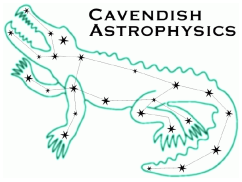
MRO Delay Line

Shear Camera Software Functional Description

The Cambridge Delay Line Team

INT-XXX-VEN-nnnn rev 0.1



Cavendish Laboratory
Madingley Road
Cambridge CB3 0HE
UK

Objective

To describe the design and implementation of the shear camera software.

Scope

This document forms part of the documentation for the delay line final design review. It describes the design and implementation of the shear camera software and also includes a section on the hardware used and its place in the context of the delay line as a whole. Additionally, it references the tip-tilt system on the delay line trolley because the two subsystems work together to correct delay line beam shear.

For further information on the tip-tilt hardware, please see the document “Trolley Electronics Design Description”.

For further information on the tip-tilt test software, please see the document “Trolley Software Functional Description”.

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

2 Shear Camera Hardware

The shear camera 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. The return beam is then compressed for use by the metrology system, but part of it is tapped off by a beam splitter for measurement of the shear.

This light produces a spot on a screen made of 0.38mm thick red nylon shim. Due to scattering within the screen, the spot appears on the other side as an approximately Lambertian light source, which is then imaged by the shear camera.

The camera itself is a Unibrain Fire-i BBW 1.3, which transmits monochrome 640×480 pixel video to the shear computer via an IEEE1394a ("Firewire") interface and cable at 30Hz. The cable is 10m long, allowing the shear camera computer to be placed in the outer beam combining area where it does not contribute to the inner beam combining area heat load.[1] The camera itself dissipates just 1W.

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

[1]The IEEE1394a standard states that the maximum length of the 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 the shear camera.

