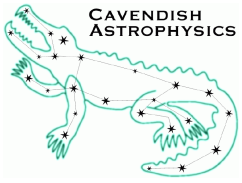
MRO Delay Line

Production Shear Sensor Software Functional Description

INT-406-VEN-1005

Bodie Seneta
bodie@mrao.cam.ac.uk

rev 1.0
23 January 2010



Cavendish Laboratory
Madingley Road
Cambridge CB3 0HE
UK

Change Record

Revision	Date	Authors	Changes
0.1	2010-01-20	EBS	First draft.
0.2	2010-01-21	EBS	Changed screen dump and associated text after discovering and fixing broken camera gain control.
1.0	2010-01-23	EBS	Changed title. Clarified decimation issues. Minor wording changes.

Objective

To describe the design and implementation of the shear sensor production software, and the differences between this implementation and the prototype implementation.

Reference Documents

RD1	MRO Delay Line Derived Requirements	INT-406-VEN-0107
RD2	(Prototype) Shear Camera Functional Description	INT-406-VEN-0105
RD3	Trolley Electronics Design Description	INT-406-VEN-0112
RD4	Metrology System & VME Hardware Design Description	INT-406-VEN-0113

Applicable Documents

AD1	Requirement specifications for the MROI "production" delay line software	INT-406-CON-0101
AD2	Production Trolley Software Functional Description	INT-406-VEN-1002
AD3	Network Message Protocols and Telemetry/Status/Logs File Format	INT-406-VEN-1007

Scope

This document forms part of the documentation for the preliminary production software design review. It describes the design and implementation of the shear sensor software and its context within the hardware and the delay line as a whole. Additionally, this document references the tip-tilt system on the delay line trolley because the two subsystems work together to correct delay line beam shear.

This document is very similar to RD2. Those readers who are familiar with that document may wish to concentrate on Section 4, which describes how the production shear camera software differs from the prototype.

For further information on the tip-tilt hardware, please see RD3. For further information on the tip-tilt software, please see AD2. For further information on the shear camera housing and conditioning optics, please see RD4.

Contents

1 Introduction

The delay line trolley introduces an optical delay into the science beam by transporting a catseye longitudinally through the delay line pipe. The pipe specification allows the pipe to have lateral excursions from its centreline of up to 5mm, hence as the trolley moves along the pipe the return beam will experience shear that changes as a function of trolley position. Natural subsidence of the pipe will also change the shear. If left unchecked, this shear will cause a reduction in the science beam fringe visibility and possibly loss of fringe counts by the metrology system.

A feedback loop has been devised to measure and compensate for changes in beam shear. A shear camera and computer determine a centroid for the metrology laser light returned from the delay line trolley, while the secondary mirror on the trolley tilts to change the shear. The two work together in real time to minimise shear.

This document describes the shear sensor and how it interacts with other components of the delay line. It is divided into two sections, which document the shear sensor hardware and the supporting software respectively.

2 Shear Sensor Hardware

The shear sensor hardware is illustrated in the delay line context in Figure 1.

Light from the metrology laser passes through a beam expander and the catseye on the delay line trolley. As the trolley moves along its pipe, irregularities in the pipe straightness cause the catseye to move laterally with respect to the incoming beam. This introduces shear into the return beam, which is compressed for use by the metrology system, but a beam splitter is also used to tap off part of the beam for shear measurement.

This beam passes into a custom camera housing via a commercial laser line interference filter that rejects ambient light, thereby improving the contrast ratio. It is then compressed a further 2.5$\times$ by a commercial beam compressor to produce a spot on the camera CCD surface, which is then digitised by the camera. For further information on these parts, please refer to RD4.

The camera itself is a Unibrain Fire-i BBW 1.3, which transmits monochrome 640$\times$480 pixel video to the shear computer via an IEEE1394a ("Firewire") interface and cable at 30Hz. The cable is 15m long (10m passive and 5m repeater cables joined) [1]. The camera itself dissipates just 1W, allowing the shear sensor computer to be placed

[1]The IEEE1394a standard states that the maximum length of a cable should be 5m. This is a limitation based on signal dispersion which can be overcome via appropriate manufacturing techniques. 10m long cables are now commercially available and no problems have been discovered in their use with repeater cables and the camera.

