New Advances in Optical Design for Astronomical Spectrographs

A Thesis Submitted for the Degree

of

Doctor of Philosophy of the University of London

by

Alan Stuart Radley

Department of Physics and Astronomy

University College London

University of London

1996

This thesis is dedicated to Philip and Ellen Radley, without whose love and support it would not have been possible.

Abstract

The successful commissioning in 1992 of the Keck telescope heralded a new era in very large telescope construction. The Keck has a 10 metre collecting aperture, and today other projects to build very large telescopes are now underway. The Gemini project aims to build twin 8m telescopes, one for each hemisphere of the earth. The thesis work presented here deals with the optical design of high resolution spectrographs for telescopes of this size. Several different aspects of optical spectrograph design are investigated, including dioptric and catadioptric collimators and mosaic cross-dispersing prisms. Thesis content begins with a brief overview of some historical aspects of spectrograph system design. Subsequent work looks at techniques for establishing the optical feasibility of new generation spectrographs, and is largely concerned with finding viable camera arrangements. Towards this aim the author developed Inter-Opt, a new kind of optical ray tracing program. The Inter-Opt program comprises a user-interface that facilitates an interactive approach to the optimization of camera systems. This user-interface uses a new conceptual design methodology that enhances the efficiency of the dialogue between user and computer during camera design. Chapter 2 details the specific advantages of the new optimization techniques that the user-interface facilitates. Later Chapters deal with the optical design of a number of different spectrograph cameras for the Gemini High Resolution Optical/UV Spectrograph (HROS). Application of the Inter-Opt program to this camera study has demonstrated the feasibility of a camera for HROS at resolution 120,000. Additionally, in Chapter 6 the optical testing and commissioning results for a Short Schmidt camera are presented. Finally we conclude by looking forward to likely developments in optical spectrograph design in the future.

Acknowledgments

I would very much like to thank the following people for their help, support and friendship during preparation of this thesis: Francisco Diego, David Walker, Ron Szumski, Nigel Pugh, David Rees, David Brooks, Andrew Charalambous, Mark Dryburgh, Lee Hubbard, Paolo D'Arrigo, Bruce Bigelow, Sug-whan Kim, Ee-Eul Kim, Young-soo Kim, Heshmat Jamshidi, Gil Nixon, Barry Mohan, Trevor Savidge and Sarah Dunkin. My special thanks also to Ron Szumski, for giving me the opportunity to engage in insightful discussion concerning interactive ray tracing techniques. Thanks also to Nigel Pugh for helping me to get to grips with the concepts of object-oriented programming for Windows.

Glossary

AAT	Anglo Australian Telescope
BFL	Back Focal Length
BIT	Basic Interaction Task
CCD	Charge Coupled Device
CIT	Complex Interaction Task
CFHT	Canada France Hawaii Telescope
FSR	Free Spectral Range
FWHM	Full Width Half Maximum
EFL	Equivalent Focal Length
ESO	European Southern Observatory
HDS	High Dispersion Spectrograph
HIRES	The Keck High Resolution Echelle Spectrograph
HROS	The Gemini High Resolution Optical/UV Spectrograph
JNLT	Japan National Large Telescope
IAU	International Astronomical Union
IBM	International Business Machines
MMT	Multiple Mirror Telescope
MTF	Modulation Transfer Function
NA	Numerical Aperture
NOAO	National Optical Astronomical Observatory
OOP	Object Oriented Paradigm
OPD	Optical Path Difference
PC	Personal Computer
QIT	Quantify Interaction Task
QSO	Quasi Stellar Object

RAM	Random Access Memory
RMS	Root Mean Square
SARG	High Resolution Spectrograph for the TNG Telescope
SST	Spectroscopic Survey Telescope
TNG	Galileo National Telescope
UCLES	University College London Echelle Spectrograph
UKIRT	United Kingdom Infra-red Telescope
UV	Ultra-Violet
VDU	Visual Display Unit
VLT	Very Large Telescope
VR	Virtual Reality
3DIT	3 Dimensional Interaction Task

Contents

List Of Figures

List of Tables

Chapter 1

Recent Progress in Astronomical Spectrograph Design

1.1 Introduction

Astronomical observation has its roots in ancient history. To primitive man, the apparent regularity of celestial events was the only sure way to measure the passing of time. Early communal societies relied on these time scales to predict seasonal changes and so permit cultivation of crops. Ever since mankind has attempted to better understand the workings of the Universe through astronomical observation. Up until the seventeenth century, these efforts were limited to facts gathered by naked eye observations alone. Then in 1610 Galileo Galilei became the first person to view the heavens through a telescope. Galileo's spyglass revealed the night sky as never before, and the frontiers of the observable universe were at once extended by one hundred fold. Ever since that time, astrophysical knowledge has been provided almost entirely by telescopes of one type or another. Telescope designs have been continually enhanced throughout the intervening centuries. These improvements fall into three categories: wider spectral coverage, improved sensitivity and increased size.

Early telescopes saw the universe as it appears through visible light. This arose naturally due to the spectral response of the human eye with is only sensitive to optical radiation in the range 0.4 μm to 0.7 μm. Modern astronomical observations are obtained over far wider regions of the electromagnetic spectrum, from short wavelength gamma rays at 10^{-8} μm, through to radio waves with wavelengths up to about 1m. Despite the wealth of information provided by such broad spectral coverage, electromagnetic energy in the Universe is largely concentrated in the optical and near infrared regions. Improvements in telescope and detector technologies, for use specifically at these wavelengths, has therefore remained a primary concern. Semiconductor detectors have been especially useful in this regard, particularly the CCD (Charge Coupled Device). These devices can register almost every photon of optical radiation that fall on them, and are therefore ideal for study of very faint astrophysical sources. However CCD sensitivity is nearing a limit, and in order to further improve our ability to detect faint optical sources larger telescopes are required. As a result astronomers are now embarking on the construction of ground based optical telescopes with apertures from 8 to 10 metres [33][45][74].

Large aperture telescopes necessitate the building of larger associated instrumentation. Additionally, developments in active and adaptive optics technologies, now make improved imaging performance feasible for these telescopes [10][16]. These advances drive the development of new kinds of optical instrumentation. Novel optical configurations are required, and designing these systems is a difficult and challenging process. Today digital computers are commonly employed to facilitate the design of optical systems in general.

One of the first applications of digital computers in the late 1940s was in the field of optical design. Computers were ideally suited to performing the numerically intensive ray tracing calculations involved. Throughout the 1950s, the topic of automatic optical design using a computer was much discussed [4][78]. This led to the emergence in 1963 of a fully automatic design method using a computer, based around definition of a Merit Function [17][58]. Variants on this automatic method are still in common use today within a new generation of ray tracing programs. These modern programs take advantage of the recent dramatic increase in computational power, and are now capable of ray tracing speeds of more than 150,000 ray surfaces per second. As a result a great deal more ray tracing is now possible, allowing the optical designer to explore the properties of many more designs than in the past. Today programs also operate on computers with efficient graphical environments, and ray tracing programs are now becoming more user friendly. Overall these advances now provide the optical designer with enhanced analytical abilities with which to research novel optical systems.

In the realm of gamma ray and x-ray astronomy, the detectors have sufficient energy resolution to allow direct spectral analysis of incoming photons. At longer wavelengths however, it is necessary to split the incoming radiation into separate wavelength bands to allow monochromatic analysis. Broad and intermediate band spectroscopic analyses rely on the use of filters that pass light in standard bands such as the Johnson and Morgan UBV system [31]. Narrow band work concentrates on isolating spectral lines, and requires some form of spectroscope or more commonly a spectrograph to perform this task. In the next section, we discuss historical developments in the design of these instruments.

1.2 Historical Aspects of Spectrograph Design

In 1801, Thomas Young first demonstrated the wave nature of light through interference from a diffraction grating, and made the important link between colour of light and wavelength [79]. Joseph Fraunhoffer built on this initial work throughout the next twenty-five years, and introduced spectral lines as wavelength standards. He also used his skills as an optician to obtain and classify high quality spectra of the sun and brighter stars [20]. This foundation work led Gustov Kirchoff to discover the presence of sodium in the sun in the late 1850s [37]. Kirchoff and Bunsen would later announce the presence of other elements in the brighter stars, marking the emergence of stellar spectroscopy as a science.

Astrophysics progressed at a rapid rate throughout the mid 1850s, due in large part to the development of efficient spectrographs employing both slit and collimator elements. These early instruments composed all refracting optics, and used several prisms in order to achieve dispersion and isolate spectral lines. Often located at the Cassegrain focus of giant achromatic refracting telescopes, these spectrographs had a number of associated problems. Instrument flexure due to attachment to a moving telescope was a primary problem, in addition to the effects of temperature changes and the detrimental chromatic properties of spectrograph lenses. In the mid 1880s the introduction of the Coude optical configuration was heralded as a major advance, allowing a large spectrograph to be located at a fixed position [41]. This arrangement bypassed the instrument flexure problems of Cassegrain location, and also enabled location of a large spectroscope of high resolving power in a thermally controlled environment.

Fraunhoffer's early work used gratings to achieve dispersion, and grating spectrographs potentially offer a number of advantages over prism based instruments. Gratings provide linear dispersion, as well as dispersion in the red and lower levels of transmission loss over long trains of prisms. However early gratings were inefficient and not often used for stellar work until the development of blazed gratings that concentrate most of the diffracted light into first order spectra.

Up until the early 1930s, a key problem in spectrograph design emanated from the use of short focal length lens cameras. These catadioptric cameras provided very poor imagery for wide field photographic spectra. The invention in 1932 of the Schmidt camera offered a solution to this fundamental difficulty in spectrograph design, because this system provides superior image definition over a very wide field [53]. A modification of the Schmidt system replaced the short focal length lenses, and was found to provide excellent definition in the resulting stellar spectra.

With the development of the Schmidt camera, all the key elements of a modern spectrograph were in place by the late 1930s. However, not until the introduction of the echelle grating in the early 1960s did very high resolution spectroscopy of stars become commonly achievable. Before looking specifically at these echelle based spectrographs, it is important to overview the system design aspects for spectrographs generally.

1.3 Spectrograph System Design

A spectrograph consists of an entrance slit, collimator, dispersing element, and imaging camera. Figure 1.1 shows a typical grating spectrograph in schematic form. We can use this diagram to discuss the general design considerations of any spectrograph instrument. Light from the telescope enters the instrument through the slit. Ideally the slit width is equal to the diffraction limit of the grating aperture, but this condition is usually relaxed to maximize light transmission. A collimator aligns the light beam onto the dispersing element, in this case a grating. Once through the grating some form of camera images the spectrum into its component lines. In an echelle spectrograph this arrangement is different, for the spectral orders tend to overlap, and it is necessary to separate them in order to allow analysis. A cross dispersing element fits this purpose, commonly a prism or grating providing dispersion in the orthogonal plane. It is possible to arrange these basic system components in a variety of configurations. For a particular instrument finding the optimum form for a specific operating specification is the key goal. A number of researchers have reported on systematic methods to help the designer in this task [3]. Considering the spectrograph as a subsystem of the whole astronomical observing system is important, allowing emphasis on the design features which critically affect observational goals.

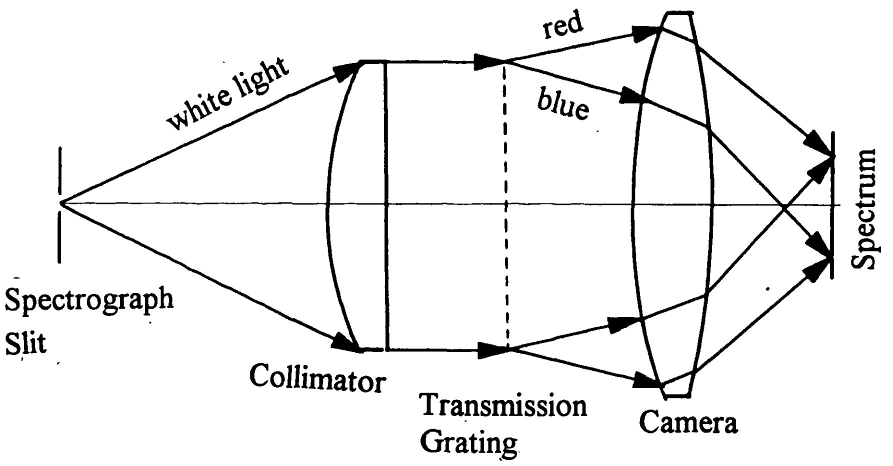


Figure 1.1 : Basic Grating Spectrograph Configuration

A fundamental parameter of any spectrograph is the intrinsic resolving power, and is defined as $\lambda / \delta \lambda$. Where λ is the observed wavelength and $\delta \lambda$ is the smallest detected spectral component. The starting point for system design is to choose the desired resolving power for a given nominal operating slit width. System design centers around meeting this primary goal, and often the process requires the designer to evaluate a number of potential system configurations using figures of merit. A general figure of merit for any spectrograph prototype is just the product of resolving power and entrance slit width. Another commonly used figure of merit is the product of the number of resolved spectral elements and entrance slit width, and is more concerned with camera efficiency over wide field angles. However these figures of merit do not allow direct comparison of overall instrument throughput, and fail to consider important factors such as efficiency of the system in different spectral regions. Quantitative comparison of the efficiency of different spectrograph designs therefore requires more sophisticated comparison methods. Before looking at these issues, we now define the standard equations that are the starting point in instrument design.

A standard equation for any spectrograph states that the projected slit width is dependent on the ratio of focal lengths of camera and collimator :

Projected Slit Width *(W)* = Physical Slit Width *(S)* * Ratio ..... **Equation 1.1**

where Ratio = Camera Focal Length / Collimator Focal Length

Another important equation shows that linear dispersion in any spectrograph depends on the focal length of the camera alone :

Linear Dispersion , $\delta x / \delta \lambda$ =

Focal Length Camera * $\delta\phi / \delta \lambda$ .....**Equation 1.2**

Where x represents linear dispersion in mm, λ refers to wavelength units, and ϕ represents dispersion angle in radians.

These basic equations indicate that higher resolution can be achieved by narrowing the slit (with corresponding lower signal to noise), using a CCD with smaller pixels, or using a camera of longer focal length. System design begins by specifying such basic characteristics as resolving power, slit width, detector characteristics, and overall instrument operating wavelength range. The design process then continues in an attempt to determine the optimum route to the desired system configuration that meets these criteria. The design process is an iterative cycle, which begins by comparing a variety of theoretical camera and instrument layouts. At each stage it is necessary to evaluate the different designs using a number of different criteria. See Appendix F, for a diagram introduced by the author to compare the efficiency of a variety of potential instrument configurations for the HROS instrument.

Often a multitude of different designs are possible for any particular specification. The situation becomes very complex, with design evaluation required on number of interdependent attributes simultaneously. For instance stray light can corrupt the images at the detector, and can arise in a number of ways. For instance at the grating or from scattering within the camera. Additionally some observations require exceptionally stable wavelength measurements. This requirement places tight tolerances on instrument thermal stability and mechanical rigidity. In general, spectrograph instruments are divided into two basic types. The first being spectrograph-like instruments [3], where a need for wide wavelength coverage exists. For this kind of instrument camera focal length is determined by considering the linear field of the imaged spectrum in the primary dispersion plane. Alternatively spectrometer-like instruments have a camera focal length that is determined without regard to the total width of the camera field. We now consider briefly the design of spectrometer-like instruments, in order to clarify the importance of a systematic approach.

It is useful to briefly consider the detailed design considerations for any instrument. For any spectrograph, the telescope aperture, focal length and hence focal ratio are known constants. Fixed also are the central wavelength of the detector, and the order of interference of the grating for that wavelength. Additionally the nominal grating characteristics are chosen, and assumed to be temporarily fixed. Also the linear dispersion (per unit wavelength) is fixed for this single design case. Five free parameters remain. These are R - the instrument resolving power, S - the nominal entrance slit width, W- the slit width image, and finally i + d and i - d. The latter parameters contain i which is the angle of incidence on the grating and d which is the angle of diffraction from the grating. To begin the design cycle R and S are fixed, and also a detector is chosen. Also, initially the sampling interval and the detector pixel size are used to calculate the required slit image size. The instrument monochromatic camera relative aperture can be expressed in a simple formula. The entrance slit width in radians on the sky is multiplied by the telescope aperture divided by W. The value of camera relative focal ratio can now replace the value of W in our set of variable parameters. At this stage a preliminary choice of grating is made through another parameter, which is 0.5 * (the blaze angle) * (i +d). This equation is only correct in the case of an echelle grating. This formula provides an initial value for grating width. See Figure 1.2 for a schematic of an echelle grating. Grating efficiency curves are used to select this blaze angle for a particular wavelength and groove spacing. Note the blaze angle of an echelle grating is simply the angle at which grooves are inclined for preferential reflection of light into one spectral region.

The choice of collimator-camera separating angle has to be chosen with regard to the spectrograph space limitations. However too large an angle implies aberration problems for long slit work. At this stage values for resolving power, slit width, slit image width, detector pixel size and sampling, grating width, grating blaze angle, and monochromatic camera focal ratio have been fixed. A value for i-d has yet to be fixed, but once fixed i and d provide the required apertures and focal lengths of camera and collimator. Determining values of i and d, the angles of incidence and refraction from the grating, is a choice dependent on the required angle between camera and collimator axes in order to prevent beam obstruction. Space constraints in spectrograph layout are important factors in this decision. Another issue impinging on the choice of i and d is the variation of grating efficiency for various values. Note that this changes according to known tabled values for any particular grating. Other issues to be considered are variations in magnification and dispersion when an instrument is used with a variety of different gratings. These factors may be significant, leading to efficiency losses due to different sized slit images and vignetting effects.

At this stage in the design process, the instrument has all five parameters fixed. The next step involves iteration through these stages a number of times with the various parameters altering in order find the optimum solution. We shall now consider two possible reasons for the first solution being untenable. Firstly if the value of grating width is larger than can be constructed, then the maximum sized grating should be placed into the equations and the spectrograph parameters recalculated. Another problem may be that the focal lengths may turn out to be far too long for the required instrument space envelope. A folded system can be constructed, otherwise the grating width or entrance slit width must be reduced.

Once the basic parameters have been fixed, the spectrometer prototype design can be tested on paper. This process requires evaluation of the preliminary design with different possible wavelengths, gratings, and optical camera arrangements. These calculations are done in order to give a feeling for performance in practice. In this section we have described the design issues for a spectrometer like system. In the next section we look at spectrograph-like instruments, and discuss the requirements and properties of a camera for such an instrument.

1.4 The Role of the Spectrograph Camera

The design of a spectrograph-like instrument begins by examining whether an adequate field is possible with the likely camera type. The role of the camera is to image the component spectral lines sharply throughout the desired wavelength range, without degradation of the line images due to the introduction of image aberrations. Various different kinds of camera can suffice for this purpose. However the most important feature of these cameras over imaging systems is that the aperture stop is located at the diffraction grating. Additionally in a spectrograph camera two aberrations are tolerated. Firstly chromatic difference of magnification is unimportant as it merely alters the dispersion slightly. Additionally in any spectrograph distortion is largely unimportant. A range of possible cameras must be tested for applicability to the design over the required field angles. Here we note that in an echelle spectrograph, with a cross dispersing element, the camera field is in the form of an Echellogram. See Appendix I for an example of an Echellogram from the UCLES instrument. In this case therefore, the camera must provide adequate imaging performance over a two dimensional format.

Evaluation of camera designs proceeds, and if no form can be found that satisfies the imaging criteria, we can again try altering the basic spectrograph parameters. One way of altering the basic parameters is to increase the camera focal length, slit width and linear camera field simultaneously. This can lead to cameras with smaller field angles in relative degree terms at the camera entrance pupil. Smaller field angles are associated with improved imaging in general. However note that the angle i-d must be altered during this process, to enable the larger aperture of the camera to fit into the required spectrograph geometry. If none of these options are feasible, then the basic parameters and performance of the instrument are reconsidered. We now briefly describe the kinds of steps involved this process.

We begin by placing an increased value of camera focal length into the equations of the previous section. We can now study the effects of such a change on the overall instrument performance. Increasing the camera focal length alone will reduce the relative angular field. Cameras that work satisfactorily with regard to aberration sizes, are often tenable over smaller field angles. Increasing the focal length of the camera for an instrument specification is however not a matter only of scaling whilst preserving slit width. The smaller field of such a camera sometimes allows use of a camera design with fewer elements, and so lower corresponding transmission losses. Improved light transmission may be the result for the instrument as a whole. Increasing the value for camera focal length has the effect that the apertures of camera and collimator may increase slightly whilst the entrance slit width falls. Increasing the angular camera field increases the value of N*S. Here N is the number of resolved spectral elements. This means N*S, the wavelength range, increases faster than the slit width decreases. A gain in photons covered at different wavelengths occurs. However this advantage requires careful comparison against those photons lost due to a smaller entrance slit. Attainment of the desired wavelength range leaves N as a constant (filled detector case). Further reduction of the slit width continues as camera focal length increases. A compromise occurs, where one optimizes the solution of shorter camera focal length and wider entrance slit, against longer focal length and improved camera efficiency due to better slit images. Note here that for an echelle instrument, blaze angle falls as camera focal length increases, altering instrument geometry slightly.

This discussion has demonstrated that a large family of possible instrument designs must be investigated in order to locate the optimum solution for any spectrograph. Optimizing the entire system requires that a number of factors to be taken into consideration. It is necessary to asses the relative merits of each instrument option on a range of criteria. These criteria include instrument throughput, wavelength coverage, likely affects on integration times, and instrument geometry. The implications of different types of camera design on overall system performance must be fully taken into account. Plotting the various options on a decision tree, (see Appendix F), will help asses the different routes possible. Camera design has implications on the design of completely separate parts of the telescope-spectrograph system. For instance a particular camera design may require a smaller slit width, indicating the need for an image slicer to improve instrument throughput. Development of a new instrument concept relies heavily on the feasibility of certain camera designs. Camera development is an integral part of spectrograph design as a result. Spectrograph design for larger telescopes requires a specific design approach, and in the next section we introduce some of the issues involved.

1.5 Spectrograph Design for 8m Class Telescopes

Considering telescopes of the same focal ratio, one with an 8m aperture will have a focal length twice that of one with only a 4m aperture. This means that the plate scale in the focal plane of the 8m telescope will be smaller. As a result a star image will be physically doubled. The slit must also be twice as wide as a result. Equation 1.1 indicates that it is necessary to halve camera focal length in order to retain the same physical slit image on the detector. Yet from the linear dispersion equation we can see that by doing so we reduce the linear dispersion of the spectrograph. In order to retain the instrument resolution, the entire instrument requires scaling by a factor of two. This implies a very large beam size for such instruments.

Large beam spectrographs of this type present a number of problems, not the least of which is the requirement for a short focal length camera with very fast focal ratio. One way to avoid a very large beam size is to place the instrument on a telescope with a faster focal ratio. Placing a spectrograph at a Cassegrain station therefore allows for a reduced beam size over a giant Nasmyth instrument that would be the alternative.

A primary task of the new generation of very large telescopes will be to probe the Universe at high spectral resolution. In the next section we briefly discuss the scientific goals of modern high resolution spectroscopy.

1.6 The Scientific Case for High Spectral Resolution

New scientific advances are predicted by combining the light gathering power of a large aperture with the methods of high resolution spectroscopy. An improvement in the quality of research will result, enabling the chemical composition, abundance and structure of astrophysical objects to be known to greater accuracy. Unsaturated weak lines are most often used for such abundance determinations. High spectral resolution is necessary for the isolation of these weak lines from multiple blends.

Much remains to be learned about the chemical enrichment history of the Galaxy and wider Universe. High spectral resolution is necessary for study of weak lines in the absorbing clouds along the line of sight of Quasi-Stellar-Objects (QSO).

This type of observation will allow study of the evolution of the early Universe. For example, metal line absorption spectra from Quasars will allow study of the dynamics, ionization state and abundance of clouds in galactic halos [63]. Additionally a great deal remains to be learned about the nature of stellar atmospheres. High precision study of spectral line profiles will enable analysis of the dynamics of stellar oscillations, the hydrodynamics of stellar atmospheres, duplicity and rotation of stars, and stellar evolution. Additionally astroseismology requires extremely high precision radial velocity determinations for stars on the order 3-4 cm per second. These kinds of observations allow the search for planetary companions and circumstellar discs around bright stars.

Improvements in spectroscopic analysis can therefore help us understand fundamental questions about the Universe and its workings. A key goal of current large telescope projects is study of astrophysical objects at high spectral resolution. Meeting this goal requires the building of a new generation of high resolution spectrographs. The echelle grating remains the basis for these instruments, and in the next section we discuss the general characteristics of these kinds of spectrographs.

1.7 Echelle Grating Spectrographs

Gratings disperse light using a combination of diffraction and interference, and provide intrinsically higher dispersion than is possible with prisms alone. The method of dispersion employed is called division of amplitude. The effect is also known as Fraunhoffer dispersion after Joseph Fraunhoffer who first experimented with the technique in the early nineteenth century [20]. Nearly one hundred and fifty years later, a unique type of grating was proposed by Harrison [26]. This is called the echelle grating, and operates at high orders of interference. It has been used in recent times to obtain resolutions greater than 400,000. Importantly for use in Astronomy, an echelle grating has three times the throughput of the classical grating. The echelle grating has wide shallow grooves, and is designed for incidence angles greater than 45 degrees. Figure 1.2 shows the groove geometry of a standard echelle grating.

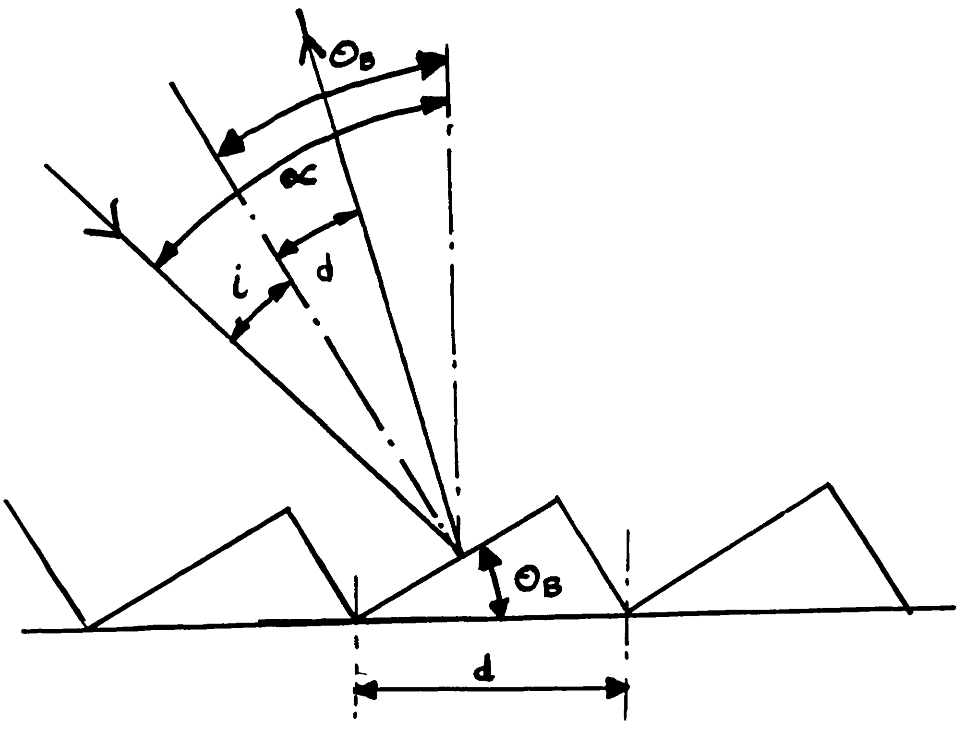


Figure 1.2 The Groove Geometry of an Echelle Grating

The first echelle spectrograph ever built for astronomical use was completed in 1967. This instrument was fixed at the Cassegrain focus of the University of Wisconsin 0.91m telescope [54]. This instrument was used to study semidiffuse interstellar features, by observing early type stars down to magnitude 7. In the early 1970s several other Cassegrain based instruments were constructed at various sights around the world. Notable among these is the ESO CASPEC instrument, which is still in operation today. This spectrograph has recently been upgraded with a new 512 by 512 pixel CCD detector. These improvements now allow observations of faint objects towards the blue end of the spectrum, successfully registering lines as far blue as 3130 Angstroms [46].

Towards the end of the 1970s, an echelle spectrograph was built for location at the Cassegrain focus of the 2m San Paulo telescope in Mexico. Chapter 6 of this thesis describes the design of a replacement short Schmidt camera for this spectrograph, in order to overcome a design fault present in the original camera arrangement. Throughout the 1980s a number of echelle spectrographs were built and located at Coude stations. The Hamilton echelle spectrometer is a notable example, fixed in position at the Coude focus of the 3m Shane telescope at the Lick observatory. This instrument is capable of 50,000 resolution for objects as faint as 16.5 magnitude [64]. Additionally, the continued success of the UCLES spectrograph on the AAT 4m telescope in Siding springs, Australia, has demonstrated the efficiency of prism cross dispersion [11]. The successful completion in 1993 of the HIRES spectroscope, located at the Nasmyth foci of the giant 10m Keck telescope, began a new phase in echelle spectrograph design. This instrument is the first of a new generation of spectrographs for very large aperture telescopes.

The optical design of an echelle spectrograph for use on an 8m-10m class telescope presents a number of new challenges. Largely, these difficulties arise from the new scientific goals for these projects. These aims require that these instruments simultaneously provide spectra from the near infrared region of the spectrum through to the near ultraviolet [71][72]. New highly efficient optical designs are now required as a result. Importantly these designs should offer both high transmission and highly achromatic properties throughout this broad spectrum. The next section presents an overview of thesis work completed by author to investigate the feasibility of these new instruments.

1.8 Thesis Work

This Chapter has presented an overview of recent developments in spectrographic instrumentation for ground based astronomy. Subsequent Chapters deal with the new optical design methods required to build high resolution spectrographs for 8m class telescopes. The thesis work presented here has been completed by the author with the aid of a three year Access Fund Postgraduate Scholarship. During this period the author worked within the Optical Science Laboratory, a research group within the Department of Physics and Astronomy at University College London. The author also received support in the form of a three month studentship from this department.

The Optical Science Laboratory continues to work on the detailed design of the High Resolution Optical/UV Spectrograph (HROS) for the Gemini 8m telescope project. During the last two years this group has completed a number of preliminary design studies for this instrument [49][68][72]. A significant part of the work presented in this thesis was completed towards the goal of demonstrating the optical feasibility of this instrument.

Chapter 2 begins by discussing the development, and testing of a new approach to optical camera design. This Chapter describes the capabilities of an interactive ray tracing program developed by the author. The program comprises a graphical user-interface developed to enhance the communication, or dialogue, between the optical designer and computer during system optimization. This user-interface enables the designer to visualize, and participate in, the real time solving of specific problems during camera optimization. Chapter 2 begins by introducing and classifying the basic constructs of the new design methodology. Detailed aspects of the conceptual and functional design of this user-interface are also presented. In addition the capabilities and efficiency of the program in practice are detailed. Presented in this Chapter also is optimization data in order to demonstrate the program's specific features during the design of Schmidt and Houghton type spectrograph cameras. Ray tracing validation results are given in Chapter 2, in order to demonstrate the accuracy and precision of Inter-Opt's results for two different optical systems. Ray tracing data for these systems is compared with that supplied by the Zemax [81] software and additionally a standard optics text book [56].

The capabilities of the interactive design methodology, and its application, are described by the author in two papers. The author presented a paper at the International Symposium in Optical Science, Engineering and Instrumentation, in 'Novel Systems Design and Optimization' on July 10th 1995 in the San Diego Convention Center [47]. This paper is entitled 'Interactive Optimization of a Novel Sub-aperture Spectrograph Camera' Additionally the author described the new Software's features at the 'Optical Designers Meeting' held at Imperial College on the 21st April 1995, where a paper was presented on 'Interactive Spectrograph Camera Design'.

In Chapters 5 and 6 we describe the successful application of the Inter-Opt software to the optical design of a number of cameras. Chapter 3 introduces the overall design philosophy for the HROS instrument, and demonstrates the feasibility of employing a mosaic prism cross dispersing element within an echelle spectrograph of this type. This work involved the writing of two 400 line computer programs in the C language, in order to calculate the light losses incurred at prism boundaries. Conclusions were subsequently incorporated into a paper presented at the proceedings of the E.S.O. Workshop in High Resolution Spectroscopy [69]. Chapter 4 describes the results of a lens collimator feasibility study. The properties of both achromatic and apochromatic lenses composed of a variety of UV transmitting materials are considered in this Chapter. Additionally this Chapter deals with establishing the feasibility of a Cassegrain location for the High Resolution Optical/UV Spectrograph (HROS) from the perspective of reflecting collimator aberrations.

Chapter 5 presents the results of an optical camera design study, in order to demonstrate the feasibility of the HROS instrument at a resolution of 120,000. In this Chapter we contrast the properties of a number of camera designs, including a unique off-axis design which has an unobscured pupil. This work was incorporated into an internal design study for the HROS project [49]. This research was also presented at the International Symposium for Astronomical Instrumentation for the Twenty First Century, where the author was co-author of a paper presented on 'A Re-examination of High Resolution Spectrographs For Large Telescopes - The Cassegrain Solution' [70]. In Chapter 6, a description of the optical design and testing results for a Short Schmidt camera of f-number 1.6 are presented. Finally in Chapter 7, we summarize thesis work and look forward to future developments in the field of high resolution spectroscopy. The next Chapter discusses the advantages of an interactive approach to optical design, and additionally presents the capabilities of a new type of ray tracing program.

Chapter 2

Development of an Interactive Optical Design Program

2.1 Introduction

This Chapter describes a computer program developed by the author for spectrograph camera optical design. This software tool provides unique ray tracing features in order to facilitate interactive design of camera systems. The next section details with the advantages of interactive optimization as compared to the automatic technique. Later parts of this Chapter describe the software's new design methodology and specialized features in detail.

2.2 Computer Aided Optical Design

The modern optical engineer makes extensive use of computer ray tracing aids. During the last fifty years a number of new developments have occurred in the field of optical computing. These advances have impacted the engineer's job in different ways, but perhaps most notable has been the introduction of automatic optimization methods. In the next section we briefly review automatic optimization.

2.2.1 General Principles of Optical Design

An optical system is conceptually simple, consisting usually of spherical optical elements with specific separations. The designer's task is to find the optimum materials, shapes, and separations of the various elements. A well-corrected system is identified by comparison with paraxial imagery. The paraxial region is imagined to be infinitesimally close to the optical axis, where mathematical ray tracing is defect free. Aberrations in the image-forming characteristics of optical systems occur in lenses of finite aperture and fields of view, well outside this paraxial approximation. Two basic aberration types exist, geometric and chromatic aberrations.

Seidel first investigated aberrations in a systematic fashion and developed an aberration polynomial to quantify their contributions to overall image defects [8]. In this equation, first order properties equate to paraxial imaging. Additionally third order or primary aberrations are present and exhibit a cubic relationship with ray height from the optic axis. These primary aberrations are known as spherical aberration, coma, astigmatism, petzval field curvature and distortion. Higher order aberrations are also predicted by this equation, and often must be taken into account during system design [8]. Additionally in the case of a very highly corrected optical system, with tiny levels of residual geometric aberrations, the system may be diffraction limited. This refers to the fact that real systems exhibit a fundamental limit in imaging performance, due to the wave motion of light. As a result the finite aperture of an optical system has a diffraction limited blur spot size. Correction of geometric aberrations beyond this limit has no effect on image quality. The diameter of the airy pattern characteristic of this limit is given in Equation 2.1.

$\beta = 2.44 * \lambda * F/\#$ ....Equation 2.1

Where β = Diffraction Limited Blur Spot Diameter in Radians,

λ is Wavelength in Metres, and

$F/\#$ = Focal Number or Equivalent Focal Length / Aperture

A simple photographic triplet may have as many as 10^8 possible configurations, and most will exhibit poor imaging characteristics. Finding a design with acceptably small levels of aberrations is a difficult task. Knowledge of aberration theory is required to enable the designer to adopt a systematic approach to optical system design. A well-corrected design exists in a region of the parameter space where low levels of third order aberrations are present. This greatly restricts the search for an acceptable solution. However a balance between lower and higher order aberrations is often required in order to meet the imaging specification. Optimization therefore involves a large number of complex relationships, and is seen to be a highly non-linear process. Optical designers use a combination of analytical and empirical techniques to find a suitable lens design for a particular task. These methods require a considerable amount of optical ray tracing, which is computationally intensive. As a result computers are employed as ray tracing aids. During the last twenty years computing power has increased enormously, and designers can now optimize optical systems far more quickly than in the past. Additionally, with the emergence of automatic optimization methods, the designer is now able capitalize on the power of the computer to explore many hundreds of different designs. We now look briefly at the basic theory behind one such method, termed Merit Function optimization of optical systems.

2.2.2 Automatic Design Using the Merit Function Method

Automatic optimization techniques drive an optical design towards a desired optimum. At present these programs require that certain characteristics of this optimum be specified in advance by an optical engineer. A defect or Merit Function can then be calculated for any real system, representing in a single number the worth of a design with respect to the optimum form. This number comprises the sum of the squares of the image defects, and usually also includes physical characteristic defects. The latter types of defect are calculated from ideal targets for specific attributes like component apertures and separations, and allow control of such characteristics. Both kinds of defects are quantified during Merit Function calculation for any optical system, simply by evaluating how much greater they are than a specified target value. A smaller Merit Function number is therefore taken to represent an improved lens with more desirable characteristics. A single optimization cycle occurs using an algorithm that changes each parameter of the lens by a small increment, and calculates the resulting change in each part of the Merit Function. A matrix of partial derivatives is created of defects with respect to parameters. This is often solved by use of a damped least squares method which selects the new lens parameters by following the gradient that leads to the local Merit Function minimum. Optimization proceeds in individual steps or cycles that move towards this local Merit Function minimum. However the local Merit Function minimum is often superseded by the existence of a global Merit Function minimum. In order to overcome this disadvantage some modern optimization programs now have the ability to perform global optimization [19]. Automatic optimization requires that certain weights be allocated to all of the system characteristics in the Merit Function. The designer is therefore able to quantify the relative importance of each characteristic during optimization. In this way the designer can shape the final characteristics of an optimized design. The success of automatic design is therefore largely dependent on the precise form of the Merit Function chosen.

2.2.3 Limitations of the Merit Function Method

The success of automatic optimization depends on definition of a suitable Merit Function. Specification of a suitable Merit Function requires significant skill.

This expertise comes partly from knowledge of the characteristics of similar optical systems that have in the past proven successful. Additionally a particular design form will often exhibit unique properties. As a result, optical design involves learning about the properties of a system through empirical and analytical analysis. This is in order to gain design specific knowledge of the design form under consideration. Using the automatic method this process occurs by observing the effects of optimizing with different Merit Functions and starting locations. However using the Merit Function method the designer has difficulty identifying any interrelationships present. Design knowledge is slow to acquire as a result, and definition of a suitable Merit Function becomes a difficult and slow process.

Ideally optimization should proceed in such a way that the human designer rapidly gains knowledge of the characteristics of the design form under investigation. In the next section we discuss how an interactive approach to optimization can facilitate this kind of knowledge acquisition.

2.2.4 The Need for Interactive Design Methods

It is our sense of involvement with the physical world that gives us the ability to predict events. The techniques of Virtual Reality (VR) give humans the ability to view and interact with the mathematical world of the computer. The term refers to real time three dimensional interactive computer graphics. These methods are known to provide improved problem solving abilities for complex tasks. Optical design involves the solving of multiple problems simultaneously. At present, optical design programs do not provide adequate visualization of the complex nature of the situation. Clearer visualization of the parameter-aberration 'landscape' is required. The ability to interact with these visualizations will also give the designer freedom to explore specific and promising regions of this space. In this way the attributes of a particular system form can be more rapidly explored. For this method to work, the optical design process is presented to the user in such a way that design decisions occur in real time.

A need exists for increased real time user participation in the design process, and methods to facilitate this will help overcome some of the drawbacks inherent in automatic optimization. The majority of the current array of optical design programs have been in existence for a number of years, and their basic form has not changed significantly. The transportation of these programs into high quality graphical user-interface environments like Microsoft Windows has recently become a general trend. However no efforts have been made to alter the actual method of optical design within these user-interface environments. The remainder of this Chapter concerns the interactive design methodology for a new user-interface, developed specifically to improve the efficiency of the optical design process.

2.2.5 General Features of Interactive User-Interfaces

The quality of any new user-interface is often a major factor in whether users enjoy using a software tool. This Chapter deals with the goal of enhancing the user-interface in ways that improve the human user's design abilities. This interface is composed of windows, icons and interaction devices. This term also refers to the mouse driven actions that comprise the interface between human user and the computing environment. To aid discussion of user-interface features, it is common to define the term Basic Interaction Task (BIT) [18] to mean indivisible tasks, i.e. positioning cursors, selecting text and entering alphanumeric quantities. Additionally a Composite Interaction Task (CIT) refers to multiple sequences of individual BITs.

A user-interface comprises the following fundamental building blocks. Firstly *techniques* consisting of a number of BITs and CITs. Additionally program *design tasks* are set to facilitate program goals. We have chosen the basic interaction device to be the common *Mouse* in order to ensure compatibility with a large number of standard IBM personal computers.

Ergonomic factors are emphasised in most interactive systems in order to facilitate user efficiency whilst communicating with the computer. The concept of user-computer *dialogue* is a key one in the design of any interactive system, and the main difference between human-human dialogue and user-computer dialogue is the that computer allows use of images in order to enhance communicative abilities. Additionally the interface language, or set of common assumptions and knowledge, should be efficient and allow communication of information in a timely manner. Various forms of information *feedback* are used to communicate information to the user in these programs, which is a key component of user-computer dialogue design. The designer of an interactive system must attempt to provide the most efficient feedback possible, which often may be in graphical terms.

User-computer dialogue has two important commonalties with language. These are the meaning of the language or its content, and its form or how it is displayed. Interface design has two elements, these being the conceptual and functional elements. Conceptual design concerns the principal concepts of the application to be understood by the user. Conceptual design often relies on the use of a metaphor to encapsulate ideas from something that the user is already familiar. Metaphors enhance the chances of efficient user-computer dialogue, and the choice of an appropriate set of analogies is important early on in the software development cycle. Whereas functional design specifies the detailed information and functionality of the interface. Functional design concerns what information is required for each operation on each object for example. In the next sections we describe the Inter-Opt user-interface by looking firstly at aspects of software engineering, language and software structure. Later sections deal with the conceptual design of the new user-interface that is the core of the interactive ray tracing program.

2.3 Inter-Opt: An Interactive Ray Tracing Program

2.3.1 Software Design Aspects

In this section we describe the development of the Inter-Opt ray tracing program. The software was coded in the C++ language [60] for use on an IBM personal computer. This is a superset of the standard C programming language, and allows the evolution of an *object-oriented* approach to the development of ray tracing code. In the next section the basic concepts of object-oriented programming are introduced.

2.3.1.1 The Case for an Object-Oriented Philosophy

C++ offers the programmer efficient methods for data abstraction, that is defining new data types that closely match the concepts involved in developing an application. These are called classes and offer an efficient means to hold information. Any particular instance of a class is known as an object. Not only can these objects hold 'private' information, but member subroutines can manipulate this data. Messages are then passed to objects that are intelligent enough to be able to manipulate their own data in accordance with the messages.

Object-oriented programming also seeks to exploit communality of types and achieves this through a technique called inheritance. Inheritance refers to one class of object inheriting information from another class. As a result of these advantages, object-oriented code tends to be smaller, and also easier to maintain, than programs written using standard techniques. The final version of Inter-Opt (version 2.3.6) has 19,000 lines of code, printing out on some three hundred pages of A4. The application of object-oriented techniques enabled the development of an efficient program structure. Also programs like Inter-opt, that require interactive graphics, can be far simpler to create using inheritance and the object-oriented approach.

2.3.1.2 Data Abstraction to Model Real World Lenses and Light Rays

Design of a large program requires that the software analyst adopt systematic approaches in order to help develop an efficient program structure. C++ aids systematic program design by allowing the user to concentrate on those real world concepts and relationships that must be modelled. The primary real world concepts that must be modelled in any ray tracing program are light rays, and also surface boundaries between media with different optical properties, i.e. refraction and dispersion. The first step in low level design of the new ray tracing program was therefore identification of these as fundamental artifacts of the software model, and representation of these in the C++ language.

One main header file is employed in which these ray and surface classes are defined.(see Appendix A) A flow diagram for the program is shown in Appendix D. The author identified paraxial and real rays as separate class types. This decision related to the different kinds of ray tracing equations involved. Additionally, a surface class holds information about the individual optical surfaces that comprise any optical system. In the latest version of the program (Version 3.0), a new lens object is implemented, in order to encapsulate those properties specific to a real world lens. This lens object holds or contains those surface class objects that comprise itself, and also pointers to media objects that hold properties of the optical media that are present between its various surface boundaries. The lens object header file is shown in the Appendix G, showing the various functions that comprise its implementation.

A base class called 'shape' holds positional information for graphical objects that can be printed on the VDU screen. This shape class contains member subroutines that facilitate translation of shapes around this screen. A daughter class for shape is specified, and termed surface. This surface class holds individual lens surface dimensions. Because this surface class has shape as its base class it inherits the screen movement routines. The camera schematic that Inter-Opt prints on the VDU screen is composed of a number of surfaces. As a result of inheritance this camera schematic can be magnified and translated around the screen simply by accessing the shape member subroutines.

With these basic constructs of the program implementation identified, in the next sections we take an overview of the structure of Inter-Opt.

2.3.1.3 Design of the Ray Tracing Routines

Inter-Opt uses an object-oriented approach to perform the basic ray tracing functions that are the core of any ray tracing program. The function called Real_tra.cpp shown as file number 67 in Appendix C, traces a real ray from one surface to the next. The routine is very general in nature, and can trace rays of different wavelengths through spherical, aspheric and conic surface types. Note that this may not be the best way to arrange this task to obtain the greatest possible ray tracing speed, because the routine has to check which surface type it is dealing with in a standard 'if test'. This wastes computing time, as the surface should automatically call different ray tracing routines depending upon its specific type. However in order for this to occur the ray tracing routines should be members of either a lens or surface class, and not as in version 2.3.6 members of the ray class. Version 3.0 of the program is currently under development and has ray tracing routines as member functions of the lens class, allowing for the calling of different types of ray tracing routines.

Referring to the file print out shown in Appendix G, the Real_Trace routine is a member of the real ray class. In order for a ray to be traced through the system, this Real_Trace routine is passed a surface object, and the ray is then traced through the corresponding surface. Note that Inter-Opt relies on having the fastest possible ray tracing speed in order to facilitate real time design methods, and so efficiency of ray tracing is an important issue. Within version 2.3.6 of the program in the routine Real_Trace, passing of data in the form of a surface object is not the optimum method. This is due to the large amount of surface data, i.e. variables, that must be passed to the routine upon every single surface ray trace. Such a method involves large overheads in memory management. Version 3.0 of the program therefore passes only ray object pointers to the ray tracing routines. This is in order to reduce the amount of data passed to simply a single address that references each ray object, thus improving the efficiency.

In order to trace a ray through the entire lens, a 'for loop' is required. Its function is to step in turn through all the surfaces in the system, sequentially passing each surface to the Real_Trace function in order to trace the ray sequentially through the system. In a similar way a routine called Real_bac, for each ray to be traced, traces an imaginary ray backwards from the entrance pupil to the first physical surface, in order to determine where each real ray enters the system. Additionally similar routines exist for paraxial ray tracing, and are located in the files called Trace and Traceb.

In addition to the low level ray tracing routines described above, the program also has a number of other calculation routines for use by the higher level routines that comprise the overall program. The routines called Third and Third2 perform calculation of the third order Seidel coefficients for the optical system. An important part of the program design, is the ability to enable specific surfaces to be 'solved', that is acquire specific curvatures and separations to automatically cancel specific aberrations. Function Des4 contains a set of equations that allow the back surface curvature and separation of a field flattening lens to automatically assume the required values to eliminate Petzval field curvature. Additionally a solve is done for axial colour in this routine. Importantly these solves can be done at any point in the code, simply by calling this routine. Another set of routines work in a similar way and are called Des5 and Des6. Both of these routines have their code given in Appendix G, and are used to facilitate automatic design of meniscus lenses for Maksutov cameras, as well as doublet corrector lenses for Houghton systems.

2.3.1.4 Overall Software Structure Design

In this section we look at the coding aspects of the Inter-Opt software as a whole, prior to a detailed look at the design methodology of the user-interface in following sections. Inter-Opt required the implementation of a number of novel features that were largely untested previously. The degree of success of these ideas could not be assured in advance. All of the program concepts therefore required incremental development and alterations as program design progressed. The program development therefore, whilst following an overall plan, evolved during the 2 year time scale of the project.

The initial version of the program was developed primarily for optimizing Short Schmidt cameras. One of the first ways tested to achieve this was by allowing the user to alter the aspheric coefficients of a modelled camera in real time. The method proved so successful that the author decided to apply this simple idea, real time camera alterations and plotting of aberration graphs, to other camera parameters with the aim of correcting different aberrations. In later sections of this Chapter we describe the evolution of these ideas into a conceptual design framework for the overall program. In this section we briefly describe the low level implementation of the program structure.

The first stage of software design is to identify the primary program tasks, in order to split the coding into manageable sub-tasks. The Inter-Opt software has 7 distinct areas. These areas are : loading lenses from computer storage disc, saving lenses to disc, ray tracing routines, display of lens schematics, calculation and display of blur spot diagrams, interactive aberration plots, and parameter space maps. The Inter-Opt software is split into 92 separate files, as shown in Appendix C, where the names and functions of these files are given for reference. Appendix D shows a number of data flow diagrams for the overall program. These pseudo-code diagrams show only the necessary level of detail to clarify the structure of the program, whilst clearly representing relationships between the various sections of code. The Inter-Opt program, once running, prompts the user to enter a lens name to load. Control is then passed to a lens load routine (number 29 in Appendix C). This function performs the task of loading the lens parameters from disc, and creating surface objects that comprise the lens object in the computers memory. Once complete the program automatically locates the entrance pupil and traces a paraxial ray to locate the focal plane by calculating the back focal distance. The routine that performs these tasks is shown in diagrammatic form in flowchart 3. Once complete this flowchart is exited, and program control is returned to the loop represented by flowcharts 1 and 2.

Flowcharts 1 and 2 depict the main program loop, where the software awaits user input, in the form of a key press to invoke the routine required. At this stage the program execution flow is held at the top level menu screen. Flow control can then be passed to lower level menus and screens as the user selects appropriate keys. This BIT is a single letter key hit on the PC keyboard, and is the main menu selection task that controls flow of control to the various routines that comprise the program.

From the top menu screen, the user can select any of 16 separate letters and numbers that direct program flow control to the 7 main areas into which the program structure is classified. One of these areas is individual ray tracing routine invocation. These routines are selected from the main menu using letters 'R' for real ray trace and 'P' for paraxial ray trace. Once invoked, the user defined ray trace routines call the files Par3 to perform a single paraxial ray trace through the lens system, or RR1 to perform a single real ray trace through the entire system. Both these routines call descendant routines as necessary to perform single ray traces through the lens object. See Appendix G for specific details of these routines in C++.

The next main section of the program structure is display of a lens schematic, that is a visual representation of the lens on the VDU screen. To invoke this routine the user hits the button D from the main menu screen and control is passed to the routine Schem, (see the flowchart in Appendix D for the structure of this routine) which holds the necessary routines to display a view of the optical system. The program user can also select display of blur spot diagrams for the system loaded, by hitting key S. On occurrence of this event control is passed to the routine Real. This function aims a set of rings of rays to the entrance pupil of the system. These rays are then traced through to the image plane where magnified views can be selected of the resulting image blur spots.

2.3.1.5 Design of Function Time: A Real Time Interactive Screen

One of the unique aspects of the structure of this ray tracing program is the ray intercept plot screen routine, and its structure is shown in a flowchart in Appendix D. This routine is held in the file called Time. The Time subroutine checks for user BITs that alter the lens parameters. If the user makes any parameter change, the resulting changes to the aberration graphs are plotted in real time on the VDU screen. The Time routine involves a significant amount of computing, and links to a large number of different routines. The routine displays all the lens parameters in quantified scientific notation. The Time routine waits for the user to click the mouse on any of 90 separate 'buttons'. With these buttons the user performs BITs that are used to alter system parameters and display characteristics of the screen. With reference to the Time flowchart, once the user has 'clicked' a button with the mouse, a changed lens object is created in the computers memory. Ray tracing routines then update the back focal length, and also trace other rays through the system in order to re-display the aberration graphs. A number of routines are called as necessary, for instance Graph2 and descendants for polychromatic aberration plots. Alternatively control is passed to Graph and descendants for monochromatic aberration plots. Not shown in flowchart

Time, are a number of other solves that can be applied to a modelled camera so that certain surfaces are ties to real time calculation of third order aberrations.

Due to the procedural nature of the object-oriented code, at any point the software designer is able to perform complex tasks such as calculate the aberrations of the lens using very simple commands. An object-oriented approach to the development of the Time routine has simplified its structure, enabling an efficient hierarchical separation of the application concepts: lenses and rays, from their complex implementation in the C++ language. The advantage to this approach is a natural concentration on lens attributes, and the resulting ability able to alter them at a specific point in the program code. For instance in file Time if the user alters the lens aperture, all that need be programmed into the code is to sense this change then alter the lens object's aperture. This can be achieved in one line of code by sending a message to the lens object that it has a new aperture size, and then again pass ray objects to the lens object to recalculate its new attributes. Looking through the C++ code for function Time, given in Appendix G, we see that the routine senses a large number of possible screen co-ordinates for a possible mouse click. Should one be detected, then the lens parameter the user requires altering is changed at that point. Later in the Time code we see a set of routines called as a result of any change. The first one called is routine EFL1, which performs a paraxial refocus for the lens, which may or may not have changed as a result of the parameter change. Next the routine Graph is called which calculates, and plots on the VDU screen the HTanU curves for the lens in question. Other routines are also called as necessary in a similar way if the user has selected their display. In this way the Time routine relies on the hierarchical structure of the code allowed by the object-oriented approach.

2.3.1.6 Design of Function Spaceb: A Parameter-Space Visualization Screen

The final section of the program to be implemented is the parameter-space map section. The structure of this section is shown in a flowchart in Appendix D, which shows the contents of a file called Spaceb. This flowchart shows the routines that allow the user to move in three dimensions in a virtual world. This world that represents the parameter-aberration 'landscape' of a camera form. The Spaceb file has a number of descendant files that enable the user to maneuver within this space. One descendant file, called Mapb, calculates an area of this landscape for a user chosen region.

The conceptual design of the parameter-space map screen is described in later parts of this Chapter. The functional design of this screen relied on the ability to ray trace a very large number of cameras, and save the results in computer memory. In order to facilitate this process, once ray traced, the aberration results for each camera are saved in computer memory in an abbreviated form. A C++ class called datapoint achieves this task, and its form is shown in Appendix G. Class datapoint holds aberration information for each camera, in order to allow real time display of this data in parameter-space visualizations.

Within the routines that make up Spaceb, the monochromatic transverse aberrations are calculated in a routine called Notgraph. This routine is termed Notgraph to indicate that these aberrations are calculated but not plotted on the VDU screen. Notgraph simply returns the value in microns of the on or off-axis aberration blur spot size in the tangential direction. This aberration size is calculated for a set of twenty meridional rays traced through the system aperture. In this way the aberrations of the camera in question are calculated, and then saved in class Datapoint, for use at a later stage in the interactive design process. Another aberration calculation routine, called Spher, calculates the size of the spherochromatism for an individual camera and returns a number that is saved to memory. This number is then represented as a line in three dimensional space during the design mode. The number is termed Ab_plot2. It is the result of tracing thirty rays, at different apertures and in red and blue wavelengths, through a single camera system. The Ab_plot2 number is the sum of the differences in focal lengths, at red and blue wavelengths, of rays of various apertures. In this way spherochromatism is represented as a single number for saving as a variable in class datapoint. The design of Spaceb relies on being able to calculate aberration sizes for different cameras, in order to compare and contrast their properties during real time design.

The Mapb routine has two other descendant routines termed Des8 and Des8b. These functions eliminate axial colour using automatic Petzval aberration solves on specific surfaces. Another file called Drawmp, displays on the VDU an area of the parameter space calculated by the program in a batch mode. Finally Mapc calculates a set of ten such maps in batch mode, whilst saving the results in memory for real time display during a design phase. The structure of this Spaceb file relies on the ability of the ray tracing code to perform a large amount of computing calculations, ray tracing and data reduction of results, whilst summarizing this data and saving it as aberration results as memory objects (called the datapoint class) in C++. These memory objects, one for each camera, can then be called up from memory and represented in the parameter-space maps.

The Inter-Opt software is based around C++ objects. The resulting simplified program structure is possible as a result of intelligent data objects, that can interpret their own messages. This approach has allowed concerted software effort to be placed in the development of a new concept for the interface between user and computer. In the next section we describe the conceptual design of this interface, and examine its unique methodology in detail.

2.3.2 User-Interface Dialogue Conceptual Design

In this section we describe the basic aims of the new interactive user-interface. Inter-Opt is targeted specifically at the design of Schmidt, Maksutov and Houghton systems. The program contains a number of facilities that are common to other ray tracing programs, and we can separate these more usual aspects of the program, i.e. blur spot diagrams, lens schematics, and general ray tracing routines, from the novel aspects of the program. We shall use the term user-interface to refer to those features that allow interactive optimization to be achieved.

The first key aim of the user-interface is to allow the user to clearly identify and visualize the problems involved during camera optimization. Whilst the second aim is to enable the user to participate in real time solving of these problems. Conceptual design aims to approach these somewhat general aims through specific application concepts that are termed the users model of the application. The conceptual design is based on objects and relationships between objects, where here we refer not to C++ objects but to the concepts that make up the user-interface. The basic object in the new user-interface is termed the lens object. Use of the term lens object is a general reference to any optical system, and here usually refers to a catadioptric camera. In this case lens refers to a modelled camera system that has some specific attributes of a real world camera. In other words the composite mirrors, lenses and light rays that can be ray traced through this lens.

A key idea in enhancing the user-interaction of this interface is real time design. This indicates that the modelled lens object must be variable on very small time frames, to enable the properties of a large number of different lens objects to be explored by the user. A central interface concept is that the lens object in this new methodology differs from a real world lens in a crucial aspect, that is that its attributes can alter in real time as if they were made of a malleable substance. We shall identify this as a central concept and term it the *fluid-lens*, in order to indicate that its form can be altered and is not fixed. A fluid-lens can be physically changed in real time whilst its properties are examined graphically. A large number of different real world lenses exist that can be applied to a particular task, and some will match the design specification more closely. We can envisage a multi-dimensional space of many possible lenses that exist with different parameters and properties. The human user might wish to maneuver through this space in real time as if flying over a landscape. One can imagine that as the user travels over the terrain, he will become more familiar with its specific attributes. This *parameter-space* is the second basic concept of our new user-interface, and can be made analogous to real space, having three dimensions of space and one of time. The familiarity of these two metaphors will help the user identify with the form of user-computer dialogue being used in the program.

Throughout the description of the program development that follows we refer to two separate interaction tasks, one type is classified as a BIT and is the *quantify interaction task*, (QIT), whilst another type is a CIT and is referred to as a *3D interaction task* (3DIT). The quantify interaction tasks in this user-interface shall be simple click events on buttons with a mouse in order to change the size of a variable parameter, either in fluid-lens alteration or during displayed other information. The three dimensional interaction tasks center around the parameter-space map screen.

This screen enables the user to maneuver within visualizations of this space to select areas of the space in which cameras 'exist' with favorable properties.

This section has introduced the basic conceptualization of the user-computer dialogue. In the next section we describe those aspects of the program that are commonly used in other ray tracing programs.

2.3.3 The Basic Ray Tracing Facilities of Inter-Opt

The core program provides on-axis ray tracing of optical systems with up to 12 surfaces with either spherical, conic or aspheric shape. Other capabilities include paraxial and real ray tracing, optical system schematics, third order Seidel calculations, plotting of blur spot diagrams, and display of aberration intercept plots. Figures 2.1 and 2.2 show schematics of a photographic triplet and catadioptric system from the Inter-Opt program. Figure 2.3 shows blur spot diagrams for a catadioptric system. The row of plots across the top of this screen show spots for three focal positions. These spots are for the on axis field position, and the central set of coloured spots are at the paraxial focus. To the left and right of this central position are blur spots for two other focal positions, one for each focus stepped by 0.01mm. Each individual spot is replicated in three colours that represent three separate wavelengths as specified by the refractive indices in the lens file for that system. The lower sets of spots are identical to the top plots, but are for the maximum field point as specified by the user.

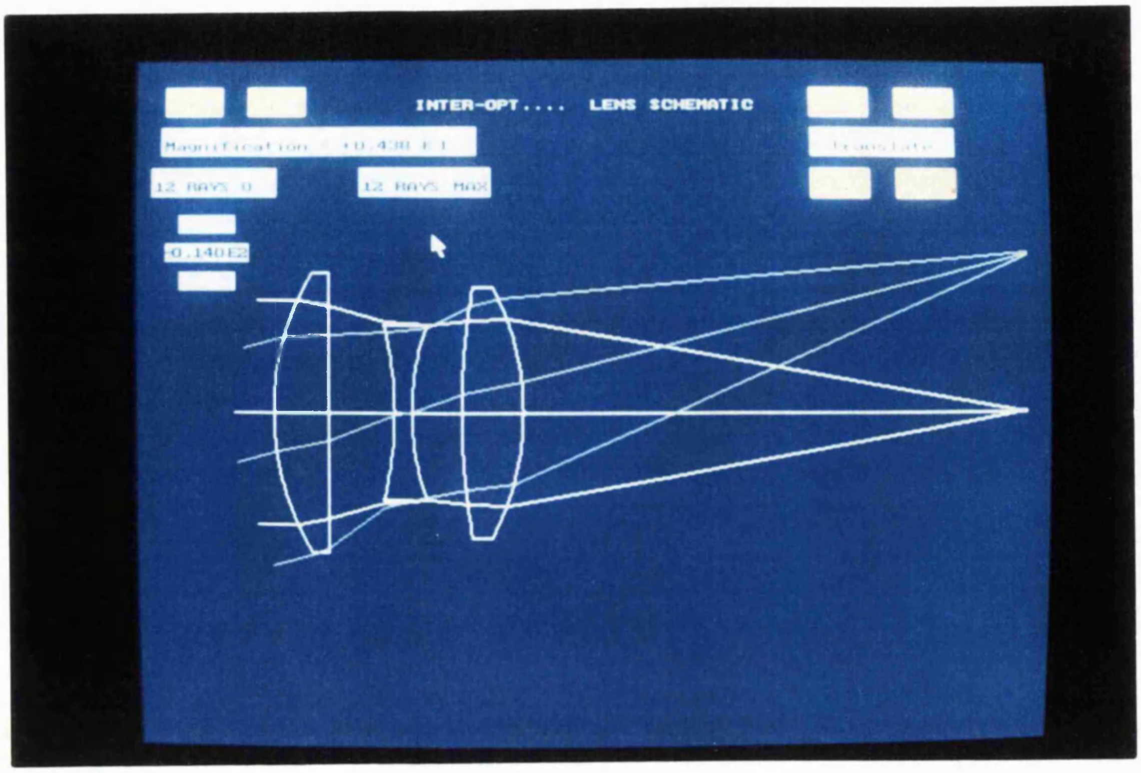


Figure 2.1: Inter-Opt Lens Schematic: A Photographic Triplet Design

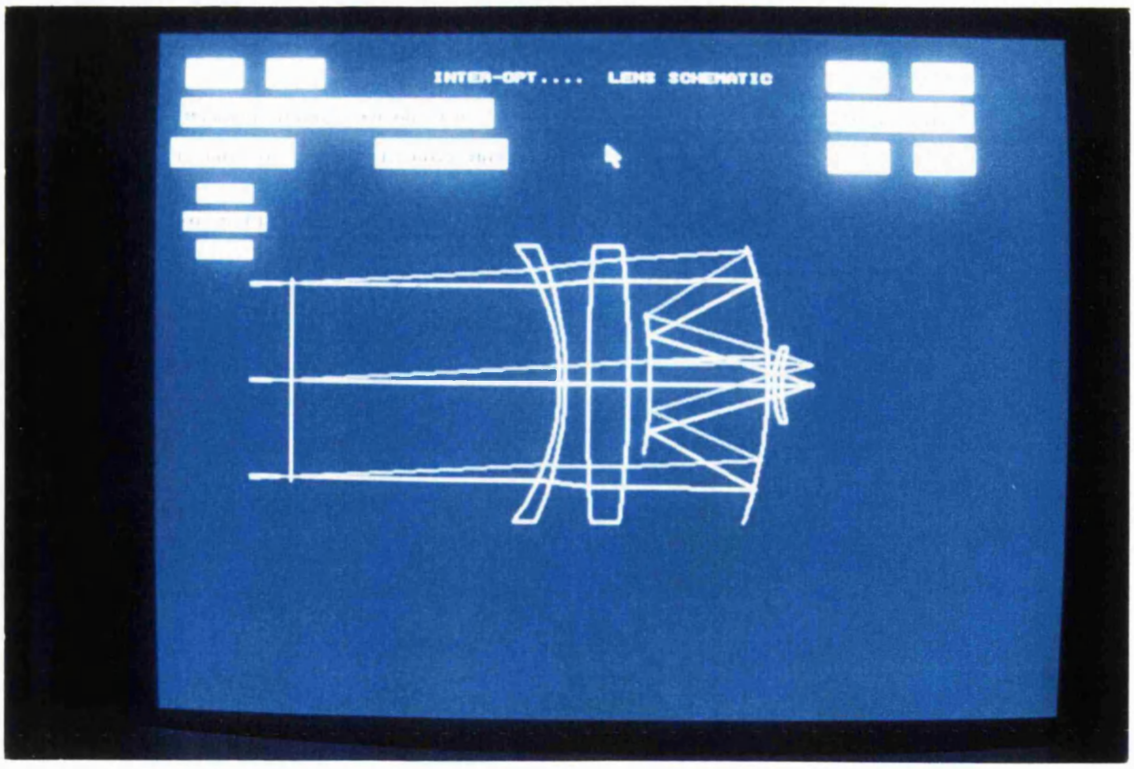


Figure 2.2: Inter-Opt Schematic: Cassegrain Catadioptric System

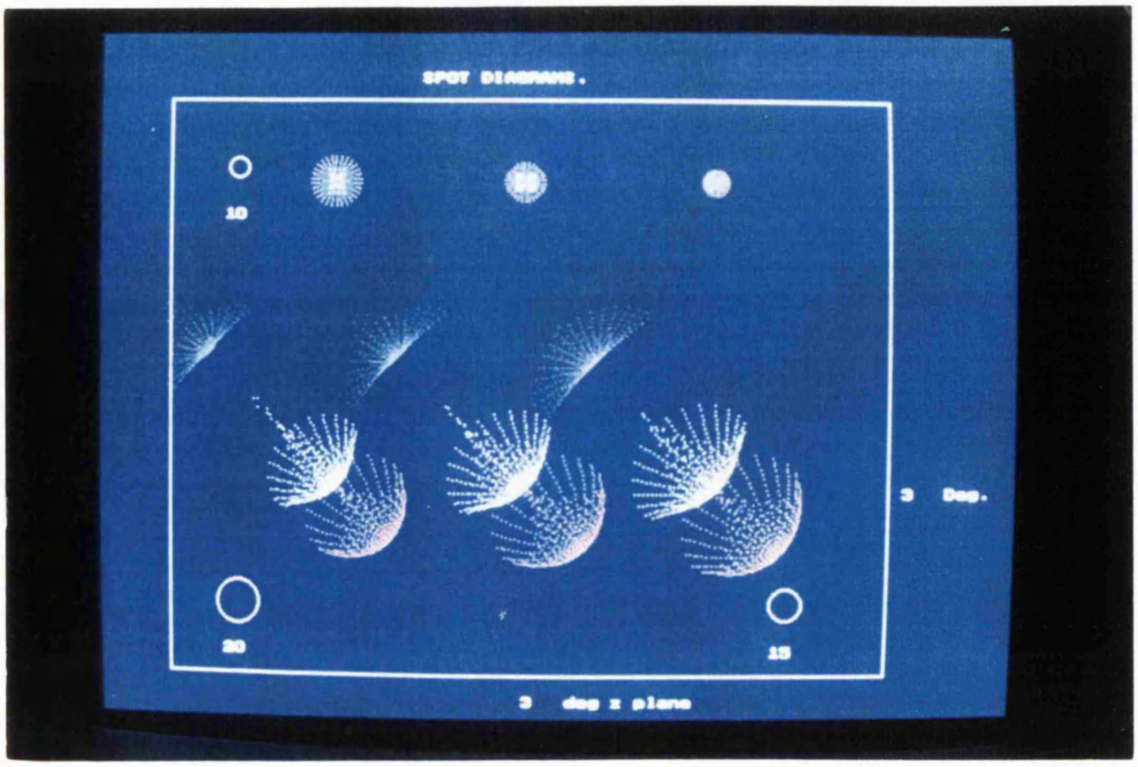


Figure 2.3: Inter-Opt Blur Spot Diagrams for a Cassegrain System

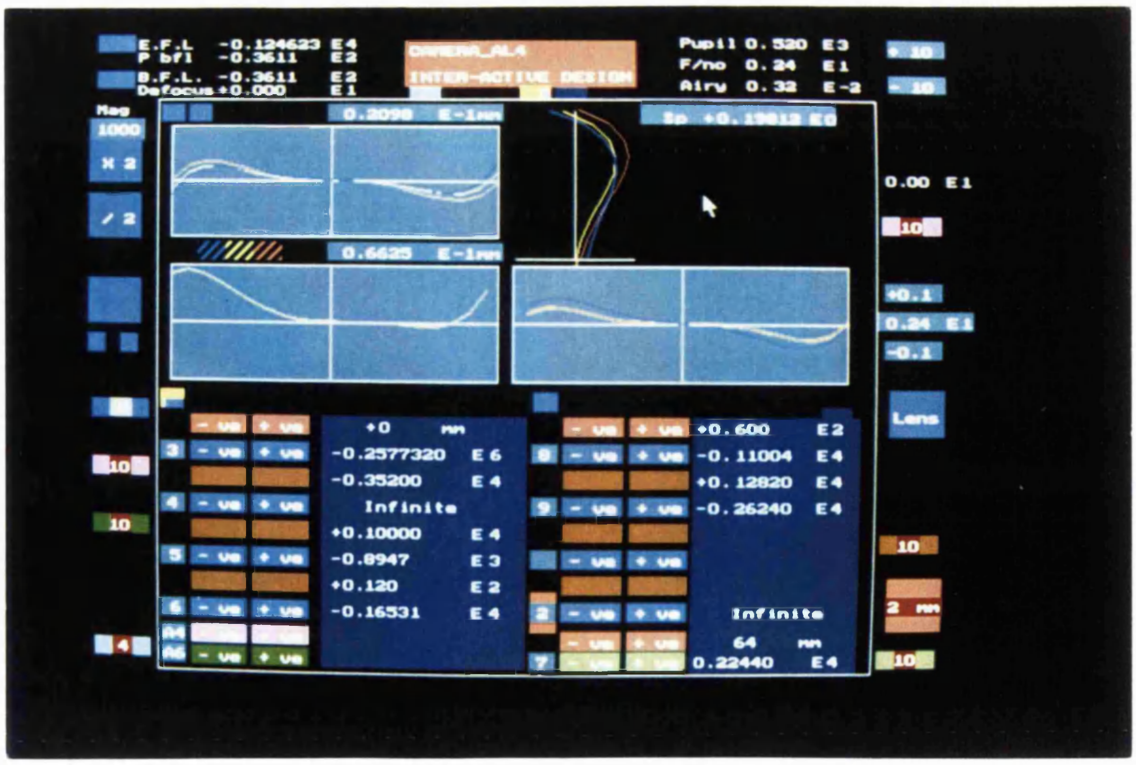

```
E.F.L    -0.124623 E4          Pupil 0.520 E3
P bfl    -0.3611   E2          F/no  0.24  E1
B.F.L.   -0.3611   E2          Airy  0.32  E-2
Defocus  +0.000    E1

0.2098  E-1mm                  Sp +0.19812 E0
0.6625  E-1mm

0.00 E1
0.24 E1

+0          mm                 +0.600    E2
-0.2577320  E6                 -0.11004  E4
-0.35200    E4                 +0.12820  E4
  Infinite                     -0.26240  E4
+0.10000    E4
-0.8947     E3
+0.120      E2                   Infinite
-0.16531    E4                 64        mm
                               0.22440   E4
```

Figure 2.4 : The Interactive Ray Intercept Plot Screen

2.3.4 The Interactive Ray Intercept Plot Screen

In Figure 2.4 a photograph of the ray intercept plot screen is shown, which displays the ray intercept plots as commonly used to evaluate lens designs. This screen is one of the main interactive tools available in the program. Using a mouse, the program user can change any lens parameter in the fluid-lens, whilst real time ray tracing instantly updates these plots. In this way the effect of any parameter change on aberrations becomes immediately apparent to the designer. This ability to alter the properties of a fluid-lens, whilst in real time observing the effects graphically, is unique to this program. A number of coloured 'buttons' are displayed in this screen, and these are QITs that allow mouse control of the various screen functions. For a detailed explanation of button functions see the Inter-Opt help manual in Appendix B. The three turquoise rectangular plots in this screen are ray intercept plots in the image plane. The two left hand plots result from the tracing of meridional ray fans. For each plot, intercept height is plotted (vertically) against the slope of the rays (horizontally) as they emerge from the lens. Ray slope is plotted as the tangent of the emerging angle. Intersection heights are relative to the paraxial ray height for the field ray in question. Note that paraxial rays are infinitesimally close to the optic axis and are not aberrated. The upper left plot displays results for an on axis fan of meridional rays, whilst the lower left hand plot shows the maximum field point ray fan. The third plot on the right hand side shows results for a fan in the orthogonal sagittal plane plotted against intercept in that plane. This right hand plot reveals any astigmatism present and is for a fan of rays at the maximum field point. Real time design provided by this screen is a very powerful technique, for instance when designing a Schmidt camera system the user must calculate the correct values for the aspheric coefficients of the corrector plate. Using Inter-Opt this can be achieved simply by altering these coefficients, and watching the effects on spherical aberration in real time. The procedure can be completed in only a few seconds for any camera system. In figure 2.5 the results of altering the aspheric coefficients are shown for a Schmidt camera of 600mm focal length and f-number 3.0. These the plots show on axis meridional plots for three different values of A4 and A6. Starting at the top of the Figure and moving to the lower graphs, we see the results of altering the A4 coefficients and A6 coefficients in real time. In the pictures depicted here only around ten clicks of the Mouse were required to remove spherical aberration from the system completely.

By 'clicking' with a mouse on appropriate parts of the screen using QITs, the user is free to alter system parameters such as back focal length, field angle, and aperture. Additionally any individual surface can have its radius of curvature, aspheric coefficients and position altered using the mouse. In the top right hand corner of Figure 2.4 the longitudinal spherochromatism plot is displayed. In a Schmidt camera it is necessary to give the aspheric corrector plate a precise amount of base curvature in order to introduce a specific difference of focus of red and blue wavelengths. The introduction of this aberration balances the effects of residual spherochromatism that limits the performance of the Schmidt system. The result of altering in real time the base curvature of the plate is shown in Figure 2.6. During this procedure the program instantly updates the spherochromatism plots, enabling the required amount of curvature to be found that exactly balances the effects of spherochromatism. Figure 2.6 shows aberration plots for a Schmidt camera as it progressed through this procedure. Graph set 1 shows the aberrations in the starting design. At this stage the 600mm focal length camera of f-number 3.0 has a large amount of residual spherochromatism, as well as under-corrected spherical aberration. The blue wavelength aberration size is quantified above each plot. In the top plot this number is 27 microns due to the residual variation of spherical aberration with wavelength. Figure 2.6 shows individual aberration sets 2,3 and 4, which represent gradually increasing amounts of base curvature on the corrector plate. Each set is an individual time frame taken during the interactive design process. The difference between each aberration set is an increasing difference of focus between blue and red wavelengths. The corrector plate curvature is only changing from 20626.0 to 20445.6, during this procedure. The final camera design, shown as set 4, has a reduced blue (or short wavelength) aberration size of 18 microns.

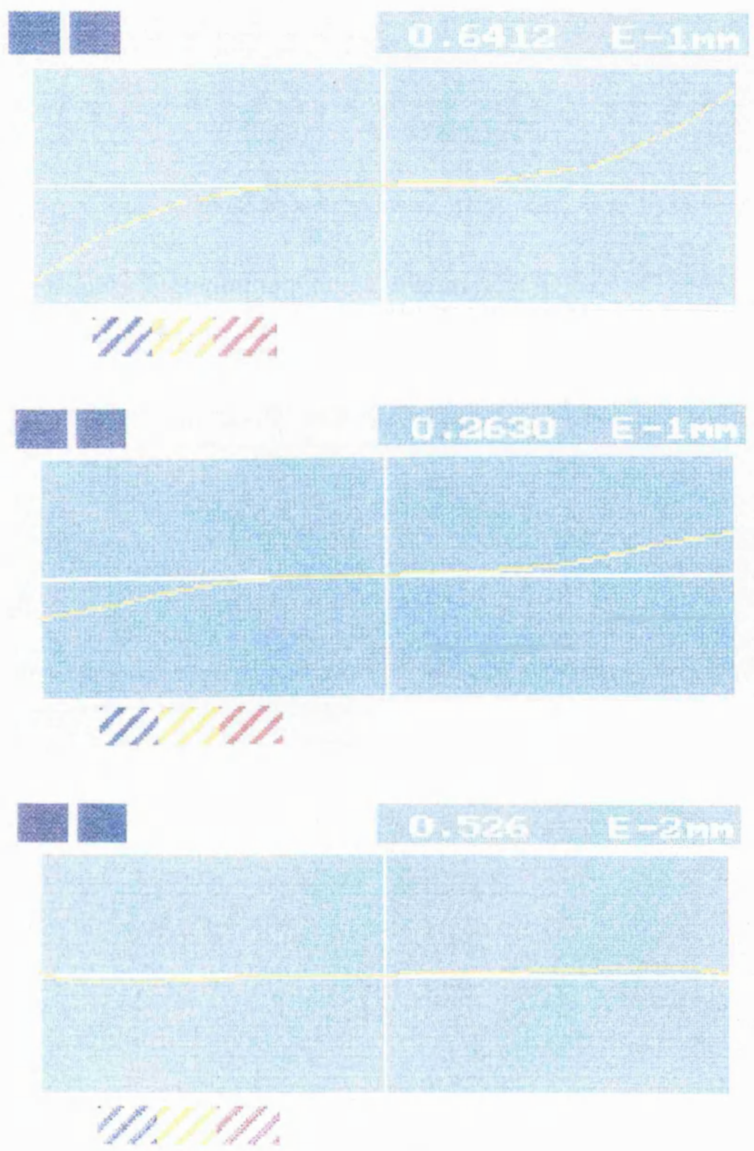


Figure 2.5 Optimizing the Aspheric Coefficients of the Corrector Plate in Real Time for a Schmidt Camera

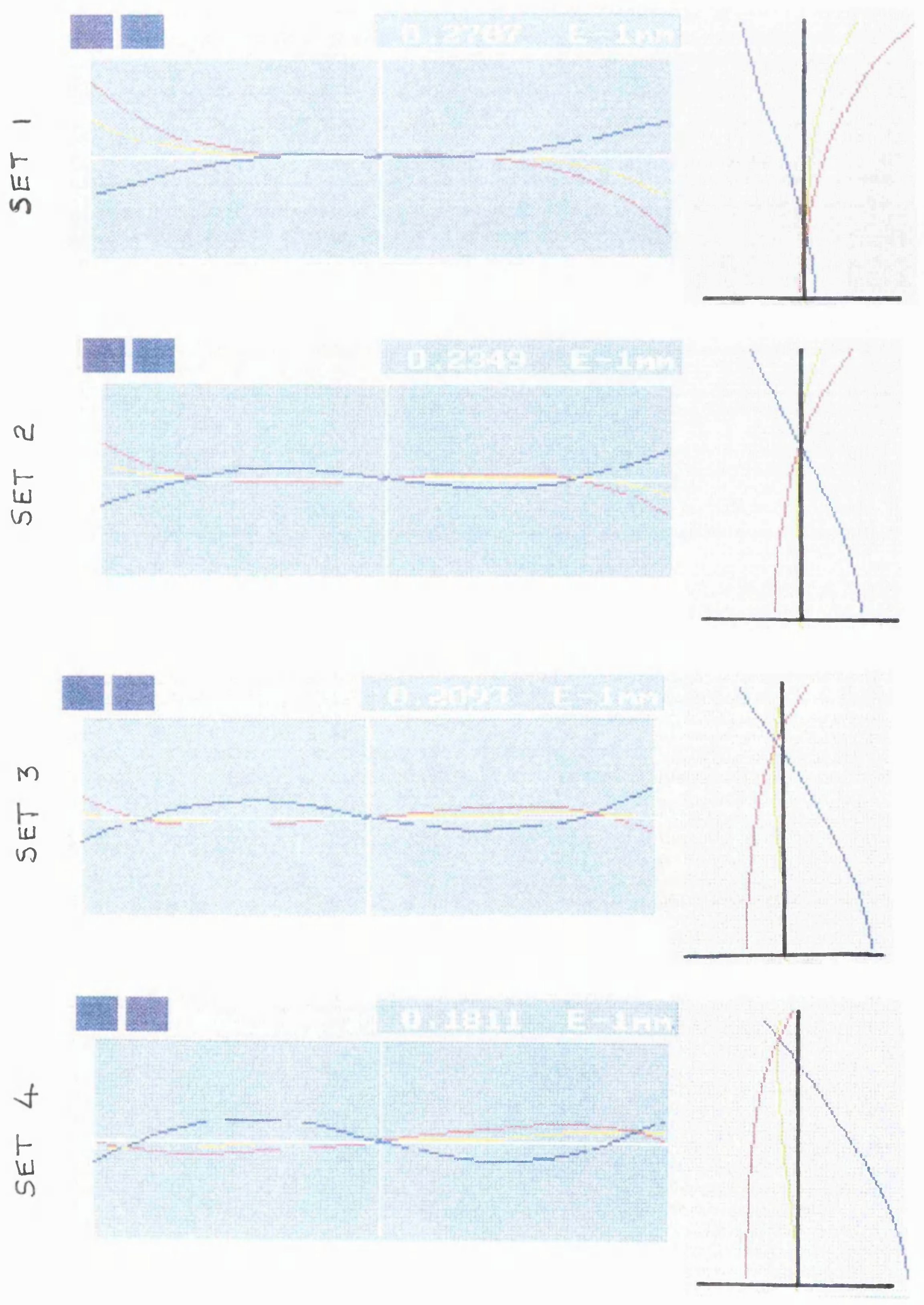


Figure 2.6 : Real Time Balancing of Axial Colour at the Corrector Plate to Reduce the Effects of Spherochromatism

During this process, the program user can watch how the curves change shape. This enables him/her to easily find the required size of change to the radius of curvature. This procedure can be very efficiently completed in only a very short time-span using Inter-Opt. This is a key advantage of the method over other less interactive methods. The effect is very subtle, and tiny changes in the base curvature can effect aberrations strongly on cameras of high numerical aperture.

2.3.5 A New Type of Aberration Graph: Real Time Display of Aberration Contributions

One unique aspect of the ray intercept plot screen, that is not available on any other current program, is that the user can select display of aberration contributions. Once in this mode, the aberration plots are displayed as described above, except for each curve two extra contributory curves are displayed. These curves represent third and higher order contributions to the real aberration curve, and are shown in Figure 2.7. The third order curve is calculated using Seidel equations, and is purple in colour. Additionally these graphs show the higher order aberration contribution. This curve is pink in colour, and is calculated to be the difference between the quantified third and real aberration. The top graph is for an on axis ray fan. In this graph the purple curve is simply the calculated Seidel third order spherical aberration contribution at different aperture values. Also in this graph the pink curve is simply the quantified difference between the purple and real aberration values shown in yellow, for different aperture values. The pink curve represents the higher order contributions to aberration size. The lower graph shows aberrations for an off-axis ray fan. In this graph the purple curve is calculated from the sum of the Seidel spherical and coma quantified values at specific apertures. Please note the off axis curve is only correct in absence of Petzval curvature as a result of not including this in the contribution quantification.

