Design and implementation of a general main axis controller for the ESO telescopes

Stefan Sandrock, Nicola Di Lieto, Lorenzo Pettazzi, Toomas Erm
European Southern Observatory, Karl-Schwarzschild-Str. 2, 85748 Garching bei München, Germany

ABSTRACT

Most of the real-time control systems at the existing ESO telescopes were developed with "traditional" methods, using general purpose VMEbus electronics, and running applications that were coded by hand, mostly using the C programming language under VxWorks.

As we are moving towards more modern design methods, we have explored a model-based design approach for real-time applications in the telescope area, and used the control algorithm of a standard telescope main axis as a first example.

We wanted to have a clear work-flow that follows the "correct-by-construction" paradigm, where the implementation is testable in simulation on the development host, and where the testing time spent by debugging on target is minimized. It should respect the domains of control, electronics, and software engineers in the choice of tools. It should be a target-independent approach so that the result could be deployed on various platforms.

We have selected the Mathworks tools Simulink, Stateflow, and Embedded Coder for design and implementation, and LabVIEW with NI hardware for hardware-in-the-loop testing, all of which are widely used in industry. We describe how these tools have been used in order to model, simulate, and test the application. We also evaluate the benefits of this approach compared to the traditional method with respect to testing effort and maintainability.

For a specific axis controller application we have successfully integrated the result into the legacy platform of the existing VLT software, as well as demonstrated how to use the same design for a new development with a completely different environment.

Keywords: telescope axis control, real-time systems, model-based design, Matlab-Simulink-Stateflow, automatic code generation, embedded systems, hardware-in-the-loop testing, correct-by-construction, V-model.

1. INTRODUCTION

All of the major telescopes at the ESO observatories of Paranal and La Silla in Chile are equipped with the same standard main axis controller, which in total amounts to more than sixty instances of tracking axes for azimuth and altitude/elevation, as well as adapter/rotator and instrument co-rotator axes. The original design of the controller hardware and software was made about fifteen to twenty years ago, and became part of the standard technologies that were established for the VLT. It uses VMEbus crates, equipped with PowerPC CPU boards and dedicated digital and analog I/O cards to interface with the field hardware. The actual controller was then implemented in software based on the VxWorks real-time operating system and embedded into the general VLT-software framework. All the relevant controller and filter algorithms were hand-coded and programmed in ANSI-C. They require the target-system to run, whereby test and simulation possibilities are very limited.

In terms of requirements and performance, the old controller was intended for the “large” ESO telescopes, in particular the NTT and then the 8-meter Unit Telescopes of the VLT. Over time, however, the same controller was then also employed with some variations for the “smaller” telescopes at the observatories, including the VLTI Auxiliary Telescopes (AT, 1-meter-class), and the VST and VISTA survey telescopes. This re-use was beneficial for maintenance and support, but also implied to accept some compromises for implementation and performance.

The old design was essentially a cascaded position and velocity controller, the latter could optionally be an external device. The position controller used three separate algorithms (PI, polynomial, square-root) that were switched based on

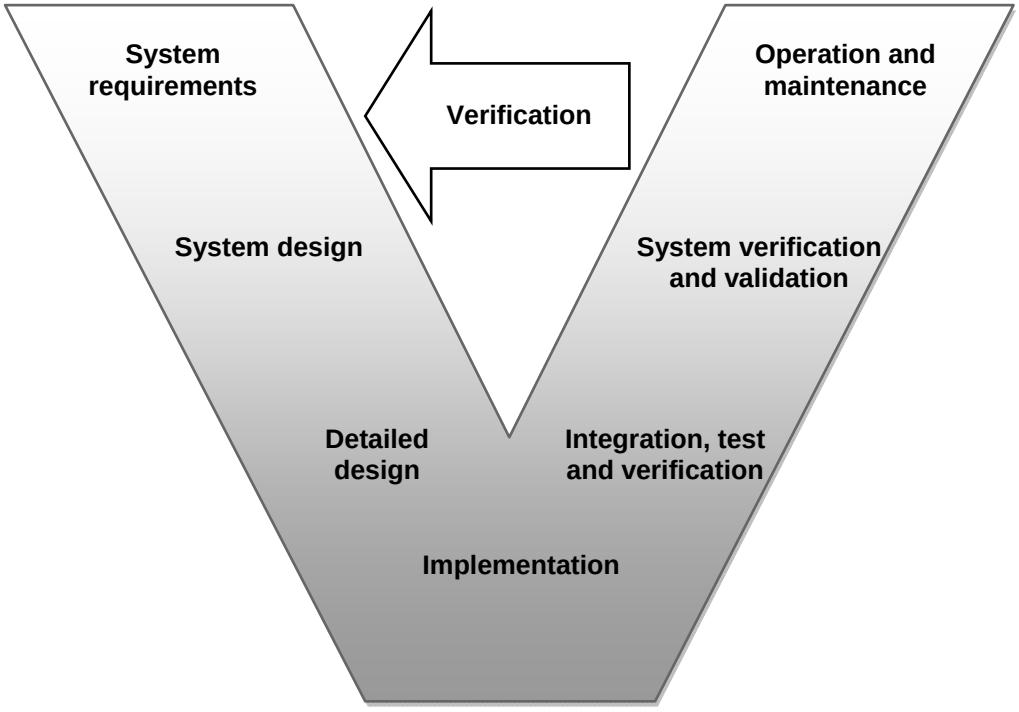
the magnitude of the position error. While internally coded as a digital controller, it uses purely analog input/outputs from the encoder and tachometer inputs, to the velocity or torque reference outputs.

For new ESO projects in the instruments and telescopes area, but also to upgrade existing systems, we are now aiming to provide a new standard solution for these applications which is able to provide better performance and maintainability. The driving factors for a new design methodology were to overcome some of the original drawbacks mentioned above, including better testability, integrated simulation facility, reduction of coding errors and testing effort. As further detailed in the next chapter, we adopted model-design and follow the so-called V-model and the "correct-by-construction" paradigm, including the option to generate target code from the models.

