the calculated current will be pretty high when voltage is close to zero. When voltage is small, a decrease of the voltage leads to an increase of current. This would also make the controller fail.

To bring our model closer to reality, we included a constant power loss in between DC bus and motor. This yields the following equation:

$$I = \frac{I_b U_b - P_{loss}}{U} \approx \frac{(I_b - I_{loss}) U_b}{U}$$

To calculate the power loss we did not just use the bus current meassured at zero voltage. Instead, we reused data collected previously on the relationship between bus current and torque. See figure 3. Force depends linearly on current. In the static case, current depends linearly on voltage. Thus, bus current should grow quadratically

ID	Documentation		Here	
	Name	Description	Name	Symbol
-	-	-	Current	I
77	Cur	Actual current (Actual value)	Voltage	U
112	ActFMotorCurrent	Actual DC Bus Current in A	Bus current	I_b
113	ActFMotorSupply	Actual DC Bus Voltage in V	Bus voltage	U_b
120	-	-	Nominal bus current	
121	-	-	Max bus current	
126	-	-	Nominal bus current overshoot time	
127	-	-	Max bus current overshoot time	
130	-	-	Max voltage	

TABLE I

LIST OF MOTOR VARIABLES THAT ARE IMPORTANT FOR THE TORQUE CONTROLLER. COMPARING DOCUMENTATION VS. OUR NOMENCLATURE BASED ON OUR EXPERIMENTS

with torque. After squaring the torque we get a linear graph. The data can now be approximated by a line using least squares.

The data is approximated by:

$$\text{torque}^2 = 12828 I_b - 4531$$

The linear approximation is zero at $I_{loss} = 0.3532A$, which is a little bit higher that the 0.21 A meassured before.

IV. CURRENT CONTROLLER

After being able to estimate the current, we can actually try to control it. We used an integral gain only to set the voltage based on current error. I tried different gains. Small gains made the controller rather slow and large gains caused a large overshoot and heavy oscillations. A gain of two seemed to be a good compromise. The resulting controller is shown below.

$$U(t) = 2 \int_0^t (I(\tau) - I^d(\tau)) d\tau$$

Figures 5, 6, 7 and 8 show the response of the closed-loop to different desired currents.

We also tried to include a derivative and proportional gain. This would normally help counter oscillation and overshoot. However, in this case it did not help. When choosing the gains very small, the addiotional derivative and proportional terms hardly had any effect. When choosing them larger, the system became unstable and oscillated wildly. The reason for that is probably the noise in the estimated current. This is caused by the way we are estimating current and the fact that the measurement of bus current is very noisy. Taking the derivative of noise makes it even worse.

To counter that problem I tried to low-pass filter the current. This results in less noise but also a small time lag. This time lag made the original, pure integral

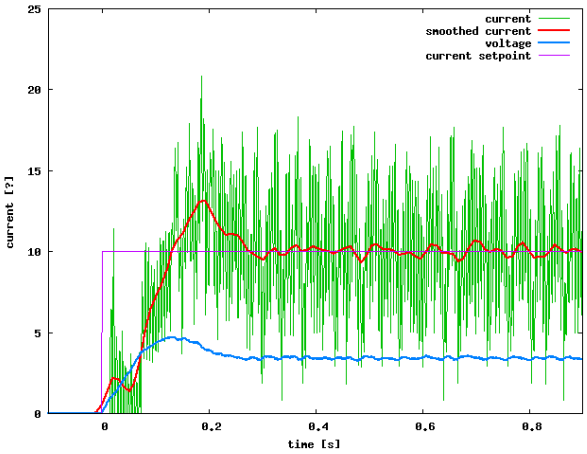

Fig. 5. Step response of the closed-loop system to a 10 A step in desired current at rest.

controller oscillate even more. With the applied filter we were able to use derivative and proportional gains. These got the oscillation back to the level of the original, pure integral controller without a filter. However, we were not able to achieve a better performance than the original controller. That is why I kept the pure intregral controller and do not use any filtering.

V. CURRENT-TORQUE RELATIONSHIP

After being able to control the current, we need to know the relationship between current and torque in Nm in order to govern the torque.

So far, in the experiments when we related some variable to torque, we always used the raw data from the force sensor. In order to get the real torque, we need to transform the force sensor value into a torque with known unit. To do that, we put different weights on the end of the robot arm and measured the values from the sensor. The results are shown in table II.