The curve contribution facility is unique to this program. The method helps the user identify occasions when higher order aberrations are limiting any further improvement to a particular design. This screen allows the contributory sum effects of various aberrations on the size of blur spots to be very precisely and quickly quantified. These methods allow the effects of design changes on the fluid-lens to be rapidly assessed, as aberration contributions are more fully conceptualized. Use of these new aberration graphs allows the user to quickly picture and conceptualize where the aberrations originate at any stage of the design process. Importantly these new graphs aid detection of which aberration is contributing the most to the size of a blur spot at any field position, and hence enable a precise evaluation of the limiting aberrations that prevent further improvements to the geometrical imaging performance.

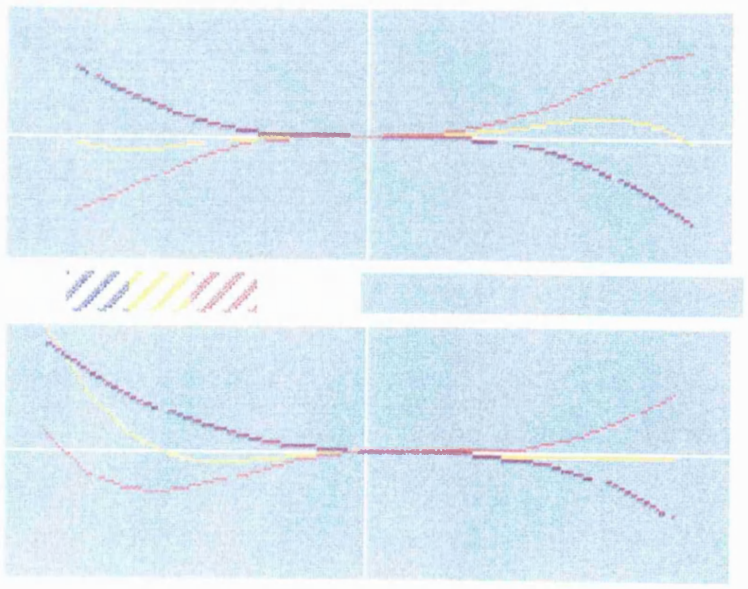


Figure 2.7 : Real Time Calculation and Display of Aberration Contributions

2.3.6 A New Interactive Method of Reducing Coma Within a Field Flattened Schmidt Camera

The interactive screen allows precise targeting of the problems encountered in optical design. One example occurs in the design of a field flattening lens for use in a catadioptric camera systems. This process is described and illustrated in the following sections. A number of solves can be set in the Time routine, where certain surfaces are 'tied' to third order aberrations generated in the rest of the system. These solves are used to correct axial colour and Petzval field curvature. During real time design, individual surfaces can then automatically acquire the power or position required to cancel these aberrations. The program user can then concentrate on bending system parameters to correct other geometric aberrations.

In his paper on Short Schmidt cameras [76], Wynne states that experimentation with field flattener shape can allow a specific shape to be found in which coma is eliminated from the camera by adjusting the separation between mirror and corrector. The author found that this procedure can be achieved efficiently using Inter-Opt by adjusting the shape of the field flattening lens within the fluid-lens camera in real time.

Note here that Inter-Opt automatically finds the thickness and power (rear curvature) of this field flattening lens, in order to correct the axial colour of the lens and the Petzval field curvature introduced elsewhere in the system. To adjust the shape of the lens in the Inter-Opt software, the user simply changes the radius of curvature of the first surface of this lens. Lens 'bending' in real time, allows the shape to be found that makes the off-axis meridional aberration curve symmetrical. Figure 2.8 shows a set of off-axis (field position 3 degrees in the y plane) monochromatic tangential (on the left) and sagittal (on the right) HTanU curves for a Schmidt

Camera. These graphs are for a Schmidt camera of f-number 3.0 and 600mm focal length, taken at different stages of the real time design process. The top set of curves, labeled set 1, show off axis curves for the camera at the start of field lens design.

At this stage the only aberration that has been corrected is spherical aberration using a corrector plate at the center of curvature of a mirror. Additionally specific aspheric coefficients have been found to correct this aberration initially. Set 1 shows the design in this initial state of correction. Set 2 shows the results of altering the shape of the field flattening lens by changing the front curvature until a symmetrical shape is achieved for the off axis meridional aberration curve. This is achieved by repeated clicking of the button that alters the radius of curvature of the first surface of the field lens.

During this procedure, the user increases or decreases the radius as necessary, whilst watching which has the desired effect on the camera aberrations. Once a symmetrical curve has been achieved, (as shown in set 2 of Figure 2.8) the user adjusts the mirror-corrector separation until the off axis aberration size is at a minimum. Repeated iteration of these two procedures for a few cycles allows coma to be considerably reduced from any camera system. Set 4 shows the results after altering the front curvature again until the curves are again symmetrical, whilst set 5 shows the results after again altering the mirror-corrector separation. This procedure can often be achieved in a few minutes using Inter-Opt, and is described in the help manual in Appendix B.

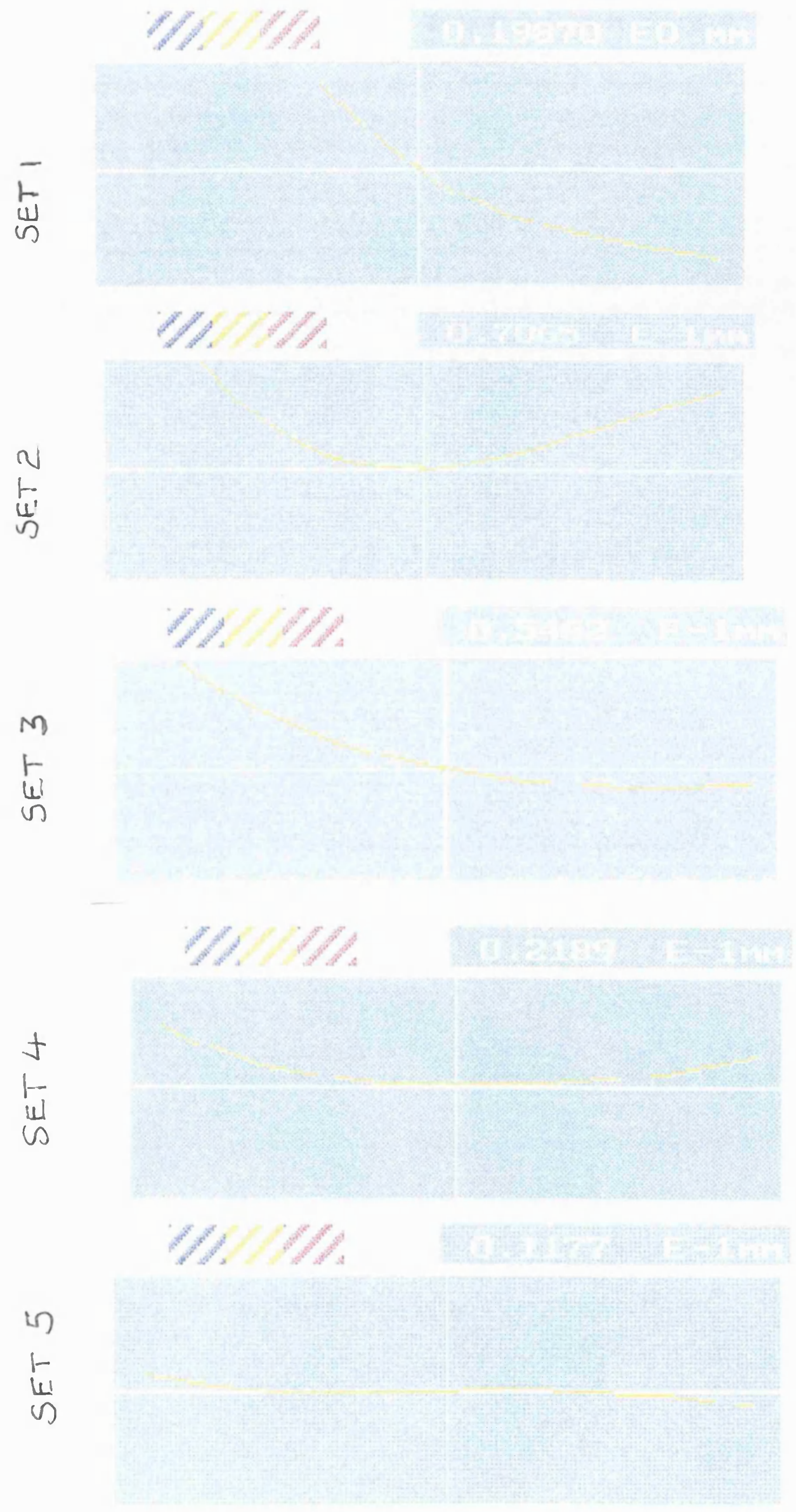


Figure 2.8 : Real Time Correction of Coma During the Design of a Schmidt Camera

These plots show the results of design changes after only a few minutes. Watching how the aberration curves change shape, enables very precise control of the effects of design changes. The procedure has not to the author's knowledge been achieved in this way before. The method is an extremely efficient one, enabling correction of coma (and often astigmatism) in a very simple and rapid fashion from camera systems in general. The method has been found to work on Maksutov and Houghton systems in the same cyclic manner. Interactive design has been demonstrated to shorten the time taken to fully optimize a Schmidt camera. Enhanced familiarity with a system's characteristics is the result, enabling the designer to find the best design routė to the form he/she is searching for.

2.3.7 Interactive Screen: Specific Advantages of this User-Computer Dialogue Form.

The fluid-lens concept has proven to be an advantageous way of enhancing user interaction. The method enables system characteristics to be altered in real time. Additionally, real time graphs provide enhanced dialogue between user and computer. As a result the user is able to clearly visualize the effects of changing specific parts of the system in isolation, or in combination with other parts. Improved feedback allows more information to be communicated to the interface user. Feedback is enhanced in two distinct ways, firstly real time responses to alteration of fluid-lens parameters. When responses occur too slowly humans have difficulty conceptualizing what is occurring. The second feedback improvement is to the information displayed, in the form of graphical communication of information. Inter-Opt allows the user to visualize when one aberration is moving, relative to another one, more slowly with respect to changes in a particular parameter. This facility therefore enables relationships to be closely monitored during the design process. The Time routine allows the user to make tiny increments to parameters, that can be scaled to just the required size to change an aberration by a required amount. If an increment is too large, the user simply alters the size of the increment on that parameter until it has the desired effect. A very precise control of aberration changes is the result, through specifically scaled parameter increments. Improvements to the dialogue between user and computer, have in this screen, enabled the human user to participate more efficiently during the design of a particular system. In the next section we describe a new method for displaying aberrations developed to aid the design of Houghton camera systems.

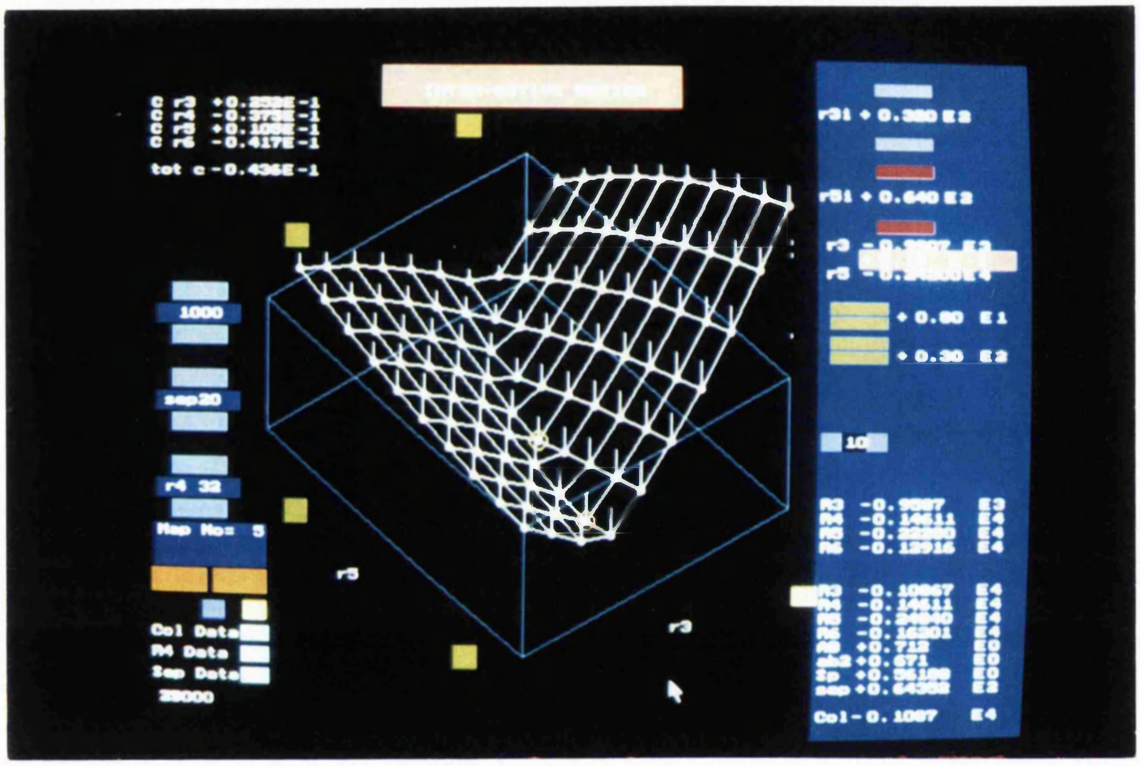
2.3.8 A New Optimization Technique Based on User Interaction with Multi-dimensional 'Parameter-Space Maps'

The second core concept of the new user-interface is the ability to fly over the 'terrain' that composes the parameter-aberration space of a camera form. Many different cameras 'exist' in this space. Inter-Opt enables the user to explore this space, and move in real time to areas where promising cameras are located. The Inter-Opt program applied this concept specifically to the design of a field flattened Houghton camera. The program allows the user to visualize and interactively explore the complex parameter-aberration space of these systems. The principle of the Houghton camera is described in detail in Chapter 5. Here we note that such a system has a separated doublet, in air, in place of the usual aspheric plate used in the Schmidt system. Also the cameras investigated here have a field flattening lens located close to the focal plane. Design of these cameras involves two extra surfaces over the Schmidt form, and this complicates the process of identification of any parameter-aberration relationships present. To overcome this difficulty the author applied the new concept of parameter-space maneuvering to the design of these complex types of systems.

The Inter-Opt software probes regions of the design space of a particular camera form, by ray tracing a large number different cameras in batch mode. This procedure takes several minutes on an 66 Mhz. IBM P.C. . During this time aberration results for each of these cameras are saved in RAM memory. Once this procedure is complete, the user is then able to explore the results using real time three dimensional aberration displays. The program presents the user with a view of the parameter-aberration space of the system in the form of a three dimensional 'data cube'. In Figure 2.9 we can see a photograph of this screen.

Each point on the map represents the results of tracing 75 rays through a single Houghton camera, with a unique shape of doublet corrector and field flattening lens. The location in the vertical direction of each point is proportional to the size of the spherical aberration of that camera. Points closer to the bottom of the screen indicate smaller relative spherical aberration. The horizontal directions on the data cube represent cameras with differing values of the first and third radii of the doublet corrector A vertical line attached to each camera data point represents the relative size of the spherochromatism for that particular camera.

For the purposes of this discussion we have labeled from 1 to 4 the surfaces of the full aperture air spaced doublet of the Houghton system. (Picture two separated singlet lenses in front of a spherical mirror, located in approximately the same position as the aspheric plate in a Schmidt system. See Chapter 5, Figure 5.5 for an example system)



```
c r3  +0.[illegible]E-1
c r4  -0.[illegible]E-1
c r5  +0.10[illegible]E-1
c r6  -0.417E-1
tot c -0.436E-1

1000
sep20
r4 32
Map No=  5
Col Data
R4 Data
Sep Data
[illegible]000

r31 + 0.380 E2
r51 + 0.640 E2
r3  -0.[illegible] E3
r5  -0.[illegible] E4
+ 0.80  E1
+ 0.30  E2
10

R3  -0.9[illegible]   E3
R4  -0.14611   E4
R5  -0.2[illegible]   E4
R6  -0.12916   E4

R3  -0.10[illegible]   E4
R4  -0.14611   E4
R5  -0.[illegible]   E4
R6  -0.16[illegible]   E4
[illegible] +0.712     E0
[illegible] +0.671     E0
[illegible] +0.[illegible]   E0
sep +0.[illegible]   E2
Col -0.1087    E4
```

Figure 2.9 : The Parameter-Aberration Space Map Screen

The parameter-aberration space map allows the user to visualize the results of simultaneously changing the first and third radii of the doublet corrector. At the same time, the fourth radius of curvature of the corrector is automatically solved for each camera, and takes the required value in each case to cancel the first order axial colour of the doublet. Also in order for the program to calculate the spherochromatism correctly for each camera, automatic design of a self achromatic field flattening lens proceeds for each camera represented by an individual point. This ensures that the first order axial colour sum is zero for each camera system, at least temporarily whilst within this screen. Additionally, each field flattening lens automatically assumes the correct power to make the Petzval field curvature zero for that particular camera. These solves mean the program is performing a directed search of the parameter-aberration space, in regions where axial colour and field curvature are near to zero. The program user can select a particular camera from the map by using the mouse. That camera can be set as the default for other aberration screens by clicking appropriately.

Another feature allows the effects of simultaneously changing three radii of the doublet corrector of this camera form to be investigated. This requires a significant amount of ray tracing with up to one thousand camera systems being ray traced, requiring a total of 7500 rays to be traced. On a 486 DX33 PC this takes 120 seconds. The program saves the results of this ray tracing in the computer memory as C++ objects, storing aberration results for each camera. Once complete, the user can select different values for the second radius of curvature using the mouse, whilst in real time each associated map is instantly displayed on the screen. In this way real time displays are used as a third parameter-aberration visualization tool.

The user gains the impression of flying over the terrain that makes up the parameter-space, and can easily visualize regions where aberration minima lie. Also available as another parameter to alter in real time is the doublet corrector separation, which often allows correction of spherochromatism. Correction of spherochromatism is especially difficult in these types of systems, whilst simultaneously correcting primary spherical, coma and axial colour. Using parameter-space maneuvering, the user can remove spherochromatism completely from Houghton systems in a very efficient way. Figures 2.10 to 2.15 show the application of the new method to the design of a Houghton system of f-number 1.6. The system is a camera design of focal length 480 mm.

These series of graphs show changes to the parameter-space maps at various stages during optimization of a Houghton Camera. Figure 2.16 shows the resulting reduction in aberrations for a set of on-axis HTan U aberration curves. Individual sets in this Figure correspond with particular camera designs at different stages of the design process. We note that spherochromatism in absence of axial colour is quantified by the sum integrated space between red blue and yellow curves. The plots in set 1 of 2.16 show large amounts of spherochromatism, which is gradually reduced until in set 5 the aberration has been almost completely eliminated. Spherochromatism can be qualitatively measured by the reader in these graphs by looking at the distance between red and blue wavelength curves along the aperture of the system. On these graphs red curves show aberrations at 7000 Angstroms, and Blue curves are for 3000 Angstroms. The camera depicted in set 5 in these plots, has multiple coloured curves that are very close together, indicating a high degree of chromatic correction. That camera represents the best camera found during the entire optimization cycle. The spherochromatism correction procedure was completed by the author from first principles in less than ten minutes. During this time the author explored the properties of more than 20,000 cameras, before selecting the most suitable one displayed circled with a ring in Figure 2.15.

We now briefly follow the design process, with reference to Figures 2.10 to 2.15. Map 1 in Figure 2.10 shows a very long range view of the space, with large increments between parameter changes of around several hundred millimeters in radius increment. In this plot a number of deep valleys exist with relatively small levels of spherical aberration. The user clicks onto the circled camera lowest in the valley, and the program then displays that region of the landscape around that particular camera. Set 2 in Figure 2.11 shows a zoomed in close up of that region of space, where parameter increments between cameras are now far smaller, between 64 and 128mm.

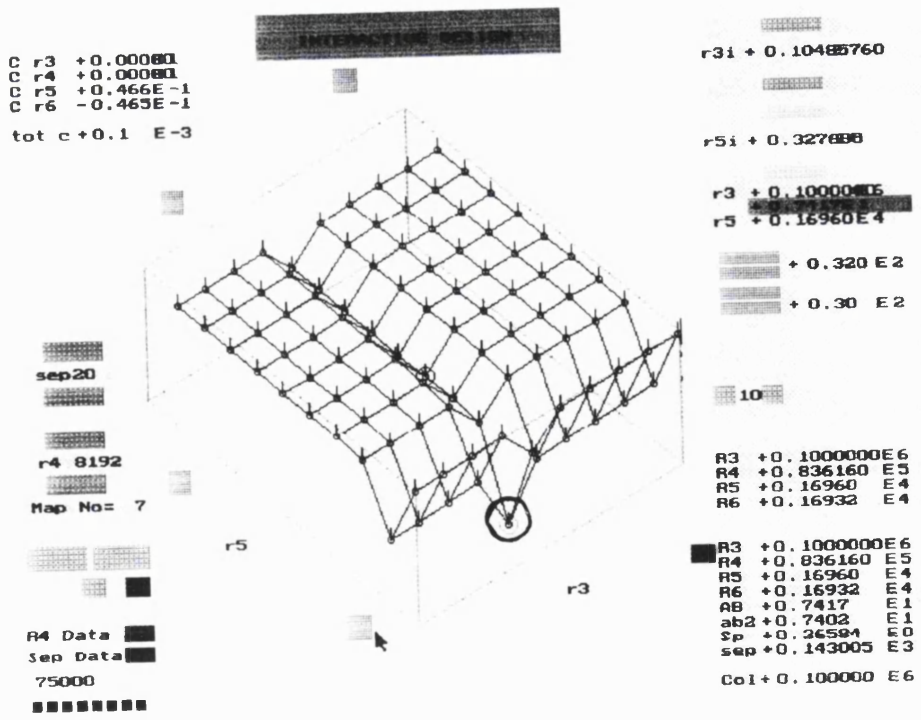
The aberration scales are larger in this particular plot, in order to enhance the relatively smaller aberration sizes in this region. Figure 2.11 shows that the circled camera has the lowest aberrations, indicating a move towards that region of space may prove fruitful in the search for the best solution. The vector towards this region of space is directed outwards from the screen.

Clicking appropriately a number of times, to move through several maps and so several hundred cameras, provides the map shown in Figure 2.12, where the aberrations are much smaller. Note that the length of the line attached to each point in camera space, is the quantified amount of Spherochromatism. Also in Figure 2.12 the region of space seen has cameras whose lines are quite long, indicating large residuals of this aberration. At this stage the aberration curve (set 1 in Figure 2.16) corresponds with the best camera out of the 100 in this map. On this map that camera is ringed in black.

Moving through the design space to minimize the length of the line and hence spherochromatism, is shown in maps in Figures 2.13, 2.14 and 2.15. Note during this procedure that aberration curves in sets 1, 2, and 3 correspond to these three maps, showing a gradual reduction in spherochromatism as the user moves to increasingly more favorable regions of the space. Each map plot represents the results of scanning through 1000 cameras, and is the best map in a set of ten. The final optimization process depicted by these three maps , and the curves shown in Figure 2.16, show the results of scanning through five thousand separate camera systems.

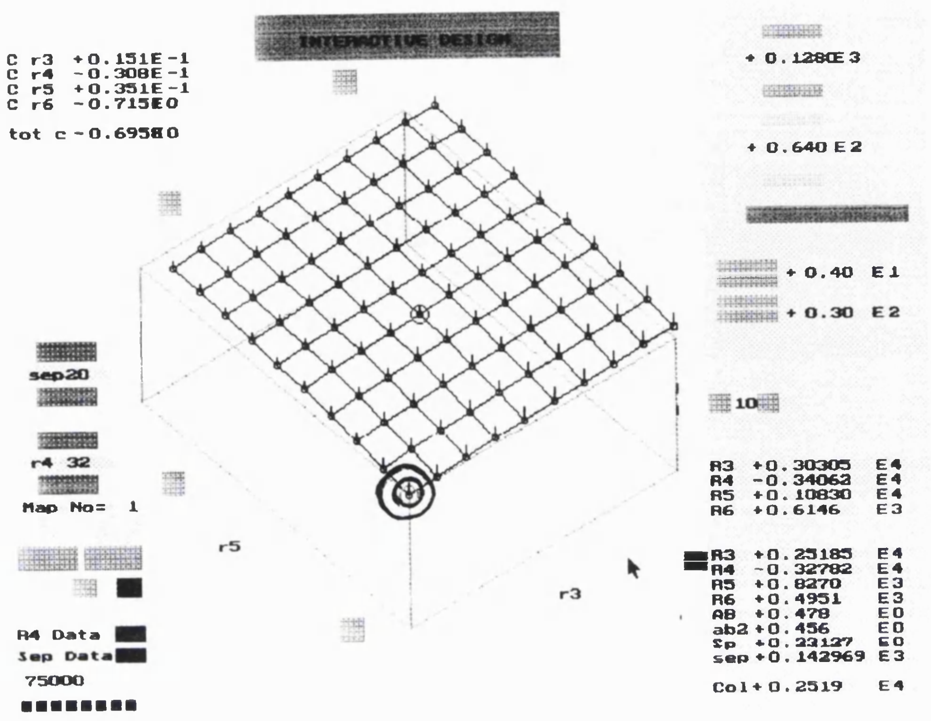
The final design is marked with a ring in Figure 2.15. Note that the levels of quantified spherochromatism are much smaller in this region of parameter space. As a result the line length has been scaled up to be three times the size, in order to clearly show the camera with the smallest residual.

It is important to notice from the map in Figure 2.10, that the user begins by exploring wide regions of the design space. Here the increments between separate camera systems are large, i.e. in this map increments of greater than 100 mm in the radius parameter are the case for example. The user is seeing a very large area of the parameter space at first therefore, whilst later maps zoom in on progressively smaller and more detailed regions. In this way the user is identifying trends in the data in a very efficient way. Unlike other optimization methods, this technique does not require every possible camera to be ray traced along the route to the best design. Other less directed optimization methods effectively waste computing effort exploring countless fruitless options.



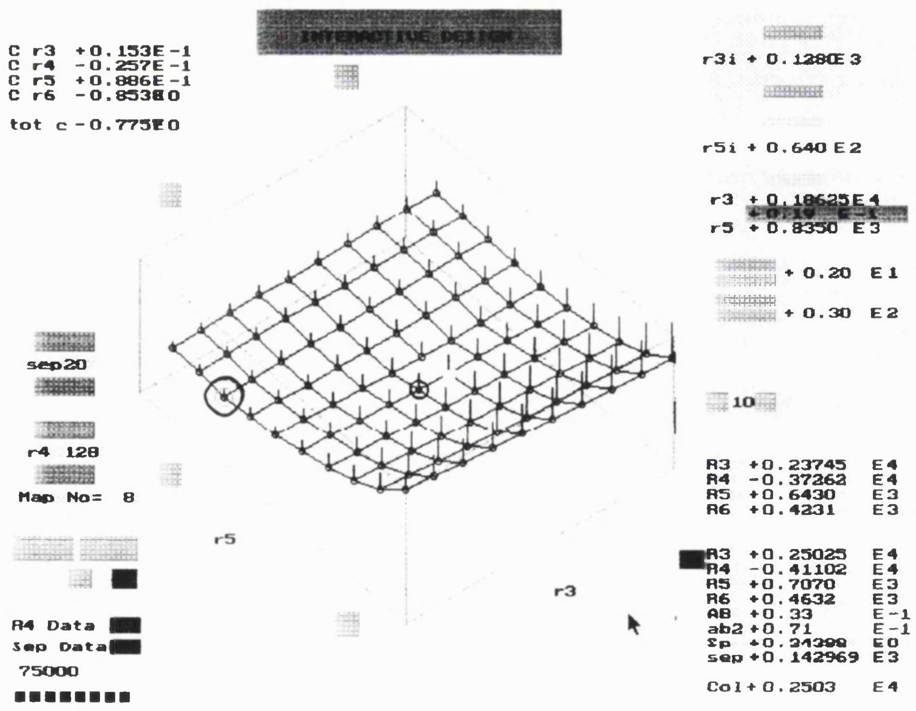
```
C r3  +0.00080
C r4  +0.00080
C r5  +0.466E -1
C r6  -0.465E -1
tot c +0.1   E -3

r3i + 0.10485760
r5i + 0.327680
r3  + 0.100000E6
r5  + 0.16960E4
+ 0.320 E2
+ 0.30  E2

R3  +0.1000000E6
R4  +0.836160  E5
R5  +0.16960   E4
R6  +0.16932   E4

R3  +0.1000000E6
R4  +0.836160  E5
R5  +0.16960   E4
R6  +0.16932   E4
AB  +0.7417    E1
ab2 +0.7402    E1
Sp  +0.36594   E0
sep +0.143005  E3

Col+0.100000  E6
```

Figure 2.10 : Stage 1 in Interactive Design: View of Parameter Space Map



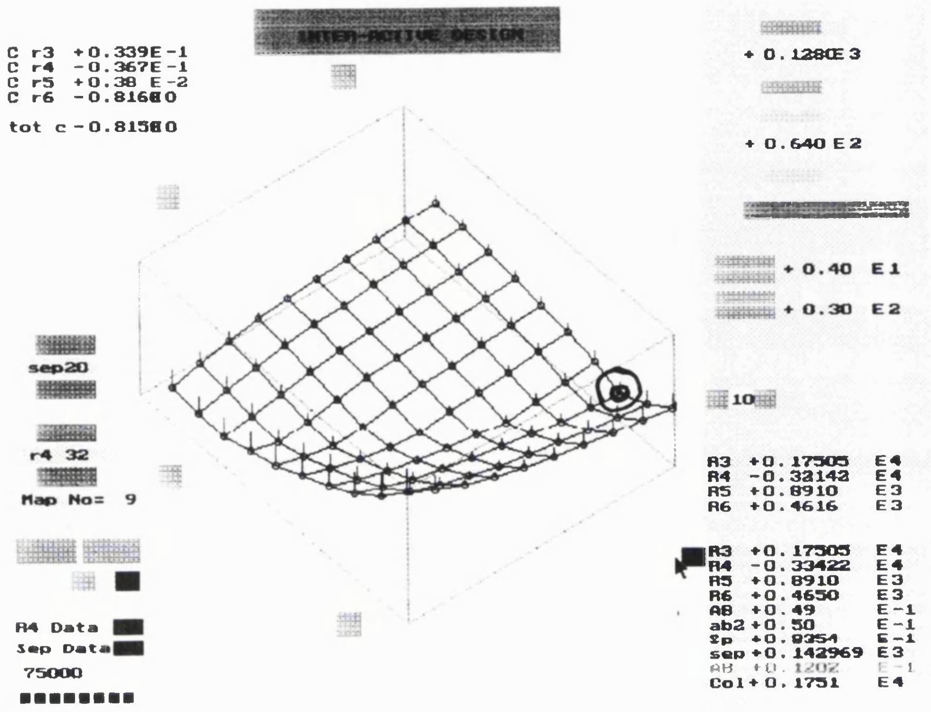
```
C r3  +0.151E-1
C r4  -0.308E-1
C r5  +0.351E-1
C r6  -0.715E0
tot c -0.695E0

+ 0.128E 3
+ 0.640 E 2
+ 0.40  E 1
+ 0.30  E 2

R3  +0.30305   E4
R4  -0.34062   E4
R5  +0.10830   E4
R6  +0.6146    E3

R3  +0.25185   E4
R4  -0.32782   E4
R5  +0.8270    E3
R6  +0.4951    E3
AB  +0.478     E0
ab2 +0.456     E0
Σp  +0.23127   E0
sep +0.142969  E3

Col+0.2519     E4
```

Figure 2.11 : Stage 2 in Interactive Design: View of Parameter Space Map



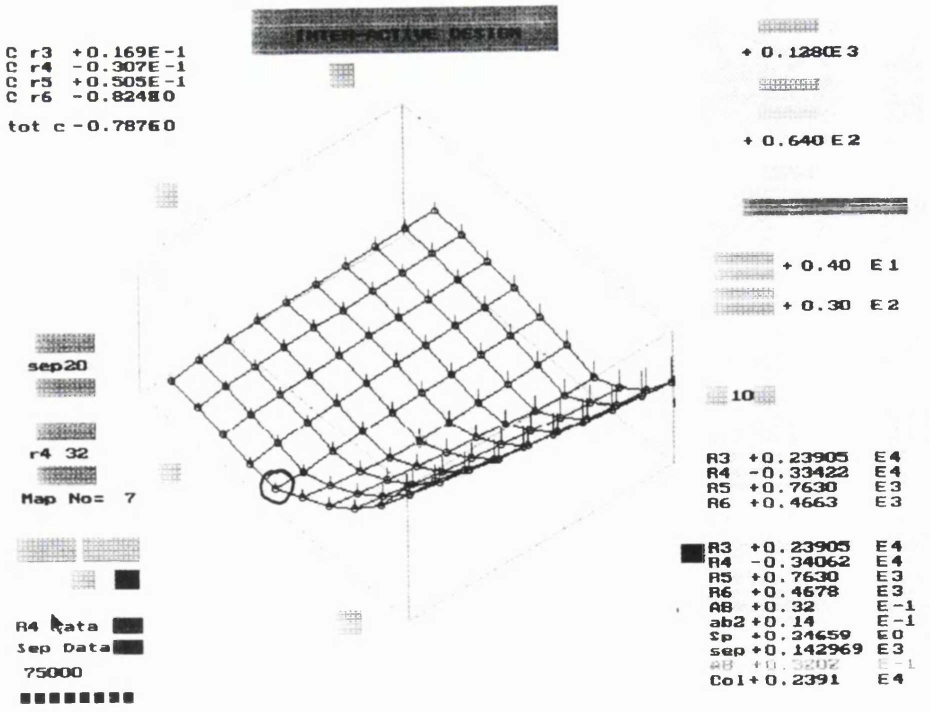
```
C r3  +0.153E-1
C r4  -0.257E-1
C r5  +0.886E-1
C r6  -0.853E0
tot c -0.775E0

r3i + 0.128E 3
r5i + 0.640 E 2
r3  + 0.18625E 4
      + 0.19   E-1
r5  + 0.8350  E 3
+ 0.20  E 1
+ 0.30  E 2

R3  +0.23745   E4
R4  -0.37262   E4
R5  +0.6430    E3
R6  +0.4231    E3

R3  +0.25025   E4
R4  -0.41102   E4
R5  +0.7070    E3
R6  +0.4632    E3
AB  +0.33      E-1
ab2 +0.71      E-1
Σp  +0.24399   E0
sep +0.142969  E3

Col+0.2503     E4
```

Figure 2.12 : Stage 3 in Interactive Design: View of Parameter Space Map



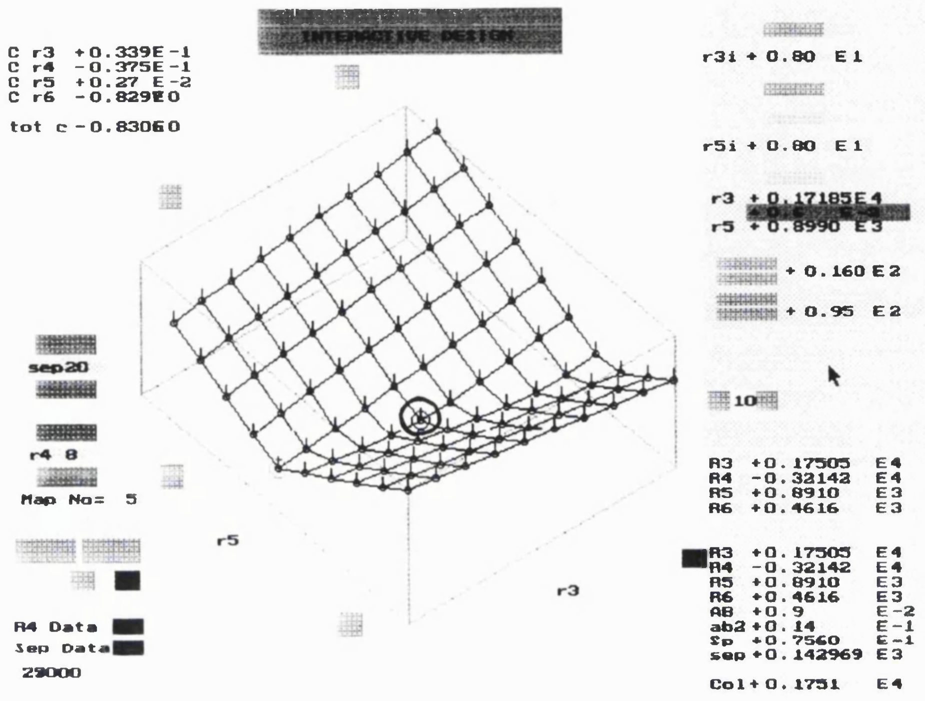
```
C r3  +0.339E-1
C r4  -0.367E-1
C r5  +0.38 E-2
C r6  -0.8160 0
tot c -0.8158 0

sep20
r4 32
Map No=  9
R4 Data
Sep Data
75000

+ 0.1280E 3
+ 0.640 E 2
+ 0.40  E 1
+ 0.30  E 2
10

R3  +0.17505   E4
R4  -0.32142   E4
R5  +0.8910    E3
R6  +0.4616    E3

R3  +0.17505   E4
R4  -0.33422   E4
R5  +0.8910    E3
R6  +0.4650    E3
AB  +0.49      E-1
ab2 +0.50      E-1
Sp  +0.8354    E-1
sep +0.142969  E3
AB  +0.1202    E-1
Col +0.1751    E4
```

Figure 2.13 : Stage 4 in Interactive Design: View of Parameter Space Map



```
C r3  +0.169E-1
C r4  -0.307E-1
C r5  +0.505E-1
C r6  -0.8248 0
tot c -0.7876 0

sep20
r4 32
Map No=  7
R4 Data
Sep Data
75000

+ 0.1280E 3
+ 0.640 E 2
+ 0.40  E 1
+ 0.30  E 2
10

R3  +0.23905   E4
R4  -0.33422   E4
R5  +0.7630    E3
R6  +0.4663    E3

R3  +0.23905   E4
R4  -0.34062   E4
R5  +0.7630    E3
R6  +0.4678    E3
AB  +0.32      E-1
ab2 +0.14      E-1
Sp  +0.34659   E0
sep +0.142969  E3
AB  +0.3202    E-1
Col +0.2391    E4
```

Figure 2.14 : Stage 5 in Interactive Design: View of Parameter Space Map



```
INTERACTIVE DESIGN

C r3  +0.339E-1
C r4  -0.375E-1
C r5  +0.27 E-2
C r6  -0.829E0

tot c -0.8306 0

sep20
r4 8
Map No=  5

R4 Data
Sep Data
29000

r3i + 0.80  E1
r5i + 0.80  E1

r3  + 0.17185E4
r5  + 0.8990  E3

+ 0.160 E2
+ 0.95  E2

10

R3  +0.17505   E4
R4  -0.32142   E4
R5  +0.8910    E3
R6  +0.4616    E3

R3  +0.17505   E4
R4  -0.32142   E4
R5  +0.8910    E3
R6  +0.4616    E3
AB  +0.9       E-2
ab2 +0.14      E-1
Sp  +0.7560    E-1
sep +0.142969  E3

Col+0.1751     E4
```

Figure 2.15 : Stage 6 in Interactive Design: View of Parameter Space Map

The Inter-Opt program gives the user improved flexibility to explore the parameter-aberration space of a Houghton type camera design. Normally optimization of these systems involves correction of the following 6 primary aberrations: spherical, spherochromatism, coma, astigmatism, field curvature, and longitudinal chromatic. Also the designer must deal with six radii of curvature, plus at least three separations, in total 9 parameter dimensions. This adds up to a 15 dimensional space that must be explored, if we don't include a number of higher order aberrations. Optimization of Houghton systems is therefore a very complex process. However using Inter-Opt, longitudinal colour and field curvature are automatically corrected using one separation and two radii of curvature. The problem is reduced to a 10 dimensional space, and a directed search can proceed in this space where these two aberrations are close to zero.

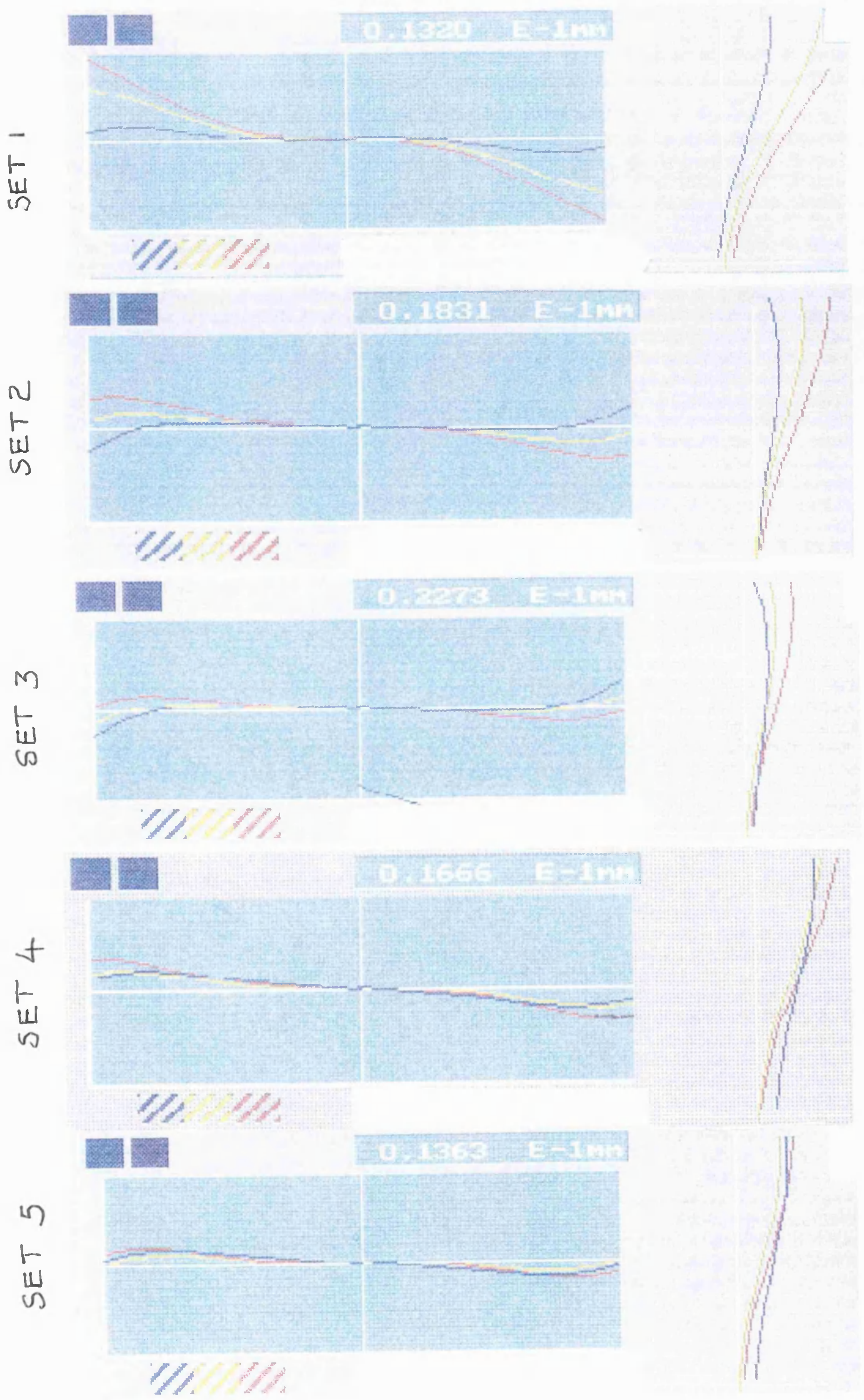


Figure 2.16 Series of On-axis Meridional HTanU Graphs for a Houghton Camera During the Interactive Design Process

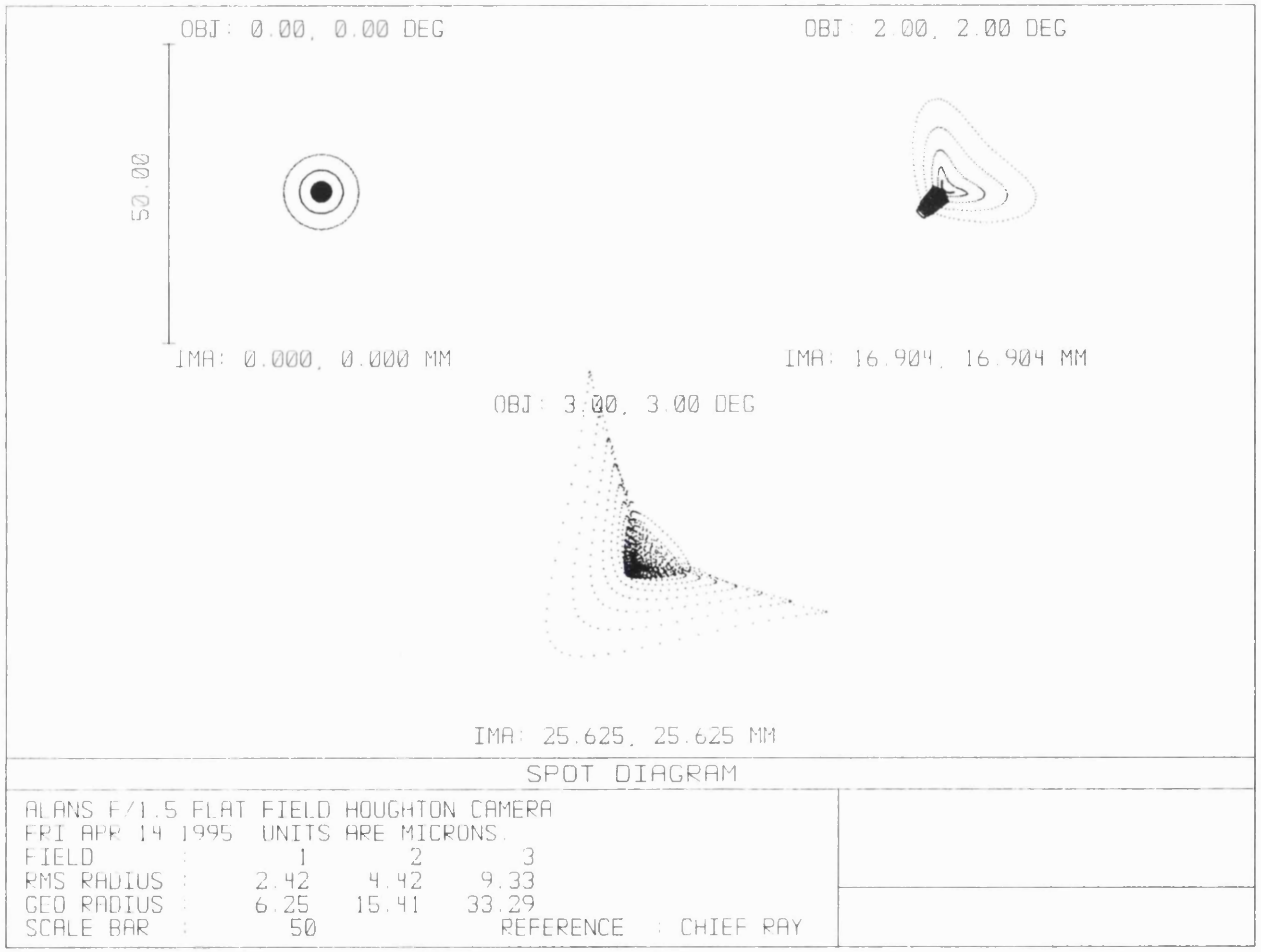
Using the new aberration plots, real time access to 4 dimensions of parameter space is possible, whilst at the same time viewing 3 dimensions of aberration space. Improved visualization of the characteristics of a particular camera form is the result, and good solutions can often be quickly located. The new parameter-space map concept is an integral part of the development of a user-interface that provides enhanced dialogue between computer and user during Houghton camera optimization. The improved dialogue takes the form of efficient feedback on the effects of simultaneously changing several parameters at once in a camera form.

2.3.9 Advantages of Improving User-Computer Dialogue During Optimization through the concept of 'Parameter-Space Maps'

The concept of allowing the user to maneuver through parameter-aberration visualizations in real time, as if flying over a mountainous region of the earth, is a very efficient optimization method. This parameter-space flying metaphor is very natural to the human user, and is made up of two key attributes, transferring the multidimensional problem into 3 dimensions of space and one of time, and also allowing interaction with this space in the form of basic maneuvering abilities just as in the real world. All the parameters that need changing, and aberrations that are to be quantified, must be represented in this virtual world. Parameters are represented as space dimensions horizontally, or dimensions changing in real time. Additionally aberrations are represented as height or length of lines in this imaginary world. The user is immediately familiar with the concept of living and moving in a three dimensional world in real time. This familiarity makes the dialogue between user and computer much more complete and simple to understand than might otherwise have been the case. Importantly as a result of using concepts that the user is familiar, feedback of information during optimization is improved in both quality and quantity. The user can scan through the properties of hundreds of thousands of cameras in only a few minutes. During this process carefully eliminating areas of the terrain where probing searches are proving unpromising. The user is able to scan large volumes of information very quickly, and identify trends in the data that send him off to promising regions of this design space. Feedback is enhanced in a specific visual way as a result of the new representation technique. General optimization problems emanate from the need to correct a number of aberrations simultaneously, and identification of relationships is especially difficult using ordinary optimization methods. These relationships become visible when flying around parameter space, and the user rapidly conceptualizes the complex nature of the situation more fully. The problem of camera optimization is transposed into the real time domain with the new parameter-space maps, and as a result the user has a greater degree of control of the decisions being made during design. Also the visual representation of the properties of a large number of cameras aids comprehension of information that would be very difficult and time consuming to plot out by hand. Interactive design is enhanced with the new maps, and the user has a great degree of freedom to make such design changes as are found to be necessary. Importantly the technique places the human user in the driving seat, allowing him to direct the optimization process in a very precisely controlled manner.

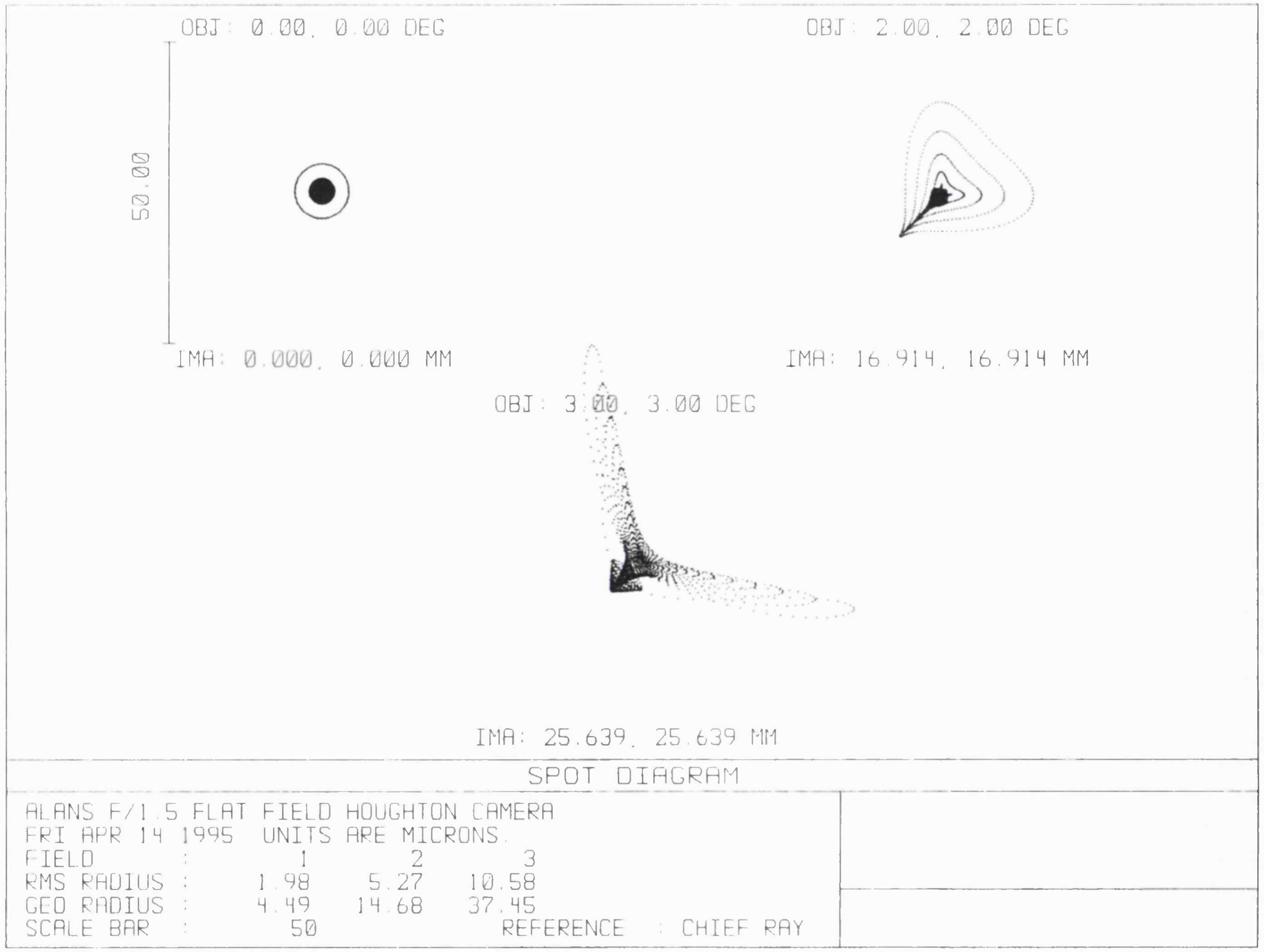
2.4 Efficiency Advantages of the New User-Interface Over Other Optimization Techniques

It is important to compare and contrast the relative efficiency of one method of optimization with another, in order to highlight the relative merits of any particular method. In this section we present blur spot data for a test system optimized using Inter-Opt to demonstrate its capabilities in this regard.



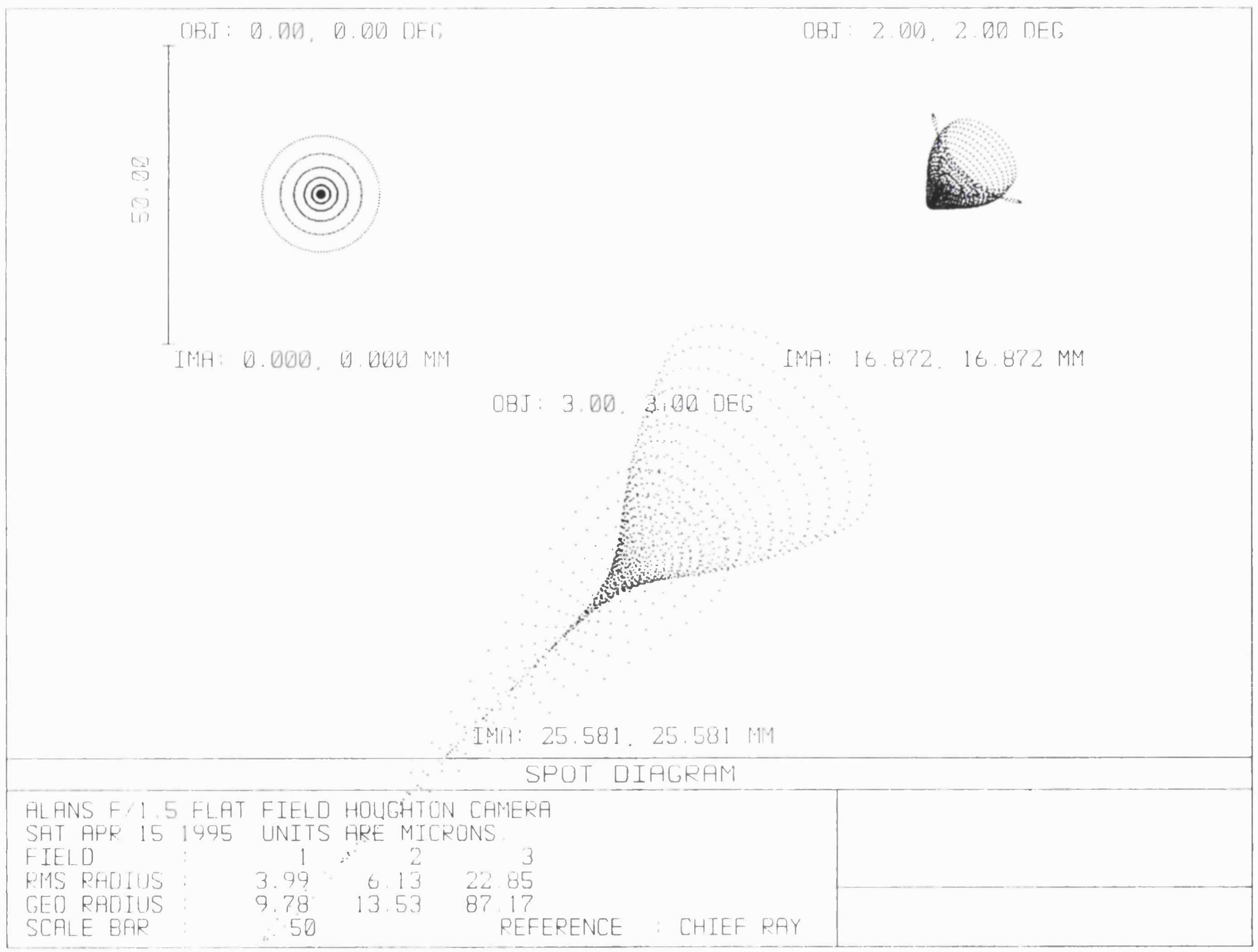
```
ALANS F/1.5 FLAT FIELD HOUGHTON CAMERA
FRI APR 14 1995  UNITS ARE MICRONS.
FIELD        :      1       2       3
RMS RADIUS   :    2.42    4.42    9.33
GEO RADIUS   :    6.25   15.41   33.29
SCALE BAR    :     50           REFERENCE  : CHIEF RAY
```

Figure 2.17 : Blur Spot Diagrams at 5500 Angstroms for a Houghton Camera



```
ALANS F/1.5 FLAT FIELD HOUGHTON CAMERA
FRI APR 14 1995  UNITS ARE MICRONS.
FIELD       :      1       2       3
RMS RADIUS  :    1.98    5.27   10.58
GEO RADIUS  :    4.49   14.68   37.45
SCALE BAR   :     50           REFERENCE  : CHIEF RAY
```

Figure 2.18 : Blur Spot Diagrams at 8000 Angstroms for a Houghton Camera



```
ALANS F/1.5 FLAT FIELD HOUGHTON CAMERA
SAT APR 15 1995   UNITS ARE MICRONS.
FIELD       :         1        2        3
RMS RADIUS  :      3.99     6.13    22.85
GEO RADIUS  :      9.78    13.53    87.17
SCALE BAR   :        50          REFERENCE   : CHIEF RAY
```

Figure 2.19 : Blur Spot Diagrams at 3200 Angstroms for a Houghton Camera

Directly comparing design methods is difficult to achieve in practice as optimization methods, by their nature, are exceptionally flexible procedures. Use of a particular method in any case is only one example of a multiple of possible variants of its application. Perhaps a better way of judging the relative merits of different techniques is to compare the aberration sizes in similar designs that they produce. A number of spectrograph camera designs are given in later chapters that were entirely optimized using the Inter-Opt software. Additionally in this section we present the results of an optimization run for a very high aperture (f-number 1.6) 480 mm focal length camera. The imaging results are given in the form of spot diagrams at a variety of field positions.

Comparing the imaging performance of this camera with similar published cameras, indicates that the camera compares favorably with similar or better aberration residuals. The blur spot diagrams in Figures 2.17, 2,18 and 2,19 show images at different field positions for 5000, 8000 and even 3200 Angstroms. The calculated average RMS image diameter from these diagrams is 14.9 microns over all the field angles and wavelengths presented here. Whilst the camera by no means represents a fully optimized solution, it exhibits highly achromatic properties and small levels of aberrations for a camera of this size and aperture. The camera design was optimized in only half an hour. During this time the author scanned through more than 30,000 separate camera systems using the parameter-space map visualizations. The Inter-Opt software offers a very efficient optimization alternative to automatic methods therefore, providing similar results in shorter time spans. The software is highly efficient in terms of the optical designer's time spent exploring one form of camera, and can free time to look at other promising options.

	Radius of Curvature	Separation	Media
Surface 1 = Echelle as Stop	Infinite	0	Air
Surface 2	1670.509	640.00	Air
Surface 3	-3278.151	46.840	Fused Silica
Surface 4	907.005	142.969	Air
Surface 5	459.785	15.866	Fused Silica
Surface 6	-1041.797	1377.607	Mirror
Surface 7	-124.619	-488.92	Air
Surface 8	-661.914	-22.130	Fused Silica
Image Plane	Infinite	-21.609	Air

Table 2.1 : The Parameters of a Houghton Camera of 480mm Focal Length

The result is that the designer can explore the applicability of a wider range of solutions to meet a particular specification. The designer can quickly evaluate and discard, those solution forms that look unpromising, and concentrate effort instead on design types that are most suitable. The likelihood of finding the ideal form is therefore higher using the new method, because of the improved optimization efficiency inherent in the new interactive design methodology.

2.5 Validating the Ray Tracing Results from the Inter-Opt Code

2.5.1 The Need for Data Validation

For a new software tool it is important to be able to validate the data and information it produces. The results of extensive testing of the Inter-Opt software are presented in this section. Comparison of data output from the program with another commercial ray tracing program has demonstrated the high precision and accuracy of the results. The comparison program is widely used for optical design and is called Zemax [80]. Additionally data output has been checked against the standard optics text entitled Modern Optical Engineering [56].

Before describing the validation checks in detail, we note that two minor bugs have been found in the Inter-Opt code that are reported in the user manual in Appendix B. These errors do not relate to the basic ray tracing equations, but to slight inaccuracies in the way the data is displayed. The Inter-Opt software has been used to ray trace more than 100 optical systems, and these display errors have not been found to be significant. However an appraisal of the effects of these two errors on system evaluation is given in the following sections. In the next section we directly compare the ray tracing data from the Zemax and Inter-Opt programs for a photographic triplet lens.

2.5.2 Checking the Precision of the Inter-Opt Ray Tracing Equations for a Photographic Triplet Lens

On Page 322 of Modern Optical Engineering [56] data for a photographic triplet lens is given, including ray tracing data and third order aberration calculations. This lens has been ray traced at a wavelength of 4500 Angstroms using Inter-Opt to check the precision of the ray tracing results. In the text the equivalent focal length (EFL) at this wavelength is given as 100.004 mm and the entrance pupil position is calculated to be 24.905 mm to the right of surface 1. Additionally the back focal length (BFL) is shown as 79.336 mm in this text. In Table 2.2 we show the parameters of this photographic triplet, whilst in Figure 2.20 we show paraxial ray tracing results, in millimeters, as calculated by Inter-Opt. The Inter-Opt ray tracing results shown in Figure 2.20 are in accordance with the results in Text 1, and the precision of the paraxial data displayed here is found to be greater than +/- 0.5 micron. Figure 2.20 shows single ray trace data for surface 2 onwards, including individual surface intersection heights and paraxial ray angles. Note that in Inter-Opt surface 1 is the entrance pupil, whilst surface 2 is the first physical surface of the system.

Surface	Radius	Separation	Material
Surface 1	40.940	0.0	Air
Surface 2	Infinite	8.740	SSK4
Surface 3	-55.650	11.050	Air
Surface 4	39.750	2.780	SF12
Surface 5	107.560	7.630	Air
Surface 6	-43.560	9.540	SSK4
Image Plane	Infinite	73.336	Air

Table 2.2: The Parameters of a Photographic Triplet Test System

```
E.F.L.   100.004        Paraxial Ray Trace       B.F.L     79.336
Number of Surfaces =6    Physical Stop =   4
  Surface         Ray Y              nu           nu'

    2            18.500000      0.000000    -0.278810
    3            16.993010     -0.278810    -0.278810
    4            13.912154     -0.278810    -0.116565
    5            13.715641     -0.116565     0.107371
    6            14.534885      0.107371     0.023994
    7            14.676447      0.023994    -0.184992




Entrance Pupil Radius      18.500
Entrance Pupil Position    24.905
Focal Ratio =               2.703
```

Figure 2.20: Inter-Opt Paraxial Ray Tracing Results at 4500 Angstroms for a Photographic Triplet Lens

Initially Inter-Opt gave a paraxial back focal length for the system of 79.3357 mm, which did not agree with that given in the original text. This disagreement was subsequently resolved as the old version of the book was found to be in error, and in the revised version the text is now in agreement with Inter-Opt. An important option in any modern ray tracing program is the calculation and display of Seidel third order aberration contributions. On page 322 of Modern Optical Engineering [56] the results of calculation of the Seidel coefficients for this lens design are given. Whilst for comparison in Figure 2.21 we show the Seidel coefficients and corresponding transverse aberrations as calculated by the Inter-Opt software, again for a wavelength of 4500 Angstroms. In this Figure, individual surface Seidel coefficients are given for surface 2 onwards, and the last but one line in this figure shows total Seidel coefficients for the system. Whilst the last line shows these sums converted to transverse aberration measure.

```
                 Third Order Aberration Contributions
Spher       Coma       Astig       Petz       Dist      Long.Chrom
S1          S2          S3          S4          S5         C1

 0.40282    0.0872    0.01891    0.19937    0.04730   -1.69474
 0.22744   -0.2804    0.34590   -0.00000   -0.42657   -0.96149
-0.91789    0.6145   -0.41147   -0.15128    0.37678    2.50457
-0.42177   -0.3143   -0.23431   -0.21179   -0.33250    2.11252
 0.08393    0.1249    0.18613    0.07588    0.39018   -0.71456
 0.78158   -0.2319    0.06885    0.18837   -0.07634   -1.55817

 0.15611    0.0000   -0.02598    0.10055   -0.02116   -0.31188
-0.4219   -0.00002    0.07022   -0.27176    0.05719   -0.31188
```

Figure 2.2.1: Seidel Coefficients as Calculated by the Inter-Opt Software

The results shown in Figure 2.21 agree with the results in Modern Optical Engineering [56] to an equivalent accuracy of +/- 1 micron in the transverse aberration measure. The slight discrepancy between these results is due to the use of materials in the Inter-Opt modelled lens that are a fairly close, but not an exact match, to the characteristics given in the text. In section 2.5.3 below comparative data are given for a Schmidt system with Fused Silica elements. Employing known optical media in this system allows the use of identical optical characteristics for both programs. As a result the Seidel coefficient results agree to a much higher level of precision, around 5 decimal places.

2.5.3 Checking the Precision of the Inter-Opt Ray Tracing Equations for a Schmidt Camera

In this section we describe ray tracing tests for a catadioptric Schmidt system in order to check the aspheric surface calculations for agreement with Zemax. The system we ray traced is shown in Table 2.3. Both Inter-Opt and Zemax calculated identical values for both the BFL and EFL, relatively -19.696 mm and -699.535 mm. Again these results are for a wavelength of 4500 Angstroms. The precision was found to be within +/- 0.5 microns on these results, and they are shown in Figure 2.22 for the Inter_Opt program. Figures 2.23 and 2.24 show real ray tracing data for the Schmidt system for a marginal ray at a height of 10mm and field angle of 3 degrees. Both Figures give ray intersection data for the same real ray traced through the camera, with intersection heights and angles in the same arc-tangent format. In the Zemax ray tracing data, surface 1 corresponds with surface 2 for the Inter-Opt ray data. Also please note that the programs use different coordinate reference letters. On the Zemax data output X-direction cosine = Z on Inter-Opt, Y-direction cosine = Y on Inter-Opt, and Z direction cosine = X on Inter-Opt. Additionally x distance on Zemax = z on Inter-Opt, whilst the y distances are the same, and z on Zemax = x on Inter-Opt. These results show a high degree of agreement, and the Inter-Opt ray tracing is demonstrated to be accurate to the fifth decimal place in the y co-ordinate direction. These equations show that the real ray tracing routines have total errors that amount to less than 1/10 of a micron in the image plane. Additionally these results demonstrate that the basic Inter-Opt real ray tracing routines can deal with reflecting systems that contain aspheric surfaces as in this case.

	Radius of Curvature	Separation	Media
Surface 1 = Stop	Infinite		Air
Surface 2	Infinite	436.8	Air
Surface 3 A4 = A6 =	-33657.171 +1.435e-10 +3.591e-17	21.000	Fused Silica
Surface 4	-1528.800	1162.000	Air
Surface 5	-295.135	-698.000	Air
Surface 6	1471.698	-49.166	Fused Silica
Image Plane	Infinite	-19.696	Air

Table 2.3: The Parameters of a Schmidt System

```
E.F.L.  -699.535          Paraxial Ray Trace        B.F.L    -19.696
Number of Surfaces =6      Physical Stop =    1
  Surface             Ray Y             nu           nu'

     2             117.000000      0.000000      0.000000
     3             117.000000      0.000000      0.000000
     4             117.000000      0.000000     -0.001618
     5             115.119403     -0.001618     -0.152219
     6               8.870254     -0.152219     -0.166212
     7               3.294267     -0.166212     -0.167254



Entrance Pupil Radius      117.000
Entrance Pupil Position      0.000
```

Figure 2.22: Inter-Opt Paraxial Ray Trace Data at 4500 Angstroms for a Schmidt System

```
         INTER-OPT v2.3.6 ..Interactive Optical Design

                          Real Ray Trace
Number of Surfaces =6
  Surface    x           y           z           X           Y           Z
          0.000000  10.000000   0.000000   0.998630   0.052336   0.000000
     2    0.000000  10.000000   0.000000   0.998630   0.052336   0.000000
     3    0.000000  32.891718   0.000000   0.999362   0.035710   0.000000
     4   -0.016629  33.641521   0.000000   0.998653   0.051880   0.000000
     5   -2.883897  93.858865   0.000000   0.997483   0.070902   0.000000
     6   -3.330460  44.212891   0.000000   0.995375   0.096070   0.000000
     7    0.539378  39.841087   0.000000   0.988142   0.153542   0.000000
     8    0.000000  36.696793   0.000000   0.988142   0.153542   0.000000
```

Figure 2.23: Inter-Opt Real Ray Trace Data at 4500 Angstroms for a Schmidt System

```
Ray Trace Data
LENS HAS NO TITLE.
Sun Aug 06 1995
Units are Millimeters.

Wavelength          :     0.450000 microns
X Field Coord (Hx) :      0.000000
Y Field Coord (Hy) :      1.000000
X Pupil Coord (Px) :      0.000000
Y Pupil Coord (Py) :      1.000000

Real Ray Trace Data:

Surf        X-coord      Y-coord       Z-coord    X-cosine    Y-cosine    Z-cosine
OBJ        Infinity     Infinity      Infinity   0.0000000   0.0523360   0.9986295
  1     0.0000E+000  1.0000E+001   0.0000E+000   0.0000000   0.0523360   0.9986295
  2     0.0000E+000  3.2892E+001   0.0000E+000   0.0000000   0.0357104   0.9993622
  3     0.0000E+000  3.3642E+001  -1.6629E-002   0.0000000   0.0518804   0.9986533
  4     0.0000E+000  9.3859E+001  -2.8839E+000   0.0000000   0.0709017   0.9974833
  5     0.0000E+000  4.4213E+001  -3.3305E+000   0.0000000   0.0960696   0.9953746
  6     0.0000E+000  3.9841E+001   5.3927E-001   0.0000000   0.1535396   0.9881425
  7     0.0000E+000  3.6697E+001   0.0000E+000   0.0000000   0.1535396   0.9881425
```

Figure 2.24: Zemax Real Ray Trace Data at 4500 Angstroms for a Schmidt System

```
                  Third Order Aberration Contributions
Spher      Coma       Astig       Petz       Dist     Long.Chrom
S1         S2         S3          S4         S5         C1

 0.00000   0.0000    0.00000    0.00000    0.00000    0.00000
 0.00000   0.0000    0.00000    0.00000    0.00900    0.00000
-0.10015  -0.0202   -0.00359    0.00035   -0.00945    0.09827
 0.10258   0.0039    0.00015   -0.04919   -0.00189    0.00000
-0.00991   0.0072   -0.00537    0.04047   -0.02584    0.17865
 0.00851   0.0083    0.00817    0.00812    0.01596   -0.09205

 0.00102  -0.0007   -0.00063   -0.00025   -0.01221    0.18487
 0.0030  -0.00209   -0.00189   -0.00074   -0.03650    0.18487
```

Figure 2.25: Seidel Coefficients as Calculated by Inter-Opt for a Schmidt System

```
Listing of Aberration Coefficient Data
LENS HAS NO TITLE.
Sun Aug 06 1995

Wavelength       :         0.4500 microns
Petzval radius   :    1.5123E+005
Optical Invariant:         6.1317

Seidel Aberration Coefficients:

WARNING: Seidel aberrations may be inaccurate for non-standard surfaces!

Surf SPHA  S1  COMA  S2  ASTI  S3  FCUR  S4  DIST  S5  CLA (CL)  CTR (CT)
STO   0.00000   0.00000   0.00000   0.00000   0.00000   0.00000   0.00000
  2   0.00000   0.00000   0.00000   0.00000   0.00900   0.00000   0.00000
  3  -0.10015  -0.02029  -0.00359   0.00035  -0.00945   0.00000   0.00000
  4   0.10258   0.00396   0.00015  -0.04919  -0.00189   0.00000   0.00000
  5  -0.00991   0.00729  -0.00537   0.04047  -0.02584   0.00000   0.00000
  6   0.00850   0.00834   0.00817   0.00811   0.01596   0.00000   0.00000
IMA   0.00000   0.00000   0.00000   0.00000   0.00000   0.00000   0.00000
TOT   0.00102  -0.00070  -0.00063  -0.00025  -0.01221   0.00000   0.00000
```

Figure 2.26: Seidel Coefficients as Calculated by Zemax for a Schmidt System

In order to check that the third order Seidel equations could accurately cope with these two conditions, in Figures 2.25 and 2.26 we show the results as calculated by Inter-Opt and Zemax for the Schmidt system. Again the Inter-Opt data compares favorably with Zemax, showing identical precision to the 6th decimal place. This section has demonstrated that the ray tracing calculations of both Inter-Opt and Zemax agree to a high level of precision. These results indicate that the ray tracing equations within Inter-Opt are working reliably.

2.5.4 Validation of The Inter-Opt Graphical Displays for a Schmidt System

We have now established the accuracy of the ray tracing equations that are the basis of the different options available in the Inter-Opt program. Additionally it is important to test the graphical displays for accuracy, in order to ensure reliable indication of the performance of any optical system. Inter-Opt provides four kinds of graphical displays of information: basic ray intercept plots, longitudinal focus shift plots, blur spot diagrams, and 3 dimensional aberration landscape views. Of these, the standard Zemax ray tracing package can be used to check the validity of all but the last kind which is unique to Inter-Opt. However these latter displays are known to give accurate optimization results, as they provide camera designs with a high degree of aberration correction.

In Figures 2.27 and 2.28 we show comparative transverse ray intercept plots for the Schmidt design from the previous section. Figure 2.28 shows graphs as given by the Zemax software, whilst Figure 2.27 shows the corresponding graphs from Inter-Opt. In both Figures curves are drawn for identical wavelengths, these being 3000, 4500 and 7000 Angstroms. These are represented by blue, green and red colours in these plots. Comparison of the two graphs shows that the results are almost identical, except for a minor error mentioned in the help manual. This refers to the fact that the Inter-Opt curves do not quite extend to the end of the graphs, and ray intercept plots are shown slightly compressed in the aperture direction. This latter error does not effect the users comprehension of the aberration sizes however, and is not significant in this context. Another error mentioned in the help manual concerns the longitudinal change in focus with aperture plots, which do not work for curved focal plane systems. This could potentially be more serious as it introduces a false aberration that the user may not be aware of. It is prudent therefore to recognise that with near field flat focal plane systems, as in the cameras in this thesis, the error will not be significant. We note that the shapes and crossings of different wavelengths on the aberration graphs are near identical between these two programs, validating the plotting routines used in Inter-Opt. Also note that the scales are similar, but not identical on these plots. On the Zemax plots 10 microns is represented by 10.5mm +/- 0.5mm. Whilst on the Inter-Opt plots 10 microns is 9.5mm +/- 0.5 mm.

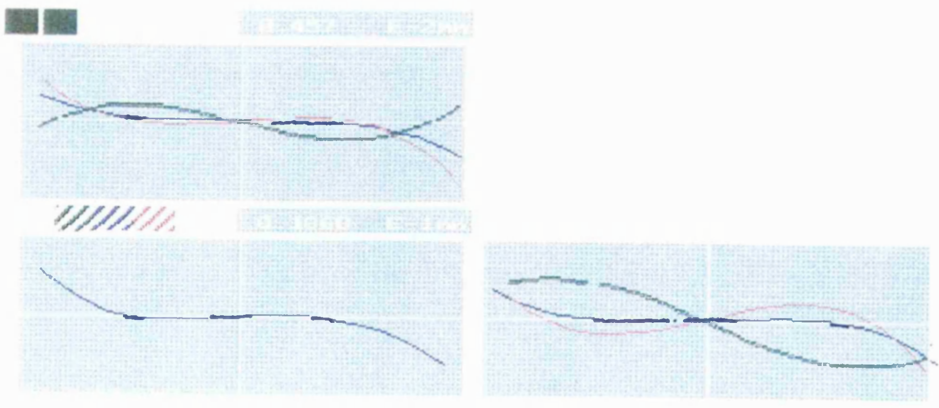


Figure 2.27: Inter-Opt Ray Intercept Plots for a Schmidt System

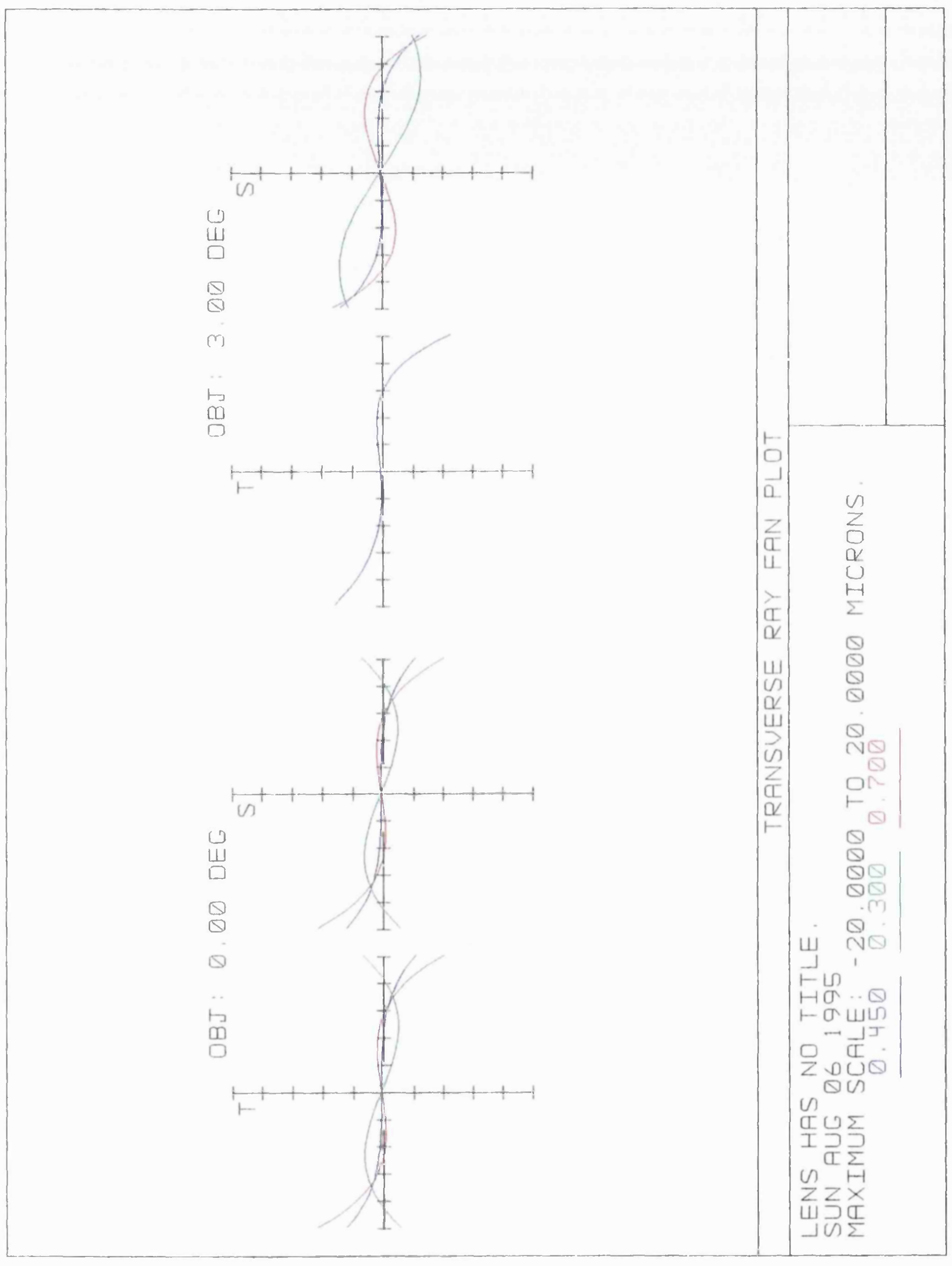


Figure 2.28: Zemax Ray Intercept Plots for a Schmidt System

Another information display mode that requires validation in the Inter-Opt software is the plotting of blur spot diagrams. These are commonly used to depict the true geometrical imaging performance of a particular system. Figure 2.30 shows scaled blur spot diagrams from Zemax, whilst Figure 2.29 shows the corresponding plots from the Inter-Opt program to allow direct comparison. These blur spot diagrams are for the same wavelengths as displayed in the ray intercept curves in Figures 2.27 and 2.28.

