The Laser Appendix by: Thomas Moeslund

March 24, 1998

Chapter 1

preface

This tech-report is the documentation of a movable laser and the software which controls it. The laser is located at Center for PersonKommunikation (CPK), Aalborg University, Denmark.

The work presented here was done in the fall of 1997 as part of the multidisciplinary effort under the Multimedia and Multi-modal User Interfaces (MMUI) initiative.

Under this initiative the laser is used as an alternative output modality.

Currently it is used in two demonstrators developed by the work-team of MMUI; 'A Tour Guide System' and 'A Virtual Airhockey Game', and a student project within the MMUI initiative.

Both the documentation and code can freely be used if a reference to this tech-report is clearly stated. The code and documentation files can be found in /usr/local/Laser/ on the PC at CPK named **R2D2**

The physical set up of the laser and the low level control software is developed by Henrik Lausen from Laser Interface. Documentation about this stuff can be found in [1].

Chapter 2

Introduction

In every interface between a human and a machine, it should be possible for the machine to give feedback to the user. In normal interfaces images/graphics and sounds are used by the computer. When you want to design a new kind of interface, which is the case for the MMUI initiative, where the dependency between the human and the old modalities (keyboard, mouse and screen) should be broken, one need to find a new way for the machine to give feedback to the user. Therefore it was chosen to use a laser.

A contact was made to the Danish company, Laser Interface. An agreement was set up where the company delivered a laser system (laser, lenses, mirrors, controllers etc.) and the low level software to control this laser.

We, the workgroup within the MMUI initiative, should then develop a high level driver which made it possible to control the laser from another computer.

The laser is together with a speech synthesizer, used by the projects within the MMUI initiative to give response to the user. The laser can generate a dot which can be used to point with, it can draw shapes e.g. circles to indicate an area, it can draw a path from one location to another e.g. in a map application etc.

It is all together a very nice output modality to have when designing multi modal systems.

In the rest of this tech report the laser system designed by Laser Interface and the high level software driver to control it is presented.

2.1 Controlling the Laser

The laser is being controlled from a DOS PC, which only task is to control the laser. Therefore this DOS PC is again controlled by another computer where the actually program, which needs the laser, is running. The laser and the controlling PC are not designed to give feedback, making the interaction a one way communication.

This master/slave structure can be seen in figure 2.1 where the arrows indicate one way communication. For further details see [1].

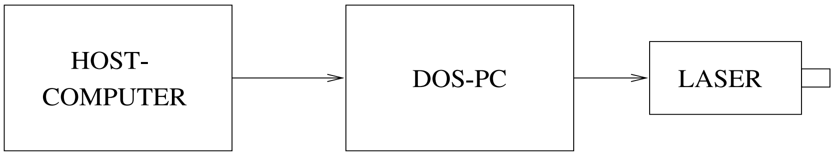

Figure 2.1: The master/slave control of the laser.

Since the laser is used to draw paths and dots at different locations, it must be movable. This is obtained by moving two small mirrors which reflect the laser beam. In figure 2.2 a schematic representation of the laser is shown.

The laser beam is generated in the laser diode and the more current this diode receives the more power the laser beam will have. This is known as the modulation of the laser. The diode is a red (640nm) laser capable of generating 15mW at 100% modulation. It is small (1cm) and cheap (500-1000$) but not very focused. Therefore the lenses are added to compensate for this. After the beam has been focused it hits the two small mirrors (5x10mm) which position determines where the laser beam ends up.

The control of the modulation and the control of the mirrors are extremely fast. You can scan 600 points at 50 Hz, which corresponds to drawing a path with 100 corners and updating it without the human eye noticing it.

However, the more points the lesser light pr. point, meaning that not too many points should be present at the same time since this will make it harder to see the laser beam. To solve this problem you can not just buy a more powerful laser, because the power spectrum of the laser moves into the inferred area as the power increases. This can be solved by buying a green laser, but then the price increases with a factor 10.

Another way of increasing the visibility of the laser is to lower the intensity of the surrunding light. If you do so you should be aware of the effect this can have on the image quality if a camera is involved.