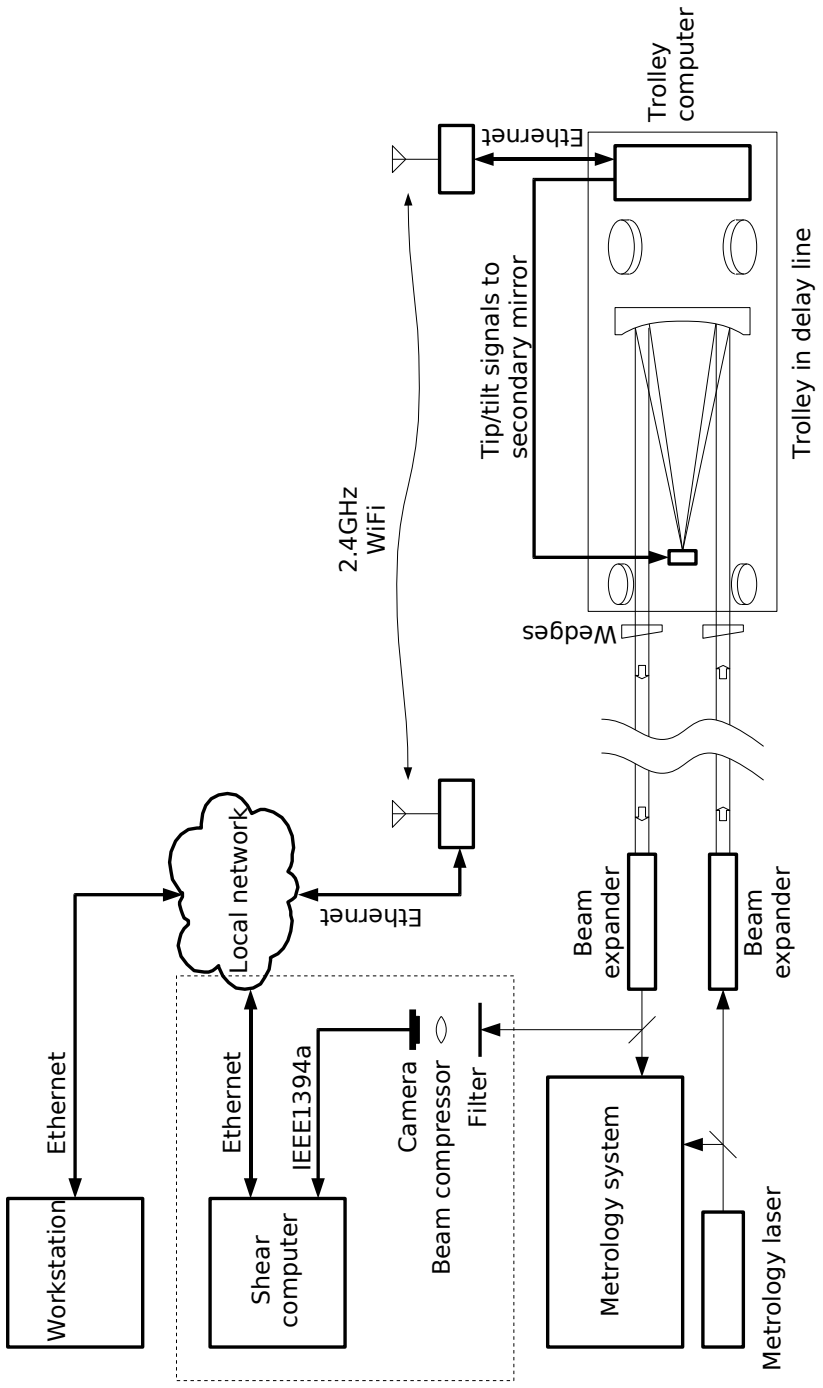


Figure 1: Cartoon of the shear sensor hardware (enclosed by the dotted line) in the context of the other interacting delay line subsystems.

in the outer beam combining area where it does not contribute to the inner beam combining area heat load.

The IEEE1394a transmission protocol is used because it is an open standard and hence is well supported by the shear computer operating system (Linux). It should also be possible to use other IEEE1394a cameras with the shear system should the Fire-i become unavailable.

The camera sends video frames to the shear computer, a rack mounted Eurotech Apollo, at 30Hz. This computer has a 1.8MHz Pentium M processor, 512MB of RAM and a 40GB hard disk. It also has a built-in IEEE1394 interface (one reason it was chosen) although a wide variety of interface cards in a standard PC would also be expected to work. For each frame, the computer calculates the position of the laser spot relative to a fiducial position using a centroiding algorithm (see Section 3). The result is then sent via ethernet and a Wi-Fi link (described in RD3) to the delay line trolley computer, where it is converted into analog tip and tilt signals for the piezo-electrically driven secondary mirror on the catseye. The effect of tilting this mirror is to change the shear of the return metrology beam such that it cancels the shear introduced by the pipe.

