CONTROLLING CILEX-APOLLON LASER BEAMS ALIGNMENT AND DIAGNOSTICS SYSTEMS WITH TANGO

M. Pina, J-L. Veray, J-L. Paillard, B. Breteau, LULI, CNRS, Ecole Polytechnique, CEA, UPMC, Palaiseau, France

Abstract

Cilex-Apollon is a high intensity laser facility delivering at least 5 PW pulses on targets at one shot per minute, to study physics such as laser plasma electron or ion accelerator and laser plasma X-Ray sources. Under construction, Apollon is a four beam laser installation with two target areas. To control the laser beam characteristics and alignment, more than 75 CCD cameras and 100 motors are dispatched in the facility and controlled through a Tango bus. The image acquisition and display are made at 10 Hz. Different operations are made on line, at the same rate on acquired images like binarizing, centroid calculation, size and energy of laser beam. Other operations are made off line, on stored images. The beam alignment can be operated manually or automatically. The article presents the architecture, functionality, performances and feedback from a first deployment on a demonstrator.

LASER AMPLIFIER

Cilex-Apollon will deliver at least 5 PW, 75 J, 15 fs. To achieve these performances, the main laser amplification section is composed of five multi-pass amplifiers. Each amplifier stage has to be precisely adjusted to keep the energy at the nominal level. The beam passes through a TiSa crystal several times (Fig. 1). So the beam alignment needs to be done precisely. To correct the drifts caused by temperature and ensure the repeatability, an automatic alignment loop has been installed at each amplifier entrance.

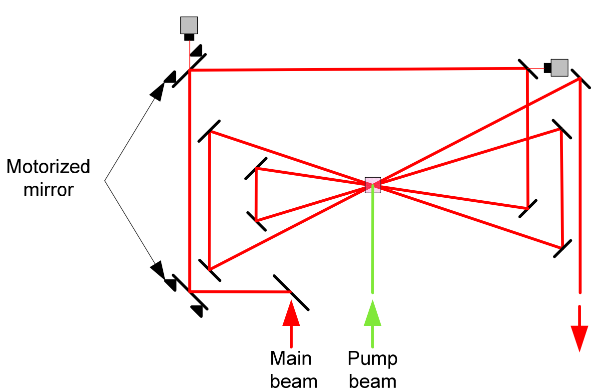

Figure 1: Laser amplifier stage.

MOTION CONTROL

Automatic Alignment Loop

A typical alignment loop is composed of two motorized mirrors and two cameras. The automatic mode is based on a close loop using a transfer matrix and can correct both centering and pointing of the laser beam. Eq. 1 defines the number of motor's steps to do, to correct the beam position.

$$\begin{bmatrix} Mx \\ My \end{bmatrix} = \begin{bmatrix} C11 & C12 \\ C21 & C22 \end{bmatrix} * \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \qquad (1)$$

Mx, My: number of motor steps on x and y (in steps)
Cii : motor coefficients for x and y movement (in steps/pixel). In the case of an orthogonal optical setup, C12 and C21 are equal to zero.
Δx, Δy: shift of the centroid beam relative to the reference (in pixels)
To improve the position accuracy while keeping a good response time, the automatic alignment loop uses the 2 motors speed. A position threshold permits to determine the limit to change the movement speed. With this function the system correct quickly the large errors and accurately the small errors.
To improve the alignment stability, a stability zone is defined around the reference point. When the beam is in this zone the system considers that the beam is aligned. The beam position used to determine the correction is the beam centroid which is calculated by the devCalcul Device Server (DS). Fig. 2 shows you the different parameters used to the automatic alignment:

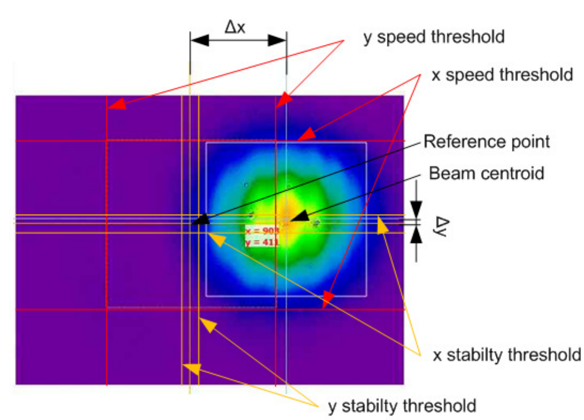

Figure 2: Automatic alignment parameters.