The selection of tools for design and implementation are made in order to be closely oriented on common "industry-standards" and to provide at the same time enough flexibility to support the various requirements of existing as well as future telescope tracking axis controllers within ESO. The Mathworks toolset for Matlab/Simulink are the natural choice in this case. Maintenance and long-term support aspects also play an important role, since existing system are facing obsolescence and replacement/upgrades shall be supported for a sufficient time into the future.

2. METHODOLOGY

The V-model is a lifecycle process model for developing mechatronic systems including both software and hardware. It is particularly well suited to the development of modern digital control systems. The V-model is oriented to individual projects rather than to an organization. The project is split into stages, each with an input product and an output product. These are living products in the sense that they are refined and updated over the course of the project. The project stages do not follow each other linearly; when the product from one phase is complete, it is verified and only when the verification has succeeded the next phase begins, moving onto a deeper level of detail, until all detail is worked out and the project is completely implemented and verified. The ongoing verification intrinsic of this process builds confidence in the project; problems are identified and corrected at the earliest opportunity in a tight feedback loop, thus reducing the cost and effort to fix them. The process can be illustrated graphically on a diagram resembling the letter V – hence the name.



Over the last two decades the automotive and aeronautic industries have increasingly and successfully relied on the simulation of mechatronic systems and control algorithms. By providing a common framework for design and communication across different engineering disciplines, this approach fits particularly well in the V-model. The simulation of control algorithms has naturally evolved towards the automatic generation of computer code from the model, enabling engineers to continuously test the design in increasing level of detail, as it is developed.

Model-based design can be defined as using a system-level model to generate an executable specification that describes unambiguously, in a mathematical form, the behavior of the system being developed. Usually the foundation of such a model is a lumped parameter mathematical model that describes the dynamic relationship between inputs and outputs of the system through ordinary differential equations (ODEs). In cases where the system becomes too difficult or time consuming to model effectively, a variety of black-box empirical modeling approaches, based on empirical data, is used instead. It is also possible and common practice to use empirical data for improving the accuracy of a first-principle mathematical model, for example by identifying the correct value of one or more parameters through system identification techniques, resulting in a grey-box model.

The system level model is used to design and simulate the control algorithms. This allows early testing to quickly find and eliminate any problems that may lead to incorrect or possibly unsafe operation of the controlled system. The continuous testing and verification lead eventually to a so called **correct-by-construction design** where by the time it is implemented on the final hardware, the control system will require no major changes. Standard tests can be defined by modeling and later used to verify the consistency of the implementation; this ensures consistency and immediate exposure of the impact of design changes and quick linking of the change to the cause of any discrepant behavior.

Often the system to be controlled (plant) is not available until the latter stages of implementation; in these the so called **Hardware-In-the-Loop testing** bridges the gap. HIL testing involves implementing the plant model in a real time test system that emulates the behavior, inputs and outputs of the real plant. The HIL test system is then connected to the real control hardware, running the real control software; consistently with the V-model, this allows verification, early identification of any mistakes/problems and quick correction, thus reducing their impact on the cost and schedule.

The V-model outlined above is a general lifecycle process which is intentionally defined to be independent from the specific selection of both development environment and target platform. This feature is particularly important and allows for an easy application of the V-model paradigm across all ESO's projects. This statement will be demonstrated in chapter 5 of the present paper.

3. REQUIREMENTS

This chapter provides an overview of the requirements for the new axis controller, as defined in [1].

Functions:

The ESO standard telescope axis controller is intended to serve as standardized solution for all applications related to the control of telescope main tracking axes (altitude, azimuth, rotator, adapter), with the possibility to use it also for applications with similar requirements (e.g. instrument co-rotator). It is supposed to preserve compatibility with previous ESO solutions for the VLT, so that an upgrade to the new controller is feasible ideally without interface changes, and as far as possible with respect to existing obsolete hardware and/or software.

The following main functions shall be fulfilled:

- Control a single axis with a specified performance.
- Access the specified sensors and actuators and process the data accordingly.
- Provide interfaces for interlock and limits handling.
- Provide services to configure and tune the axis control loops.
- Provide services to perform maintenance, system diagnostic, performance measurement, and monitoring of the various components.

The basic purpose of the controller is to drive the telescope axis under position control to follow the apparent motion of an astronomical object in the axis coordinates. The ESO standard telescope axis controller shall in principle follow the classic scheme of a cascaded position and velocity controller with PID characteristic and feed-forward path. In addition, the following components shall exist:

- Rate limiters at its inputs shall be used in order to apply maximum input errors for position, velocity, and acceleration with the aim of improved stability characteristics.
- A state observer shall be incorporated, which generates estimated velocity and position values with increased quality compared to the measured feedback sensor signals.
- For test and maintenance, a diagnostic facility shall exist, which provides the means to monitor essential performance parameters, and to apply test signals for control-loop tuning and test purposes.

Operational Modes:

The operational mode defines what the controller is supposed to do:

- Off: The axis is not driven (open loop). Output torque and velocity references are set to zero, inputs are ignored.
- Velocity Control (VC): The axis is driven in velocity control. The velocity loop is closed (while the position loop is opened), using the user-defined velocity reference as instantaneous set-point.
- Position Control (PC): The axis is driven in position control.

The position and velocity loops are closed, using the user-defined position reference as instantaneous set-point.

In addition there shall be modes intended for test and maintenance.

Switching of modes shall be possible at any time.

When switching between the operational modes, the usual rate limitations shall be applied.

Interfaces:

The following lists define the interfaces of the controller. All input signals shall meet real-time constraints, i.e. they are subject to strict timing deadlines with respect to the clock rate of the controller.

Inputs:

- Operational mode selection: To select the operational mode, i.e. position or velocity control mode, respectively special modes for maintenance and test.
- Time reference: Provides clock reference signal with user-defined frequency, on which the servo rate is based.
- Position target: In position control mode is used as instantaneous set-point value.
- Actual position: Sensor to measure the actual position (typically encoder)
- Velocity target: In velocity control mode used as instantaneous set-point value.
- Actual velocity: Sensor to measure the actual velocity (typically tacho).
- Enable signals: If asserted then the position/velocity control loop shall be closed.