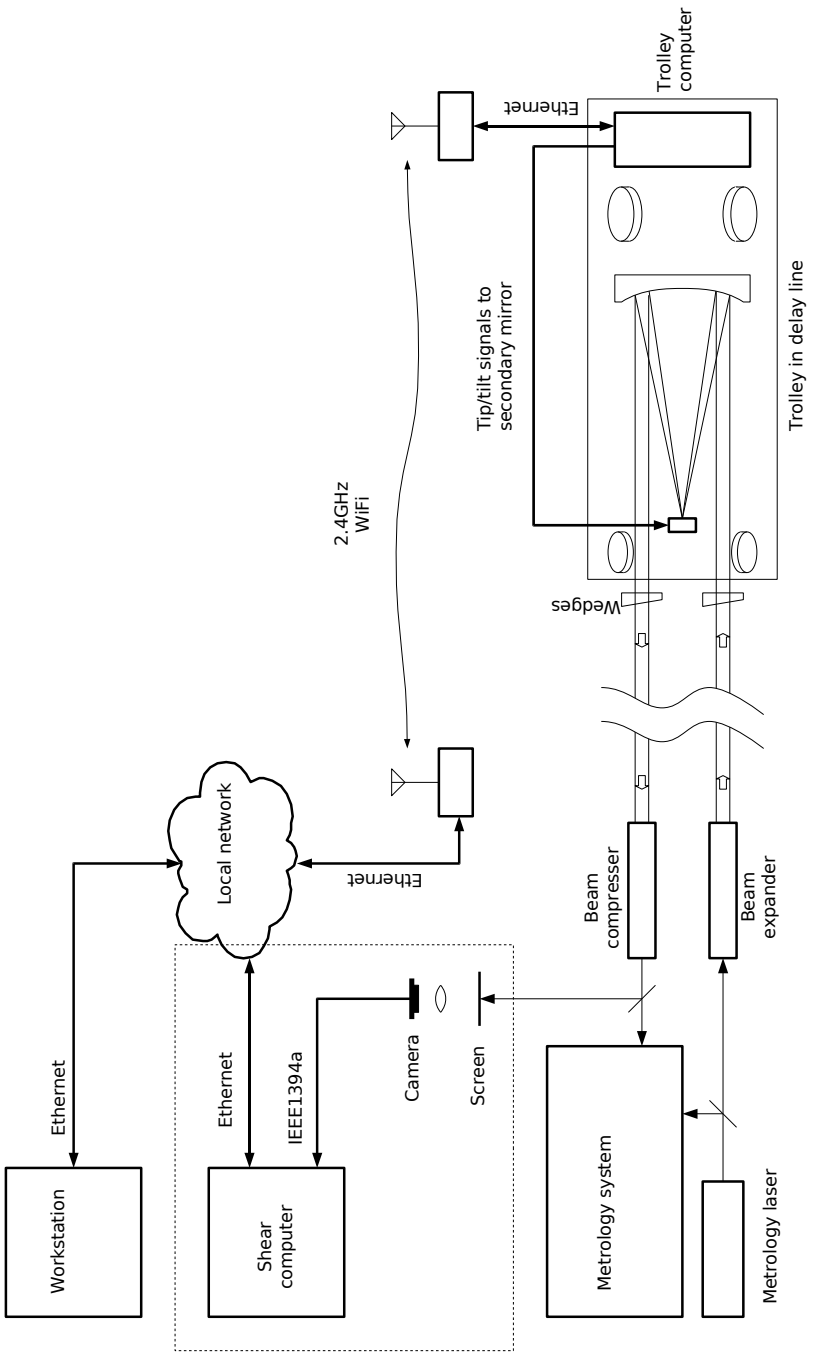


Figure 1: Cartoon of the shear camera hardware (enclosed by the dotted line) in the context of the other interacting delay line subsystems.

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 Arcom Apollo, at 30Hz. This computer has a built-in IEEE1394 interface (one reason it was chosen) but 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 to the delay line trolley computer, where it is converted into analog tip and tilt signals for the piezoelectrically 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 metrology light will contaminate the science beam. Hence wedges are 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 Camera Test Software

Test software has been written to demonstrate that the hardware is able to remove the shear from the returned delay line beam. Here, the software development and execution environments and the shear camera program that are used to perform this task 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 camera 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 camera 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. The official release is version 1, but as release of the much improved version 2 is imminent a version 2 release candidate has been used instead, as per the advice of the developer.
- *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 occurs on a Debian Linux "testing" (currently "lenny") system using the *gcc* compiler. However, the shear camera computer runs Fedora Core 6 linux, which by default does not include some of the libraries used by the shear camera software. Therefore these libraries are statically linked at compile time so that the program will run on a standard Fedora Core 6 linux computer without requiring installation of additional packages. The development and execution computers both use the Intel x86 architecture and the software contains no linux kernel code so an executable developed on one will run directly on the other.

Finally, the shear camera 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 camera 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 across the network to a file on the workstation using NFS.

3.2 Program Architecture

An overview of the shear camera 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 camera and sets up some network connection timers and a 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.

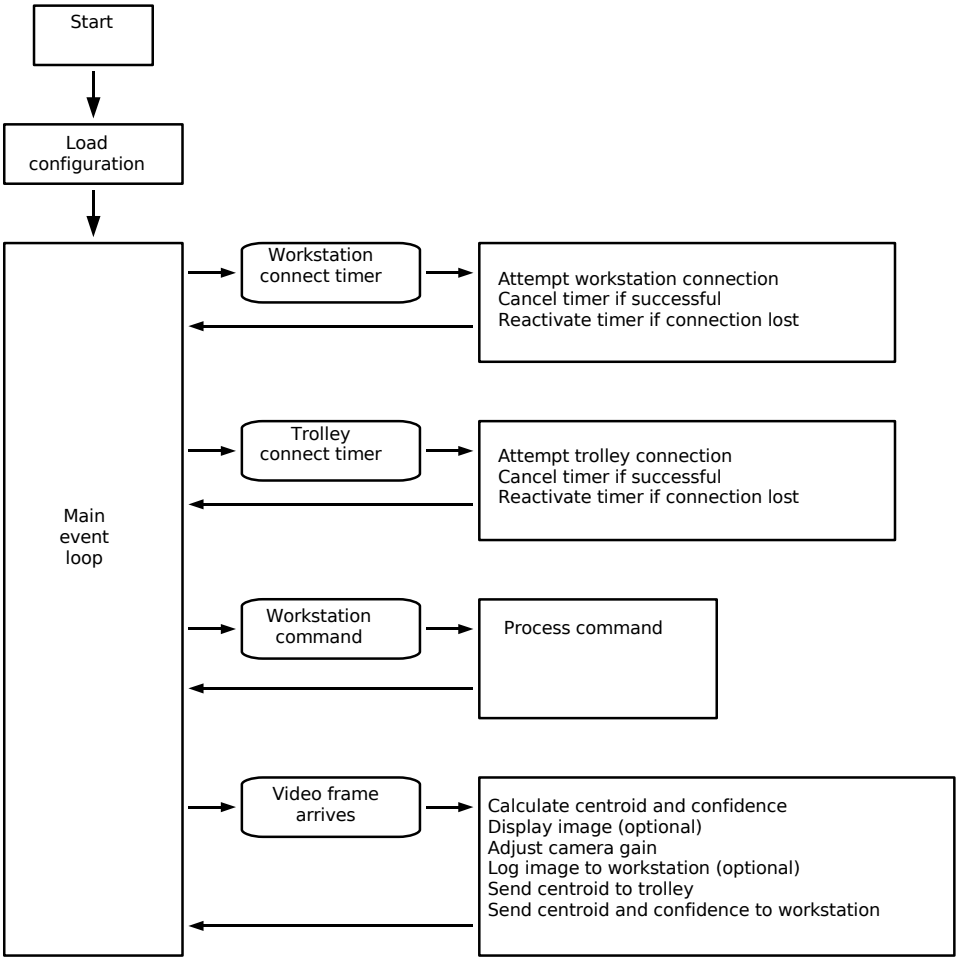


Figure 2: Overview of the shear camera program architecture.