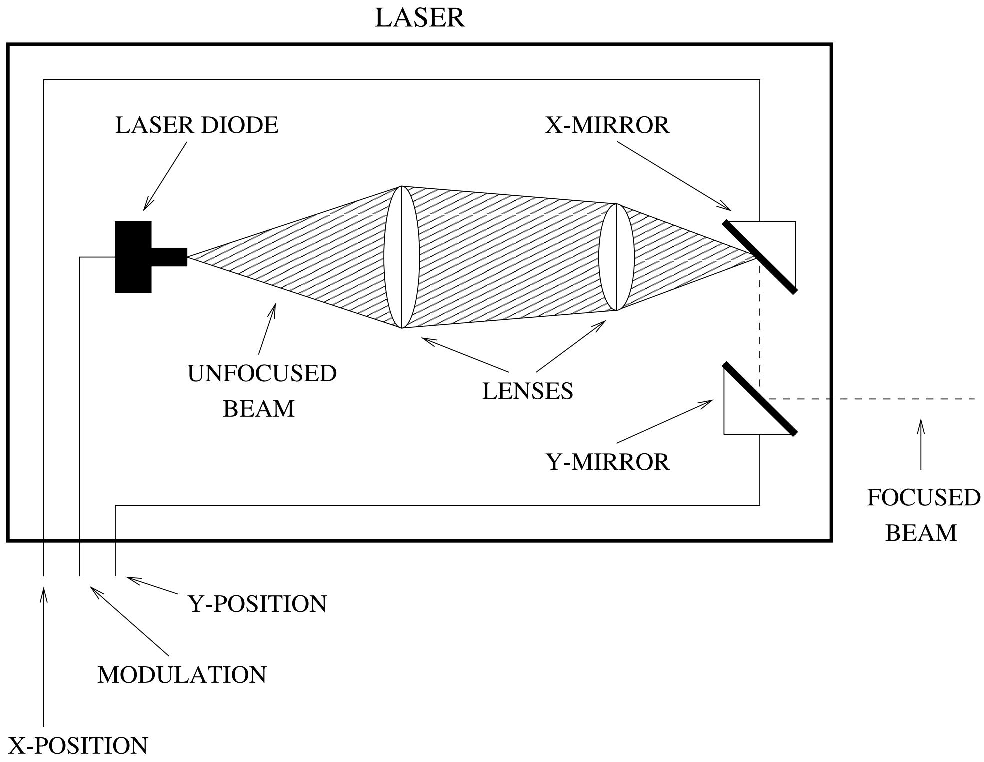

Figure 2.2: The internal structure of the laser.

2.2 Calibrating the Laser

Before the laser can be used by any system it must be calibrated.

The direction of the beam which is send from the laser corresponds to the angles of the two mirrors. There is a linear dependency between the different angles, meaning that the output will correspond to Cartesian coordinates if the laser is projected onto the inside of a sphere. This is usually not the case since the laser normaly is projected on to a flat table. Therefore the coordinates used by the laser will be distorted (depending on the distance to the focus point on the table) with respect to a standard coordinate system and some kind of correction algorithm must be made.

How this correction should look like depends on the application. When the laser is projected onto a flat table (pedendicular to the laser) the correction can be done using invers circle arcs.

The correction algorithm transform the laser's coordinate system, into a standard Cartesian system which can be calibrated by linear transformation.

The calibration also depends on the application.

2.3 The Serial Interface

The software to control the laser from the DOS-PC is develloped by Laser Interface, [1]. The software to control the DOS-PC is called the the serial interface and will be descriped in the following.

2.3.1 Introduction

The serial interface is used to remotely control the laser from a Linux PC. The functionality of the serial interface is to send commands and coordinates for controlling the laser. The commands which can be send to the laser controller are equivalent to the commands which can be issued directly from the user interface on the DOS PC. The communication between the serial interface and the laser controller is only one way so it is only possible to send information to the laser controller. A consequence of this is that the serial interface has to keep track of what is sent to the laser controller.

2.3.2 Layers

The serial interface consists of three layers of functionality: A C-API providing functions for controlling the laser, an error check and mirror layer which makes sure that coordinates sent to the laser are within the aloud range, and finally a communication layer which takes care of actually sending information to the laser controller. The different layers of the serial interface are shown on figure 2.3.

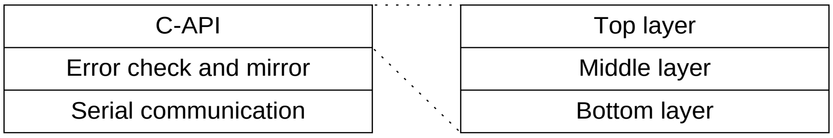

Figure 2.3: The layers of serial interface and the C-API.