Outputs:

- Torque reference: Commanded torque, to be used by connected motor drives/amplifiers.
- Velocity reference: Commanded velocity, to be used by an external velocity controller (if existing).

4. DESIGN AND IMPLEMENTATION

The foundation of the controller is two nested digital Proportional+Integral+Derivative controllers, position and velocity. An observer is included in order to track the axis states and, among others, synthesize the velocity signal from position measurements, for optional use instead of the velocity directly measured by a sensor on the system.

No mode switching is used to prevent overshoots during large transients. Rather, care is taken to limit velocity and acceleration of the position set-point to ensure that only smooth, physically realizable target trajectories are actually used as target by the control loop. Thus the error signals are never given a chance to grow large and a single set of firm control gains can be used during slewing as well as tracking.

Particular care is taken to preserve the continuity of the controller states and outputs; to prevent integrator windup during output limiting; and finally to initialize the controller internal states. As a result enabling the velocity or position loops and toggling between observed or measured signals can be done at any time.

The controller has been designed in Simulink and the block diagram is executable to allow simulation of the specification and code generation using the Simulink/Embedded coder toolboxes. The following figure depicts the position/velocity controller block diagram. Details are contained in [2].

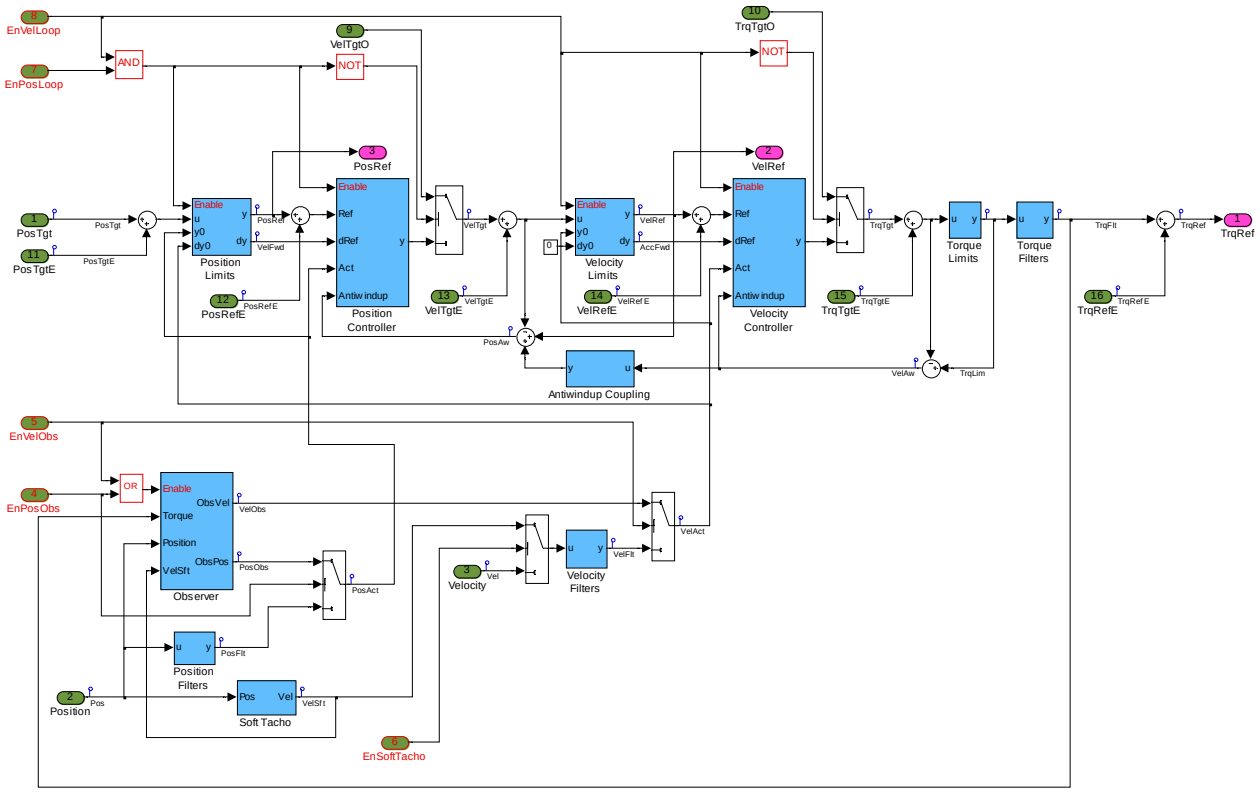


Figure 1. Simulink model of the axis controller.

An important requirement is the ability to inject test signals at various points in the controller data path and to log internal variables at the full controller sample rate, for debugging and measurement purposes. This is achieved by adding inputs for the excitation signals to the schematic and by naming the internal variables of interest. The test signals are produced by a separate, hand written embedded signal generator subroutine before being input to the schematic. The internal variables are sampled and then written to data files by a separate real-time logging task.

