University of Alberta

DEVELOPMENT OF A 32-CAPILLARY DIRECT-READING MULTI-WAVELENGTH SPECTROMETER FOR CAPILLARY ELECTROPHORESIS WITH LASER-INDUCED FLUORESCENCE DETECTION

BY

RONG JIANG ©

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirement for the degree of **Doctor of Philosophy**

Department of Chemistry

Edmonton, Alberta

Fall, 1998

0-612-34784-2

University of Alberta
Library Release Form

Name of Author: Rong Jiang

Title of Thesis: Development of a 32-Capillary Direct-Reading Multi-wavelength Spectrometer for Capillary Electrophoresis with Laser-Induced Fluorescence Detection

Degree: Doctor of Philosophy

Year this Degree Granted: 1998

Permission is hereby granted to the university of Alberta Library to reproduce single copies of this thesis and to lend or sell such copies for private, scholarly, or scientific research purposes only.

The author reserves all other publication and other rights in association with the copyright in the thesis, and except as hereinbefore provided, neither the thesis nor any substantial portion thereof may be printed or otherwise reproduced in any material form whatever without the author's prior written permission.

Rong Jiang

Rong Jiang
39 Bacon street
Waltham, MA 02451-4360
USA

Sept. 24, 1998

Date

UNIVERSITY OF ALBERTA

FACULTY OF GRADUATE STUDIES AND RESEARCH

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled Development of a 32-Capillary Direct-Reading Multi-Wavelength Spectrometer for Capillary Electrophoresis with Laser-Induced Fluorescence Detection submitted by Rong Jiang in partial fulfillment of the requirements for the degree of **Doctor of Philosophy**.

....

Dr. Norman J. Dovichi (Supervisor)

....

Dr. Frederick F. Cantwell

....

Dr. Mark T. McDermott

....

Dr. Monica M. Palcic

....

Dr. Martin J. Somerville

....

Dr. David H. Burns (External)

DATED: September 18, 1998

Abstract

A 32-capillary direct-reading multi-wavelength spectrometer was developed for separation and detection of fluorescently labeled biological macromolecules using capillary electrophoresis with laser-induced fluorescence detection. This spectrometer provides high sample throughput by analyzing 32 different samples simultaneously. In addition, the spectrometer is able to measure the spectral profile of a fluorescent dye. Discrimination of macromolecules with different labels is achieved by the comparison of the emission profiles of the labels. The spectrometer was characterized by the limit of detection, and further examined by four-color DNA sequencing with capillary gel electrophoresis and peptide analysis with capillary zone electrophoresis.

The excitation source of the spectrometer was a multi-line argon ion laser (488 nm, 514.5 nm). The laser beam was introduced into a 32-capillary sheath-flow cuvette to simultaneously excite fluorescence from the sample streams of 32 capillaries. The fluorescence was then color-dispersed by the dispersion optics, and the spectra were detected by a CCD camera which was controlled by the data acquisition software through a Macintosh computer.

The optical design of the dispersion optics was assisted by calculation and computer-modeling. The spectral dispersion was achieved by the use of two identical camera lenses (55 mm f/1.4) and an F2 equilateral prism. The fluorescence light collected from the cuvette was collimated by the first lens, color-dispersed by the prism, and then imaged onto the detector by the second lens.

Data acquisition was performed with a software written in C. The software controls the exposure and data transfer of the CCD camera, writes spectral data onto hard disk, and allows the user to select the effective exposure regions and to monitor one spectral channel of any selected capillary. Data analysis was also performed automatically by using the procedures written in Igor Pro.

In addition to the instrument development, separation conditions of DNA sequencing with linear polyacrylamide were optimized using a 5-capillary instrument. Sequencing accuracy and read-length are increased by operating capillaries at high temperatures; about 600 bases can be sequenced within 2 h at a 150 V/cm electric field and 60 °C in 5%T polyacrylamide.

to my mother and my husband

Acknowledgments

I am most grateful to my supervisor, Dr. Norman Dovichi, for his excellent guidance, support, encouragement, and kindness throughout my study.

I thank all the members in NLLL. In particular, I would like to thank Dr. JianZhong Zhang for his help on the optics and instrumentation; Mr. Darren Lewis for his suggestions and help on the data acquisition software design; Mr. Eric Carpenter for his tutoring in C programming; Dr. John Crabtree and Dr. Sue Bay for their help on the sheath-flow cuvette design and manufacture.

Lastly, I would like to thank my husband for his love and encouragement, and my mother, sister, and brother back home for their unconditional love and support.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

ABI:	Applied Biosystems, Inc.
ANSI:	American National Standard Institute
AO:	acousto-optic
APS:	ammonium persulfate
Ar:	argon
CBQ:	(3-(p-carboxybenzoyl)quinoline-2-carboxaldehyde
CCD:	charge-coupled device
CE:	capillary electrophoresis
CZE:	capillary zone electrophoresis
DAQ:	data acquisition
DNA:	deoxyribonucleic acid
FITC:	fluorescein isothiocyanate
FQ:	(3-(2-furoyl)quinoline-2-carboxaldehyde
FWHM:	full width at half maximum
GRIN:	gradient index
ID:	inner diameter
KCN:	potassium cyanide
KP:	shortpass
LIF:	laser-induced fluorescence
LP:	longpass
LOD:	limit of detection
LOD_c:	limit of detection in concentration
LOD_m:	limit of detection in number of molecules
N.A.:	numerical aperture
OD:	outer diameter