As the incoming science beam is parallel to the metrology beam and traverses the same optical system, its shear also gets corrected even though no science beam light is diverted into the shear camera. However, as the foci of the two beams on the secondary mirror are potentially the same there is a risk that scattered metrology light will contaminate the science beam. Hence wedges will be placed at the metrology input and output of the catseye to tilt the beam and shift the metrology focus position on the secondary. Again, this occurs without loss of science beam light.

3 Shear Sensor Software

Here, the software development and execution environments and the shear sensor program that are used to perform shear correction are described.

3.1 Development and Execution Environment

The target operating system used is Linux, chosen for the absence of licence fees, previous development experience by the authors and open source code availability. The development language is C because it is familiar to all programmers in the delay line team, although the shear sensor software is written in an object-oriented way to facilitate porting to C++ should that prove desirable.

The *glib* library is used as a framework. This library is more well known as the foundation for the *gnome* graphical user interface but can also be used independently,

as it is here. *Glib* sets up an event loop which can be interrupted by "watches", such as the arrival of a frame of video data, a packet on an ethernet port, or an internal timer signal. The use of *glib* in this way makes the shear sensor program fully event-driven.

Other libraries of interest are:

- *Libdc1394*, a high-level application programming interface for communication with IEEE1394 cameras via the linux kernel. Version 2 was released in 2008 and all prototype code has been ported over to use it.
- *LibX11* and *libXv*, to allow shear camera video to be displayed on a computer monitor in real time, locally or remotely.
- *LibVFlib3*, a font rasteriser, for printing text such as time stamps and centroid position on video images.
- *Libavcodec*, a video compression library, to reduce network bandwidth and storage use when video is logged to disk over the network.

Development and execution occur on a Debian Linux "testing" (currently "squeeze") system using the *gcc* compiler. However, the code should work with any Linux distribution that supports *libdc1394* version 2, and development could also be done on any similarly configured x86 computer.

Finally, the shear sensor computer clock is kept synchronised with the other computers in the delay line system using the NTP protocol so that telemetry data can be compared between systems. The shear sensor receives commands and sends real-time and archival data via a custom protocol developed in-house for this purpose. Video data is currently stored on the local hard disk when video logging is on, but will eventually be sent to the workstation for forwarding to the Interferometer Supervisory System (ISS).

3.2 Program Architecture

An overview of the shear sensor program architecture is illustrated in Figure 2.

On startup, the program loads a local configuration file that sets various parameters. These include an identifier string, internet protocol addresses and port numbers for communication with the workstation and trolley, video frame sizes and rates, whether video should be displayed, and a conversion factor from pixels to metres.

The program then initialises the sensor and sets up some network connection timers and a local video display window if one is required. Once initialisation is complete, the program enters the main event loop, which as described above is managed by *glib*. It waits there for various events to happen. The possible events are described in more detail below.