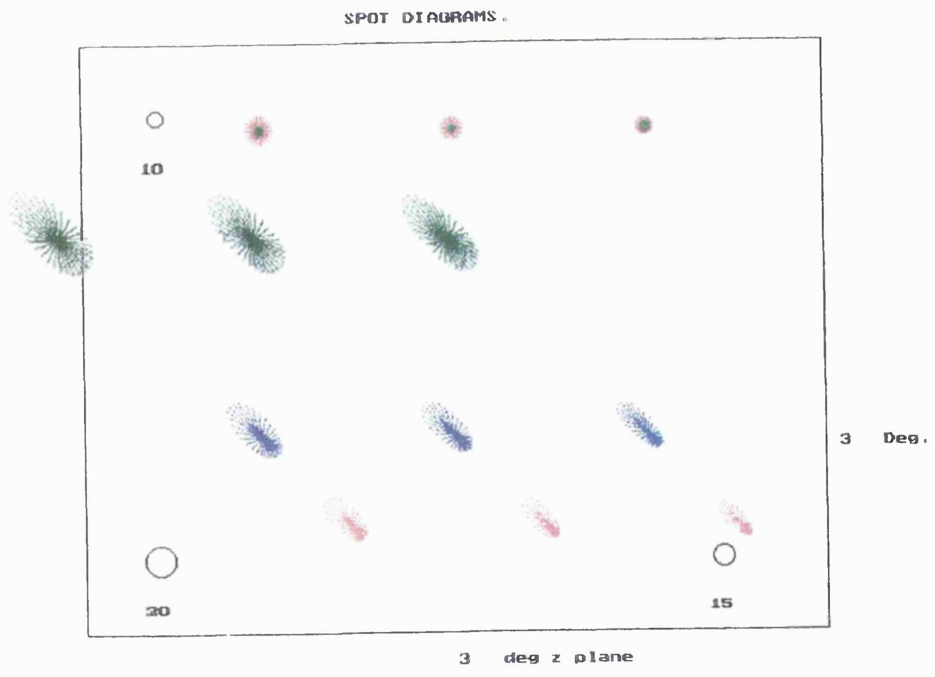
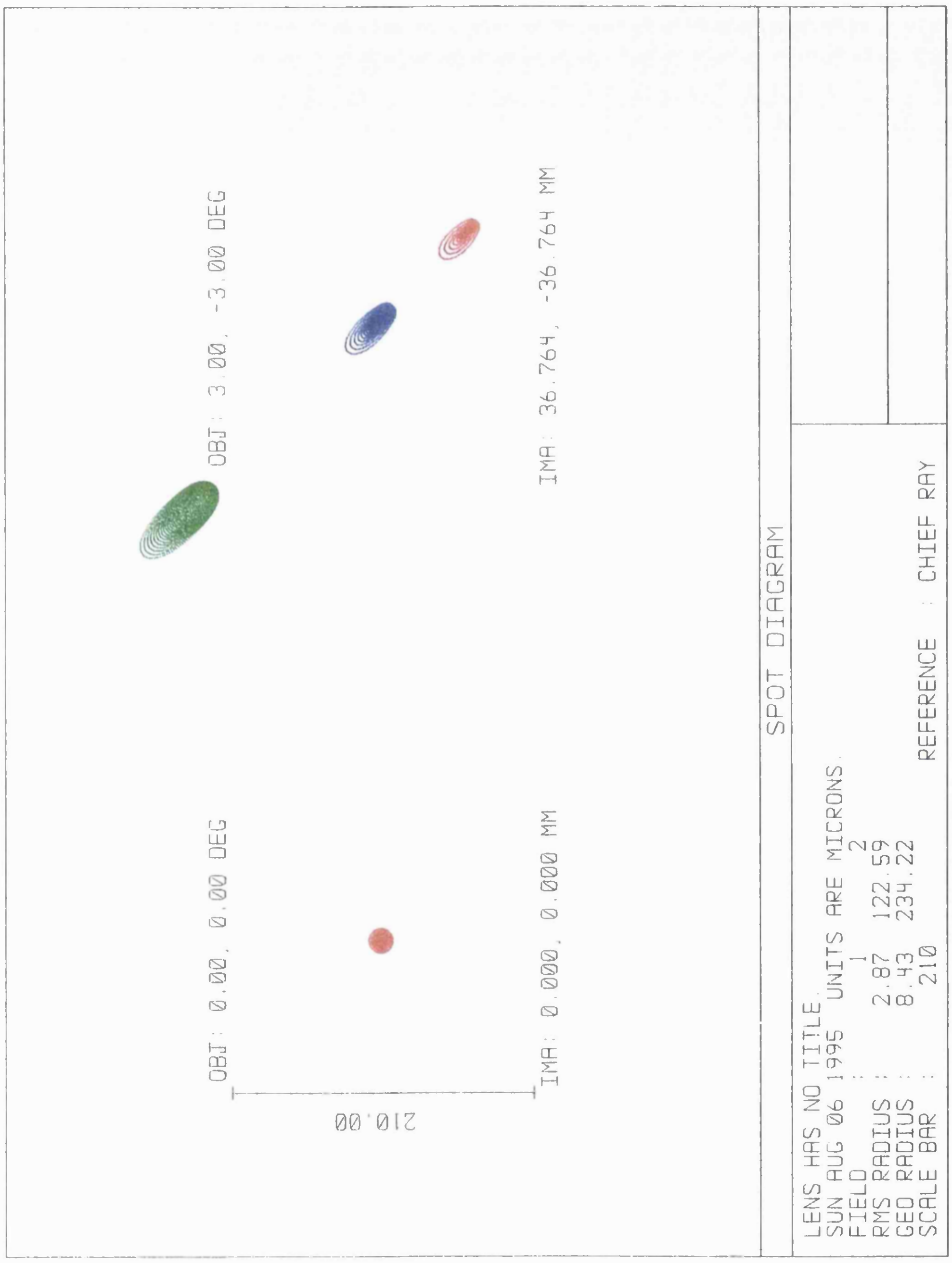


Figure 2.29: Inter-Opt Blur Spot Diagrams for a Schmidt System



```
LENS HAS NO TITLE.
SUN AUG 06 1995  UNITS ARE MICRONS.
FIELD       :      1       2
RMS RADIUS  :   2.87  122.59
GEO RADIUS  :   8.43  234.22
SCALE BAR   :    210             REFERENCE  : CHIEF RAY
```

Figure 2.30: Zemax Blur Spot Diagrams for a Schmidt System

Using Figures 2.29 and 2.30 we can directly compare the two programs output, and access the accuracy of the Inter-Opt routines. The scale on Figure 2.30 is 51.2 microns per 10mm +/ 0.5mm. Whilst the scale in Figure 2.29 is 50 microns per 10mm +/- 0.5mm. In Figure 2.30, the green spot (representing a wavelength of 3000 Angstroms) measures 14mm +/- 0.5 mm which corresponds to 71.6 +/- 2.5 microns. Whilst in Figure 2.29, the same spot measures 13.8mm +/- 0.5mm, or 69.0 +/- 2.5 microns. The two plots from Zemax and Inter-Opt show close agreement, with spot sizes and positions agreeing to within 2.5 microns, which is an accuracy of less than 4 percent on spot size. Note that the precision of aberration determination is dependent on the measuring scale, which in turn depends on the scale chosen for blur spot plots. The scale on these plots was chosen to show polychromatic blur spot positions. To obtain greater accuracy for an individual blur spot size, a larger display scale can be selected.

This work has demonstrated that the results provided by the blur spot plotting from the Inter-Opt software is accurate to a high degree of precision. The ray tracing plots provide accurate assessment of individual blur spot sizes. Additionally the relative positioning of multiple wavelength blur spots is accurate to a similar precision.

2.5.5 Ray Tracing Validation: Conclusions

This section has demonstrated that the Inter-Opt software provides reliable ray tracing data that is comparable with the existing Zemax optical design package. Direct comparison of the results produced by both packages has shown that the data they produce is in close agreement. Inter-Opt provides a high level of accuracy and precision in the data it provides. This has been demonstrated explicitly for two different kinds of optical system, a Schmidt camera and a photographic triplet lens.

Both numerical and graphical Inter-Opt output was compared closely with Zemax for the Schmidt camera ray tracing data. Additionally Inter-Opt has also been specifically tested against a particular photographic triplet design given in a well known text, and the data supplied was found to be accurate and reliable. Significant additional testing has been achieved during working use of the Inter-Opt software. These other checks include all the camera designs provided in this thesis, which were optimized using Inter-Opt. Subsequent independent testing of these various designs using Zemax and additionally optical testing (See Chapter 6) has further validated the programs accuracy and precision.

2.6 Conclusion and Future Work

This Chapter has dealt with the complexity of relationships in optical design, and the resulting difficulties in the process of system optimization. Automatic optimization methods are often used to help the optical designer to locate a suitable system. However automatic methods currently have drawbacks that limit their usefulness. The automatic method limits human participation in, and direction of, the optimization process. This the result of the user not being able to adequately visualize the complex effects of experimental design changes whilst using such techniques. Additionally optimization occurs in a prolonged cycle, often a few minutes, during which the user has limited opportunity to participate in the design process.

This chapter has described the development of a novel design methodology for a ray tracing program, and is based around a new type of user-interface that enhances user interaction during camera optimization. This new user-interface is based on two core analogies. These are required to aid dialogue between user and computer. The analogies are a fluid-lens whose properties are alterable in real time, and a parameter-space visualization through which the user can maneuver. Within these two core concepts, the author developed a number of novel design techniques. These included new aberration contribution graphs that allow the precise balancing of certain aberrations against the effects of others. Additionally the facility for real time alteration of apsheric coefficients and curvatures is offered as a unique option. This capability allowing precise quantification of the aberration effects of alterations to a fluid lens. Also a new way of calculating and representing spherochromatism, in terms of a line length within a 3 dimensional space, has been developed and tested specifically for the parameter-space map method of optimization. This method is highly efficient in removing the effects of this aberration completely from Houghton camera systems. The new parameter-space maps enable the interface user to fully visualize the 'landscape' of the multi-dimensional camera design space. As a result the user can move on promising vectors within this space. At the same time, the user is able to constantly alter course through this space, in order to avoid local mimima that hide the presence of a better global aberration minimum.

The Inter-Opt program significantly enhances quality of the dialogue between user and computer during system optimization. The user-interface provides volumes of visual information concerning many different camera forms. As a result the user is able to visualize the characteristics of a single design type more clearly. Additionally the user has the freedom to adapt the design strategy during the design process. Expeditious altering of the design approach, in response to what is found in this parameter space, is a unique ability inherent in the new optimization method.

Adaptive optimization strategies are the key advantage of the technique over merit function based methods. The interface user is able to constantly alter his/her approach in response to real time information feedback about the design in question.

This is achieved primarily because information is conveyed between user and computer in a ways the user can easily assimilate and control in real time. The Inter-Opt program facilitates interactive learning during system design, which is the key to allowing the optical designer to use his/her skills more naturally when researching novel optical systems.

Thesis work has demonstrated that the new user-interface methodology is highly efficient when applied to the optimization of camera systems. These techniques could also be applicable to other areas of optical design, and in the future it is likely that other optical design programs will include these types of interactive facilities. The author is now working on the design of an improved version of the program that will involve ray tracing of around 0.1 million camera systems whilst saving the results in the computer's memory. This represents a considerable computational task, taking around 25 minutes on a Pentium 100 Mhz IBM Personal Computer. Ray tracing of so many separate systems is being done to provide the user with the opportunity to explore far larger regions of the design space of a camera form, and in many more dimensions and detail than possible with the current program. At the same time new ideas for displaying this information in parameter-space visualizations are being implemented. These methods should further improve the quality of the dialogue between human user and computer during real time design, and so aid human participation in, and direction of, the optimization process.

Chapter 3

The High Resolution Optical/UV Spectrograph

3.1 Introduction

This Chapter introduces the design philosophy for the High Resolution Optical/UV Spectrograph (HROS). In this Chapter also, a brief consideration of the complex alternative design routes for HROS is presented. Additionally during this discussion we refer to the advantages and disadvantages of similar systems. Later sections go on to consider the feasibility of replacing monolithic cross dispersing prisms within such a spectrograph with mosaic prism tree arrangements. A light loss efficiency study is presented for these mosaic prism arrangements for the ultra-violet and infra-red regions of the electromagnetic spectrum. In the next section a brief review of the design specification for the HROS project is presented.

3.2 The High Resolution Optical/UV Spectrograph

The Optical Science Laboratory continues to work towards the design of the High Resolution Optical/UV Spectrograph (HROS) for the Gemini 8m telescope project. This instrument is now beginning detailed design at OSL, and is due for commissioning in the year 2000. Preliminary instrument flexure studies indicate that HROS can be located at the Cassegrain focus for the majority of observations [68]. It is anticipated that the resulting spectrograph flexure problems can be largely passively controlled through intelligent construction. Additionally any residual flexures can be removed using actively controlled actuators on the spectrograph collimator. For ultra high stability observations the spectrograph can be removed from Cassegrain station and placed on a fixed table with an optical fibre feed from the telescope Cassegrain focus. In Figure 3.1 below we show a schematic of the HROS instrument located at the Cassegrain focus of the Gemini telescope.

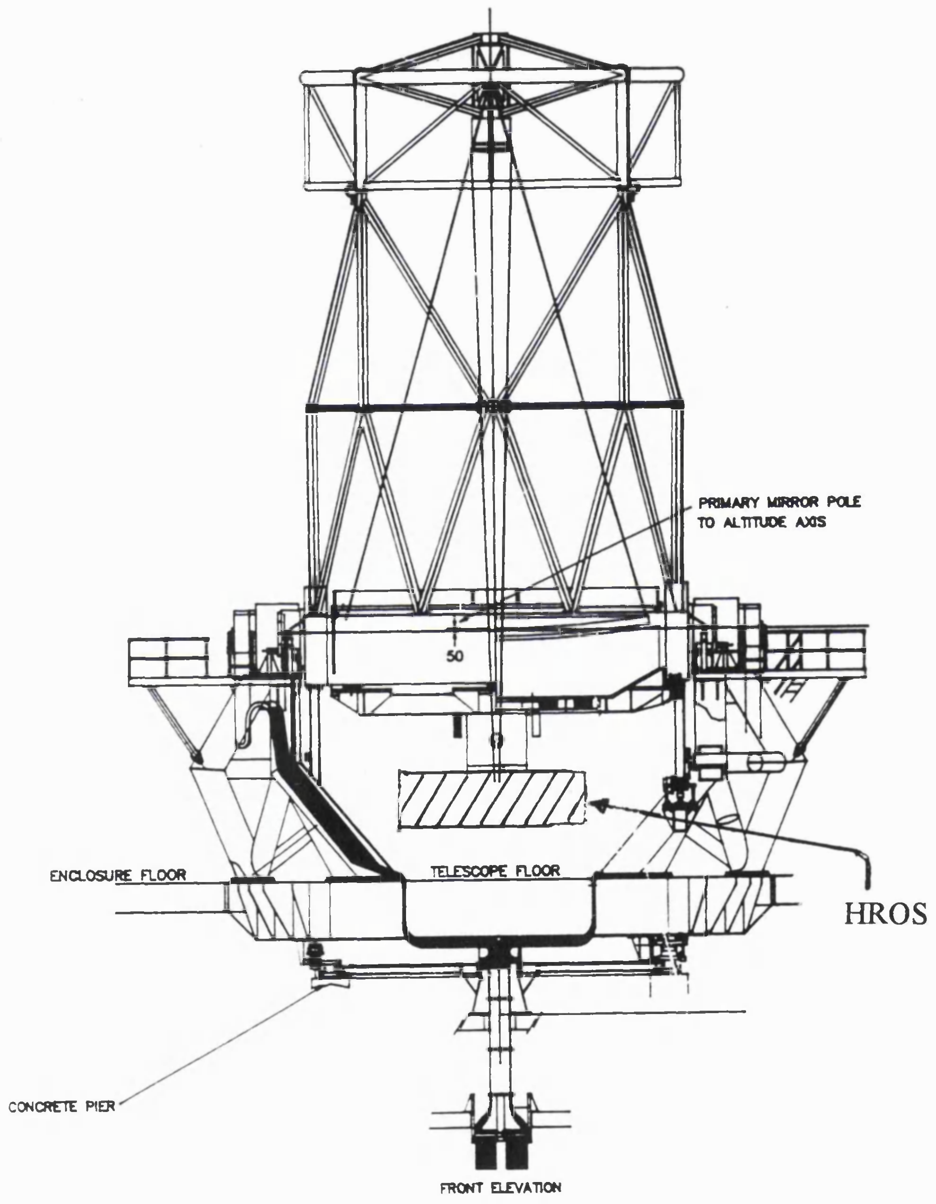


Figure 3.1 : Schematic of HROS at the Cassegrain Focus of the Gemini 8m Telescope

An important performance measure in any spectrograph is the product of slit width and resolution. Obtaining the highest possible factor here is the driving force behind much of the design work for an instrument like HROS. For a given resolving power, it is useful to think of ways that throughput could be increased for the HROS instrument. Increasing the beam size is one obvious way to achieve this, enabling an increase in slit width, hence greater throughput at the slit. However this indicates mosaicing of echelle gratings, which may not be practical for stability reasons in a Cassegrain instrument. Immersing the echelle in a dielectric medium such as fused silica, would also provide an effective increase in length of the echelle. This option is currently under investigation at OSL for use on the HROS instrument, but has negative implications for camera design. Echelle immersion requires a shorter focal length camera, by a factor equivalent to the index of refraction of the immersing material. Preliminary studies indicate that such a camera may prove feasible for the HROS instrument at a resolution of 120,000 [49]. Another way to increase instrument throughput is to use an image slicer, which enables more starlight to enter the slit jaws. Additionally significant developments in the field of adaptive optics are likely in the near future, and we can anticipate a reduction in the size of the seeing disc as a result.

Another possible way to gain a small amount (10 to 15 percent) in throughput is to employ an unobscured camera, which is often just a normal camera used in sub-aperture mode [49]. However these cameras have severe design difficulties, and have not often been implemented. Chapter 5 describes a feasibility study for an unobscured camera for the HROS instrument. Overall, much work has yet to be undertaken to investigate these various options for HROS. At the time of writing the precise form of HROS is yet to be decided, however Chapters 4 and 5 describe work to investigate likely optical arrangements. The next section of this Chapter describes the systematic design process undertaken for the HROS feasibility study.

3.3 Systematic Design of HROS With Reference to Existing Spectrograph Types

Several alternative design forms are possible for any echelle spectrograph, and in the Appendix F these different options are shown for HROS in a diagrammatic form. The author in collaboration with David Walker [72] introduced this diagram for the HROS study to aid conceptualization of the findings of the design process. This diagram shows that HROS has two fundamental alternatives for the instrument configuration, these being related to either Cassegrain or Nasmyth location. In the next section we shall look at the optical properties of the Nasmyth options for this instrument, before looking in later sections at the alternative advantages inherent in the Cassegrain solutions. Additionally the characteristics of example design forms, (both existing and planned) are described at relevant points during this analysis. We evaluate the design strategies of these other instruments, which have also been optimized for location on other large aperture telescopes (4-10m aperture mirrors). In particular we briefly review the following instruments: the Keck HRES instrument, the planned 8m aperture Very Large Telescope's (VLT) spectrograph, the UCLES on the AAT, and the planned SARG spectrograph for the Galileo National Telescope (TNG). We also review the likely performance characteristics of the planned instrument for the 10m aperture JNLT telescope called Subaru.

3.3.1 A Nasmyth Location For HROS: The Options

When comparing the various options possible for HROS, it is necessary to evaluate the designs on a number of separate dimensions. We shall focus here on those dimensions that relate to optical instrument performance. These are throughput, resolution and spectral purity, in addition to wavelength coverage and slit width. Additionally issues like single exposure wavelength coverage, inter-order spacing, and achieving adequate image stability are also important considerations.

We now consider the likely implications of locating HROS at a Nasmyth station, in comparison to at Cassegrain. Firstly it is important to realize that Nasmyth instruments have certain intrinsic optical disadvantages. Nasmyth instruments lose around 15 percent of light at the reflection from the Nasmyth flat mirror. Also the beam de-rotator required at Nasmyth loses another 4-30 percent of light depending on the design form employed. Additionally the large beams involved at Nasmyth make use of a grating cross disperser necessary.

However gratings have intrinsically lower transmission efficiencies. This results in a worst case situation at specific wavelengths, in which only 50 percent transmission is achieved compared to the 80 percent offered by prism cross dispersion. The smaller beam size of a Cassegrain instrument therefore enables the advantages of Prism cross dispersion to be obtained, in addition to the other throughout advantages discussed above. Thus providing intrinsically higher throughput if the two alternative designs are identical in other regards. Of course alternative designs are rarely equal in other regards, and the above discussion is overly simplistic. For instance we have only considered throughput, and have not take into account the many other dimensions of analysis neccessary. In the following sections we examine these issues in more detail.

3.3.1.1 An instrument Based on In-Plane Beam Separation

With reference to Appendix F, the Nasmyth option tree branch for HROS is shown on the left hand side of the diagram. Previously in this thesis (section 1.5) we demonstrated that a Nasmyth location requires a large beam size, and for HROS this would be around 300mm to satisfy the optical specification [72]. Two options are possible for such an instrument, with two different kinds of echelle gratings to provide primary dispersing power.

The first of these is termed the in-plane echelle, in which the incident and emerging beams from the echelle are separated in the plane of dispersion of the grating. This option can be used with either grating or prism cross dispersion to separate out the overlapping orders characteristic of such instruments. However grating cross dispersion is less efficient (in throughput terms) than the alternative prism case, and also provides uneven order separation.

The HIRES (High Resolution Echelle Spectrometer) instrument, located on the Keck telescope represents a successful example of this form [62][63]. The instrument provides a resolution of 67,000. In Figure 3.2 a schematic of the optical layout of HIRES is shown. The instrument has a 304mm collimated beam size, which is obtained by employing a 1 by 3 mosaic echelle configuration of total size 304mm by 1219 mm. These R2.8 echelle gratings have 52.6 lines per millimeter, and are mosaiced together by kinematically mounting the blanks onto a granite support slab. For this large beam instrument, even the cross dispersing grating is a mosaic composed of two gratings of total size 304mm by 609mm.

With reference to Figure 3.2 we now briefly describe the optical layout of this instrument. This diagram shows a simplified schematic of HIRES, where the primary dispersing direction is in the same plane as the collimator separating angle. Light from the slit falls onto the collimating mirror and is then directed onto the echelle mosaic, where it is separated from the incoming beam in the plane of dispersion before falling on the cross dispersing grating mosaic. Following reflection from the cross disperser the beam falls on the camera correcting elements before the component lines are imaged on the detector. This detector consists of a CCD that is a Tetronix 2048 by 2048 pixel device with 24 micron pixels. The detector spans a wavelength region from 0.3 to 1.1 microns.

The system efficiency of the HIRES plus Keck telescope is well documented [61]. The system achieves around 6 to 8 percent efficiency at peak near 6000 Angstroms. This figure includes a loss of 39 percent of light due to the presence of 3 aluminum mirrors which are necessary due to the Nasmyth location of the instrument. It is important to note that the HIRES instrument has a spectral format that is too large for the CCD to obtain in a single exposure. The Free Spectral Range (FSR) of the instrument at 10,000 Angstroms is around twice the width of an individual CCD, whilst at 3000 Angstroms the FSR is 3/4 of the physical detector size. This size discrepancy between detector and spectral format means that the desired spectral region to be observed is brought onto the CCD by rotation of the echelle and/or cross disperser. An example of this process is shown in Appendix I, with an example Echellogram from UCLES which shows the CCD outline to be moved over the desired region.

Overall the Keck instrument has been demonstrated to be a highly successful optical configuration, which employs some novel techniques to overcome the problems inherent in the sheer scale of the instrument. These are brought about by having a Nasmyth location on a very large aperture telescope. The main drawbacks to the design are that a significant percentage of light is lost within the instrument, due largely to the use of a grating cross disperser. Additionally the scale of the imaged spectrum in the cross dispersion direction makes simultaneous wavelength coverage quite small. One exposure can capture only around 2200 Angstroms centred on 5300 Angstroms.

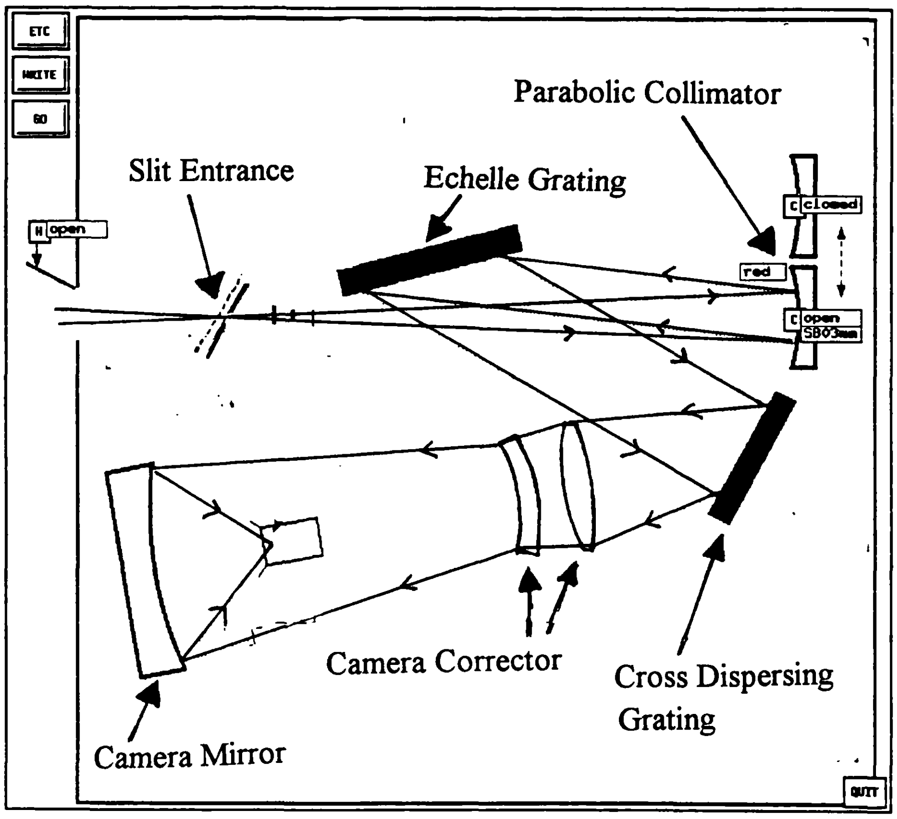


Figure 3.2 : Schematic of the Optical Layout of the Keck HIRES Instrument

Located on the back of a 10m aperture telescope, HIRES has advantageous signal to noise ratios when compared to other instruments on smaller telescopes. We also note that this instrument required a large echelle mosaic which presented serious manufacturing difficulties [63]. HIRES is achieving greater than 50 percent speed gains over the other spectrographs existing currently, however we shall see that the design is not optimal for the HROS case. It is possible to improve upon this signal to noise ratio in the HROS instrument by going to a more efficient design. HIRES also has the intrinsic disadvantage that its efficiency is significantly reduced in the ultra-violet region of the spectrum. The worst efficiencies (at the ends of the FSR) make for a transmission efficiency of only 45 percent. Additionally the 15 percent loss due to the presence of a Nasmyth flat, makes for a design that is potentially 60 percent less efficient than an alternative design that does not have these intrinsic problems. Since the UV is of critical interest, it is important to try to overcome these drawbacks intrinsic to this type of design form.

The alternative in-plane form of instrument employs prism cross dispersion, again for a Nasmyth instrument. However due to the beam size, monolithic prisms are impractical and so we reject them for HROS due to the manufacturing difficulties involved. The alternative is to use segmented cross dispersing prisms, (described in section 3.5) which offer a possible alternative design route. We have analyzed these options in detail for HROS [see page 15 in document 69]. However these kinds of optical arrangements, which have mosaic prism arrangements in single optical pass, do not fit into the space allowable for HROS. Additionally these layouts have poor efficiency due to vignetting light losses [see pages 15-16 in document 72]. Double pass Mosaic Prism arrangements overcome some of these problems. However these layouts were also rejected (see section 3.5.1). Locating the prism in front of the echelle grating is also rejected due to the large angles coming off the echelle that causes the mosaic to work poorly and give significant vignetting problems. As a result the requirement for a large wavelength range during a single exposure is not met.

The next alternative branch is shown in Appendix F, and involves locating the mosaic prism arrangement in front of a collimating mirror used in double pass. That is the beam from the slit passes first through the cross dispersing mosaic prism arrangement, and upon reflection off the collimator passes again through this arrangement. In this process the beam achieves sufficient cross dispersion. We have labeled this instrument option 1 in Appendix F.

3.3.1.2 An Instrument Based on a Quasi-Littrow Echelle

An echelle grating used in Littrow configuration has the incident and reflected beams separated by use of a slight gamma inclination angle orthogonal to the plane of the dispersion. For HROS we investigated the idea of basing such an instrument around two different types of echelle, one being an R4 and the other an R2 echelle. Again two possible versions of such an arrangement are possible for each echelle configuration. The first option shown in Appendix F uses an R4 echelle, which provides increased angular dispersion. Additionally this type of echelle allows for a moderate beam size for the instrument of around 150mm. This option, when combined with grating cross dispersion, is similar to the planned European Southern Observatories VLT instrument [14][15]. The VLT Nasmyth instrument will have a 200mm beam, and use twin mosaiced echelle gratings of total size 200mm by 800mm. The instrument has a planned resolving power of up to 200,000. This option would employ a Baranne white light pupil arrangement shown in Figure 4.1. A likely optical layout for the VLT instrument is shown in Figure 3.3 below.

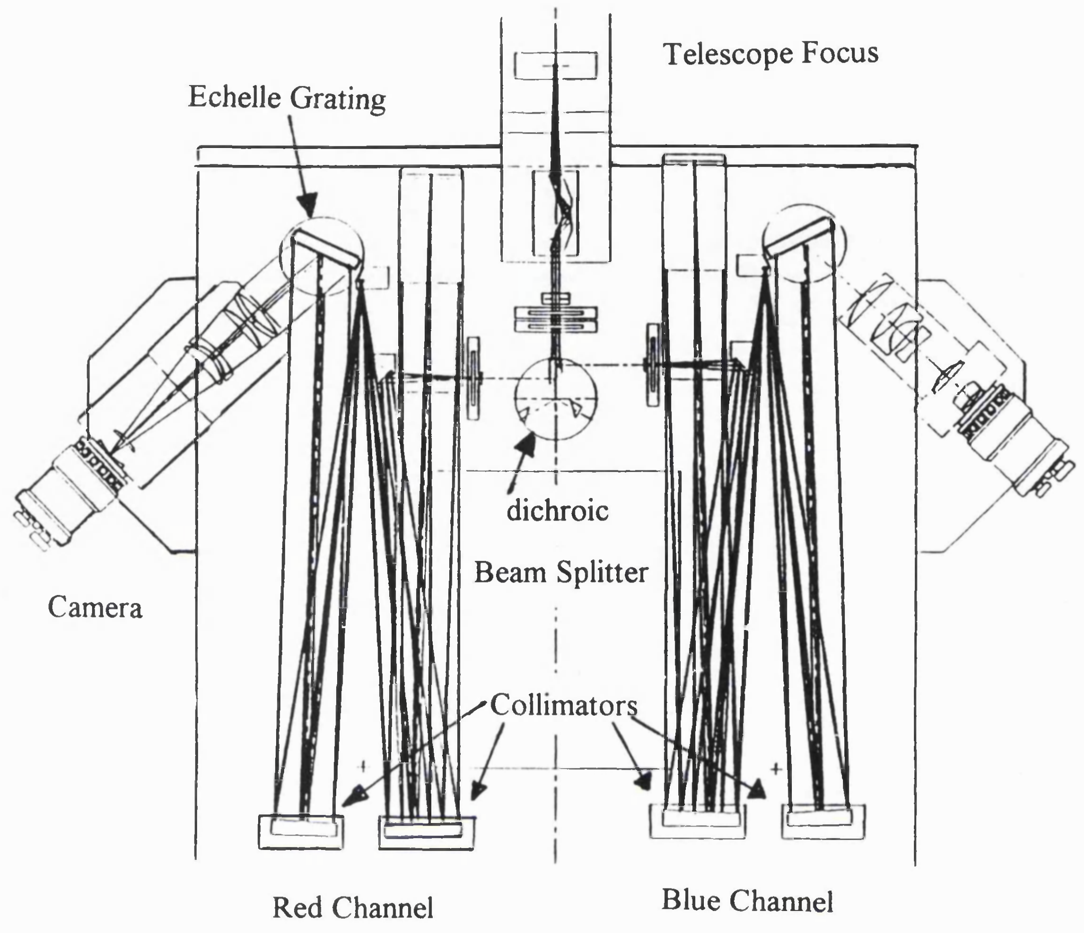


Figure 3.3 : ESO VLT Spectrograph Design With Separate Red and Blue Channels

Light entering the instrument through the slit is split into two separate channels, one for long and the other for short wavelengths. Analysis of the light proceeds on separate arms of the instrument. Once into one side of the instrument, light is collimated by an off axis paraboloid, and then reflected onto the grating arrangement. On leaving this grating, the beam once again falls onto the collimator which forms an intermediate white light pupil at a transfer mirror. This instrument requires twin cameras, one optimized for each of both red and blue channels. Such an arrangement for HROS requires that the instrument contain twin collimating mirrors in each channel, and this causes reduced efficiency due to the corresponding increase in reflection losses. Additionally the echelle must be 'scanned', or angled for small subsections of the proposed working wavelength region. This option is rejected for HROS due to the large number of reflections within the instrument, and the resulting reduced throughput efficiency. The prism cross dispersion version of this design is also rejected because it does not allow sufficient inter-order separation.

The other version of the Quasi-Littrow instrument option for HROS would use an R2/2.8 echelle. This version unfortunately for HROS would have a large beam size, with all the intrinsic disadvantages including larger camera aperture and hence larger aberration residuals. Two optical layouts were considered, the first with a high gamma angle. Instruments of this type look similar to the VLT separate blue-red channel type of instrument. The JNLT project plans to build a similar high resolution spectrograph termed the High Dispersion Spectrograph (HDS) with a resolution of 100,000. This instrument would have an entrance slit of around 60 arcseconds on the sky [30]. HDS will employ pupil transfer optics behind the echelle grating in a similar fashion to the VLT spectrograph described previously. This will be a Nasmyth instrument that aims to have a wavelength coverage from 3000 to 10000 Angstroms, using a very large format mosaic CCD detector in the focal plane of each camera. This detector is formed from 8 individual CCD chips of 2000 by 4000 pixels of 12 micron size, and has a 96 mm square format.

The other Quasi-Littrow R2/2.8 option we considered for HROS was a high gamma angle option, however in this case the large angle at the echelle causes slit images to tilt excessively at the extreme wavelengths. This drawback causes large aberrations at near ultra violet wavelengths. To overcome such difficulties the echelle could be scanned between exposures to different angles, in order to direct certain wavelengths into the centre of the CCD. However this results in dynamic range problems (low wavelength coverage) and we reject this option on these grounds. Alternatively a low gamma angle, reflecting collimator option can be envisaged, this time requiring a white light pupil system. Three different forms of this option are possible. The first of these requires a dioptric double pass collimator. The author examined such an option, and the results are described on page 23 of the design feasibility report for HROS [72], in addition in Chapter 4 of this thesis. This option was found to be untenable over such wide wavelength spans. This option is therefore envisaged to have a poor dynamic range and so is rejected on these grounds. The other two options for this white light pupil design branch were ray traced in detail by the author, and the conclusions are detailed on pages 24 and 25 of the HROS feasibility report [72]. We have labeled these options A and B. Aberration problems and space constraints made both these options unfeasible. This work is also described in detail in Chapter 4 of this thesis.

3.3.1.3. Conclusions: The Nasmyth Option

This section has presented a brief overview of the preliminary design process for a Nasmyth version of HROS. Additionally we have reported on the characteristics of three other Nasmyth spectrographs, two of which are now planned for completion later this decade. We have discussed the strengths and weaknesses of using such designs for a Nasmyth version of HROS. This work identified a preferred instrument configuration (Option 1) for HROS, with a 300 mm beam, using an in-plane R2 echelle with segmented prism cross dispersion. However this instrument would only fit into the planned Nasmyth space by using extra folding mirrors, which would have reduced the efficiency of the spectrograph considerably. Before ending our brief analysis of the different types of design possible for HROS, we shall briefly review the properties of another option not explicitly present on the option tree. This instrument is being planned for the TNG telescope, and is a high resolution spectrograph termed SARG [73]. This appears to be a hybrid of several of the designs we have considered in the above sections. It is a fibre-fed, white light pupil echelle spectrograph with prismatic cross disperser. The instrument aims to provide a resolution of between 25,000 and 200,000, with a 3700 Angstrom to 9000 Angstrom operating range, and a 200mm beam size. The design aims to use twin cameras, one for each operating mode, and uses a Quasi-Littrow configuration for the echelle grating. Interestingly the short focus camera is to be of the all transmitting lens type, whilst the long camera is of the Maksutov-Cassegrain form. The SARG design uses a novel form of fibre spectrograph interface providing high efficiency, but the design has a number of disadvantages for application to the HROS instrument. Importantly, fibre optics transmit poorly in the ultra-violet, whilst a key design goal of HROS is to probe this spectral region. Additionally the design, like the VLT spectrograph, has a large number of internal reflections and so has reduced throughput efficiency. Following analysis of the Nasmyth options for HROS, OSL proceeded to asses the feasibility of a Cassegrain location for HROS. The following sections present a brief overview of this work. This study was instrumental in the decision by the Gemini project management to reject the Nasmyth focus altogether from the southern Gemini telescope on which HROS will be mounted [70][72].

3.3.2 A Cassegrain Location for HROS: The Options

3.3.2.1 Option 1: Grating Cross Dispersion

The first option that can be envisaged for such an instrument is again the version which uses a grating to achieve cross dispersion. Once again this option is rejected due to the stringent requirements that HROS have the highest possible efficiency.

3.3.2.2 Option 2: Prism Cross Dispersion

This instrument option tree can be categorized into two separate types, the first being an 'UCLES' clone [11], which has proven to be an extremely efficient instrument located at the 3.9m aperture Anglo Australian Telescope. In Figure 3.4 we show a schematic of the UCLES layout. The diagram shows collimation of a 150mm instrument beam onto a set of three cross dispersing prisms. Once through these, the separated beams fall onto an echelle and are then reflected onto a 700mm focal length camera. UCLES employs exchangeable R2 gratings of 31.6 and 79.0 lines per millimeter for two different resolutions.

The primary resolution is 80,000 and the wavelength range covered is 3000 to 7000 Angstroms in up to four separate exposures. The minimum order separation is 3.5 arc seconds, and the entrance slit width is around 0.25 arc seconds. The instrument is capable of obtaining exposures as far as 11000 Angstroms into the near infra-red region. Twin detectors are employed, one for short wavelengths, which is a photo-electron counting system. Whilst alternatively for longer wavelengths a CCD can be used. UCLES has been demonstrated to be a highly efficient instrument, often achieving high combined telescope-instrument throughout values. The design is labeled as a possible form for HROS on the tree of different design configurations.

Alternatively another innovative design form can be envisaged for HROS, and uses an echelle grating that is immersed in Fused Silica, and as a result achieves theoretical efficiency advantages over its alternative. This form can once again be categorized into two types, the first being one in which the prisms are all located prior to dispersion. Alternatively it is possible to make the immersing block of silica perform a secondary function, aiding cross dispersion.

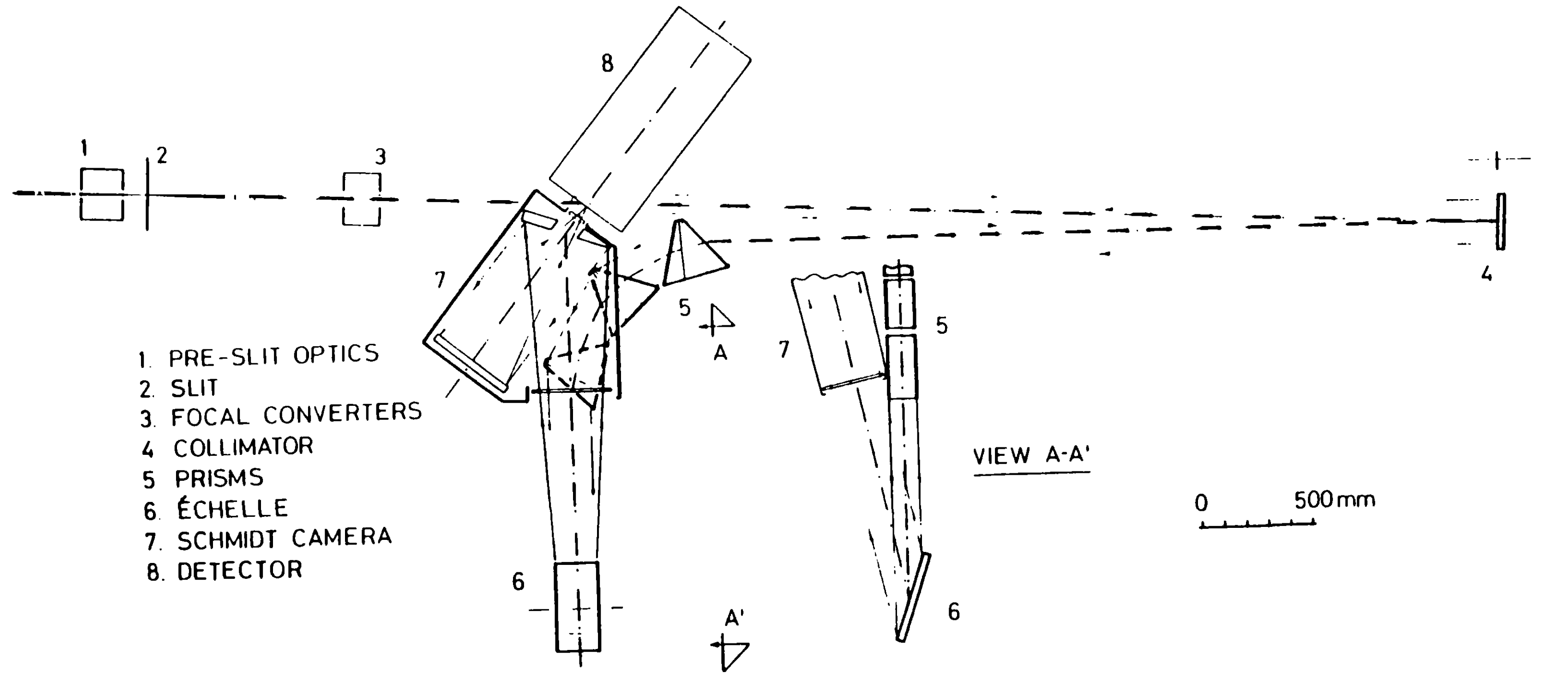


Figure 3.4 : A Schematic of the UCLES Optical Arrangement

This second option has a number of advantages, and is shown in Appendix H in schematic form. For HROS these advantages arise largely due to the the simple optical configuration the immersed echelle provides. Only two prisms are required to achieve adequate cross dispersion. Also no transfer mirrors are required within the spectrograph. This is an important point, because much of the work described here was achieved with the aim of fitting HROS into the small space allowable at Cassegrain. Additionally, the design provides enough space to fit interchangeable cameras for twin operating resolutions, one above the other by 'flipping' the echelle grating around. On Appendix F this is labeled as option 2. In the next section we assume that this is the option chosen for HROS, and much of the work on camera designs in later sections of this thesis assumes that HROS will be largely similar. Assuming we chose option 2 to be the best design form for HROS, we can now contrast this design with its likely competitors on the basis of their most important performance parameters.

3.3.3 Conclusion: Specific Advantages of the Option Chosen for HROS

In order to decide if HROS will be competitive when compared to other instruments with similar design goals, we now evaluate the instruments likely performance characteristics. Simple throughput calculations indicate that HROS will be the most efficient spectrograph on a large telescope. Some competing instruments, like Keck have a 60 percent (in arc seconds on the sky) wider entrance slit. This indicates an extra loss for HROS of around 30 or more percent of light from the stellar seeing disc. However HROS is to be efficiency optimized for transmission throughout the entire working wavelength span. The HROS instrument will therefore recover this loss, and has the potential to achieve greater throughput than its competitors. HROS gains 15 percent light loss over Nasmyth instruments, due to the absence of a Nasmyth flat mirror and another 5 - 30 percent (depending on design type) because it doesn't require a beam de-rotator. Additionally employing prism cross dispersion gains 30 percent over gratings on some exposures. In total these figures make the throughput of HROS highly competitive, even though the HIRES and HDS instruments have the advantage of increased (100/64) light grasp due to larger telescope apertures of around 10m.

Another salient issue relevant to this discussion, is the potential impact of adaptive optics on instrument performance. The Gemini telescope project plans the use of an adaptive optics system [16], using similar principles to the existing Keck system [75]. Assuming these adaptive optics programs are equally successful, and stellar images are reduced by the same factor in arc seconds on the sky, the HROS instrument has the potential to benefit more than its competitors. This is the inevitable result of the fact that the competitors will irretrievably lose more of this light as a result of the design form followed, i.e. within their spectrographs and at the Nasmyth flat. Whereas the simpler and more efficient HROS design, with less internal reflections and losses, will gain more light as a result of working with adaptive optics.

Additionally HROS will provide superior spectropolarimetry due to the fact that the Cassegrain focus does not cause the polarization of the light and avoids the time-varying polarization of Nasmyth instruments. Comparison of the proposed HROS design with similar existing or planned instruments, has demonstrated that this instrument potentially offers superior performance. For HROS we have considered all the possible design types, and evaluated all the possible combinations in a systematic fashion using a tree diagram. The use by the author of this visual aid, the option tree, makes the conceptualization of the different designs decisions easier to follow. This diagram has enabled the implications of multiple design decisions to be fully considered, and the alternative design paths were clearly identified using this method. A systematic design process has provided the optimum form for the HROS instrument.

3.4 A Mosaic Prism Feasibility Study

Location of HROS at the Cassegrain station of the Gemini telescope requires that a large mass of the order 2000 kg be supported at that position. Reducing this mass is considered important to ease the burden on the supporting structure. HROS will employ prism cross dispersion, and the large blocks of fused silica required will be quite massive. Mosaic arrangements of small prisms could in theory replace single cross dispersing prisms for large instruments like HROS. The novel concept of the mosaic prism arrangement was developed at OSL, and was first reported in the proceedings of the E.S.O. Workshop in High Resolution Spectroscopy [69]. Subsequent work completed by the author developed the concept specifically for the HROS instrument, and was reported in a paper presented at the S.P.I.E.'s Symposium of Telescopes for the Twenty First Century [70]. In this section the feasibility of mosaic prism tree arrangements is established for application to large beam spectrographs. The University College London Echelle Spectrograph (UCLES) employs two large fused silica cross dispersing prisms [11]. These prisms have dimensions that accept a 150 mm spectrograph beam. The manufacture of single monolithic prisms of even larger size would involve serious manufacturing difficulties. The author carried out ray tracing studies to test the feasibility of replacing very large single prisms with mosaic prism arrangements. For this study the author modelled mosaic arrangements composed of sets of 4 individual prisms. This number of constituent prisms was chosen for use with a spectroscope of around 300mm beam size. Increasing the segmentation of such a beam using more prisms leads to corresponding increases in vignetting losses.

3.4.1 The Double Pass Mosaic Prism Cross Disperser

An efficient way to achieve cross dispersion in an echelle spectrograph is to locate a single prism in close proximity to the grating, and allow both incident and reflected beams to undergo dispersion through the prism. In this way the beam in double pass achieves maximum dispersion using only a single prism, offering significant mass and cost savings.

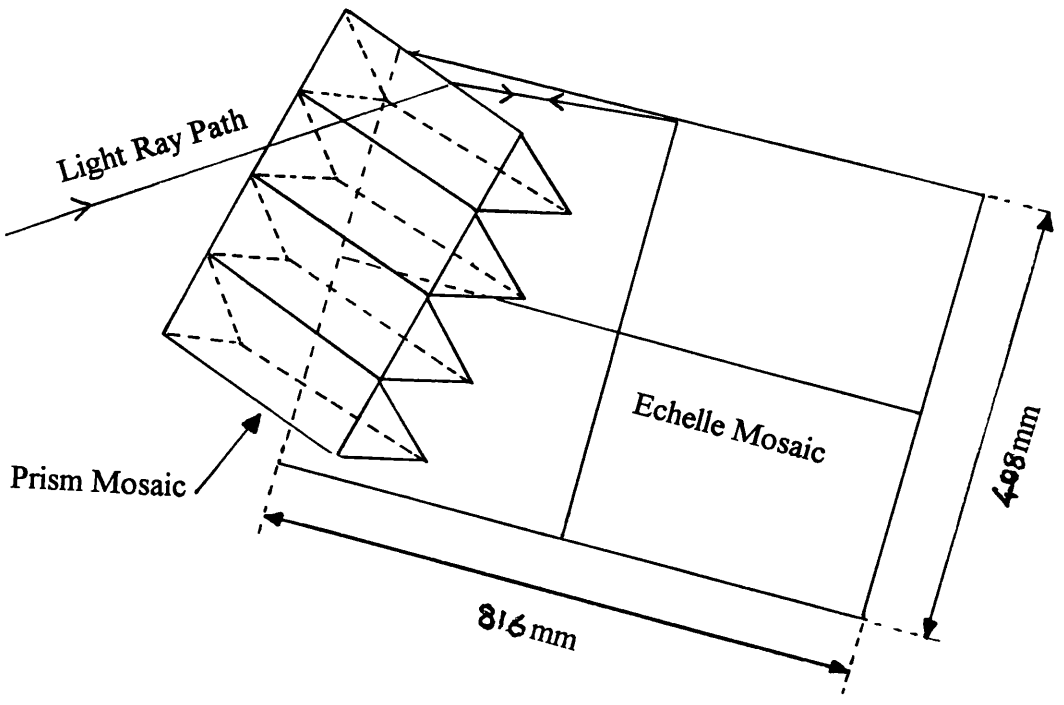


Figure 3.5: The Double Pass Mosaic Prism Arrangement

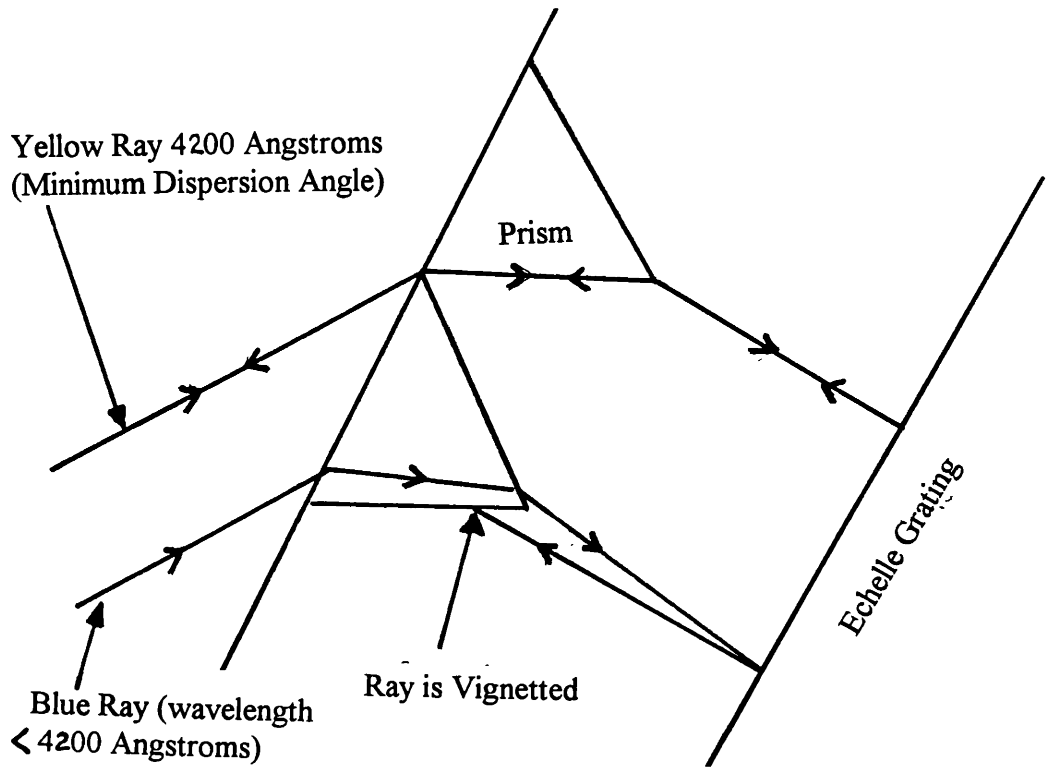


Figure 3.6: Blue Light Ray Vignetting in a Double Pass Mosaic Prism Arrangement

Replacing this prism with a mosaic of four as shown in figure 3.5, potentially offers even greater mass reduction. In order to investigate the optical properties of this mosaic tree cross dispersing prism the author developed a ray tracing program in C, and tested a variety of arrangements on an IBM P.C. Each prism is designed to work at minimum dispersion for yellow light at 4200 Angstroms. A significant drawback to this layout is that a proportion of light is lost from the pupil of an individual prism due to a vignetting effect. For an individual prism, a light ray shorter than the minimum dispersion wavelength undergoes greater refraction. If on entering, such a ray is close enough to the lower edge of an individual prism in the mosaic, it is vignetted and does not transmit through the arrangement. Vignetting can occur either on first or second pass through a single prism, and in both cases the ray impinges on the prism bottom face. An example of a ray that is not transmitted is shown in Figure 3.6. Depending on an individual ray's wavelength and distance from the lower edge on hitting the prism, it may or may not be transmitted through such an arrangement. As a result a proportion of rays are vignetted from the pupil that enters this prism echelle arrangement. All wavelengths shorter than the minimum dispersion wavelength suffer some degree of light loss. The large beam size of a Nasmyth spectrograph indicates mosaicing of echelle gratings, and in order to test the applicability of mosaic prisms to such an instrument we adopted a nominal arrangement.

We assumed a mosaic of four 64 degree blaze angle R2 echelle gratings, each with dimensions of 408 by 204 mm, as shown in figure 3.5. Ignoring the dead areas in between these individual gratings we then ran a number of ray tracing tests using mosaic arrangements of four prisms with a variety of apex angles. Each individual prism in the mosaic would accept a 102 mm beam. In Figure 3.7 we show the wavelength efficiency results for a 4 prism arrangement. The graph shows light loss as a function of wavelength and prism apex angle for short wavelengths down to 3000 Angstroms.

These results show that light losses of a double pass arrangement are over twenty percent at 3000 Angstroms for all prism apex angles in the 4 prism version. These losses are very high due to the large distance of the prisms from the echelle grating, causing small angles at the grating to be transposed into large displacements near the bottom of individual prisms. Increasing the number of prisms in this arrangement will result in proportional increases in light losses. Similar percentage losses result for wavelengths longer than the minimum dispersion angle, except that vignetting occurs for rays near the top of each prism.

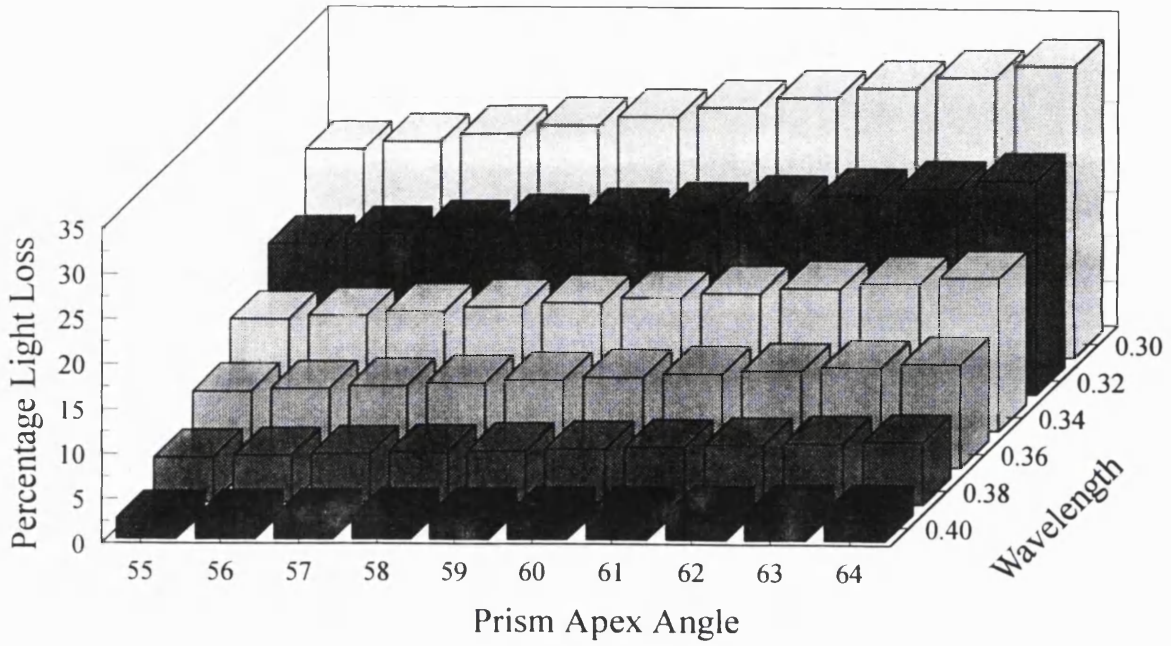


Figure 3.7: Double Pass Mosaic Prism Ultra-Violet Efficiency Graph

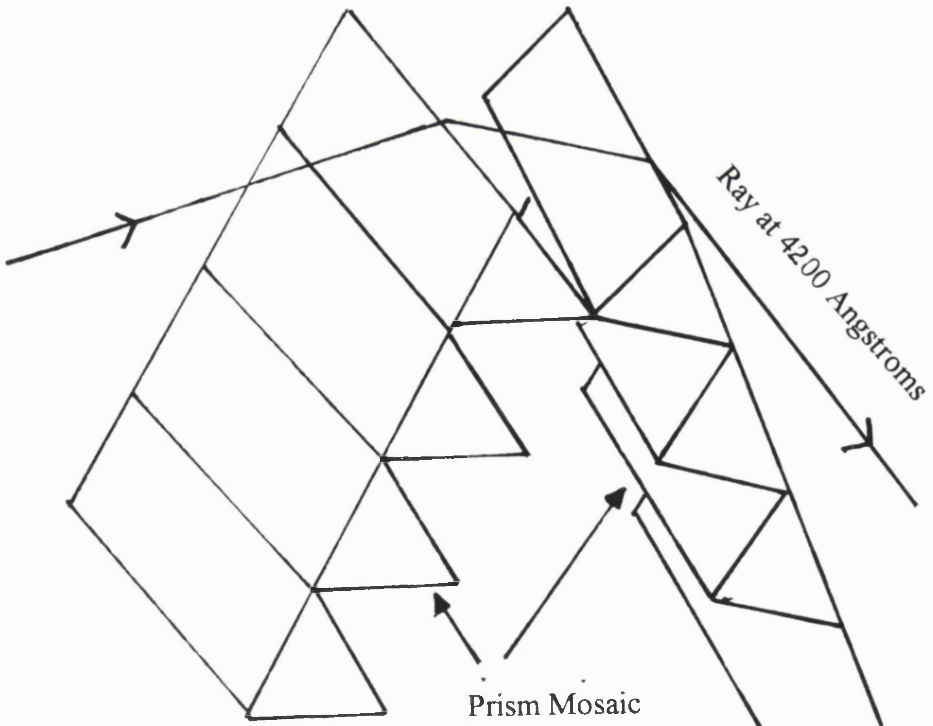


Figure 3.8 : The Single Pass Mosaic Prism Arrangement

3.4.2 The Single Pass Mosaic Prism Cross Disperser

Another method to achieve cross dispersion in an echelle spectrograph is to locate a double set of prisms in a tree of 8 prisms, used in single pass. For the purposes of this study we replace the single prism with a mosaic of 8 as shown in figure 3.8. A mosaic tree of 8 prisms was modelled using ray tracing code developed specifically for the task by the author. Each prism is designed to work at minimum dispersion for yellow light at 4200 Angstroms. A proportion of light is lost from the pupil of an individual prism due to a vignetting effect. Light losses are intrinsically lower in this arrangement, and at 3000 Angstroms the minimum prism losses are for a prism with an apex angle of 55 degrees and amount to just over four percent.

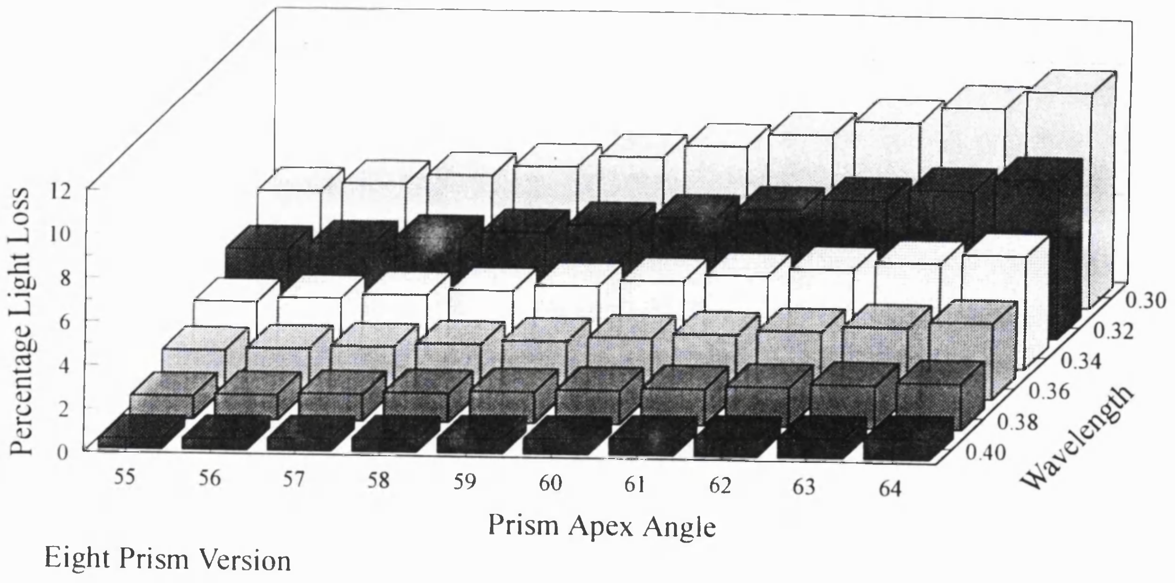


Figure 3.9 Single Pass Mosaic Prism Ultra- Violet Efficiency

Wavelengths shorter than the minimum dispersion wavelength again suffer light losses, and in Figure 3.9 we show the results for an 8 prism arrangement. The graph shows light loss as a function of wavelength and prism apex angle for short wavelengths down to 3000 Angstroms. Light losses were also investigated for wavelengths longer than the minimum dispersion angle and found to be of similar magnitude, at 55 degrees losses at 7000 Angstroms were 5.8 percent.

3.5 Conclusion: Mosaic Prism Feasibility Study

In this Chapter we have presented the results of a feasibility study into mosaic prism arrangements for large beam spectrographs. This work was done specifically for incorporation into a very large beam Nasmyth spectrograph. These types of instruments would require mosaicing of echelle gratings, and we adopted a nominal echelle configuration to match an instrument with a 408 mm beam size. Vignetting light losses for a double pass mosaic prism configuration were shown to be unacceptably high. However this work has demonstrated the feasibility of a single pass mosaic prism arrangement for use on very large beam instruments. Vignetting light losses for such an arrangement are less than 10 percent throughout a wavelength range of 3000 to 7000 Angstroms. A significant reduction in prism mass is offered by such arrangements, and they offer an alternate route to efficient cross dispersion for very large beam Nasmyth instruments. These techniques may also prove advantageous when designing Cassegrain instruments.

In order to successfully build high resolution spectrographs for 8m class telescopes, novel instrument configurations are required. In the next Chapter we investigate some new kinds of collimator arrangements for use within these new generation instruments.

Chapter 4

Collimator Design for High Resolution Spectrographs

4.1 Introduction

This Chapter describes the theoretical design, and subsequent modeling, of some new kinds of spectrograph collimator arrangements Collimator geometry has significant implications on the feasibility and space envelope of any proposed instrument. The design of a collimator system is therefore a fundamental starting point for the development of any new spectrograph. A number of different collimator arrangements were investigated for the HROS spectrograph, and the ray tracing results of these studies are presented here.

As discussed in Chapter 1 the sole purpose of a collimator is to align or collimate light from the spectrograph slit onto the echelle grating. It is important that in performing this task any aberrations introduced not degrade the final projected spectral line width at the detector. Off-axis or 'offset aperture' parabolic mirrors have often been successfully employed within spectrographs as collimators. Section three of this Chapter reports on the feasibility of different parabolic mirror arrangements, for integral use on the HROS instrument. First in the next section the possibility of employing dioptric lens collimators is considered for use within high resolution spectrographs like HROS.

4.2 Lens Collimator Design for Optical/UV Spectrographs

Employing a dioptric collimator can potentially reduce the space taken up by a Nasmyth spectrograph. Lens collimators may also be advantageous for Cassegrain instruments, again potentially reducing the space requirements for such an instrument. We now consider the feasibility of lens collimators specifically for a proposed Nasmyth version of HROS.

4.2.1 The Baranne Design for a Nasmyth HROS

A modified 'Baranne' white light pupil option for a Nasmyth instrument was suggested for HROS by OSL [72]. In Figure 4.1 we see an example of this type of system. Such a system has the advantage of forming an intermediate pupil, allowing the spectrum to be split up into separate channels for subsequent secondary collimation and imaging. For such a system to work a primary collimator would be required and is used in double pass. For HROS the wavelength range of such a hypothetical lens would be from 3000 to 10000 Angstroms. However it is well known that few materials exist that have high throughput in the near ultra violet down to 3000 Angstroms, the design goal for HROS. We selected a number of these materials for possible application to the design of such a lens. At the outset, it was recognized that the design of a lens with simultaneous chromatic correction over such a wide wavelength region was unlikely to be successful. However the results of such a study would serve to show over what kind of wavelength region such a system might be feasible. The materials used for this study were Fused Silica, Sodium Chloride, and Fluorite. Additionally we employed the new Schott materials Ultran 20 and 30 that also have high transmission in this region of the electromagnetic spectrum. A theoretical study of the feasibility of dioptric collimators employing these near ultra violet transmitting materials is detailed below.

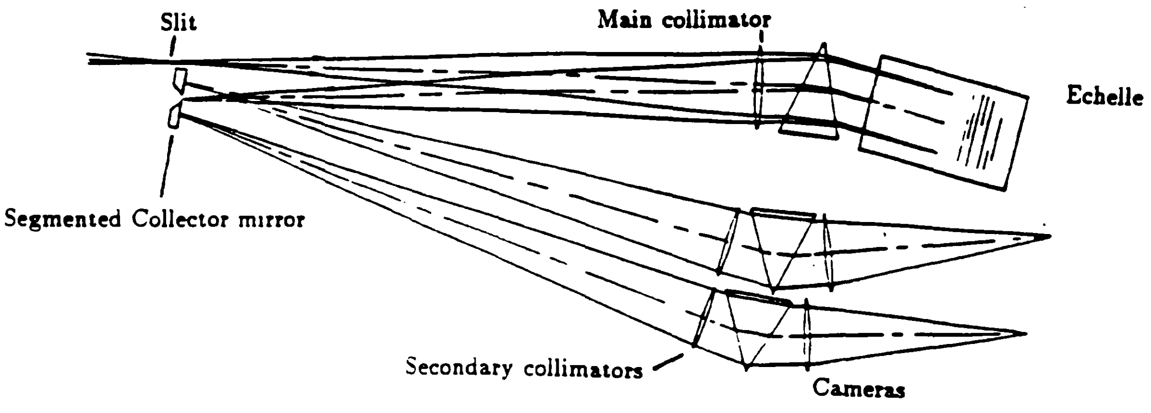


Figure 4.1 : The Modified Baranne Layout Showing Dioptric Collimators

The heart of the Baranne design is the double pass lens in front of the echelle, which is required to work throughout the entire wavelength coverage of the instrument. In order to investigate the feasibility of such a lens we theoretically evaluated a number of achromatic lens designs. It is evident from Figure 4.1 that the primary collimator will be required to work at a small off axis angle. This fact was not accounted for in the following feasibility study as the overriding design problem was considered to be correction of chromatic aberrations. (in addition to Spherical Aberration) Additionally the results of this study are relevant to the secondary collimators that are required to work over only a subset of the overall wavelength range.

4.2.2 Dispersion and Achromatic Lenses

In the late seventeenth century Isaac Newton first passed white light through a prism, and discovered light to be composed of different wavelengths that are dispersed upon refraction [44]. Snell's law explains this simply as being due to the fact that blue light has a higher refractive index and hence undergoes a greater degree of bending than red during refraction. The refractive index (N) of any material is defined as the ratio of the velocity of light in a vacuum with respect to the velocity within the medium. Early singlet lens designs suffered from blue colour blur as a result of dispersion and the corresponding intrinsic change of focus with wavelength. In the eighteenth century a number of different lens designers found it possible to minimize this effect, and unite blue and red wavelengths to a common focus by use of a positive crown and weaker negative flint glass combination. In lenses of this type the greater dispersion provided by the flint glass compensates for the dispersion introduced at the crown element. The primary design task considered here is the reduction of residual longitudinal chromatic aberrations present in achromatic doublet lenses. Achromatic doublet lenses are corrected for spherical aberration, and also chromatic aberration at two wavelengths 6563 Angstroms (termed c) and 4861 Angstroms (termed f). That is the lenses are designed to have the same focal length at these two wavelengths. This is achieved using the well-known technique referred to as Conrady's D-d method of achromatism [9].

The dispersive power of any material can be expressed as a number, often called the Abbe number (V) representing the level of dispersion with respect to a central (d) wavelength at 5893 Angstroms.

$V = {}^{N_d - 1}\big/_{N_f - N_c}$ ....**Equation 4.1**

Another intrinsic property of any glass is called the relative partial dispersion. This represents the dispersion of any material in other wavelength regions, with respect to the Abbe dispersion. We can represent the partial dispersion in any specific part of the spectrum by an equation of the following form:

$V = {}^{N_f - N_e}\big/_{N_f - N_c}$ ....**Equation 4.2**

If we plot a graph of Abbe (or V) number against the relative partial dispersion we obtain a relatively linear relationship for most materials. Such a graph is shown in Figure 4.2 for the materials we investigated, and the normal line about which most materials are found is marked.

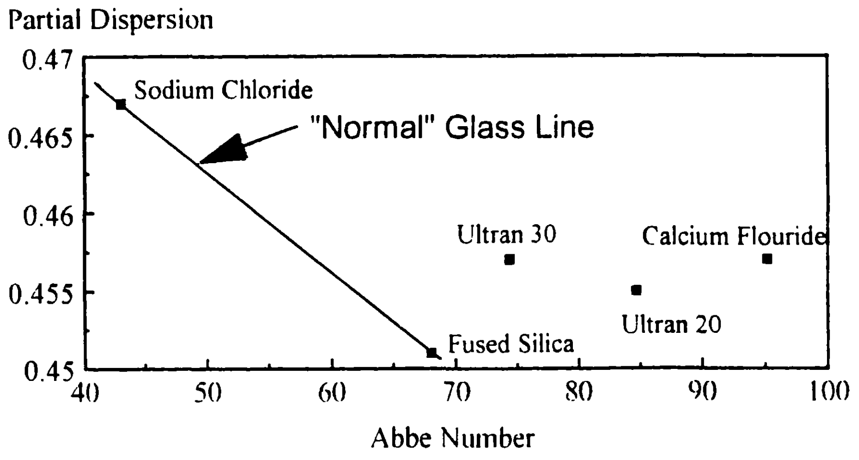


Figure 4.2 :Graph of Abbe Number Against Partial Dispersion

Before looking at lens design in detail, we have plotted the dispersion curves for the materials under study in Figure 4.3.

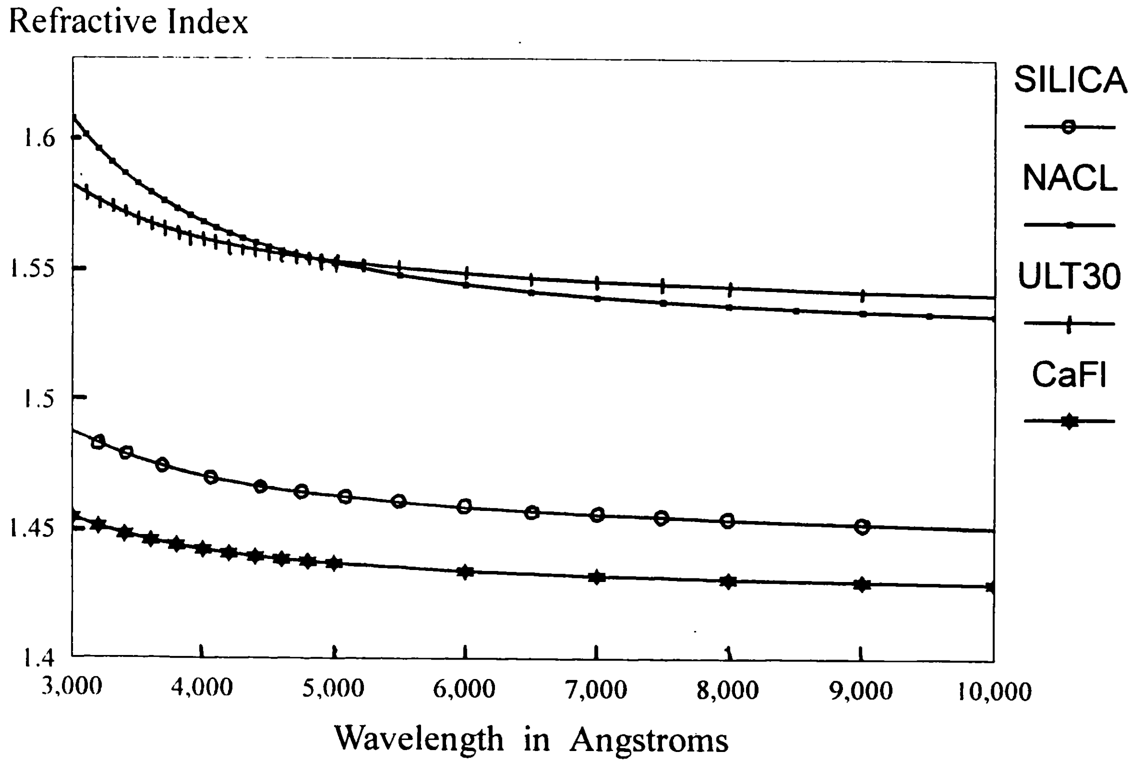


Figure 4.3 : Dispersion Curves for the Materials Under Study

From the dispersion curves we note a common feature of any material, that is at shorter wavelengths the refractive index tends to increase. This fact results in markedly higher dispersion in the blue part of the spectrum, and makes simultaneous correction of chromatic aberration in the far blue and red regions difficult. The situation is more complex than indicated in the graph. On a micro scale, significant non-linear variation of refractive index with wavelength exists. This has often been termed irrationality of dispersion and it is the root cause of residual chromatic aberrations (secondary spectrum) left over in any achromatic doublet design.

4.2.3. An Achromatic Doublet Composed of Fused Silica and Sodium Chloride

In order to test our designs for possible collimator use, and to compare one design type with another composed of differing materials, a nominal design goal was set. A lens of focal ratio 18 and aperture of 200 mm were the basic characteristics of this collimator design. The beam size for a Nasmyth HROS would be at least 300 mm, and so this lens design would require scaling for such an instrument. On an 8m telescope the corresponding plate scale for a nominal beam (f-number of 18) is 1.43 arc seconds per millimeter. Our preliminary design goal was to produce a lens based collimator with aberrations less than 0.2 seconds of arc as projected on the sky (0.28 mm in the telescope focal plane) with the lens illuminated in parallel light in single pass. This imaging performance would be required over the widest wavelength range possible. Achromatic doublets comprise a crown and higher dispersing flint glass. From Figure 4.2 it is evident that Fused Silica and Sodium Chloride fit this somewhat crude criteria. A contact cemented doublet was designed using the characteristics of these materials with 'crown' element of Fused Silica and 'flint' element of Sodium Chloride. However ray tracing showed this hypothetical achromat did not exhibit the required imaging performance due to large residual secondary spectrum in the visible range from 4600 to 6500 Angstroms. This secondary spectrum term refers to the fact that a lens corrected to have the same focal length at two separate wavelengths may still fail to unite other wavelengths at this focus. Usually secondary spectrum is defined as being the distance of the blue or red focus from the combined c-f focus (for visible work at least). Difference in the refracting power of a material at different wavelengths (partial dispersion) is the underlying cause behind this and similar residual longitudinal chromatic aberrations.

Lenses composed of Fused Silica and Sodium Chloride were found to have a significantly large secondary spectrum within the visible range. This pair of materials do not make a good combination and are said to be not well 'matched' for lens design purposes. In the next section we discuss the requirements that two materials must meet to reduce this secondary spectrum effect to an absolute minimum.

4.2.4. Fluorite as a Basis for Achromats

The chromatic properties of Fluorite are well suited for producing achromats. Firstly we note that this material has a very low refractive index. With respect to a lens element made of another optical material of identical power Fluorite will also have intrinsically lower dispersion. (Assuming these elements are immersed in similar optical media) In order to produce an achromat with reduced secondary spectrum, materials are needed with different V numbers but partial dispersions that are similar. In Figure 4.2 we can see that Fluorite departs significantly from the usual linear relationship between V number and relative partial dispersion as found in most other glasses and materials. Fluorite therefore allows this condition to be satisfied in many achromatic doublets if the combined materials are matched carefully together. We should also note that Ultran 20 and 30 also depart significantly from the normal linear relationship. These materials can therefore also facilitate the design of a well corrected achromat.

In the next section a number of lens combinations based around Fluorite are evaluated. This theoretical work was achieved using iterative methods described by Kingslake [36]. The author adopted these methods and translated them into a spreadsheet for use on Lotus 123. An achromatic doublet design composed of Fluorite and Fused Silica was theoretically tested for use as a primary collimator. This work involved examining the blur spot diagrams produced by ray tracing an optimized design. Rings of light rays focused on infinity were traced through the lens, and the aberrations evaluated at the best focus. (This focus was determined by finding which BFL gave the circle of least confusion) The combination showed excellent imaging (small transverse aberrations) from 4500 to 8000 Angstroms. However below 4500 Angstroms the doublet showed increasing levels of chromatic difference of focus, and these shorter wavelength rays fell outside the 0.14 mm circle of good imaging performance. Note that 0.14mm was chosen as the required tolerance for acceptable aberration sizes for single pass through the lens, half the specification discussed in section 4.2.3. This larger tolerance accounts for the effects of double pass of light through the primary collimator. At 3000 Angstroms the spot size was 2 mm in size. The difference of focus throughout the wavelength range is shown in Figure 4.4, where the lens in question is lens A2.

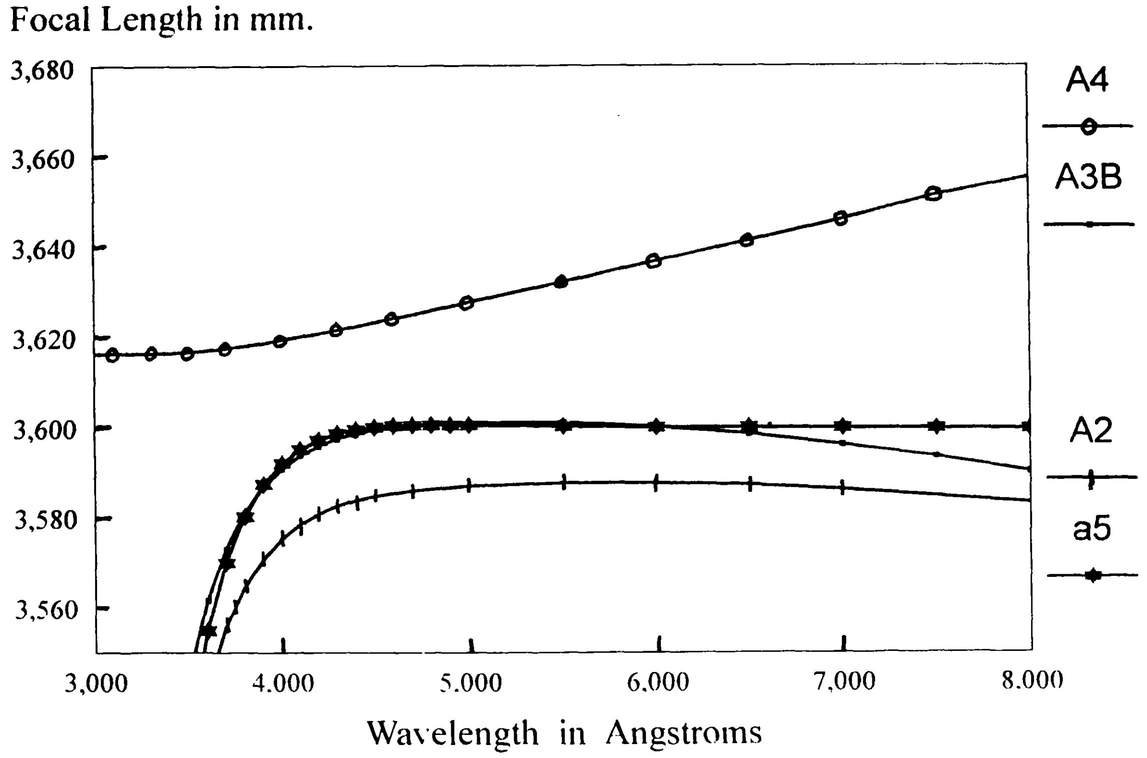


Figure 4.4 : Graph of Focal Length Against Wavelength for 4 Lens Designs

	Radius of Curvature	Separation	Glass
Surface 1	1221.901		Air
Surface 2	-717.865	35	Fluorite
Surface 3	10639.543	35	Fused Silica
Image Plane		3554.0	Air

Table 4.1 : The Parameters of an Achromatic Lens Collimator (Lens A2) Composed of Fluorite and Fused Silica

Another look at Figure 4.2 shows that Fluorite and Ultran 30 have almost identical partial dispersion values. Satisfying this condition allows a very well corrected achromat to be constructed from these materials. We modelled such an achromat (termed A5), and whilst it offered reduced chromatic aberration down to 4100 Angstroms, this doublet was found to exhibit difference of focus at near ultra-violet wavelengths. This can be seen in Figure 4.4, showing difference of focal length with colour. It is important to note that this lens is very well corrected in the conventional sense, for any lens working well below 4400 Angstroms is considered to be approaching apochromatic correction. These designs are sometimes called semi-apochromats. This type of design has materials with dispersion properties that are well matched, and such a lens could be employed as a secondary collimator within the Baranne arrangement. These materials may also be suitable for lenses that are used in other areas of astronomical instrumentation. For use as the primary collimator however, any hypothetical lens would require a greater degree of chromatic correction.

	Radius of Curvature	Separation	Glass
Surface 1	641.863		Air
Surface 2	-714.959	22.575	Fluorite
Surface 3	2244.384	22.575	Ultran 30
Image Plane		2554.5	Air

Table 4.2 : The Parameters of an Achromatic Lens Collimator (Lens A5) Composed of Fluorite and Ultran 30

The achromatic designs adopted so far have all suffered from residual chromatic aberrations over certain regions of the propose working spectrum. The reason for this can be termed the 'irrationality of dispersion' of the various optical materials employed. This term is given to mean that dispersion properties of the optical materials do not conform to simple mathematical modeling , and that the relative partial dispersions of the materials are not the same. In a narrow region of the electromagnetic spectrum dispersion might approximate to a simple quadratic function. Yet over wider wavelength spans the dispersion cannot be easily modelled, and the reduction of chromatic aberrations becomes significantly harder. In order to design a lens with reduced chromatic aberration over a large spectral region, apochromatic designs are required. Apochromatic lenses are specifically designed to unite three wavelengths to a common focus, and in the next section we evaluate their applicability for use as a spectrograph collimator.

4.2.5. Apochromatic Collimator Designs

In order to reduce the effects of residual secondary spectrum, we investigated the properties of apochromatic lenses that comprise three elements with specifically chosen optical properties. The theory of apochromatic triplet lenses is well known so we shall not attempt a full mathematical treatment here. Suffice it to say that an apochromatic triplet can be designed from three materials that obey a relationship shown in Figure 4.2.

A well-corrected triplet with small residual chromatic aberration can be designed if the three chosen optical materials form a widely spaced triangle on such a diagram, as shown in Figure 4.2. It can be seen that three of our four materials meet this condition - Fluorite, Fused Silica and Ultran 30.

	Radius of Curvature	Separation	Glass
Surface 1	1464.129		Air
Surface 2	-753.580	21.56	Fluorite
Surface 3	1030.928	21.56	Fused Silica
Surface 4	7462.687	21.56	Ultran 30
Image plane		3552.48	Air

Table 4.3: The Parameters of an Apochromatic Lens Collimator (Lens A3B) Composed of Fluorite, Fused Silica, and Ultran 30

An apochromatic design tested by the author composed of these materials showed considerably reduced chromatic aberration in the region from 4100 Angstroms to 7000 Angstroms, but continued to display large residuals below the 4100 Angstrom point.

We confirmed that 100 percent of encircled energy fell within 0.2 arc seconds as projected on the sky for all wavelengths within this range. However the image degrades substantially below 4100 Angstroms. The reason for this can be seen in Figure 4.4, which shows the change in focus below 4100 Angstroms for this lens termed A3B.