In a first test implementation the real time C code generated from the schematic above was integrated within the existing ESO axis controller real time software, which also includes a state machine. Following the success of the test on the VLTI Auxiliary Telescope main axis controller, it was decided to also re-implement the state machine in

Simulink/Stateflow. As the states of the old state machine are tightly controlled and part of the interface, they were just reproduced in Stateflow, without a complete re-engineering. The resulting schematic is as follows:

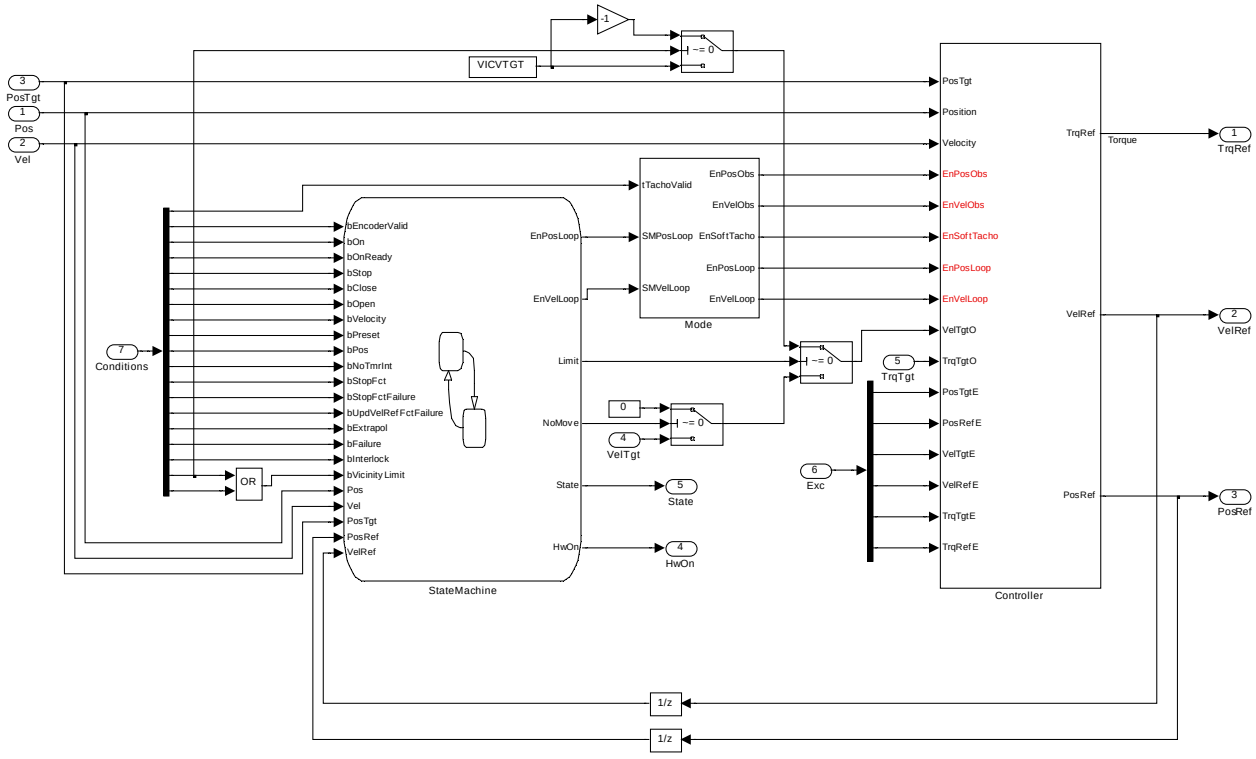


Figure 2. Simulink/Stateflow model of the controller with statemachine.

The old state machine was designed from a state transition table and then coded by hand in C, which resulted in a source code that was very difficult to maintain. Expressed in the form of a state diagram, the new version also added hierarchical states, and as a result becomes much more readable and easier to maintain, although essentially the same states and transitions are implemented.

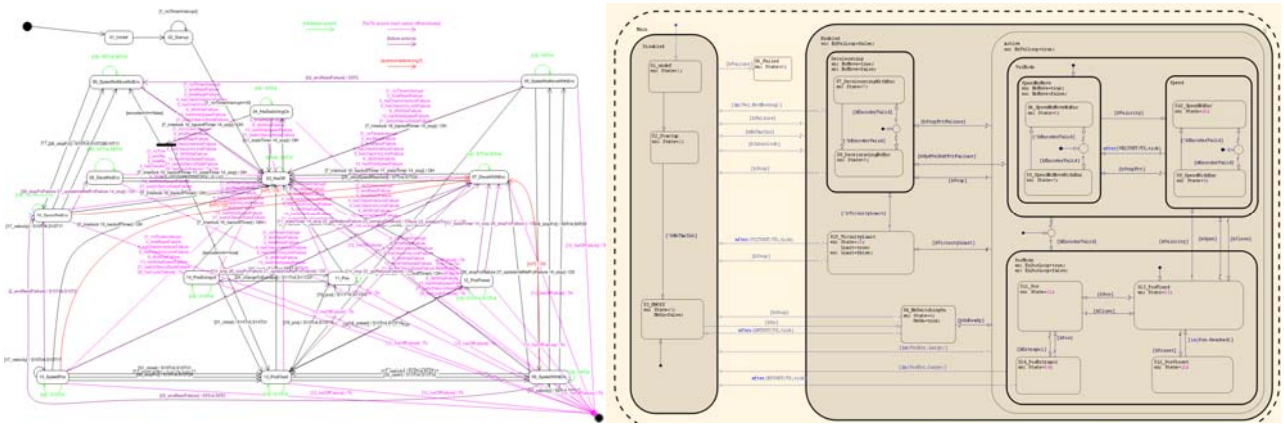


Figure 3. Comparison between old (left) and new (right) state machine chart.

After testing the schematic using a simple lumped parameter model of the telescope axis dynamics, real time code was generated by Simulink/Embedded coder and linked into the existing VxWorks project. The initial real time implementation was tested on a Hardware-in-the-Loop test system running on a National Instruments CompactRIO chassis.

5. RESULTS

In this chapter we present the results that we have achieved for two separate projects, namely for the upgrade of the main axis controllers of the VLTI Auxiliary Telescopes at Paranal, and for a new axis controller in the frame the E-ELT prototyping activity that was tested at the azimuth axis of a VLT Unit Telescope.

5.1 Upgrade of VLTI Auxiliary Telescopes main axes controller

For the specific case of the VLTI Auxiliary Telescopes (AT), a test period was scheduled in Nov/Dec 2011 in order to prove the concept of the new controller for this application, and to resolve some other longstanding problems with the existing axis control system – in particular the tendency to sporadic oscillations and even loss of control resulting in software failure modes being activated, traceable to a problematic tachometer signal under certain conditions.