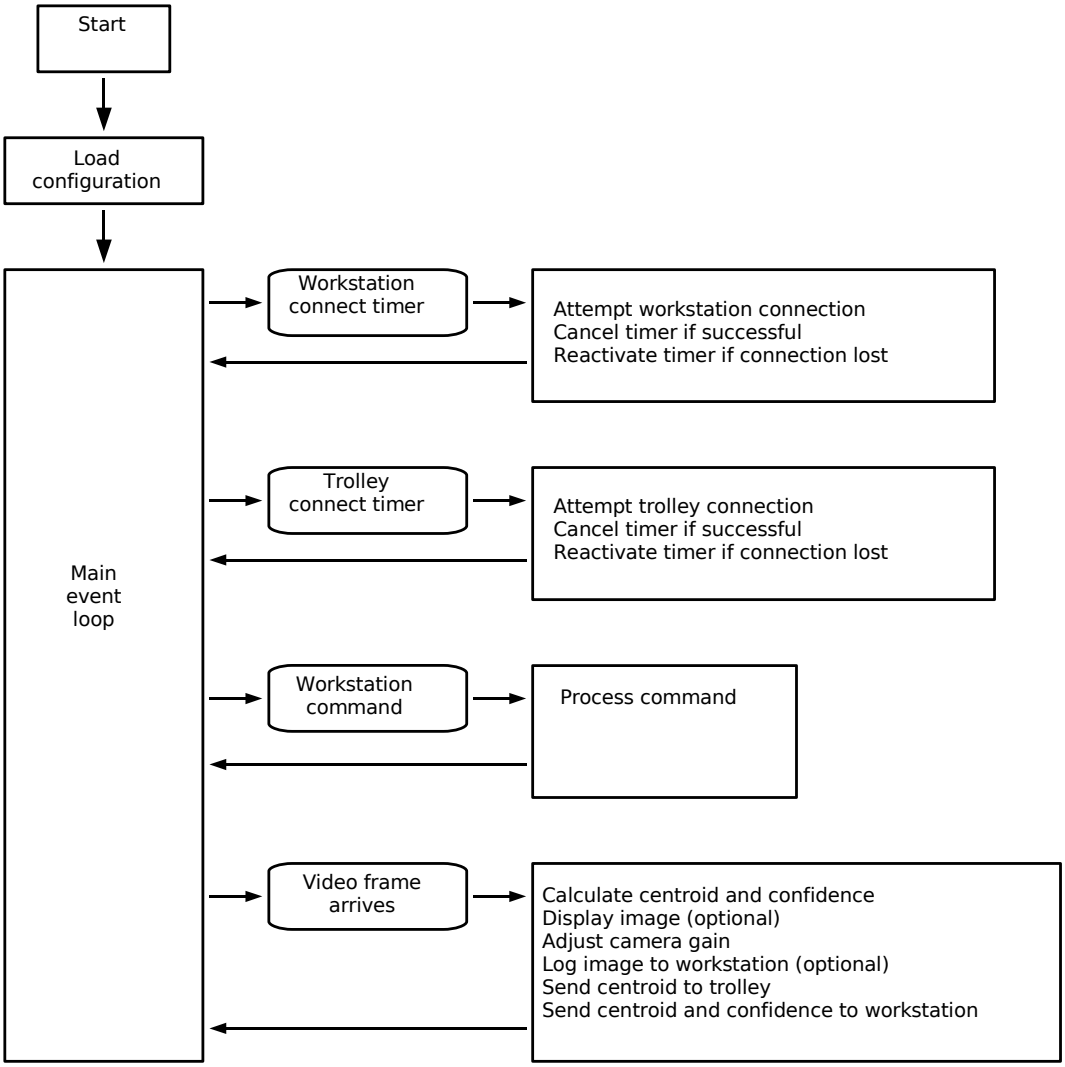


Figure 2: Overview of the shear sensor program architecture.

3.2.1 Workstation connect timer

During initialisation, a timer is set to cause an event once every five seconds (this value can be changed in the configuration file). On this signal the program attempts to initiate a connection to the workstation via the network. If the connection is successful, the program is able to receive commands from the workstation and the timer is cancelled. If no connection is made, further connection attempts will be triggered by the timer until one is successful.

3.2.2 Workstation connection failure

If the connection to the workstation is broken, perhaps by a network or workstation problem, this event causes the timer described in Subsection 3.2.1 to be reinitialised, so that the shear sensor will reconnect to the workstation automatically once the problem is fixed.

3.2.3 Trolley connect timer

The trolley connect timer behaves like the workstation connect timer, except that it attempts to connect to the trolley so that shear data can be sent there for correction of the shear.

3.2.4 Trolley connection failure

If the network connection with the trolley fails the timer in 3.2.3 gets reinitialised so that a connection can be reestablished once the failure is rectified.

3.2.5 Command arrives from workstation

When connected, the workstation can send commands to the shear sensor computer to change the behaviour of the shear program. When a packet arrives on the ethernet port it is firstly checked to see if it is from the workstation, is intended for this particular shear sensor, and is an allowed shear sensor command. If it passes these tests it is parsed and action is taken according to the command.

Commands are used to change the fiducial position, which is the target shear around which the loop is closed, so that small adjustments to the return beam shear can be made without moving the camera. They can also tell the program to save or to send every nth frame of video, or to stop doing so. This is all achieved by setting flags which are then tested and acted upon when video frames arrive, as described in Subsection 3.2.6 below.

3.2.6 Video frame arrives from camera

This signal occurs when data arrives on the IEEE1394 port from the camera. Most of the code in the program has been written to handle this particular event.

The data is firstly read from the port and reconstructed into a digital image of the metrology laser spot. A centroid and a "confidence" of the image are then calculated.

Care must be taken when selecting a suitable centroiding algorithm, as it must be immune to the effects of laser speckle. A variety of algorithms have been trialled, and a simple "centre of gravity" centroid has been found to be the most accurate, provided one modification is made: pixels in the lowest quartile of the pixel intensity range are ignored in the centroid calculation. This removes the influence of the darker but non-zero pixels visible around the laser spot on the calculation.

The confidence is a measure of how confident the shear program is that the derived centroid is correct. It is a measure of the jitter in the centroid position over the last few frames (the number is currently set to 30 frames), weighted by the contrast in the image. The purpose of the confidence calculation is to determine if the metrology beam has been blocked. If the confidence drops below a configurable value, the shear is calculated to be zero so that the trolley tip-tilt mirror stays in its current position rather than going to random positions dictated by shear camera noise. It also allows a rapid recovery if the beam dropout is momentary.