4.2.6. A Prototype Design for an Ultra-Violet Sub-collimator

Lenses are not usually designed specifically for the near ultra-violet spectral region, and this is due partly to the lack of materials that have high transmittance at these wavelengths. Also, standard dispersion formulae fail to accurately predict the dispersion of a material below about 3650 Angstroms. However the author proceeded to study theoretical lenses specifically for this region using standard refractive index tables. The effects of such inaccuracies were not accounted for in the following work. In reality it would be necessary to use a very large number of data points (refractive index against wavelength) measured from real 'melts' of the materials under study. Only in this way would it be possible to accurately predict the imaging properties of real lenses used in the near ultra violet. However the following work serves to illustrate the feasibility of designing these lenses.

An apochromatic design was theoretically designed to work from 3000 Angstroms up to the most red part of the spectrum possible. This work was again achieved using Kingslake's methods, but altered by the author to take into account the somewhat shorter wavelength range being considered. New partial dispersion values were calculated for the region of interest in the near ultra violet. These re-defined partial dispersion values were as follows :

$$V_{UV} = {N_{3500} - 1}\Big/{N_{3150} - N_{3600}}$$ ....**Equation 4.3**

$$P_{UV} = {N_{3150} - N_{3300}}\Big/{N_{3150} - N_{3600}}$$ ....**Equation 4.4**

The next step is to select the materials to be used for the Apochromat, and we chose Fused silica, Sodium Chloride and Fluorite. It is important to note that Sodium Chloride is hygroscopic, and use of such a lens would require special precautions as a result. However the resulting design showed theoretical imaging that met the previously defined specification in the wavelength range 3000 - 4200 Angstroms. Such a lens might therefore suffice for use as secondary collimator, that is only required to work over a subsection of the proposed instrument working wavelength range.

4.2.7 Conclusion: Lens Collimators for a Nasmyth HROS

If a Baranne concept was feasible for a Nasmyth HROS, then the collimator is required to work over a range of at least 3000 to 7000 Angstroms. The work in the previous section has shown that this goal is not feasible with the materials currently available. A Baranne concept for HROS, if feasible, must therefore be of the reflecting type. In the next section we describe work undertaken by the author to investigate the feasibility of a reflecting Baranne design for a Nasmyth HROS.

	Radius of Curvature	Separation	Glass
Surface 1	2877.698		Air
Surface 2	-1115.540	21.00	Fused Silica
Surface 3	-3286.395	21.00	Sodium Chloride
Surface 4	-2714.006	21.00	Fluorite
Image plane		3597.64	Air

Table 4.4: The Parameters of an Apochromatic Lens Collimator (Lens A4) Composed of Fused Silica, Sodium Chloride and Fluorite

4.3 A Reflecting Baranne Design for a Nasmyth HROS

In the reflecting version, the first lens collimator is replaced by a parabolic collimator in double pass. The author modelled a simplification of the configuration using an off-axis parabolic mirror and a flat mirror that could be tilted to the correct angle in place of the echelle and adjacent cross disperser prism [72]. With the echelle near Littrow giving negligible anamorphism, this is a valid approximation. The Littrow configuration refers to an echelle in which the incident and reflected beams are not separated in the plane of dispersion of the grating. Three possible reflecting Baranne arrangements were investigated, and are detailed below.

4.3.1. Layout A: No Cross Dispersion Prior to the Intermediate Echellogram

This arrangement requires the slit to be oriented in the tangential plane, with respect to the off-axis parabola. Note that this slit orientation is not always the most desirable for minimum aberrations. In this version a beam of f-number 19 diverges from the Nasmyth focus, which is incident upon an off-axis parabola. The mirror collimates the beam onto an echelle grating (a plane mirror in our model). We show the arrangement in figure 4.5.

In order for the collimated beam to correctly align with a grating placed immediately below the slit, an angle of 2 degrees is required between the optical axis of the telescope and the axis of the parabola. After dispersion from the grating rays are again reflected off the parabola and imaged above the slit length. In such an arrangement no cross dispersion has been introduced prior to the formation of an intermediate Echellogram above the slit.

Aberrations are introduced when rays strike a parabolic mirror at significant angles with respect to the optic axis. It follows that wavelengths diffracted at large angles from the echelle suffer from larger aberrations. In order to correctly model the echelle dispersion using a plane mirror, the mirror was angled specifically in the tangential and sagittal planes for each individual wavelength to be investigated.

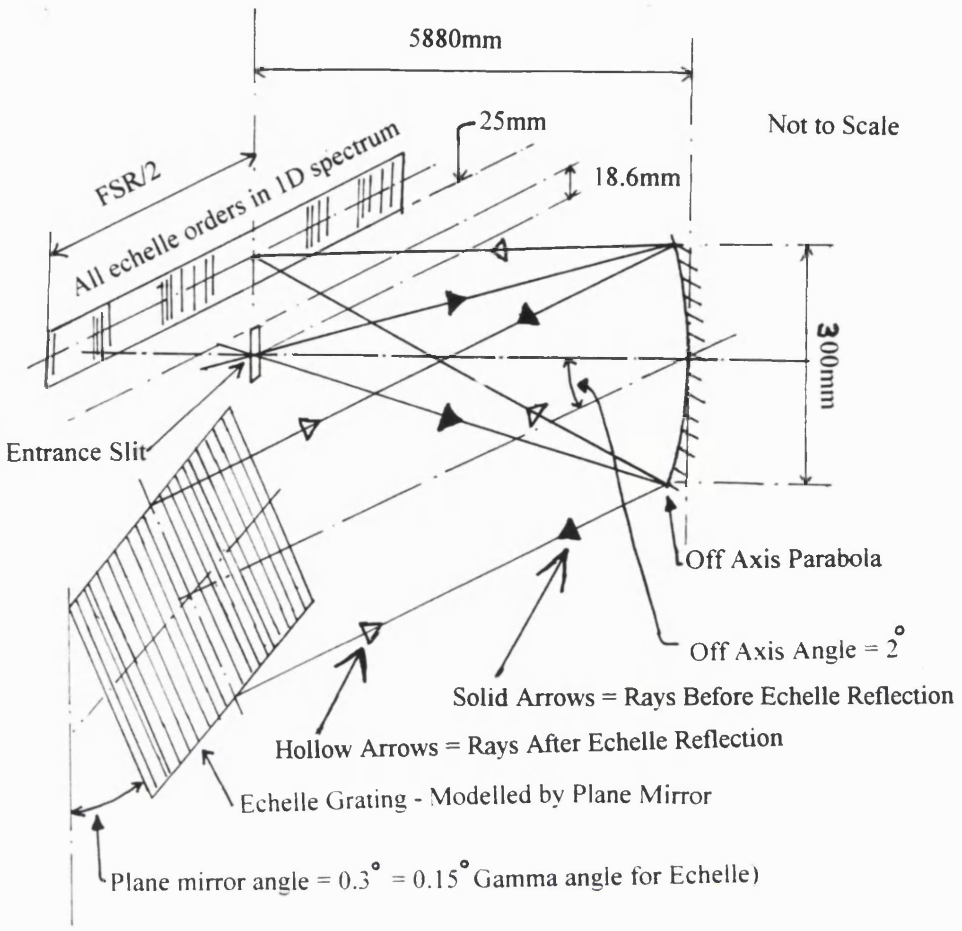

5880mm
25mm
Not to Scale
18.6mm
FSR/2
All echelle orders in 1D spectrum
300mm
Entrance Slit
Off Axis Parabola
Off Axis Angle = 2°
Solid Arrows = Rays Before Echelle Reflection
Hollow Arrows = Rays After Echelle Reflection
Echelle Grating - Modelled by Plane Mirror
Plane mirror angle = 0.3° = 0.15° Gamma angle for Echelle)

Figure 4.5 : The Reflecting Baranne: Schematic of the Ray Tracing Model

The author modelled two variants of this arrangement in order to investigate the effects of these aberrations on slit images at different wavelengths. These two layouts were identical apart from using the dispersion characteristics of 79g/mm and 31g/mm R2 echelle gratings respectively, each with a beam size of 300mm. These results do not include aberration effects due to the length of the slit, but this was taken to be +/- 30 arc seconds to allow room for beam clearance from the intermediate Echellogram. The intermediate Echellogram was set at 25mm above the slit.

Analysis of spot diagrams using the 79g/mm echelle model revealed that the aberrations are greater than 0.5mm at worst, equivalent to 0.65 arc seconds on the sky. This work assumes that complete orders up to 7000 Angstroms are accommodated, and this gives a field of +/- 300mm either side of the slit. With an R2 echelle of 31g/mm the field size is reduced to +/- 120mm and the spot sizes are smaller and found to be 0.3mm or less, representing 0.4 arc seconds on the sky. The smaller levels of aberrations of this latter option are due to the intrinsically shorter FSR of this type of echelle. Both of these options were rejected due to the unacceptable effects of geometric aberrations on spectrograph performance. The author also investigated toroidal collimating surfaces, but concluded they offer little advantage in this application where good imaging performance is required over a large number of field positions.

4.3.2 Layout B: Cross Dispersion Prior to the Intermediate Echellogram

The author also modelled several other variants of this type of collimator arrangement [see document 72 on page 24]. These consisted of optical layouts in which cross dispersion is achieved in front of the echelle, producing a 2 dimensional intermediate Echellogram above the slit. These options require even larger incidence angles on the parabolic mirror at second pass. As a result the aberrations worsen considerably.

4.3.3 Layout C: A Twin Off-axis Parabola Arrangement

By addition of a second off axis parabolic mirror to the above arrangement we can investigate another option [72]. Two parabolic mirrors are located face to face, with one collimating light onto an echelle grating and another identical mirror re-collimating it again after reflection from the first mirror. However this system requires a large physical area, and is unfeasible for HROS due to space limitations.

4.4 Conclusion

The collimator feasibility studies presented here were undertaken specifically to investigate various HROS optical arrangements. The design of a single lens collimator to work from 3000 to 11000 Angstroms is unfeasible. This is the result of the intrinsic change in focus present in current design types. A transmitting Baranne design for HROS is as a result not achievable. The reflecting version of the layout may prove feasible for a large beam Nasmyth spectrograph. However, HROS was found to have very limited space constraints that militated against such a solution. Initial work indicates that an off-axis angle of 4 degrees is required for the off-axis parabola for a Cassegrain HROS. A model of the system has been ray traced and collimator aberrations are acceptable for such an instrument [72].

The next Chapter goes on to look at spectrograph camera design, specifically in order to investigate the feasibility of the HROS instrument at a resolution of 120,000.

Chapter 5 : Camera Feasibility Study for the High Resolution Optical/UV Spectrograph

5.1 Introduction

The optical feasibility of any particular spectrograph arrangement is critically dependent on the viability of the camera configuration. In this Chapter several different types of spectrograph camera design are investigated for applicability to the HROS instrument. Among these designs are short Schmidt, Maksutov and Houghton systems, and a new sub-aperture camera design. Prior to a detailed discussion of specific designs, we now review the general properties of any spectrograph camera.

5.2 The General Properties of a Spectrograph Camera

The overall spectrograph specification will dictate the required values of camera focal length and numerical ratio. Aside from these requirements a particular camera must also exhibit certain other desirable properties. A key consideration is that the camera configuration allow clearance for incident and refracted beams from the echelle grating. Two basic layouts are possible, one being a Cassegrain configuration with the requirement for an external focus. Alternately the focal plane can be located inside a cryogenic camera.

A spectrograph camera must also provide high transmittance over the entire working wavelength range of the spectrograph. This is in order to ensure the maximum signal to noise ratio within an individual spectral line. The use of optical components like Fused Silica, with high transmittance in the near ultra-violet region, is becoming common as a result. Throughput considerations also dictate that vignetting losses should be kept to an absolute minimum.

Importantly a spectrograph camera should have low levels of residual geometric aberrations, so that the resulting image spread falls within the minimum projected slit width in the instrument focal plane. We should note that this constraint is only stringently required in the plane of dispersion of the echelle. However very poor image quality in the orthogonal plane could degrade signal to noise due to increased image spreading, and resulting binning of image across many detector pixels. As a result astigmatism in the orthogonal dispersion plane can be tolerated to some extent but should not be excessive.

A variety of different camera designs may be conjured up that meet some or all of the above criteria. Yet each will have its own merits, and for any particular application an optimum design must be chosen. We will now briefly overview some of the different camera types, and contrast their differences.

5.3 Prototype Camera Designs For HROS

5.3.1 The Classical Schmidt Camera

In 1910, Gustov Kellner patented a new optical system that consisted of a spherical mirror and lens located at the mirror's centre of curvature [32]. Kellner marketed his invention as a new type of projecting lamp that produced a collimated beam of light over great distances. This system was later popularized by Bernhard Schmidt in his paper of 1932, in which he describes its application as a high speed photographic camera [53]. The Schmidt camera has since proved indispensable for use in astro-photography, offering excellent image sharpness over a wide field of view. Figure 5.1 shows the system with a circular aperture stop placed at the centre of curvature of a spherical mirror.

Arranging the stop in this manner ensures that all incident parallel ray bundles will exhibit symmetry in the cone of light after reflection. This in effect leaves every off-axis light beam that passes through the aperture stop with its own axis of symmetry about the principal ray. Therefore coma, astigmatism and distortion and other off-axis aberrations are not introduced. Spherical aberration remains, and this is removed by use of an aspheric correcting lens placed at the centre of curvature. This plate also introduces a small amount of first order difference of colour to reduce the effects of spherochromatism that set a limit on the useful focal ratio of the basic Schmidt system. This latter function is achieved by giving the aspheric corrector a small base radius of curvature.

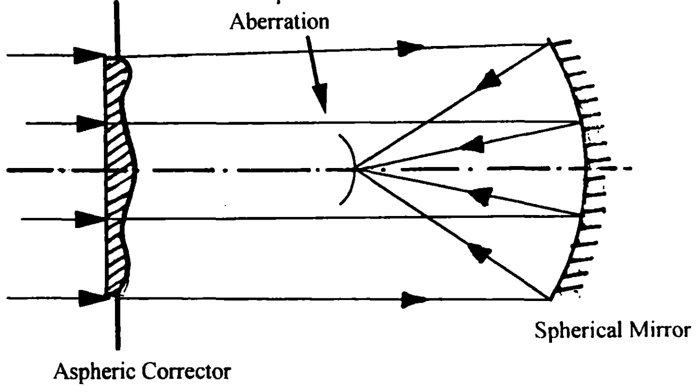


Figure 5.1 : The Classical Schmidt Camera

A major drawback of the classical Schmidt system is that the focal image lies on a curved surface. This aberration has often been reduced by bending photographic film into a shape that followed this curve. However this solution is inadequate as resulting photographs are distorted, and in any case modern semiconductor devices usually require a flat image surface. Schmidt suggested placing a plano-convex field flattener immediately prior to the image surface to correct this defect. This technique introduces coma as well as increasing chromatic aberrations. However by designing the Schmidt corrector and field flattening lens as a single system, it is possible to reduce these effects to a minimum.

A number of variations on Schmidt's original design are possible, and several types have seen use as the imaging camera within astronomical spectrographs.

5.3.2 Application of the Schmidt System for Use as a Spectrograph Camera

In a spectrograph camera the effective aperture stop in one plane is the diffraction grating. Monochromatic ray cones diverge from the grating and are the field rays for the camera. In a high resolution echelle spectrograph, the cross dispersing element causes the separate monochromatic pupils to be displaced relative to each other on the grating surface. This complicates modelling of the camera entrance pupil. The design of a spectrograph camera should take these effects into account, and in this Chapter we show the monochromatic pupils incorrectly located at identical positions on the grating for simplicity. In reality camera imaging performance must be evaluated fully on the spectrograph ray tracing model. The design of a Schmidt system for use as a high resolution spectrograph camera therefore presents a different problem from the usual case. A shorter Schmidt camera is required.

A number of people have reported on possible designs that might suffice for this purpose (see [1] and [76]), in which the aspheric plate is moved towards the mirror in order to balance the coma introduced by the field flattening lens. A notable form described as a shorter Schmidt system was reported by Wynne in 1977 [76]. Wynne wrote a seminal paper on the application of the Schmidt camera to astronomical spectrographs. In this paper Wynne draws on several design techniques in creating his unique approach to Schmidt camera design.

Modern optical spectrographs operate over very wide wavelength regions. The reduction of secondary spectrum effects within a camera design becomes crucial therefore, unless the designer is willing to tilt the camera focal plane. In his systems Wynne used a self achromatic field flattening lens to reduce this residual colour aberration. The principles behind these lenses were first described by D.Gabor in 1941 [21], and later applied by Maksutov [43] to the design of the Bowers/Maksutov Camera.

To correct Petzval field curvature a lens of a specific optical power is placed near the focal plane. If this lens is of the self achromatic type it will introduce no focal shift for different wavelengths. It is a fact that many shapes of lens exist that will suffice for this purpose. Lens shape refers to the fact that a variety of different values of front and back curvatures are possible that produce a lens of any specific power. It is possible therefore to find a shape for this lens that allows existing coma and astigmatism to be reduced by moving the corrector towards the spherical mirror. The extra spherical aberration introduced by this lens is easily removed by a slight adjustment of the specified corrector plate aspheric coefficients. The resulting camera has reduced axial colour aberrations as well as tiny residual monochromatic aberrations. Note the size and effect of these remaining aberrations will of course be dependent on the focal ratio or speed of the camera.

5.3.3 The HROS Camera Specification at a Resolution of 120,000

The primary resolution for HROS is set at 120,000, dictating a focal length of 1310mm for the camera to achieve this resolution. These calculations assume an R2 echelle with 2 theta = 12 degrees between incident and refracted beams. For an echelle grating, R refers to the tangent of the blaze angle of the grating, so an R2 grating has a blaze angle of 63.4 degrees. These calculations are compatible with 2.5 pixels per projected slit width. The echelle presents an anomorphic beam onto the camera corrector set some 1288mm from the echelle. The monochromatic beam dimensions are 220mm in the plane of the echelle dispersion and 160mm in the orthogonal one. The camera therefore may be said to have a speed of about f-number 6.0 in the echelle dispersion plane. A twin CCD format is one possible detector arrangement for HROS, where the CCDs form a 'T' layout. One nominal 2K by 4K CCD has dimensions of 60mm by 30mm with 15 micron pixels. The resulting semi-field for a 1310mm camera is 1.3 degrees in the echelle dispersion plane, and 1.96 degrees in the other.

5.3.4 A Short Schmidt Camera for HROS

The author designed and tested a family of Wynne type short Schmidt cameras in order to evaluate the potential of this design form for HROS. A new interactive approach to camera design was applied to this feasibility study in order to speed up the design process. Camera optimization was achieved using the Inter-Opt software developed by the author, which enabled the parameter-aberration space of this design form to be rapidly probed. We then evaluated the best of a number of Schmidt designs, which is shown in the Figure 5.2.

In the design shown in Figure 5.2 the beam or pupil obstruction by the field flattening lens of the on axis field position is 20 percent. This obstruction percentage has been estimated by simple graphical means, and we included a nominal 1m diameter main telescope secondary mirror in these calculations. Figure 5.3 shows the entrance pupil mapped onto the correcting lens for the on axis wavelength. Figure 5.4 shows the entrance pupil at the extreme wavelength position mapped onto the correcting lens. It can readily be estimated that light losses are lower here, and we have concluded the loss to be 8 percent for those field positions in the corners of the CCD mosaic. Elsewhere in the spectrum losses will be between these values of 8 and 20 percent.

Examination of blur spot diagrams at the polychromatic focus in which the CCD would be fixed, shows that 100 percent of the encircled energy falls within a 15 micron circle over all field angles and wavelengths. From this work we conclude that a Schmidt design has been shown to be feasible for HROS at a resolution of 120,000.

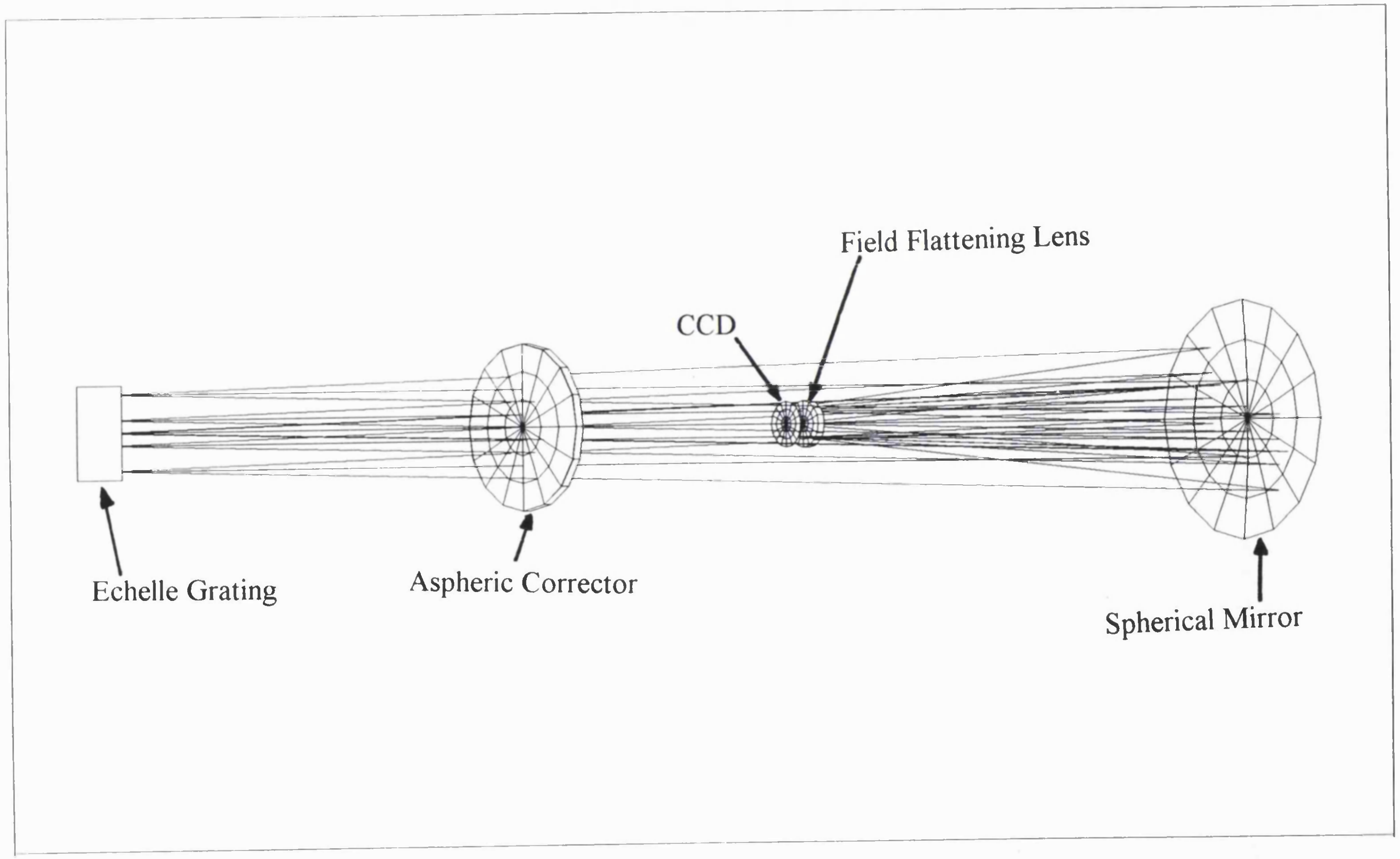


Figure 5.2 : Schematic of a Schmidt Camera of 1310mm Focal Length

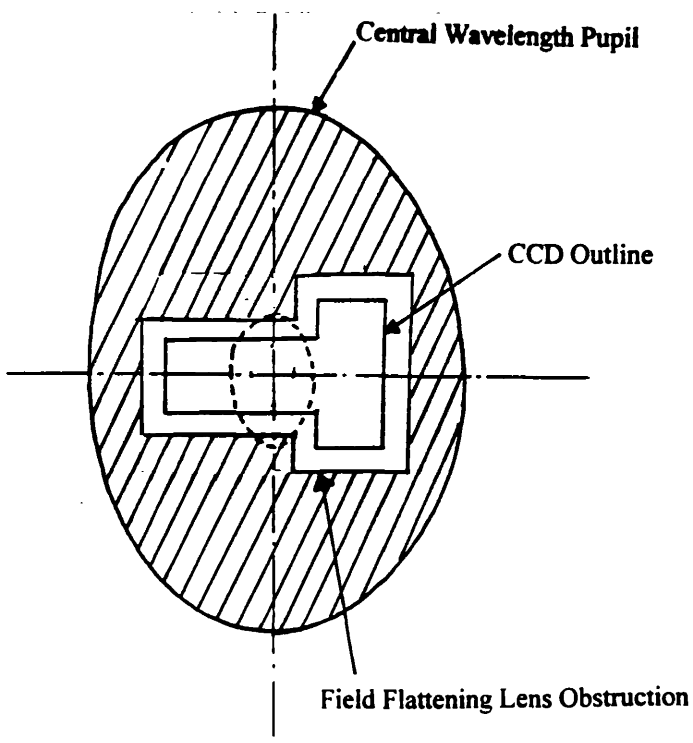


Figure 5.3 : The On-Axis Pupil Mapped onto the Field Flattening Lens to Show Beam Obstruction Losses

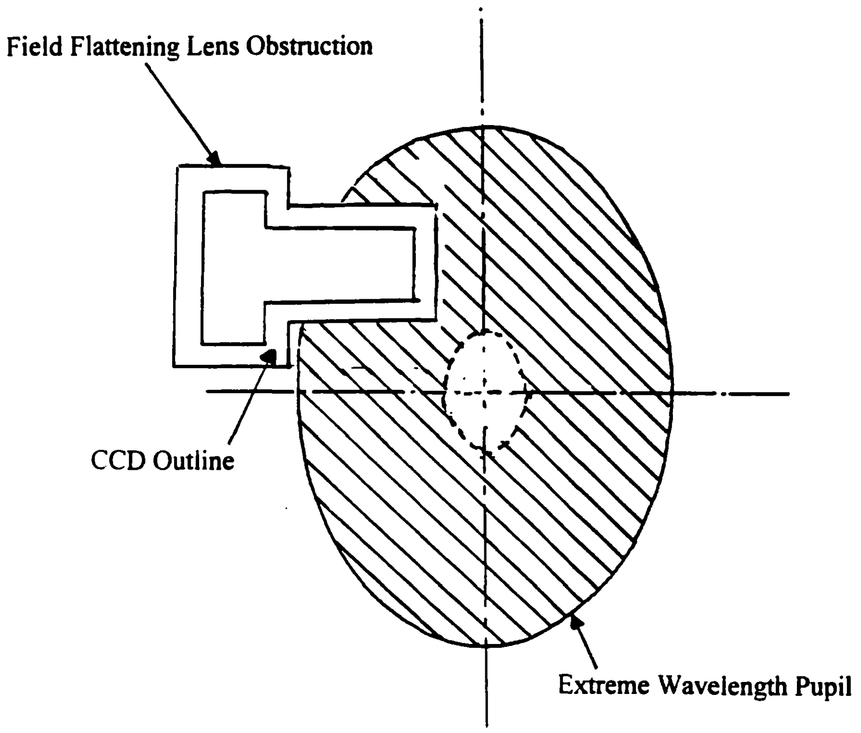


Figure 5.4 : The Extreme Field Pupil Mapped onto the Field Flattening Lens to Show Beam Obstruction Losses

During the design process a compromise position for the field flattening lens was chosen. Moving the lens closer to the focal plane allows for a thinner lens and hence smaller lens diameter, with corresponding lower levels of obstruction losses. However in doing this, to correct coma we found it necessary to lengthen the camera by several hundred millimeters. Two basic families of Schmidt designs are seen to exist, ones that are shorter and exhibit higher obstruction losses, as well as longer ones with lower intrinsic losses.

	Radius of Curvature	Separation	Media
Surface 1 = Echelle as Stop	Infinite		Air
Surface 2	Infinite	1288.880	Air
Surface 3 A4 = A6 =	-1960784.31 +2.443e-11 +3.591e-17	21.000	Fused Silica
Surface 4	-2811.674	2200	Air
Surface 5	-467.845	-1332.000	Air
Surface 6	-9833.809	-22.085	Fused Silica
Image Plane	Infinite	-55.307	Air

Table 5.1 : The Parameters of a Short Schmidt Camera of 1310mm Focal Length

5.3.5 A Modified Maksutov Camera for HROS

Gabor first described in principle how a meniscus lens could correct the aberrations of a spherical mirror. But it was D.Maksutov who really applied himself to solving the problems in detail. In his seminal paper of 1944 he described a new type of catadioptric system [43]. The camera was also described by Bouwers [7] but it has become widely known as the Maksutov camera. In his paper Maksutov details a method of avoiding the use of aspheric surfaces in camera systems. This is advantageous as aspherics are exceedingly difficult to manufacture. The method employs a near concentric full aperture meniscus lens that corrects the spherical aberration of a mirror without introducing axial chromatic aberration (the meniscus is a self achromatic lens).

Because the meniscus corrector is almost centred on the aperture stop, only small amounts of non symmetrical aberrations are introduced and so coma and astigmatism are greatly reduced. In this paper Maksutov makes the useful observation that it is possible to reduce these by changing the distance between the meniscus and mirror. One basic advantage of the Maksutov camera is that in the form where the meniscus is concave towards the mirror, the system is shorter than the alternative Schmidt system. This allows clearance for incident and refracted beams from the echelle grating.

It has also been noted that the Maksutov, intrinsically, has lower levels of spherochromatism than the Schmidt camera. However due to the steep radius of curvature of the meniscus element, large amounts of higher order spherical aberration (5th, etc.) is present in cameras of high aperture. Thickening the meniscus can reduce this but the effect remains in very fast systems. However very large cameras of the type considered here would end up impractical as a result, due to the large blocks of silica required.

The shell thickness in a Maksutov camera is usually chosen to correct primary longitudinal colour. However we chose an alternative method to correct this basic chromatic difference of focus, one that allows the radius of curvature to differ slightly to correct this aberration. Usually the corrector is located so that both radii of curvature of the shell are centred on the aperture stop to eliminate coma. For the HROS camera the stop is located some 1288 mm from the echelle and so the fundamental principle of the Maksutov camera is violated. We were able to step around this problem by introducing a self achromatic field flattening lens to the system, and optimizing all the system parameters interactively using Inter-Opt.

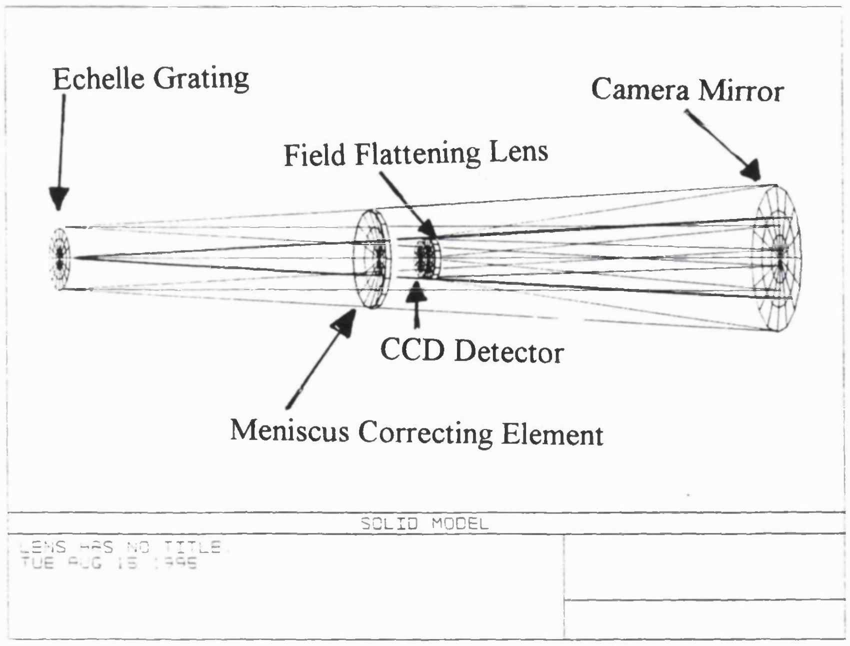


Figure 5.5 : Schematic of a Maksutov Camera Modified for Use as a Spectrograph Camera

A large number of designs were tested, and we finally converged on one solution whose parameters are shown in table 5.2. Examination of blur spot diagrams for this camera over the required field showed its performance is well suited to 15 micron pixels, although the design has slightly worse imagery than the alternative Schmidt form previously considered in this Chapter. The obstruction area of the field flattening lens leads to light losses lower than in the Schmidt form. In the Maksutov design the investigated pupil losses were slightly lower than in the Schmidt form.

Notably the modified Maksutov design we looked at for HROS was significantly shorter than the previous Schmidt form. The Maksutov was 1376 mm long as opposed to 2000mm for the Schmidt, and this fact could prove advantageous during mechanical design of HROS. We conclude that a Maksutov design is feasible for HROS at 120,000 resolution. However in light of the superior imaging performance of the Schmidt, a Maksutov is only preferred when short camera length is of paramount consideration.

	Radius of Curvature	Separation	Media
Surface 1 = Echelle as Stop	Infinite	0	Air
Surface 2	-436.491	1288.800	Air
Surface 3	-447.760	21.000	Fused Silica
Surface 4	-2811.674	1580.000	Air
Surface 5	-425.313	-1376.000	Air
Surface 6	-18900.019	-11.651	Fused Silica
Image Plane	Infinite	-40.501	Air

Table 5.2 : The Parameters of a Maksutov Camera of 1310mm Focal Length

5.3.6 A Modified Houghton Camera for HROS

A large number of other optical configurations could in theory be applied to spectrograph camera design. Most notable among these are designs in which the corrector plate is replaced by a separated doublet. This camera was first investigated by J. Houghton, and has the advantage of being entirely composed of spherical optics [29]. In Houghton's original type, a zero power twin lens combination is located in front of a spherical mirror. Houghton's design incorporated lenses composed of glasses with similar dispersion properties (often the same material). Other people worked along similar principles, using third order theory to design fairly slow systems (around an f-number of 10). The author modified the original Houghton design to test its applicability for HROS as a camera at resolution 120,000.

The classical form of Houghton camera has a curved focal plane, and the aperture stop is usually the first lens. To apply these techniques to spectrograph camera design a number of changes to the original design concept are required. We set the echelle grating as the stop in one plane, and add a self achromatic field flattening lens close to the focal plane. The author adopted a unique new method to achromatise the doublet corrector. In these systems it is important that the sums of the coefficients of longitudinal colour at each of the four surfaces of the doublet are equal to zero. The method of achieving this varies. One common technique is to sum the coefficients of the first two surfaces, and equally distribute the opposite sign of this coefficient sum between the other two surfaces. This is achieved by giving each of the back two surfaces a specific radius of curvature. However this method only gives the designer two radii of curvature to vary, surface one and two, whilst the other two automatically follow to produce a combined colour coefficient sum of zero. To give an extra degree of freedom during the optimization process the author cancelled the colour coefficient from the first three surfaces entirely by the fourth surface. In this way three surfaces could be varied and only one surface had a fixed or 'solved' radius of curvature. (See Appendix E for the mathematical formulae.)

We finally converged on a design shown in Figure 5.6, after ray tracing over 100,000 separate camera systems using Inter-Opt. The design has a number of key advantages over the previous camera forms. Notably it offers the same excellent imagery of the Schmidt form. In Figure 5.7 we show blur spot diagrams for this design. The blur spots diagrams at 5500 Angstroms are shown at on axis and extreme field positions. From figure 5.6 we can see that this camera is also significantly shorter than the Schmidt form.The addition of 2 extra surfaces for the Houghton design indicates around 4 percent extra transmission losses over the spectrum. However these designs allow the field flattener to be located closer to the focal plane, and as a result a smaller diameter lens is required. This lens presents a smaller obstruction to the polychromatic pupils. In this design the resulting field flattener losses are an average of 3 percent lower than in the Schmidt case.

This result means that the obstruction light loss almost offsets the disadvantage due to the extra transmission loss. Therefore the Schmidt and Houghton designs exhibit similar throughput properties. However if anti-reflection coatings were applied to the corrector surfaces of both cameras, the balance would be tipped in favour of the Houghton Design. In this case the extra two surfaces allow for greater relative transmission savings over the alternative Schmidt form.

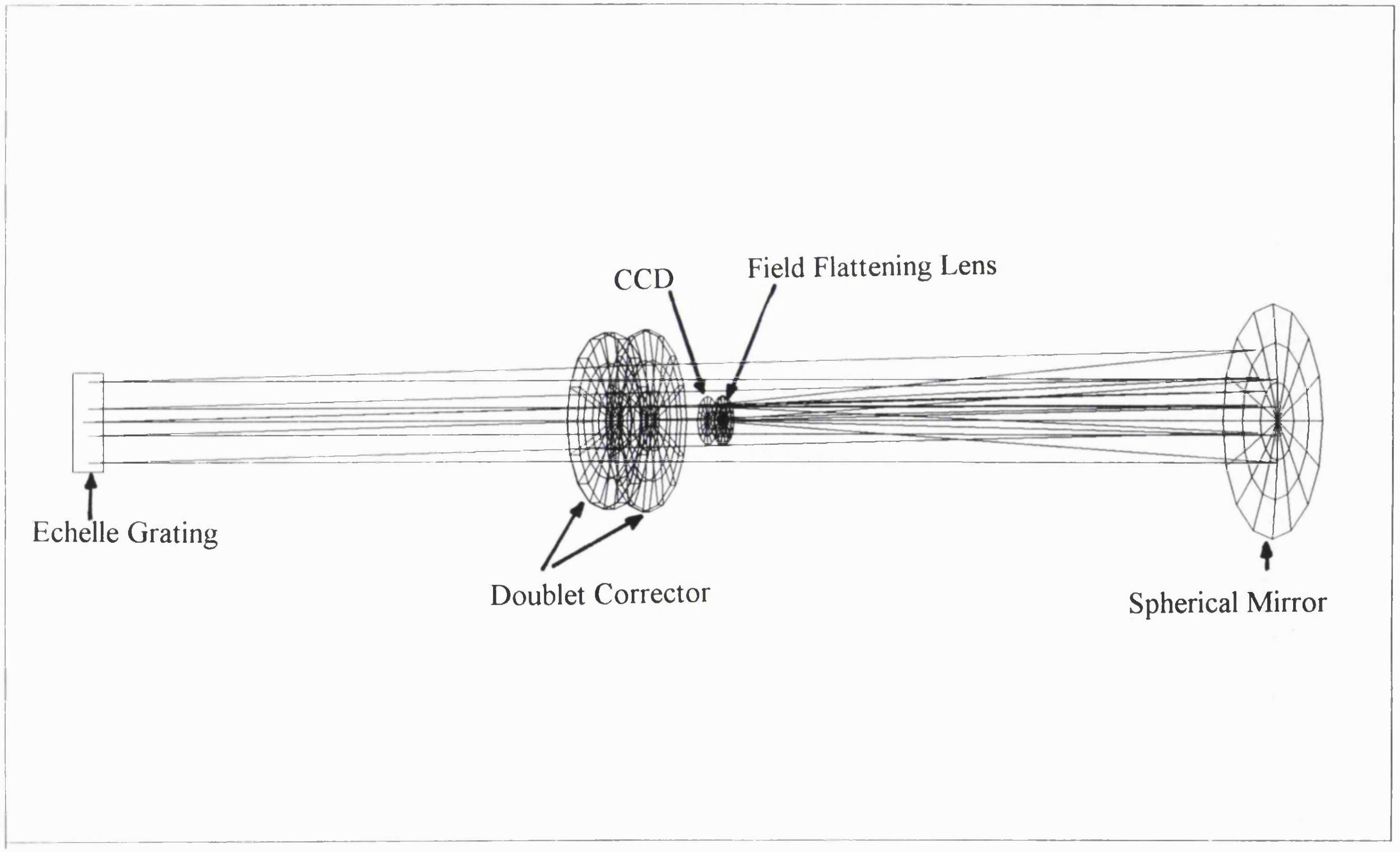


Figure 5.6 : Schematic of a Houghton Camera of 1310mm Focal Length

	Radius of Curvature	Separation	Media
Surface 1 = Echelle as Stop	Infinite	0	Air
Surface 2	-1032.716	1288.800	Air
Surface 3	-1552.891	20.000	Fused Silica
Surface 4	-2403.384	68.790	Air
Surface 5	-1391.266	20.000	Fused Silica
Surface 6	-2804.97	1543.492	Air
Surface 7	-415.365	-1366.139	Air
Surface 8	-7655.795	-9.537	Fused Silica
Image Plane	Infinite	-36.159	Air

Table 5.3 : The Parameters of a Houghton Camera of 1310mm Focal Length

5.3.7 The Feasibility of an Unobscured Camera for HROS

The preceding camera designs were all based around systems that exhibit symmetry about the optical axis. However it is possible to design a camera in which we employ a spherical mirror but only use it in sub-aperture mode, in order to prevent vignetting due to the presence of detector or Cassegrain mirror. Light losses within such a camera will be substantially reduced. The basic layout is shown in Figure 5.8. These systems result from a parent on-axis design that has been cut in half. In fact the parent camera must be 'cut' somewhat above the half way position as shown in Figure 5.8, to ensure that the pupils are not obscured by the CCD and field flattening lens.

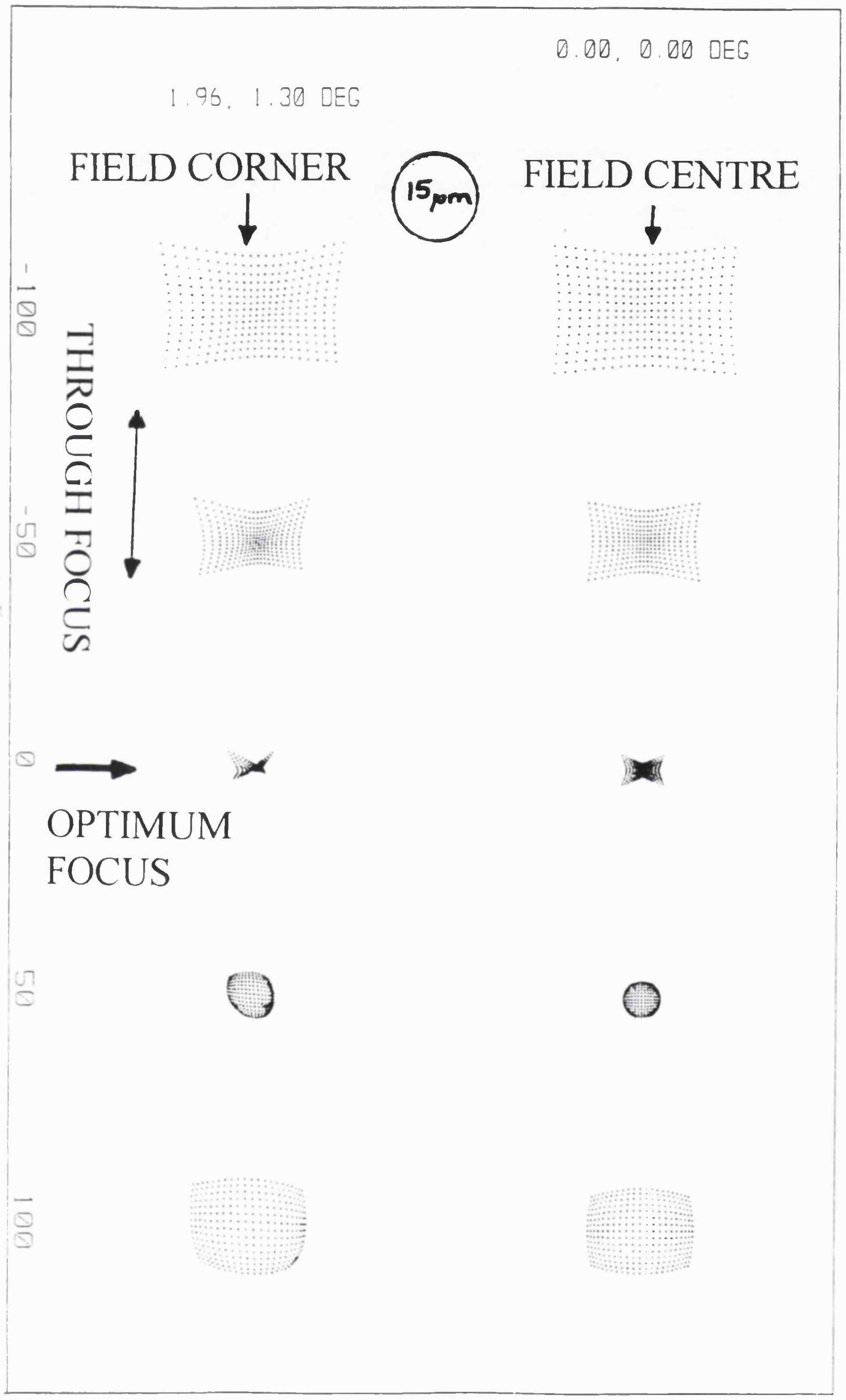


Figure 5.7 : Blur Spot Diagrams at 5500 Angstroms for a Houghton Camera of 1310mm Focal Length

The designer of an off-axis camera must develop a camera with slightly less than half the focal ratio of the required camera, and then evaluate its performance in sub-aperture mode. In these systems, it is advantageous to locate the field flattening lens at approximately the same distance in front of the spherical mirror as the doublet corrector. This is in order to ensure the largest possible focal ratio for the parent camera. Locating the lens closer to the mirror requires a greater offset for the unobscured camera to ensure the pupils are not vignetted at the field flattening lens obstruction. This would indicate the need for a smaller focal ratio parent camera, which is more difficult to design within aberration tolerances. We should note that the imaging performance of an sub-aperture camera is not simply extrapolated from the on-axis parent design. The pupil has been stopped by approximately one third so it seems logical to propose that the spots should be proportionally smaller by the same degree. This is not the case as an off-axis design has a greater proportion of pupil energy (i.e. more rays) concentrated in the outer regions of each pupil. This can result in blur spot diagrams with a different energy structure than was the case in the on-axis design. As a result the geometric blur spot size in this case is reduced by only around 5 microns when shifting the aperture.

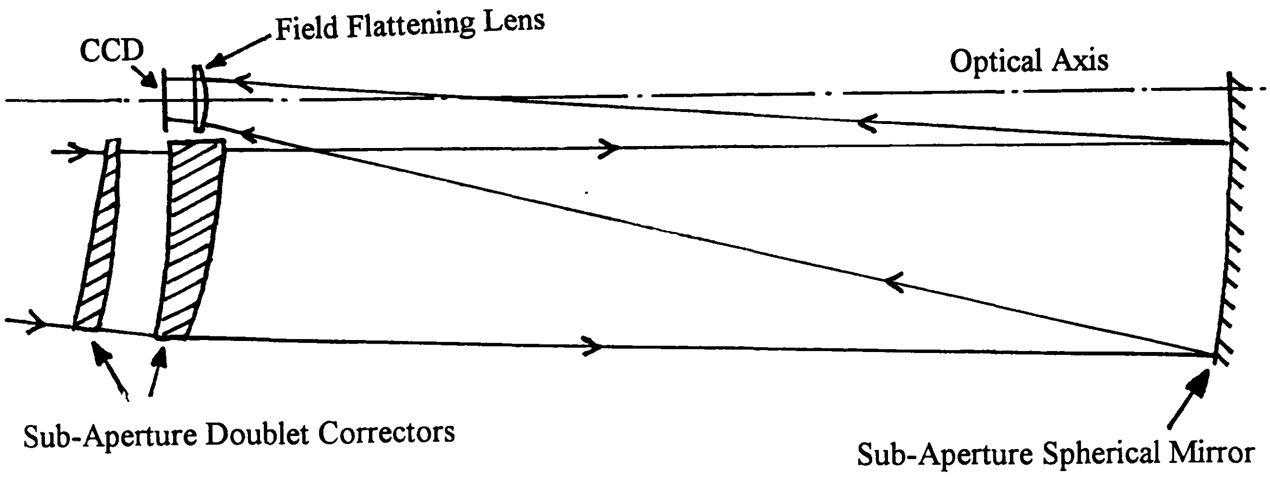

Field Flattening Lens
CCD
Optical Axis
Sub-Aperture Doublet Correctors
Sub-Aperture Spherical Mirror

Figure 5.7: A Sub-Aperture Camera With An Unobscured Pupil

	Radius of Curvature	Separation	Media
Surface 1 = Echelle as Stop	Infinite	0	Air
Surface 2	-962.881	1085.92	Air
Surface 3	-1890.681	13.031	Fused Silica
Surface 4	-2584.467	69.881	Air
Surface 5	-1180.833	65.155	Fused Silica
Surface 6	-2849.409	1376.940	Air
Surface 7	-408.976	-1400.830	Air
Surface 8	-5379.236	-8.680	Fused Silica
Image Plane	Infinite	-36.027	Air

Table 5.4 : The Parameters of an Unobscured Camera of 1310mm Focal Length

The geometric blur spot radii for this camera are 23, 23 and 33 microns at 5500, 11000 and 3000 Angstroms respectively, in the plane of the echelle dispersion. (Errors on these values are less than +/- 1 micron) blur spot diagrams at these three wavelengths are shown in Figure 5.9, and each spot is plotted next to the 45 micron projected slit for HROS. These spot diagrams are for a back focal length of 36.075 mm at which the optimum focus was found, and as such assume no refocus for different wavelengths.

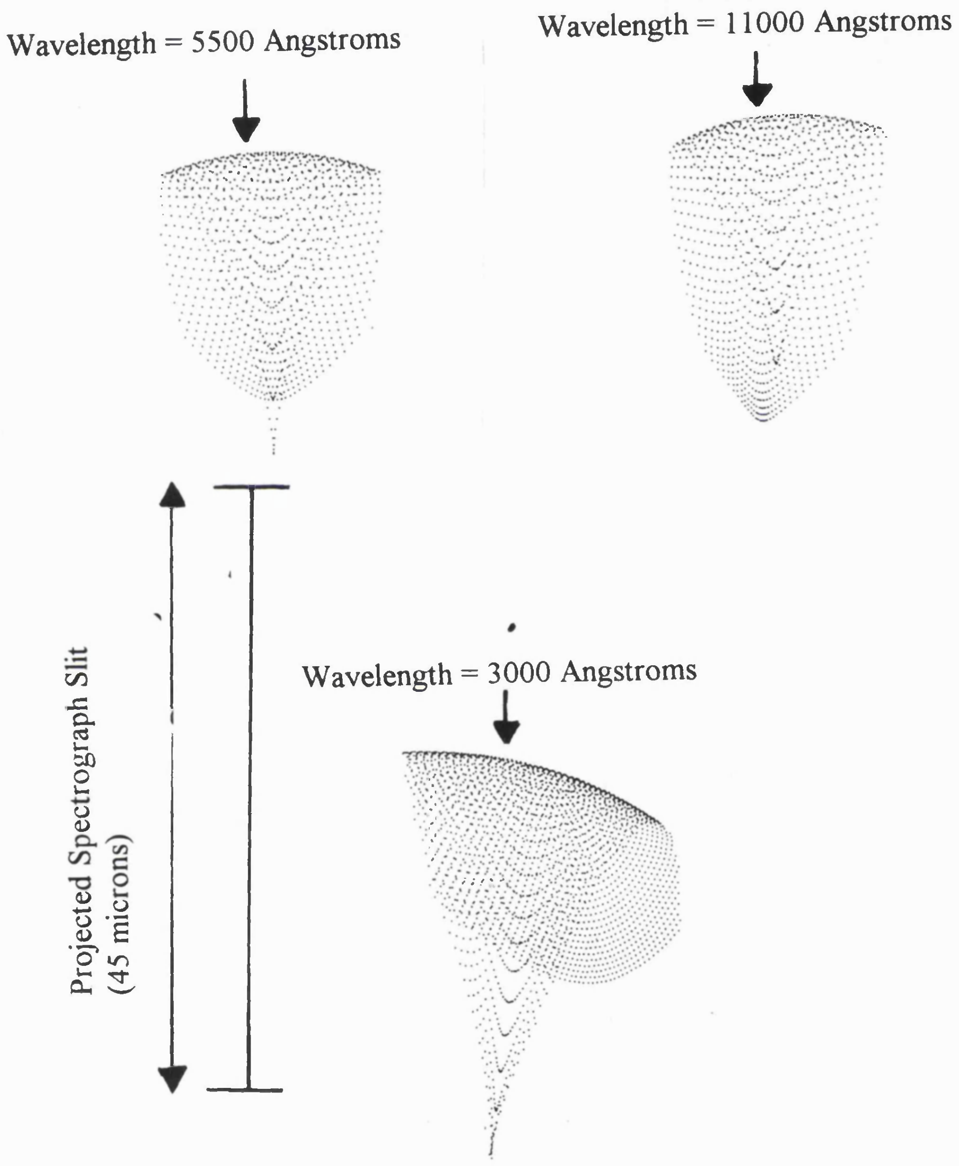

Wavelength = 5500 Angstroms
Wavelength = 11000 Angstroms
Wavelength = 3000 Angstroms
Projected Spectrograph Slit
(45 microns)

Figure 5.9 Blur Spot Diagrams for an Unobscured Camera of 1310mm Focal Length

From these results we conclude that this camera is compatible with a slit sampling factor of 2.5. These results show that greater than ninety percent of the available energy falls within 20 microns at 3000 Angstroms. However when contrasted with the on-axis alternative camera, a gain of around 12 percent transmission is achieved through not having a CCD obstruction. In light of this fact we conclude that this sub-aperture design is as efficient as its alternative on-axis form at 3000 Angstroms, and exhibits significantly better throughput in other wavelength regions. This camera form exhibits acceptably small levels of spherochromatism, coma, field curvature and axial colour. The dominant residual aberration in this camera is spherical aberration. This uncorrected zonal spherical aberration interestingly becomes evident in the blur spot diagrams in an unfamiliar manner. The blur spot diagrams appear comatic in shape as a result of an offset aperture. One significant advantage of the off-axis design is that the camera would allow free access to the CCD. This allows the rapid interchanging of CCD detector arrangements between exposures. On long integration exposures the turnaround time of such an instrument will be significantly improved over contemporary spectrographs.

5.4 Conclusions: The HROS Camera Feasibility Study

5.4.1 Optical Implications of Camera Use On the HROS Instrument

In this chapter we have contrasted the properties of several different kinds of spectrograph camera. This study was completed with the aim of identifying the form of camera that offered the best overall optical performance for the HROS instrument. In this study, all transmitting lens cameras were not considered as these systems require a large number of elements (i.e. greater than 3) to achieve adequate imaging performance. Optical systems containing many separated optical elements are associated with a high transmission loss, and additionally stray light problems. Additionally good wide field imaging performance is difficult to achieve in lens systems. Historically therefore, the spectrograph designer has turned to various forms of catadioptric cameras for best results. This Chapter has explored the properties of a variety of such systems, in order to demonstrate the feasibility of a camera for the HROS instrument. The envisaged HROS specification at the time of this work [49] required a primary resolution of 120,000, dictating a camera of focal length 1300mm and a speed of f-number 6.0 in monochromatic light. The designs tested here are significantly faster at other wavelengths.

It is important to recognise that this feasibility study has not modelled these cameras in the same way as they would be used on the spectrograph. The imaging properties of the cameras presented here would be altered due to the optical dispersion occurring at the prisms present in the real instrument. The primary effect of this will be that the polychromatic pupils on the echelle will be displaced due to dispersion produced at the prism. This effectively increases the height of rays at the camera correcting plate, producing an increase in image aberrations. A secondary, less important effect is that the entrance pupil of the camera is complex, and located at a different distance from the prism in the cross dispersing plane. This latter effect is unlikley to have significant effects on the camera aberrations.

The effects of modelling these cameras on a rigorous model of the HROS instrument are unknown in detail. However since all the designs were optimized to a very high degree of correction, it is unlikely that camera imaging performance will deteriorate to such an extent as to make the cameras untenable. Additionally it is likely that the oblique rays, which have largest impact on blur spot size, will be largely vignetted at the corrector plate of the camera, and so not deteriorate the image quality. Establishing the best camera form for HROS requires a critical review of the most promising types of catadioptric systems. In the next section we overview the findings of the author's work in this regard.

5.4.2 Critical Comparison of the Key Performance Characteristics of the Alternative Camera Forms

In order investigate the feasibility of the HROS instrument, the author carried out an extensive ray tracing study. This involved the systematic exploration of the characteristics of three different types of camera, namely variants on the Schmidt, Maksutov and Houghton types. Other types of camera have been applied to wide field optical system design, however these largely involve adding additional surfaces to give the designer extra degrees of freedom. In this study, obtaining maximum camera transmission was a key design goal. As a result the author attempted to minimize the number of optical surfaces present in each design. Therefore we have not explored camera solutions that have a Cassegrain mirror, or internal folding mirror present. Additionally we have not looked at designs with full aperture triplet correcting elements or other derived forms.

For the purposes of this study, we explored in detail the properties of the Short Schmidt camera. The characteristics of many hundreds of different Schmidt cameras were analysed, before settling on the best design for HROS. The Schmidt design has a well-known residual aberration called spherochromatism that limits the useful speed range over which it can be used. However for HROS, the camera did not require a very low f-number, and this aberration did not degrade the performance of this camera. However the Schmidt design has another intrinsic drawback. This is the difficulty of manufacturing the aspheric plate, which requires precise polishing to produce the required figure. Another drawback that is specific to this particular application of the Schmidt form, is that during the design phase the camera length had to be traded off against field flattener thickness, and hence field flattener aperture and beam obstruction. This caused the optimized design to end up over 2 metres in length. Overall the Schmidt form was found to be applicable for the HROS instrument as a long focal length camera for use at this resolution.

The next family of designs we investigated included cameras based around the Maksutov optical system. These systems are characterized by having a single full aperture meniscus element to correct the spherical aberration of a spherical mirror. The author modified Maksutov's original system, by adding a field flattening lens close to the focal plane. After exploring the imaging characteristics of many hundreds of different designs using the Inter-Opt software, the author located a design offering excellent imaging properties. The limiting aberration for systems of this type is often higher order spherical aberration, which prevents the systems application at very low f-number. In this particular design case however, higher order spherical aberration did not prevent achievement of the necessary imaging specification. This specific design has a key advantage over the Schmidt form, in that it is shorter in overall length.

The author explored the properties of another type of optical system. This is called the Houghton camera, and uses a doublet corrector to remove the mirror's aberrations, thus giving the designer two extra surfaces with which to balance aberrations. Again the author modified the original Houghton system by adding a field flattening lens close to the focal plane. This type of system has intrinsic imaging advantages when contrasted with the other two forms of camera. The two extra surfaces enable the designer to precisely balance aberration contributions from specific surfaces against those being produced elsewhere in the camera. The limiting aberrations of both Schmidt and Maksutov can be eliminated by careful design techniques. For any particular combined doublet shape and power, spherical aberration and axial colour can be effectively eliminated. Additionally lens bending often allows the reduction of coma to a low level. Importantly the Houghton design also enables spherochromatism to be completely eliminated in systems down to an f-number of 1.0, whereas correction of this aberration becomes impossible beyond an f-number of 2.0 in the Schmidt design form. The Maksutov form requires a highly curved meniscus element, which causes large amounts of higher order spherical aberration in fast systems. In contrast careful design of a Houghton system enables this aberration to be eliminated, even in systems with an f-number as low as 1.0. As a result of these intrinsic advantages, in general the Houghton design is preferred if superior imaging performance is the key criterion. However in the specific case of the long focus camera, imaging performance of the other two types of camera was found to be adequate for the HROS instrument. Comparison of the different designs is made on other criteria in this case therefore.

Notably for the long focus camera, the Houghton design has more relaxed manufacturing tolerances than the competing camera forms. Additionally in this specific case the Houghton design was found to offer slight throughput advantages, and so is the preferred choice for this instrument. The author also researched the properties of sub-aperture camera forms that are recognized through not having a central obstruction due to the presence of a field lens and detector. The author developed a new type of camera arrangement for specific application to this study [47]. This arrangement has throughput advantages over the alternative on-axis camera types. Additionally, these sub-aperture designs may prove beneficial in other areas, as they offer advantages resulting from the accessibility of detector arrangements. These cameras do not require evacuation, as the CCD detector can be cooled directly. A sub-aperture design form was shown to be a good alternative when compared to the axially symmetric systems.

This preliminary study has demonstrated the feasibility of a variety of camera forms for HROS at a resolution of 120,000. Additionally an interactive approach to optical design has facilitated a complete exploration of the properties of each design form. This work has identified several new forms of Houghton camera that may prove applicable in other areas of spectrographic instrumentation. Exceptionally fast, wide field cameras are now required in spectrographic instrumentation. Design of faster camera systems remains an active research area as a result. In the next section we draw together the results of this camera research study, in order to help identify design routes to these new systems.

5.4.3 Recommendations for the HROS Short Focal Length Camera

Spectrograph feasibility studies like those described above are largely centred around finding a camera configuration that can supply the required optical imaging performance. This work has focused on locating the optimal configuration of the various optical components for the HROS instrument at resolution 120,000. In addition to the long focus camera, the HROS instrument is required to have a shorter focal length camera of around 500mm to provide a secondary resolution of 50,000. This camera will be required to provide adequate imaging performance at a polychromatic f-number of around 1.0. The systematic design studies completed in this chapter, have shown that the Houghton camera is best suited for this purpose. Preliminary ray tracing studies performed by the author on a similar camera for the

UCLES spectrograph (f-number 1.0, and 500mm focal length) have shown such a camera is compatible with a worst image spread over a 7 degree field of around 30 microns. Based around the preliminary work described in this Chapter, final optimization of a short focal length camera for HROS is currently ongoing at OSL.

5.4.4 Camera Design Recommendations in General

The camera feasibility studies described in this Chapter have been a major part of the design studies undertaken by OSL for the HROS instrument [49][72]. These studies were central to demonstrating the optical feasibility of HROS. This instrument is employing a novel configuration that pushes camera designs to the limit in terms of imaging performance. The author adopted a systematic design strategy for these studies. This approach began by eliminating lens cameras and other multi-element systems before moving on to explore the properties of hybrid systems based around three common camera forms. Of these three forms, the Houghton system clearly offers superior aberration correction and improved imaging performance at any particular focal length and relative aperture. On the basis of imaging, manufacturing and transmission criteria, for the HROS instrument a modified Houghton camera is the preferred form. This is the case for the detailed study performed on the long focal length camera. Additionally it is likely that a similar modified Houghton camera will be the preferred choice for a short focal length camera. This is because the other Schmidt and Maksutov forms are unlikely to meet the stringent imaging specification required at the lower resolution mode. Additionally the author has developed the concept of a new form of sub-aperture camera for this instrument. Sub-aperture camera forms may in the future offer key advantages to the instrument designer. In particular, the advent of even larger format CCD detectors may push for camera designs that do not suffer the detector vignetting effects to quite the extent that would be expected. This Chapter has reported on the results of the application of a new interactive approach to camera design. These methods have allowed systematic comparison of the properties of three different camera types. Careful comparison of these different camera forms has highlighted the relative merits of each type. Contrasting the different forms has enabled identification of the most suitable type for use within the HROS instrument.

Chapter 6: Optical Design and Testing of a Short Schmidt Camera of F-number 1.6

6.1 Introduction

This Chapter describes the optical design of an Short Schmidt camera to replace an existing camera within an echelle spectrograph in Mexico. The author performed final optimization of this camera using the interactive optical design program detailed in Chapter 2. Following the optical and mechanical design phases this camera was constructed and tested within the workshops in the Optical Science Laboratory.

6.2 Camera Optical Design

The 2m telescope at San Paulo in Mexico is mounted with an echelle spectrograph. Prior to 1993 this instrument employed a folded Schmidt camera with a design fault that revealed itself in the form of excessive vignetting at extreme field positions. Francisco Diego and the author collaborated on the optical design of a new Short Schmidt camera for this instrument in order to attempt to correct this defect. The layout of the new camera is shown in Figure 6.1. The camera consists of a fused silica correcting plate, folding flat, spherical mirror, and field flattening lens. The majority of the optical design work was completed with Kidger ray tracing software [34]. Once this initial design phase was complete, the various optical components were manufactured. Prior to assembly the author modelled the camera using the Inter-Opt program, and again re-optimized the arrangement interactively using this software. The only parameters that could be altered were optical separations at this stage. However it is well known that coma can sometimes be reduced in Schmidt systems of this type by altering the mirror-aspheric plate separation.

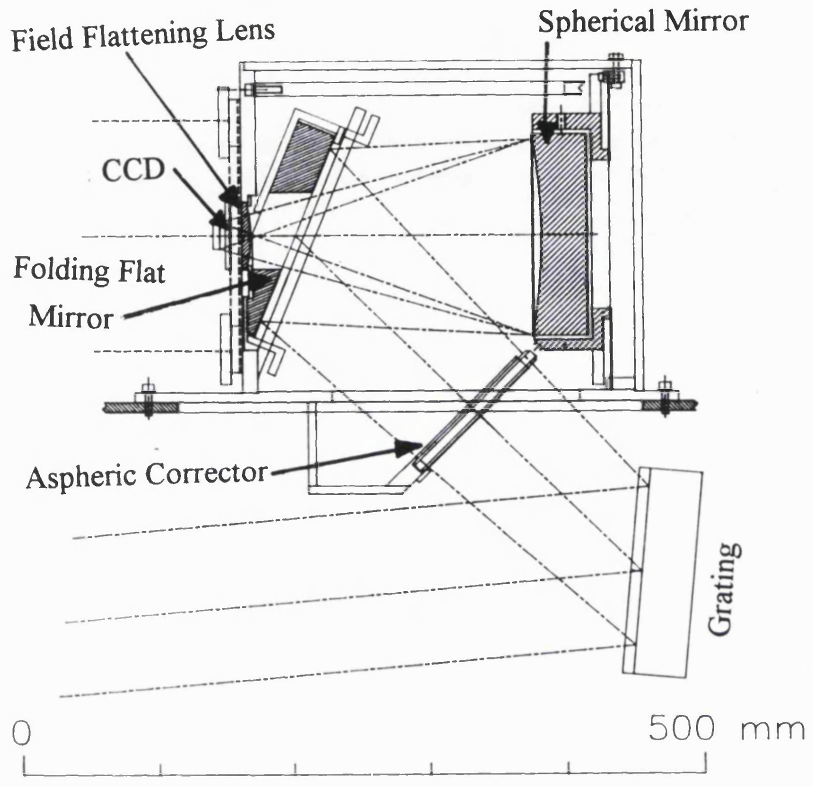


Figure 6.1 : The Optical Layout of a Short Schmidt Camera of F-number 1.5

Prior to optimization the camera displayed a large amount of residual coma at extreme field positions and the blur spot size was greater than 50 microns at extreme wavelength positions. After interactive optimization using Inter-Opt, the new arrangement showed a significant reduction in coma, with blur spot diagrams around twenty-five microns in size. In figures 6.2 and 6.3 we can see the blur spot diagrams for this camera before and after final optimization. Each spot is shown in three colours, representing wavelengths of 3000, 5500 and 7000 Angstroms, in blue, yellow and red colours respectively.

It is important to note that Inter-Opt traces each pupil at three wavelengths in polychromatic spot mode, as is normal for such a ray tracing program. However in a spectrograph camera, each pupil is monochromatic. Evaluating the camera performance requires that the user only think in terms of single colour blur spots at appropriate field positions. The off-axis imaging was found to be improved by 50 percent if the distance between mirror and plate was altered by only 10mm. This fact only became apparent when the author altered the modelled corrector plate's position using Inter-Opt software, whilst simultaneously watching how the aberrations changed in real time. As a result of these ray tracing tests the camera specification was altered to include the new separation between mirror and the aspheric plate. The components were then assembled to take into account this new arrangement. The aspheric plate is shown in Figure 6.4, whilst assembled components are shown in Figure 6.5.

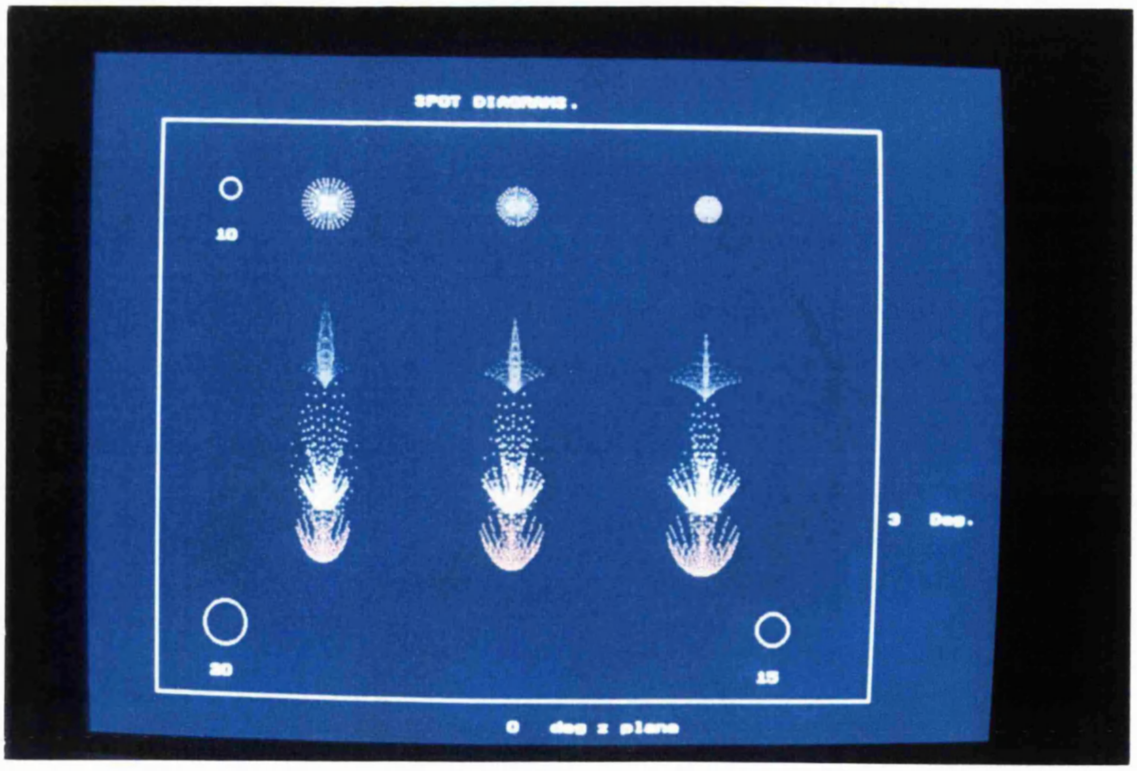


Figure 6.2 : Short Schmidt Blur Spot Diagrams Before Final Optimization

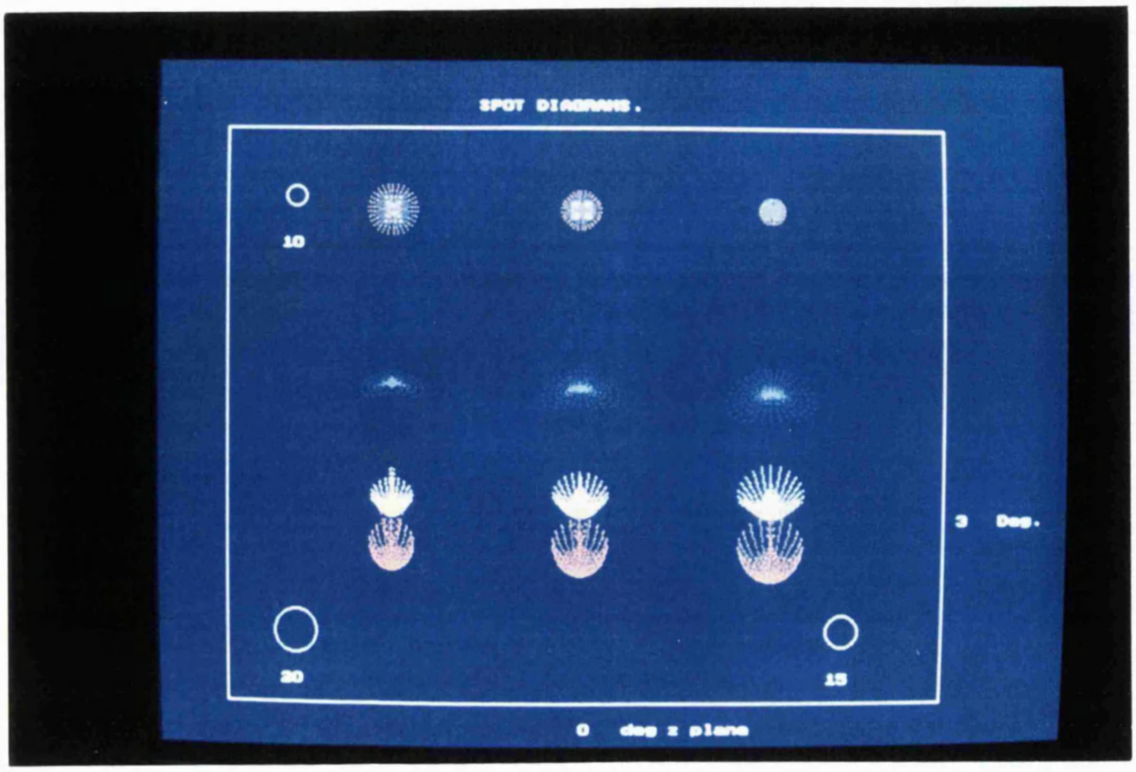


Figure 6.3 : Short Schmidt Blur Spot Diagrams After Final Optimization

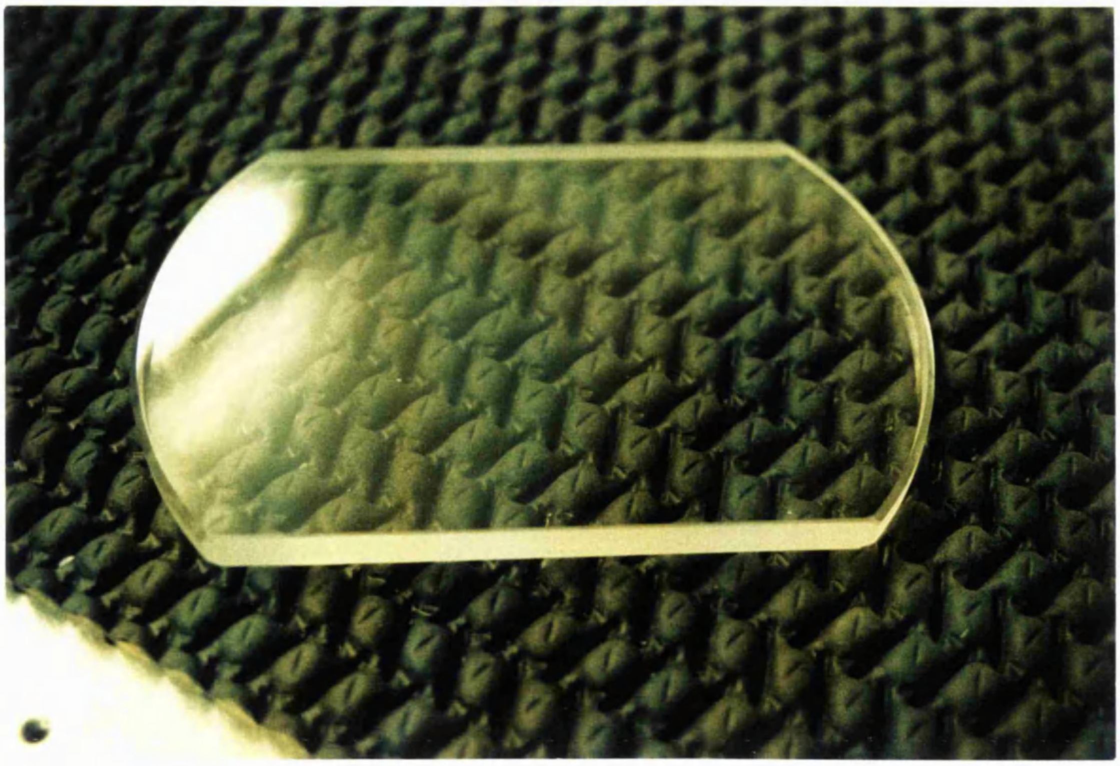


Figure 6.4 Photograph of the Fused Silica Correcting Plate

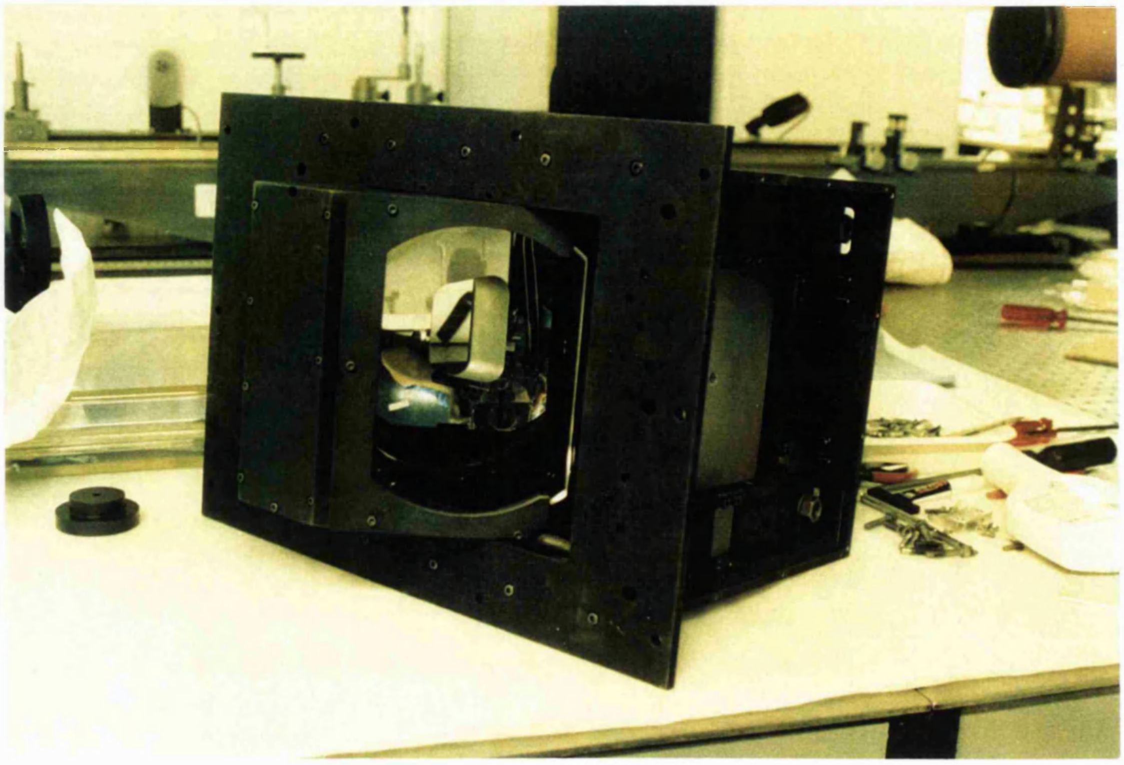


Figure 6.5: Photograph of the Assembled Schmidt Camera Components

6.3 Optical Testing Using a Celestron to View the Focal Plane

Optical alignment and testing of the camera were undertaken prior to shipping to Mexico. Optical testing was qualitatively achieved by placing an illuminated 20 micron ball bearing exactly at the focal plane of the Schmidt. This ball was then viewed by aligning a Celestron telescope at the correct angle, and looking through the eyepiece. This gave a magnified view of the ball about ten times bigger than reality, due to the relative ratio of focal lengths of Schmidt and Celestron systems. The presence of any aberration would be apparent using this technique. A variety of field positions were tested by moving the ball bearing to the corners of where a CCD would be located, and viewing the ball image. Using this method aberrations were found to be acceptably small, of the order of twenty microns or less.

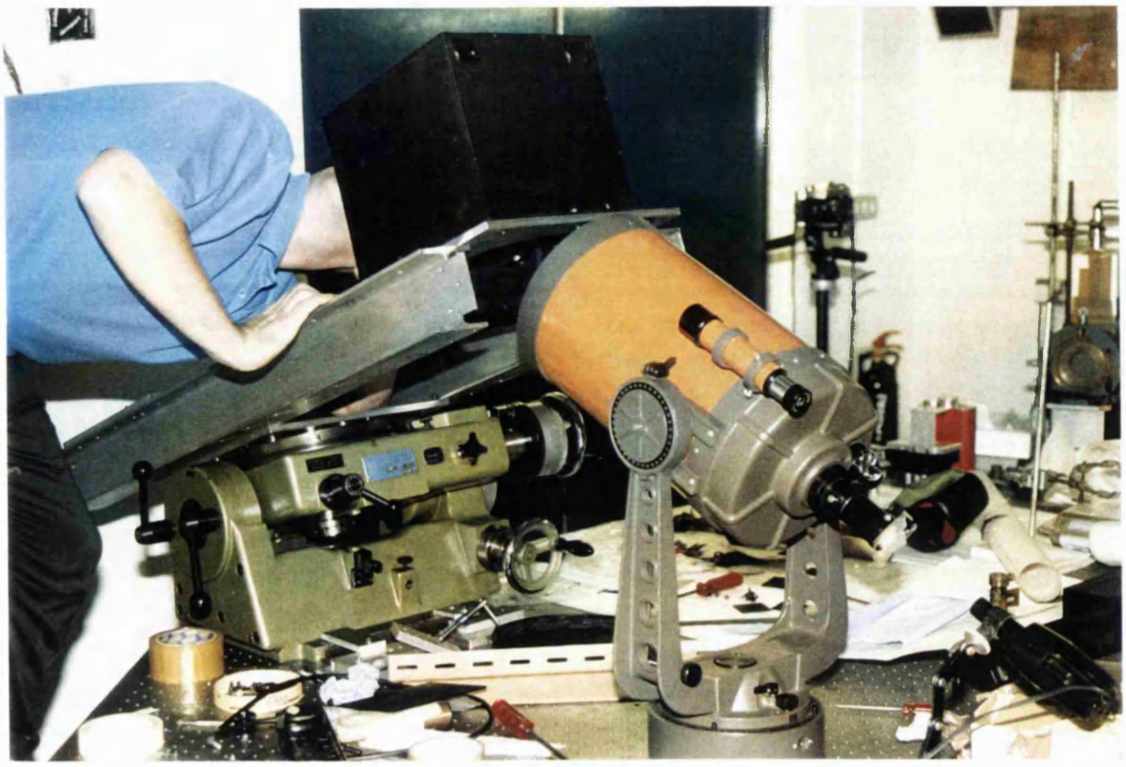


Figure 6.6 : Optical Testing Using a Celestron to View the Focal Plane Images

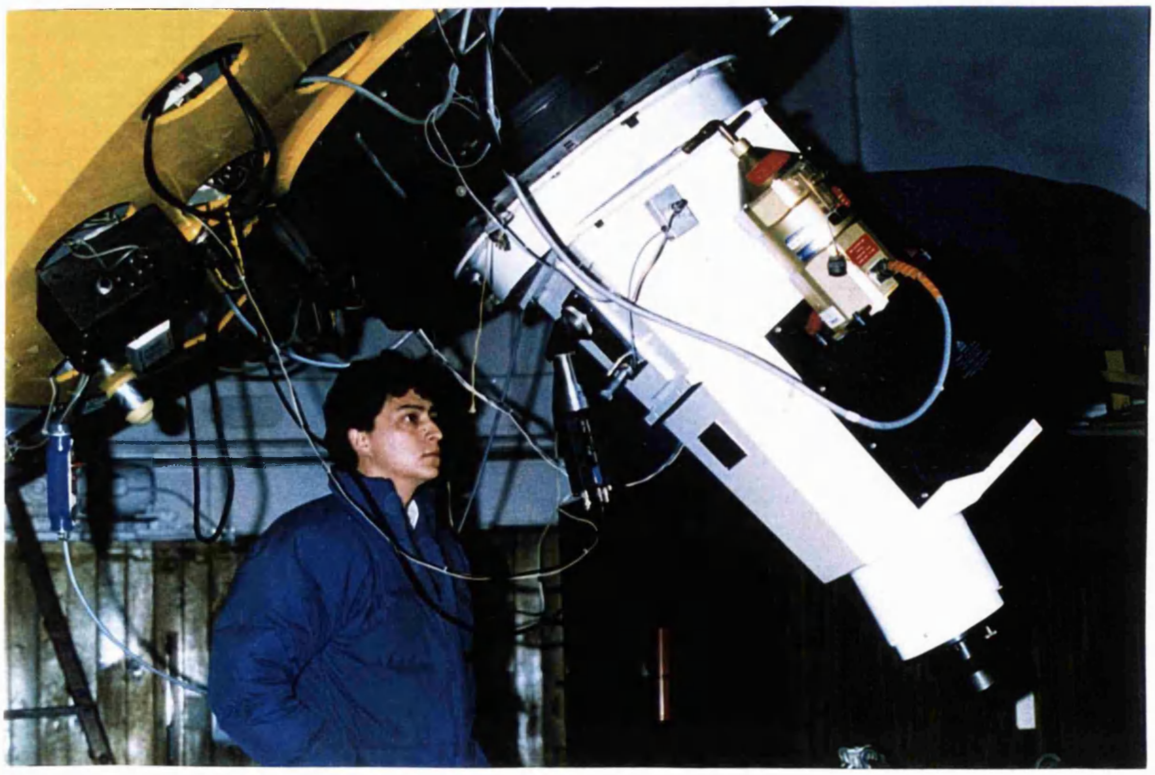


Figure 6.7 : The Schmidt Camera Located at the Cassegrain Focus of the Telescope

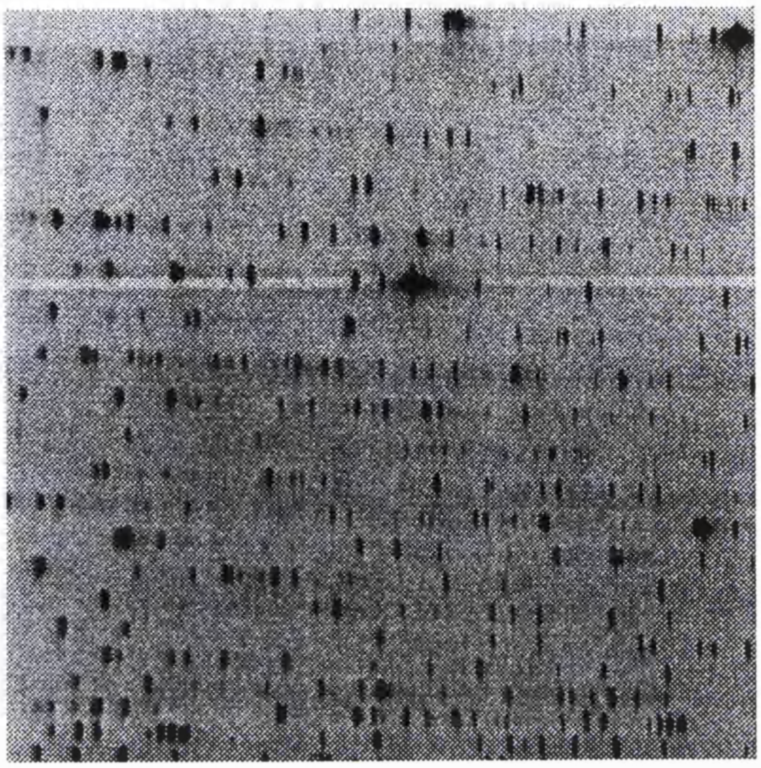


Figure 6.8 : Photograph of a Helium-Argon Calibration Lamp Spectrum

6.4 Commissioning Results

The new camera has now been installed on the spectrograph and a number of stellar objects observed using a CCD. Figure 6.7 shows a photograph of the camera in position at the Cassegrain focus of the telescope. Whilst Figure 6.8 shows a spectrum as recorded by the camera for a Helium-Argon comparison lamp. Initial examination of various stellar spectra indicates that the camera is working in accordance with the predicted theoretical ray tracing results and subsequent bench testing at UCL. In Figure 6.9 a wavelength calibrated spectrum taken with the new camera is shown for the star P-cygni.