Following the installation of the new software the problems have disappeared altogether. The tachometer problems have been worked around by dynamically switching to using the derivative of the encoder signal (Soft Tacho) instead of the real tachometer signal when the latter is detected faulty (for example due to spikes, or following an analog sensitivity switch).

Furthermore the tracking performance analysis shows tracking errors down by approximately 60% compared to the old controller. As a result, the software is now in full operation on all Auxiliary Telescopes. The following pictures show the behavior of the new software, also including some comparisons with the old performance. The optional observer velocity control strategy was also tested on AT1 and as expected the measured torque rejection improved at low frequency without extra amplification at higher frequency. Details are contained in [5].

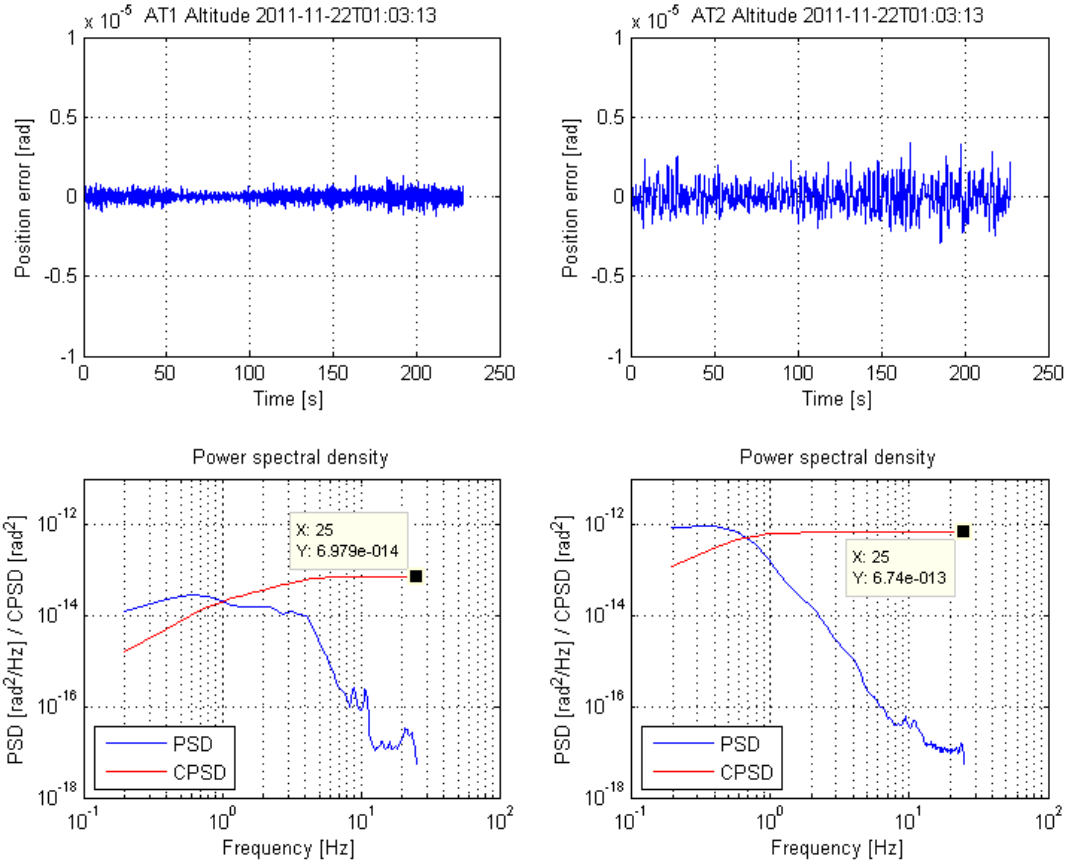


Figure 4. Comparison of performance in wind: AT1 with new software (left), AT2 with old software (right).

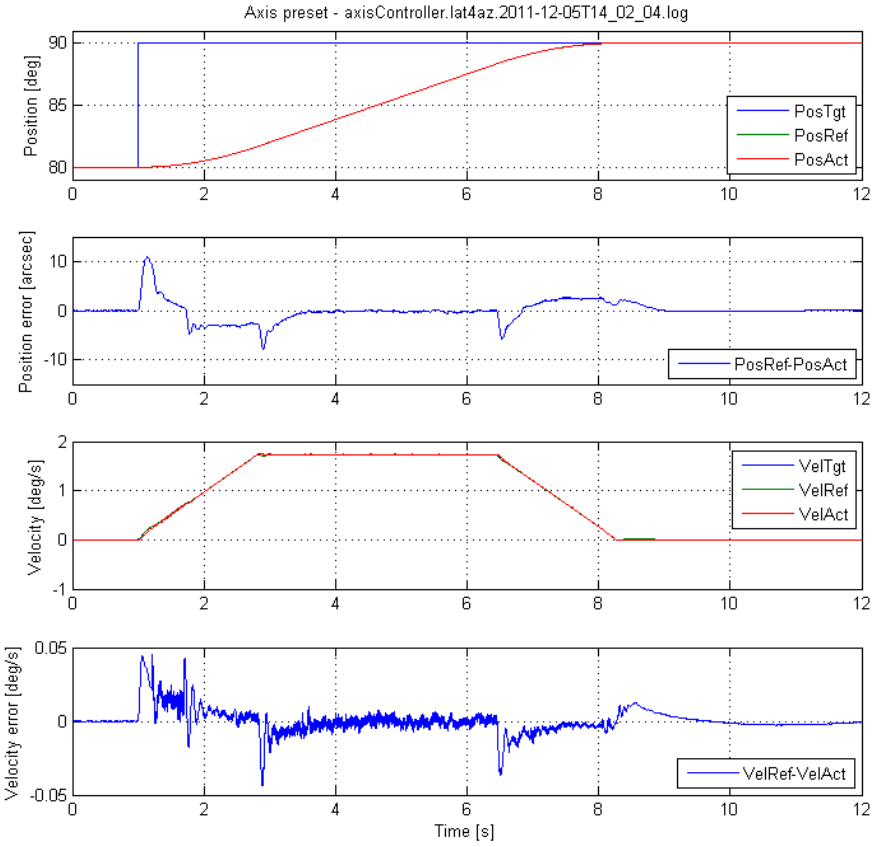


Figure 5. Behaviour during preset (slewing).