This alignment system corrects the beam position five times per second. In practice, this system is used to

ISBN 978-3-95450-139-7

correct the error due to the temperature, so it is a slow drift. That's why the most of time our alignment loop correct the beam position one time per second, to limit perturbation on the laser.

Manual Motion Control

Each motor installed on the laser chain can be controlled manually. The movements can be done by mouse click on a GUI's button or by a gamepad controller. This second method is very simple to use, low cost and compatible with many gamepads. The compatibility with different gamepads is done by the python library *"pygame"*. Thanks to this control method where the users can control simultaneously 4 motors at 3 different speeds (step by step, low, and high speed) for each, with 4 fingers. Moreover the users can change the controlled motors by the gamepad, with the GUI or the gamepad. It is very useful and appreciated by the users, mainly because now they can move the target on its three axes regardless the kind of motors. Fig. 3 shows a target movement on the three axes.

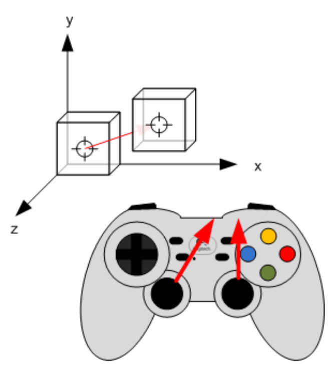

Figure 3: Manual movement of th target.

Note: Another advantage of this motion control is the mobility, by using wireless gamepads. So the users are free to move around the laser, within the respect of Bluetooth limitations.

ACQUISITION SYSTEM

Image Acquisition

The image acquisition works at 10Hz, the frequency of the laser to avoid losing data. So the cameras are trigged by an electric signal synchronous of the laser. On Apollon, two main kind of monochromatic camera are used; GigE cameras and FireWire cameras with different resolutions; 1600x1200, 1296x966, 640x480. Most of them are gigabit Ethernet cameras. To control these cameras, the LIMA library is used, a C++ camera library developed for TANGO. This library works with 16 bits image.

Image Calculation

The acquired images are used to diagnostic the laser beam. To extract the different information, a calculation software was developed to compute the beam centroid, beam barycentre, beam energy, beam size, binary image of the beam or an alarm if the beam centroid goes outside a circular zone which is defined by the user.

Image Server

The image acquisition and image calculation software run on the same PC to avoid network overload with the images exchange between these two software modules. In our architecture, dedicated PCs acquire images and calculate the beam characteristics. These PCs are named image server. Different configurations have been tested; the final one consists in an image server with three network interfaces: one for TANGO, and two others for the cameras. This solution is a good compromise between cost and reliability, because a server manages many cameras with minimal CPU load, and maximum use of Ethernet network bandwidth. This setup uses two networks to run up to eight cameras acquiring 12 bits images at more than 10 frames per second. With this solution, around 70% of the PC's CPU and Ethernet network are used. Fig. 4 represents the image server hardware architecture.

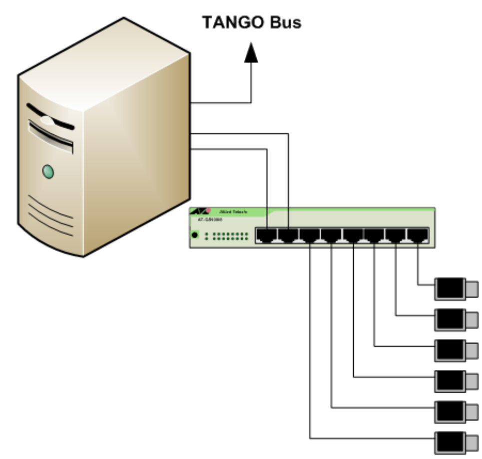

Figure 4: Image server hardware architecture.

SOFTWARE ARCHITECTURE AND GUIS

Software Architecture

The acquisition system and motion control system make an automatic alignment system. The Apollon command-control is based on the framework TANGO and developed in Python. This solution permits us to use several operating systems, like Windows or Linux. This functionality is essential for the alignment functioning, because Lima (library used for acquisition system) works on Linux and the PILMOT motion controller works only on a windows platform (due to the .dll library). Moreover TANGO ignores the material, so if you code the device drivers with the same framework you can run each kind of device with the same global architecture. For example in our alignment loop, a motion controller can be replaced by another one among the supported models (today three motion controller models are supported: RSAI PILMOT,

ISBN 978-3-95450-139-7

Newport Agilis and NewFocus 8752), without changing the architecture, just running the corresponding motor driver. Fig. 5 shows the software architecture of our alignment test system.