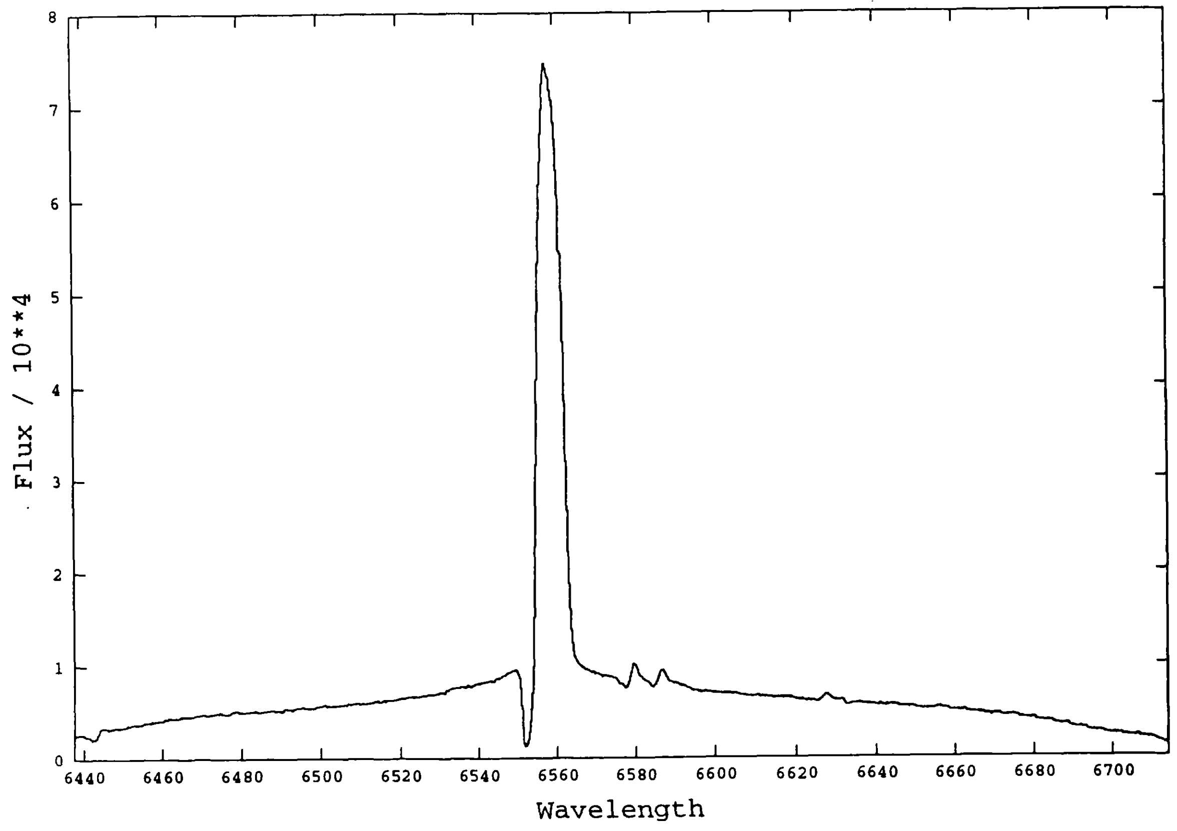


Figure 6.9 : P-Cygni Wavelength Calibrated Spectrum

The commissioning data has shown that this camera has significantly lower levels of vignetting losses for off axis field positions than the previous camera. As a result the camera is obtaining superior signal to noise ratios within individual spectral lines close to the CCD edges.

6.5 Conclusion

Optimization of the Short Schmidt camera was completed using an interactive design approach that successfully reduced the off-axis aberrations by a factor of two. Using the Inter-Opt program the author was able to alter the mirror-corrector separation in the ray tracing model, whilst watching the effects on aberrations in real time. This final optimization process was completed in only a few minutes. The procedure could have been achieved using conventional ray tracing software, however using these programs the task would have taken considerably more time to complete.

The commissioning results for the new Schmidt camera show imaging performance is in accordance with that predicted by ray tracing with the Inter-Opt software.

Chapter 7

Summary and Conclusion

This thesis has been concerned with certain aspects of echelle spectrograph design for 8m class telescopes. Chapter 1 reviewed the history of spectrograph development, and discussed the scientific goals for a new generation of high resolution instruments. These goals, combined with the large size and improved imaging performance of modern telescopes, push for novel spectrograph optical configurations. Thesis work has delt with a variety of new optical design techniques, in order to help identify viable design paths to successful instruments. Increasingly, spectrograph construction also involves the application of technologies from a number of different fields. In summing up the work presented here, it is therefore important to place thesis work in the context of current technological developments.

Greater light gathering power for optical telescopes is a clear world-wide objective. The Gemini and VLT projects plan to complete construction of four separate 8m telescopes around the turn of the century [45][51][74]. Projects of this type will forge an exciting new direction for ground based optical astronomy. Emphasis will be on advances in active and adaptive optics technologies, in order to fully realize the potential of these large telescopes. Significant progress has already been made in the adaptive optics field for astronomy. A successful working system has now been in operation on the CFHT telescope for several years now in Hawaii. Additionally the Keck adaptive optics program aims to demonstrate the feasibility of these techniques for a very large aperture telescope [75]. Building on the work of these groups others are now preparing to implement similar adaptive optics systems for 8m class telescopes [16]. At the same time, important advances have also been made in the active components field for astronomy. Active control systems for the 10m segmented Keck mirror have proven highly successful, [10] and other systems are now under development for lightweight monolithic 8m class mirrors [74].

Additionally, real time active control for instruments located at Cassegrain is likely to be an area of progress in the near future. Preliminary studies indicate that instrument flexure, previously associated with Cassegrain spectrographs, can be largely eliminated through a combination of passive and active methods [68]. The ray tracing work described in Chapters 3 and 4 of this thesis has demonstrated the feasibility of a Cassegrain location for the HROS instrument. A Cassegrain spectrograph has certain advantages when compared to the Nasmyth alternative. These intrinsic benefits arise from the lower number of oblique reflections, and the intrinsically smaller slit size of a Cassegrain spectrograph. Notably as a result, Cassegrain spectrographs have greater potential for increased throughput, in addition to pure polarization performance. As a result HROS may become the first of a new breed of large high resolution optical spectrographs mounted at Cassegrain.

The Ultra High Resolution Spectrograph has now been in operation for over 1 year at the AAT. This instrument has successfully obtained observations of the B Pictoris circumstellar Ca 2 K line at a resolving power of 900000. This circumstellar line profile was resolved for the first time using this instrument with the aid of a new type of image slicer that provides significantly higher signal throughput [12]. Additionally the Keck HIRES instrument was recently commissioned, and is also being used for unique science. HIRES has been used to observe metal-poor halo dwarf stars, in order to study the abundance of Beryllium in the early universe. Both high resolution and high signal to noise were required to allow these types of observation.

HIRES is obtaining speed gains greater than 50 over previous observations of these near ultra-violet lines [62][63].

Encouraged by these discoveries a number of new projects are now underway to construct high resolution spectrographs for large aperture telescopes. A high resolution spectrograph is now being planned for use on the planned 10m aperture Spectrographic Survey Telescope (SST) [61]. Another high resolution spectrograph is currently under construction for the 4m aperture Galileo National Telescope (TNG) [23]. Additionally detailed design work on the HROS spectrograph for the southern Gemini telescope is currently being undertaken at OSL. This instrument should begin operation at the turn of the century, and aims to take four years for design and construction. Taken together, these projects will forge an exciting and informative future for high resolution spectroscopy as these new instruments come on line.

The trend towards larger telescopes and associated instrumentation drives detector manufacturers to investigate physically larger format devices. Large area CCDs are now used with dimensions of 30mm by 60mm and 2K by 4K pixels. These devices are also purposely made double edge buttable to facilitate mosaicing to form even larger detector formats. The HROS camera feasibility studies discussed in Chapter 5 were undertaken with such detectors in mind. At the outset the feasibility of a long focal length camera with good wide field imaging performance was unproven. The author's work has demonstrated the optical feasibility of a such a camera for HROS at a resolution of 120,000. Additionally a sub-aperture camera design was shown to be a viable alternative at this resolution. This latter design has the advantage of detector accessibility, and may allow for more efficient turn around between exposures, through rapid detector exchange. Further advances in large format detectors are likely, and as a result in the future even larger field camera designs will require investigation. CCD mosaicing has already enabled one group to develop a very large device of 8196 by 8196 pixels, which is 120 square mm in size [6]. Additionally in the future CCDs are likely to have a reduced pixel size down to 7 microns, further constraining the acceptability of large aberrations in camera design. Optical camera designs will increasingly be pushed to the limit as a result, and new design methods are required to locate these high performance solutions.

The ray tracing program described in Chapter 2 of this thesis was developed to facilitate spectrograph camera optical design. This is achieved through a new user-interface that enhances the dialogue between user and computer during the process of camera optimization. User-computer dialogue is improved within this interface by the introduction of two metaphors. These analogies help the formation of a set of common assumptions between user and computer, in order to aid communication in a natural way using concepts with which the user is already familiar. The first analogy is a fluid-lens that allows system properties to be changed in real time. Secondly a parameter-space map visualization, where the program user maneuvers through parameter-aberration space as if in an aeroplane over mountainous terrain. These two analogies are used to aid conceptualization by the user of the language of the interface. Through these concepts the central aims of the new user-interface are implemented, these being real time optimization, and improved visualization of the parameter-aberration space of a camera type.

During functional design of the user-interface, a new set of methods for representing the problems of optical camera design were developed. For instance the fluid-lens aberration screen allows real time display of aberration contributions on the HTanU graphs, a unique facility offered in this program. These aberration contribution graphs allow the user to quickly discover specifically which elements in a system contribute most directly to a certain aberration size. Another unique functional part of the design of the user-interface was the calculation, and subsequent representation of spherochromatism as a line length in the parameter space landscape. These visualizations uniquely enable the user to conceptualize 6 dimensions in the design space simultaneously. Spherochromatism is especially difficult to remove from high aperture cameras using normal optimization techniques. However this method of representation made simultaneous correction of spherical aberration, spherochromatism, axial colour, and Petzval field curvature relatively straight forward. Maneuvering through a 3 dimensional landscape visualization of the design space of a camera form, has been found to be a highly efficient optimization technique. This analogy is particularly natural to the human user, and has been found to be a very powerful concept around which to base user-computer dialogue. The human user discovers that real time design within such a 3 dimensional virtual world, is a place where his/her natural skills can be applied with great success. The possibilities for representation of many more dimensions of aberrations with this technique remain unexplored. Extra lines and 3 dimensional geometric shapes could be added to represent new aberration dimensions. Whilst implementation of a 'space ball' three dimensional mouse in future programs may aid efficient maneuvering within these visualizations. A new area within the optical design field is anticipated from this work. Real time optimization techniques may become an efficient addition to the optical designers tested set of conventional methods and techniques.

Increased user friendliness is a common feature of modern optical ray tracing programs. Efficient graphical user-interfaces within these programs now facilitate rapid analysis of optical systems. Programs like Zemax and Optica now offer a wide range of features that aid system evaluation [80][81]. These features provide detailed graphical information that helps the user decide when a design is suitable for a particular task. However optical design is a difficult and complex process. Successful system design requires that the program user possess significant skill, often built up over many years of experience and training. Skills of this type are increasingly required by people from other fields, who must attempt system design without this experience. For success these people must be rapidly trained in the techniques of optical system design. Programs like Inter-Opt, that provide a highly interactive design approach, can help these novice users acquire this expertise in shorter time spans. This is achieved as users become more familiar with the kinds of problems involved in system optimization. Additionally interactive programs like Inter-Opt can be very useful in the various teaching fields of optics. Here students will have the opportunity to participate in, and visualize more fully, the complex mathematical relationships involved. Students knowledge will increase at faster rates, due to increased conceptualization of the theories involved, and also new familiarity with the multiple factors involved during system design.

Modern astronomical instrument designs are becoming ever more complex, with sub-aperture, off-axis and multiple configuration optics becoming more common. For these instruments, the numerous ways in which system components interact and behave is now a significant design issue. Designer comprehension of these problems is an area of difficulty, and the techniques of virtual reality (VR) may help here. VR can allow a complete model of the system under study to be placed into the computer. The human user can then explore the properties, both optical and mechanical, of a variety of instrument configurations. The Inter-Opt program took its basic idea, (real time system design using three dimensional graphics) from the VR concept. As such it may be the first of a new generation of optical design programs that are capable of modelling complex optical systems, and allowing the user to explore their properties in real time. Such programs will aid rapid prototyping of complex instrument designs, allowing the designer to explore many more options than is currently possible. Unique optical instrument designs can be researched using these methods, and astronomical systems will be more fully optimized for a particular application.

The development of instrumentation for ground based optical astronomy is beginning an exciting phase. A new generation of large telescopes will be turned towards the night sky. Sensitive instruments mounted on the back of these giant telescopes will provide fresh information about faint astrophysical phenomena. A wealth of data will be provided, and newly efficient data reduction methods will allow rapid assimilation of the increased volume of information. Overall, these advances are certain to increase mankind's knowledge of the underlying processes at work in the Universe.

The thesis work presented here will hopefully provide a building block from which others can progress the instrument designs investigated by the author. It is hoped that these instruments will one day become a reality, and see a long and successful service within the astronomical community.

Bibliography

[1] J.B. Baker

A Family of Field Flat Cameras,

Proceedings of American Philosophical Society, 1940, Vol. 82,p330.

[2] E. Betensky

Post-modern Lens Design

Optical Engineering, August 1993, Vol.32, No.8, p.1750.

[3] R.Bingham

Grating Spectrometers and Spectrographs Re-Examined

Q.Jl. R.astr.Soc., 1979, 20, p 395.

[4] G.Black

Proc. Phys. Soc. 1955, B, 68, p.729.

[5] M.Born and E.Wolf

Principles of Optics,

Pergamon Press., 1991. p.256-375.

[6] T. Boroson , R.Reed, W.Y.Wong,

Development of a 8192 * 8192 CCD Mosaic Imager, 1994, p.877,

in Instrumentation in Astronomy V111,

Proceedings of the S.P.I.E. Symposium of Telescopes for

the Twenty First Century. Volume 2198, Part One of Two ,

Kona Hawaii, 1994.

[7] A. Bouwers

Achievements in Optics,

Elsevier Publishing Company Inc. , 1946. p.16-45.

[8] H.A.Buchdahl

Optical Aberration Coefficients

Oxford, 1954, p. 37-39.

[9] A.E.Conrady

Applied Optics and Optical Design,

Dover, New York, 1957., p90 -100.

[10] R.W.Cohen Et.Al.

Performance of the W.M.Keck Telescope Active Mirror

Control System., 1994,

in Instrumentation in Astronomy V111,

Proceedings of the S.P.I.E. Symposium of Telescopes for

the Twenty First Century. Volume 2199, p.83.

Kona Hawaii, 1994.

[11] F.Diego

U.C.L. Echelle Spectrograph (UCLES) for the Coude Focus of

The Anglo Australian Telescope : Optical Design and Performance

1988, Ph.D. Thesis., University College London.

[12] F.Diego

The UHRF : Spectral Resolution To the Limit ,1994, p.274.,
in Instrumentation in Astronomy V111,
Proceedings of the S.P.I.E. Symposium of Telescopes for
the Twenty First Century. Volume 2198, Part One of Two ,
Kona Hawaii, 1994.

[13] W.G.Driscoll
Handbook of Optics,
McGraw Hill, Inc., 1978. 7-84.

[14] S.D'Odorico, F.M.Moorwood, J.Beckers
Instrumentation For The ESO Very Large Telescope,
J.Optics (Paris), 1991, Vol.22, no. 2, p85-98.

[15] S.D'Odorico
Options For High Resolution Spectroscopy at Optical Wavelengths
At the ESO VLT, p. 193,
ESO Workshop in High Resolution Spectroscopy with the VLT,
Garching, 11-13 February 1992.

[16] B.L.Ellerbrook, Et.AL.,
Adaptive Optics Performance Analysis For the Gemini 8m
Telescope Project, 1994,
in Adaptive Optics in Astronomy,
Proceedings of the S.P.I.E. Symposium of Telescopes for
the Twenty First Century. Volume 2201, p.421.,
Kona Hawaii, 1994.

[17] D.P. Feder

Automatic Optical Design,

Applied Optics (December 1963), Vol.2,No.12,p. 1209.

[18] Foley , Van Dam, Feiner, Hughes

Computer Graphics, Principles and Practice, 2nd Edition,

The Systems Programming Series,

Addsion-Wesley Publishing Company,

Chapters 9 and 10.

[19] G.W.Forbes and A.E.Jones

Towards Global Optimization with Adaptive Simulated Annealing,

Intl.Lens. Design. Conf., G.Lawrence, (1990) Ed.,

Proc.SPIE 1354, p144.

[20] J.Fraunhoffer

Denkschr. akad. Wiss. Munchen, 8 (1821-1822). Ann.d.Physik,

74 (1823,337). reprinted in his collected works (Munich 1888),

51,117,1821.

[21] D.Gabor

Improvements in Photographic Objectives,

British Patent no. 544694.,1941.

[22] E. Glatzel and R.Wilson

Adaptive Automatic Correction in Optical Design, Appl. Opt. 7, 265 (1968)

[23] R.G.Gratton, R.K.Bhatia, A.Cavazza
High-Resolution Spectrograph For the Galileo National Large Telescope,
in Instrumentation in Astronomy V111, p309,
Proceedings of the S.P.I.E. Symposium of Telescopes for
the Twenty First Century. Volume 2198, Part One of Two ,
Kona Hawaii, 1994.

[24] W. Han
New Developments For CCDs,
Ph.D. Thesis, 1993, University College London.

[25] C.F.Harmer
Pupil Imagery in Astronomical Spectrographs,
Mon.Mot.R.astr.Soc. 1974, 167, p311 -318.

[26] G.R.Harrison
J.Opt.Soc.Amer., 39.,p.522.

[27] J.B. Hearnshaw
The analysis of Starlight,
Cambridge University Press, 1986, p.20.

[28] R.E.Hopkins
Global Optimization - is it a Misnomer ?,
Optical Engineering (August 1993) , Vol.32, no.8, p.1721.

[29] J.L.Houghton

Combinations of Spherical Lenses to Replace Non-Spherical

Refracting Surfaces in Optical Systems,

U.S.Patent 2,350,112 (1944)

[30] M. Iye

Instrumentation Plans For the Proposed Japan

National Large Telescope (JNLT).

in Instrumentation in Astronomy V111, p153,

Proceedings of the S.P.I.E. Symposium of Telescopes for

the Twenty First Century. Volume 2198, Part One of Two ,

Kona Hawaii, 1994.

[31] W.J.Kaufmann

Universe,

W.H.Freeman and Company,

New York, 1985, p.338.

[32] G.A.H. Kellner

Projecting Lamp. Patent Application no.969,785,

filed on August 2, 1907.

[33] Kodaira Keiichi

Outline of the JNLT Project

National Astronomical Observatory , Tokyo Japam

Astrophysics and Space Science, 1989, Volume 160, pp.137.

[34] M.Kidger

User Manual, for SIGMA 5.9 , An Optical Design Program

1992, Published by Kidger Optics LTD.

[35] H.C.King

The History of The Telescope,

Dover Publications, Inc. ,1955.p.10-25.

[36] R.Kingslake

Lens Design Fundamentals,

Academic Press, Inc., 1978. p123-135.

[37] G.Kirchoff

Monatsberichte Berliner Akad, (1859) p.662

[38] C.R.Kitchin

Astrophysical Techniques,

Adam Hilger., 1984. p.267-276.

[39] M. Laikin

The Future of Optical Design,

Optical Engineering (August 1993), Vol.32, No.8, p1729.

[40] E.H.Linfoot

Recent Advances in Optics,

Oxford. p.176-228.

[41] M. Loewy

Comptes Rendus de l'Academie des Sciences, (1883), 96, 735.

[42] A.Mackintosh

Advanced Telescope Making Techniques,

William Bell Inc. , 1977, p.240-250.

[43] D.D.Maksutov

New Catadioptric Meniscus Systems,

Journal of the Optical Society of America., 1944.,

Volume 35, no.5.,p.270.

[44] I. Newton

Phil. Trans. R.Soc.,6., 1672, (no.80), 3075

[45] G. Oertel

The NOAO 8M Telescopes , I,II,II , 1989,

Proposal to the National Science Foundation,

The Association of Universities for Research in Astronomy, Inc,

[46] L.Pasquini, A.Gilliotte,

CASPEC's New Look,

The Messenger, European Southern Observatory,

No.65, September 1991, p.50.

[47] A.S.Radley

Interactive Optimization of a Novel Sub- Aperture Spectrograph Camera,

1995, in Novel Optical Systems : Design and Implimentation,
at SPIE's International Symposium in Optical Science and Engineering,
Volume 2537, p.139.

[48] A.S.Radley
An Investigation into some aspects of a CCD Mosaic Imaging
Camera For use in Space Applications,
M.S.c. Thesis, 1991, University College London.

[49] A.S.Radley , R.Schmuzki, F.Diego, D.D.Walker,
Design Study for an Unfolded Cryogenic Camera and Immersed
Echelle For HROS. A Final Report.
U.C.L. Design study for the Gemini Project 1994.

[50] Raffaele G. Gratton, Et.Al.,
High Resolution Spectrograph For
the Galileo National Telescope, 1994, p309,
in Instrumentation in Astronomy V111,
Proceedings of the S.P.I.E. Symposium of Telescopes for
the Twenty First Century. Volume 2198, Part One of Two ,
Kona Hawaii, 1994.

[51] D.J.Robertson, C.M. Mountain
Gemini Instrumentation,
in Instrumentation in Astronomy V111, p143,
Proceedings of the S.P.I.E. Symposium of Telescopes for
the Twenty First Century. Volume 2198, Part One of Two ,

Kona Hawaii, 1994.

[52] H.Rutten and M.Van. Venrooij
Telescope Optics,
William Bell Inc., 1988.p106-107.

[53] B. Schmidt.
Mittenlugen der Hamburger Sternuarte, 1932,
7, no 36.

[54] D.J. Schroeder
An Echelle Spectrometer-Spectrograph for Astronomical Use
Applied Optics, Vol.6.,No.11, November 1967

[55] R.D. Sigler
Compound Catadioptric Telescopes with All Spherical Surfaces,
Applied Optics, 15 May 1978, Vol.,17, No.10, p.1519.

[56] W.J.Smith
Modern Optical Engineering,
McGraw-Hill Inc. , 1990, Vol.2, p.281-315

[57] W.J.Smith
Modern Lens Design : A Resource Manual,
McGraw-Hill, Inc., 1992. p.286-287.

[58] G.H. Spencer

A Flexible Automatic Lens Correction Procedure,
Applied Optics (December 1963), Vol.2, No.12, p.1257.

[59] L.M.Step Et.Al.
Gemini Primary Support System, 1994,
in Instrumentation in Astronomy V111,
Proceedings of the S.P.I.E. Symposium of Telescopes for
the Twenty First Century. Volume 2199, p.223.
Kona Hawaii, 1994.

[60] B.Stroustrup
The C++ Programming Language,
Addison-Wesley Publishing Company. ,1991.

[61] R.G.Tull
High Resolution Spectrograph for the Spectroscopic Survey Telescope
in Instrumentation in Astronomy V111, p674,
Proceedings of the S.P.I.E. Symposium of Telescopes for
the Twenty First Century. Volume 2198, Part One of Two ,
Kona Hawaii, 1994.

[62] S. S. Vogt
Keck Observatory Report No.101, 1993,
Berkeley; University of California Lawrence Berkeley Laboratory.

[63] S.S. Vogt Et.Al.
HIRES : The High Resolution Echelle Spectrograph

Spectrometer on the Keck Ten-Meter Telescope, 1994, p.362.
S.Vogt Et.al. in Instrumentation in Astronomy V111,
Proceedings of the S.P.I.E. Symposium of Telescopes for
the Twenty First Century. Volume 2198, Part One of Two ,
Kona Hawaii, 1994.

[64] S.S. Vogt,
The Lick Observatory Hamilton Echelle Spectrometer,
Instrumentation for Ground Based Optical Astronomy,
Proceedings of the Ninth Santa Cruz Summer Workshop
in Astronomy and Astrophysics,
July 13-July 24,1987 , Lick Observatory.

[65] G.Walker
Astronomical Observations : An Optical Perspective,
Cambridge University Press. , 1987. p.151 -177.

[66] D.D.Walker, R.G.Bingham, F.Diego
Proc. SPIE, Vol. 1235. 1990. p.535.

[67] D.D.Walker
Optical Instruments: Where We Go From Here,
Optics In Astronomy,
Cambridge University Press., 1993. p.89 -102.

[68] D.D.Walker, M. Dryburgh, B. Bigelow
Control of Flexure in the Gemini Instrumentation,

Design Study report commissioned by the Gemini Project Office,
Tuscon, 1992.

[69] D.D.Walker, R.G.Bingham , F.Diego, A.S.Radley,
Echelle Spectrographs for 8m Telescopes,
Proceedings of the E.S.O. Workshop in High Resolution Spectroscopy
with the VLT. Garhing, 1993., p.271.

[70] D.D.Walker, A.S.Radley, Francisco Diego, Andrew Charalambous,
Mark Dryburgh, Bruce C. Bigelow
A Re-examination of High Resolution Spectrographs For Large
Telescopes - The Cassegrain Solution,
in Instrumentation in Astronomy V111,
Proceedings of the S.P.I.E. Symposium of Telescopes for
the Twenty First Century. Volume 2198, Part Two
of Two, Kona Hawaii, 1994.

[71] D.D.Walker, A.S.Radley, F.Diego, M.Dryburgh, A.Fish,
D.Mills, ACharalambous
An Echelle Spectrograph For the MMT Conversion
U.C.L. Design Report for the MMT Telescope in
Mt.Hopkins Arizona., 1993.

[72] D.D.Walker, A.S.Radley, F.Diego
The High Resolution Spectrograph for Gemini
U.C.L. Design Study For The Gemini Project., 1993.

[73] S.Weller
Optics Software : The Next Generation
in Photonics Spectra, December 1992. p.112.

[74] R.West
The ESO VLT Project : Current Status
in The Messenger, 1986, no.46, p.1.

[75] P.L.Wizinowich Et.Al.,
W.M.Keck Observatory Adaptive Optics Program, 1994,
in Adaptive Optics in Astronomy,
Proceedings of the S.P.I.E. Symposium of Telescopes for
the Twenty First Century. Volume 2201, p.22.,
Kona Hawaii, 1994.

[76] C.G.Wynne
Shorter than a Schmidt,
Mon. Not. R. Astron. Soc., 1977.180, p485-490.

[77] C.G.Wynne
Chromatic Correction of Wide Aperture Catadioptric Systems,
Nature., 1947, no.4055,p.91.

[78] C.G.Wynne
Lens Design by Electronic Digital Computer,
Proc. Phys.Soc.,1959, LXX11, 77.

[79] T.Young

Phil. Trans. Roy.Soc., 1802, II,12.

[80] Zemax

Optical Design Program,

Users Guide, Version 3.0, 1994.

[81] Optica

Specialised Optical Design Program,

Notebook for Mathematica Version 2.2,

Wolfram Research, U.S., 1994.

Appendix A: Inter-Opt Program Class Definitions

// A surface class.

```
class surface
{

 char t_c[3];                          // surface type
 char material_c[7];                   // media after surface
 double c_c,                           // surface base curvature
 clear_c,                              // surface clear semi-aperture
 a2_c, a4_c, a6_c,e_c,                 // surface ashperic coefficients
 sep_c,                                // surface seperation
 rifront_c, rifront2_c, rifront3_c,    // ri front media
 riback2_c, riback3_c, riback_c,       // ri back media
 dispfront_c, dispback_c ;             // media dispersions

 public :
 surface(char *t, char *material,
        double c = 0,
        double clear = 0 , double a2 = 0,
        double a4 = 0, double a6 = 0,
        double e =0, double sep = 0,
        double rifront = 1,
        double rifront2 = 1, double rifront3 = 1,
        double dispfront = 0,
        double riback = 1,
        double riback2 = 1, double riback3 = 1,
        double dispback = 0)
        { strncpy(t_c,t,4) ;strncpy(material_c,material,8);
               c_c = c ;
               clear_c = clear; a2_c = a2; a4_c = a4;
               a6_c = a6; e_c =e;
               sep_c = sep; rifront_c = rifront;
               rifront2_c = rifront2; rifront3_c = rifront3;
               dispfront_c = dispfront;
               riback_c = riback ;
               riback2_c = riback2; riback3_c = riback3;
               dispback_c = dispback;
               }

        // members to alter and read surface objects

        int newc11(double cc1);
        int newa1(double aa1, double aa2);
```

```
          int newd22(double dd2);
          int newstop(double n);
          int checkstop();
          int checka();
          int check4();
          int checkx();
          int check5();
          int check6();
          int newa4(double aa4);
          int newa2(double aa2);
          int newe(double ee2);
          int newa6(double aa6);
          int newn23f();
          int oldn23f();
          int newn23b();
          int oldn23b();
          // end members to alter and read surface objects

           // friend of classes  pray and rray
           // able to freely access their members
          friend class pray;
          friend class rray;
          // end friend classes pray and rray

  ~surface()
  {
  }
};
```

// end surface class

// paraxial ray class

```
class pray  // paraxial meridional ray objects
{

  double  y,      // ray hieght in y plane
          u,      // ray u angle
          u_dash,
          nu,
          nu_dash ;

  public :
  pray(double y_c = 0, double u_c = 0, double u_dash_c = 0,
  double nu_c = 1,
  double nu_dash_c =1)
  {y = y_c; u = u_c ; u_dash = u_dash_c;
```

```
   nu = nu_c ; nu_dash = nu_dash_c ;}

   // members to trace paraxial rays through surface objects

   int trace(surface this_surf, surface next_surf) ;
   int traceth(surface this_surf, surface next_surf);
   int traceefl(surface this_surf, surface next_surf);
   int trace_stop(surface this_surf, surface next_surf) ;
   int back_trace(surface this_surf, surface next_surf) ;
   int first_ray_coards_back(surface this_surf, surface next_surf);

   // end members to trace paraxial rays through surface objects

   int newpray(double yx,double ux,double u_dashx,
              double nux,double nu_dashx);

  ~pray()
  {
  }
} ;
```

// end paraxial class

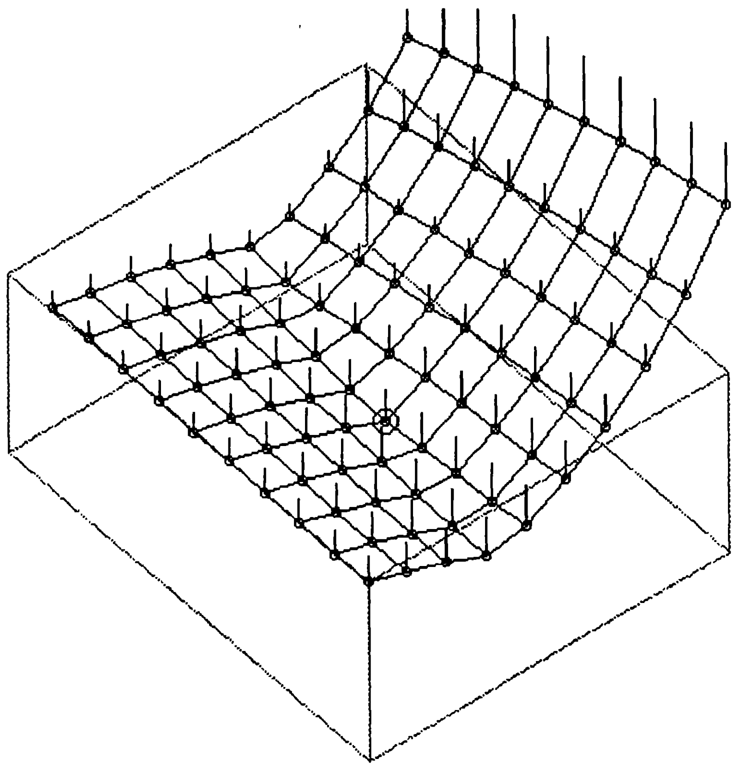
// general ray class

```
class rray
{

  double  x, // surface intersection of ray in x plane
          y, // surface intersection of ray in y plane
          z, // surface intersection of ray in z plane
          X, // X direction cosine of ray
          Y, // Y direction cosine of ray
          Z; // Z direction cosine of ray

  public :
  rray(double x_c = 0, double y_c = 0, double z_c = 0 ,
  double X_c = 1,
  double Y_c =1, double Z_c = 0)
  {x = x_c; y = y_c ; z = z_c;
   X = X_c ; Y = Y_c ; Z = Z_c;}

   // members to trace real rays through surface objects

   int real_trace(surface &this_surf) ;
   int real_back_trace(surface &this_surf) ;
   int newray(double xx, double yy, double zz, double XX,
             double YY, double ZZ);
  // end members to trace real rays through surface objects

 ~rray()
  {
  }
};
```

// end general ray class

Appendix B: Inter-Opt Program Help Manual

Interactive Optical Design Software



User Manual

by

Alan Radley

June 1994

Optical Science Laboratory

University College London.

About Inter-Opt

Inter-Opt is an optical ray tracing program that aids interactive design through real time aberration plots. The ray tracing code is object oriented and was implemented in C++. The program was specifically developed for the design of catadioptric spectrograph cameras for use on 8m class telescopes. Among these camera types are Short Schmidts, Maksutov and Houghton systems. If you have any comments or suggestions please contact...

Alan Radley,

Optical Science Lab,

Astronomy & Physics Dept.,

University College London, WC2.

General Hints

Inter-Opt is run from the DOS command prompt by typing optical. Once running the welcome screen waits for either 'H' to invoke on line help or 'L' to load a lens. After the user has selected a file number the lens is loaded and its contents displayed on the screen, the user then hits any key to do an initial paraxial ray trace for that lens. The paraxial ray trace searches for the limiting aperture stop and uses this data to locate and create an entrance pupil. Also the paraxial image surface is located and set as the default image position. The system now waits for the user to select an option from the lower menu.

Note if the camera has a doublet corrector (i.e. Houghton type) press 'O' after loading in order for any subsequent ray tracing to be correct. Among the options available from the lower menu are .... 'X'- Exit from program, 'P' - single paraxial ray trace, 'R' - single real ray trace, 'K' ray trace speed test, 'O' doublet corrector mode, 'L' - list current lens parameters, '2' - save the current lens note you must use this format

... NAME.LNS, 'F' - interactive parameter maps, 'G' design Sigler doublet corrector, 'L' - load lens, 'T' third order contributions, 'I' interactive screen, 'D' draw lens schematic, and 'S' display spot diagrams, also 'M' memory set for parameter map design, and '8' scale lens to a new focal length.. Note also that when in interactive or spot diagram screens they require a mouse peripheral, and to leave these screens type 'X' once. Inter-Opt has been developed specifically for the design of catadioptric cameras for integral use in spectrographs. The program is not a general purpose ray tracing program and has not been tested for a wide variety purposes. The code is known to give reliable results for camera systems in which the entrance pupil is located at the physical stop, the first physical surface. Development time was not available for the extension of such tests to systems where the stop is located within the system, or for lenses with an object located closer than infinity.

Ray Tracing Format

Inter-Opt follows the co-ordinate system and ray tracing equations from Chapter 10 of Modern Optical Engineering, 2nd Ed. 1990, by Warren J. Smith Published by Mcgraw Hill Inc. Only axially symmetric systems can be ray traced, and all data is in mm. The optic axis is labelled x, with y denoting the tangential plane and z the sagittal. Also the X,Y and Z direction cosines used by Inter-Opt correspond to N, M and L direction cosines as specified by Kidger Optics (the usual English system) which uses z as the optical axis. We show a table for reference :

(Inter-Opt on Left):

X = N (Kidger)	Inter-Opt axis x = z
Y = M	y = y
Z = L	z = x

Lens File

Inter-Opt has a specific file format that must be followed for all lenses to be loaded. Each file must have the extension (.lns). The first line of any lens file takes the following form: Schmidt_Alan3 6 117 1 3 0 0 The first 13 characters specify a title for the lens (up to 15 chars) An integer follows and indicates the total number of surfaces (not including either the entrance pupil or image plane).The second number 117 (real) is the semi-diameter of the physical stop. The third number 1 (integer) is a check digit that specifies the image distance = 1 for infinite object, = 0 for finite. The third number 3 (real) is the semi - field angle in degrees (y plane object). The fourth number 0(real) is the image curvature at the focus and is zero in this case for a plane image surface.

```
Schmidt_ALAN3 6 117 1 3 0 0
0                 120         c           0            Air
1                 0           1           0            1          0
1                 0           1           0            1          0

0.000000e+00      150.00      c           436.8        Air
1                 0           1           0            1          0
1.465566          0.006759    1.455296    0.006759     1.487802   0.006759

-0.00002971135    150.00      a           0.0          1.435e-10  3.591E-17

                                          21.000000    Silica
1.465566          0.006759    1.455296    0.006759     1.487802   0.006759
1                 0           1           0            1          0

-0.000654107797   220.00      cm          1162.000     Air
1.000000          0           1           0            1          0
-1                0           -1          0            -1         0

-0.00338827801    50.00       c           -698         Air
-1.000000         0           -1          0            -1         0
-1.465566         0.006759    -1.455296   0.006759     -1.487802  0.006759

0.00067948712     50.00       c           -49.166      Silica
-1.465566         0.006759    -1.455296   0.006759     -1.487802  0.006759

-1                0           -1          0            -1         0
```

Figure A: Example Schmidt Camera Lens File Format

The fifth number (real) on the first line in the file specifies a defocus from the paraxial focal position. Following this line the data for each surface are given, each surface having three associated lines. The first of these 3 lines looks like this 0.0 150.0 c 436.8 Air .The above line for our imaginary surface begins with a curvature (real)= 0.0 (one over radius in mm). Next a semi-clear aperture = 150.0 (real) is given. Note the physical stop must have the smallest clear diameter, and the user must check that it is small enough and the correct stop surface has been used in the initial paraxial ray trace. Incorrect ray trace results will occur if the program calculates the wrong surface as the stop.

Next on the first line of this surface specification the surface type is specified by c on page 2, (type is up to two characters long) one of these : c = spherical surface, cm = mirror, a = aspheric, ma = mirror aspheric, or me = mirror conic with eccentricity constant. Next is the name of the media prior to the surface . This is up to five characters. The following line gives the optical characteristics of the media prior to the surface. The first number on this line is the refractive index (ri) of the material prior to the surface in yellow light or the mid wavelength used. Next is a value for the dispersion dn ={ri(f)- ri(c)} of this material. The following 4 numbers are ri for blue, dn, and ri for red, dn. Note dn is the same in each case here, and this repeated data is for future compatibility of present lens files, and leaves room for future versions of Inter-Opt that may require partial dispersion values instead.

The third surface line has the same format as the previous line yet specifies the media following the surface. After that line the data for a new surface begins as before. However if a surface is type mirror i.e. cm then all refractive indices after it are reversed in sign as are the signs of all spacings. Another difference that must be noted is that an aspheric surface must be given type a or ma. Also give A2 (=0 if not parabolic) coefficient, followed by A4 and A6. Note never a give value for the surface curvature if specifying a2 (i.e. parabola) which has 1 over a2 = 4 times the focal length. Also Inter-Opt allows conics to be specified in an alternative way. When specifying a conic mirror (me), an eccentricity constant (real) must be given immediatly after the surface type flag. Note the eccentricity constant e is +1 for a parabolic surface, and e > 1 Hyperbola and e < 1 > 0 for a prolate ellipse. Note also the Konic Constant K = - (e*e).

Third Order Contributions

Selecting 'T' will invoke calculation and display of the third order Seidel contributions for each surface. The Seidel coefficients are displayed for each physical surface of the system, along with longitudinal and lateral chromatic coefficients. After surface data the Seidel sum is shown, and below the appropriate sum converted to transverse aberration. If this is the first time the third order aberrations have been calculated for this lens, then the system will ask the user to type in the first surface of the field corrector (whether required or not). This information is used in interactive mode when designing a field flattening lens, and is explained in help under field flattener design.

Interactive Design

Selecting this option from the menu screen brings up the interactive aberration display screen. Ray intercept curves are displayed and lens parameters can be altered whilst real time ray tracing instantly updates these plots. The usual transverse H-TanU curve is displayed in the top left for an on axis point. Just below this is a multi coloured button (M1) that invokes 3 wavelength aberration displays. Below the first plot is the transverse H-tanU curve for the maximum field position -

note buttons (F1 & F2) allow this to be temporally changed. To the right hand side is a plot for the other sagittal plane.

Below the ray intercept plots are a set of data for the lens in question. Radii of curvature are given for surface 3 onwards, along with associated buttons which are light blue and labelled Step_3 for surface 3 etc. Each button has two parts to increase or decrease the radii of that surface. Each Step_? button is increased or decreased by an amount specified by the Blue_Inc button. It takes a nominal value of 10 to begin with and higher values indicate larger steps for the radii of the surface.

Note this screen is for cameras in which the first physical surface = 2 is the aperture stop (Surface 1 is the entrance pupil). Because the stop is not displayed in interactive mode, the first surface shown is surface 3. The aberration plots can be scaled using the light blue scale buttons (Scale1, Scale2) in the upper left hand corner. Above these buttons the current magnification factor for these plots is displayed. In the top left hand corner two light blue defocus buttons (Defocus 1 & 2) allow defocus from the paraxial focal plane. In the top right hand corner the pupil size is shown and this can be altered via blue buttons Pupil_1 & 2 Above the tangential plots the size of each aberration is displayed in a turquoise rectangle in mm.

Also to select longitudinal spherical aberration representation in three colours hit button (Sphero_1). This aberration plot can be scaled via the Purple_Scale button on the right hand side of the display. Also note these plots show rays from left to right if the lens has positive focal length, and is reversed for negative back focal length. When displayed the longitudinal spherical aberration also gives a quantified value for spherochromatism in box to the right of the plots labelled Sp. This is a function of the area between the red and blue curves integrated along the image height - note this does not adjust for axial difference of focus and so only truly represents spherochromatism in the absence of first order colour.

Buttons A4 and A6 can be used to alter aspheric coefficients for an aspheric surface. The appropriate scale factor buttons are Pink_Inc for A4 and Green_Inc for A6. Note button aspheric_no is used to select which surface is aspheric. Note also buttons Change_1 and Change_2 allow the surface labelled 2 initially to be altered to any other surface number, allowing access to the radii of any particular surface. If this surface is equal to 3 then the radii is altered by red_inc instead of blue_inc - this is to allow modelling of spherical collimator and camera simultaneously. One final word of warning, in a program of this nature where a great deal of flexibility has been given to the user - it is important that the user ensure that any changes to the parameters of a system are reasonable.

Among the buttons not yet described is Sep_3 etc., and these are for altering surface separations. These buttons have increment values scaled by button Brown_Inc. However not all separations are altered with these buttons. The Red Sep_3 etc. buttons are for changing separations that tend to be larger i.e. mirror separations, and are altered by correspondingly higher values as specified by Red_Inc. Note that not all of the buttons in the interactive screen have been explained - this is because they are for specific design tasks and are detailed under the appropriate camera instructions from the main help menu.

Lens Schematic

Once selected this option displays a y-x plane schematic of the lens. The lens is drawn with 3 on axis rays traced and 3 off axis rays at maximum field. It is possible to change the scale of this drawing, by using the mouse and clicking on the buttons in the top left hand corner. This is done in real time whilst within this screen. Also the lens can also be translated around the screen by use of the four buttons in the right hand corner. Ray fans can traced and field angles altered, all in real time to aid rapid examination of any design. To exit from this screen type 'X'.

Schmidt Design

Inter-Opt aids the design of Short Schmidt type cmeras. This is achieved using techniques first described by C.G. Wynne in 'Shorter than a Schmidt' in Mon. Not. of the Royal Astronomical Society 1977, 180, p485-490. The designer creates text a file of the required format with spherical mirror of radius required to give about the correct focal length. See example test.lns. Except a plano field corrector is located the correct distance from the aperture stop (grating in a spectroscope). The mirror is located approximately where required, and a plano field flattening lens is placed near the focal plane. In other words all curvatures (except mirror) are initially made equal to zero.

Also one surface of the field corrector is given type a for aspheric. The values for apheric coefficients should be set to small arbitrary values initially. Next the user enters the required optical characteristics of the corrector and field flattening lenses - i.e. ri's and dispersion value. Now run Inter-Opt, change scale to about 125 using Scale_2, and to begin with change the aspheric coefficient values whilst in interactive design screen until on axis spherical aberration is corrected. Now run 'T'- third order aberrations- from main menu, and type in the first surface number of the field flattening lens - i.e. 6 in example alan3.lns.

Now go back into interactive design mode and hit the 'Field_Lens_1' button, which turns yellow. Once again adjust the aspheric coefficients to reduce spherical aberration. The fact that Field_Lens_1 is 'on', means the field flattening lens thickness and rear curvature follow surface one to correct field curvature and axial colour.(see appropriate help topic) The iterative procedure is in two steps : 1) requires that the user alter surface 6 curvature until the off axis tangential HTanU curve (lower left hand plot) is curving upwards, and is symmetrically shaped. 2) Once this is achieved the user alters the separation between the mirror and field corrector - i.e. Sep_Mirror - until the aberration reading on the off axis plot is reduced.

Now repeat these two procedures iteratively until a minimum aberration value is achieved - coma is eliminated .Now it may be necessary to change the aspheric coefficients slightly during this procedure to adjust spherical aberration to a minimum as the parameters change. Once finished you will have a system free of spherical aberration, coma, and Petszal field curvature. However some axial colour remains and must be balanced against spherochromatism. So turn on button Sphero_1, and adjust the base curvature of the aspheric plate until a balance is achieved between for minimum colour blur. The design is complete and should not require further optimisation.

Maksutov Design

The original concept of a Maksutov was reported in the Journal of The Optical Society of America, 1944 volume 35 no5 p270, by D.D. Maksutov. These systems consist of a spherical mirror with a meniscus shell having radii of curvature approximately centred on a stop infront of this corrector. The second surface radii of this shell is chosen to correct axial colour. Two of the original Maksutov systems exist as Inter-Opt files and can be loaded under the names Mak34.lns and Mak35.lns. Both these systems have a curved focal plane. We shall now load Mak34.lns and demonstrate the design of a meniscus shell for such a system. Please note that Inter-Opt does not calculate longitudinal spherochromatism plots correctly for curved focal plane systems.

Load the file Mak34.lns and go into interactive mode. Write down the aberration values for both 3 colour on and off-axis plots, as quantified in the rectangles above these plots. Next hit the meniscus button Men_1, and proceed to click either of Men_2 or Men_3 to alter the front radii of curvature of the meniscus. These buttons Men_1 and Men_2 work in the same way as Step_3 etc., by adding or subtracting a value to the radii of surface 3. However at the same time Men_1 and Men_2 also find the correct radii of curvature of the second surface for zero axial colour. Basically they allow the user to bend the lens until the correct shape is found to correct the spherical aberration of the primary mirror. You can change the size of a step using Blue_Inc.

With a few changes to Men_2 and Men_3, the user should be able to reduce the aberrations of Maksutov's system slightly. Maksutovs camera's are already well optimised, but the same principles can be applied to the design of a Maksutov from scratch. Also it is possible to design a Flat -Field Maksutov with Inter-Opt, look at ALSUPX9D.LNS This type of Maksutov is designed using the same procedure as a Shorter Schmidt by altering the shape of the field flattening lens whilst simultaneously altering the mirror -corrector separation. The only change is that the user alter the shape of the meniscus corrector using Mak_2 and Mak_3 instead of A4 and A6 to correct Spherical aberration.

Houghton Design

Inter-Opt facilitates the design of Houghton type camera's using interactive displays. J.L Houghton first described how to replace the aspheric Schmidt corrector plate with two full aperture all-spherical air spaced lenses. These lenses would have a combined optical power close to zero to correct axial colour. Sigler continued his work based around third order theory, and we implemented this into inter- opt, by selecting 'G' a doublet corrector is designed that corrects spherical aberration and coma for slow (f\4-f\5) systems. Camera's of faster focal ratio require more sophisticated methods as solutions are more difficult to find.

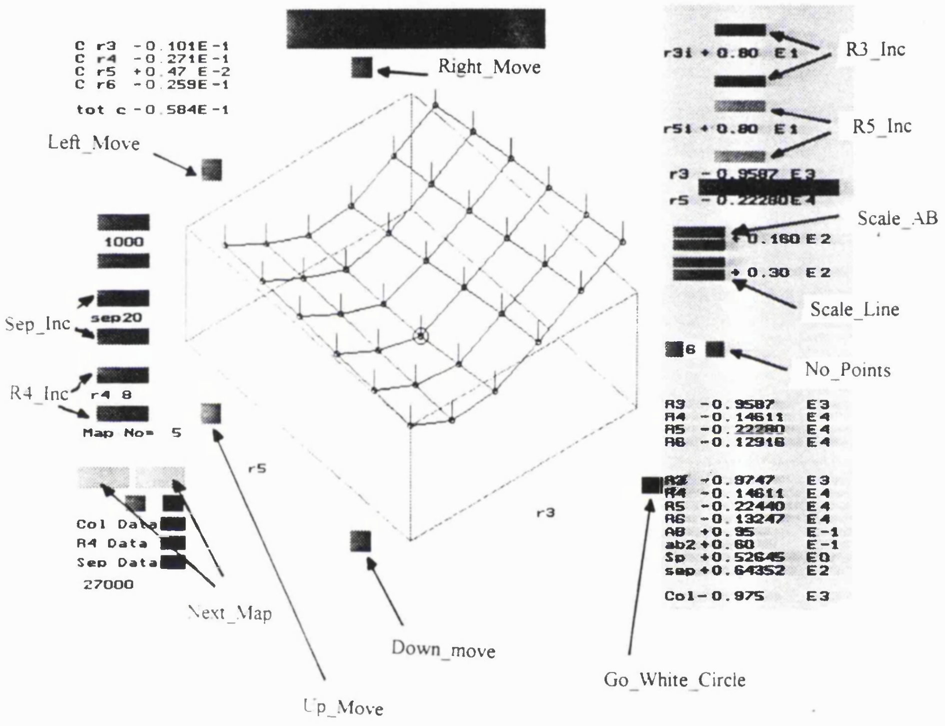
The author has developed a new method of displaying aberrations for the design of very fast, large aperture cameras. The method relies on the coefficients of axial colour as calculated by paraxial ray tracing. By selecting button Doublet_Mode_1, a solve is set on surfaces 4 and 6, in that they automatically assume radii of curvature that cancel the coefficient of first order colour introduced at surfaces 3 and 5. The user can see this by displaying 'T' third order C1 contributions, the four values for the corrector will add up to zero after design. The resulting doublet can be bent by altering 3 and 5, and the solves on surfaces 4 and 6 ensure the two air spaced lenses have a zero first order axial colour.

Button Doublet_Mode_2 allows us to set the program into a doublet corrector mode in which only the last surface takes a radii required for zero colour. This gives the user the freedom to change surfaces 3,4,and 5 at will to find a shape that corrects spherical aberration, coma and axial colour, as well as possibly spherochromatism. The author developed a new interactive method of optimising these types of Houghton cameras. This is displayed on the interactive parameter map screen, and is explained in the next section, the function of this map screen is to design a doublet of the correct shape to eliminate spherical aberration and spherochromatism, whilst having a shape that allows coma to be eliminated by changing the mirror-corrector separation and careful field flattener design.

Interactive Parameter Map Design

This screen aids the design of Houghton systems which have a doublet all-spherical lens corrector (air spaced). Note the screen only works for Houghton systems with a single field flattening lens located close to the image focal plane. This screen is invoked by typing 'F', the user is then sees a parameter map as it is calculated for the camera system loaded. The user is presented with a view of the parameter space of the system in the form of a three dimensional schematic 'data cube'. Each point on the map represents the results of ray tracing 75 rays through a single camera system with a specific shape of corrector and field flattening lens. The user initially sees a set of data for 36 separate camera systems. One horizontal direction on the cube shows the aberrations of cameras with various values of r3, whilst and the other horizontal direction represents camera aberrations with various values of r5. The height of a camera point in the vertical plane represents the size of the on-axis spherical aberration of that camera. Therefore the camera with the lowest level of spherical aberration will be in a 'valley' at the apex of the lowest point of the curve. A line attached to each camera data point represents the spherochromatism of that camera, and the camera with lowest spherochromatism will have the shortest line.

The 3d parameter-aberration map that is shown allows the user to see the results of changing r3 and r5 in a camera system, with r6 automatically taking the required radii to cancel first order axial colour. Note in order for the system to calculate the spherochromatism correctly the system also automatically designs a self achromatic field flattening lens for each camera - making first order axial colour zero for each camera.



```
C r3  -0.101E-1
C r4  -0.271E-1
C r5  +0.47 E-2
C r6  -0.259E-1

tot c -0.584E-1

1000
sep20
r4 8
Map No= 5

Col Data
R4 Data
Sep Data
27000
```

```
r3i +0.80  E1
r5i +0.80  E1
r3 -0.9587 E3
r5 -0.22280E4
+0.180 E2
+0.30  E2
6

R3  -0.9587   E3
R4  -0.14611  E4
R5  -0.22280  E4
R6  -0.12916  E4

R3  -0.9747   E3
R4  -0.14611  E4
R5  -0.22440  E4
R6  -0.13247  E4
AB  +0.95     E-1
ab2 +0.60     E-1
Sp  +0.52645  E0
sep +0.64352  E2

Col-0.975     E3
```

Figure B : Interactive Parameter Map Buttons

The user can select a particular camera with the mouse (a white circle appears around the 'camera data point' in question by clicking near that point. The camera ringed in white has its parameters shown at the lower right hand corner of the screen. By clicking on button go_white_circle the system is sent to the camera ringed in white, and upon leaving this screen that camera will be used for other screens. The user can change the vertical scale by hitting buttons scale_ab, with the current scale factor being shown next to these buttons. Also the length of the spherochromatism lines are scaled with buttons scale_line. The number of data points on each horizontal direction is initially set at 6, but can be altered to either 2 or 10 by clicking button no_points. Additionally The buttons left_move, right_move allow the user to move in parameter space by altering r3. Clicking this button moves the entire map to the left or right, in other words we 'jump' by a large step in r3. (The step is equivalent to half the map size in r3. Buttons Up_move and Down_move are similar but move in r5 instead.

The buttons r3_inc and r5_inc allow the user to alter the size of the step on the radius of curvature between each camera data point. In addition to changing r3 and r5, the system allows the user to visualise the effects of changing r4 also. This requires a lot of ray tracing and is invoked by clicking R4_Data. Thus if no_points is 10 the computer ray traces 100 cameras with various r3 and r5 values for each of 10 r4 values (r4 is stepped by increments of r4_inc) - thus the computer traces 1000 cameras in total - 75 rays through each - total 75000 rays - saving the results of tracing each camera in memory. Depending on the speed of the computer this takes varying amounts of time - (120 seconds on 486 dx33), and once complete a message is displayed. Now by clicking next_map the user is able to alter r4 whilst each map is instantly displayed. Effectively we can use time as a third parameter visualisation method. Upon each click of next map the computer rings the camera with the lowest spherical aberration in red and displays the size of that aberration in red at the lower right hand corner. By altering r4 in this manner interactive design is aided and the user can rapidly visualise where aberration minima lie. Button sep_data is similar to r4_data except it calculates 10 sets of 100 cameras with the air gap changing by value sep_inc.

Field Flattener Design

The Program greatly simplifies the design of a self achromatic field flattening lens that corrects Petzval field curvature without introducing any axial colour into the system. The design of these lenses as an integral part of a Schmidt camera was first described by Wynne in his article on Shorter Schmidts . The author has implemented this work into Inter-Opt, but speeded up the time taken to achieve any design through real time interactive graphical displays. Field flattening lenses are designed using Inter-Opt simply by selecting third order berrations and typing in the first surface number of the lens.

Next go into Interactive design, and turn on button Field_Lens_1, turning it yellow. Now the user can change the radii of curvature of the front element of this lens, whilst the program automatically finds a back surface radii to correct Petsval field curvature by giving the lens the correct optical power. Also the correct lens thickness is calculated simultaneously to ensure the lens does not introduce any axial colour. Note the lens may end up very thick as a result.

The user can verify what has happened by looking at the Third order aberrations and checking that Petsval =0 (S4) and that Longitudinal (C1) chromatic coefficients at the front and back of the lens are now equal and opposite- hence the lens introduces no axial colour. Now going back into interactive design the user can 'bend' the field flattening lens simply by changing the front curvature with solve button Field__Lens_1 turned on. This allows the correct lens shape to be found to balance the aberrations introduced elsewhere in the system. The procedure is remarkably easy to use, and the experienced user rapidly finds the required lens.

Spot Diagrams

This option draws 3 wavelength geometrical spot diagrams for both on axis and user chosen off axis field positions. The user enters the number of rings of 30 rays that are incident on the entrance pupil. Next they are prompted for the field positions in both y and z planes. Finally a defocus off set from the default paraxial image plane can be specified. The through focus Spots are then plotted with an increment of -0.01 of a mm plotted on the left and +0.01mm on the right hand side of the chosen focal position.

Interactive design screen - Basic Mouse Functions

Firstly it is important that the user realise that this screen is mouse operated and cannot be used from the keyboard alone. Should you find yourself mouse-less with the screen running - simply press key 'X' to exit. The mouse has two basic functions within inter-opt - you can select or press 'BUTTONS', by depressing the left hand mouse button. These buttons are specified overleaf. The other function is to press the right hand mouse button and 'drag' the cursor around to draw on the screen. Drawing on the screen can be useful when the user wishes to - remember an aberration size etc.

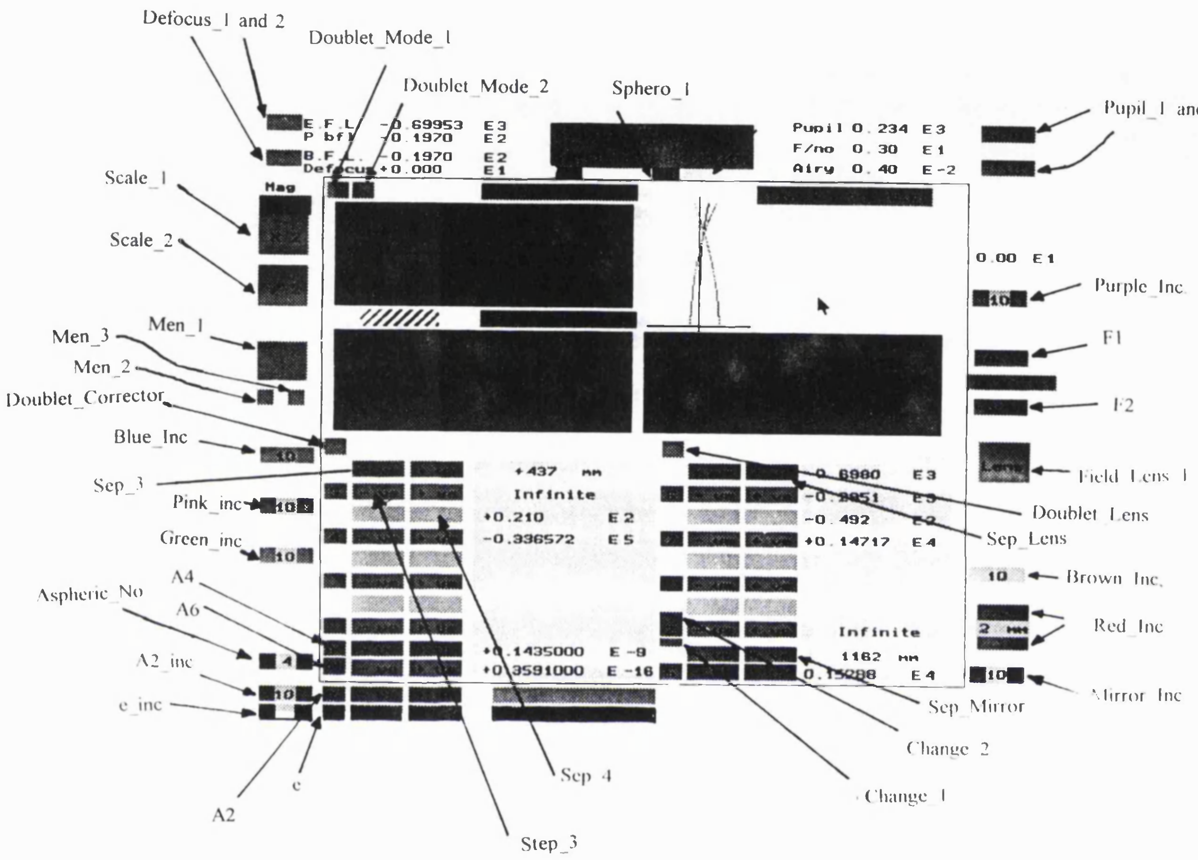


Figure C: Buttons on the Interactive Screen

Button Functions - see Figure C for Positions

1) Step_3 = A button coloured light Blue, with two sections labelled -ve and +ve, this button is used to increase or decrease the radii of curvature of surface 3. The user does so by clicking the appropriate half depending on whether an increase or decrease in radii of curvature is required. Step_4, Step_5 etc. refer to the appropriate surfaces 4,5 etc. The current radii of curvature of this surface is shown to the immediate right of each button. To the left of each Step_? button is a tiny box that tells the user which

surface that button is referring to. Their are a total of 8 Step_? buttons coloured light blue. Also their is an extra one coloured light green which refers to the mirror radii of curvature.

2) Blue_Inc - This button is coloured light blue and specifies the amount by which each blue Step_? button is increased or decreased. The button has three parts, with the middle section displaying the current increment factor. This is set to 10 initially and can be increased or decreased by clicking on either side of the button. Note if you click left the number 10 will decrease, and this means the increment factor is being reduced, likewise when you increase the factor a bigger step in radii is being set.

3) Sep_4 and Brown_Inc, are both coloured light brown and Sep_4 is used to increase or decrease the separation of surface 4 from the previous surface. It is operated in the same manner as Step_3 except Brown_Inc is the button which changes the size of the step.

4) Sep_3 and Sep_Mirror and Sep_Lens are coloured red and used in the same way as Sep_4 except they are for separations that tend to be larger in cameras, and so have Red_Inc that steps by greater amounts (a few mm) which is displayed as 2 mm to begin with. In other words if you click the separation button of the mirror its separation from the corrector will change by 2 mm.

5)Scale_1 and Scale_2 are light blue and for scaling the magnification of the aberration plots i.e. HTanU curves. The size of the current magnification is shown just above these two large square buttons.

6) A2 , A4 and A6 are similar to Sep_? buttons except they increment the aspheric coefficients of the surface indicated on button Aspheric_No which is used to change the number of surface whose aspheric coefficients are displayed. Note each aspheric coefficient has a scale button, Pink_inc , Green_inc and A2_inc, that controls the size of changes made by the A2, A4 and A6 buttons.

7) Mirror_Inc is used to change the step size of the radii of curvature of the primary mirror, and is used in conjunction with Step_Mirror. (Both coloured light green)

8) Doublet_Corrector is a button that is used to set the ray tracing into doublet corrector mode -i.e. if the lens is a Houghton type then this should be clicked before anything else is done.

9) Field_Lens_1 button is light blue and located at the right hand edge of the screen, if it should be clicked it will turn yellow to indicate 'on' for field flattening lens design. Note if you click this button and a 'beep' is heard - then the 'T' option has not been run first to calculate third order aberration coefficients which are required to use this button. Note if you click it a second time this solve is turned off.

10) Doublet_Mode_1 is used to set the program a solve on surfaces 3,4,5,and 6, and this button will only work if the program is in doublet mode to begin with i.e. (doublet corrector has been clicked or 'O' typed). The button causes surfaces 4 and 6 to follow 3 and 5 for zero axial colour. In other words the user is free to alter the radii of curvature of surfaces 3 and 5 but 4 and 6 will automatically take the required curvatures to cause the two lenses to have a combined axial chromatic aberration of zero. Clicking the button a second time turns it off.

11_ Doublet_Mode_2 is similar to above but the user is free to vary 3,4,and 5, whilst surface 6 takes the radii of curvature necessary for the doublet to have no axial chromatic aberration. Clicking the button a second time turns it off.

12) Defocus_1 and 2 are used to move the focal plane either towards or away from the paraxial focal position. The aberration graphs are updated automatically as they are when any other button is clicked by the mouse.

13) Pupil_1 and 2 are used to alter the pupil height and hence entrance ray heights, changing the focal ratio also which is displayed in real time.

14) Sphero_1 is used to turn on/off spherochromatism graphs. Note these do not work for systems with a curved focal plane. Note Purple_Inc is used to scale this graph - which has a scale -0.1mm to +0.1mm at scale 10 on Purple_Inc.

16) Change_1 and Change_2 allow the user to display any lens surface, and alter it by use of the adjacent buttons.

17) If a surface is of type conic and has type me, then the eccentricity constant e is displayed at the bottom of the screen opposite a label e. This can be changed by buttons adjacent to e and a scale button called e_inc in a similer manner to the aspheric coefficients.

Errors and Bugs in Inter-Opt v1.6

A number of bugs remain in the Inter-Opt code. The program was implemented in Borland C++ v4.0. One bug is that the HTanU curves in interactive mode do not always progress all the way across the graphs, this does not indicates vignetting.

Another error found is that for lens or camera systems with curved focal planes, Inter-Opt does not correctly display longitudanol aberration plots. Also when displaying a lens schematic the rays sometimes fail to be meet the image plane - this error is temporary and can be eliminated by hitting 'X' and then 'I' the 'X' and 'D'- the rays should then be correctly drawn. Another error relates to spot diagrams, which sometimes do not display blue and red spots correctly (shown oversized by a large factor). This error cannot be corrected without re-loading the lens in question and immediately displaying spots. Additionally the program exhibits random memory errors which can cause the P.C. to crash or respond unpredictably. The author believes this is not due to problems with the C++ code itself, but is related to the way in which memory allocation has been set up within the Borland development system - thus producing an executable file which has memory compatibility problems. This conclusion was reached as the code works perfectly for several minutes then crashes. It is therefore prudent that the user only load one lens at a time, then exit the program before loading the next lens. Also the P.C. may sometimes require re-booting to cancel memory errors that Inter-Opt can produce.

Notes on Installing and Running Inter-Opt

Inter - Opt requires an IBM PC 386 20 MHz with a co-processor, VGA colour screen and Attached Mouse, as a minimum operating environment. The program is DOS based, and requires the user to copy the following files to the directory from which the user wishes to run Inter-Opt : OPTICAL.EXE, EGAVGA.BGI, all files *.LNS, and help filed *.Z. Once all these files are copied the user can run the program by typing OPTICAL and hitting return.

Appendix C: Inter-Opt program C++ Module Descriptions

C++ File Name	Function Name	Description of Task	Called By
1) Back.cpp	Back_Trace	Backwards Paraxial Ray Trace from one Surface to Pervious One	Par.cpp
2) Check.cpp	Check	Function to check surface separation	Time.cpp
3) Check1.cpp	Check1	Function to check Aspheric Coeffs	Time.cpp
4) Check4.cpp	Check4	Function to check surface curvature	Time.cpp
5) Check5.cpp	Check5	Function to check surface separation	Time.cpp
6) Check6.cpp	Check6	Function to check if surface is Aspheric Type	Time.cpp
7) Checkx.cpp	Checkx	Function to check surface Characteristics	multiple
8) Curve1.cpp	Curve1	Monochromatic Sagittal Off Axis H-TanU Curve	Graph.cpp
9) Curve11.cpp	Curve11	3 Colour Sagittal Off Axis H-TanU curve	Graph2.cp p
10) Curve3.cpp	Curve3	Magnification Finder for left half of H-TanU curves	Graph.cpp Graph2.cp p
11) Curve4.cpp	Curve4	Magnification Finder for right half of H-TanU Curves	Graph.cpp Graph2.cp p
12) Des4.cpp	Des4	Automatic Self Achromatic Field Flattener Design	Time.cpp
13) Des5.cpp	Des5	Automatic Self Achromatic Meniscus Corrector Design	Time.cpp
14) Des8.cpp	Des8	Automatic Houghton Doublet Corrector Design Wynne Method	Time.cpp
15) Des8b.cpp	Des8b	Automatic Houghton Doublet Corrector Design - A.Radley Method	Time.cpp

16) Drawl.cpp	Drawl	Prints Lens Schematic + Traced Rays on VDU Screen - class definitions as follows = surff draw member (calls sketch at a specific scale) , surff sketch member which draws single lens surface on VDU, ray_tra function which draws a single ray through the lens on VDU.	Schem.cpp
17) Drawmp.cpp	Drawmp	Reads from memory and Draws a Single Houghton Parameter Map on VDU	Spaceb.cpp
18) Efl.cpp	Efl1	Routine to create new E.F.L. and B.F.L. for use when system parameters are changing in real time	Time.cpp
19) Ent_p.cpp	Ent_p	Traces paraxial ray back from Stop to first surface, then calculates entrance pupil position	Paraxial.cp p
20) First.cpp	Coards	Creates Entrance Pupil surface object	Par.cpp
21) Global.cpp	Global	Global Variables Defined	mulltiple
22) Graph.cpp	Graph	Plots Monochromatic HTanU curves	Time.cpp
23) Graph2.cpp	Graph2	Plots 3 colour HTanU curves	Time.cpp
24) Graph4.cpp	Graph4	Plots On axis Third and Higher order contribution HTanU curves	Time.cpp
25) Graph5.cpp	Graph5	Plots Off Axis Third and Higher contribution HTanU curves	Time.cpp
26) Help.cpp	Help	Invokes On line Help System	Opt4.cpp
27) List.cpp	List	Lists Lens Files On disc	Load.cpp
28) Listlens.cpp	Listlens	Prints Lens Parameters to VDU	Opt4.cpp
29) Load.cpp	Load	Loads a single Lens from Disc	Opt4.cpp

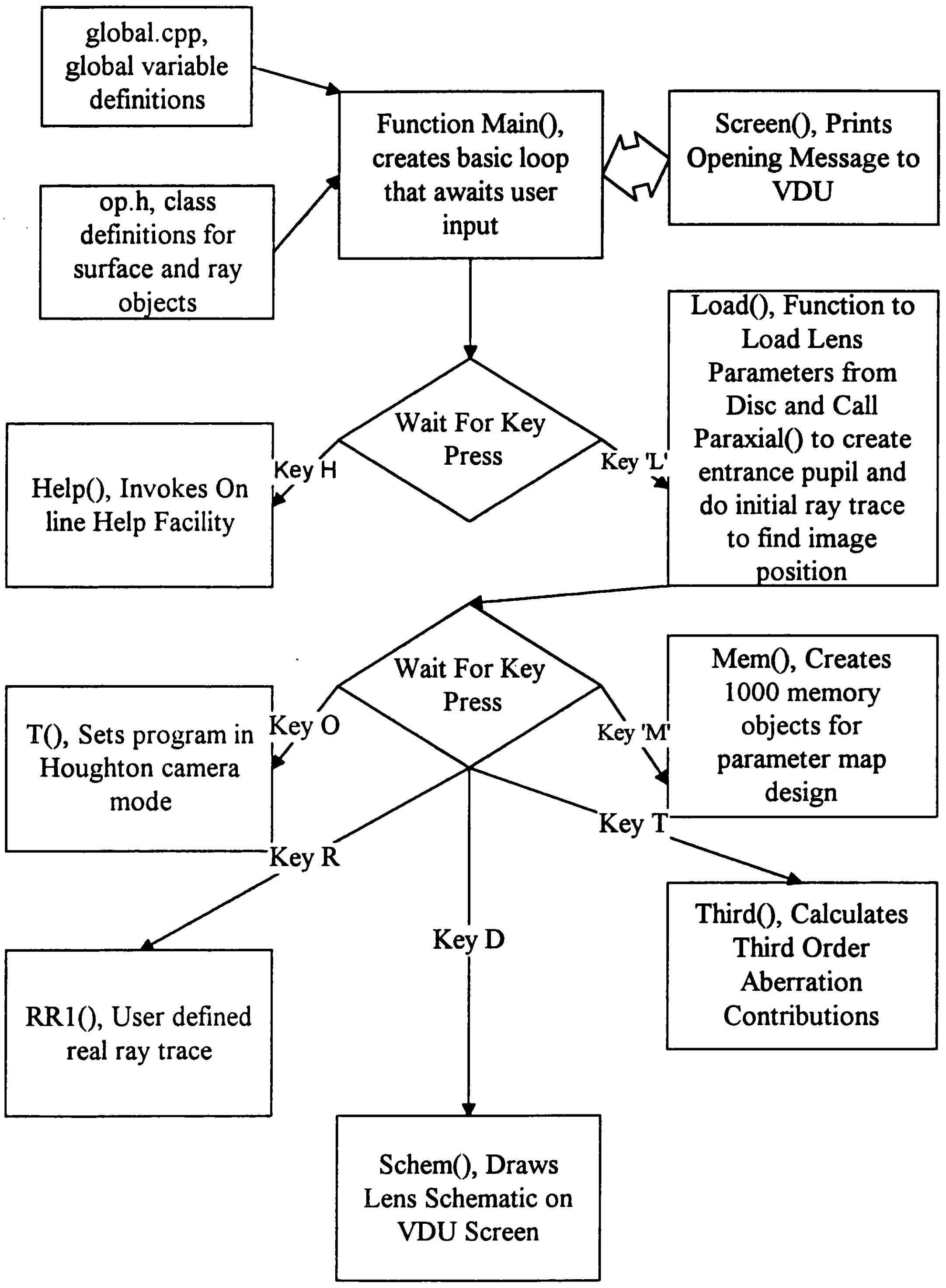
30) Mapb.cpp	Mapb	Calculates Single Data Map for Houghton Camera	Spaceb.cpp
31) Mapc.cpp	Mapc	Calculates Multiple Stack of Maps for Houghton Camera	Spaceb.cpp
32) Mem.cpp	Mem	Sets PC's memory for Houghton Parameter Map design	Opt4.cpp
33) Newa1.cpp	Newa1	Changes surface A coeff	Time.cpp
34) Newa2.cpp	Newa2	Changes surface A2 coeff	Time.cpp
35) Newa4.cpp	Newa4	Changes surface A4 coeff	Time.cpp
36) Newa6.cpp	Newa6	Changes surface A6 coeff	Time.cpp
37) Newc11.cpp	Newc11	Changes surface curvature	Time.cpp
38) Newcurv.cpp	Newcurv	Changes image surface curvature	Time.cpp
39) Newd22.cpp	Newd22	Changes surface separation	Time.cpp
40) Newdata.cpp	Newdata	Creates new data map memory object	Mapc.cpp
41) Newe.cpp	Newe	Changes surface conic e	Time.cpp
42) Newimage.cpp	Newimage	Creates New Image surface	Time.cpp
43) Newn23.cpp	Newn23	Function to make r2 and r3 same as r1, front ri's only	Time.cpp
44) Newn23b.cpp	Newn23b	Function to make r2 and r3 same as r1, back ri's only	Time.cpp
45) Newpoint.cpp	Newpoint	Creates new data map memory object	Time.cpp
46) Newpray.cpp	Newpray	Function to change paraxial ray input parameters.	multiple
47) Newray.cpp	Newray	Function to change real ray input parameters	multiple
48) Newstop.cpp	Newstop	Changes surface separation	Time.cpp
49) Notgrap2.cpp	Notgrap2	Calculates on axis aberration curve size	Mapb.cpp Mapc.cpp
50) Notgraph.cpp	Notgraph	Calculates off axis aberration curve size	Mapb.cpp Mapc.cpp
51) Oldn23.cpp	Oldn23	Function to make r2 and r3 as before, front ri's only	
52) Oldn23b.cpp	Oldn23b	Function to make r2 and r3 as before, back ri's only	

53) Open.cpp	Open	Prints opening message to VDU	Opt4.cpp
54) Opt4.cpp	Opt4	Main program Loop	None
55) Page.cpp	Page	Prints single help page to VDU	Help.cpp
56) Par.cpp	Raytrace	Calls routines to create entrance pupil (First.cpp) and traces paraxial ray to find E.F.L.and B.F.L.	Paraxial.cp p
57) Par2.cpp	Par2	Paraxial Ray trace function -general routine	multiple
58) Par3.cpp	Par3	Paraxial Ray trace function - user defined routine	Opt4.cpp
59) Paraxial.cpp	Paraxial	Calls routines to find stop surface position (Stop.cpp), and routine to backtrace from stop to first surface (Ent_p.cpp) to find entrance pupil position and then calls (Par.cpp) to create entrance pupil surface object and find efl and Bfl	Load.cpp
60) Plot1.cpp	Plot1	Function to plot Sagittal and Tangential Field Curvature	Time.cpp
61) Plot2.cpp	Plot2	Function to Plot Longitudinal Spherochromatism a	Time.cpp
62) Readb.cpp	Readb	Checks for button hits used in spaceb.cpp	Spaceb.cpp
63) Readp.cpp	Readp	Reads a data memory object	Mapb.cpp
64) Readpoin.cpp	Readpoin	Reads a data memory object	Mapb.cpp
65) Real.cpp	Real	Routine to Plot 3 Colour Spot Diagrams on VDU, calls Spot.cpp and Spot1.cpp	Opt4.cpp

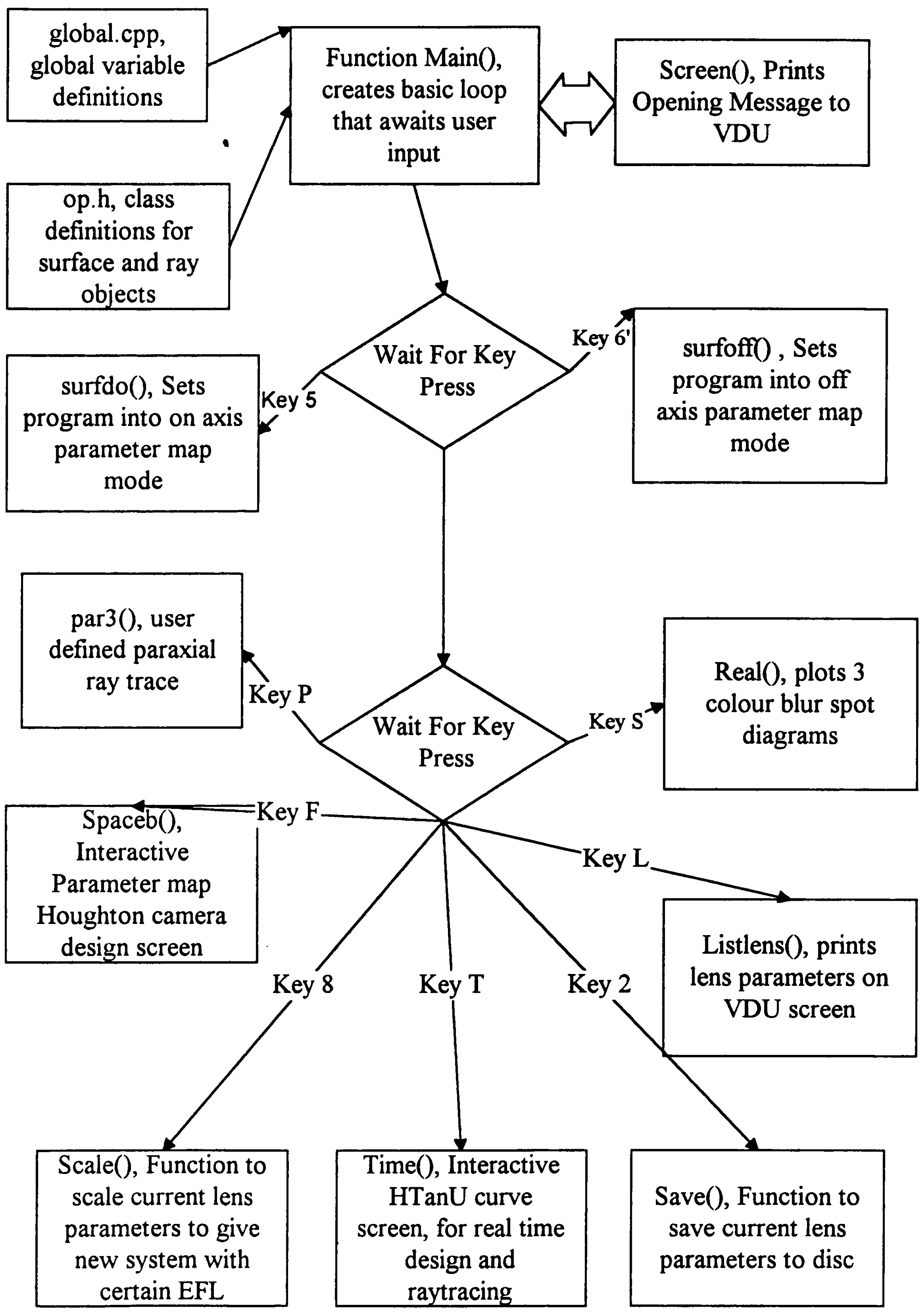
66) Real_bac.cpp	Real_bac	Function to trace a real ray back from the entrance pupil to the first physical surface	multiple
67) Real_tra.cpp	Real_tra	Function to trace a real ray from one surface to next. General routine for spherical, aspheric and conic surfaces types.	multiple
68) Real2.cpp	Real2	Function to trace a real ray through a lens- only for chief ray to find spot position.	multiple
69) RR1.cpp	RR1	User defined real ray trace	Opt4.cpp
70) Save.cpp	Save	Function to save lens parameters to disc.	Opt4.cpp
71) Scale.cpp	Scale	Routine to Scale entire lens to a required E.F.L.	Opt4.cpp
72) Schem.cpp	Schem	Prints Lens Schematic	Opt4.cpp
73) Screen.cpp	Screen	Clears VDU screen and prints program title	multiple
74) Sig1.cpp	Sig1	Designs Slow f/4-f/5 Houghton Corrector	Opt4.cpp
75) Spaceb.cpp	Spaceb	Routine For Parameter Map Design, calls Mapb.cpp, Mapc.cpp, Drawmp.cpp , Readp.cpp, Readb.cpp	Opt4.cpp
76) Spher.cpp	Spher	Function to quantify spherochromatism - integrates area between red and blue curves and gives a nominal value	Mapb.cpp Mapc.cpp
77) Spot.cpp	Realspot2	Off axis Spot Diagram Plotting routine for VDU	Real.cpp
78) Spot1.cpp	Realspot3	On axis Spot Diagram Plotting routine for VDU	Real.cpp
79) Stop.cpp	Stop	Routine to determine which surface is physical stop	Paraxial.cp p
80) Surf.cpp	Surf	Sets Program in Off Axis Map Design Mode	Opt4.cpp