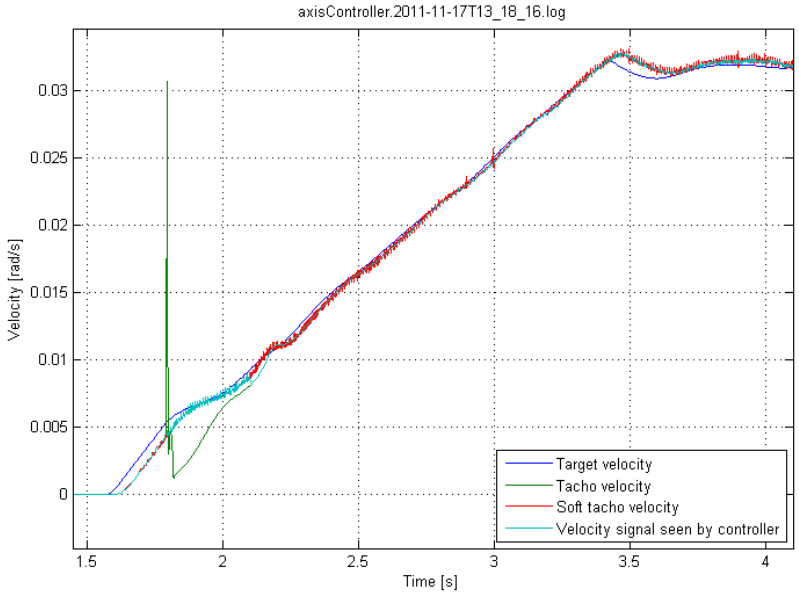


Figure 6. Behaviour of dynamical switch to work around tachometer problem.

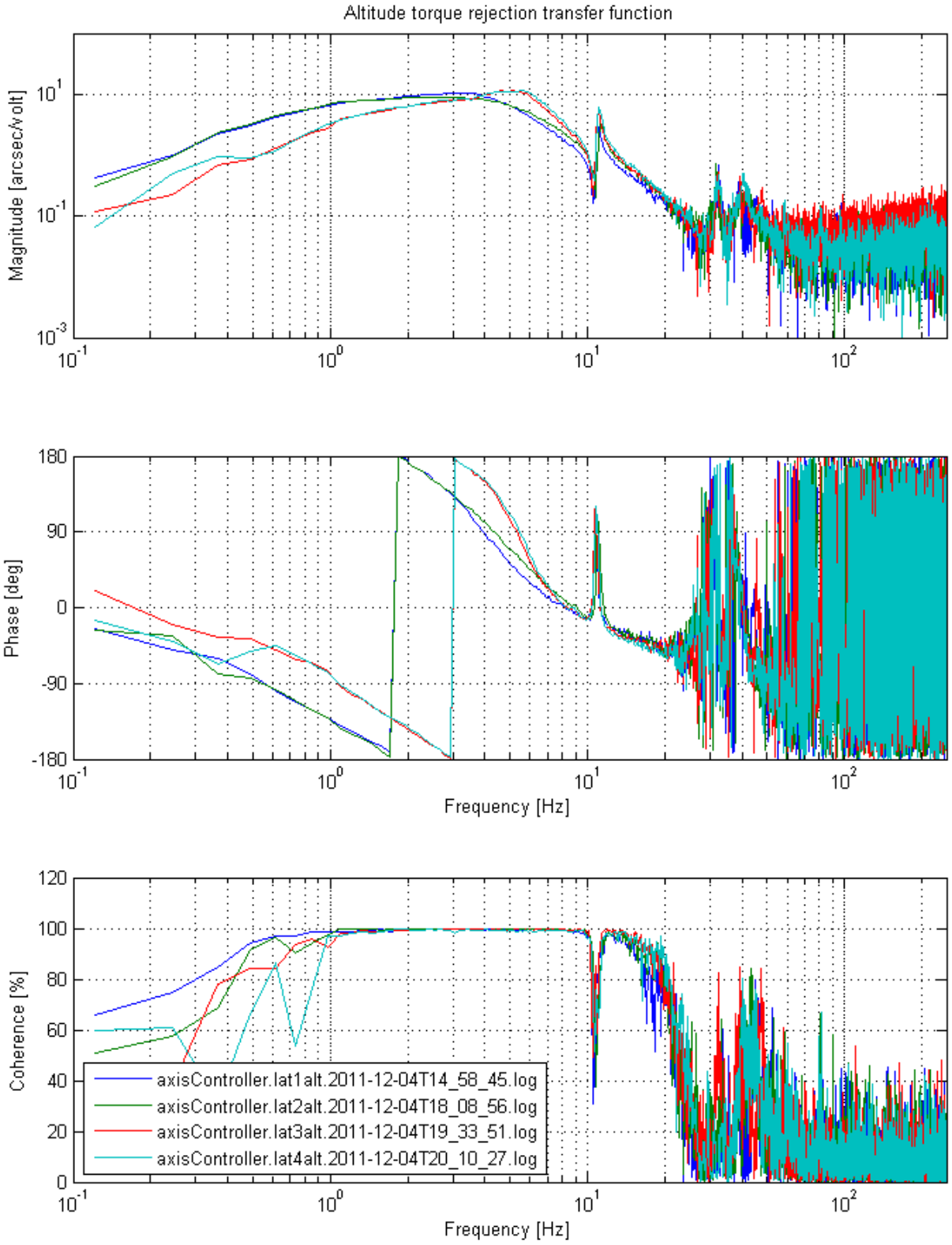


Figure 7. Measurement of torque rejection transfer function for the four VLTI Auxiliary Telescopes.