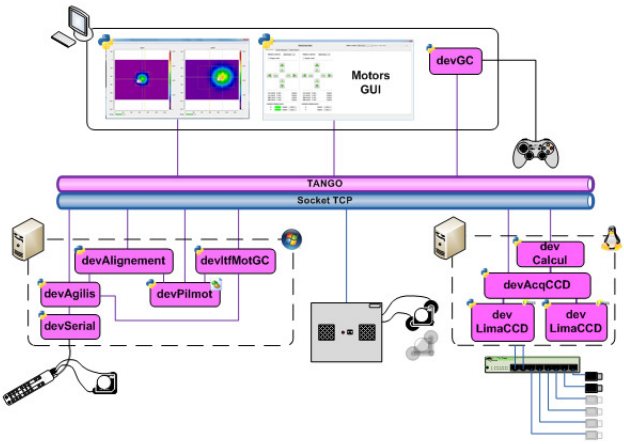

Figure 5: Automatic alignment software architecture.

A motor may need several drivers, like the Agilis controllers which need a communication driver (*devSerial* for a serial communication) and the motor driver (*devAgilis*). For the PILMOT driver the communication process is included in the dll. On motor server run the alignment driver (*devAlignment*) and the driver which do the interface between the motor drivers and the gamepad driver. The acquisition system has 3 level of driver: at the first level the camera driver (*devLimaCCD* for the Giga Ethernet cameras and *dev1394* for the FireWire cameras), at the second level the acquisition driver (*devAcqCCD*), this driver manages the image acquisition and put the images on the TANGO bus, at the third level the calculation driver (*devCalcul*). The supervision PC hosted the gamepad driver and the GUIs for acquisition system and the motion control system.

GUIs

Two GUIs are used for the alignment system; one for the motion control and one for the acquisition system.

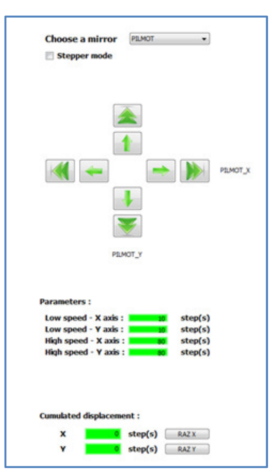

Figure 6: Motion control GUI.

The motion control GUI gives the possibility to control manually with the mouse, the motors declared in an xml file) (Fig. 6). Another part of this GUI offers the possibility to change the motor controlled by the gamepad, to modify some motors parameters and to control the alignment loops.

The acquisition system GUI displays on the main view, images of the declared cameras in an xml file. It's possible to get more information on an image by clicking on it. A new window (Fig. 7) gives the calculation driver's results, the beam profiles, the possibility to change the camera name and an access to the camera and calculation configuration (camera gain, alarm zone, binarization threshold…), in another window.

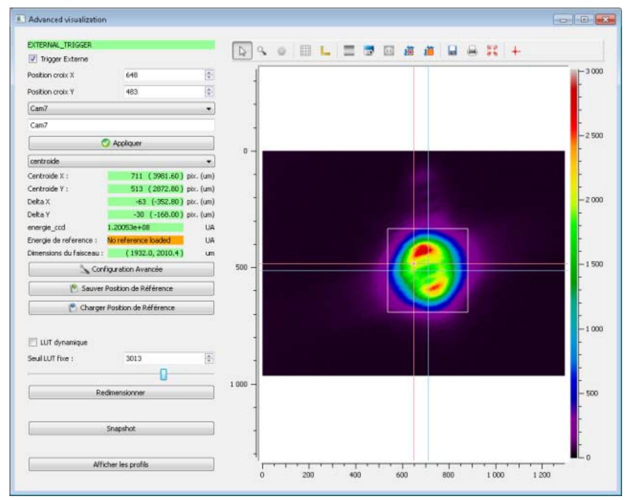

Figure 7: Advance image window.

CONCLUSION

An application using the distributed control system Tango framework in the domain of laser beam alignment has been presented. The setup has been validated with several types of hardware, cameras and motors controllers, both on Linux and Windows platforms. Interfacing new hardware will just need to develop the DS to communicate with the supplier driver. Up to now, this setup was deployed on the laser front end and then will be used in the different stages of the Apollon laser system. Some evolutions are considered such as speed improvement, more complex setup, and connection to the machine security system [1] to provide machine security functions trigged by results of image processing DS.

REFERENCES

[1] M. Pina and al, "Cilex-Apollon Synchronization and Security System", MOPPC045, these proceedings.

ISBN 978-3-95450-139-7