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 camera 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 camera 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 camera, and is an allowed shear camera 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 also tell the program to save every nth frame to a video file (n is chosen by the user), 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. Additionally, the video logging command also causes a file to be opened for saving the video data.

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 on the screen. 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: the intensities of the brightest and dimmest pixels in the image are found and all pixels below the average of these 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, and works because the spot is symmetric.

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 configureable 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 diplay or logging options have been selected in the shear camera configuration file, the image is then annotated with the shear camera 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.

This image can be displayed locally or remotely. As it is rendered using the Xwindows Xvideo interface, best performance is achieved by using the local graphics hardware of the shear camera 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 image display be turned off during normal operation.

The Fire-i camera is equipped with automatic gain control, which adjusts the cam-

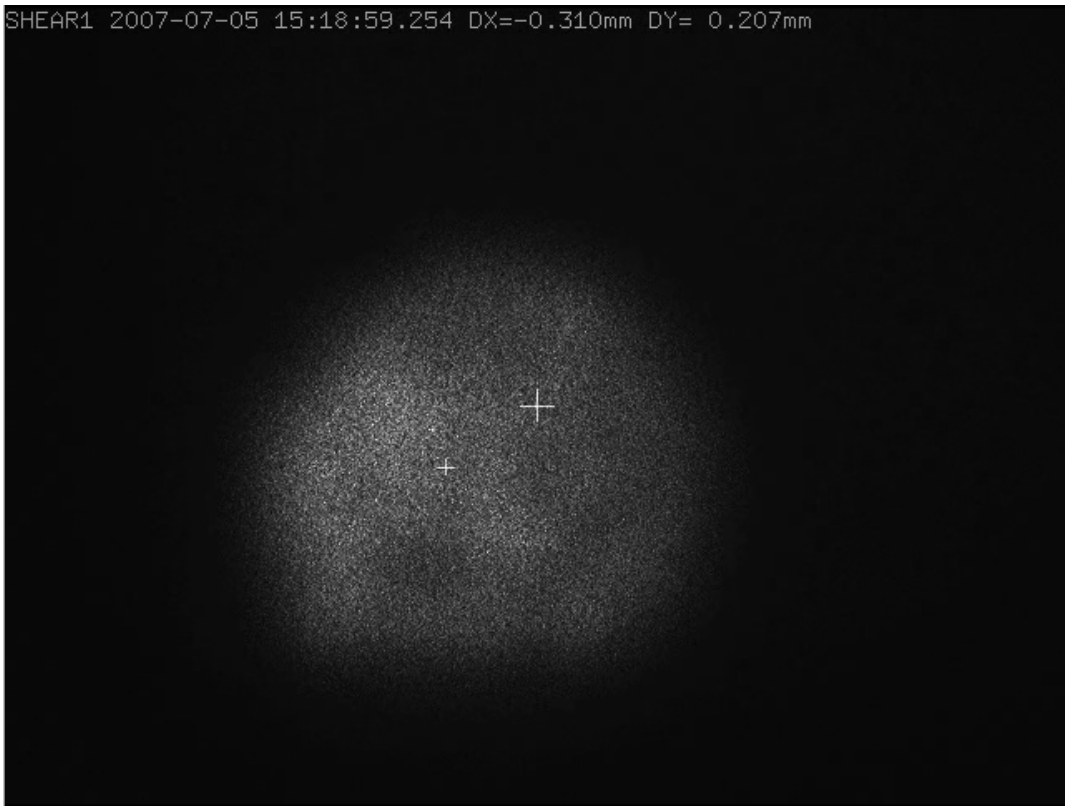


Figure 3: Frame of logged video from the shear camera program, showing the annotations. The large crosshair is the fiducial point, the small one is the calculated centroid. In this case the servo is off so that the crosshairs are separated, and the calculated centroid is to the left of the spot centre because of the brightness assymmetry in this particular example.

era 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.

If the workstation has commanded that every nth frame be logged, frames are compressed and saved in MPEG2 format. The resulting movie can then be replayed in any media player, 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 as the format is encumbered by patents an unencumbered alternative, such as Ogg Theora, may be adopted in future.

As mentioned previously, the video is currently saved to the shear computer disk but will eventually 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 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 and confidence values to the workstation, where they are displayed in real time for the user and optionally logged to disk for testing and fault diagnosis.

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 camera servo is well able to perform both of these tasks.