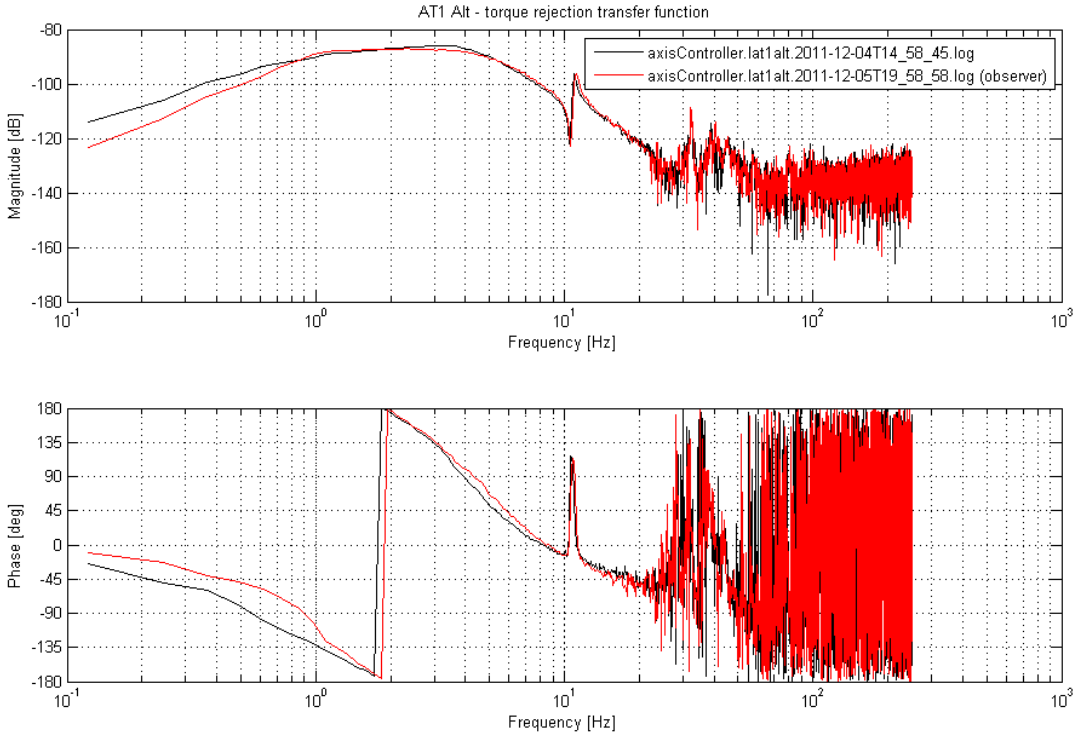


Figure 8. AT1 Altitude torque rejection transfer function improvement with observer.

5.2 E-ELT Prototyping as VLT Unit Telescope Azimuth axis controller

The prototyping activity for the E-ELT main axis controller offered the technical opportunity to apply the process and algorithm described in the previous sections to the (re-)design of the azimuth axis controller of the VLT Unit Telescope (UT) [4]. In order to comply with the standards defined for the E-ELT axis controller prototype (i.e. controller to be implemented in M-Language for further integration in LabView environment), Matlab code fulfilling the Simulink executable specification presented in section 4 has been generated and encapsulated into a Level-1 S-Function. Such a Matlab script could be directly integrated in the LabView environment by using MathScript nodes. Additionally, measurements of the open-loop telescope response have been used in order to identify the torque-to-azimuth-angle transfer function of the UT telescope at 90 degrees altitude. The resulting model of the UT telescope and the S-Function implementation of the axis controller have been integrated together in order to build a Simulink model of the closed loop system to be used for model-based controller tuning and SIL test. A HIL test bench could be also easily set up exploiting the abovementioned components. Both HIL and SIL test benches have been extensively used during the pre-commissioning phase to optimize the prototype controller configuration parameters and remove possible implementation inconsistencies.

The results of the axis controller commissioning activity are presented in the figures below. The closed loop velocity transfer function associated to the prototype controller is compared against the one of the existing controller implemented in the VME Local Control Unit (LCU). The prototype controller shows better attenuation of noise at frequencies above 30Hz and yields a slightly higher closed loop bandwidth with respect to the system in closed loop with the VME LCU controller (left hand side plot). The right hand side plot of Figure 9 shows a comparison between the closed loop disturbance rejection transfer function achieved by the prototype controller against the one achieved by VME LCU. The performance of the two controllers is fully comparable.

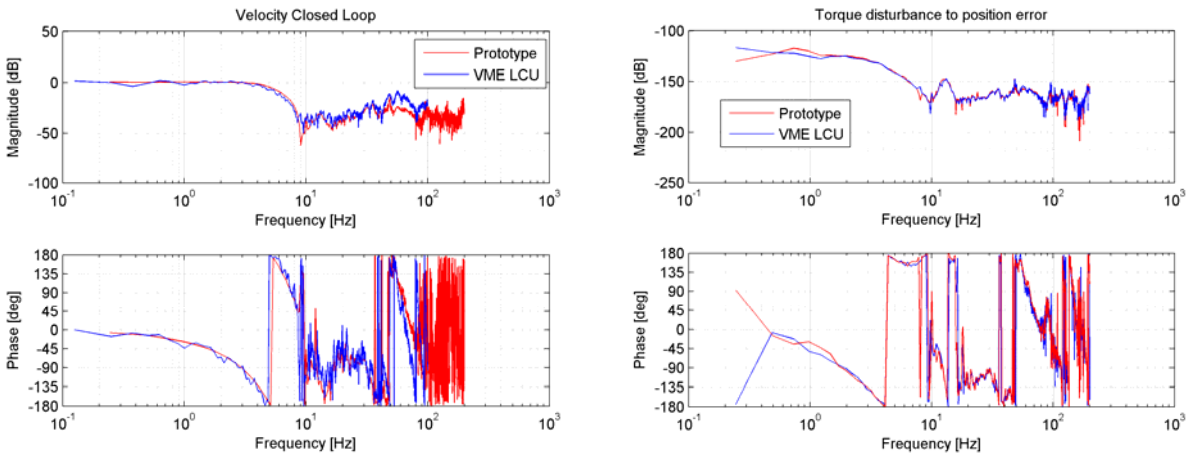


Figure 9. Closed loop velocity and position transfer function of prototype controller (left) vs. original controller (right).