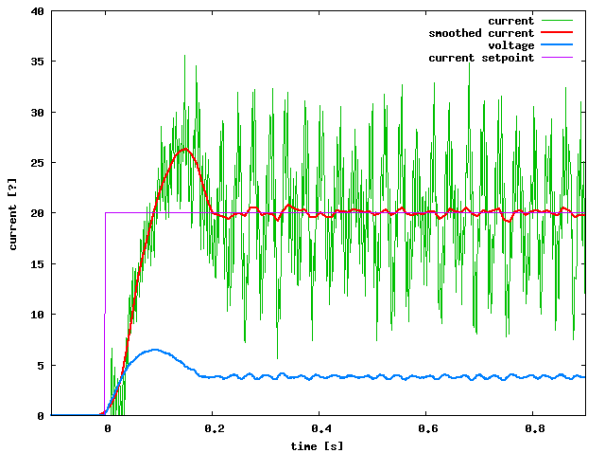

Fig. 6. Step response of the closed-loop system to a 20 A step in desired current at rest.

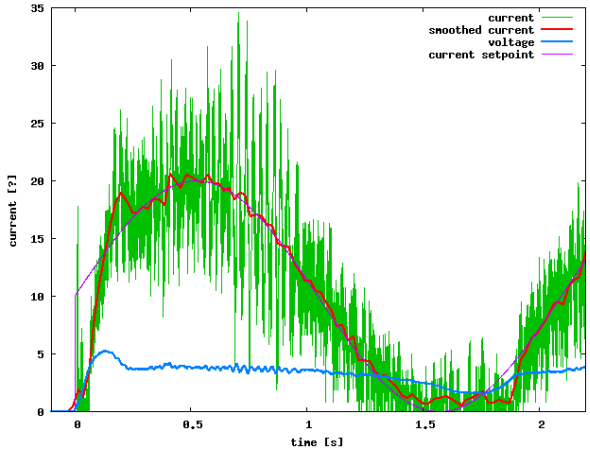

Fig. 7. Response of the closed-loop system to a continously changing, sinosoidal desired current at rest.

weight	x	y	z
0 lbs	-1.6116	-0.0232	-4.0575
0 lbs	-1.6114	-0.0232	-4.0576
5 lbs	-1.7454	-0.0235	-4.0572
10 lbs	-1.8892	-0.0238	-4.0567
15 lbs	-2.0276	-0.0237	-4.0562
20 lbs	-2.1626	-0.0238	-4.0555

TABLE II

FORCE SENSOR READINGS FOR DIFFERENT WEIGHTS. THE VALUES ARE AVERAGES OVER SEVERAL HUNDRED MEASSUREMENTS.

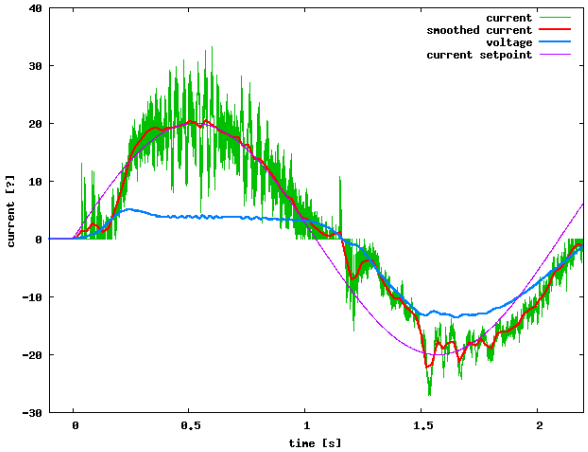

Fig. 8. Reponse of the closed-loop system to a continously changing, sinosoidal desired current. For a desired current smaller than 0, the system starts moving.

Transforming pounds in Newtons and solving by least squares gives a sensor value to force relationship:

$$\text{force} = \text{sensor value} \cdot 161.1058 \text{ N}$$

The length from the module center to the weights was about 88 cm in this experiment. Further transforming force into torque yields the sensor value to torque relationship:

$$\text{torque} = \text{sensor value} \cdot 141.7733 \text{ Nm}$$

With this information we can acribe real units to the y-axis of figure 9. We linearly approximated the sample data using least squares. Samples for a voltage smaller than 1 (shown in green) have been excluded from the approximation. The result is shown as a blue line in the graph. The slope of the line is

$$K_t = 4.2657 \frac{Nm}{?}$$

Multiplying current with this constant gives us torque.