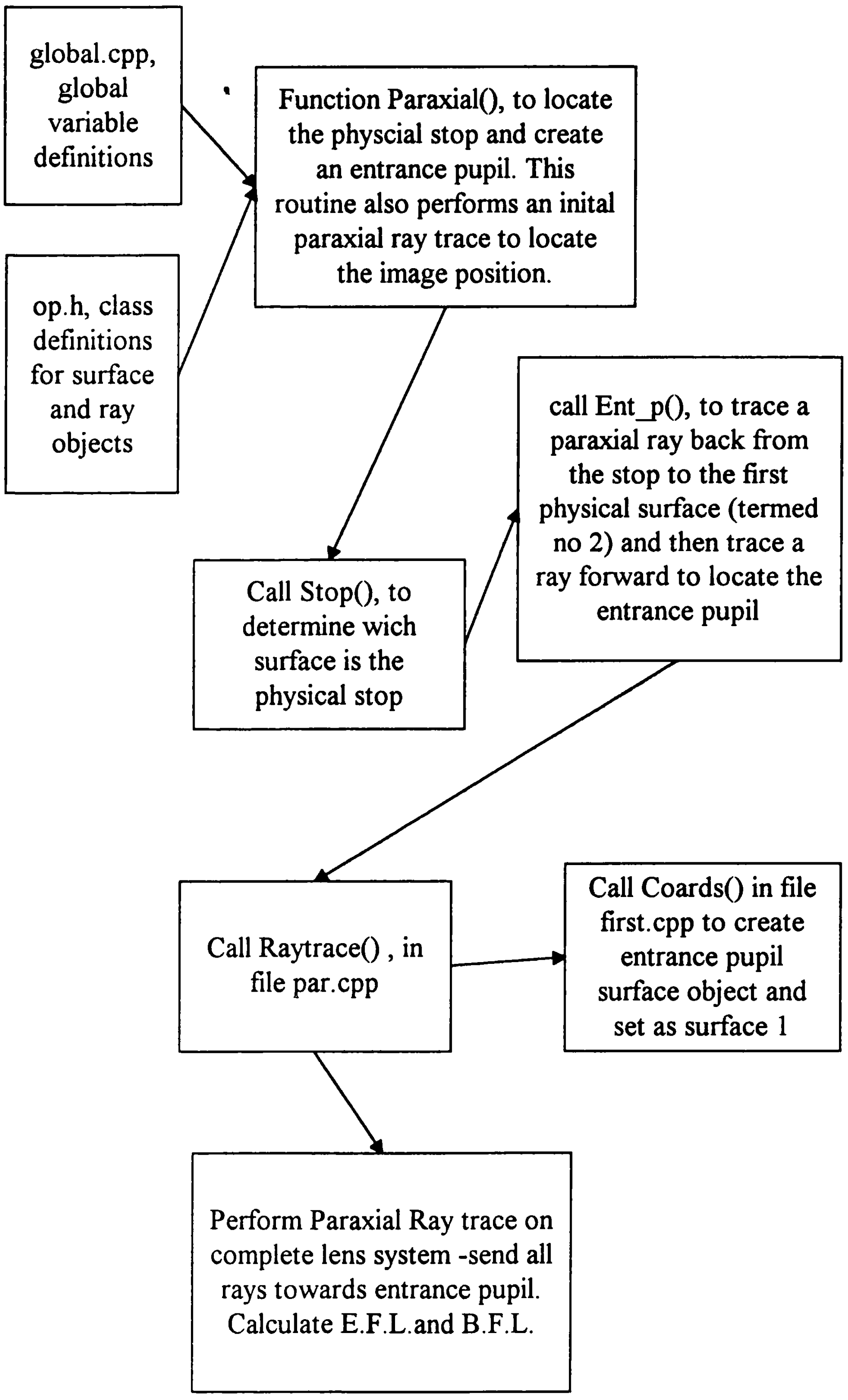
81) Surf2.cpp	Surf2	Sets Program in On Axis Map Design Mode	Opt4.cpp
82) T.cpp	T	Houghton Camera Design Mode	Opt4.cpp
83) Test1.cpp	Test	Test Ray Trace Speed	Opt4.cpp
84) Third.cpp	Third	Calculates Third order Seidel Aberration Coeffs, calls backtrace.cpp and traceth.cpp	Opt4.cpp
85) Third2.cpp	Third2	Third order calculation routine.	multiple
86) Third3.cpp	Third3	Third order calculation routine	multiple
87) Time.cpp	Time	Interactive HTan-U Curve Design Screen Routine	Opt4.cpp
88) Trace.cpp	Trace	Traces paraxial Ray through a single surface	multiple
89) Trace1.cpp	Trace_Stop	Traces paraxial ray through a single surface, used only by stop.cpp	Stop.cpp
90) Traceefl.cpp	Tracefl	Traces paraxial ray through entire system to find E.F.L	multiple
91) Traceth.cpp	Traceth	Traces paraxial ray through single surface, for use with Third order routines.	Third.cpp Third2.cpp Third3.cpp
91) Vno.cpp	Vno	Coverts Vno to dn	Opt4.cpp
92) Draw.h	-	Class Definitions for Shape Class.	multiple

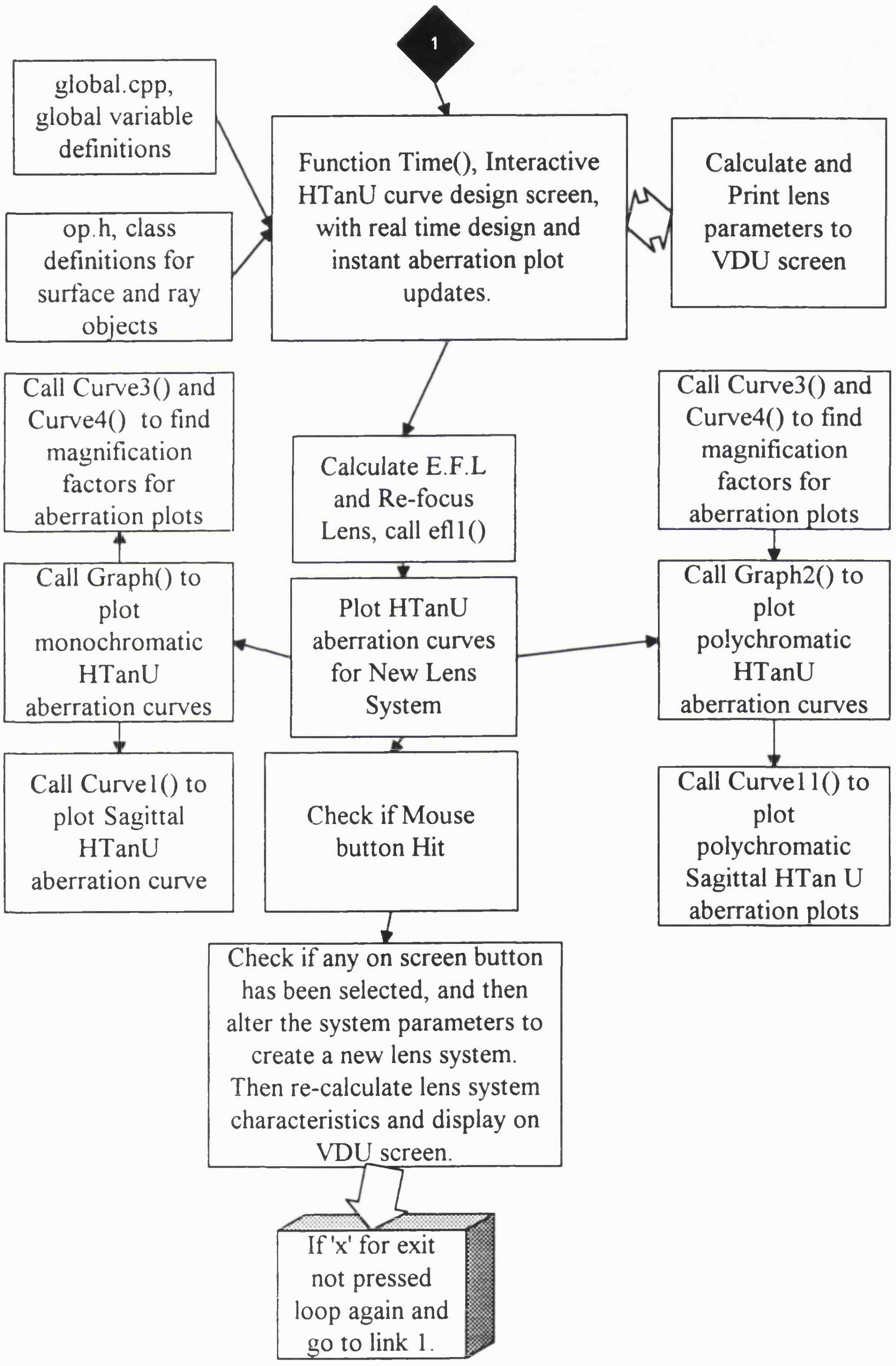
Appendix D: Inter-Opt Program Data Flow Diagrams

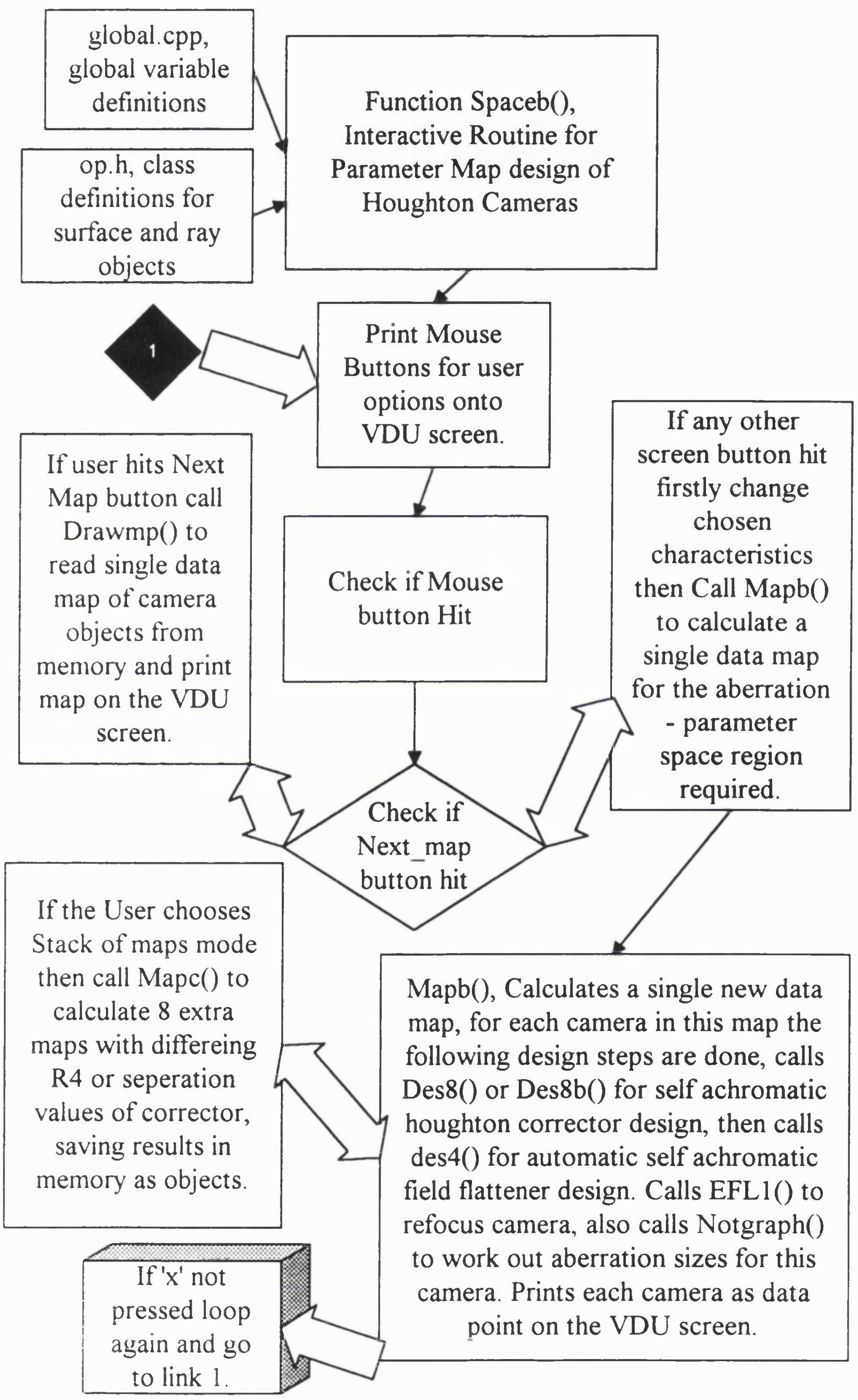


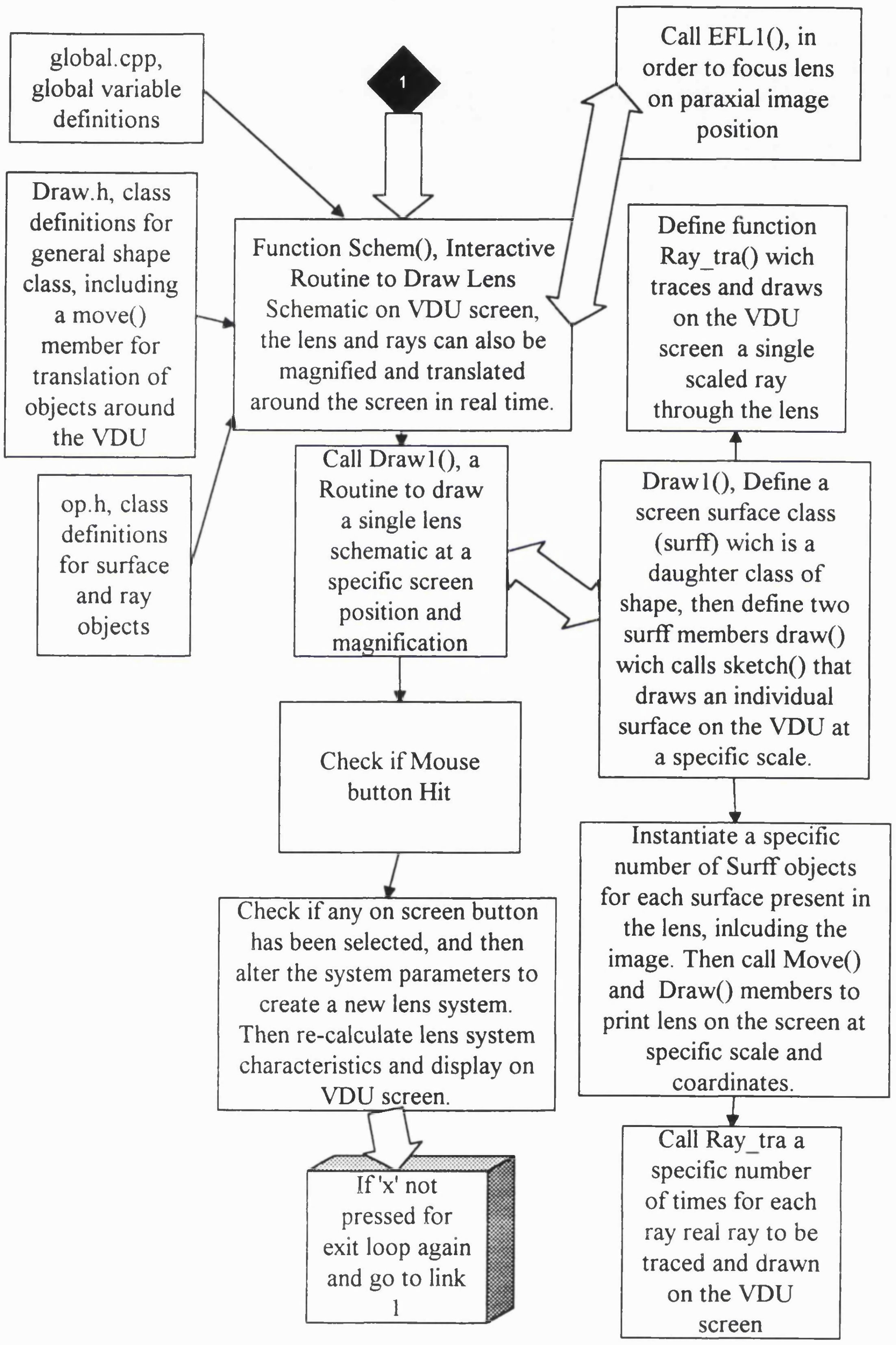
global.cpp, global variable definitions
op.h, class definitions for surface and ray objects
Function Main(), creates basic loop that awaits user input
Screen(), Prints Opening Message to VDU
Wait For Key Press
Key H
Help(), Invokes On line Help Facility
Key 'L'
Load(), Function to Load Lens Parameters from Disc and Call Paraxial() to create entrance pupil and do initial ray trace to find image position
Wait For Key Press
Key O
T(), Sets program in Houghton camera mode
Key 'M'
Mem(), Creates 1000 memory objects for parameter map design
Key R
RR1(), User defined real ray trace
Key D
Schem(), Draws Lens Schematic on VDU Screen
Key T
Third(), Calculates Third Order Aberration Contributions


global.cpp, global variable definitions
op.h, class definitions for surface and ray objects
Function Main(), creates basic loop that awaits user input
Screen(), Prints Opening Message to VDU
Wait For Key Press
Key 5
surfdo(), Sets program into on axis parameter map mode
Key 6'
surfoff() , Sets program into off axis parameter map mode
Wait For Key Press
Key P
par3(), user defined paraxial ray trace
Key S
Real(), plots 3 colour blur spot diagrams
Key F
Spaceb(), Interactive Parameter map Houghton camera design screen
Key L
Listlens(), prints lens parameters on VDU screen
Key 8
Key T
Key 2
Scale(), Function to scale current lens parameters to give new system with certain EFL
Time(), Interactive HTanU curve screen, for real time design and raytracing
Save(), Function to save current lens parameters to disc


global.cpp, global variable definitions
op.h, class definitions for surface and ray objects
Function Paraxial(), to locate the physcial stop and create an entrance pupil. This routine also performs an inital paraxial ray trace to locate the image position.
Call Stop(), to determine wich surface is the physical stop
call Ent_p(), to trace a paraxial ray back from the stop to the first physical surface (termed no 2) and then trace a ray forward to locate the entrance pupil
Call Raytrace() , in file par.cpp
Call Coards() in file first.cpp to create entrance pupil surface object and set as surface 1
Perform Paraxial Ray trace on complete lens system -send all rays towards entrance pupil. Calculate E.F.L.and B.F.L.


1
global.cpp, global variable definitions
op.h, class definitions for surface and ray objects
Function Time(), Interactive HTanU curve design screen, with real time design and instant aberration plot updates.
Calculate and Print lens parameters to VDU screen
Call Curve3() and Curve4() to find magnification factors for aberration plots
Calculate E.F.L and Re-focus Lens, call efl1()
Call Curve3() and Curve4() to find magnification factors for aberration plots
Call Graph() to plot monochromatic HTanU aberration curves
Plot HTanU aberration curves for New Lens System
Call Graph2() to plot polychromatic HTanU aberration curves
Call Curve1() to plot Sagittal HTanU aberration curve
Check if Mouse button Hit
Call Curve11() to plot polychromatic Sagittal HTan U aberration plots
Check if any on screen button has been selected, and then alter the system parameters to create a new lens system. Then re-calculate lens system characteristics and display on VDU screen.
If 'x' for exit not pressed loop again and go to link 1.


global.cpp, global variable definitions
op.h, class definitions for surface and ray objects
Function Spaceb(), Interactive Routine for Parameter Map design of Houghton Cameras
1
Print Mouse Buttons for user options onto VDU screen.
If user hits Next Map button call Drawmp() to read single data map of camera objects from memory and print map on the VDU screen.
Check if Mouse button Hit
If any other screen button hit firstly change chosen characteristics then Call Mapb() to calculate a single data map for the aberration - parameter space region required.
Check if Next_map button hit
If the User chooses Stack of maps mode then call Mapc() to calculate 8 extra maps with differeing R4 or seperation values of corrector, saving results in memory as objects.
Mapb(), Calculates a single new data map, for each camera in this map the following design steps are done, calls Des8() or Des8b() for self achromatic houghton corrector design, then calls des4() for automatic self achromatic field flattener design. Calls EFL1() to refocus camera, also calls Notgraph() to work out aberration sizes for this camera. Prints each camera as data point on the VDU screen.
If 'x' not pressed loop again and go to link 1.


global.cpp, global variable definitions
1
Call EFL1(), in order to focus lens on paraxial image position
Draw.h, class definitions for general shape class, including a move() member for translation of objects around the VDU
Function Schem(), Interactive Routine to Draw Lens Schematic on VDU screen, the lens and rays can also be magnified and translated around the screen in real time.
Define function Ray_tra() wich traces and draws on the VDU screen a single scaled ray through the lens
op.h, class definitions for surface and ray objects
Call Draw1(), a Routine to draw a single lens schematic at a specific screen position and magnification
Draw1(), Define a screen surface class (surff) wich is a daughter class of shape, then define two surff members draw() wich calls sketch() that draws an individual surface on the VDU at a specific scale.
Check if Mouse button Hit
Instantiate a specific number of Surff objects for each surface present in the lens, inlcuding the image. Then call Move() and Draw() members to print lens on the screen at specific scale and coardinates.
Check if any on screen button has been selected, and then alter the system parameters to create a new lens system. Then re-calculate lens system characteristics and display on VDU screen.
If 'x' not pressed for exit loop again and go to link 1
Call Ray_tra a specific number of times for each ray real ray to be traced and drawn on the VDU screen

Appendix E: Equations of Axial Chromatic Coefficients

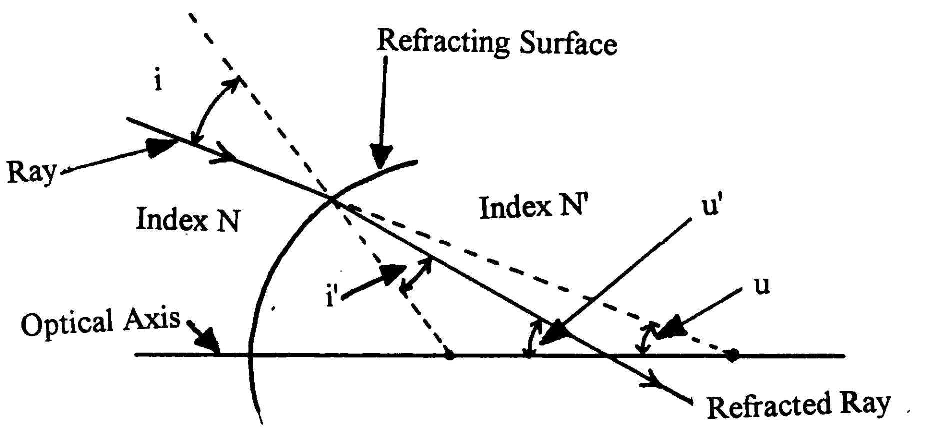


Figure A : Refraction of a Ray at a Spherical Surface

Seidel third order aberration contributions can be calculated from the data of two paraxial rays, an axial ray through the rim of the entrance pupil, and a paraxial principal ray from an off axis point through the centre of the entrance pupil. We shall call these rays respectively ray 1 and ray 2. Using Third order theory and the Seidel equations, it can be shown that the Seidel coefficient for paraxial transverse axial chromatic aberration contribution from any surface in an optical system (TAchC) is equal to :

$$\text{TAchC} = -y * i / N'_k \, u'_k * (\delta N - N/\ N' * \delta N) \text{ .... Equation A}$$

where y = refers to the axial ray (1) height in the meridional plane.

i = refers to the axial ray (1) angle of incidence to the surface

N'_k = index of media of axial ray after passing through the optical system

u'_k = slope pf axial ray after passing through the optical system

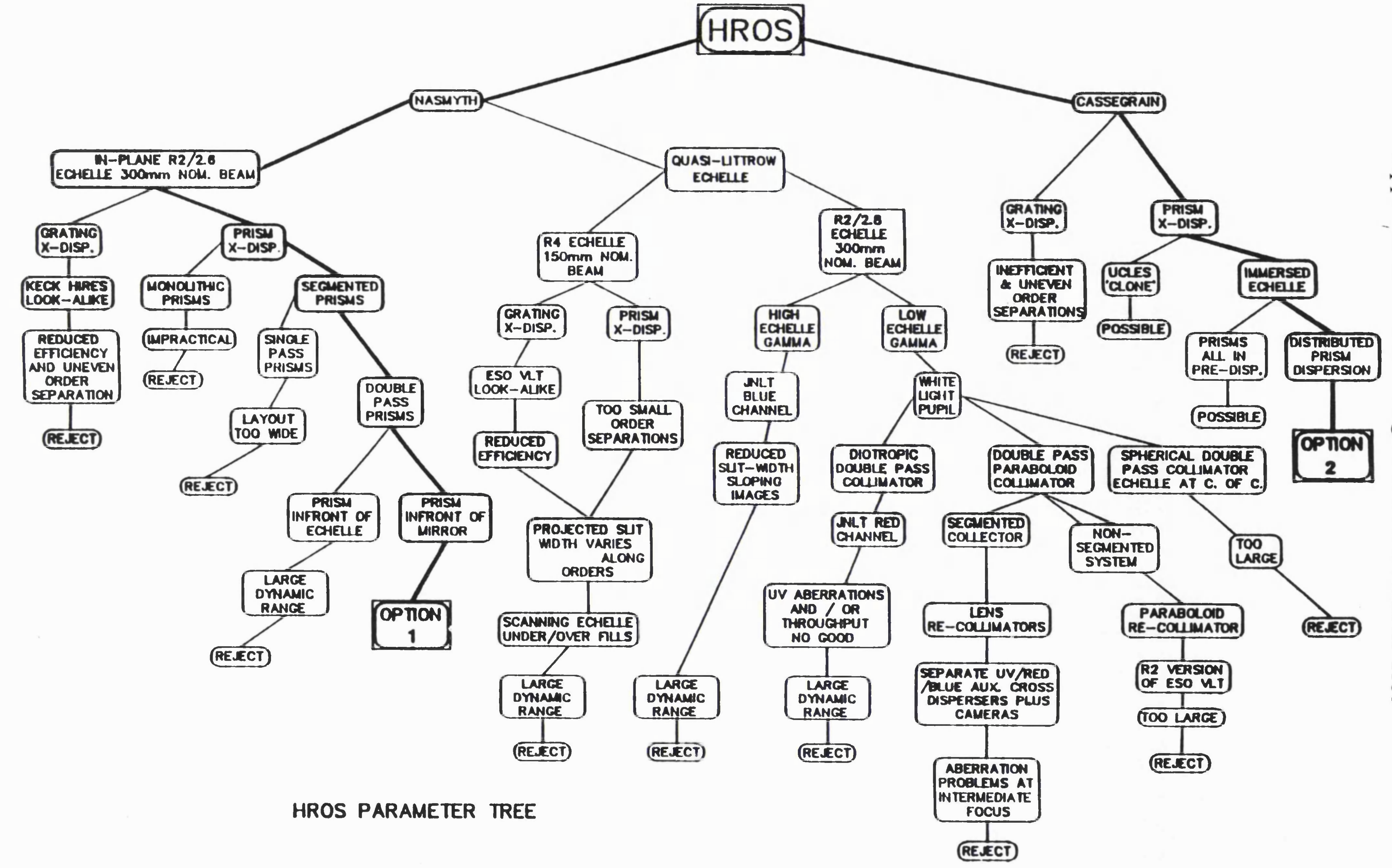
δN = $N_f - N_c$

N_f = Refractive of media index at c wavelength

N_c = Refractive index of media at f wavelength

The Seidel coefficient for axial colour can be obtained by multiplying Equation A by the factor ($-2\, N'_k * u'_k$). The axial colour coefficient for an individual surface then becomes :