Finally, the following figure shows the response of the closed loop system to a reference position ramp signal of 10^{-3} rad/s (approximately 200 arcsec/s). Both reference signal (denoted by PosRef) and actual telescope position (denoted by PosAct) are shown in the left hand side plot. Additionally, in the right hand side plot the RMS of the position error computed on a sliding window of 15 seconds is displayed. The maximum RMS error is below 0.0204 arcseconds during tracking which is consistent with the UT blind tracking performance requirement.

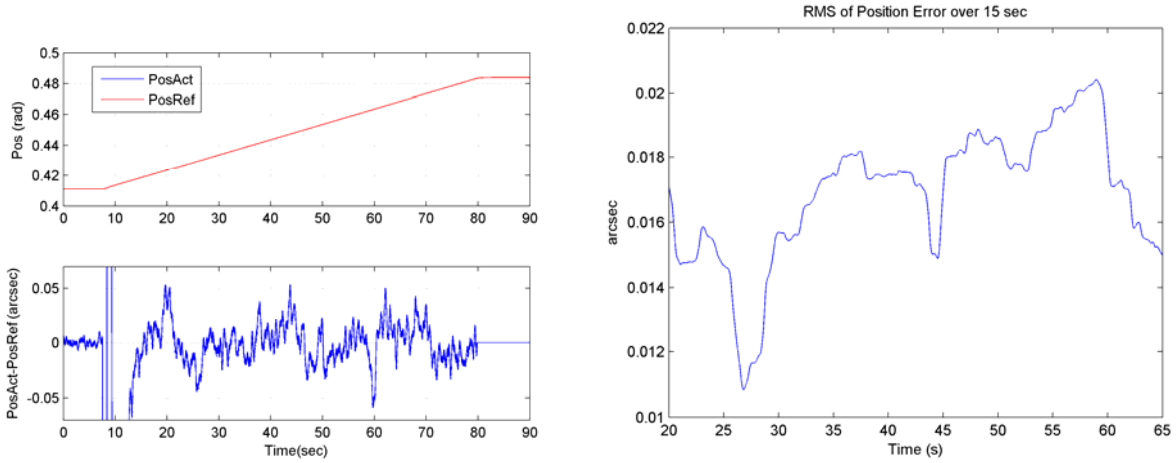


Figure 10. Time response of closed loop UT azimuth axis to ramp command.

It is worth mentioning that during the test campaign in Paranal the output current capability of the controller prototype hardware showed to be insufficient to drive the amplifier over the entire expected operational range. As a consequence of this, integrator wind-up caused large transient errors in telescope angular velocity to occur when commanding large slew accelerations. The SIL tool developed during the pre-commissioning activity could be easily adapted to reproduce such effect and therefore played a key role in the timely identification of the misbehavior occurring at the interface between the prototype controller and telescope drive.

6. CONCLUSIONS

We can conclude that the chosen methodology is in principle adequate for our controller designs, and the performance requirements of the two described examples have been met or exceeded with the implemented controller. The integration activities into the target systems were relatively straight-forward, but required additional manual work.

We have gained confidence in the correctness of the automatic code generation attached to Simulink/Stateflow, i.e. Embedded Coder from Mathworks; we have not found any occurrence of a software problem that could be attributed to the code generation; we did find, however, a number of "bugs" in the hand-coded integration frameworks.

Using this methodology, the testing approach has changed considerably, and in a very positive way. The possibility to simulate the controller already on the development system reduced the test effort on the actual target system. The HIL device made it possible to test the deployed target system under real conditions already in the lab (Garching), and that again minimized the test effort at the operational site (Paranal), where testing time is a scarce resource.

We believe that maintainability of the new solution has improved. The previous situation resembled virtually a deadlock, where any changes were avoided due to unpredictable and often un-testable effects on the target system as a result of a code change, with the risk of operational down-time. Now, with anticipated test and simulation possibilities, this can be handled in a better way.

The workflow supports the interaction of involved engineers from distinct domains (Control, Software, Electronics), with clear responsibilities and interfaces between them.

Obviously this workflow can only be uninterrupted if licenses for the used Mathworks tools are available throughout the whole chain. Therefore an upgrade of the related licensing infrastructure becomes necessary in order have the same equipment at development and production sites (Garching and Paranal in our case). At the same time the type of work is quite different between the two sites (e.g. new designs will most likely take place in Garching and sub-sequent maintenance at Paranal), which possibly implies a licensing scheme mismatch that will have to be resolved.

In the future we may explore the option to include the Simulink/Stateflow models into an over-all model-based design of new projects, which is based on the system-level description language SysML.

7. ACKNOWLEDGEMENTS

The authors would like to express their appreciation for the support from Electronics and Software teams at Paranal during test and commissioning phases, and to thank in particular Rodrigo Huerta and Guillermo Valdes for their contribution to the verification and the participation in the testing of the new axis controller.

REFERENCES

[1] ESO Standard Telescope Axis Controller Requirements Specification, GEN-SPE-ESO-50000-5301, Issue 1, ESO technical archive (2011).
[2] ESO Standard Telescope Axis Controller Control Algorithm Specification, GEN-SPE-ESO-50000-5302, Issue 1, ESO technical archive (2011).
[3] ESO Control Engineering Handbook, GEN-SPE-ESO-50000-4908, Issue 1, ESO technical archive (2010).
[4] E-ELT Control System Prototyping on VLT - Azimuth Axis Controller Verification Report, E-TRE-ESO-449-1173, ESO technical archive (2011).
[5] VLTI-AT Main Axis Controller Upgrade Test Report, VLT-TRE-ESO-15150-5503, Issue 2, ESO technical archive (2012).