VI. RELAXING MOTOR LIMITS

Work presented in this section was done in cooperation with Mike Stilman and Martin Levihn.

To achieve the desired speed for torso movements of Golem Krank, we need to be able to apply high torques. By default the Schunk modules have quite conservative limits on the maximum bus current and applied voltage. These keep us from achieving high torque, although the motor would physically be able to achieve them.

When trying to command voltage above the limit, the voltage actually applied equals the limit. When

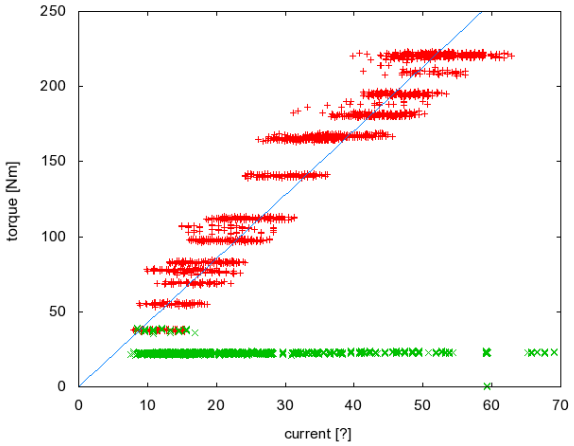

Fig. 9. Current-torque reationship. Current is estimated as described in section III. Torque is measured using the force-torque sensor at rest.

the bus current stays above the limit for a defined amount of time, the motors shut down completely. For the bus current there are actually two differrent limits, which have different overshoot times defined. There is a nominal and a maximum bus current. The maximum bus current is a relatively strict limit allowing only a very short over shoot while the nominal limit can be violated for several seconds. Shooting over the voltage limit is not as big of a problem because the voltage will automatically limited to meet the limit. Fot the bus current, however, we should try not to reach the limit because the motor will shut down and needs to be reseted before it takes any more commands.

Table I shows the motor parameters that store the limits. All these parameters are not mentioned in the documentation. There is also a flag in the motor configuration word, called ”allow full current” in the documentation, which makes the motor ignore the voltage limit.

We were able to command high voltages by setting the flag to allow for full voltage. We were also able to set new values for the bus current limits. This enabled us to achieve higher bus currents than the previous limit. However, our new, higher bus current limits are not enforced. Currently, we do not know why this is the case.

Instead, we enforce the new limits through our software, which constantly measures the bus current and shuts down the motor if it is above the limit. However, this might not be as fast and reliable as an enforcement directly on the module. The hard limit should not be reached during normal operation as we do not want the motor to shut down. Thus, in addition to this hard limit, which leads to a shut down, we try to set the voltage such that the bus current stays below a lower soft limit.

The modules are equipped with 15 A fuses for the DC bus. This is the physical limit for the bus current we should not exceed because replacing the fuses is a lot work. We did that several times during our experiments.

VII. Conclusion and Future Work

In this paper we showed all the steps that were necessary to implement a torque controller for the Schunk modules. We have a working torque controller. However, it still needs improvement in order to work reliable. The possible improvements include a different current controller and a better enforcement of relaxed motor limits.

Currently we try to govern torque by controlling current and doing a straight-forward calculation from current to torque. However, the estimation of current is based on assumptions and a noisy bus current measurement. An alternate and proably better method would be to model the motor dynamics and do open-loop control using the angular velocity of the motor to set the voltage based on the desired torque. This would avoid overshoot and oscillation and would also make it easier to design the controller such that we stay below the hard limit for the bus current.

There are still results from experiments with the motors that we cannot explain. A complete understanding of the motor might not be necessary but it would increase confidence in our approach. Also, so far we do not have a hard proof, e.g. a nice graph, that shows that voltage is actually voltage and not current as claimed by the documentation.

We need to find out why the motor limits are not enforced any more after they have been changed. Currently, the controller is not very good at avoiding the hard bus current limit. The soft limit has to be set way below the hard limit to ensure that the hard limit is never reached, which keeps us from achieving higher torques. It will probably become easier to avoid the hard limit once we have a model of the motor dynamics and do open-loop control on it because there will not be overshoot and not as much oscillation.

To implement the current controller and to change the motor limits, we needed to adapt PCIO. As the design of PCIO did not allow for an easy adaption and as we did not want to spend a lot of time on rewriting PCIO, we often implemented quick hacks, which included copying large chunks of code. At some point PCIO needs to be rewritten and cleaned-up.