$$C = y * i * (\delta N - N/\ N' * \delta N) \text{....... Equation B}$$



HROS PARAMETER TREE

Appendix G: Inter-Opt Program C++ Code Printouts

```
//#include <c:\virtual\optics\ray.h>
#include <float.h>
 #include <io.h>
 #include <stdio.h>
 #include <stdlib.h>
 #include <conio.h>
 #include <iostream.h>
 #include <string.h>
 #include <math.h>


#ifndef __LENS_H
#define __LENS_H

# define  PI 3.14159265359

// Media Class
class  TMedia {
public  :

  TMedia(char *name, double dispersion_c,
         double ri_1_c = 1.45,
         double ri_2_c =1.55, double ri_3_c =1.41,
         double b1_c=0, double b2_c=0, double b3_c=0,
         double c1_c=0, double c2_c=0, double c3_c=0);

         char name[10];
         double ri_1, ri_2, ri_3;
         double dispersion;
         double b1, b2, b3, c1, c2, c3 ;

   int Calculate_ris(double landa_1, double landa_2, double landa_3);

  ~TMedia();

protected :
private :
};
// End Media Class

 TMedia* Glass_cat[1000];

// Surface Class
class  TSurface {

public  :

  TSurface(char *type_c,
           TMedia *media_back_ptr_c,
           double radius_of_curvature_c = 0,
           double semi_clear_aperture_c = 0,
           double separation_c = 0,
           double aspheric_2_c = 0, double aspheric_4_c = 0,
           double aspheric_6_c = 0, double aspheric_8_c = 0,
           double aspheric_10_c =0,
           double eccentricity_c =0);

  TMedia *media_back_ptr;
  // new 31 oct 95

  double  delt_y, delt_z;
  double delt_X, delt_Y, delt_Z;

  int non_axially_sym;

  // new 31 oct 95
  double x11;

  double S1,S2,S3,S4,S5,C1,C2;
  double Petsval_radius;
  char type[9];
  int separation_infinity;
  int radius_infinity ;
  double semi_clear_aperture;
  double separation;
  double aspheric_2;
  double aspheric_4;
  double aspheric_6;
  double aspheric_8;
  double aspheric_10;

  double eccentricity;
  double pupil_position;
  int sqap;
  int conic;
  int mirror;
  double separation_after;
  double y_semi_aperture;
  double x_semi_aperture;
  double radius_of_curvature;


  ~TSurface();

protected :

private :

};
// End Surface Class

// Lens Class
class  TLens {

public :
  TLens(int no_of_surfaces_c);

  TSurface* Surface_ptr[30];
  int no_failed_rays;
  int no_of_glasses;
  int image_surface_number;
  int no_of_surfaces;
  char lens_name[15];
  char lens_type[15];
  int Focus_paraxial;
  double x_field_angle_1;
  double x_field_angle_2;
  double y_field_angle_1;
  double y_field_angle_2;
  double wavelength_1;
  double wavelength_2;
  double wavelength_3;
  int primary_wavelength;
  int pos;
  int posi;
  double back_focal_length;
  double equivalent_focal_length;
  double paraxial_bfl;
  double focal_ratio;
  double entrance_pupil_diameter;
  double entrance_pupil_position;
  int stop;
  int print_results;
  int print_results_third;

  int Set_lens_media_ri();
  int Load_catalogue(char catalogue[15],TMedia* Glass_cat[1000]);
  int Load_from_disc(const char lens_tag[15],TMedia* Glass_cat[1000]);
  int Save_to_disc(const char lens_tag[15]);
  int List(TDC &dc);
  int Third(TParaxial_ray *light_ray_ptr,TDC& dc3);
  int Make_field_lens(int lens_front_surface,TParaxial_ray *light_ray_ptr,TDC& dc3);
  int Make_maksutov_meniscus(int lens_front_surface,TParaxial_ray *light_ray_ptr,TDC
& dc3);
  int Make_axial_colour_zero(int lens_front_surface,TParaxial_ray *light_ray_ptr,TDC
& dc3);
  int Locate_Entrance_Pupil(TParaxial_ray *light_ray_ptr);
  int Focus(TParaxial_ray *this_ray_ptr,TDC& dc3);
  int Test_ray_trace_speed(TLightray *light_ray_ptr, TDC& dc);

  int Paraxial_lens_trace(TParaxial_ray *light_ray_ptr,TDC& dc);

  int Lens_trace(TLightray *light_ray_ptr,TDC& dc);
  int Lens_trace_cod(TRaycod *light_ray_ptr,TDC& dc);

  int Lens_fan_trace_t(TLightray* Light_ray,int colour,double fieldxx,double fieldyy
,TDC &dc);
  int Lens_fan_trace_s(TLightray* Light_ray,int colour,double fieldxx,double fieldyy
,TDC &dc);
  int Lens_fan_trace_h(TLightray* Light_ray,int colour,TDC &dc);
  int Lens_fan_trace_f(TRaycod* Light_ray,int colour,TDC &dc);
  int Lens_fan_trace_f2(TRaycod* Light_ray,int colour,TDC &dc);
  int Lens_spot_trace(TLightray* Light_ray,int colour,double fieldxx,
       double fieldyy,TDC &dc, int  no_dots, int no_rings);

  int Sketch_Lens(double scale,TDC &dc,
                int x, int y, int xs, int ys);



  ~TLens();

protected :

private :
   int trace_paraxial_ray(TParaxial_ray *light_ray_ptr,TDC& dc);
   int Paraxial_trace_single_surface(int surface_number,TParaxial_ray *light_ray_ptr
,TDC& dc);
   int back_trace_paraxial(int surface_number, TParaxial_ray *this_ray_ptr,TDC& dc);
   int trace_paraxial(int surface_number, TParaxial_ray *this_ray_ptr,TDC& dc);

   int Trace_single_surface(int surface_number,TLightray *light_ray_ptr,TDC& dc);
   int Back_trace(int surface_number, TLightray *this_ray_ptr,TDC& dc);
   int Trace(int surface_number, TLightray *this_ray_ptr,TDC& dc);

   int Trace_single_surface_cod(int surface_number,TRaycod *light_ray_ptr,TDC& dc);
   int Back_trace_cod(int surface_number, TRaycod *this_ray_ptr,TDC& dc);
   int Trace_cod(int surface_number, TRaycod *this_ray_ptr,TDC& dc);


};
// End Lens Class

//TLens *TLens_ptr ;

#endif  // __LENS_H
```

```
  // spaceb.cpp
// Function for Paramter Map Design

 #include "c:\bc4\optical\op.h"
 #include <stdio.h>
 #include <graphics.h>
 #include <stdlib.h>
 #include <conio.h>
     //   #include "mouse.h"
 extern void mouse_infor(int *right, int *left, int *x_cd, int *y_cd);
 extern void mouse_show_cursor(void);
 extern int mouse_initialise(void);
 extern void mouse_range(int x1_cd, int y1_cd, int x2_cd, int y2_cd);

 int spaceb()  {
 r4_inc =8;
 double r4_save = r4_inc;
 int do_col=0;
 double col_inc=1000;
 int dont_do_many =0;
 int r4_mode=1;
 c111_inc2=8;
 c333_inc2=8;
  int do_sep=1;
  double no_rays=0;
  int sketch_map=0;
  int sketch_map2=0;
  int test_map=0;
   int draw_map=5;
   int draw_map2=14;
   int many_maps=0;
   coma_3=0; coma_4=0; coma_5=0; coma_6=0;
   int map_run=0;
   double total_c =0;
   int third_show=1;
   int plot_r4=0;
   double ccc1 =0;
   int go_circle =0;
   int mem_xx=-10,mem_yy=-10;
   int mem_point=1;
   int first_point=0;
   int virt=0, virt2=0;
   int plane_bit=0;
   double ccc2  =0;
   char * c444;
   char * new_stop2;
   char bit144[30];
   char bit166[30];
   char bit244[30];
   int bit14=0, bit24=0;
   int no_points =6;
   char a666[25];
   int des88 =0;
   int des88b=1;
   int loop_no=1;
   int axial =1;
   int midx=0,midy=0;
   extern int schem2(double ,double ,double );
    char * s_size;
   int keypress=0,plot_bit=0,left=0,right=0,x=0,y=0;

   int gdriver = DETECT, gmode, errorcode;


 // TRANSFER

 double n_third_d[20],n2_third_d[20],
 y_third_d[20], u_third_d[20], u2_third_d[20],
 c_third_d[20], d_third_d[20], d2_third_d[20], sep_third_d[20];
```

```
int flag_paraxial_y = 1;
int pos = 1;
double mem_ang = field_angl;


int dont_clear =1;

  if (graphics == 1){
if (back_graph == 1) {
setgraphmode(getgraphmode()); }
   }



  if (graphics == 0) {

 setviewport(50, 50, getmaxx()-50, getmaxy()-50, 1);
 int gdriver = DETECT, gmode, errorcode;
 initgraph(&gdriver, &gmode, "");
 errorcode = graphresult();

 if (errorcode != grOk)  /* an error occurred */
 {
    printf("Graphics error: %s\n", grapherrormsg(errorcode));
    printf("Press any key to halt:");
    getch();
    exit(1);             /* return with error code */
 }
 graphics =1;

} // END IF GRAPHICS ALREADY DONE

      // LEFT HAND BLUE STRIPE
  midx = 50;
  midy = 55;
  setfillstyle(SOLID_FILL, 0);
  bar(midx-50, midy-74, midx+100, midy+20);
  // RIGHT HAND BLUE STRIPE

     // button for new map with r4 +sep changed
   setfillstyle(SOLID_FILL, 6);
   midx = 25;
   midy = 370;
   bar(midx-20, midy-8, midx+20,midy+8);

   setfillstyle(SOLID_FILL, 6);
   midx = 70;
   midy = 370;
   bar(midx-20, midy-8, midx+20,midy+8);


     // end button for new map with r4 +sep changed


   setfillstyle(SOLID_FILL, 9);
   midx = 50;
   midy = 390;
   bar(midx-8, midy-6, midx+8,midy+6);

   setfillstyle(SOLID_FILL, 9);
   midx = 80;
   midy = 390;
   bar(midx-8, midy-6, midx+8,midy+6);

   // BUTTONS FOR NUMBER OF POINTS
   setfillstyle(SOLID_FILL, 9);
   midx = 480;
   midy = 270;
   bar(midx-8, midy-6, midx+8,midy+6);
```

```
    setfillstyle(SOLID_FILL, 9);
    midx = 510;
    midy = 270;
    bar(midx-8, midy-6, midx+8,midy+6);

    setfillstyle(SOLID_FILL, 0);
    midx = 495;
    midy = 270;
    bar(midx-8, midy-6, midx+8,midy+6);

    itoa(no_points,a666,10);
    outtextxy(489,267,a666);
    //outtextxy(583,400,"mm");

    // END BUTTONS FOR NUMBER OF POINTS

    // BUTTON FOR GO TO CURRENT CURSER CIRCLE POINT
      //  setfillstyle(SOLID_FILL, 12);
        //   midx = 250;
       // midy = 400;
      //  bar(midx-8, midy-6, midx+8,midy+6);

    // END GO TO CIRCLE POINT

/*      // BUTTONS FOR COL INC
    setfillstyle(SOLID_FILL, 3);
    midx = 40;
    midy = 170;
    bar(midx-20, midy-6, midx+20,midy+6);

    setfillstyle(SOLID_FILL, 3);
    midx = 40;
    midy = 200;
    bar(midx-20, midy-6, midx+20,midy+6);

    setfillstyle(SOLID_FILL, 1);
    midx = 40;
    midy = 185;
    bar(midx-30, midy-6, midx+30,midy+6);

    c444 = fcvt(col_inc,0,&bit14,&bit24);
       // outtextxy(10,182,"Col");
    outtextxy(25,182,c444);


    // END BUTTONS FOR COL INC
  */



    // BUTTONS FOR R4 INC
    setfillstyle(SOLID_FILL, 3);
    midx = 40;
    midy = 290;
    bar(midx-20, midy-6, midx+20,midy+6);

    setfillstyle(SOLID_FILL, 3);
    midx = 40;
    midy = 320;
    bar(midx-20, midy-6, midx+20,midy+6);

    setfillstyle(SOLID_FILL, 1);
    midx = 40;
    midy = 305;
    bar(midx-30, midy-6, midx+30,midy+6);

    c444 = fcvt(r4_inc,0,&bit14,&bit24);
    outtextxy(15,302,"r4");
    outtextxy(40,302,c444);
```

```
   // END BUTTONS FOR R4 INC

   // BUTTONS FOR sep INC
   setfillstyle(SOLID_FILL, 3);
   midx = 40;
   midy = 230;
   bar(midx-20, midy-6, midx+20,midy+6);

   setfillstyle(SOLID_FILL, 3);
   midx = 40;
   midy = 260;
   bar(midx-20, midy-6, midx+20,midy+6);

   setfillstyle(SOLID_FILL, 1);
   midx = 40;
   midy = 245;
   bar(midx-30, midy-6, midx+30,midy+6);

   c444 = fcvt(sep_inc,0,&bit14,&bit24);
   outtextxy(15,242,"sep");
   outtextxy(40,242,c444);


   // END BUTTONS FOR sep INC

   // button to chose iether r4 or collimator for map

      // setfillstyle(SOLID_FILL, 1);
      // midx = 40;
      // midy = 200;
      // bar(midx-30, midy-6, midx+30,midy+6);

   // end button for iether r4 or collimator

   // BUTTON TO TURN ON COLLIMATOR STACK CALCULATION
// setfillstyle(SOLID_FILL, 11);
// midx = 80;
 //   midy = 406;
 //   bar(midx-10, midy-5, midx+10,midy+5);


   // END   TURN ON COLLIMATOR STACK CALC

 //   outtextxy(5,403,"Col Data");
   outtextxy(5,418,"R4 Data");
   outtextxy(5,433,"Sep Data");

   // BUTTON TO TURN ON r4 STACK CALCULATION
   setfillstyle(SOLID_FILL, 11);
   midx = 80;
   midy = 421;
   bar(midx-10, midy-5, midx+10,midy+5);
        //   outtextxy(10,440,"Rays");

   // END GO TO TURN ON STACK CALC

   // BUTTON TO TURN ON sep STACK CALCULATION
   setfillstyle(SOLID_FILL, 11);
   midx = 80;
   midy = 436;
   bar(midx-10, midy-5, midx+10,midy+5);
        //   outtextxy(10,440,"Stack");

   // END GO TO TURN ON STACK CALC

mouse_show_cursor();
 mouse_range(0,0,getmaxx(),getmaxy());

//while ( (keypress != (('X')||('x')) )){
   while ( (keypress != ('X')) && (keypress != ('x')) ){
```

```
//while ( (keypress != 'X') ){
plot_bit =0;
keypress = 'p';
left =0;


while  ( (keypress != 'X')&& (keypress != 'x') &&
   (keypress != 'D')&&
   (keypress != 'Q')&&
   (keypress != 'A')&&
   (keypress != 'W')&&
   (keypress != 'S')&&

   (keypress != 'U')&&
   (left ==0 ))
      {

      if  (kbhit()){
        keypress = getch();}

      if  (loop_no != 1){
       mouse_infor(&right, &left, &x, &y);
     }
     else if  (loop_no == 1)  { left =1;}



     if  (right == 1){ putpixel(x,y,YELLOW);}
    //   left=1;

      }

  if   ((keypress != 'X')&&(keypress != 'x'))  {

  loop_no = (loop_no+1);
  do_sep=0;
  // CHECK IF CLICKED IN AREA OF MAP
  midx =270;
  midy = 370;

    //     midx = 230;
 //     midy = 20;
    //     setfillstyle(SOLID_FILL, 12);
    //     bar(midx-60, midy-15, midx+145, midy+15);

   //  outtextxy(200, 20, "INTERACTIVE DESIGN");

    if   (loop_no >2)  {
  setcolor(0);
  circle(mem_xx,mem_yy,10);
  setcolor(15);
 //  for (virt2 = 1; virt2 <= (no_points); virt2=virt2+1)  {
 //       first_point = 1;
   for  (virt = 1; virt <= (no_points*no_points); virt=virt+1)  {

      if  (draw_map==5){
      (*pointptr[virt]).readp();}
      if  (draw_map==1){
      (*one_pointptr[virt]).readp();}
      if  (draw_map==2){
      (*two_pointptr[virt]).readp();}
      if  (draw_map==3){
      (*three_pointptr[virt]).readp();}
      if  (draw_map==4){
      (*four_pointptr[virt]).readp();}
      if  (draw_map==6){
      (*six_pointptr[virt]).readp();}
      if  (draw_map==7){
      (*seven_pointptr[virt]).readp();}
      if (draw_map==8){
      (*eight_pointptr[virt]).readp();}
      if (draw_map==9){
      (*nine_pointptr[virt]).readp();}

        /*  if (draw_map2==15){
      (*pointptr_b[virt]).readp();}
      if (draw_map2==11){
      (*one_pointptr_b[virt]).readp();}
      if (draw_map2==12){
      (*two_pointptr_b[virt]).readp();}
      if (draw_map2==13){
      (*three_pointptr_b[virt]).readp();}
      if (draw_map2==14){
      (*four_pointptr_b[virt]).readp();}
      if (draw_map2==16){
      (*six_pointptr_b[virt]).readp();}
      if (draw_map2==17){
      (*seven_pointptr_b[virt]).readp();}
      if (draw_map2==18){
      (*eight_pointptr_b[virt]).readp();}
      if (draw_map2==19){
      (*nine_pointptr_b[virt]).readp();}
          */

          if ( ( ((x - xxp) < 10) & ((x-xxp) > -10) )&
           ( ((y - yyp) < 10) & ((y-yyp) > -10) )  )
             { circle(xxp,yyp,10);
             mem_xx = xxp;
             mem_yy = yyp;
             mem_point = virt;
             scale_change2 =1; }

}
  // }


} // loop no >
 // END CHECK IF CLICKED IN AREA OF MAP


 // CHECK FOR AXIAL CORRECTION TWO SURF (DES8) OR LAST SURF (DES8B)

 if (   ( (x >42) & (x <58) ) &( (y>384) & (y<396)  )) {

 if (des88 == 1){des88 =0;

 }
 else if (des88 == 0){des88=1;
 }

 }

 if (   ( (x >72) & (x <88) ) &( (y>384) & (y<396)  )) {

 if (des88b == 1){des88b =0;

 }
 else if (des88b == 0){des88b=1;
 }

 }
     // col inc
/*    if (   ( (x >20) & (x <60) ) &( (y>194) & (y<206)  )) {

   col_inc= col_inc-50;

   }

    if (   ( (x >20) & (x <60) ) &( (y>164) & (y<176)  )) {
```

```
    col_inc= col_inc+50;

    }
   */
//    setfillstyle(SOLID_FILL, 1);
//    midx = 40;
 //   midy = 185;
 //   bar(midx-30, midy-6, midx+30,midy+6);

//    c444 = fcvt(col_inc,0,&bit14,&bit24);
       //    outtextxy(10,262,"col");
//    outtextxy(25,182,c444);

      // col inc

 // R4 INC

    if (    ( (x >20) & (x <60) ) &( (y>314) & (y<326)   )) {

    r4_inc= r4_inc/2;

    }

     if (    ( (x >20) & (x <60) ) &( (y>284) & (y<296)   )) {

    r4_inc= r4_inc*2;

    }

   setfillstyle(SOLID_FILL, 1);
   midx = 40;
   midy = 305;
   bar(midx-30, midy-6, midx+30,midy+6);

   c444 = fcvt(r4_inc,0,&bit14,&bit24);
   outtextxy(15,302,"r4");
   outtextxy(40,302,c444);
   // R4 INC
   // sep inc
    if (    ( (x >20) & (x <60) ) &( (y>254) & (y<266)   )) {

    sep_inc= sep_inc/2;

    }

     if (    ( (x >20) & (x <60) ) &( (y>224) & (y<236)   )) {

    sep_inc= sep_inc*2;


    }

    setfillstyle(SOLID_FILL, 1);
   midx = 40;
   midy = 245;
   bar(midx-30, midy-6, midx+30,midy+6);

   c444 = fcvt(sep_inc,1,&bit14,&bit24);
   outtextxy(15,242,"sep");
   outtextxy(40,242,c444);




      // sep inc


 if (    ( (x >70) & (x <90) ) &( (y>416) & (y<426)   )) {
    if (many_maps == 1){
     setfillstyle(SOLID_FILL, 11);
     midx = 80;
     midy = 421;
     bar(midx-10, midy-5, midx+10,midy+5);
     many_maps=0;

    }
    else if (many_maps == 0){
     setfillstyle(SOLID_FILL, 14);
     midx = 80;
     midy = 421;
     bar(midx-10, midy-5, midx+10,midy+5);
    do_sep=0;
    many_maps=1;
    r4_mode=1;
    //if (r4_inc > 100) { r4_inc =8;}

//  outtextxy(10,450,"Tracing Data Stack");
//  no_rays = no_points*no_points*76*9;
    no_rays =27000;

    c444 = fcvt(no_rays,0,&bit14,&bit24);
  // outtextxy(10,430,"Rays");
     setfillstyle(SOLID_FILL, 1);
     midx = 10;
     midy = 452;
     bar(midx-10, midy-5, midx+55,midy+5);

//  outtextxy(10,450,c444);

    }

    }

    // button for stack for sep
     if (    ( (x >70) & (x <90) ) &( (y>431) & (y<441)   )) {

    if (many_maps == 1){
     setfillstyle(SOLID_FILL, 11);
     midx = 80;
     midy = 436;
     bar(midx-10, midy-5, midx+10,midy+5);
     many_maps=0;

    }
    else if (many_maps == 0){
     setfillstyle(SOLID_FILL, 14);
     midx = 80;
     midy = 436;
     bar(midx-10, midy-5, midx+10,midy+5);
    do_sep=1;
    r4_mode=0;

    if (t2_run ==0){
       r4_mode =1;}

    many_maps=1;
    no_rays =27000;

     setfillstyle(SOLID_FILL, 1);
     midx = 10;
     midy = 452;
     bar(midx-10, midy-5, midx+55,midy+5);


    }

    }
```

```
 // end button for stack for sep

 // button for staCK for collimator
/*   if (   ( (x >70) & (x <90) ) &( (y>401) & (y<411)  )) {


 if (many_maps == 1){
  setfillstyle(SOLID_FILL, 11);
  midx = 80;
  midy = 406;
  bar(midx-10, midy-5, midx+10,midy+5);
  many_maps=0;

 }
 else if (many_maps == 0){
  setfillstyle(SOLID_FILL, 14);
  midx = 80;
  midy = 406;
  bar(midx-10, midy-5, midx+10,midy+5);
 //do_sep=1;
 if (t2_run ==1){
  r4_mode=0;}
 else { cout <<"\a"; dont_do_many=1;}

 many_maps=1;
 do_col=1;
// col_inc=r4_inc*125;
 if (dont_do_many ==1) {many_maps=0;}
 no_rays =27000;

  setfillstyle(SOLID_FILL, 1);
  midx = 10;
  midy = 452;
  bar(midx-10, midy-5, midx+55,midy+5);


 }

 }


 // end button for stack for collimator
*/

 // CHECKING CHANGED MAP FOR NEW R4 INC
 if (   ( (x >5) & (x <45) ) &( (y>364) & (y<376)  )) {
   if (draw_map !=1) {
    draw_map = draw_map -1;}
   else { draw_map =9;}

 sketch_map =1;

 }

  if (   ( (x >50) & (x <90) ) &( (y>364) & (y<376)  )) {
    if (draw_map !=9){
     draw_map = draw_map +1;}
    else {draw_map=1;}

  sketch_map=1;

 }

 // sketch new sep map
 if (   ( (x >5) & (x <45) ) &( (y>379) & (y<391)  )) {
   if (draw_map2 !=11) {
    draw_map2 = draw_map2 -1;}
   else { draw_map2 =17;}
 cout << "\a";
 sketch_map =1;
 sketch_map2=1;

 }

  if (    ( (x >50) & (x <90) ) &( (y>379) & (y<391)   )) {
    if (draw_map2 !=17){
    draw_map2 = draw_map2 +1;}
    else {draw_map2=11;}

  cout << "\a";
  sketch_map=1;
  sketch_map2=1;

 }


 // end sketch second sep map

  setfillstyle(SOLID_FILL, 1);
  midx = 10;
  midy = 340;
  bar(midx-2, midy-10, midx+80,midy+19);


 c444 = fcvt(draw_map,0,&bit14,&bit24);
 outtextxy(80,332,c444);
 outtextxy(10,332,"Map No=");

// c444 = fcvt(draw_map2,0,&bit14,&bit24);
// outtextxy(70,345,c444);
// outtextxy(10,345,"Map_B =");

 if (sketch_map==1){scale_change2 =1;}

 // CHECKING CHANGED MAP FOR NEW R4 INC

 // CHECKING NO_POINTS CHANGE CHANGE

 if (    ( (x >472) & (x <488) ) &( (y>266) & (y<274)   )) {
  if (no_points > 2 ){
   no_points = no_points -4;}
  else { cout << "\a";}

 }

 if (    ( (x >502) & (x <518 )) &( (y>266) & (y<274)   )) {
  if (no_points < 7){
   no_points = no_points +4;}
  else {cout << "\a";}

 }

 if (no_points ==10){
 no_rays =75000;}

 if (no_points ==6){
 no_rays =27000;}

 c444 = fcvt(no_rays,0,&bit14,&bit24);
 outtextxy(10,450,c444);

 // CHECKING NO_POINTS CHANGE

// CHECKING GO CIRCLE HIT

 if (    ( (x >454) & (x <470)) &( (y>371) & (y<383)   )) {
  go_circle =1;
 }
```

```
   else
    { go_circle =0;}
   // END CHECKING GO CIRCLE HIT

   // BUTTON FOR IETHER R3 OR R4 IN ONE DIRECTION
    if (    ( (x >454) & (x <468)) &( (y>444) & (y<456)   )) {
       if (plot_r4 ==0){
      plot_r4 =1;}
       else if (plot_r4 ==1){
      plot_r4 =0;}

    }
    // END R3 OR R4


 if (des88 == 1){

    setfillstyle(SOLID_FILL, 14);
    midx = 50;
    midy = 390;
    bar(midx-8, midy-6, midx+8,midy+6);


   }
   else if (des88 == 0){
    setfillstyle(SOLID_FILL, 9);
    midx = 50;
    midy = 390;
    bar(midx-8, midy-6, midx+8,midy+6);

   }

   if (des88b == 1){

    setfillstyle(SOLID_FILL, 14);
    midx = 80;
    midy = 390;
    bar(midx-8, midy-6, midx+8,midy+6);


   }
   else if (des88b == 0){
    setfillstyle(SOLID_FILL, 9);
    midx = 80;
    midy = 390;
    bar(midx-8, midy-6, midx+8,midy+6);

   }

   if (des88 ==1) { axial=2;}

   if (des88b ==1) {axial=1;}

   // END CHECK


   // check if any mouse buttons hit in area passes
   change =0;
   readb(x,y,300,300);
     //     read(x,y,100,100);
   change =1;

   // check - sends back c111_inc2, c333_inc2,
   // scale_p2, r3_shift2, r5_shift2, scale_change2

   if (sketch_map==1){scale_change2 =1;}
   if (sketch_map2==1){scale_change2 =1;}


   if (scale_change2 ==0){
```

```
  // LEFT HAND BLUE STRIPE
  midx = 50;
  midy = 55;
  setfillstyle(SOLID_FILL, 0);
  bar(midx-50, midy-74, midx+100, midy+60);
  // RIGHT HAND BLUE STRIPE


  // before calling map must do a read for that area of screen !!
  //c111_inc2 etc associated with map always
  midx = 550;
  midy = 55;
  setfillstyle(SOLID_FILL, 1);
  bar(midx-80, midy-74, midx+70, midy+450);
  readb(x,y,300,300);
  // no pointss buttons
   setfillstyle(SOLID_FILL, 9);
   midx = 480;
   midy = 270;
   bar(midx-8, midy-6, midx+8,midy+6);

   setfillstyle(SOLID_FILL, 9);
   midx = 510;
   midy = 270;
   bar(midx-8, midy-6, midx+8,midy+6);

   setfillstyle(SOLID_FILL, 0);
   midx = 495;
   midy = 270;
   bar(midx-8, midy-6, midx+8,midy+6);

   itoa(no_points,a666,10);
   outtextxy(489,267,a666);

  // end no point sbuttons


 // COVERS MAP
  midx = 270;
  midy = 330;
  setfillstyle(SOLID_FILL, 0);
  bar(midx-180, midy-400, midx+180, midy+300);
 // COVER SMAP

  midx = 230;
  midy = 20;
  setfillstyle(SOLID_FILL, 12);
  bar(midx-60, midy-15, midx+145, midy+15);

 outtextxy(200, 20, "INTERACTIVE DESIGN");

   // BUTTON FOR GO TO CURRENT CURSER CIRCLE POINT
   setfillstyle(SOLID_FILL, 14);
   midx = 462;
   midy = 377;
   bar(midx-8, midy-6, midx+8,midy+6);

   // END GO TO CIRCLE POINT
   // BUTTON FOR R4 INSTEAD OF R3
      // setfillstyle(SOLID_FILL, 11);
// midx = 462;
  //  midy = 450;
    //   bar(midx-8, midy-6, midx+8,midy+6);

   // END BUTTON FOR R4 INSTEAD OF R3


    //     midx = 100;
 // midy = 80;
```

```
   //  setfillstyle(SOLID_FILL, 0);
   //  bar(midx-80, midy-74, midx+70, midy+55);
    // read(x,y,100,100);

    // if (map_run ==1){
    //   for (int count = 1; count <= (no_points); count++) {
     //  delete pointptr[count];
     //  }
   //  }
   // title section
      // setfillstyle(SOLID_FILL, 12);
       //   midx = 300;
       //   midy = 30;
       //   bar(midx-90, midy-24, midx+83,midy+10);
       //   outtextxy(230, 30, "INTER-ACTIVE DESIGN");

   // end title section


     if ((many_maps==1)&(no_points==6)){
     outtextxy(140,450,"Ray Tracing 360 Camera's");}

     if ((many_maps==1)&(no_points==10)){
     outtextxy(140,450,"Ray Tracing 1000 Camera's");}

     if (do_col ==1){ r4_save = r4_inc ;r4_inc = col_inc; }

     if (surf ==0) {
     mapb(r4_mode,do_sep,many_maps,c444_inc2,third_show,plot_r4,
     no_points,axial,x,y,c111_inc2,
     c333_inc2,r3_shift2,r5_shift2,270,330,scale_p2,0);
     }
     else if (surf ==1){
     mapb(r4_mode,do_sep,many_maps,c444_inc2,third_show,plot_r4,
     no_points,axial,x,y,c111_inc2,
     c333_inc2,r3_shift2,r5_shift2,270,330,scale_p2,field_angl);
     }

     if (do_col ==1){ r4_inc = r4_save ;}

     if (many_maps ==1){
      setfillstyle(SOLID_FILL, 0);
      midx = 300;
      midy = 440;
      bar(midx-160, midy-20, midx+100,midy+20);
     outtextxy(140,450,"**** Finished ****");}

     map_run=1;
     //mem_ang = field_angl;

    // map(axial,x,y,c111_inc2,c333_inc2,r3_shift2,r5_shift2,100,100,scale_p2,field
_angl);


     }
     else if (scale_change2 ==1){
      scale_change2 =0;

      midx = 550;
     midy = 55;
     setfillstyle(SOLID_FILL, 1);
     bar(midx-80, midy-74, midx+70, midy+450);

   // LEFT HAND BLUE STRIPE
     midx = 50;
     midy = 55;
     setfillstyle(SOLID_FILL, 0);
     bar(midx-50, midy-74, midx+100, midy+60);
     // RIGHT HAND BLUE STRIPE
```

```
 // buttons for no points
   setfillstyle(SOLID_FILL, 9);
   midx = 480;
   midy = 270;
   bar(midx-8, midy-6, midx+8,midy+6);

   setfillstyle(SOLID_FILL, 9);
   midx = 510;
   midy = 270;
   bar(midx-8, midy-6, midx+8,midy+6);

   setfillstyle(SOLID_FILL, 0);
   midx = 495;
   midy = 270;
   bar(midx-8, midy-6, midx+8,midy+6);

   itoa(no_points,a666,10);
   outtextxy(489,267,a666);
 //buttons for numbe rpoints

readb(x,y,300,300);
scale_change2=0;
// midx = 550;
//  midy = 55;
//  setfillstyle(SOLID_FILL, 1);
 // bar(midx-80, midy-74, midx+70, midy+155);

 if (sketch_map==1) {
   //   mapb(many_maps,c444_inc2,third_show,
   //   plot_r4,no_points,axial,x,y,c111_inc2,
//  c333_inc2,r3_shift2,r5_shift2,270,330,scale_p2,0);
    // COVERS MAP
  midx = 270;
  midy = 330;
  setfillstyle(SOLID_FILL, 0);
  bar(midx-180, midy-400, midx+180, midy+300);

  midx = 230;
  midy = 20;
  setfillstyle(SOLID_FILL, 12);
  bar(midx-60, midy-15, midx+145, midy+15);

 outtextxy(200, 20, "INTER-ACTIVE DESIGN");


 // COVER SMAP
   setfillstyle(SOLID_FILL, 1);
   midx = 10;
   midy = 347;
        //   bar(midx-2, midy-4, midx+80,midy+6);

   // (*surfaceptr[4]).check4();
   // if (draw_map ==1){ (*one_pointptr[mem_point]).readp();}
  /*     if (draw_map==1){ (*one_pointptr[1]).readp();}
   if (draw_map==2){ (*two_pointptr[1]).readp();}
   if (draw_map==3){ (*three_pointptr[1]).readp();}
    2    if (draw_map==4){ (*four_pointptr[1]).readp();}

//(*surfaceptr[r3p]).check4();
c111 = r4p;


if (c111 !=0){
c111 = 1/c111;
}
else if (c111==0){
outtextxy(4,347,"Infinite");
plane_bit=1;
}
if (plane_bit!=1){
c444 = fcvt(c111,1,&bit14,&bit24);
outtextxy(30,347,c444);
itoa(bit14,bit144,10);
itoa(bit24,bit244,10);
  if (bit24 !=0){
  outtextxy(4,347,"-");}
  else {   outtextxy(4,347,"+");}
outtextxy(14,347,"0.");
outtextxy(85,347,bit144);
outtextxy(75,347,"E");
}
    //   outtextxy(73,300,"R4");
*/

if (sketch_map2 ==1){
      drawmp(draw_map2,0,1, c444_inc2,1,plot_r4,no_points,
      axial, x,y, c111_inc2, c333_inc2,
      r3_shift2, r5_shift2,
      270, 330, scale_p2, 0);}
else {
     drawmp(draw_map,0,1, c444_inc2,1,plot_r4,no_points,
      axial, x,y, c111_inc2, c333_inc2,
      r3_shift2, r5_shift2,
      270, 330, scale_p2, 0);}

 }

  }

  //clrscr();


//   map(1,1,0,0,370,260,3,0);


// TIME DELAY

//  for (int rinse=1;rinse<1000000;rinse++){}

// TIME DELAY

// PRINT OUT CHOSEN DOUBLET
 // print out curvatures

//(*pointptr[mem_point]).readp();
      if (draw_map==5){
      (*pointptr[mem_point]).readp();}
      if (draw_map==1){
      (*one_pointptr[mem_point]).readp();}
      if (draw_map==2){
      (*two_pointptr[mem_point]).readp();}
      if (draw_map==3){
      (*three_pointptr[mem_point]).readp();}
      if (draw_map==4){
      (*four_pointptr[mem_point]).readp();}
      if (draw_map==6){
      (*six_pointptr[mem_point]).readp();}
      if (draw_map==7){
      (*seven_pointptr[mem_point]).readp();}
      if (draw_map==8){
      (*eight_pointptr[mem_point]).readp();}
      if (draw_map==9){
      (*nine_pointptr[mem_point]).readp();}

      if (draw_map==15){
      (*pointptr_b[mem_point]).readp();}
      if (draw_map==11){
      (*one_pointptr_b[mem_point]).readp();}
      if (draw_map==12){
```

```
      (*two_pointptr_b[mem_point]).readp();}
      if (draw_map==13){
      (*three_pointptr_b[mem_point]).readp();}
      if (draw_map==14){
      (*four_pointptr_b[mem_point]).readp();}
      if (draw_map==16){
      (*six_pointptr_b[mem_point]).readp();}
      if (draw_map==17){
      (*seven_pointptr_b[mem_point]).readp();}
      if (draw_map==18){
      (*eight_pointptr_b[mem_point]).readp();}
      if (draw_map==19){
      (*nine_pointptr_b[mem_point]).readp();}


//(*surfaceptr[r3p]).check4();
ccc2 = r3p;
//chnage
//ccc2 = 1/c111;


if (plane_bit==0){
c444 = fcvt(ccc2,1,&bit14,&bit24);
outtextxy(528,370,c444);
itoa(bit14,bit144,10);
itoa(bit24,bit244,10);
  if (bit24 !=0){
  outtextxy(500,370,"-");}
  else {   outtextxy(500,370,"+");}
outtextxy(510,370,"0.");
outtextxy(595,370,bit144);
outtextxy(585,370,"E");
}
outtextxy(473,370,"R3");
plane_bit =0;

//(*surfaceptr[r4p]).check4();
ccc2 = r4p;
//chnage

if (plane_bit==0){
c444 = fcvt(ccc2,1,&bit14,&bit24);
outtextxy(528,380,c444);
itoa(bit14,bit144,10);
itoa(bit24,bit244,10);
  if (bit24 !=0){
  outtextxy(500,380,"-");}
  else {   outtextxy(500,380,"+");}
outtextxy(510,380,"0.");
outtextxy(595,380,bit144);
outtextxy(585,380,"E");
}
outtextxy(473,380,"R4");
plane_bit =0;



//(*surfaceptr[r5p]).check4();
ccc2 = r5p;
//chnage

if (plane_bit==0){
c444 = fcvt(ccc2,1,&bit14,&bit24);
outtextxy(528,390,c444);
itoa(bit14,bit144,10);
itoa(bit24,bit244,10);
  if (bit24 !=0){
  outtextxy(500,390,"-");}
  else {   outtextxy(500,390,"+");}
outtextxy(510,390,"0.");
outtextxy(595,390,bit144);
outtextxy(585,390,"E");
}
outtextxy(473,390,"R5");
plane_bit =0;

//(*surfaceptr[r6p]).check4();
ccc2 = r6p;

if (plane_bit==0){
c444 = fcvt(ccc2,1,&bit14,&bit24);
outtextxy(528,400,c444);
itoa(bit14,bit144,10);
itoa(bit24,bit244,10);
  if (bit24 !=0){
  outtextxy(500,400,"-");}
  else {   outtextxy(500,400,"+");}
outtextxy(510,400,"0.");
outtextxy(595,400,bit144);
outtextxy(585,400,"E");
}
outtextxy(473,400,"R6");
plane_bit =0;

    //   (*surfaceptr[r6p]).check4();
ccc2 = abp;
//chnage

if (plane_bit==0){
c444 = fcvt(ccc2,3,&bit14,&bit24);
outtextxy(528,410,c444);
itoa(bit14,bit144,10);
itoa(bit24,bit244,10);
  if (bit24 !=0){
  outtextxy(500,410,"-");}
  else {   outtextxy(500,410,"+");}
outtextxy(510,410,"0.");
outtextxy(595,410,bit144);
outtextxy(585,410,"E");
}
outtextxy(473,410,"AB");
plane_bit =0;

ccc2 = ab2p;
//chnage

if (plane_bit==0){
c444 = fcvt(ccc2,3,&bit14,&bit24);
outtextxy(528,420,c444);
itoa(bit14,bit144,10);
itoa(bit24,bit244,10);
  if (bit24 !=0){
  outtextxy(500,420,"-");}
  else {   outtextxy(500,420,"+");}
outtextxy(510,420,"0.");
outtextxy(595,420,bit144);
outtextxy(585,420,"E");
}
outtextxy(473,420,"ab2");
plane_bit =0;

ccc2 = sphero2;
//chnage
   // if (ccc2 != 0){cout << "\a";}

if (plane_bit==0){
c444 = fcvt(ccc2,5,&bit14,&bit24);
outtextxy(528,430,c444);
itoa(bit14,bit144,10);
itoa(bit24,bit244,10);
```

```
    if (bit24 !=0){
    outtextxy(500,430,"-");}
    else {    outtextxy(500,430,"+");}
  outtextxy(510,430,"0.");
  outtextxy(595,430,bit144);
  outtextxy(585,430,"E");
  }
  outtextxy(473,430,"Sp");
  plane_bit =0;


  ccc2 = sepp;
  //chnage

  if (sepp!=0){
  c444 = fcvt(ccc2,3,&bit14,&bit24);
  outtextxy(528,440,c444);
  itoa(bit14,bit144,10);
  itoa(bit24,bit244,10);
    if (bit24 !=0){
    outtextxy(500,440,"-");}
    else {    outtextxy(500,440,"+");}
  outtextxy(510,440,"0.");
  outtextxy(595,440,bit144);
  outtextxy(585,440,"E");
  }
  outtextxy(473,440,"sep");
      //   plane_bit =0;

// print out coma for selected doublet

  ccc2 = coma_3;

  if (plane_bit==0){
  c444 = fcvt(ccc2,4,&bit14,&bit24);
  outtextxy(75,30,c444);
  itoa(bit14,bit144,10);
  itoa(bit24,bit244,10);
    if (bit24 !=0){
    outtextxy(47,30,"-");}
    else {    outtextxy(47,30,"+");}
  outtextxy(58,30,"0.");
  outtextxy(110,30,bit144);
  outtextxy(100,30,"E");
  }
  outtextxy(3,30,"C r3");
  plane_bit =0;

  ccc2 = coma_4;

  if (plane_bit==0){
  c444 = fcvt(ccc2,4,&bit14,&bit24);
  outtextxy(75,40,c444);
  itoa(bit14,bit144,10);
  itoa(bit24,bit244,10);
    if (bit24 !=0){
    outtextxy(47,40,"-");}
    else {    outtextxy(47,40,"+");}
  outtextxy(58,40,"0.");
  outtextxy(110,40,bit144);
  outtextxy(100,40,"E");
  }
  outtextxy(3,40,"C r4");
  plane_bit =0;

  ccc2 = coma_5;

  if (plane_bit==0){
  c444 = fcvt(ccc2,4,&bit14,&bit24);
  outtextxy(75,50,c444);
```

```
   itoa(bit14,bit144,10);
   itoa(bit24,bit244,10);
     if (bit24 !=0){
     outtextxy(47,50,"-");}
     else {    outtextxy(47,50,"+");}
   outtextxy(58,50,"0.");
   outtextxy(110,50,bit144);
   outtextxy(100,50,"E");
   }
   outtextxy(3,50,"C r5");
   plane_bit =0;


   ccc2 = coma_6;

   if (plane_bit==0){
   c444 = fcvt(ccc2,4,&bit14,&bit24);
   outtextxy(75,60,c444);
   itoa(bit14,bit144,10);
   itoa(bit24,bit244,10);
     if (bit24 !=0){
     outtextxy(47,60,"-");}
     else {    outtextxy(47,60,"+");}
   outtextxy(58,60,"0.");
   outtextxy(110,60,bit144);
   outtextxy(100,60,"E");
   }
   outtextxy(3,60,"C r6");
   plane_bit =0;

// end print out coma

   total_c = coma_3 + coma_4 + coma_5 + coma_6;
   ccc2 = total_c;

   if (plane_bit==0){
   c444 = fcvt(ccc2,4,&bit14,&bit24);
   outtextxy(75,80,c444);
   itoa(bit14,bit144,10);
   itoa(bit24,bit244,10);
     if (bit24 !=0){
     outtextxy(47,80,"-");}
     else {    outtextxy(47,80,"+");}
   outtextxy(58,80,"0.");
   outtextxy(110,80,bit144);
   outtextxy(100,80,"E");
   }
   outtextxy(3,80,"tot c");
   plane_bit =0;


 // end print curvatures


 // print out col rad of curv
   if (r_colp !=0) {ccc2 = (r_colp);}
   //chnage
   if (r_colp ==0) {
    outtextxy(528,460,"Infinite");}
   else {

   if (plane_bit==0){
   c444 = fcvt(ccc2,0,&bit14,&bit24);
   outtextxy(528,460,c444);
   itoa(bit14,bit144,10);
   itoa(bit24,bit244,10);
     if (bit24 !=0){
     outtextxy(498,460,"-");}
     else {    outtextxy(498,460,"+");}
   outtextxy(510,460,"0.");
```

```
  outtextxy(595,460,bit144);
  outtextxy(585,460,"E");
  } }
  outtextxy(473,460,"Col");
  plane_bit =0;


// end print out col rad of curv


  // END PRINT OUT CHOSEN DOUBLET

  // go circle
      if(go_circle==1){
   cccl = 1/r3p;
   (*surfaceptr[ss3]).newcl1(cccl);

   cccl = 1/r4p;
   (*surfaceptr[ss4]).newcl1(cccl);

   cccl = 1/r5p;
   (*surfaceptr[ss5]).newcl1(cccl);

   cccl = 1/r6p;
   (*surfaceptr[ss6]).newcl1(cccl);

   if (r4_mode==0){
     if (r_colp != 0){
       cccl = 1/r_colp;}
   (*surfaceptr[3]).newcl1(cccl);}

   //if (do_sep ==1){

     if (t2_run==1){(*surfaceptr[7]).newd22(sepp);}

     if (t2_run==0){(*surfaceptr[5]).newd22(sepp);}

   //}
      }
  // end go circle
    many_maps=0;
   sketch_map=0;
   sketch_map2=0;
    dont_do_many =0;
    do_col=0;
    //if (r4_inc > 100) { r4_inc =8;}

     setfillstyle(SOLID_FILL, 11);
     midx = 80;
     midy = 421;
     bar(midx-10, midy-5, midx+10,midy+5);

     setfillstyle(SOLID_FILL, 11);
     midx = 80;
     midy = 436;
     bar(midx-10, midy-5, midx+10,midy+5);

    // setfillstyle(SOLID_FILL, 11);
    // midx = 80;
    // midy = 406;
    // bar(midx-10, midy-5, midx+10,midy+5);


 } // end keypress if statement

 } // end loop

    field_angl = mem_ang;
    restorecrtmode();
     screen();
```

```
return 0;

} // end space
```

```
  // newpoint.cpp
  // function tp create new point for use in mapb(0
    //  #include <c:\borlandc\optical\global.cpp>
      #include <c:\bc4\optical\op.h>


      int datapoint::newpoint(int xw, int yw ,
      double sepw,double r_colw,double r3w,
      double r4w, double r5w, double r6w, double abw, double ab2w,
      double spherow,double th3w,double th4w,double th5w,
      double th6w)
{

  x = xw;
  y = yw;
  sep = sepw;
  r_col = r_colw;
  r3 = r3w;
  r4 = r4w;
  r5 = r5w;
  r6 = r6w;
  ab = abw;
  ab2 =ab2w;
  sphero_ab = spherow;
  th3 = th3w;
  th4 = th4w;
  th5 = th5w;
  th6 = th6w;
 return 0;
}
```

```
 // newdata.cpp
  // function to create new data point in map
    //  #include <c:\borlandc\optical\global.cpp>
      #include <c:\borlandc\optical\op.h>


      int datapoint::newdata(int xw, int yw , double r3w,
      double r4w, double r5w, double r6w, double abw, double ab2w,
      double spherow,double th3w,double th4w,double th5w,
      double th6w)
{
  x = xw;
  y = yw;
  r3 = r3w;
  r4 = r4w;
  r5 = r5w;
  r6 = r6w;
  ab = abw;
  ab2 =ab2w;
  sphero_ab = spherow;
  th3 = th3w;
  th4 = th4w;
  th5 = th5w;
  th6 = th6w;

 return 0;
}
```

```
  // mapc.cpp
 // test  Map  +++ spherochromatism Printout

  #include "c:\bc4\optical\op.h"
  #include <stdio.h>
  #include <graphics.h>
  #include <stdlib.h>
  #include <conio.h>
      //   #include "mouse.h"
  extern void mouse_infor(int *right, int *left, int *x_cd, int *y_cd);
  extern void mouse_show_cursor(void);
  extern int mouse_initialise(void);
  extern void mouse_range(int x1_cd, int y1_cd, int x2_cd, int y2_cd);
  int mapc(double surface5,int map_no,double shift,int multiple_maps, double c4_in
c,int th,int r2,int steps,
  int axial, int x,int y, double c1_inc, double c3_inc, double r3_s, double r5_s,
  int pos_xx, int pos_yy, int mag_mm, double angle_aa)  {

  // axial is a falg if = 1 then des8b if axial = 2 then des8()
  //coma_3 =0;
  //coma_4 =0;
  //coma_5 =0;
  //coma_6 =0;
  //double shift=0;
  //double inc=0;
    int inc =0;
  int xx=0, yy=0;
  double sphero_ab=0;
  int num=0;
  int first_point=0;
  double r33=0, r44=0, r55=0, r66=0;
  double r_col33=0;
  double ab2=0;
  double xx_now=0, yy_now=0;
  dont_clear =1;
  double sep33=0;
  (*surfaceptr[ss5]).check5();
  sep33 = D111;
   if (t2_run ==0) {
    (*surfaceptr[5]).check5();
    sep33 = D111;}


  double normalize=0,ab_plot=0,ab_plot2=0,
      c1_save=0, c2_save=0, c3_save=0, c4_save=0,
      r1_now=0, r3_now=0, fno=0, ddd50=0,d2_inc=0, d3_inc=0,
      c1_now=0, c3_now=0, bfl_mem=0, c11_inc=0, c22_inc=0,
      mid_x=0,mid_y=0,
      mem_x=0, mem_y=0,
      small_y=0,small_y2=0,
      small_x=0,midx=0,midy=0;

    int slide_y=0, slide_x=0, big_y=0,big_y2=0,
  big_x=0;
  double mem_ang = field_angl;
  double r3_shift =r3_s;
  double r5_shift =r5_s;
  double scale_p = mag_mm;
  int data_x=0, data_y=0, virt2=0, virt =0, lens =0,counter=0,field_des=0,
   des88=0,plot_3 =0,su1 =1, su2=2, su3=3, su4=4;

  // TRANSFER
  char * a444;
  char * a666;
  char * a_asp;
  char * c444;
  int thick=0,bit1=0,bit2=0,bit14=0, bit24=0, bit16=0;
  char bit144[30];
  char bit166[30];
  char bit244[30];
```

```
double  c111_inc =c1_inc;
double  c333_inc=c3_inc;

double  min_ab=10000;
double  min_x=0, min_y=0;
double  min_r1=0, min_r3=0;

    //   double pos_x=pos_xx;
//double pos_y=pos_yy;
    int pos_x=pos_xx;
    int  pos_y=pos_yy;

double mag_ab = mag_mm;
int now=1;
//int array_x[21][21];
    //   int array_y[21][21];
int shift_yy=70;
// TRANSF8ER



// GRAPHICAL PARAMETER SPACE SECTION
min_ab =10000;

  field_angl = angle_aa;



colourray =1;
    //   field_angl = 0;

if (r2 ==0){
(*surfaceptr[ss3]).check4();
  c1_save = c111;
  r1_now = ((1/c111)+r3_shift);
  (*surfaceptr[ss4]).check4();
  c2_save = c111;
  (*surfaceptr[ss5]).check4();
  c3_save = c111;
  r3_now = ((1/c111)+r5_shift);
  (*surfaceptr[ss6]).check4();
  c4_save = c111;}
else if (r2==1){
  (*surfaceptr[ss3]).check4();
  c1_save = c111;
 // r1_now = ((1/c111)+r3_shift);
  (*surfaceptr[ss4]).check4();
  c2_save = c111;
  r1_now = ((1/c111)+r3_shift);
  (*surfaceptr[ss5]).check4();
  c3_save = c111;
  r3_now = ((1/c111)+r5_shift);
  (*surfaceptr[ss6]).check4();
  c4_save = c111;}


mem_x=pos_x;
mem_y=pos_y;
//circle(mem_x, mem_y, 20);
data_x =0;
data_y =0;
slide_x=0;
slide_y=0;

   // christmas section
   int no_steps = steps;

  // double step_a =15;
   int step_a = (235/no_steps);
   // end christmas section



// note go back 5 times 50

r1_now = r1_now - (c111_inc*(no_steps/2));
// end note
r3_now = r3_now - (c333_inc*(no_steps/2));
double  y_box=0;
double  z_box=0,delt=0;

setcolor(12);

for (virt2 = 1; virt2 <= (no_steps); virt2++) {
  r3_now = r3_now + c333_inc;

  c3_now = (1/r3_now);


  (*surfaceptr[ss5]).newcl1(c3_now);
  //data_y = data_y -5;
  inc = virt2-1;

   // slide_x= ((inc*13)*0.866);
   // slide_y= ((inc*13)*0.5);



     slide_x= ((inc*step_a)*0.766);
    slide_y= ((inc*step_a)*0.643);

    // TEST SECTION


    for (virt = 1; virt <= (no_steps); virt++) {

  r1_now = r1_now + c111_inc;
  c1_now = (1/r1_now);

  if (r2==0){
    (*surfaceptr[ss3]).newcl1(c1_now);}
  else if (r2==1){
    (*surfaceptr[ss4]).newcl1(c1_now);}

  if (axial == 2){
  des8(); }

  if (axial == 1){
  des8b(); }

  // design a field_flattener
  des4();
  // design afield flattener

  flag_paraxial_y=0;
  efl1();

 // if (field_angl ==0){
   //  cout << "\a";}


  // colour 3 - blue !!!!!!!!!
  colourray =3;
  notgraph();
 // ab_plot = ((ab * mag));
  //ab_plot2 = ((ab * scale_p3));
  ab2 = ab;
  colourray=1;
  // color 3 -- blue !!!!!!!!
  // colour 1
  notgraph();
 // ab_plot = ((ab * mag));
  ab_plot = ((ab * scale_p));
  //

  if (th ==1){
  field_angl =mem_ang;
  //cout << "\a";
  third3();
  field_angl = angle_aa;
  }

   //if (field_angl ==0){
   //  cout << "\a";}


  // spherochromatism = diff between yellow and blue abs

 // sphero_ab = ab - ab2;
  spher();
  ab_plot2 = (sp_size*scale_p3);
  // end sphero


  if ((virt2 ==1)&(virt ==1)){

//  normalize = ab_plot;
  normalize = rel_shift;
  }

//  moveto(mem_x,mem_y+shift);

  small_x = data_x;
 // small_y = (data_y-ab_plot);
 small_y = ((ab_plot)-normalize)*mag_ab;
 //small_y2 = ((ab_plot2)-normalize)*scale_p3;
    small_y2 = ((ab_plot2))*scale_p3;


  big_x = (small_x*0.866);
  big_y = ((small_x*(0.5))+small_y);
  big_y2 = ((small_x*(0.5))+small_y2);

  if (virt != 1) {
   setcolor(4);
//   lineto(big_x-slide_x+pos_x,-big_y-slide_y+pos_y+shift);
  // lineto(big_x-slide_x+pos_x,-big_y2-slide_y+pos_y);
  }
   //     moveto(big_x-slide_x+pos_x,-big_y-slide_y+pos_y+shift);
    //lineto(big_x-slide_x+pos_x,-big_y2-slide_y+pos_y);
     setcolor(4);
//     lineto(big_x-slide_x+pos_x,-big_y-slide_y+pos_y-ab_plot2+shift);

  if (min_ab > ab) {
    min_ab = ab;
    min_x = (big_x-slide_x+pos_x);
    min_y = (-big_y-slide_y+pos_y);
    min_r1 =r1_now;
    min_r3 = r3_now;

  }

  if ((virt2 ==(no_steps/2))&(virt ==(no_steps/2))){

    xx_now = (big_x-slide_x+pos_x);
    yy_now = (-big_y-slide_y+pos_y);
  }

    //      circle(big_x-slide_x+pos_x,-big_y-slide_y+pos_y+shift,2);
  mem_x = big_x-slide_x+pos_x;
  mem_y = -big_y-slide_y+pos_y;
  data_x = data_x + step_a;
//  array2_x[virt][virt2] = big_x-slide_x+pos_x;
//  array2_y[virt][virt2]= -big_y-slide_y+pos_y;
       xx  = big_x-slide_x+pos_x;
       yy  = -big_y-slide_y+pos_y;

       (*surfaceptr[ss3]).check4();
       r33 =  1/c111;
       (*surfaceptr[ss4]).check4();
       r44 =  1/c111;
       (*surfaceptr[ss5]).check4();
       r55 =  1/c111;
       (*surfaceptr[ss6]).check4();
       r66 =  1/c111;
       num = num+1;
  // pointptr[num] = new datapoint(xx,yy,r33,r44,r55,r66,ab,ab2,
  // sp_size,coma_3,coma_4,coma_5,coma_6);

       (*surfaceptr[3]).check4();
       if (c111 !=0){
   r_col33 =  1/c111;}
       else {r_col33 = 0;}



  if (map_no==1){
   (*one_pointptr[num]).newpoint(xx,yy,sep33,r_col33,r33,r44,r55,r66,ab,ab2,
     sp_size,coma_3,coma_4,coma_5,coma_6);}

   if (map_no==2){
   (*two_pointptr[num]).newpoint(xx,yy,sep33,r_col33,r33,r44,r55,r66,ab,ab2,
     sp_size,coma_3,coma_4,coma_5,coma_6);}

   if (map_no==3){
   (*three_pointptr[num]).newpoint(xx,yy,sep33,r_col33,r33,r44,r55,r66,ab,ab2,
     sp_size,coma_3,coma_4,coma_5,coma_6);}

   if (map_no==4){
   (*four_pointptr[num]).newpoint(xx,yy,sep33,r_col33,r33,r44,r55,r66,ab,ab2,
     sp_size,coma_3,coma_4,coma_5,coma_6);}

   if (map_no==6){
   (*six_pointptr[num]).newpoint(xx,yy,sep33,r_col33,r33,r44,r55,r66,ab,ab2,
     sp_size,coma_3,coma_4,coma_5,coma_6);}

   if (map_no==7){
   (*seven_pointptr[num]).newpoint(xx,yy,sep33,r_col33,r33,r44,r55,r66,ab,ab2,
     sp_size,coma_3,coma_4,coma_5,coma_6);}

   if (map_no==8){
   (*eight_pointptr[num]).newpoint(xx,yy,sep33,r_col33,r33,r44,r55,r66,ab,ab2,
     sp_size,coma_3,coma_4,coma_5,coma_6);}

   if (map_no==9){
   (*nine_pointptr[num]).newpoint(xx,yy,sep33,r_col33,r33,r44,r55,r66,ab,ab2,
     sp_size,coma_3,coma_4,coma_5,coma_6);}



}
data_x = data_x -(step_a*no_steps);

r1_now = (r1_now - (no_steps*c111_inc));
// r3 increment * 6
now =0;
```

```
   }

/*      // link points
    for (virt2 = 1; virt2 <= (no_steps); virt2=virt2+1) {
     first_point = 1;
    for (virt = virt2; virt <= (no_steps*no_steps); virt=virt+no_steps) {

          if (first_point == 1){
          (*one_pointptr[virt]).readp();
          // moveto(xxp,yyp+shift);

          }
       (*one_pointptr[virt]).readp();
        setcolor(4);
           //    lineto(xxp,yyp+shift);


    first_point =0;
    }
     }
 // end link points
  */

   // fill points
   int poly[8];
   poly[0]=50; poly[1] = 50; poly[2] = 100; poly[3] =50;
   poly[4] = 100; poly[5] = 100; poly[6] = 50; poly[7] =100;

   int numb=0;
   int A=0;
   int start=1;

 /*  for (A = start; A <= (A+no_steps); A = A+no_steps) {

    for (virt2 = A; virt2 <= (virt2+1); virt2=virt2+1) {

       (*negpointptr[virt2]).readp();
        setcolor(4);
        //lineto(xxp,yyp+shift);
         poly[numb]= xxp;
         poly[numb+1] = yyp+shift;
         numb=numb+2;
    }

  }
  setfillstyle(SOLID_FILL, 2);
    // poly[0]=50; poly[1] = 50; poly[2] = 100; poly[3] =50;
    // poly[4] = 100; poly[5] = 100; poly[6] = 50; poly[7] =100;
 fillpoly(4,poly);
 // end fill points
   */

 // put parameter as they were


 // colour min spot
    setcolor(12);
 //   circle(min_x,min_y+shift,6);
    setcolor(15);
 // end color min spot
 // colour min spot
    setcolor(14);
  //  circle(xx_now,yy_now+shift,6);
    setcolor(15);
 // end color min spot

    (*surfaceptr[ss3]).newcl1(c1_save);
    (*surfaceptr[ss4]).newcl1(c2_save);
    (*surfaceptr[ss5]).newcl1(c3_save);
    (*surfaceptr[ss6]).newcl1(c4_save);
```

```
/*      flag_paraxial_y=0;
   efl1();

// end put paramters as they were

// GO TO PARAMTER MIN
  if (   ( (x > pos_xx-60) & (x<pos_xx-40 ) ) &( (y>pos_yy+15) & (y<pos_yy+25)
)) {
 // cout << "\a";
double c1_news=0, c3_news=0;
c1_news = 1/min_r1;
c3_news = 1/min_r3;

  (*surfaceptr[3]).newc11(c1_news);
  (*surfaceptr[5]).newc11(c3_news);


  if (axial == 2){
  des8(); }

  if (axial == 1){
  des8b(); }

  flag_paraxial_y=0;
  efl1();
}



// END GO

// PRINT OUT THINGS

c444 = fcvt(min_r1,1,&bit14,&bit24);
outtextxy(pos_xx+260,pos_yy-200,c444);
itoa(bit14,bit144,10);
itoa(bit24,bit244,10);
  if (bit24 !=0){
  outtextxy(pos_xx+230,pos_yy-200,"-");}
  else {    outtextxy(pos_xx+230,pos_yy-200,"+");}
outtextxy(pos_xx+243,pos_yy-200,"0.");
outtextxy(pos_xx+313,pos_yy-200,bit144);
outtextxy(pos_xx+302,pos_yy-200,"E");
    //as
c444 = fcvt(min_r3,1,&bit14,&bit24);
outtextxy(pos_xx+260,pos_yy-180,c444);
itoa(bit14,bit144,10);
itoa(bit24,bit244,10);
  if (bit24 !=0){
  outtextxy(pos_xx+230,pos_yy-180,"-");}
  else {    outtextxy(pos_xx+230,pos_yy-180,"+");}
outtextxy(pos_xx+243,pos_yy-180,"0.");
outtextxy(pos_xx+313,pos_yy-180,bit144);
outtextxy(pos_xx+302,pos_yy-180,"E");

c444 = fcvt(min_ab,3,&bit14,&bit24);
outtextxy(pos_xx+260,pos_yy-190,c444);
itoa(bit14,bit144,10);
itoa(bit24,bit244,10);
  if (bit24 !=0){
  outtextxy(pos_xx+230,pos_yy-190,"-");}
  else {    outtextxy(pos_xx+230,pos_yy-190,"+");}
outtextxy(pos_xx+243,pos_yy-190,"0.");
outtextxy(pos_xx+313-10,pos_yy-190,bit144);
outtextxy(pos_xx+302-10,pos_yy-190,"E");

outtextxy(pos_xx+200,pos_yy-295,"r3i");
outtextxy(pos_xx+200,pos_yy-235,"r5i");
outtextxy(pos_xx+205,pos_yy-200,"r3");
outtextxy(pos_xx+205,pos_yy-180,"r5");

    //   outtextxy(pos_xx+70,pos_yy-10,"Ab");
if (r2==0){
  outtextxy(pos_xx+100,pos_yy+65,"r3");}
else if (r2==1){
  outtextxy(pos_xx+100,pos_yy+65,"r4");}

outtextxy(pos_xx-130,pos_yy+30,"r5");


// END PRINT OUT THINGS
*/
// END GRAPHICAL PARAMETER SPACE SECTION
field_ang1 = mem_ang;
return 0;
} // end map
```

```
 // mapB.cpp
// test  Map  +++ spherochromatism Printout

 #include "c:\bc4\optical\op.h"
 #include <stdio.h>
 #include <graphics.h>
 #include <stdlib.h>
 #include <conio.h>
     //   #include "mouse.h"
 extern void mouse_infor(int *right, int *left, int *x_cd, int *y_cd);
 extern void mouse_show_cursor(void);
 extern int mouse_initialise(void);
 extern void mouse_range(int x1_cd, int y1_cd, int x2_cd, int y2_cd);

 int mapb(int r4_mode,int do_sepp,int multiple_maps, double c4_inc,int th,int r2,
int steps,
 int axial, int x,int y, double c1_inc, double c3_inc, double r3_s, double r5_s,
 int pos_xx, int pos_yy, int mag_mm, double angle_aa)  {
 // axial is a falg if = 1 then des8b if axial = 2 then des8()
 //coma_3 =0;
 //coma_4 =0;
 //coma_5 =0;
 //coma_6 =0;

 double sep33=0;
 (*surfaceptr[ss5]).check5();
 sep33 = D111;

// if (r4_mode==0){
     //    (*surfaceptr[7]).check5();
     //    sep33 = D111;}

 int surface5 =5;
 int ss=4;
 double c2_mem=0;
 double c_col_save=0;
 int mem_ss4=0;
 double d_save=0;
 int xx=0, yy=0;
 double sphero_ab=0;
 int num=0;
 int first_point=0;
 double r33=0, r44=0, r55=0, r66=0;
 double r_col33=0;
 double ab2=0;
 double xx_now=0, yy_now=0;
 dont_clear =1;
 double normalize=0,ab_plot=0,ab_plot2=0,
     c1_save=0, c2_save=0, c3_save=0, c4_save=0,
     r1_now=0, r3_now=0, fno=0, ddd50=0,d2_inc=0, d3_inc=0,
     c1_now=0, c3_now=0, bfl_mem=0, c11_inc=0, c22_inc=0,
     mid_x=0,mid_y=0,
     mem_x=0, mem_y=0,
     small_y=0,small_y2=0,
     small_x=0,midx=0,midy=0;

   int slide_y=0, slide_x=0, big_y=0,big_y2=0,
 big_x=0;
 double mem_ang = field_angl;
 double r3_shift =r3_s;
 double r5_shift =r5_s;
 double scale_p = mag_mm;
 int data_x=0, data_y=0, virt2=0, virt =0, lens =0,counter=0,field_des=0,
  des88=0,plot_3 =0,su1 =1, su2=2, su3=3, su4=4;

 // TRANSFER
 char * a444;
 char * a666;
 char * a_asp;
 char * c444;
```

```
int thick=0,bit1=0,bit2=0,bit14=0, bit24=0, bit16=0;
char bit144[30];
char bit166[30];
char bit244[30];
double new_c=0;
double c111_inc =c1_inc;
double c333_inc=c3_inc;

double min_ab=10000;
double min_x=0, min_y=0;
double min_r1=0, min_r3=0;

    //   double pos_x=pos_xx;
//double pos_y=pos_yy;
    int pos_x=pos_xx;
    int  pos_y=pos_yy;

double mag_ab = mag_mm;
int now=1;
//int array_x[21][21];
    //   int array_y[21][21];
int shift_yy=70;
// TRANSF8ER



// GRAPHICAL PARAMETER SPACE SECTION
min_ab =10000;

  field_angl = angle_aa;



colourray =1;
    //   field_angl = 0;

if (r2 ==0){
(*surfaceptr[ss3]).check4();
  c1_save = c111;
  r1_now = ((1/c111)+r3_shift);
  (*surfaceptr[ss4]).check4();
  c2_save = c111;
  (*surfaceptr[ss5]).check4();
  c3_save = c111;
  r3_now = ((1/c111)+r5_shift);
  (*surfaceptr[ss6]).check4();
  c4_save = c111;}
else if (r2==1){
  (*surfaceptr[ss3]).check4();
  c1_save = c111;
 // r1_now = ((1/c111)+r3_shift);
  (*surfaceptr[ss4]).check4();
  c2_save = c111;
  r1_now = ((1/c111)+r3_shift);
  (*surfaceptr[ss5]).check4();
  c3_save = c111;
  r3_now = ((1/c111)+r5_shift);
  (*surfaceptr[ss6]).check4();
  c4_save = c111;}


mem_x=pos_x;
mem_y=pos_y;
//circle(mem_x, mem_y, 20);
data_x =0;
data_y =0;
slide_x=0;
slide_y=0;

   // christmas section
```

```
    int no_steps = steps;

   // double step_a =15;
    int step_a = (235/no_steps);
    // end christmas section



// note go back 5 times 50

r1_now = r1_now - (c111_inc*(no_steps/2));
// end note
r3_now = r3_now - (c333_inc*(no_steps/2));
double y_box=0;
double z_box=0,delt=0;

// DRAW BOX
    setcolor(3);
    int inc =0;
    double pos_x2 = pos_x;
    double pos_y2 = pos_y+90;
    moveto(pos_x2,pos_y2);
    lineto(pos_x2,pos_y2-90);
    moveto(pos_x2,pos_y2-90);
    y_box = 180*0.577;
    lineto(pos_x2+180,-y_box+pos_y2-90);
    moveto(pos_x2+180,-y_box+pos_y2-90);
    lineto(pos_x2+180,-y_box+pos_y2);
    moveto(pos_x2+180,-y_box+pos_y2);
    lineto(pos_x2,pos_y2);
    moveto(pos_x2,pos_y2);

    delt = (180*0.866);

    lineto(pos_x2-180,pos_y2-delt);
    moveto(pos_x2-180,pos_y2-delt);
    lineto(pos_x2-180,pos_y2-delt-90);
    moveto(pos_x2-180,pos_y2-delt-90);

    lineto(pos_x2,pos_y2-90);

    moveto(pos_x2-180,pos_y2-delt-90);

    lineto(pos_x2,-y_box+pos_y2-90-delt);
    moveto(pos_x2,-y_box+pos_y2-90-delt);
    lineto(pos_x2,-y_box+pos_y2-delt);
    moveto(pos_x2,-y_box+pos_y2-delt);
    // xx
    lineto(pos_x2-180,pos_y2-delt);
    moveto(pos_x2-180,pos_y2-delt);

 moveto(pos_x2,-y_box+pos_y2-delt-90);

    lineto(pos_x2+180,-y_box+pos_y2-90);

    moveto(pos_x2,-y_box+pos_y2-delt);
    lineto(pos_x2+180,-y_box+pos_y2);

setfillstyle(SOLID_FILL,2);
midx = 110;
midy = 130;
bar(midx-8, midy-8, midx+8,midy+8);

setfillstyle(SOLID_FILL,2);
midx = 110;
midy = 320;
bar(midx-8, midy-8, midx+8,midy+8);

setfillstyle(SOLID_FILL,2);
midx = 230;
```

```
midy = 420;
bar(midx-8, midy-8, midx+8,midy+8);

setfillstyle(SOLID_FILL,2);
midx = 230;
midy = 50;
bar(midx-8, midy-8, midx+8,midy+8);


// END DRAW BOX


setcolor(12);

for (virt2 = 1; virt2 <= (no_steps); virt2++) {
  r3_now = r3_now + c333_inc;;

  c3_now = (1/r3_now);


  (*surfaceptr[ss5]).newc11(c3_now);
  //data_y = data_y -5;
  inc = virt2-1;

   // slide_x= ((inc*13)*0.866);
   // slide_y= ((inc*13)*0.5);



      slide_x= ((inc*step_a)*0.766);
     slide_y= ((inc*step_a)*0.643);

     // TEST SECTION


     for (virt = 1; virt <= (no_steps); virt++) {

  r1_now = r1_now + c111_inc;
  c1_now = (1/r1_now);

  if (r2==0){
    (*surfaceptr[ss3]).newc11(c1_now);}
  else if (r2==1){
    (*surfaceptr[ss4]).newc11(c1_now);}

  if (axial == 2){
  des8(); }

  if (axial == 1){
  des8b(); }

  // design a field_flattener
  des4();
  // design afield flattener

  flag_paraxial_y=0;
  efl1();

 // if (field_angl ==0){
   //  cout << "\a";}


  // colour 3 - blue !!!!!!!!!
  colourray =3;
  notgraph();
 // ab_plot = ((ab * mag));
  //ab_plot2 = ((ab * scale_p3));
  ab2 = ab;
  colourray=1;
  // color 3 -- blue !!!!!!!!
 // colour 1
 notgraph();
// ab_plot = ((ab * mag));
 ab_plot = ((ab * scale_p));
 //

 if (th ==1){
 field_angl =mem_ang;
 //cout << "\a";
 third3();
 field_angl = angle_aa;
 }

  //if (field_angl ==0){
  //  cout << "\a";}


 // spherochromatism = diff between yellow and blue abs

// sphero_ab = ab - ab2;
 spher();
 ab_plot2 = (sp_size*scale_p3);
 // end sphero


 if ((virt2 ==1)&(virt ==1)){

 normalize = ab_plot;
 rel_shift=normalize;
 }

 moveto(mem_x,mem_y);

 small_x = data_x;
// small_y = (data_y-ab_plot);
small_y = ((ab_plot)-normalize)*mag_ab;
//small_y2 = ((ab_plot2)-normalize)*scale_p3;
   small_y2 = ((ab_plot2))*scale_p3;


 big_x = (small_x*0.866);
 big_y = ((small_x*(0.5))+small_y);
 big_y2 = ((small_x*(0.5))+small_y2);

 if (virt != 1) {
  lineto(big_x-slide_x+pos_x,-big_y-slide_y+pos_y);
 // lineto(big_x-slide_x+pos_x,-big_y2-slide_y+pos_y);
 }
   moveto(big_x-slide_x+pos_x,-big_y-slide_y+pos_y);
   //lineto(big_x-slide_x+pos_x,-big_y2-slide_y+pos_y);
    lineto(big_x-slide_x+pos_x,-big_y-slide_y+pos_y-ab_plot2);

 if (min_ab > ab) {
   min_ab = ab;
   min_x = (big_x-slide_x+pos_x);
   min_y = (-big_y-slide_y+pos_y);
   min_r1 =r1_now;
   min_r3 = r3_now;

 }

 if ((virt2 ==(no_steps/2))&(virt ==(no_steps/2))){

   xx_now = (big_x-slide_x+pos_x);
   yy_now = (-big_y-slide_y+pos_y);

 }
```

```
    circle(big_x-slide_x+pos_x,-big_y-slide_y+pos_y,2);
   mem_x = big_x-slide_x+pos_x;
   mem_y = -big_y-slide_y+pos_y;
   data_x = data_x + step_a;
//  array2_x[virt][virt2] = big_x-slide_x+pos_x;
//  array2_y[virt][virt2]= -big_y-slide_y+pos_y;
        xx  = big_x-slide_x+pos_x;
        yy  = -big_y-slide_y+pos_y;

        (*surfaceptr[ss3]).check4();
        r33 =  1/c111;
        (*surfaceptr[ss4]).check4();
        r44 =  1/c111;
        (*surfaceptr[ss5]).check4();
        r55 =  1/c111;
        (*surfaceptr[ss6]).check4();
        r66 =  1/c111;
        num = num+1;
   // pointptr[num] = new datapoint(xx,yy,r33,r44,r55,r66,ab,ab2,
   // sp_size,coma_3,coma_4,coma_5,coma_6);

       (*surfaceptr[3]).check4();
        if (c111 !=0){
    r_col33 =  1/c111;}
        else {r_col33 = 0;}


    (*pointptr[num]).newpoint(xx,yy,sep33,r_col33,r33,r44,r55,r66,ab,ab2,
      sp_size,coma_3,coma_4,coma_5,coma_6);

}
data_x = data_x -(step_a*no_steps);

r1_now = (r1_now - (no_steps*c111_inc));
// r3 increment * 6
now =0;
  }

    // link points
    for (virt2 = 1; virt2 <= (no_steps); virt2=virt2+1) {
     first_point = 1;
    for (virt = virt2; virt <= (no_steps*no_steps); virt=virt+no_steps) {

         if (first_point == 1){
         (*pointptr[virt]).readp();
       moveto(xxp,yyp);

         }
       (*pointptr[virt]).readp();
        lineto(xxp,yyp);
    first_point =0;
    }
     }
// end link points


// put parameter as they were


// colour min spot
   setcolor(12);
   circle(min_x,min_y,6);
   setcolor(15);
// end color min spot
// colour min spot
   setcolor(14);
   circle(xx_now,yy_now,6);
   setcolor(15);
// end color min spot
```

```
   (*surfaceptr[ss3]).newcl1(c1_save);
   (*surfaceptr[ss4]).newcl1(c2_save);
   (*surfaceptr[ss5]).newcl1(c3_save);
   (*surfaceptr[ss6]).newcl1(c4_save);

   flag_paraxial_y=0;
   efl1();

// end put paramters as they were

// second map

 if (multiple_maps ==1){
   // clock
   if (do_sepp !=1 ){
   setfillstyle(SOLID_FILL, 11);
   midx = 10;
   midy = 470;
   bar(midx-3, midy-3, midx+3,midy+3);

   setfillstyle(SOLID_FILL, 11);
   midx = 20;
   midy = 470;
   bar(midx-3, midy-3, midx+3,midy+3);

   setfillstyle(SOLID_FILL, 11);
   midx = 30;
    midy = 470;
   bar(midx-3, midy-3, midx+3,midy+3);

   setfillstyle(SOLID_FILL, 11);
   midx = 40;
   midy = 470;
   bar(midx-3, midy-3, midx+3,midy+3);

   setfillstyle(SOLID_FILL, 11);
   midx = 50;
   midy = 470;
   bar(midx-3, midy-3, midx+3,midy+3);

   setfillstyle(SOLID_FILL, 11);
   midx = 60;
   midy = 470;
   bar(midx-3, midy-3, midx+3,midy+3);

   setfillstyle(SOLID_FILL, 11);
   midx = 70;
   midy = 470;
   bar(midx-3, midy-3, midx+3,midy+3);

   setfillstyle(SOLID_FILL, 11);
   midx = 80;
   midy = 470;
   bar(midx-3, midy-3, midx+3,midy+3);

// r4_mode
   c2_mem = c2_save;

  // below changed 1/6/94
  // (*surfaceptr[ss3]).check4();
      (*surfaceptr[ss4]).check4();
  // above changed 1/6/95

   c_col_save = c111;
   ss=ss4;

   if (t2_run==0){
    (*surfaceptr[4]).check4();
    c_col_save =c111;
    c2_save = c_col_save;}
   if (r4_mode==0){
  // mem_ss4 = ss4;
   //cout << "\a";
   ss = 3;
   surface5 = 7;
   (*surfaceptr[surface5]).check5();
    sep33 = D111;

   (*surfaceptr[ss]).check4();
   c_col_save =c111;
   c2_save = c_col_save;

   }
  // else { ss4 =4;}
// r4_nmode
   // clock

   if (ss4 == 6){ surface5 =7;}


   field_angl =mem_ang;
   new_c = (1/c2_save)+(4*r4_inc);
   new_c=1/new_c;
  (*surfaceptr[ss]).newc11(new_c);

mapc(surface5,1,90,multiple_maps, c4_inc,1,r2,steps,
axial, x,y, c1_inc, c3_inc,
r3_s, r5_s,
pos_xx, pos_yy, mag_mm, angle_aa);

    //   cout << "\a";
   setfillstyle(SOLID_FILL, 15);
   midx = 10;
   midy = 470;
   bar(midx-3, midy-3, midx+3,midy+3);


   (*surfaceptr[ss]).newc11(c2_save);

  new_c = (1/c2_save)+(3*r4_inc);
   new_c=1/new_c;
  (*surfaceptr[ss]).newc11(new_c);

mapc(surface5,2,90,multiple_maps, c4_inc,1,r2,steps,
axial, x,y, c1_inc, c3_inc,
r3_s, r5_s,
pos_xx, pos_yy, mag_mm, angle_aa);

   setfillstyle(SOLID_FILL, 15);
   midx = 20;
   midy = 470;
   bar(midx-3, midy-3, midx+3,midy+3);


  (*surfaceptr[ss]).newc11(c2_save);

 new_c = (1/c2_save)+(2*r4_inc);
   new_c=1/new_c;
  (*surfaceptr[ss]).newc11(new_c);

mapc(surface5,3,90,multiple_maps, c4_inc,1,r2,steps,
axial, x,y, c1_inc, c3_inc,
r3_s, r5_s,
pos_xx, pos_yy, mag_mm, angle_aa);


  (*surfaceptr[ss]).newc11(c2_save);
```

```
    setfillstyle(SOLID_FILL, 15);
    midx = 30;
    midy = 470;
    bar(midx-3, midy-3, midx+3,midy+3);


  new_c = (1/c2_save)+(r4_inc);
    new_c=1/new_c;
  (*surfaceptr[ss]).newcl1(new_c);

mapc(surface5,4,90,multiple_maps, c4_inc,1,r2,steps,
axial, x,y, c1_inc, c3_inc,
r3_s, r5_s,
pos_xx, pos_yy, mag_mm, angle_aa);

  (*surfaceptr[ss]).newcl1(c2_save);

    setfillstyle(SOLID_FILL, 15);
    midx = 40;
    midy = 470;
    bar(midx-3, midy-3, midx+3,midy+3);


     new_c = (1/c2_save)-(r4_inc);
    new_c=1/new_c;
  (*surfaceptr[ss]).newcl1(new_c);

mapc(surface5,6,90,multiple_maps, c4_inc,1,r2,steps,
axial, x,y, c1_inc, c3_inc,
r3_s, r5_s,
pos_xx, pos_yy, mag_mm, angle_aa);

    setfillstyle(SOLID_FILL, 15);
    midx = 50;
    midy = 470;
    bar(midx-3, midy-3, midx+3,midy+3);


    (*surfaceptr[ss]).newcl1(c2_save);

   new_c = (1/c2_save)-(2*r4_inc);
    new_c=1/new_c;
  (*surfaceptr[ss]).newcl1(new_c);

mapc(surface5,7,90,multiple_maps, c4_inc,1,r2,steps,
axial, x,y, c1_inc, c3_inc,
r3_s, r5_s,
pos_xx, pos_yy, mag_mm, angle_aa);

    setfillstyle(SOLID_FILL, 15);
    midx = 60;
    midy = 470;
    bar(midx-3, midy-3, midx+3,midy+3);


  (*surfaceptr[ss]).newcl1(c2_save);

  new_c = (1/c2_save)-(3*r4_inc);
    new_c=1/new_c;
  (*surfaceptr[ss]).newcl1(new_c);

mapc(surface5,8,90,multiple_maps, c4_inc,1,r2,steps,
axial, x,y, c1_inc, c3_inc,
r3_s, r5_s,
pos_xx, pos_yy, mag_mm, angle_aa);


  (*surfaceptr[ss]).newcl1(c2_save);

    setfillstyle(SOLID_FILL, 15);
    midx = 70;
    midy = 470;
    bar(midx-3, midy-3, midx+3,midy+3);


  new_c = (1/c2_save)-(4*r4_inc);
    new_c=1/new_c;
  (*surfaceptr[ss]).newcl1(new_c);

mapc(surface5,9,90,multiple_maps, c4_inc,1,r2,steps,
axial, x,y, c1_inc, c3_inc,
r3_s, r5_s,
pos_xx, pos_yy, mag_mm, angle_aa);

  (*surfaceptr[ss]).newcl1(c2_save);

    setfillstyle(SOLID_FILL, 15);
    midx = 80;
    midy = 470;
    bar(midx-3, midy-3, midx+3,midy+3);

  }      //end if not do sep

    else if (do_sepp ==1) {

    setfillstyle(SOLID_FILL, 11);
    midx = 10;
    midy = 470;
    bar(midx-3, midy-3, midx+3,midy+3);

    setfillstyle(SOLID_FILL, 11);
    midx = 20;
    midy = 470;
    bar(midx-3, midy-3, midx+3,midy+3);

    setfillstyle(SOLID_FILL, 11);
    midx = 30;
     midy = 470;
    bar(midx-3, midy-3, midx+3,midy+3);

    setfillstyle(SOLID_FILL, 11);
    midx = 40;
    midy = 470;
    bar(midx-3, midy-3, midx+3,midy+3);

    setfillstyle(SOLID_FILL, 11);
    midx = 50;
    midy = 470;
    bar(midx-3, midy-3, midx+3,midy+3);

    setfillstyle(SOLID_FILL, 11);
    midx = 60;
    midy = 470;
    bar(midx-3, midy-3, midx+3,midy+3);

    setfillstyle(SOLID_FILL, 11);
    midx = 70;
    midy = 470;
    bar(midx-3, midy-3, midx+3,midy+3);

    setfillstyle(SOLID_FILL, 11);
    midx = 80;
    midy = 470;
    bar(midx-3, midy-3, midx+3,midy+3);

  if (r4_mode==0){
    ss = 3;
    surface5 = 7;
    (*surfaceptr[surface5]).check5();
     sep33 = D111;
```

```
   c2_save = c_col_save;

   }

      (*surfaceptr[surface5]).check5();
       d_save  = D111;

   // clock
   field_angl =mem_ang;

   new_c = (d_save)+(4*sep_inc);

 (*surfaceptr[surface5]).newd22(new_c);

mapc(surface5,1,90,multiple_maps, c4_inc,1,r2,steps,
axial, x,y, c1_inc, c3_inc,
r3_s, r5_s,
pos_xx, pos_yy, mag_mm, angle_aa);

   setfillstyle(SOLID_FILL, 15);
   midx = 10;
   midy = 470;
   bar(midx-3, midy-3, midx+3,midy+3);


   (*surfaceptr[surface5]).newd22(d_save);

  new_c = (d_save)+(3*sep_inc);

 (*surfaceptr[surface5]).newd22(new_c);

mapc(surface5,2,90,multiple_maps, c4_inc,1,r2,steps,
axial, x,y, c1_inc, c3_inc,
r3_s, r5_s,
pos_xx, pos_yy, mag_mm, angle_aa);

   setfillstyle(SOLID_FILL, 15);
   midx = 20;
   midy = 470;
   bar(midx-3, midy-3, midx+3,midy+3);


 (*surfaceptr[surface5]).newd22(d_save);

 new_c = (d_save)+(2*sep_inc);

 (*surfaceptr[surface5]).newd22(new_c);

mapc(surface5,3,90,multiple_maps, c4_inc,1,r2,steps,
axial, x,y, c1_inc, c3_inc,
r3_s, r5_s,
pos_xx, pos_yy, mag_mm, angle_aa);

   setfillstyle(SOLID_FILL, 15);
   midx = 30;
   midy = 470;
   bar(midx-3, midy-3, midx+3,midy+3);


 (*surfaceptr[surface5]).newd22(d_save);


 new_c = (d_save)+(sep_inc);

 (*surfaceptr[surface5]).newd22(new_c);

mapc(surface5,4,90,multiple_maps, c4_inc,1,r2,steps,
axial, x,y, c1_inc, c3_inc,
r3_s, r5_s,
pos_xx, pos_yy, mag_mm, angle_aa);

 (*surfaceptr[surface5]).newd22(d_save);



   setfillstyle(SOLID_FILL, 15);
   midx = 40;
   midy = 470;
   bar(midx-3, midy-3, midx+3,midy+3);


    new_c = (d_save)-(sep_inc);

 (*surfaceptr[surface5]).newd22(new_c);




mapc(surface5,6,90,multiple_maps, c4_inc,1,r2,steps,
axial, x,y, c1_inc, c3_inc,
r3_s, r5_s,
pos_xx, pos_yy, mag_mm, angle_aa);

   setfillstyle(SOLID_FILL, 15);
   midx = 50;
   midy = 470;
   bar(midx-3, midy-3, midx+3,midy+3);


   (*surfaceptr[surface5]).newd22(d_save);

  new_c = (d_save)-(2*sep_inc);

 (*surfaceptr[surface5]).newd22(new_c);

mapc(surface5,7,90,multiple_maps, c4_inc,1,r2,steps,
axial, x,y, c1_inc, c3_inc,
r3_s, r5_s,
pos_xx, pos_yy, mag_mm, angle_aa);

   setfillstyle(SOLID_FILL, 15);
   midx = 60;
   midy = 470;
   bar(midx-3, midy-3, midx+3,midy+3);


 (*surfaceptr[surface5]).newd22(d_save);

 new_c = (d_save)-(3*sep_inc);

 (*surfaceptr[surface5]).newd22(new_c);

mapc(surface5,8,90,multiple_maps, c4_inc,1,r2,steps,
axial, x,y, c1_inc, c3_inc,
r3_s, r5_s,
pos_xx, pos_yy, mag_mm, angle_aa);


 (*surfaceptr[surface5]).newd22(d_save);

   setfillstyle(SOLID_FILL, 15);
   midx = 70;
   midy = 470;
   bar(midx-3, midy-3, midx+3,midy+3);

   setfillstyle(SOLID_FILL, 15);
   midx = 80;
   midy = 470;
   bar(midx-3, midy-3, midx+3,midy+3);
```

```
      new_c = (d_save)-(4*sep_inc);

    (*surfaceptr[surface5]).newd22(new_c);


   mapc(surface5,9,90,multiple_maps, c4_inc,1,r2,steps,
   axial, x,y, c1_inc, c3_inc,
   r3_s, r5_s,
   pos_xx, pos_yy, mag_mm, angle_aa);

      setfillstyle(SOLID_FILL, 15);
      midx = 80;
      midy = 470;
      bar(midx-3, midy-3, midx+3,midy+3);


      (*surfaceptr[surface5]).newd22(d_save);

         c2_save = c2_mem;


      }
       // end do_sepp

    }

   /* drawmp(1,40,multiple_maps, c4_inc,1,r2,steps,
   axial, x,y, c1_inc, c3_inc,
   r3_s, r5_s,
   pos_xx, pos_yy, mag_mm, angle_aa);

     */

   // end second map

   // GO TO PARAMTER MIN
     if (    ( (x > pos_xx-60) & (x<pos_xx-40 ) ) &( (y>pos_yy+15) & (y<pos_yy+25)
)) {
    // cout << "\a";
   double c1_news=0, c3_news=0;
   c1_news = 1/min_r1;
   c3_news = 1/min_r3;

     (*surfaceptr[ss3]).newc11(c1_news);
     (*surfaceptr[ss5]).newc11(c3_news);


     if (axial == 2){
     des8(); }

     if (axial == 1){
     des8b(); }

     flag_paraxial_y=0;
     ef11();
   }



   // END GO

   // PRINT OUT THINGS

   c444 = fcvt(min_r1,1,&bit14,&bit24);
   outtextxy(pos_xx+260,pos_yy-200,c444);
   itoa(bit14,bit144,10);
   itoa(bit24,bit244,10);
     if (bit24 !=0){
     outtextxy(pos_xx+230,pos_yy-200,"-");}
```

```
  else {    outtextxy(pos_xx+230,pos_yy-200,"+");}
outtextxy(pos_xx+243,pos_yy-200,"0.");
outtextxy(pos_xx+313,pos_yy-200,bit144);
outtextxy(pos_xx+302,pos_yy-200,"E");
    //as
c444 = fcvt(min_r3,1,&bit14,&bit24);
outtextxy(pos_xx+260,pos_yy-180,c444);
itoa(bit14,bit144,10);
itoa(bit24,bit244,10);
  if (bit24 !=0){
  outtextxy(pos_xx+230,pos_yy-180,"-");}
  else {    outtextxy(pos_xx+230,pos_yy-180,"+");}
outtextxy(pos_xx+243,pos_yy-180,"0.");
outtextxy(pos_xx+313,pos_yy-180,bit144);
outtextxy(pos_xx+302,pos_yy-180,"E");

c444 = fcvt(min_ab,3,&bit14,&bit24);
outtextxy(pos_xx+260,pos_yy-190,c444);
itoa(bit14,bit144,10);
itoa(bit24,bit244,10);
  if (bit24 !=0){
  outtextxy(pos_xx+230,pos_yy-190,"-");}
  else {    outtextxy(pos_xx+230,pos_yy-190,"+");}
outtextxy(pos_xx+243,pos_yy-190,"0.");
outtextxy(pos_xx+313-10,pos_yy-190,bit144);
outtextxy(pos_xx+302-10,pos_yy-190,"E");

outtextxy(pos_xx+200,pos_yy-295,"r3i");
outtextxy(pos_xx+200,pos_yy-235,"r5i");

outtextxy(pos_xx+205,pos_yy-200,"r3");
outtextxy(pos_xx+205,pos_yy-180,"r5");

    //   outtextxy(pos_xx+70,pos_yy-10,"Ab");
if (r2==0){
  outtextxy(pos_xx+100,pos_yy+65,"r3");}
else if (r2==1){
  outtextxy(pos_xx+100,pos_yy+65,"r4");}

outtextxy(pos_xx-130,pos_yy+30,"r5");


// END PRINT OUT THINGS

// end put paramters as they were


// END GRAPHICAL PARAMETER SPACE SECTION
field_angl = mem_ang;
return 0;
} // end map
```

```
  // DRAWMP.cpp
 // test  Map  single map Printout with one value of r4

  #include "c:\bc4\optical\op.h"
  #include <stdio.h>
  #include <graphics.h>
  #include <stdlib.h>
  #include <conio.h>
      //   #include "mouse.h"
  extern void mouse_infor(int *right, int *left, int *x_cd, int *y_cd);
  extern void mouse_show_cursor(void);
  extern int mouse_initialise(void);
  extern void mouse_range(int x1_cd, int y1_cd, int x2_cd, int y2_cd);
  int drawmp(int map_no,double shift,int multiple_maps, double c4_inc,int th,int r
2,int steps,
  int axial, int x,int y, double c1_inc, double c3_inc, double r3_s, double r5_s,
  int pos_xx, int pos_yy, int mag_mm, double angle_aa)  {

    int no_steps = steps;
    int plane_bit=0;
  // axial is a falg if = 1 then des8b if axial = 2 then des8()
  //coma_3 =0;
  //coma_4 =0;
  //coma_5 =0;
  //coma_6 =0;
  //double shift=0;
  //double inc=0;
    int inc =0;
  int xx=0, yy=0;
  double sphero_ab=0;
  int num=0;
  int first_point=0;
  double r33=0, r44=0, r55=0, r66=0;
  double ab2=0;
  double xx_now=0, yy_now=0;
  dont_clear =1;
  double normalize=0,ab_plot=0,ab_plot2=0,
      c1_save=0, c2_save=0, c3_save=0, c4_save=0,
      r1_now=0, r3_now=0, fno=0, ddd50=0,d2_inc=0, d3_inc=0,
      c1_now=0, c3_now=0, bfl_mem=0, c11_inc=0, c22_inc=0,
      mid_x=0,mid_y=0,
      mem_x=0, mem_y=0,
      small_y=0,small_y2=0,
      small_x=0,midx=0,midy=0;

    int slide_y=0, slide_x=0, big_y=0,big_y2=0,
  big_x=0;
  double mem_ang = field_angl;
  double r3_shift =r3_s;
  double r5_shift =r5_s;
  double scale_p = mag_mm;
  int data_x=0, data_y=0, virt2=0, virt =0, lens =0,counter=0,field_des=0,
   des88=0,plot_3 =0,su1 =1, su2=2, su3=3, su4=4;

  // TRANSFER
  char * a444;
  char * a666;
  char * a_asp;
  char * c444;
  int thick=0,bit1=0,bit2=0,bit14=0, bit24=0, bit16=0;
  char bit144[30];
  char bit166[30];
  char bit244[30];

  double c111_inc =c1_inc;
  double c333_inc=c3_inc;

  double min_ab=10000;
  double min_x=0, min_y=0;
  double min_r1=0, min_r3=0;
```

```
    //   double pos_x=pos_xx;
//double pos_y=pos_yy;
    int pos_x=pos_xx;
    int  pos_y=pos_yy;

double mag_ab = mag_mm;
int now=1;
//int array_x[21][21];
    //   int array_y[21][21];
int shift_yy=70;
// TRANSF8ER



// GRAPHICAL PARAMETER SPACE SECTION
min_ab =10000;

  field_angl = angle_aa;



colourray =1;
    //   field_angl = 0;
double y_box=0;
double delt=0;
// DRAW BOX
   setcolor(3);
   inc =0;
   double pos_x2 = pos_x;
   double pos_y2 = pos_y+90;
   moveto(pos_x2,pos_y2);
   lineto(pos_x2,pos_y2-90);
   moveto(pos_x2,pos_y2-90);
   y_box = 180*0.577;
   lineto(pos_x2+180,-y_box+pos_y2-90);
   moveto(pos_x2+180,-y_box+pos_y2-90);
   lineto(pos_x2+180,-y_box+pos_y2);
   moveto(pos_x2+180,-y_box+pos_y2);
   lineto(pos_x2,pos_y2);
   moveto(pos_x2,pos_y2);

   delt = (180*0.866);

   lineto(pos_x2-180,pos_y2-delt);
   moveto(pos_x2-180,pos_y2-delt);
   lineto(pos_x2-180,pos_y2-delt-90);
   moveto(pos_x2-180,pos_y2-delt-90);

   lineto(pos_x2,pos_y2-90);

   moveto(pos_x2-180,pos_y2-delt-90);

   lineto(pos_x2,-y_box+pos_y2-90-delt);
   moveto(pos_x2,-y_box+pos_y2-90-delt);
   lineto(pos_x2,-y_box+pos_y2-delt);
   moveto(pos_x2,-y_box+pos_y2-delt);
   // xx
   lineto(pos_x2-180,pos_y2-delt);
   moveto(pos_x2-180,pos_y2-delt);

 moveto(pos_x2,-y_box+pos_y2-delt-90);

   lineto(pos_x2+180,-y_box+pos_y2-90);

   moveto(pos_x2,-y_box+pos_y2-delt);
   lineto(pos_x2+180,-y_box+pos_y2);

setfillstyle(SOLID_FILL,2);
midx = 110;
```

```
midy = 130;
bar(midx-8, midy-8, midx+8,midy+8);

setfillstyle(SOLID_FILL,2);
midx = 110;
midy = 320;
bar(midx-8, midy-8, midx+8,midy+8);

setfillstyle(SOLID_FILL,2);
midx = 230;
midy = 420;
bar(midx-8, midy-8, midx+8,midy+8);

setfillstyle(SOLID_FILL,2);
midx = 230;
midy = 50;
bar(midx-8, midy-8, midx+8,midy+8);


// END DRAW BOX

   // link points
   for (virt2 = 1; virt2 <= (no_steps); virt2=virt2+1) {
    first_point = 1;
   for (virt = virt2; virt <= (no_steps*no_steps); virt=virt+no_steps) {

         if (first_point == 1){
       //  (*one_pointptr[virt]).readp();
      if (map_no==1){(*one_pointptr[virt]).readp();}
      if (map_no==2){(*two_pointptr[virt]).readp();}
      if (map_no==3){(*three_pointptr[virt]).readp();}
      if (map_no==4){(*four_pointptr[virt]).readp();}
      if (map_no==5){(*pointptr[virt]).readp();}
      if (map_no==6){(*six_pointptr[virt]).readp();}
      if (map_no==7){(*seven_pointptr[virt]).readp();}
      if (map_no==8){(*eight_pointptr[virt]).readp();}
      if (map_no==9){(*nine_pointptr[virt]).readp();}

      if (map_no==11){(*one_pointptr_b[virt]).readp();}
      if (map_no==12){(*two_pointptr_b[virt]).readp();}
      if (map_no==13){(*three_pointptr_b[virt]).readp();}
      if (map_no==14){(*four_pointptr_b[virt]).readp();}
      if (map_no==15){(*pointptr_b[virt]).readp();}
      if (map_no==16){(*six_pointptr_b[virt]).readp();}
      if (map_no==17){(*seven_pointptr_b[virt]).readp();}
      if (map_no==18){(*eight_pointptr_b[virt]).readp();}
      if (map_no==19){(*nine_pointptr_b[virt]).readp();}

      moveto(xxp,yyp+shift);


         }
          //   (*one_pointptr[virt]).readp();
   if (map_no==1){setcolor(15);(*one_pointptr[virt]).readp();}
   if (map_no==2){setcolor(15);(*two_pointptr[virt]).readp();}
   if (map_no==3){setcolor(15);(*three_pointptr[virt]).readp();}
   if (map_no==4){setcolor(15);(*four_pointptr[virt]).readp();}
   if (map_no==5){setcolor(15);(*pointptr[virt]).readp();}
   if (map_no==6){setcolor(15);(*six_pointptr[virt]).readp();}
   if (map_no==7){setcolor(15);(*seven_pointptr[virt]).readp();}
   if (map_no==8){setcolor(15);(*eight_pointptr[virt]).readp();}
   if (map_no==9){setcolor(15);(*nine_pointptr[virt]).readp();}

   if (map_no==11){setcolor(15);(*one_pointptr_b[virt]).readp();}
   if (map_no==12){setcolor(15);(*two_pointptr_b[virt]).readp();}
   if (map_no==13){setcolor(15);(*three_pointptr_b[virt]).readp();}
   if (map_no==14){setcolor(15);(*four_pointptr_b[virt]).readp();}
   if (map_no==15){setcolor(15);(*pointptr_b[virt]).readp();}
   if (map_no==16){setcolor(15);(*six_pointptr_b[virt]).readp();}
   if (map_no==17){setcolor(15);(*seven_pointptr_b[virt]).readp();}
   if (map_no==18){setcolor(15);(*eight_pointptr_b[virt]).readp();}
   if (map_no==19){setcolor(15);(*nine_pointptr_b[virt]).readp();}


      // setcolor(4);
       //circle(xxp,yyp+shift,2);

       if (min_ab > abp) {
          min_ab = abp;
          min_x = xxp;
          min_y = yyp;
       }

       lineto(xxp,yyp+shift);
       circle(xxp,yyp+shift,2);

   first_point =0;
   }
    }

    int maxt=0;
// end link points
for (virt = 1; virt <= (no_steps*no_steps); virt=virt+no_steps) {
   first_point =1;
    maxt = no_steps+maxt;
    for (virt2 = virt; virt2 <= (maxt); virt2=virt2+1) {
         if (first_point == 1){
       //  (*one_pointptr[virt2]).readp();
      if (map_no==1){ (*one_pointptr[virt2]).readp();}
      if (map_no==2){ (*two_pointptr[virt2]).readp();}
      if (map_no==3){ (*three_pointptr[virt2]).readp();}
      if (map_no==4){ (*four_pointptr[virt2]).readp();}
      if (map_no==5){(*pointptr[virt2]).readp();}
      if (map_no==6){(*six_pointptr[virt2]).readp();}
      if (map_no==7){(*seven_pointptr[virt2]).readp();}
      if (map_no==8){(*eight_pointptr[virt2]).readp();}
      if (map_no==9){(*nine_pointptr[virt2]).readp();}

      if (map_no==11){ (*one_pointptr_b[virt2]).readp();}
      if (map_no==12){ (*two_pointptr_b[virt2]).readp();}
      if (map_no==13){ (*three_pointptr_b[virt2]).readp();}
      if (map_no==14){ (*four_pointptr_b[virt2]).readp();}
      if (map_no==15){(*pointptr_b[virt2]).readp();}
      if (map_no==16){(*six_pointptr_b[virt2]).readp();}
      if (map_no==17){(*seven_pointptr_b[virt2]).readp();}
      if (map_no==18){(*eight_pointptr_b[virt2]).readp();}
      if (map_no==19){(*nine_pointptr_b[virt2]).readp();}

      moveto(xxp,yyp+shift);
      // first_point=0;
         }
         // (*one_pointptr[virt2]).readp();
   if (map_no==1){ setcolor(15); (*one_pointptr[virt2]).readp();}
   if (map_no==2){ setcolor(15); (*two_pointptr[virt2]).readp();}
   if (map_no==3){ setcolor(15); (*three_pointptr[virt2]).readp();}
   if (map_no==4){ setcolor(15); (*four_pointptr[virt2]).readp();}
   if (map_no==5){setcolor(15);(*pointptr[virt2]).readp();}
   if (map_no==6){setcolor(15);(*six_pointptr[virt2]).readp();}
   if (map_no==7){setcolor(15);(*seven_pointptr[virt2]).readp();}
   if (map_no==8){setcolor(15);(*eight_pointptr[virt2]).readp();}
   if (map_no==9){setcolor(15);(*nine_pointptr[virt2]).readp();}

   if (map_no==11){ setcolor(15); (*one_pointptr_b[virt2]).readp();}
   if (map_no==12){ setcolor(15); (*two_pointptr_b[virt2]).readp();}
   if (map_no==13){ setcolor(15); (*three_pointptr_b[virt2]).readp();}
   if (map_no==14){ setcolor(15); (*four_pointptr_b[virt2]).readp();}
   if (map_no==15){setcolor(15);(*pointptr_b[virt2]).readp();}
   if (map_no==16){setcolor(15);(*six_pointptr_b[virt2]).readp();}
   if (map_no==17){setcolor(15);(*seven_pointptr_b[virt2]).readp();}
   if (map_no==18){setcolor(15);(*eight_pointptr_b[virt2]).readp();}
   if (map_no==19){setcolor(15);(*nine_pointptr_b[virt2]).readp();}

      // setcolor(4);
       //circle(xxp,yyp+shift,2);

       lineto(xxp,yyp+shift);

       circle(xxp,yyp+shift,2);
       setcolor(11);
       lineto(xxp,(yyp+shift-(sphero2*scale_p3)));
       setcolor(15);
       moveto(xxp,yyp+shift);
   first_point=0;
   }
    }

setcolor(12);
circle(min_x,min_y,4);

c444 = fcvt(min_ab,5,&bit14,&bit24);
outtextxy(528,450,c444);
itoa(bit14,bit144,10);
itoa(bit24,bit244,10);
  if (bit24 !=0){
  outtextxy(500,450,"-");}
  else {   outtextxy(500,450,"+");}
outtextxy(510,450,"0.");
outtextxy(595,450,bit144);
outtextxy(585,450,"E");
outtextxy(473,450,"AB");
setcolor(15);

  // fill points
  int poly[8];
  poly[0]=50; poly[1] = 50; poly[2] = 100; poly[3] =50;
  poly[4] = 100; poly[5] = 100; poly[6] = 50; poly[7] =100;

  int numb=0;
  int A=0;
  int start=1;

// END GRAPHICAL PARAMETER SPACE SECTION
field_angl = mem_ang;
return 0;
} // end map
```

```
  //  draw1.cpp
 // SCHEMATIC DRAWING OF LENSES ROUTINE
#include <iostream.h>
#include <stdio.h>
#include "c:\bc4\optical\draw.h"

#include "c:\bc4\optical\op.h"


    #include <graphics.h>
   #include <stdlib.h>
   #include <conio.h>
   #include <stdio.h>
   #include <string.h>
   #include <math.h>
   #include <string.h>
   #include <iostream.h>
 //  #include "mouse.h"
   // #include "mouse.h"

void put_point(int a, int b)
{
putpixel(a,b,15);

}

     void surff::draw(double scale)
  {
     point c(n.x,n.y);
     double angle=0,xxx=0,yyy=0;
     double max_yy=0;

     max_yy = read_maxy();
     xxx= (c.x) + (radius*scale);

     if (radius <0){
      // radius=-radius;
   xxx= (c.x) - (radius*scale);

    sketch(xxx,n.y, max_yy*scale, radius*scale);

   }
     else {
     sketch(xxx, n.y, max_yy*scale, radius*scale);}

   }

   int surff::sketch(int x,int  y, double maxy,double rad) {
   int cxx = x-rad;
   int first =0;
   int step = 5;
   // above step in degrees
   int ang2=0;
    double ang1=0;
   double step1 =maxy/20;
   double yy=0, xx=0;
   double ang=0;

   if (rad>0){
   yy=0;
   xx=0;
  /* double ang1=0;
  double step1 =maxy/20;
  int yy=0, xx=0;
  double ang;   */

  //step1 = 90-step1;
   double rad_ss = (3.141592654/180);
```

```
      for (int u=0; u < 20; u = u+1)
      {
        if (first==0) {
      xx = cxx;
      moveto(xx,y);
      first =1;
            }

      else         {

          if (first==1) {
            // angl = angl + step1;
            // xx = rad-((rad*cos(rad_ss*angl)));
            // yy = ((rad*sin(rad_ss*angl)));
            yy = yy + step1;

          xx = yy/rad;
           ang = (asin(xx))*(360/(2*3.141592654));
           angl =cos(rad_ss*(ang));
           xx =rad-(rad*angl);

         if (yy<maxy) {
            lineto((xx+(cxx)),y+yy);
           //    putpixel((cxx)+xx,y+yy,15);
                 s_dot.x =((cxx)+xx);
                 s_dot.y = (y+yy);

                  }
             }

             }
      }


yy=0;
first =0;
angl =0;


       for (int up=0; up < 20; up = up+1)
       {
      if (first==0) {
         xx = cxx;
         moveto(xx,y);
         first =1;
               }
      else {
           if (first==1) {
            // angl = angl - step1;
           yy = yy + step1;

           xx = yy/rad;
           ang = (asin(xx))*(360/(2*3.141592654));
           angl =cos(rad_ss*(ang));
           xx =rad-(rad*angl);

         //    yy = yy+ 3;
          //   xx = sqrt(((rad*rad)-(yy*yy)));

           //  yy = ((rad*sin(rad_ss*angl)));
           if (yy>-maxy){
           lineto((xx+(cxx)),y-yy);
         // putpixel((cxx)+xx,y-yy,15);
              n_dot.x= (xx+(cxx));
              n_dot.y =(y-yy);

           }
             }
             }
       }
```

```
  } // end if
  else   {

  double angl=0;
  first =0;

  yy=0, xx=0;
  rad=-rad;
  cxx = (x-rad);
  //rad=-rad;
  //step1 = 90-step1;
  double rad_ss = (3.141592654/180);


         for (int us=0; us < 20; us = us+1)
         {
        if (first==0) {
           xx = cxx;
           moveto(xx,y);
           first =1;
                 }
        else {
           if (first ==1) {
             //angl = angl + step1;
             yy = yy + step1;

             xx = yy/rad;
             ang = (asin(xx))*(360/(2*3.141592654));
             angl =cos(rad_ss*(ang));
             xx =rad-(rad*angl);

             //xx = ((rad*cos(rad_ss*angl)));
             //yy = ((rad*sin(rad_ss*angl)));
                 if (yy>-maxy){
              lineto((cxx-xx),y+yy);
               //   putpixel((cxx-xx),y+yy,15);
                 s_dot.x= ((cxx)-xx);
                 s_dot.y =(y+yy);

                 }
             }
               }
     }


yy=0;
first =0;
angl =0;


  for (int uw=0; uw < 20; uw = uw+1)
    {
       if (first==0) {
     xx = cxx;
      moveto(xx,y); first =1;
       }
       else {   if (first==1) {
            yy = yy + step1;
            xx = yy/rad;
            ang = (asin(xx))*(360/(2*3.141592654));
            angl =cos(rad_ss*(ang));
            xx =rad-(rad*angl);

              if (yy<maxy){
            lineto((cxx-xx),y-yy);
              // putpixel((cxx-xx),y-yy,15);

            n_dot.x= ((cxx)-xx);
            n_dot.y=(y-yy);
```

```
              }
            }
          }
           }

    } //end else

  return  0;
  }

   int close (surff &s11, surff &s22) {

    int test_bit=0;

    test_bit = (*&s22).linkval();

   if (test_bit ==1) {

    point pos = (*&s11).north_dot();
    point pos2 = (*&s22).north_dot();
    moveto(pos.x,pos.y);
    lineto(pos2.x,pos2.y);

    point pos3 = (*&s11).south_dot();
    point pos4 = (*&s22).south_dot();
    moveto(pos3.x,pos3.y);
    lineto(pos4.x,pos4.y);

 //   setfillstyle(SOLID_FILL,15);
 //   floodfill(pos.x+3,pos.y+5, 15);
//    setfillstyle(SOLID_FILL,15);
//    floodfill(pos.x+3,pos.y-5, 15);


 }
return  0;
}



int ray_tra(double  y, double  field, double  scale1,
double  move_x1, double  move_y1) {
 double  rad_ss = (3.141592654/180);
double  scale = scale1;
double  move_x = move_x1;
double  move_y = move_y1;
double  n_third_d[30],
  n2_third_d[30],
  sep_third_d[30],
  y_third_d[30],
  x_third_d[30];

double  angle_f = field;
double  len_op = y;
double  ray_angle3,ray_angle4;

if (object_at_infinity == 1) {
// ray_angle = sin(field_angl);
 ray_angle3 =   sin( (angle_f/(360/(PI*2)))  );
 }

 if (ray_angle3 != 1) {
 ray_angle4 = sqrt(1 - (ray_angle3*ray_angle3));
 }
 else { ray_angle4 =0; cout << "\a \n"; }


 int first_call =0;
 int dot_no =500;
//double len_op = h_third_d[2] -20;
```

```
   len_op = y;
   double ddd=0;
   double y1=0;
   double ynew=0,xnew=0;

// rrayptr[dot_no] = new rray(0,len_op,0,ray_angle4,ray_angle3,0);
  (*raypointer).newray(0,len_op,0,ray_angle4,ray_angle3,0);

double d=0;

    for ( int surf = 1; surf < (lastsurf+2) ; surf++)
{

  if (surf == 1)
     {
   if (object_at_infinity == 0) {
        flag_pupil = 0;
        flag_object = 1;
        (*raypointer).real_trace( (*surfaceptr[surf]) );
        }
   else {
        flag_pupil = 1;
        (*raypointer).real_back_trace( (*surfaceptr[surf]) );

        }
     }
     else
     {
     flag_object = 0;
     (*raypointer).real_trace( (*surfaceptr[surf]));
   n_third_d[surf] = n_third;
   n2_third_d[surf]= n2_third;
   sep_third_d[surf] = sep_third;
   y_third_d[surf] = y_third;
   x_third_d[surf] =x_third;
   d= (((sep_third_d[surf])*scale ) +d);
   ddd = d + ((x_third_d[surf])*scale);
   y1 = ((y_third_d[surf])*scale);

   if (first_call ==0) {
   moveto((ddd+move_x),y1+move_y);
   first_call =1; }

   if (surf ==2) {
   moveto((ddd+move_x),y1+move_y);
   xnew = ( (ddd+move_x- (30))) ;
   ynew = (  (   (30) * (tan(rad_ss*(field))) ) ) ;
   lineto(xnew,-ynew+move_y+y1);
      // moveto((ddd+move_x),y1+move_y);
    }

   lineto(ddd+move_x,y1+move_y);
    moveto(ddd+move_x,y1+move_y);


     }


} // end for loop

    (*imageptr).newimage(bfl);

  (*raypointer).real_trace( (*imageptr) );


    //   surfaceptr[23] = new surface(0, 0,
//  image_c, 50, 0 , 0, 0, bfl, 1,1,1, 0 ,1,1,1, 0);

 // (*rrayptr[dot_no]).real_trace( (*surfaceptr[23]), (*surfaceptr[2]) );
   n_third_d[surf] = n_third;
```

```
      n2_third_d[surf]= n2_third;
      sep_third_d[surf] = sep_third;
      y_third_d[surf] = y_third;
      x_third_d[surf] =x_third;
      d= (((sep_third_d[surf])*scale ) +d);
      ddd = d + ((x_third_d[surf])*scale);
      y1 = ((y_third_d[surf])*scale);
      lineto(ddd+move_x,y1+move_y);
      moveto(ddd+move_x,y1+move_y);

    //    lineto((move_x-(40*scale)),move_y);

   // DESTROY RAY OBJECT

      // delete rrayptr[dot_no];
   // FINISH DESTRUCTION OF RAY OBJECT

    // MISSOUT MISTAKE ON NEXT LINE
    //   delete surfaceptr[dot_no];
   // MISS OUT ABOVE MISTAKE - MENT TO BE RAY I THINK

       //   delete surfaceptr[23];

      return 0;

       } // end ray_tra




   int draw1(double field1,int fan,int fan2,double scale, double move_x, double mov
e_y){
   double yyyy=0;
   double field=0;
   int ray_tra(double y, double field, double scale1,
    double move_x1, double move_y1);
   double PI = (3.141592654);

   double n_third_d[20],n2_third_d[20],
   y_third_d[20], u_third_d[20], u2_third_d[20],
   c_third_d[20], d_third_d[20], d2_third_d[20],
   sep_third_d[20], h_third_d[20], x_third_d[20];

       //   pray ray88(rad_pupil,0,0,0,0);
      (*praypointer).newpray(rad_pupil,0,0,0,0);

    for (int surf = 1; surf < (lastsurf+2) ; surf++)
   {
     if (surf == 1)
        {
        flag_pupil = 1;
        (*praypointer).back_trace( (*surfaceptr[surf]), (*surfaceptr[surf+1]) );

        }
        else
        {
        (*praypointer).traceth( (*surfaceptr[surf]), (*surfaceptr[surf+1]) );
        }
      n_third_d[surf] = n_third;
      n2_third_d[surf]= n2_third;
      y_third_d[surf] = y_third;
      u_third_d[surf] = u_third;
      u2_third_d[surf] = u2_third;
      c_third_d[surf] = c_third;
      d_third_d[surf] = d_third;
      d2_third_d[surf] = d2_third;
      sep_third_d[surf] = sep_third;
      h_third_d[surf] = h_third;
   } // end for loop
```

```
//double scale=1;
    //   if (y_third_d[2] >100) {
    //   scale = 100/y_third_d[2];}

   // if (rad_pupil <70) {
    //   scale = (70/rad_pupil);}


 char c;
 int shift = 0;

/*
  if (graphics == 1){
if (back_graph == 1) {
setgraphmode(getgraphmode()); }
   }



  if (graphics == 0) {

 setviewport(50, 50, getmaxx()-50, getmaxy()-50, 1);
 int gdriver = DETECT, gmode, errorcode;
 initgraph(&gdriver, &gmode, "");
 errorcode = graphresult();

 if (errorcode != grOk)  /* an error occurred */
  //   {
  //      printf("Graphics error: %s\n", grapherrormsg(errorcode));
   //     printf("Press any key to halt:");
   //     getch();
   //     exit(1);             /* return with error code */
   //  }
//  graphics =1;

// } // END IF GRAPHICS ALREADY DONE
   // setcolor(1);
 //    setbkcolor(1);

 //   cleardevice();



   int i=0;

   int surf1_x=0;
   int surf1_y=0;
   int surf2_x=0;
   int surf2_y=0;
   int surf2_yy=0;
   int test_bit=0;
   int bit=0, bit2;
   double d=0,max_d=0, y1=0;
  // double move_x=100, move_y=200;
   double y=0;
   double h1=0, rad1=0,sep1=0;

   // moveto(0,200);
//   circle(move_x, move_y, 20);
  //  lineto(400,200);

     surff *ssurfptr[30];


  for ( surf = 2; surf < (lastsurf+2) ; surf++)
  {
     h1 = h_third_d[surf];
     if (c_third_d[surf] == 0) {
```

```
    rad1 =100000000;}
    else {
    rad1 = (1/c_third_d[surf]);}

    sep1 = sep_third_d[surf];
    if (n_third_d[surf] >1 ){

     bit =1;
    }

     if (n_third_d[surf] <-1 ){

     bit =1;
    }

   ssurfptr[surf] = new surff(h1,point(0,0), point(0,0),rad1,bit,sep1);
   d=( (sep1*scale) +d);
   if (d>max_d) {max_d=d;}
   ssurfptr[surf]->move(move_x+d,move_y);
   setcolor(15);
   ssurfptr[surf]->draw(scale);


  close ((*ssurfptr[surf-1]), (*ssurfptr[surf]));

  /*
     if ((n_third_d[surf] >1 )&(sep1>0)){
  setfillstyle(SOLID_FILL,15);
  floodfill(move_x+d-3,-move_y, 15);
     }
    else { if  ((n_third_d[surf] >1 )&(sep1<0)){
  setfillstyle(SOLID_FILL,15);
  floodfill(move_x+d+3,-move_y, 15);}

     }
     */

  bit =0;
  bit2=0;
 }


 setcolor(14);
moveto(move_x,move_y);

  // CALCULATES CORRECT BFL
     flag_paraxial_y=0;
     efl1();
  // CALCULATES CORRECT BFL


  field = 0;


  y = +(rad_pupil-1);
  ray_tra(y, field, scale,move_x, move_y);

  y = -(rad_pupil-1);
  field =0;
  ray_tra(y, field, scale,move_x, move_y);

  y = 0;
  field =0;
  ray_tra(y, field, scale,move_x, move_y);

 if (fan ==1){

 // fan of rays

  double stepf = rad_pupil/6;
```

```
      for ( y = 0; y < (rad_pupil) ; y = y+stepf)
   {
    field =0;
    ray_tra(y, field, scale,move_x, move_y);

   }

   for ( y = 0; y > (-rad_pupil) ; y = y-stepf)
   {
    field =0;
    ray_tra(y, field, scale,move_x, move_y);

   }


   // end fan of rays
   } // end if fan ==1

   if (fan2 ==1){

   // fan of rays

    double stepf = rad_pupil/6;
      for ( y = 0; y < (rad_pupil) ; y = y+stepf)
   {
//  field = -field_angl;
   field =field1;
    ray_tra(y, field, scale,move_x, move_y);

   }

   for ( y = 0; y > (-rad_pupil) ; y = y-stepf)
   {

    ray_tra(y, field, scale,move_x, move_y);

   }


   // end fan of rays
   } // end if fan2 ==1

   setcolor(10);

// field = -field_angl;
 field=field1;
 y = -(rad_pupil-1);
   ray_tra(y, field, scale,move_x, move_y);

// field = -field_angl;
 y= rad_pupil-1;
   ray_tra(y, field, scale,move_x, move_y);

y =0;
   // field = -field_angl;

 ray_tra(y, field, scale,move_x, move_y);

    // DELETE SURFACES
   for ( surf = 2; surf < (lastsurf+2) ; surf++)
   {
   delete   ssurfptr[surf];

   }
    //END DELETE SURFACES
   // delete s1;

    //       c= getch();
   //    restorecrtmode();
```

```
return 0;
}
```

```
// DES8B.cpp
   // version for use in time.cpp and time2.cpp and space.cpp
   // DOUBLET DESIGN -  MY OWN METHOD OF COLOUR CORRECTION
   // ATTEMPT TO BALANCE FIRST ORDER
   // AXIAL COLOUR ABERRATIONS - CONTRIBUTIONS
   // FORM FIRST  THREE SURFACES ARE CANCELLED 100 PERCENT BY
   // LAST SURFACE OF DOUBLET
   // see elsewhere, ie des8c for 90 percent color correction
 #include "c:\bc4\optical\op.h"



 int des8b()  {

 double  req2_c=0, req3_c=0,
        i1=0, dn=0,i2_air=0,n_1=0,
        dn_n=0, left=0, right=0, i2=0,
        req_c=0,r1=0,r2=0,n_sq_1=0,
        d1=0,n_sq=0,sum1=0,newd=0,sum2=0,
        main_c=0, ddd1=0,d11=0, d22=0,root=0,
        c2=0,c3=0,c4=0,d=0, h2=0,d2=0,newc1=0,newc3=0,h22=0,
        ray_angle6=0,pupilyy=0,d3=0,d4=0,
        root2=0,h33=0,h44=0,c_pet=0,save=0,
        sumpet=0;


 char bit144[30];
 char bit244[30];
 int bit14, bit24;
 char * c444;
 int thick;
 int nos6 =6, nos7 =7;
 char chh;

 double  n_third_d[20],n2_third_d[20],
 y_third_d[20], u_third_d[20], u2_third_d[20],
 c_third_d[20], d_third_d[20], d2_third_d[20], sep_third_d[20],
 ri2back_d[20], ri3back_d[20];


 int surf ;
 flag_paraxial_y = 1;


 pos = 1;
   // third2();

   //pray ray7(rad_pupil,0,0,0,0);
    //   prayptr[73] = new pray(rad_pupil,0,0,0,0);
 (*praypointer).newpray(rad_pupil,0,0,0,0);

  for ( surf = 1; surf < (lastsurf+2) ; surf++)
 {
   if (surf == 1)
      {
      flag_pupil = 1;
      (*praypointer).back_trace( (*surfaceptr[surf]), (*surfaceptr[surf+1]) );

      }
      else
      {
      (*praypointer).traceth( (*surfaceptr[surf]), (*surfaceptr[surf+1]) );
      }
    n_third_d[surf] = n_third;
    n2_third_d[surf]= n2_third;
    y_third_d[surf] = y_third;
    u_third_d[surf] = u_third;
    u2_third_d[surf] = u2_third;
    c_third_d[surf] = c_third;
    d_third_d[surf] = d_third;
    d2_third_d[surf] = d2_third;
    sep_third_d[surf] = sep_third;
    ri2back_d[surf] = ri2back;
    ri3back_d[surf] = ri3back;

 } // end for loop
     //   delete praypt4r[73];

 newcl = c_third_d[ss3];

    newc3 = c_third_d[ss5] ;
   c3 = newc3;



   req_c = 0;

   dn = ri3back_d[ss3] -ri2back_d[ss3];
   dn_n = (dn/n2_third_d[ss3]);

  il = (n2_third_d[ss3]*( (y_third_d[ss3]*newcl) + u2_third_d[ss3]) );

   double  req_cl = (y_third_d[ss3]*il)*dn_n;

  // end  find req_c for surf 1

  // find req_c for surf 2
   double  newc2 = c_third_d[ss4];

   dn = ri3back_d[ss3] -ri2back_d[ss3];
   dn_n = (dn/n2_third_d[ss3]);

  il = (n2_third_d[ss4]*( (y_third_d[ss4]*newc2) + u2_third_d[ss4]) );

   double  req_c2 = (y_third_d[ss4]*il)*dn_n;

  // end find req_c for surface 2

   double  total_req_c =  req_cl - req_c2 ;
   total_req_c = -total_req_c;
// section to make colour coeff 4 equal to minus total_req_c

  // end surface 3 colour coefficient

   dn = ri3back_d[ss5] -ri2back_d[ss5];
   if (dn == 0){
   dn_n =0; }
   else {
   dn_n = (dn/n2_third_d[ss5]);}

   il = (n2_third_d[ss5]*( (y_third_d[ss5]*c3) + u2_third_d[ss5]) );

   left = (y_third_d[ss5]/y_third_d[ss6])*il;

   if (dn_n ==0){right=0;}
   else {
   right = ( (total_req_c/y_third_d[ss6])*(1/dn_n) );}

   save = right;


   i2 = left-right;



  left = (il-i2)/(n2_third_d[ss5]*y_third_d[ss6]);
   right = ((y_third_d[ss5])*c3)/
    ( (y_third_d[ss6]));

   c4 = (right-left);
```

```
(*surfaceptr[ss6]).newcll(c4);
 return  0;
 }
```

```
  // DES5.cpp
  // Meniscus Design - for use with time.cpp

#include "c:\bc4\optical\op.h"



int des5()  {
double n_sq =0, n_sq_1=0, r2=0, r1=0, d1=0, sum2=0,
       sum1=0,dddl=0,d11=0, d22=0,
       root=0,c2=0,d=0, h2=0,d2=0,newc1=0,h22=0,
       ray_angle6=0,pupilyy=0,d3=0,d4=0,
       root2=0,h33=0,h44=0,
       c_pet=0, sumpet=0;


char bit144[30];
char bit244[30];
int bit14=0, bit24=0,thick=0,
    nos6 =6, nos7 =7;

char * c444;

char chh;

double n_third_d[20],n2_third_d[20],
y_third_d[20], u_third_d[20], u2_third_d[20],
c_third_d[20], d_third_d[20], d2_third_d[20], sep_third_d[20];

int surf=0 ;
flag_paraxial_y = 1;

pos = 1;

  if (object_at_infinity == 1) {

    ray_angle6 =   tan( ( field_angl/(360/(2*PI)) )  );
  }


  if (object_at_infinity == 0) {
    if (object_dist !=0){
    ray_angle6 = (-(0-object_h1)/object_dist);
    }
  }

   //pray ray7(rad_pupil,0,0,0,0);
    //   prayptr[61] = new pray(rad_pupil,0,0,0,0);
  (*praypointer).newpray(rad_pupil,0,0,0,0);

 for ( surf = 1; surf < (lastsurf+2) ; surf++)
{
  if (surf == 1)
     {
     flag_pupil = 1;
     (*praypointer).back_trace( (*surfaceptr[surf]), (*surfaceptr[surf+1]) );

     }
     else
     {
     (*praypointer).traceth( (*surfaceptr[surf]), (*surfaceptr[surf+1]) );
     }
   n_third_d[surf] = n_third;
   n2_third_d[surf]= n2_third;
   y_third_d[surf] = y_third;
   u_third_d[surf] = u_third;
   u2_third_d[surf] = u2_third;
   c_third_d[surf] = c_third;
```

```
    d_third_d[surf] = d_third;
    d2_third_d[surf] = d2_third;
    sep_third_d[surf] = sep_third;


} // end for loop

 // delete prayptr[61];

   newc1 = c_third_d[3];
     if (newc1!=0){
     r1 = 1/newc1;}
     else {r1=10000000;}

   n_sq = (n2_third_d[3] * n2_third_d[3]);
   n_sq_1 = (  (n2_third_d[3]-1) * (n2_third_d[3]-1) );

   d1= sep_third_d[4];
   if( ( (n_sq * r1) - (d1* (n_sq_1) ))!=0 ){

   sum2 = d1*(  (r1*(n_sq -1)) - (d1 * n_sq_1) ) /
     ( (n_sq * r1) - (d1* (n_sq_1) ) );
   }

    r2 =  r1 - sum2;

    if(r2 !=0){
    c2 = 1/r2;}

       (*surfaceptr[4]).newc11(c2);


    return 0;

    }
```

```
  // DES4.cpp
  // function to achromatise and zero the petsvil sum on
  // field flatening Lens
  // note this routine is for use with time.cpp
  // and is used in during real time graphical
  // plot design

#include "c:\bc4\optical\op.h"


double a1=0, sum44=0,ddd1=0;

char bit144[30];
char bit244[30];

int bit14=0, bit24=0;

char * c444;

int des4()  {

int thick=0, surf=0;

double d11=0, d22=0,root=0,c2=0,
       d=0, h2=0,d2=0,newc1=0,h22=0,
       ray_angle6=0,pupilyy=0,d3=0,d4=0,
       root2=0,h33=0,h44=0,c_pet=0,sumpet=0;

int nos6 =6, nos7 =7;

char chh;

double n_third_d[20],n2_third_d[20],
y_third_d[20], u_third_d[20], u2_third_d[20],
c_third_d[20], d_third_d[20], d2_third_d[20], sep_third_d[20];

flag_paraxial_y = 1;
pos = 1;

  if (object_at_infinity == 1) {
  ray_angle6 =   tan( ( field_angl/(360/(2*PI)) )   );
  }


  if (object_at_infinity == 0) {
   if (object_dist !=0){
    ray_angle6 = (-(0-object_h1)/object_dist);
   }
  }

   //pray ray7(rad_pupil,0,0,0,0);
   // prayptr[60] = new pray(rad_pupil,0,0,0,0);
(*praypointer).newpray(rad_pupil,0,0,0,0);

 for ( surf = 1; surf < (lastsurf+2) ; surf++)
{
  if (surf == 1)
     {
     flag_pupil = 1;
     (*praypointer).back_trace( (*surfaceptr[surf]), (*surfaceptr[surf+1]) );

     }
     else
     {
     (*praypointer).traceth( (*surfaceptr[surf]), (*surfaceptr[surf+1]) );
     }
   n_third_d[surf] = n_third;
   n2_third_d[surf]= n2_third;
   y_third_d[surf] = y_third;
```

```
    u_third_d[surf] = u_third;
    u2_third_d[surf] = u2_third;
    c_third_d[surf] = c_third;
    d_third_d[surf] = d_third;
    d2_third_d[surf] = d2_third;
    sep_third_d[surf] = sep_third;


 } // end for loop

  // delete prayptr[60];

   newcl = c_third_d[s6];

   if (pet !=0){
   c_pet = (1/pet);}

 if ( (-n2_third_d[s6]-1)!=0){
sumpet = (-n2_third_d[s6]*c_pet)/(-n2_third_d[s6]-1);
 }
   c2 = ((sumpet+newcl));
      a1 =( (n2_third_d[s6])* ( (y_third_d[s6]*newcl) + u2_third_d[s6]) );

   if (n2_third_d[s6]!=0){
   sum44 =   (4*c2 *a1 * (y_third_d[s6]/n2_third_d[s6]) );
   }

   root = (sqrt(  (u2_third_d[s6] * u2_third_d[s6]) + (sum44)) );

   if (c2!=0){
   h2 = ( (-u2_third_d[s6] + root) /(2*c2));
   }


   d3 = -( (y_third_d[s6] -h2)/u2_third_d[s6] );

   if (c2!=0){
   h22 = ( (-u2_third_d[s6] - root) /(2*c2));
   }

   if (u2_third_d[s6] !=0){
   d = -( (y_third_d[s6]-h22)/ u2_third_d[s6] );
   }

   if (t2_run==1){
   cout << "\a";
   d3=d;
   }

 ddd1 = d3;

 ddd1 = d3;


 // end print alternatives section

       (*surfaceptr[s6]).newcl1(newcl);
       (*surfaceptr[s7]).newcl1(c2);
       (*surfaceptr[s7]).newd22(d3);


    return 0;

    }
```

```
// time.cpp
// Function to allow Graphical Design
// Real Time Aberration plots are shown and
// updated as the user changes the lens Parameters

 #include "c:\bc4\optical\op.h"
 #include <stdio.h>
 #include <graphics.h>
 #include <stdlib.h>
 #include <conio.h>
 #include "mouse.h"

 int time()  {
   extern int schem2(double ,double ,double );

   int surfx=2,axial =2, scale_sp =10,loop_no=1,map1=0,surface4 =4,
       plane_bit=0,lens =0,counter=0,field_des=0,
       des88=0, des88b=0, des88c=0,plot_3 =0,su1 =1, su2=2, su3=3, su4=4,
       su5=5, su6=6, su7=7,su8=8, su9=9, su10 =10, su11 =11,
       su12=12, su13 = 13,su14 = 14, su15 = 15,
       sphero_3 =0, solve_h=0,scale_change =0,sphero=0,
       d2_lab=10,mouse_button=0, number_times=0, x=0, y=0,
       right=0, left=0, redraw=0,
       scale_fe=10,scale_fa2=10,scale_fa4=10,scale_fc22=10,
       scale_fa6=10, scale_fc =10, scale_fd =10,plot_bit=0,
       thick=0,bit1=0,bit2=0,bit14=0, bit24=0, bit16=0,
       bit26=0,nos6 =6, surf=0,nos7 =7,asp_no =4,rubout=0,keypress=0;

   int gdriver = DETECT, gmode, errorcode;
   int a_asp_1=0;

   char su1a[25];
   char su2a[25];
   char su3a[25];
   char surfxa[25];
   char su4a[25];
   char su5a[25];
   char sup[25];
   char sup2[25];
   char sup3[25];
   char su6a[25];
   char su7a[25];
   char su8a[25];
   char su9a[25];
   char su10a[25];
   char *su11a;
   char *su12a;
   char *su13a;
   char *su14a;

   double memsx=0,c1_save=0, r1_now=0, r3_shift=0, c2_save=0,
     c3_save=0, r3_now=0, r5_shift=0, c4_save=0,
     mem_x=0, mem_y=0, slide_x=0, slide_y=0,
     c3_now=0, c1_now=0, ab_plot=0, normalize=0,
     small_x=0, small_y=0, big_x=0, big_y=0,
     xx_now=0,yy_now=0;
   memsx = field_angl;

  int data_x=0, data_y=0, virt=0, virt2=0;

   double fno=0, ddd50=0,d2_inc=0, d3_inc=0,
     bfl_mem=0, c11_inc=0.00001, c22_inc=0.00001,
     e_inc=0,a2_inc=0,a4_inc=0, a6_inc=0, c1_inc=0.00001, c2_inc=0.00001,
     def_inc=0,pupil_inc=0,ddd1=0,
     midx=0,midy=0,stop3=0,d_inc=0,
     c11=0, c22=0,ccc1=0, ccc2=0,
     read_stop=0,d11=0, d22=0, new_stop=0,
     root=0,c2=0,d=0, h2=0,d2=0,newc1=0,h22=0,
     ray_angle6=0,pupilyy=0,d3=0,d4=0,
     root2=0,h33=0,h44=0,c_pet=0,cmain_inc=0,
     new_stop1=0,
     pet=0, sumpet=0,mem_ang=0;

int scale_cm=10;
mag_flag =1;
mag =1000;
int poly[8];


char * a444;
char *a666b;
char a666[25];
char * a_asp;
char * c444;
char * new_stop2;
int ploy[8];
char chh;
char bit144[30];
char bit166[30];
char bit244[30];
char bit266[30];
char mes2[] = "Mag";
char mes1[] = "Change";
char mes3[] = "Lens";
char mes4[] = "";
char mes5[] = "";
char mes6[] = "Men.";
char mes9[] = "+";
char mes8[] = "-";
char mm[] ="mm";
char q[] = "X 2";
char qq[] = "/ 2";
char t[] ="X 2";
char r[] ="/ 2";
char tt[]="X10";
char rr[]="/10";
char ll[] = "ON/OFF";
char a41[] ="+ ve";
char a42[] ="- ve";
char welcome[] = "INTER-OPT";
double n_third_d[20],n2_third_d[20],
y_third_d[20], u_third_d[20], u2_third_d[20],
c_third_d[20], d_third_d[20], d2_third_d[20], sep_third_d[20];

// TRANSFER

double c111_inc =1;
double c333_inc=1;
double scale_p=1;

double min_ab=10000;
double min_x=0, min_y=0;
double min_r1=0, min_r3=0;

double pos_x=370;
double pos_y=138;
double mag_ab = 3;
int now=1;
double array_x[10][10];
double array_y[10][10];
int shift_yy=70;
// TRANSFER

flag_paraxial_y = 1;
pos = 1;
mem_ang = field_angl;

  if (object_at_infinity == 1) {
  ray_angle6 =   tan( ( field_angl/(360/(2*PI)) )   );
  }
  if (object_at_infinity == 0) {
   if (object_dist !=0){
   ray_angle6 = (-(0-object_h1)/object_dist);
   }
  }

dont_clear =1;

  if (graphics == 1){
if (back_graph == 1) {
setgraphmode(getgraphmode()); }
   }



  if (graphics == 0) {

 setviewport(50, 50, getmaxx()-50, getmaxy()-50, 1);
 int gdriver = DETECT, gmode, errorcode;
 initgraph(&gdriver, &gmode, "");
 errorcode = graphresult();

 if (errorcode != grOk)  /* an error occurred */
 {
    printf("Graphics error: %s\n", grapherrormsg(errorcode));
    printf("Press any key to halt:");
    getch();
    exit(1);             /* return with error code */
 }
 graphics =1;

} // END IF GRAPHICS ALREADY DONE

  // title section
setfillstyle(SOLID_FILL, 12);
midx = 300;
midy = 30;
bar(midx-90, midy-24, midx+83,midy+10);
// end title section

mouse_initialize();
setfillstyle(SOLID_FILL, 9);
outtextxy(5,55,mes2);
midx = 20;
midy = 100;
bar(midx-20, midy-20, midx+18,midy+11);
outtextxy(10,95,t);

setfillstyle(SOLID_FILL, 9);
midx = 20;
midy = 140;
bar(midx-20, midy-20, midx+18,midy+11);
outtextxy(10,135,r);

// END SCALE BUTTON


// BOX TO TURN ON /OFF 3RD ORDER PLOTS
setfillstyle(SOLID_FILL, 11);
midx = 240;
midy = 45;
bar(midx-10, midy-4, midx+10,midy+4);


// END BOX TO TURN ON/OFF 3RD ORDER PLOTS

// BUTTONS FOR CORRECTOR PLATE

setfillstyle(SOLID_FILL, 12);
```

```
midx = 140;
midy = 280;
bar(midx-20, midy-6, midx+20,midy+6);
outtextxy(125,278,a41);


setfillstyle(SOLID_FILL, 12);
midx = 95;
midy = 280;
bar(midx-20, midy-6, midx+20,midy+6);
outtextxy(80,278,a42);

setfillstyle(SOLID_FILL, 9);
midx = 140;
midy = 298;
bar(midx-20, midy-6, midx+20,midy+6);
outtextxy(125,296,a41);

setfillstyle(SOLID_FILL, 9);
midx = 95;
midy = 298;
bar(midx-20, midy-6, midx+20,midy+6);
outtextxy(80,296,a42);

setfillstyle(SOLID_FILL, 6);
midx = 140;
midy = 316;
bar(midx-20, midy-6, midx+20,midy+6);

setfillstyle(SOLID_FILL, 6);
midx = 95;
midy = 316;
bar(midx-20, midy-6, midx+20,midy+6);

setfillstyle(SOLID_FILL, 9);
midx = 140;
midy = 334;
bar(midx-20, midy-6, midx+20,midy+6);
outtextxy(125,332,a41);

setfillstyle(SOLID_FILL, 9);
midx = 95;
midy = 334;
bar(midx-20, midy-6, midx+20,midy+6);
outtextxy(80,332,a42);

setfillstyle(SOLID_FILL, 6);
midx = 140;
midy = 352;
bar(midx-20, midy-6, midx+20,midy+6);

setfillstyle(SOLID_FILL, 6);
midx = 95;
midy = 352;
bar(midx-20, midy-6, midx+20,midy+6);

setfillstyle(SOLID_FILL, 9);
midx = 140;
midy = 370;
bar(midx-20, midy-6, midx+20,midy+6);
outtextxy(125,368,a41);

setfillstyle(SOLID_FILL, 9);
midx = 95;
midy = 370;
bar(midx-20, midy-6, midx+20,midy+6);
outtextxy(80,368,a42);

setfillstyle(SOLID_FILL, 6);
midx = 140;
```

```
 midy = 388;
 bar(midx-20, midy-6, midx+20,midy+6);

 setfillstyle(SOLID_FILL, 6);
 midx = 95;
 midy = 388;
 bar(midx-20, midy-6, midx+20,midy+6);


 setfillstyle(SOLID_FILL, 9);
 midx = 140;
 midy = 406;
 bar(midx-20, midy-6, midx+20,midy+6);
 outtextxy(125,404,a41);

 setfillstyle(SOLID_FILL, 9);
 midx = 95;
 midy = 406;
 bar(midx-20, midy-6, midx+20,midy+6);
 outtextxy(80,404,a42);


 // END BUTTONS FOR CORRECTOR PLATE


// new 94

 setfillstyle(SOLID_FILL, 13);
 midx = 140;
 midy = 460;
 bar(midx-20, midy-6, midx+20,midy+6);
 outtextxy(125,457,a41);

 setfillstyle(SOLID_FILL, 13);
 midx = 95;
 midy = 460;
 bar(midx-20, midy-6, midx+20,midy+6);
 outtextxy(80,457,a42);
  // new 94

 setfillstyle(SOLID_FILL, 13);
 midx = 140;
 midy = 475;
 bar(midx-20, midy-6, midx+20,midy+6);
 outtextxy(125,472,a41);

 setfillstyle(SOLID_FILL, 13);
 midx = 95;
 midy = 475;
 bar(midx-20, midy-6, midx+20,midy+6);
 outtextxy(80,472,a42);


// new 94
 setfillstyle(SOLID_FILL, 13);
 midx = 140;
 midy = 424;
 bar(midx-20, midy-6, midx+20,midy+6);
 outtextxy(125,422,a41);

 setfillstyle(SOLID_FILL, 13);
 midx = 95;
 midy = 424;
 bar(midx-20, midy-6, midx+20,midy+6);
 outtextxy(80,422,a42);

 setfillstyle(SOLID_FILL, 2);
 midx = 140;
 midy = 439;
 bar(midx-20, midy-6, midx+20,midy+6);
```

```
outtextxy(125,437,a41);

setfillstyle(SOLID_FILL, 2);
midx = 95;
midy = 439;
bar(midx-20, midy-6, midx+20,midy+6);
outtextxy(80,437,a42);



// BUTTONS FOR FIELD CORRECTOR

setfillstyle(SOLID_FILL, 12);
midx = 400;
midy = 280;
bar(midx-20, midy-6, midx+20,midy+6);
outtextxy(385,278,a41);


setfillstyle(SOLID_FILL, 12);
midx = 355;
midy = 280;
bar(midx-20, midy-6, midx+20,midy+6);
outtextxy(340,278,a42);

setfillstyle(SOLID_FILL, 9);
midx = 400;
midy = 298;
bar(midx-20, midy-6, midx+20,midy+6);
outtextxy(385,296,a41);

setfillstyle(SOLID_FILL, 9);
midx = 355;
midy = 298;
bar(midx-20, midy-6, midx+20,midy+6);
outtextxy(340,296,a42);

setfillstyle(SOLID_FILL, 6);
midx = 400;
midy = 316;
bar(midx-20, midy-6, midx+20,midy+6);

setfillstyle(SOLID_FILL, 6);
midx = 355;
midy = 316;
bar(midx-20, midy-6, midx+20,midy+6);

setfillstyle(SOLID_FILL, 9);
midx = 400;
midy = 334;
bar(midx-20, midy-6, midx+20,midy+6);
outtextxy(385,332,a41);

setfillstyle(SOLID_FILL, 9);
midx = 355;
midy = 334;
bar(midx-20, midy-6, midx+20,midy+6);
outtextxy(340,332,a42);

setfillstyle(SOLID_FILL, 6);
midx = 400;
midy = 352;
bar(midx-20, midy-6, midx+20,midy+6);

setfillstyle(SOLID_FILL, 6);
midx = 355;
midy = 352;
bar(midx-20, midy-6, midx+20,midy+6);
```

```
setfillstyle(SOLID_FILL, 9);
midx = 400;
midy = 370;
bar(midx-20, midy-6, midx+20,midy+6);
outtextxy(385,368,a41);

setfillstyle(SOLID_FILL, 9);
midx = 355;
midy = 370;
bar(midx-20, midy-6, midx+20,midy+6);
outtextxy(340,368,a42);

setfillstyle(SOLID_FILL, 6);
midx = 400;
midy = 388;
bar(midx-20, midy-6, midx+20,midy+6);

setfillstyle(SOLID_FILL, 6);
midx = 355;
midy = 388;
bar(midx-20, midy-6, midx+20,midy+6);


setfillstyle(SOLID_FILL, 9);
midx = 400;
midy = 406;
bar(midx-20, midy-6, midx+20,midy+6);
outtextxy(385,404,a41);

setfillstyle(SOLID_FILL, 9);
midx = 355;
midy = 406;
bar(midx-20, midy-6, midx+20,midy+6);
outtextxy(340,404,a42);

setfillstyle(SOLID_FILL, 1);
midx = 450;
midy = 335;
bar(midx-26, midy-61, midx+89,midy+113);

setfillstyle(SOLID_FILL, 1);
midx = 210;
midy = 280;
bar(midx-40, midy-6, midx+100,midy+165);

// on off box

// field corr button
setfillstyle(SOLID_FILL, 9);
midx = 580;
midy = 275;
bar(midx-20, midy-20, midx+18,midy+11);
outtextxy(563,270,mes3);
// end field corr button

// BUTTONS FOR ASPHERIC SCALING

     // SCALE CURVEATURE INC
setfillstyle(SOLID_FILL, 9);
midx = 38;
midy = 270;
bar(midx-6, midy-6, midx+6,midy+6);

setfillstyle(SOLID_FILL, 9);
midx = 9;
midy = 270;
bar(midx-6, midy-6, midx+6,midy+6);
// END SCALE CURVE INC

    // SCALE A2
```

```
 setfillstyle(SOLID_FILL, 13);
 midx = 9;
 midy = 460;
 bar(midx-6, midy-6, midx+6,midy+6);

 setfillstyle(SOLID_FILL, 13);
 midx = 38;
 midy = 460;
 bar(midx-6, midy-6, midx+6,midy+6);
 // END SCALE A2

// SCALE E
 setfillstyle(SOLID_FILL, 13);
 midx = 9;
 midy = 475;
 bar(midx-6, midy-6, midx+6,midy+6);

 setfillstyle(SOLID_FILL, 13);
 midx = 38;
 midy = 475;
 bar(midx-6, midy-6, midx+6,midy+6);
 // END SCALE E



 // SCALE A4
 setfillstyle(SOLID_FILL, 13);
 midx = 9;
 midy = 310;
 bar(midx-6, midy-6, midx+6,midy+6);

 setfillstyle(SOLID_FILL, 13);
 midx = 38;
 midy = 310;
 bar(midx-6, midy-6, midx+6,midy+6);
 // END SCALE A4

 //SCALE A6
 setfillstyle(SOLID_FILL, 2);
 midx = 9;
 midy = 350;
 bar(midx-6, midy-6, midx+6,midy+6);

 setfillstyle(SOLID_FILL, 2);
 midx = 38;
 midy = 350;
 bar(midx-6, midy-6, midx+6,midy+6);
 // END SCALE A6

 // SEPERATION SCALE
 setfillstyle(SOLID_FILL, 12);
 midx = 580;
 midy = 390;
 bar(midx-20, midy-6, midx+18,midy+6);

 setfillstyle(SOLID_FILL, 12);
 midx = 580;
 midy = 414;
 bar(midx-20, midy-6, midx+18,midy+6);
 //END SEPERATION SCALE
 // END OF BUTTONS FOR ASPHERIC SCALING

 // BUTTONS FOR MAKSUTOV MENISCUS

 setfillstyle(SOLID_FILL, 9);
 midx = 20;
 midy = 200;
 bar(midx-20, midy-20, midx+18,midy+11);
```

```
setfillstyle(SOLID_FILL, 9);
midx = 6;
midy = 225;
bar(midx-6, midy-6, midx+6,midy+6);


setfillstyle(SOLID_FILL, 9);
midx = 30;
midy = 225;
bar(midx-6, midy-6, midx+6,midy+6);




// END BUTTONS FOR MAKSUTOV MENISCUS

// BUTTONS FOR MOVING STOP

setfillstyle(SOLID_FILL, 12);
midx = 400;
midy = 426;
bar(midx-20, midy-6, midx+20,midy+6);
outtextxy(385,424,a41);

setfillstyle(SOLID_FILL, 12);
midx = 355;
midy = 426;
bar(midx-20, midy-6, midx+20,midy+6);
outtextxy(340,424,a42);

// END BUTTONS FOR MOVING STOP


// BUTTONS FOR CHANGING CURVATURE OF PRIMARY MIRROR

setfillstyle(SOLID_FILL, 10);
midx = 400;
midy = 440;
bar(midx-20, midy-6, midx+20,midy+6);
outtextxy(385,438,a41);


setfillstyle(SOLID_FILL, 10);
midx = 355;
midy = 440;
bar(midx-20, midy-6, midx+20,midy+6);
outtextxy(340,438,a42);


// END BUTTONS FOR CHANGING CURVATURE OF PRIMARY MIRROR


setfillstyle(SOLID_FILL, 10);
midx = 560;
midy = 440;
bar(midx-6, midy-6, midx+6,midy+6);

setfillstyle(SOLID_FILL, 10);
midx = 589;
midy = 440;
bar(midx-6, midy-6, midx+6,midy+6);

setfillstyle(SOLID_FILL, 4);
midx = 575;
midy = 440;
bar(midx-8, midy-6,midx+7,midy+6);

itoa(scale_cm,sup2,10);
outtextxy(567,437,sup2);
```

```
// BUTTONS TO INCREMENT DEFOCUS

setfillstyle(SOLID_FILL, 9);
midx = 20;
midy = 7;
bar(midx-15, midy-6, midx+15,midy+6);

setfillstyle(SOLID_FILL, 9);
midx = 20;
midy = 35;
bar(midx-15, midy-6, midx+15,midy+6);

// END BUTTONS TO INCREMENT DEFOCUS


// BUTTON FOR 3 COLOUR PLOTS
setfillstyle(SLASH_FILL, 9);
midx = 90;
midy = 160;
bar(midx-10, midy-6, midx+10,midy+6);

setfillstyle(SLASH_FILL, 14);
midx = 110;
midy = 160;
bar(midx-10, midy-6, midx+10,midy+6);

setfillstyle(SLASH_FILL, 12);
midx = 130;
midy = 160;
bar(midx-10, midy-6, midx+10,midy+6);



// END BUTTON TO DRAW 3 COLOUR

// COVER BOX FOR SCALE OF SPOT
setfillstyle(SOLID_FILL,3);
midx = 220;
midy = 160;
bar(midx-47,midy-6,midx+73,midy+6);

setfillstyle(SOLID_FILL,3);
midx = 220;
midy = 60;
bar(midx-47,midy-6,midx+73,midy+6);

// END COVER BOX FOR SCALE OF SPOT

// BUTTON TO SCALE SEPERATIONS

setfillstyle(SOLID_FILL,6);
midx = 560;
midy = 360;
bar(midx-6, midy-6, midx+6,midy+6);

setfillstyle(SOLID_FILL,6);
midx = 589;
midy = 360;
bar(midx-6, midy-6, midx+6,midy+6);

setfillstyle(SOLID_FILL, 4);
midx = 575;
midy = 360;
bar(midx-8, midy-6,midx+7,midy+6);

itoa(scale_fd,sup3,10);
outtextxy(567,358,sup3);
```

```
// END BUTTONS TO SCALE SEPERATIONS

// BUTTONS FOR FIELD ANGLE CHANGE
setfillstyle(SOLID_FILL, 3);
midx = 576;
midy = 187;
bar(midx-20, midy-6, midx+20,midy+6);
outtextxy(557,185,"+0.1");



setfillstyle(SOLID_FILL, 3);
midx = 576;
midy = 227;
bar(midx-20, midy-6, midx+20,midy+6);
outtextxy(557,225,"-0.1");


// END BUTTONS FOR FIELD ANGLE CHANGE

midx = 175;
midy = 110;

 setfillstyle(SOLID_FILL, 3);

bar(midx-115, midy-40, midx+115,midy+40);

midx = 415;
midy = 210;
setfillstyle(SOLID_FILL, 3);

bar(midx-115, midy-40, midx+115,midy+40);

midx = 175;
midy = 210;
setfillstyle(SOLID_FILL, 3);
bar(midx-115, midy-40, midx+115, midy+40);


// box to cover spherochromatism
midx = 350;
midy = 110;
setfillstyle(SOLID_FILL, 0);
bar(midx-51, midy-57, midx+200, midy+55);

// end box to cover spherochromatism

// box to cover f_no

   midx = 460;
   midy = 10;
   setfillstyle(SOLID_FILL, 0);
   bar(midx-40, midy-40,midx+85, midy+38);

// end box to cover f_no

// BUTTON FOR PUPIL RAD
setfillstyle(SOLID_FILL, 3);
midx = 580;
midy = 10;
bar(midx-20, midy-6, midx+20,midy+6);
outtextxy(560,8,"+ 10");

setfillstyle(SOLID_FILL, 3);
midx = 580;
midy = 37;
bar(midx-20, midy-6, midx+20,midy+6);
outtextxy(560,35,"- 10");

// BUTTON FOR PUPIL RAD
```

```
 // BUTTON TO SCALE SPHERO

 setfillstyle(SOLID_FILL, 13);
 midx = 560;
 midy = 140;
 bar(midx-6, midy-6, midx+6,midy+6);

 setfillstyle(SOLID_FILL, 13);
 midx = 589;
 midy = 140;
 bar(midx-6, midy-6, midx+6,midy+6);

 setfillstyle(SOLID_FILL, 4);
 midx = 575;
 midy = 140;
 bar(midx-8, midy-6,midx+7,midy+6);

 itoa(scale_sp,sup,10);
 outtextxy(567,138,sup);


 // END BUTTON TO SCALE SPHERO

 // BUTTON TO TURN ON/OFF SPHERO CURVES

 setfillstyle(SOLID_FILL, 13);
 midx = 315;
 midy = 45;
 bar(midx-10, midy-5, midx+10,midy+5);

 // END BUTTON TO TURN ON/OFF SPHERO CURVES

 // BUTTON TO TURN ON/OFF parameter map

//  setfillstyle(SOLID_FILL, 1);
//  midx = 340;
//  midy = 45;
//  bar(midx-10, midy-4, midx+10,midy+4);

 // END BUTTON TO TURN ON/OFF parameter map

 // BUTTON TO CHANGE ASPEHRIC SURF NO

 setfillstyle(SOLID_FILL, 11);
 midx = 9;
 midy = 435;
 bar(midx-6, midy-6, midx+6,midy+6);


 setfillstyle(SOLID_FILL,11);
 midx = 38;
 midy = 435;
 bar(midx-6, midy-6, midx+6,midy+6);

 // END BUTTON TO CHANGE ASPERIC SURF NO


     pupil_inc = 0;


     (*surfaceptr[3]).check4();
     c11_inc = .00001;

     (*surfaceptr[4]).check4();
     if (c111!=0){
     c22_inc = c111/1000;}
  // corrector plate

     d_inc= 2;
```

```
      def_inc = defocus_in;

      // second increment for surface distanace changes
      d3_inc =0.002;
      // second increment for  distance changes

   // NEW 94
      e_inc = 0.0001;
      a2_inc = 0.000001;


   // END NEW 94


    // (*surfaceptr[surface4]).checka();
      a4_inc = 0.000000000001;

    //  (*surfaceptr[surface4]).checka();
      a6_inc = 0.0000000000000001;

      (*surfaceptr[6]).check4();
      if (c111 != 0){
      c1_inc = c111/1000;}

      // main cinc

      (*surfaceptr[5]).check4();
      cccl = c111;
      if(cccl !=0){
      cmain_inc = cccl/10000;}



      // end mainc inc

      (*surfaceptr[7]).check4();
      if (c111 !=0){
      c2_inc = c111/100;}

      (*surfaceptr[7]).check5();

      d2_inc=  D111/1000;

      // PRINT OUT SURFACE NUMBERS

     setfillstyle(SOLID_FILL, 3);
     midx = 61;
     midy = 424;
     bar(midx-8, midy-6, midx+9,midy+6);

     // NEW
     setfillstyle(SOLID_FILL, 3);
     midx = 61;
     midy = 460;
     bar(midx-8, midy-6, midx+9,midy+6);
     // NEW

  // NEW
     setfillstyle(SOLID_FILL, 3);
     midx = 61;
     midy = 475;
     bar(midx-8, midy-6, midx+9,midy+6);
     // NEW

     setfillstyle(SOLID_FILL, 3);
     midx = 61;
     midy = 438;
     bar(midx-8, midy-6, midx+9,midy+6);

     outtextxy(55,422,"A4");
```

```
outtextxy(55,436,"A6");
outtextxy(55,458,"A2");
outtextxy(55,472,"e");

setfillstyle(SOLID_FILL, 3);
midx = 61;
midy = 298;
bar(midx-8, midy-6, midx+9,midy+6);

itoa(su3,su3a,10);
outtextxy(58,296,su3a);

setfillstyle(SOLID_FILL, 3);
midx = 61;
midy = 334;
bar(midx-8, midy-6, midx+9,midy+6);

itoa(su4,su4a,10);
outtextxy(58,332,su4a);

setfillstyle(SOLID_FILL, 3);
midx = 61;
midy = 370;
bar(midx-8, midy-6, midx+9,midy+6);

setfillstyle(SOLID_FILL, 3);
midx = 61;
midy = 406;
bar(midx-8, midy-6, midx+9,midy+6);

setfillstyle(SOLID_FILL, 3);
midx = 322;
midy = 298;
bar(midx-8, midy-6, midx+9,midy+6);

itoa(su6,su6a,10);
outtextxy(319,296,su6a);

setfillstyle(SOLID_FILL, 3);
midx = 322;
midy = 334;
bar(midx-8, midy-6, midx+9,midy+6);

itoa(su7,su7a,10);
outtextxy(319,332,su7a);

setfillstyle(SOLID_FILL, 3);
midx = 322;
midy = 370;
bar(midx-8, midy-6, midx+9,midy+6);


setfillstyle(SOLID_FILL, 3);
midx = 322;
midy = 406;
bar(midx-8, midy-6, midx+9,midy+6);

// surfx
itoa(surfx,surfxa,10);
outtextxy(319,404,surfxa);

setfillstyle(SOLID_FILL, 12);
midx = 322;
midy = 417;
bar(midx-8, midy-3, midx+9,midy+3);

setfillstyle(SOLID_FILL, 12);
midx = 322;
midy = 396;
bar(midx-8, midy-3, midx+9,midy+3);
```

```
    // surfx

    setfillstyle(SOLID_FILL, 3);
    midx = 322;
    midy = 440;
    bar(midx-8, midy-6, midx+9,midy+6);

    itoa(su5,su5a,10);
    outtextxy(319,438,su5a);


     // END PRINT OUT SURFACE NUMBERS

     // button for zero colour doublet design
    setfillstyle(SOLID_FILL, 9);
    midx = 61;
    midy = 60;
    bar(midx-8, midy-6, midx+8,midy+6);

     // end button for zero colour doublet design

       // button for 98 PERCENT colour doublet design
    setfillstyle(SOLID_FILL, 9);
    midx = 81;
    midy = 60;
    bar(midx-8, midy-6, midx+8,midy+6);

     // end button for 98 PERCENT  colour doublet design

     // button for doublet corrector
    setfillstyle(SOLID_FILL, 9);
    midx = 61;
    midy = 263;
    bar(midx-8, midy-6, midx+8,midy+6);

     // end button for doublet corrector

       // button for doublet lens
   //  setfillstyle(SOLID_FILL, 9);
   //  midx = 323;
   //  midy = 263;
   //  bar(midx-8, midy-6, midx+8,midy+6);

     // end button for doublet lens


// END SET INCRIMENT FACTORS

mouse_show_cursor();
 mouse_range(0,0,getmaxx(),getmaxy());

 while ( (keypress != ('X')) && (keypress != ('x')) ){

//while ( (keypress != (('X')||('x')) )){
plot_bit =0;
keypress = 'p';
left =0;


while ( (keypress != 'X')&& (keypress != 'x') &&
    (keypress != 'D')&&
    (keypress != 'Q')&&
    (keypress != 'A')&&
    (keypress != 'W')&&
    (keypress != 'S')&&

    (keypress != 'U')&&
    (left ==0 ))
       {
```

```
       if (kbhit()){
         keypress = getch();}
      if (loop_no != 1){
        mouse_infor(&right, &left, &x, &y);
      }
      else if (loop_no == 1) { left =1;}

      if (right == 1){ putpixel(x,y,YELLOW);}
     //   left=1;

       }

  loop_no = loop_no+1;
  counter = counter +1;

  setfillstyle(SOLID_FILL, 3);
 midx = 20;
 midy = 70;
 bar(midx-20, midy-5, midx +20, midy+10);

 // PRINT SURFACE NUMBERS - IN TINY TURQUOSE BOXES

 // COVER SPHERO SCALE
 setfillstyle(SOLID_FILL, 4);
 midx = 575;
 midy = 140;
 bar(midx-8, midy-6,midx+7,midy+6);

 itoa(scale_sp,sup,10);
 outtextxy(567,138,sup);
 //END COVER SPHERO SCALE
 // COVER SCALE SEP
 setfillstyle(SOLID_FILL, 4);
 midx = 575;
 midy = 360;
 bar(midx-8, midy-6,midx+7,midy+6);

 itoa(d2_lab,sup3,10);
 outtextxy(567,358,sup3);

 // END COVER SCALE SEP
   // COVER SCALE_CM
   setfillstyle(SOLID_FILL, 4);
 midx = 575;
 midy = 440;
 bar(midx-8, midy-6,midx+7,midy+6);

 itoa(scale_cm,sup2,10);
 outtextxy(567,437,sup2);

   // END COVER SCALE_CM

   // PRINT OUT SURFACE NUMBERS

     if (d_corr == 1){

    setfillstyle(SOLID_FILL, 3);
    midx = 61;
    midy = 370;
    bar(midx-8, midy-6, midx+9,midy+6);

    itoa(su5,su5a,10);
    outtextxy(60,368,su5a);

    setfillstyle(SOLID_FILL, 3);
    midx = 61;
    midy = 406;
    bar(midx-8, midy-6, midx+9,midy+6);
    itoa(su5,su5a,10);
    outtextxy(60,404,su6a);


    setfillstyle(SOLID_FILL, 3);
    midx = 322;
    midy = 298;
    bar(midx-8, midy-6, midx+9,midy+6);


    itoa(su8,su8a,10);
    outtextxy(319,296,su8a);

    setfillstyle(SOLID_FILL, 3);
    midx = 322;
    midy = 334;
    bar(midx-8, midy-6, midx+9,midy+6);

    itoa(su9,su9a,10);
    outtextxy(319,332,su9a);

    setfillstyle(SOLID_FILL, 3);
    midx = 322;
    midy = 370;
    bar(midx-8, midy-6, midx+9,midy+6);

    setfillstyle(SOLID_FILL, 3);
    midx = 322;
    midy = 406;
    bar(midx-8, midy-6, midx+9,midy+6);

    setfillstyle(SOLID_FILL, 3);
    midx = 322;
    midy = 440;
    bar(midx-8, midy-6, midx+9,midy+6);

    itoa(su7,su7a,10);
    outtextxy(319,438,su7a);

    } // end if doublet corrector

      // surfx
    setfillstyle(SOLID_FILL, 3);
    midx = 322;
    midy = 406;
    bar(midx-8, midy-6, midx+9,midy+6);

    itoa(surfx,surfxa,10);
    outtextxy(319,404,surfxa);
       // surfx

     // END PRINT OUT SURFACE NUMBERS
  // END PRINT TINY BOXES



// BOX TO COVER SPHERO




if (rubout > 1) {

midx = 350;
midy = 110;
setfillstyle(SOLID_FILL, 0);
bar(midx-59, midy-57, midx+190, midy+59);
}
// END BOX TO COVER SPHERO
```

```
  setfillstyle(SOLID_FILL, 14);
  midx = 523;
  midy = 270;
  bar(midx-10, midy-3, midx+10,midy+3);



  checka4=0;
  checka6=0;

      // DRAW PARAMETER MAPS

  if (   ( (x >330) & (x <350) ) &( (y>41) & (y< 49)  )) {

  if (d_corr==1) {
       if (map1 ==1) {
         map1 =0;
        // setfillstyle(SOLID_FILL, 1);
       //  midx = 340;
       //  midy = 45;
        // bar(midx-10, midy-4, midx+10,midy+4);
       }
       else if (map1 ==0) {
         map1 =1;
        // setfillstyle(SOLID_FILL, 14);
       //   midx = 340;
        // midy = 45;
       //   bar(midx-10, midy-4, midx+10,midy+4);
       }
   }
   else {cout << "\a"; }

  }

  // END DRAW PARAMETER MAPS



  //chang in aspheric surf no

      if (   ( (x >32) & (x <46) ) &( (y>429) & (y< 441)  )) {


      asp_no = asp_no +1;
      surface4 = surface4 +1;
  scale_change =1;
  }

  if (   ( (x >3) & (x <15) ) &( (y>429) & (y< 441)  )) {

  asp_no = asp_no -1;
  surface4 = surface4-1;
  scale_change =1;
  }
     //end change in aspheric surface no


  // BOX TO COVER ASPHERIC NO

//   a_asp_1 = asp_no -1;
      //    a_asp_1 = surface4;
   //    itoa(a_asp_1,a_asp,10);
     // outtextxy(23,432,a_asp);


  // END BOX TO COVER ASPEHRIC NO

  // BOX TO COVER FIELD ANG
```

```
   setfillstyle(SOLID_FILL, 3);
   midx = 570;
   midy = 207;
   bar(midx-19, midy-6, midx+48,midy+6);
   // END BOX TO COVER FIELD ANG



   // BOXES TO COVER GRAPH SCALE OF SPOTS
   setfillstyle(SOLID_FILL,3);
   midx = 220;
   midy = 160;
   bar(midx-47,midy-6,midx+73,midy+6);

   setfillstyle(SOLID_FILL,3);
   midx = 220;
   midy = 60;
   bar(midx-47,midy-6,midx+73,midy+6);
   // END BOSES TO COVER GRAPH SCALE OF SPOTS


   if (redraw == 1){
   rubout =4;}


   plot_bit =0;

   setfillstyle(SOLID_FILL, 7);
   midx = 23;
   midy = 270;
   bar(midx-7, midy-6, midx+8,midy+6);


   setfillstyle(SOLID_FILL, 4);
   midx = 23;
   midy = 310;
   bar(midx-7, midy-6, midx+8,midy+6);
// NEW

   setfillstyle(SOLID_FILL, 4);
   midx = 23;
   midy = 460;
   bar(midx-7, midy-6, midx+8,midy+6);

   setfillstyle(SOLID_FILL, 5);
   midx = 220;
   midy = 460;
   bar(midx-35, midy-6, midx+90,midy+6);
// TEX
// NEW

// NEW

   setfillstyle(SOLID_FILL, 4);
   midx = 23;
   midy = 475;
   bar(midx-7, midy-6, midx+8,midy+6);

   setfillstyle(SOLID_FILL, 5);
   midx = 220;
   midy = 475;
   bar(midx-35, midy-6, midx+90,midy+6);
// TEX
// NEW


   setfillstyle(SOLID_FILL, 4);
   midx = 23;
   midy = 350;
```

```
 bar(midx-7, midy-6, midx+8,midy+6);

 setfillstyle(SOLID_FILL, 4);
 midx = 23;
 midy = 435;
 bar(midx-7, midy-6, midx+8,midy+6);

 //a_asp_1 = surface4;
 itoa(surface4,a_asp,10);
 outtextxy(21,432,a_asp);

 setfillstyle(SOLID_FILL, 4);
 midx = 580;
 midy = 403;
 bar(midx-21, midy-6, midx+19,midy+6);

 // BOX FOR CHANGING SCALE OF PUPIL RAD

 // END PUPIL

// NEW A2 MOUSE SECTION
    if (    ( (x >120) & (x <160) ) &( (y>454) & (y< 466)  )) {

       a22 = (a22 + a2_inc);
 (*surfaceptr[surface4]).newa2(a22);

 }

 if (    ( (x >75) & (x <115) ) &( (y>454) & (y< 466)  )) {


       a22 = (a22 - a2_inc);
 (*surfaceptr[surface4]).newa2(a22);

 }

// NEW A2 MOUSE SECTION
// NEW e MOUSE SECTION
    if (    ( (x >120) & (x <160) ) &( (y>469) & (y< 481)  )) {

       e22 = (e22 + e_inc);
 (*surfaceptr[surface4]).newe(e22);

 }

 if (    ( (x >75) & (x <115) ) &( (y>469) & (y< 481)  )) {


       e22 = (e22 - e_inc);
 (*surfaceptr[surface4]).newe(e22);

 }

// NEW e MOUSE SECTION


 //A4 mouse section

 if (    ( (x >120) & (x <160) ) &( (y>418) & (y< 430)  )) {

       a44 = (a44 + a4_inc);
 (*surfaceptr[surface4]).newa4(a44);

 }

 if (    ( (x >75) & (x <115) ) &( (y>418) & (y< 430)  )) {


       a44 = (a44 - a4_inc);
 (*surfaceptr[surface4]).newa4(a44);
```

```
}


if (    ( (x >120) & (x <160) ) &( (y>436) & (y< 445)   )) {


      a66 = (a66 + a6_inc);
(*surfaceptr[surface4]).newa6(a66);

}

if (    ( (x >75) & (x <115) ) &( (y>436) & (y< 445)   )) {


      a66 = (a66 - a6_inc);
(*surfaceptr[surface4]).newa6(a66);

}

if (    ( (x >120) & (x <160) ) &( (y>276) & (y< 286)   )) {

(*surfaceptr[s3]).check5();

dddl = D111;
dddl = (dddl + d_inc );
(*surfaceptr[s3]).newd22(dddl);


}

if (    ( (x >75) & (x <115) ) &( (y>276) & (y< 286)   )) {

(*surfaceptr[s3]).check5();

dddl = D111;
dddl = (dddl - d_inc );
(*surfaceptr[s3]).newd22(dddl);

}
   // SCALE CHANGING IE CHANGE MAG

if (    ( (x >0) & (x <40) ) &( (y>80) & (y< 120)   )) {

    //    cout << "\a \n";
 mag = mag *2;
 scale_change=1;


}

if (    ( (x >0) & (x <40) ) &( (y>120) & (y< 160)   )) {

    //    cout << "\a \n";

    mag = mag/2;
    scale_change=1;


}

    // END SCALE CHANGE

   // MOUSE FIELC CURVE SENSING

if (    ( (x >380) & (x <418) ) &( (y>292) & (y< 304)   )) {

    //    cout << "\a \n";
       // next line added 28/10/93
    (*surfaceptr[s6]).check4();
```

```
     cccl = (c111 + (c1_inc) );
    (*surfaceptr[s6]).newc11(cccl);


 }

 if (    ( (x >335) & (x <375) ) &( (y>292) & (y< 304)   )) {

     //     cout << "\a \n";
       // next line added 28/10/93
       (*surfaceptr[s6]).check4();

       cccl = (c111 - (c1_inc) );
    (*surfaceptr[s6]).newc11(cccl);

 }

   // SURFX
 if (    ( (x >380) & (x <418) ) &( (y>400) & (y< 408)   )) {

      (*surfaceptr[surfx]).check4();
      if (surfx ==3){ cccl = (1/c111 + (d_inc)); cccl = 1/cccl;}
      else { cccl = (c111 + (c1_inc) );}

    (*surfaceptr[surfx]).newc11(cccl);
 }

 if (    ( (x >335) & (x <375) ) &( (y>400) & (y< 408)   )) {
       (*surfaceptr[surfx]).check4();
       if (surfx ==3){ cccl = (1/c111 - (d_inc)); cccl = 1/cccl;}
       else { cccl = (c111 - (c1_inc) );}

    (*surfaceptr[surfx]).newc11(cccl);
 }



   // END SURFX

 if (    ( (x >380) & (x <418) ) &( (y>328) & (y< 340)   )) {

     //     cout << "\a \n";
         //added 28/10/93
         (*surfaceptr[s7]).check4();

     ccc2= (c111 + (c2_inc) );
    (*surfaceptr[s7]).newc11(ccc2);


 }

 if (    ( (x >335) & (x <375) ) &( (y>328) & (y< 340)   )) {

     //     cout << "\a \n";
       // added 28/10/93
       (*surfaceptr[s7]).check4();

       ccc2 = (c111 - (c2_inc) );
    (*surfaceptr[s7]).newc11(ccc2);

 }

 if (    ( (x >380) & (x <418) ) &( (y>310) & (y< 322)   )) {

     //     cout << "\a \n";
     (*surfaceptr[s7]).check5();

 ddd50 = D111;
```

```
     ddd50 = (ddd50 + (d3_inc) );
    (*surfaceptr[s7]).newd22(ddd50);


}
// ADDED ON  28/10/93
if (    ( (x >335) & (x <375) ) &( (y>274) & (y< 286)  )) {
      // cout << "\a \n";
       (*surfaceptr[s6]).check5();
       ddd50 = D111;
       ddd50 = (ddd50 - (d_inc) );
       (*surfaceptr[s6]).newd22(ddd50);
}

if (    ( (x >380) & (x <418) ) &( (y>274) & (y< 286)  )) {
    // cout << "\a \n";
     (*surfaceptr[s6]).check5();
     ddd50 = D111;
     ddd50 = (ddd50 + (d_inc) );
     (*surfaceptr[s6]).newd22(ddd50);

}

// END ADDED  28/10/93

if (    ( (x >335) & (x <375) ) &( (y>310) & (y< 322)  )) {

     //    cout << "\a \n";
      (*surfaceptr[s7]).check5();

    ddd50 = D111;

     ddd50 = (ddd50 - (d3_inc) );
    (*surfaceptr[s7]).newd22(ddd50);


}

// BUTTON TO CHANGE SEPERATION OF FIRST CORR PLATE

if (    ( (x >120) & (x <160) ) &( (y>310) & (y< 322) )) {

 (*surfaceptr[s4]).check5();

ddd50 = D111;

     ddd50 = (ddd50 + (d3_inc) );
    (*surfaceptr[s4]).newd22(ddd50);
}

if (    ( (x >75) & (x <115) ) &( (y>310) & (y< 322)  )) {

(*surfaceptr[s4]).check5();

ddd50 = D111;

     //    cout << "\a \n";
       ddd50 = (ddd50 - (d3_inc) );
    (*surfaceptr[s4]).newd22(ddd50);

}



// END BUTTON TO CHANGE SEPERATION

// BUTTONS FOR DOUBLET CORRECTOR PLATE SEPERATIONS

if (    ( (x >120) & (x <160) ) &( (y>346) & (y< 358) )) {
 (*surfaceptr[5]).check5();

ddd50 = D111;

    ddd50 = (ddd50 + (d3_inc) );
   (*surfaceptr[5]).newd22(ddd50);
}

if (   ( (x >75) & (x <115) ) &( (y>346) & (y< 358)  )) {

(*surfaceptr[5]).check5();

ddd50 = D111;

    //    cout << "\a \n";
      ddd50 = (ddd50 - (d3_inc) );
   (*surfaceptr[5]).newd22(ddd50);

}


if (   ( (x >120) & (x <160) ) &( (y>382) & (y< 394) )) {

 (*surfaceptr[6]).check5();

ddd50 = D111;

    ddd50 = (ddd50 + (d3_inc) );
   (*surfaceptr[6]).newd22(ddd50);
}

if (   ( (x >75) & (x <115) ) &( (y>382) & (y< 394)  )) {

(*surfaceptr[6]).check5();

ddd50 = D111;

    //    cout << "\a \n";
      ddd50 = (ddd50 - (d3_inc) );
   (*surfaceptr[6]).newd22(ddd50);

}


// END BUTTONS FOR CORRECTOR DOUBLET PLATE SEPERATIONS


// SENSING FOR CORECTOR PLATE

if (   ( (x >120) & (x <170) ) &( (y>293) & (y< 304)  )) {

    //    cout << "\a \n";

   (*surfaceptr[s3]).check4();

    c11= (c111 + (c11_inc) );
   (*surfaceptr[s3]).newc11(c11);

}

if (   ( (x >75) & (x <115) ) &( (y>293) & (y< 304)  )) {

    //    cout << "\a \n";

      (*surfaceptr[s3]).check4();

    c11= (c111 - (c11_inc) );
   (*surfaceptr[s3]).newc11(c11);

}
```

```
if (    ( (x >120) & (x <170) ) &( (y>328) & (y< 340)   )) {

    //    cout << "\a \n";

      (*surfaceptr[s4]).check4();

       c22 = (c111 + (c22_inc) );
   (*surfaceptr[s4]).newc11(c22);

}

if (    ( (x >75) & (x <115) ) &( (y>328) & (y< 340)   )) {

    //    cout << "\a \n";

       (*surfaceptr[s4]).check4();

       c22 = (c111 - (c22_inc) );
   (*surfaceptr[s4]).newc11(c22);

}

// BUTTONS FOR HOUHTON CURVATURES

if (    ( (x >120) & (x <170) ) &( (y>364) & (y< 376)   )) {

    //    cout << "\a \n";

   (*surfaceptr[5]).check4();

    c11= (c111 + (c11_inc) );
   (*surfaceptr[5]).newc11(c11);

}

if (    ( (x >75) & (x <115) ) &( (y>364) & (y< 376)   )) {

    //    cout << "\a \n";

       (*surfaceptr[5]).check4();

    c11= (c111 - (c11_inc) );
   (*surfaceptr[5]).newc11(c11);

}



if (    ( (x >120) & (x <170) ) &( (y>400) & (y< 412)   )) {

    //    cout << "\a \n";

      (*surfaceptr[6]).check4();

       c22 = (c111 + (c22_inc) );
   (*surfaceptr[6]).newc11(c22);

}

if (    ( (x >75) & (x <115) ) &( (y>400) & (y< 412)   )) {

    //    cout << "\a \n";

       (*surfaceptr[6]).check4();

       c22 = (c111 - (c22_inc) );
   (*surfaceptr[6]).newc11(c22);
}



// END BUTTONS FOR HOUGHTON CURVATURES

// FIELD CORR DESIGN SENSOR

if (   ( (x >560) & (x <598) ) &( (y>255) & (y< 286)  )) {

  if (field_des==0){

     if (third_run ==1){
    field_des =1;}
     else {cout<< "\a";}

  }

else if (field_des ==1){

  field_des =0;}

     //  des4();

}

if (field_des==1){
  setfillstyle(SOLID_FILL, 14);
midx = 580;
midy = 275;
bar(midx-20, midy-20, midx+18,midy+11);
outtextxy(563,270,mes3);
outtextxy(560,235,mes4);
outtextxy(565,245,mes5);
}

else if (field_des ==0){
  setfillstyle(SOLID_FILL, 9);
midx = 580;
midy = 275;
bar(midx-20, midy-20, midx+18,midy+11);
outtextxy(563,270,mes3);
outtextxy(560,235,mes4);
outtextxy(565,245,mes5);
}


if (field_des==1){
des4();
}

//enf field design of lens


// ACHRO MENISCUS FOR ASPHERIC PLATE

if (   ( (x >0) & (x <38) ) &( (y>180) & (y< 211)  )) {


   des5();

}
// END ACHRO MENISCUS FOR ASPHERIC PLATE


// END CORR DESIGN SENSOR


// sensing for aspheric scale factor
```

```
   if (    ( (x >32) & (x <46) ) &( (y>304) & (y< 316)   )) {

   a4_inc = a4_inc*10;
   scale_fa4 = scale_fa4 +1;
   scale_change=1;
   }

   if (    ( (x >3) & (x <15) ) &( (y>304) & (y< 316)   )) {

   a4_inc = a4_inc/10;
   scale_fa4 = scale_fa4 -1;
   scale_change =1;

   }

// NEW
      if (    ( (x >32) & (x <46) ) &( (y>454) & (y< 466)   )) {

   a2_inc = a2_inc*10;
   scale_fa2 = scale_fa2 +1;
   scale_change=1;
   }

   if (    ( (x >3) & (x <15) ) &( (y>454) & (y< 466)   )) {

   a2_inc = a2_inc/10;
   scale_fa2 = scale_fa2 -1;
   scale_change =1;

   }

// NEW
// NEW
      if (    ( (x >32) & (x <46) ) &( (y>469) & (y< 481)   )) {

   e_inc = e_inc*2;
   scale_fe = scale_fe +1;
   scale_change=1;
   }

   if (    ( (x >3) & (x <15) ) &( (y>469) & (y< 481)   )) {

   e_inc = e_inc/2;
   scale_fe = scale_fe -1;
   scale_change =1;

   }

// NEW



   if (    ( (x >32) & (x <46) ) &( (y>344) & (y< 356)   )) {

   a6_inc = a6_inc*10;
   scale_fa6 = scale_fa6+1;
   scale_change =1;

   }

   if (    ( (x >3) & (x <15) ) &( (y>344) & (y< 356)   )) {


   a6_inc = a6_inc/10;
   scale_fa6 = scale_fa6-1;
   scale_change=1;

   }
```

```
if (   ( (x >560) & (x <600) ) &( (y>384) & (y< 396)  )) {

d_inc = d_inc*2;
scale_fd = scale_fd+1;
scale_change =1;

}

if (   ( (x >560) & (x <600) ) &( (y>408) & (y<420)  )) {

d_inc = d_inc/2;
scale_fd = scale_fd-1;
scale_change= 1;

}

if (   ( (x >32) & (x <46) ) &( (y>264) & (y< 276)  )) {

c22_inc = c22_inc*10;
c11_inc = c11_inc*10;
scale_fc22 = scale_fc22+1;
scale_change =1;
c1_inc = c1_inc*10;
c2_inc = c2_inc*10;
//cout << "\a \n";

}

if (   ( (x >3) & (x <15) ) &( (y>264) & (y< 276)  )) {

c22_inc = c22_inc/10;
c11_inc = c11_inc/10;
scale_fc22 = scale_fc22-1;
scale_change=1;
c1_inc = c1_inc/10;
c2_inc= c2_inc/10;

}

// end sensing for aspheric scale factor

// buttons for both changing c1 and correcting c2 for chromatic

if (   ( (x >0) & (x <12) ) &( (y>219) & (y< 231)  )) {

    (*surfaceptr[s3]).check4();
    c11= (c111 - (c11_inc) );
    (*surfaceptr[s3]).newc11(c11);

des5();

}

if (   ( (x >24)& (x <36) ) &( (y>219) & (y< 231)  )) {


   (*surfaceptr[s3]).check4();
   c11= (c111 + (c11_inc) );
   (*surfaceptr[s3]).newc11(c11);
   des5();

}



// end chromatic buttons

// button for d2_inc
```

```
if (   ( (x >554) & (x <566) ) &( (y>354) & (y< 366)  )) {

d3_inc = d3_inc /2;
d2_lab = d2_lab-1;
scale_change =1;

}

if (   ( (x >583)& (x <595) ) &( (y>354) & (y< 366)  )) {

d3_inc = d3_inc*2;
d2_lab = d2_lab+1;
scale_change =1;


}

// end button for d2_inc

   // BUTTONS FOR SENSING FOR MOVING STOP

   if (   ( (x >380) & (x <418) ) &( (y>420) & (y< 432)  )) {

     (*surfaceptr[s5]).checkstop();

stop3 = aperture + d_inc;

   (*surfaceptr[s5]).newstop(stop3);

 //   (*surfaceptr[3]).check5();

 //   ddd1 = D111;
 //   ddd1 = (ddd1 + d_inc );
 //   (*surfaceptr[3]).newd22(ddd1);

}

if (   ( (x >335)& (x <373) ) &( (y>420) & (y< 432)  )) {

  (*surfaceptr[s5]).checkstop();

   stop3 = aperture - d_inc;

  (*surfaceptr[s5]).newstop(stop3);


   // (*surfaceptr[3]).check5();

  //  ddd1 = D111;
  //  ddd1 = (ddd1 - d_inc );
  //  (*surfaceptr[3]).newd22(ddd1);


}


   // END MOVE STOP SENSING

   // BUTTONS TO INC/DEC SURFX
if (   ( (x >314) & (x <340) ) &( (y>393) & (y< 399)  )) {

  if (surfx != (lastsurf+1)){
    surfx = surfx +1;}
  else {cout << "\a";}
}

if (   ( (x >314) & (x <340) ) &( (y>414) & (y<420)  )) {

 if (surfx != 1){
  surfx = surfx -1;}
 else {cout << "\a";}

}



   // END SURFX

   // SENSING FOR BUTTONS FOR CGNAGING C OF PRIMARY

   if (    ( (x >380) & (x <418) ) &( (y>434) & (y< 446)   )) {

(*surfaceptr[s5]).check4();

cccl = c111;
cccl = (cccl + cmain_inc );
(*surfaceptr[s5]).newc11(cccl);

}

if (    ( (x >335)& (x <373) ) &( (y>434) & (y< 446)   )) {

(*surfaceptr[s5]).check4();

cccl = c111;
cccl = (cccl - cmain_inc );
(*surfaceptr[s5]).newc11(cccl);

}


   // END SENSING FOR CHANGING C OF PRIMARY

   // SENSING FOR SCALE CHANGE FOR PRIMARY

if (    ( (x >583) & (x <595) ) &( (y>434) & (y< 446)   )) {

cmain_inc = cmain_inc*10;
scale_cm = scale_cm+1;
scale_change =1;

}

if (    ( (x >554) & (x <566) ) &( (y>434) & (y< 446)   )) {

cmain_inc = cmain_inc/10;
scale_cm = scale_cm-1;
scale_change=1;

}



   // END SCALE CHANGE FOR PRIMARY


   // SENSE FOR DEFOCUS INCRUMENT CHANGE

if (    ( (x >5) & (x <35) ) &( (y>1) & (y< 13)   )) {


def_inc = def_inc+0.01;
scale_change=1;



}

if (    ( (x >5) & (x <35) ) &( (y>29) & (y< 41)   )) {
 def_inc = def_inc-0.01;
 scale_change=1;


 }


    // END SENSE FOR DEFOCUS INCRUMENT CHANGE

    // sensing for 3 colour button

 if (    ( (x >80) & (x <140)) &( (y>154) & (y<166)   )) {

 if (redraw ==1){redraw =0;}
 else {redraw =1;}

 }

    // end sensing for 3 colour button

    // SENSING FOR D_CORR

 if (    ( (x >53) & (x <69) ) &( (y>257) & (y< 269)   )) {


 d_corr = 1 ;

 setfillstyle(SOLID_FILL, 14);
 midx = 61;
 midy = 263;
 bar(midx-8, midy-6, midx+8,midy+6);

 }

//  if (    ( (x >315) & (x <331) ) &( (y>257) & (y< 269)   )) {

 //d_lens = 1;

//   setfillstyle(SOLID_FILL, 14);
//   midx = 323;
 //  midy = 263;
 //  bar(midx-8, midy-6, midx+8,midy+6);

 //  }


    // END SENSING FOR D_CORR

    // SENSE FOR CHANGE IN FIELD ANG

 if (    ( (x >556) & (x <596) ) &( (y>181) & (y< 193)   )) {
     //    cout << "\a \n";


 mem_ang = mem_ang + 0.1 ;
 scale_change=1;


 }

 if (    ( (x >556) & (x <596) ) &( (y>221) & (y< 233)   )) {

 mem_ang = mem_ang - 0.1;
 scale_change=1;

 }



    // END SENSE FOR CHANGE IN FIELD ANG
```

```
   // pupil rad

   if (    ( (x >560) & (x <600) ) &( (y>4) & (y< 16)  )) {


rad_pupil = rad_pupil +10 ;
scale_change=1;


}

if (    ( (x >560) & (x <600) ) &( (y>31) & (y< 43)  )) {


rad_pupil = rad_pupil -10 ;
scale_change=1;


}

   // pupil rad

   // SCALE SPHERO

  if (    ( (x >583) & (x <595) ) &( (y>134) & (y< 146)  )) {

scale_sp = scale_sp+1;
scale_3 = scale_3 *2 ;
scale_change =1;

}

if (    ( (x >554) & (x <566) ) &( (y>134) & (y< 146)  )) {

scale_sp = scale_sp-1;
scale_3 = scale_3 /2 ;
scale_change =1;


}




if (    ( (x >305) & (x <325) ) &( (y>41) & (y< 50)  )) {

  if (sphero == 1){
  sphero =0;

  }

  else if (sphero == 0){ sphero =1;}

}

if (sphero == 1) {
   setfillstyle(SOLID_FILL, 14);
   midx = 315;
   midy = 45;
   bar(midx-10, midy-5, midx+10,midy+5);

}

if (sphero ==0){
   setfillstyle(SOLID_FILL, 13);
   midx = 315;
   midy = 45;
   bar(midx-10, midy-5, midx+10,midy+5);
```

```
}

if (   ( (x >230) & (x <250) ) &( (y>41) & (y< 49)  )) {

  if (plot_3 == 1){plot_3 =0;}
  else if (plot_3 == 0){ plot_3 =1;}

}


if (   ( (x >53) & (x <69) ) &( (y>54) & (y< 66)  )) {

  if (des88 == 1){des88 =0;

  }
  else if (des88 == 0){
        if (d_corr==1) { des88 =1;}
        else {cout << "\a";}

  }

}

if (des88 == 1){
   setfillstyle(SOLID_FILL, 14);
   midx = 61;
   midy = 60;
   bar(midx-8, midy-6, midx+8,midy+6);
  }
  else if (des88 == 0){
   setfillstyle(SOLID_FILL, 9);
   midx = 61;
   midy = 60;
   bar(midx-8, midy-6, midx+8,midy+6);

  }

  if (   ( (x >73) & (x <89) ) &( (y>54) & (y< 66)  )) {

  if (des88b == 1){des88b =0;

  }
  else if (des88b == 0){
        if (d_corr==1) { des88b =1;}
        else {cout << "\a";}

  }

  }

if (des88b == 1){
   setfillstyle(SOLID_FILL, 14);
   midx = 81;
   midy = 60;
   bar(midx-8, midy-6, midx+8,midy+6);
  }
  else if (des88b == 0){
   setfillstyle(SOLID_FILL, 9);
   midx = 81;
   midy = 60;
   bar(midx-8, midy-6, midx+8,midy+6);

  }


  //     if (lens ==1) {
 //   schem2(400,120,1);
  //  }

if (sphero ==1){
```

```
plot2();}

   // END SCALE SPHERO



   // MOUSE FIELD CURVE SENSING



   // DOUBLET COLOUR CORRECTION

if (des88==1) {
des8();
}
   // END DOUBLET COLOUR CORRECTION

    // DOUBLET COLOUR CORRECTION AT 100 PERCENT LEVEL

if (des88b==1) {
des8b();
}
   // END DOUBLET COLOUR CORRECTION AT 100 PERCENT LEVEL



if ((d_corr ==1)&(done_lens==0) ){
done_lens=1;  // done_lens means program set for doublet corrector
s5 = s5+2;
s6 =s6+2;
s7=s7+2;
s8=s8+2;
s9=s9+2;
s10=s10+2;
s11=s11+2;
}

   if ((keypress != 'X') &&(keypress !='x')) {


setfillstyle(SOLID_FILL, 0);
midx = 45;
midy = 20;
bar(midx-12, midy-25, midx+180,midy+28);


 // if (scale_change == 0) { // only do plots if not
  flag_paraxial_y=0;
  efl1();

c444 = fcvt(bfl,2,&bit14,&bit24);
outtextxy(119,15,c444);
itoa(bit14,bit144,20);
itoa(bit24,bit244,20);
  if (bit24 !=0){
  outtextxy(93,15,"-");}
  else {   outtextxy(93,15,"+");}
outtextxy(103,15,"0.");
outtextxy(185,15,bit144);
outtextxy(175,15,"E");
outtextxy(35,15,"P bfl");



   bfl = bfl + def_inc;

 //  plot2();


c444 = fcvt(efl,2,&bit14,&bit24);
```

```
outtextxy(119,5,c444);
itoa(bit14,bit144,10);
itoa(bit24,bit244,10);
  if (bit24 !=0){
  outtextxy(93,5,"-");}
  else {   outtextxy(93,5,"+");}
outtextxy(103,5,"0.");
outtextxy(185,5,bit144);
outtextxy(175,5,"E");
outtextxy(35,5,"E.F.L");

c444 = fcvt(bfl,2,&bit14,&bit24);
outtextxy(119,30,c444);
itoa(bit14,bit144,20);
itoa(bit24,bit244,20);
  if (bit24 !=0){
  outtextxy(93,30,"-");}
  else {   outtextxy(93,30,"+");}
outtextxy(103,30,"0.");
outtextxy(185,30,bit144);
outtextxy(175,30,"E");
outtextxy(35,30,"B.F.L.");
outtextxy(35,40,"Defocus");



   // end calculate new efl and bfl

 //  } // end scale chnage =0

   // print out defocus increment

     c444 = fcvt(def_inc,2,&bit14,&bit24);
outtextxy(119,40,c444);
itoa(bit14,bit144,10);
itoa(bit24,bit244,10);
if (bit24 !=0){
outtextxy(93,40,"-");}
else {   outtextxy(93,40,"+");}
outtextxy(103,40,"0.");
outtextxy(185,40,bit144);
outtextxy(175,40,"E");

   // end print out defocus increment

   // GIG

   if (scale_change == 0) { // only chnage plots if not


 //   setfillstyle(SOLID_FILL, 1);
  //  midx = 450;
 //   midy = 335;
  //  bar(midx-25, midy-60, midx+80,midy+50);
   // outtextxy(80,355,a42);

rubout = (rubout +1);



field_angl = 0;
y_shift = 0;
no_sag =1;

no_times = 0;

    if (rubout > 3) {

midx = 175;
midy = 110;
setfillstyle(SOLID_FILL, 3);

bar(midx-115, midy-40, midx+115,midy+40);


 rubout =0;
 midx = 175;
 midy = 210;

 setfillstyle(SOLID_FILL, 3);

bar(midx-115, midy-40, midx+115,midy+40);

midx = 415;
midy = 210;
setfillstyle(SOLID_FILL, 3);

bar(midx-115, midy-40, midx+115,midy+40);
// max
// THIS COVERS FIELD CURV CHART


    }

 field_angl =0;

    if ((redraw ==1)&(plot_3==0)){  // 3 colour on axis plots
 done =0;
 yell =0; // yell (tangentail) is a triger to plot other
    //  colours with respect to yellow in y direction
 yell2 =0; // yell2 is same for sagital rays

 for (colourray = 1; colourray <= (3); colourray++) {
// redraw = 1;

 graph2();

 }
} // 3 colour on axis plots

colourray =1;

if ((redraw == 0)&&(plot_3 ==0)) {
 graph();

}

loop =1;
no_sag =0;
back_graph =0;

field_angl = mem_ang;
y_shift = 100;


no_times =1;



if ((redraw ==1)&&(plot_3==0)){  // 3 colour on axis plots
done =0;
yell =0;
yell2 =0;
 for (colourray = 1; colourray <= (3); colourray++) {
// redraw = 1;
 graph2();

 }
} // 3 colour on axis plots
```

```
  colourray =1;

  if ((redraw == 0)&&(plot_3==0)) {
   graph();

  }
  //graph();

 // map section deleted 4/7/94


// map section deleted 4/7/94

  loop =1;
  back_graph =0;

  // plot 3rd order graphs

  if ((redraw == 0)&&(plot_3==1)) {
  y_shift =0;
      //   memsx = field_angl;
      //   field_angl =0.1;
  field_angl = memsx;
  graph4();
      //   field_angl = memsx;


   y_shift =100;

   graph5();

  }

  // end plot 3rd order graphs

  } // END IF SCALE_CHANGE=0


  // BLUE DATA RUBOUTS HERE

  setfillstyle(SOLID_FILL, 1);
  midx = 450;
  midy = 335;
  bar(midx-25, midy-60, midx+80,midy+50);

  setfillstyle(SOLID_FILL, 1);
  midx = 487;
  midy = 433;
  bar(midx-63, midy-20, midx+52,midy+13);

  setfillstyle(SOLID_FILL, 1);
  midx = 210;
  midy = 280;
  bar(midx-40, midy-3, midx+100,midy+127);

  // BOX TO COVER FNO

     midx = 460;
     midy = 10;
     setfillstyle(SOLID_FILL, 0);
     bar(midx-38, midy-40,midx+78, midy+38);

  // END BOX TO COVER FNO

  // BOXES TO COVER DATA
  setfillstyle(SOLID_FILL, 1);
  midx = 450;
  midy = 335;
  bar(midx-26, midy-61, midx+89,midy+113);
 //   outtextxy(80,355,a42);
setfillstyle(SOLID_FILL, 1);
midx = 210;
midy = 280;
bar(midx-40, midy-6, midx+100,midy+165);
// END BOXES TO COVER DATA




// END BLUE DATA RUBOUTS

(*surfaceptr[s5]).checkstop();

new_stop1 = aperture;
new_stop2 = fcvt(new_stop1,0,&bit1,&bit2);


outtextxy(455,424,new_stop2);
outtextxy(500,424,"mm");

// PRINT OUT STOP DISTANCE FROM FIRST SURFACE
// GIG1
(*surfaceptr[s3]).check5();

ddd1 = D111;

c444 = fcvt(ddd1,0,&bit14,&bit24);
outtextxy(210,278,c444);
itoa(bit14,bit144,10);
itoa(bit24,bit244,10);
  if (bit24 !=0){
  outtextxy(200,278,"-");}
  else {   outtextxy(200,278,"+");}
outtextxy(253,278,"mm");


// END PRINT STOP DISTANCE

// CHECK FIELD CURV CORRECTOR

if (lastsurf >5){

(*surfaceptr[s6]).check5();

ddd1 = D111;

c444 = fcvt(ddd1,1,&bit14,&bit24);
outtextxy(453,278,c444);
itoa(bit14,bit144,10);
itoa(bit24,bit244,10);
  if (bit24 !=0){
  outtextxy(426,278,"-");}
  else {   outtextxy(426,278,"+");}
outtextxy(435,278,"0.");
outtextxy(520,278,bit144);
outtextxy(510,278,"E");





(*surfaceptr[s6]).check4();
//change
//ccc1 = 1/c111;
if (c111 !=0){
ccc1 = 1/c111;}
else if (c111==0){
outtextxy(453,296,"Infinite");
plane_bit=1;
```

```
}

if (plane_bit ==0){
c444 = fcvt(ccc1,1,&bit14,&bit24);
outtextxy(453,296,c444);
itoa(bit14,bit144,10);
itoa(bit24,bit244,10);
  if (bit24 !=0){
  outtextxy(426,296,"-");}
  else {   outtextxy(426,296,"+");}
outtextxy(435,296,"0.");
outtextxy(520,296,bit144);
outtextxy(510,296,"E");
}
plane_bit=0;

(*surfaceptr[s7]).check4();
//ccc2 = 1/c111;
if (c111 !=0){
ccc2 = 1/c111;}
else if (c111==0){
outtextxy(453,332,"Infinite");
plane_bit=1;
}

if (plane_bit==0){
c444 = fcvt(ccc2,1,&bit14,&bit24);
outtextxy(453,332,c444);
itoa(bit14,bit144,10);
itoa(bit24,bit244,10);
  if (bit24 !=0){
  outtextxy(426,332,"-");}
  else {   outtextxy(426,332,"+");}
outtextxy(435,332,"0.");
outtextxy(520,332,bit144);
outtextxy(510,332,"E");
}
plane_bit=0;

(*surfaceptr[s7]).check5();

ddd1 = D111;

c444 = fcvt(ddd1,1,&bit14,&bit24);
outtextxy(453,314,c444);
itoa(bit14,bit144,10);
itoa(bit24,bit244,10);
  if (bit24 !=0){
  outtextxy(426,314,"-");}
  else {   outtextxy(426,314,"+");}
outtextxy(435,314,"0.");
outtextxy(520,314,bit144);
outtextxy(510,314,"E");

  // surfx -user chosen surface

(*surfaceptr[surfx]).check4();
//ccc2 = 1/c111;
if (c111 !=0){
ccc2 = 1/c111;}
else if (c111==0){
outtextxy(453,404,"Infinite");
plane_bit=1;
}

if (plane_bit==0){
c444 = fcvt(ccc2,1,&bit14,&bit24);
outtextxy(453,404,c444);
itoa(bit14,bit144,10);
itoa(bit24,bit244,10);
```

```
   if (bit24 !=0){
   outtextxy(426,404,"-");}
   else {   outtextxy(426,404,"+");}
 outtextxy(435,404,"0.");
 outtextxy(520,404,bit144);
 outtextxy(510,404,"E");
 }
 plane_bit=0;

   // surfx - user chosen surface

     if (d_lens ==1){



 (*surfaceptr[s8]).check5();

 dddl = D111;

 c444 = fcvt(dddl,1,&bit14,&bit24);
 outtextxy(453,350,c444);
 itoa(bit14,bit144,10);
 itoa(bit24,bit244,10);
   if (bit24 !=0){
   outtextxy(426,350,"-");}
   else {   outtextxy(426,350,"+");}
 outtextxy(435,350,"0.");
 outtextxy(520,350,bit144);
 outtextxy(510,350,"E");



 (*surfaceptr[s8]).check4();
 // change
 //cccl = 1/c111;
 if (c111 !=0){
 cccl = 1/c111;}
 else if (c111==0){
 outtextxy(453,368,"Infinite");
 plane_bit=1;
 }

 if (plane_bit ==0){
 c444 = fcvt(cccl,1,&bit14,&bit24);
 outtextxy(453,368,c444);
 itoa(bit14,bit144,10);
 itoa(bit24,bit244,10);
   if (bit24 !=0){
   outtextxy(426,368,"-");}
   else {   outtextxy(426,368,"+");}
 outtextxy(435,368,"0.");
 outtextxy(520,368,bit144);
 outtextxy(510,368,"E");
 }
 plane_bit=0;

 (*surfaceptr[s9]).check4();
 //chnage
 //ccc2 = 1/c111;
 if (c111 !=0){
 ccc2 = 1/c111;}
 else if (c111==0){
 outtextxy(453,404,"Infinite");
 plane_bit=1;
 }


 if (plane_bit==0){
 c444 = fcvt(ccc2,1,&bit14,&bit24);
 outtextxy(453,404,c444);
```