2.3.3 Point and frame definition

When sending coordinate information to the laser it can be done either as single points or as frames. A point is a struct consisting of the following three integers:

```
- struct point {
-     int x; /* The x coordinate of the point */
-     int y; /* The y coordinate of the point */
-     int show; /* States whether the line ending in this point is shown or not */
- };
```

A frame in the serial interface consists of a number of points describing an image to be drawn using the laser. More specifically a frame is an array of points.

2.3.4 Functionality

The serial interface is implemented as a shared library programmed in C. The library is divided into three levels of abstraction as seen on figure 2.3:

Top level: The functions at the top level has the highest level of abstraction. When using these functions the caller does not need to produce frames before hand. The functions in this level uses a font library to send text and symbols (markings, pointers, arrows, etc.) to the laser. Currently the only function in this level is “Laser_send_text”.

Middle level: These functions are on a lower abstraction level as the caller should create frames before calling the functions. Currently two functions are present in this level: “Laser_send_frame” and “Laser_render_text”. The “Laser_send_frame” function transmits the frame making sure that the right commands are send before and after transmitting the coordinates. The “Laser_render_text” function adds text from the font library to a frame but does not send it.

Bottom level: These functions map almost directly to the protocol commands, refer to the laser manual. This is the lowest level of abstraction as the caller takes care of everything from sending the points to saving and showing the frame. This level also contains functions for making flow sequences (animation) and moving existing frames.

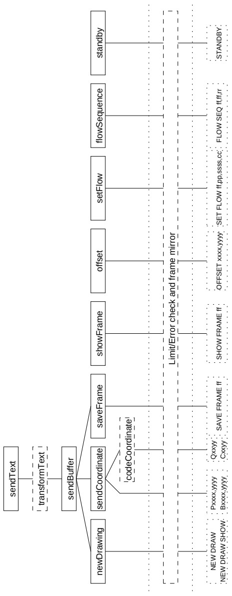

Figure 2.4: Serial protocol.

2.4 Protocol

The protocol used between the Linux PC and the DOS PC contains the following commands.

NEW DRAW The laser is set in draw mode and is ready to receive points to be drawn. The drawing is not shown while the points are transmitted.

NEW DRAW SHOW The laser is set in draw mode and is ready to receive points to be drawn. The drawing is shown while the points are transmitted.

Pxxxx,yyyy A new point. X=xxxx, Y=yyyy. This point is shown. X=[0;4095], Y: [0;4095].

Bxxxx,yyyy A new point. X=xxxx, Y=yyyy. This point is not shown. X=[0;4095], Y=[0;4095].

Qxxyy A new Point. X=xx, Y=yy. This point is shown. This command uses base 64. X=[00;oo], Y=[00;oo].

Cxxyy A new Point. X=xx, Y=yy. This point is not shown. This command uses base 64. X=[00;oo], Y=[00;oo].

SAVE FRAME ff Saves the current frame as frame number ff. The current drawing is finished. ff=[0;100].

SHOW FRAME ff Shows frame number ff. ff=[0;100].

STANDBY Sets the laser in standby mode. This means that the scanner is set in center position and the modulation of the beam is set to 0.

OFFSET xxxx,yyyy Moves the current frame to position X=xxxx, Y=yyyy. X=[0;4095], Y=[0;4095].

WRITE FILE ff Saves the frames in the buffers to a file. This does not delete the frames from the buffer. ff=[0;10].

READ FILE ff Reads a file of frames and puts them into the buffer on the DOS PC. ff=[0;10].

SET FLOW ff,pp,sssss,cc Sets a flow process for a frame. ff: frame number [0;100], pp: process number [1;16], sssss: flow steps or speed [1;65535] and cc: process curve [1;6].

FLOW SEQ f1,f2,rr Starts a flow sequence where all frames from f1 to f2 are shown in a sequence. rr states the number of times the sequence should be repeated. f1=[0;100], f2=[0;100], rr=[0;65535]. rr can be excluded and the sequence is only shown once.

REF.

[1]: LaserXI. Henrik Lausen, Laser Interface, CAT, 1997.

Preface