If the video display or logging options have been selected, the image is then annotated with the shear sensor identifier, a time stamp, and the calculated distance of the shear position from the fiducial position (the shear error). Two crosshairs are also drawn, one for the fiducial position and the other for the calculated centroid. These glyphs are rendered by inverting the most significant bit of each pixel value, so they are always visible regardless of the intensity of the original image. An example of an annotated frame appears in Figure 3.

If the ISS is available, this image (or a time series of images) can then be transmitted in a message to the workstation and forwarded to the ISS for storage or display using the message formats developed for those purposes.

For setting up or realignment the image can also be displayed locally or remotely using the Xwindows Xvideo interface. Due to the bandwidth required, best performance is achieved by using the local graphics hardware of the shear sensor computer itself, where it can be resized to fill an entire monitor if necessary. Remote display is also possible: in tests using computers equipped with 100Mbit ethernet cards on the Cavendish Laboratory's network, the remote frame rate was found to be about 10Hz and the latency about 0.5 seconds, still perfectly adequate for alignment and fault diagnosis and expected to improve further on the MROI's gigabit network. However, the reduced frame rate does slow down the response of the shear servo, so it is recommended that remote XVideo image display be turned off during normal operation.

Figure 3: Frame of logged video from the shear sensor program, showing the annotations. The large crosshair is the fiducial point, the small one is the calculated centroid. The diagonal line across the image is an artifact of the beam reducer and does not significantly affect the centroid calculation. In this case the servo is off so that the crosshairs are deliberately separated.

The Fire-i camera is equipped with automatic gain control, which adjusts the camera gain according to the average pixel intensity across the field. In the shear context, where most of the frame except for the metrology spot is dark, this causes the metrology light to saturate the detector. To overcome this, the camera's gain control has been set to manual and is now adjusted by the shear program to keep the brightest pixel in the (pre-annotated) image just below saturation so that the laser spot morphology remains clearly visible.

To minimise network bandwidth usage, the workstation can command that only every nth frame be logged and this will cause video image sequences to be decimated by n prior to transmission. As an alternative for when the ISS is not available, image sequences can be decimated, compressed and saved in MPEG2 format (the decimation factor need not be the same as for transmission). The resulting movie can then be replayed in most media players, and the time stamps allow the video to be compared with other delay line events logged by the workstation. After many

trials, an MPEG2 encoder has been found to offer the best performance and quality of compressed video but the format is encumbered by patents so an unencumbered alternative, such as Ogg Theora, may be adopted in future.

As mentioned previously, this video is currently saved to the shear computer disk but can be stored on the workstation using the NFS protocol. Using MPEG2 compression on video running at 30Hz, the data rate is about 100kbytes a second, well within the network capacity.

Finally, the shear computer sends the results of its calculations and optionally shear image sequences over the network. It sends the calculated shear error to the trolley, where it is processed and used to tilt the secondary mirror to minimise the error. It also sends the shear error, confidence values, and shear image sequences to the workstation, where they are either displayed in real time for the user and optionally logged to disk for testing and fault diagnosis, or they are forwarded to the ISS for similar treatment. Image messages transmit uncompressed video and hence will have a much higher natural data rate than the MPEG2-compressed video discussed above, but this bandwidth can be reduced through the use of image decimation.

Once all this is done, the program returns to the main event loop and waits for another event. As the video processing event is governed by the frame rate of the camera, it follows that messages are sent to the workstation and the trolley at this rate (30Hz). It has been estimated that a servo bandwidth of just 2Hz is needed to adequately close the tip-tilt servo while the trolley is tracking, and while slewing it only needs to function well enough to maintain metrology lock. The shear sensor servo is well able to perform both of these tasks.

4 Prototype–Production Differences

4.1 Hardware Differences

The shear sensor hardware is unchanged. However, the input beam now traverses an optical path similar to that of the production metrology system. In particular, only a small fraction of the light available to the prototype is available to the production system, because the metrology laser light has been split so that it services ten delay lines.

Figure 3 shows that there is still ample light available for shear determination. In fact, if the camera gain is increased (for example by using the camera's automatic gain control) then images can be made to saturate and details of the beam morphology are lost.

4.2 Software Differences

Software is also largely unchanged from the prototype. The main change (AD3) is a modification to the messaging protocol so that images as well as scalar data can be sent to the workstation for forwarding to the ISS.

The prototype relied on a pre-release version of *libdc1394* version 2. The official release occurred in 2008 and made some changes to the application programming interface. The production shear sensor software now incorporates these changes.