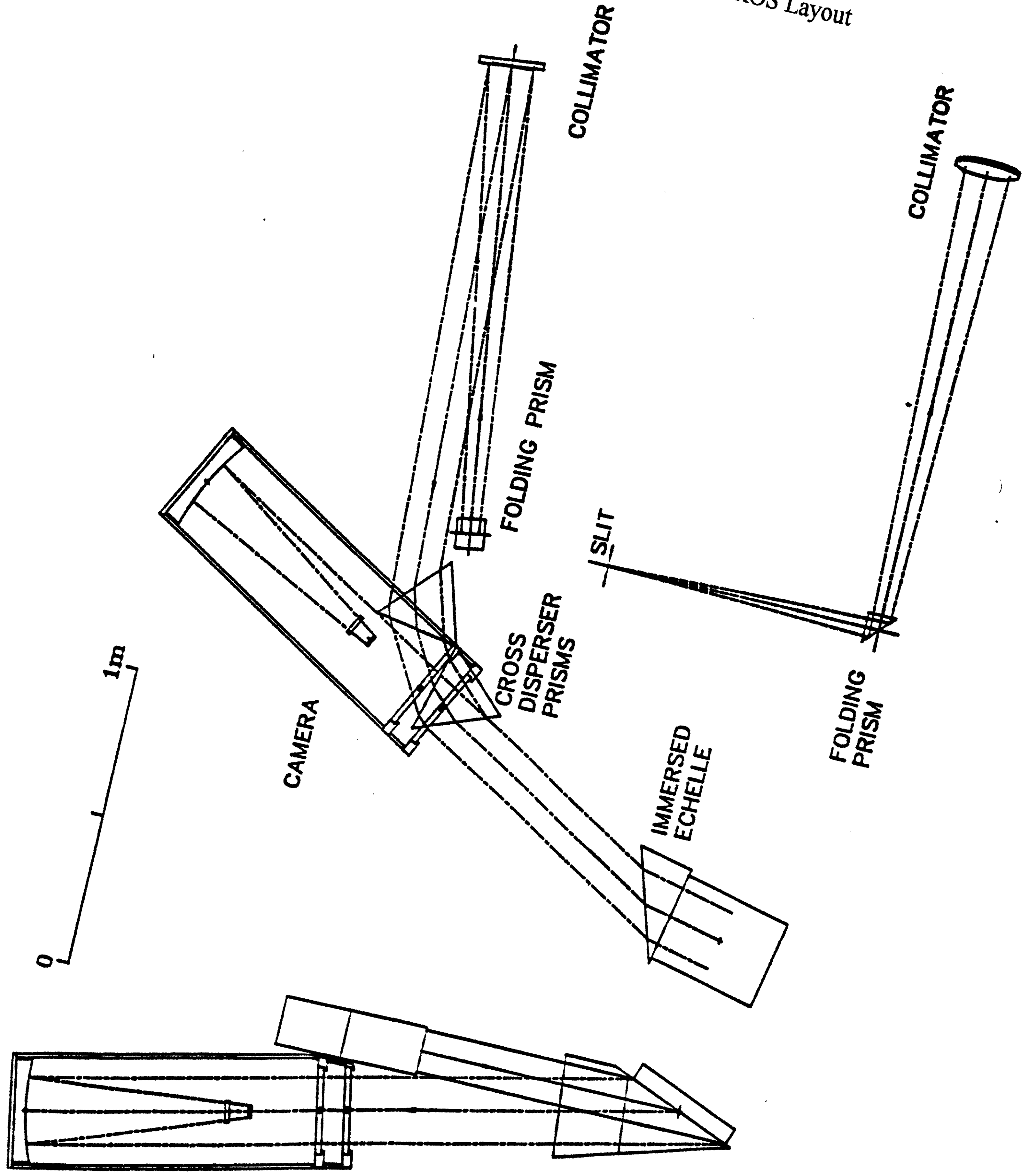
```
itoa(bit14,bit144,10);
itoa(bit24,bit244,10);
  if (bit24 !=0){
  outtextxy(426,404,"-");}
  else {   outtextxy(426,404,"+");}
outtextxy(435,404,"0.");
outtextxy(520,404,bit144);
outtextxy(510,404,"E");
}
plane_bit =0;

(*surfaceptr[s9]).check5();

dddl = D111;

c444 = fcvt(dddl,1,&bit14,&bit24);
outtextxy(453,386,c444);
itoa(bit14,bit144,10);
itoa(bit24,bit244,10);
  if (bit24 !=0){
  outtextxy(426,386,"-");}
  else {   outtextxy(426,386,"+");}
outtextxy(435,386,"0.");
outtextxy(520,386,bit144);
outtextxy(510,386,"E");
   // outtextxy(515,360,"mm");

   // END C3 C4  D2 D3
   } // end if d_corr =1

// END CHECK FIELD CURV CORR
   } // end if last 2surf >5


// CHECK CURVATURE OF CORRECTOR PLATE

(*surfaceptr[s3]).check4();

if (c111 !=0){
c111 = 1/c111;}
else if (c111==0){
outtextxy(200,296,"Infinite");
plane_bit=1;
}

if (plane_bit==0){
c444 = fcvt(c111,1,&bit14,&bit24);
outtextxy(200,296,c444);
itoa(bit14,bit144,10);
itoa(bit24,bit244,10);
  if (bit24 !=0){
  outtextxy(176,296,"-");}
  else {   outtextxy(176,296,"+");}
outtextxy(185,296,"0.");
outtextxy(286,296,bit144);
outtextxy(274,296,"E");
}
plane_bit=0;

(*surfaceptr[s4]).check4();
if (c111 !=0){
c111 = 1/c111;}
else if (c111==0){
outtextxy(200,332,"Infinite");
plane_bit=1;
}

if (plane_bit==0){
c444 = fcvt(c111,1,&bit14,&bit24);
outtextxy(200,332,c444);
```

```
itoa(bit14,bit144,10);
itoa(bit24,bit244,10);
  if (bit24 !=0){
  outtextxy(176,332,"-");}
  else {   outtextxy(176,332,"+");}
outtextxy(185,332,"0.");
outtextxy(286,332,bit144);
outtextxy(274,332,"E");
}
plane_bit=0;

(*surfaceptr[s4]).check5();

dddl = D111;

c444 = fcvt(dddl,1,&bit14,&bit24);
outtextxy(200,314,c444);
itoa(bit14,bit144,10);
itoa(bit24,bit244,10);
  if (bit24 !=0){
  outtextxy(176,314,"-");}
  else {   outtextxy(176,314,"+");}
outtextxy(185,314,"0.");
outtextxy(286,314,bit144);
outtextxy(274,314,"E");


// END CHECK CURVATURE OF CORRECTOR PLATE

if (d_corr ==1){ // doublet corrector

(*surfaceptr[5]).check5();

dddl = D111;

c444 = fcvt(dddl,1,&bit14,&bit24);
outtextxy(200,350,c444);
itoa(bit14,bit144,10);
itoa(bit24,bit244,10);
  if (bit24 !=0){
  outtextxy(176,350,"-");}
  else {   outtextxy(176,350,"+");}
outtextxy(185,350,"0.");
outtextxy(286,350,bit144);
outtextxy(274,350,"E");



(*surfaceptr[5]).check4();

if (c111 !=0){
c111 = 1/c111;
}
else if (c111==0){
outtextxy(200,368,"Infinite");
plane_bit=1;
}

if (plane_bit==0){
c444 = fcvt(c111,1,&bit14,&bit24);
outtextxy(200,368,c444);
itoa(bit14,bit144,10);
itoa(bit24,bit244,10);
  if (bit24 !=0){
  outtextxy(176,368,"-");}
  else {   outtextxy(176,368,"+");}
outtextxy(185,368,"0.");
outtextxy(286,368,bit144);
outtextxy(274,368,"E");
}
```

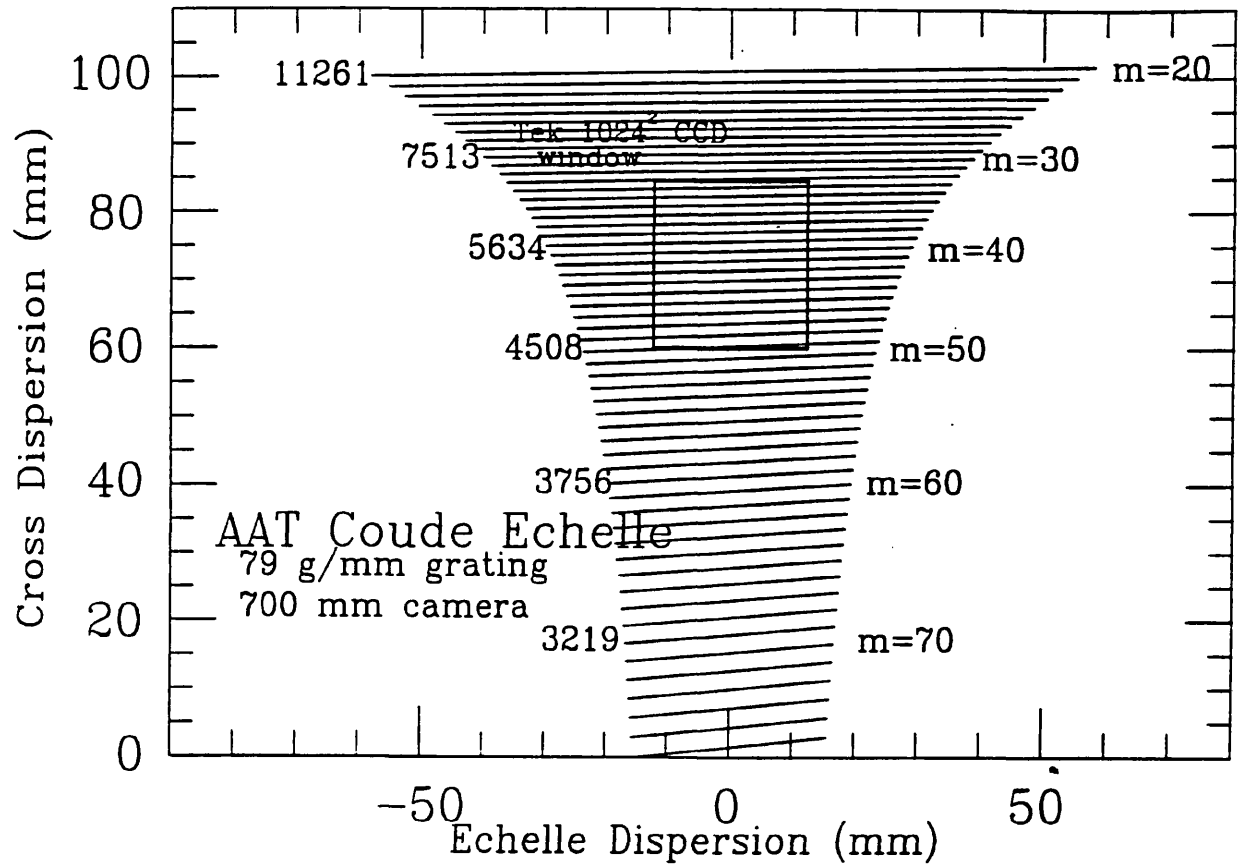
```
   plane_bit=0;

   (*surfaceptr[6]).check4();

   if (c111 !=0){
   c111 = 1/c111;
   }
   else if (c111==0){
   outtextxy(200,404,"Infinite");
   plane_bit=1;
   }

   if (plane_bit==0){
   c444 = fcvt(c111,1,&bit14,&bit24);
   outtextxy(200,404,c444);
   itoa(bit14,bit144,10);
   itoa(bit24,bit244,10);
     if (bit24 !=0){
     outtextxy(176,404,"-");}
     else {   outtextxy(176,404,"+");}
   outtextxy(185,404,"0.");
   outtextxy(286,404,bit144);
   outtextxy(274,404,"E");
   }
   plane_bit=0;

   (*surfaceptr[6]).check5();

   ddd1 = D111;

   c444 = fcvt(ddd1,1,&bit14,&bit24);
   outtextxy(200,386,c444);
   itoa(bit14,bit144,10);
   itoa(bit24,bit244,10);
     if (bit24 !=0){
     outtextxy(176,386,"-");}
     else {   outtextxy(176,386,"+");}
   outtextxy(185,386,"0.");
   outtextxy(286,386,bit144);
   outtextxy(274,386,"E");

   } // end doublet corrector

   (*surfaceptr[surface4]).check6();

// new 20/6/94
   //(*surfaceptr[surface4]).checka();

// new 20/6/94


   if (checka4 == 1){ // START PRINT aspheric coeffs

   (*surfaceptr[surface4]).checka();

// new 8/6/94
   if (a22 !=0){
   a444 = fcvt(a22,8,&bit14,&bit24);
   outtextxy(200,455,a444);
   itoa(bit14,bit144,10);
   itoa(bit24,bit244,10);

     if (bit24 !=0){
     outtextxy(176,455,"-");}
     else {   outtextxy(176,455,"+");}

   outtextxy(185,455,"0.");

   outtextxy(288,455,bit144);
   outtextxy(276,455,"E");
```

```
       // outtextxy(290,358,"mm");

}
// end new 8/6/94



   if (a44 != 0){
   a444 = fcvt(a44,16,&bit14,&bit24);
   outtextxy(200,422,a444);
   itoa(bit14,bit144,10);
   itoa(bit24,bit244,10);

     if (bit24 !=0){
     outtextxy(176,422,"-");}
     else {   outtextxy(176,422,"+");}

   outtextxy(185,422,"0.");

   outtextxy(288,422,bit144);
   outtextxy(276,422,"E");
      // outtextxy(290,358,"mm");

   a666b = fcvt(a66,23,&bit16,&bit26);
   outtextxy(200,437,a666b);
   itoa(bit16,bit166,10);

     if (bit26 !=0){
     outtextxy(176,437,"-");}
     else {   outtextxy(176,437,"+");}

   outtextxy(185,437,"0.");
   outtextxy(286,437,bit166);
   outtextxy(274,437,"E");

   } // end if a44 !=0

   y_shift =0;
   } //  END PRINT ASPHERIC COEFFS

   // NEW 94
   (*surfaceptr[surface4]).checka();
  // NEW 94

    if (e22 !=0){
   a444 = fcvt(e22,4,&bit14,&bit24);
   outtextxy(200,472,a444);
   itoa(bit14,bit144,10);
   itoa(bit24,bit244,10);

     if (bit24 !=0){
     outtextxy(176,472,"-");}
     else {   outtextxy(176,472,"+");}

   outtextxy(185,472,"0.");

   outtextxy(288,472,bit144);
   outtextxy(276,472,"E");
      // outtextxy(290,358,"mm");

}
// end new

   // scale factors

    itoa(scale_fc22,a666,10);
    outtextxy(15,268,a666);
    itoa(scale_fa4,a666,10);
    outtextxy(15,308,a666);

// new 20/6/94

    itoa(scale_fa2,a666,10);
    outtextxy(15,458,a666);

  // scale_fe =10;
    itoa(scale_fe,a666,10);
    outtextxy(15,473,a666);


// end new 20/6/94

    itoa(scale_fa6,a666,10);
    outtextxy(15,348,a666);

    itoa(d_inc,a666,10);
    outtextxy(562,400,a666);
    outtextxy(583,400,"mm");

    // new mag

    c444 = fcvt(mag,0,&bit14,&bit24);
    outtextxy(5,70,c444);

    //itoa(mag,a666,10);
    //outtextxy(5,70,a666);


    // new mag

    // PUPIL RADIUS AND FNO


    // END PUPIL RAD AND FNO


    ccc2 = (rad_pupil*2);

    c444 = fcvt(ccc2,0,&bit14,&bit24);
    outtextxy(480,4,c444);
    itoa(bit14,bit144,10);
    itoa(bit24,bit244,10);
    outtextxy(461,4,"0.");
    outtextxy(525,4,bit144);
    outtextxy(515,4,"E");
    outtextxy(415,4,"Pupil");

    ccc2 = efl/(rad_pupil*2);
    fno = ccc2;

    c444 = fcvt(ccc2,1,&bit14,&bit24);
    outtextxy(480,20,c444);
    itoa(bit14,bit144,10);
    itoa(bit24,bit244,10);
    outtextxy(461,20,"0.");
    outtextxy(525,20,bit144);
    outtextxy(515,20,"E");
    outtextxy(415,20,"F/no");

    ccc2 = (1000*(2.44*fno*0.55e-6));

    c444 = fcvt(ccc2,4,&bit14,&bit24);
    outtextxy(480,36,c444);
    itoa(bit14,bit144,10);
    itoa(bit24,bit244,10);
    outtextxy(461,36,"0.");
    outtextxy(525,36,bit144);
outtextxy(515,36,"E");
outtextxy(415,36,"Airy");

// MIRROR RAD OF CURV

(*surfaceptr[s5]).check4();
//ccc2 = c111;
if (c111 !=0){
ccc2 = 1/c111;
}
else if (c111==0){
outtextxy(443,438,"Infinite");
plane_bit=1;
}

if (plane_bit==0){
c444 = fcvt(ccc2,1,&bit14,&bit24);
outtextxy(443,438,c444);
itoa(bit14,bit144,10);
itoa(bit24,bit244,10);
  if (bit24 !=0){
  outtextxy(410,438,"-");}
  else {    outtextxy(410,438,"+");}
outtextxy(427,438,"0.");
outtextxy(520,438,bit144);
outtextxy(510,438,"E");
}
plane_bit=0;

 // end of scale_change =0


// END MIRROR RAD OF CURV
// end scale factors

setfillstyle(SOLID_FILL, 1);
midx = 523;
midy = 270;
bar(midx-10, midy-3, midx+10,midy+3);



} // ens exit not pressed if statement

scale_change =0;
} // end while loop

  field_angl = mem_ang;
  restorecrtmode();
   screen();
   printf("Total number of Graphics Loops = %d",counter);
   int c = getch();

  dont_clear =0;
  clrscr();

   return 0;

   }
```

Appendix H: Schematic of the HROS Layout



Appendix I: Example Echellogram from UCLES


AAT Coude Echelle
79 g/mm grating
700 mm camera
Tek 1024^2 CCD window
m=20
m=30
m=40
m=50
m=60
m=70
Cross Dispersion (mm)
Echelle Dispersion (mm)