PMT:	photomultiplier tube
R6G:	rhodamine 6G
RF:	radio frequency
SDS:	sodium dodecyl sulfate
SLR:	single-lens reflex
TEMED:	tetramethylethylenediamine
TIR:	total internal reflection
L:	capillary length
Δh:	the height difference between the sheath-flow buffer reservoir and the waste container
δ_{min}:	minimum deviation
ϕ_1:	incident angle to the first face of the prism

Chapter 1. Introduction

1.1. CAPILLARY ELECTROPHORESIS

Electrophoresis can be described most simply as the movement of charged particles in an electric field. In the 1930s, Tiselius inaugurated the modern science of electrophoresis with an elegant series of experiments in a tube apparatus.[1] Tiselius showed that protein molecules from human serum could be separated in free solution on the basis of differences in charge and mass, under the influence of an electric field. In free zone electrophoresis, Joule heating causes convective disturbances, which severely degrade resolution. To overcome this problem many workers employed solid supports, such as cellulose acetate, agarose, and polyacrylamide.

An alternative, capillary zone electrophoresis, has been popularized recently by Jorgensen and Lukacs.[2] The high surface-to-volume ratio of micro-bore fused silica capillary, which has a typical inner diameter (ID) of 25 to 100 μm, provides a rapid transfer of Joule heat generated during electrophoresis. This allows the use of high electric fields for fast separation. It has been demonstrated that the high voltages used in capillaries (typically 10 to 30 kV) can produce both rapid separations and good resolution.[3] The high surface-to-volume ratio also obviates the need for an anticonvective medium, and allows for efficient electrophoretical separation in free solution. Additionally, the use of capillaries has promoted electrophoresis into an instrumental technique. The flexible capillaries are easily interfaced with sample vial such as a microtitre plate and have the potential for automatic sample loading and analyzing.

1.1.1 Capillary Zone Electrophoresis

In capillary zone electrophoresis (also called open-tube capillary electrophoresis), the capillary is filled with separation buffer. Sample is applied only briefly, after which the injection end of the capillary is re-immersed in the separation buffer and an electric field is applied for separation. The electrophoretic mobility (μ_{el}) of an ion is proportional to its charge-to-radius ratio, and the direction of the electrophoretic movement depends on the type of charge the ion carries. For example, a negatively charged ion electrophoretically moves towards the anode, whereas a positively charged ion electrophoretically moves towards the cathode, and a neutral particle shows no electrophoretic movement.

In an uncoated fused silica capillary, the inner surface has many silanol (Si-OH) groups which are deprotonated at pH > 2.[4] The negatively charged inner surface of the capillary introduces a movement of the bulk solution caused by the migration of the capillary wall's counterions towards the cathode. This is called electroosmotic flow (μ_{eo}), which is a flat and plug-like flow and does not cause band broadening. The electroosmotic flow is an additional component to an ion's movement. The apparent mobility of an ion depends on the direction and value of its electrophoretic flow and the electroosmotic flow. For example, in an uncoated capillary filled with basic buffer solution, the electroosmotic flow is towards cathode. An anion electrophoretically moves toward anode, so the apparent mobility of the anion is $\mu_{el} - \mu_{eo}$. The apparent mobility of a cation is $\mu_{el} + \mu_{eo}$, while that of a neutral particle is μ_{eo}. In some applications, electroosmotic flow is useful because anions, cations, and even neutral particles can be analyzed simultaneously. In other applications, the capillary should be well coated to eliminate electroosmotic flow in order to improve reproducibility.

1.1.2 Capillary Gel Electrophoresis

In open-tube capillary electrophoresis, ions with the same charge polarity are separated according to their charge-to-radius ratio. Unfortunately, this charge-to-radius ratio is essentially unchanged for DNA fragments of different length.[5] A sieving medium such as a polymer solution is poured into the capillary to provide size-dependent separation of DNA fragments. When an electric field is applied along the capillary, the negatively charged DNA molecules migrate towards the anode and are separated according to their size while they go through the pores formed by the polymer chains. Short fragments go through the pores easily and migrate quickly, while large fragments are retained by the polymer pores and migrate relatively slowly.

Since the first introduction of polyacrylamide gel into fused silica capillary for separation of single stranded DNA by Karger,[6] capillary gel electrophoresis has become a powerful tool for separation of DNA sequencing fragments.[7] Recently, multiple capillary DNA sequencers have been investigated by many researchers.[8-12] The multiple capillary sequencer using sheath-flow technology developed by Dovichi and Kamdara is being commercialized by Perkin-Elmer.[8-10]

1.2 DNA SEQUENCING

1.2.1. Slab Gel Sequencer

Autoradiographic Detection

DNA sequencing is used to identify genes and mutations, to confirm successful site-directed mutagenesis, and to analyze phage-display libraries. Two sequencing techniques were reported in 1977, one a chemical degradation method developed by Maxam and Gilbert and the other an enzymatic method developed by Sanger's group.[13,14] The vast majority of DNA sequence is determined by use of Sanger's chain-termination method. In the original version of the technology, a set of radioactively labeled DNA fragments was generated in four reactions. Each reaction used a chain-terminating dideoxynucleotide, which generated a sequencing ladder wherein all fragments terminated at all locations of that nucleotide. The sequencing ladders were separated by size in adjacent lanes of a high resolution polyacrylamide gel and detected by use of autoradiography. The sequence was interpreted from the pattern of alternating bands in the lanes corresponding to the terminal base of the fragment.

The use of radioactive labels and autoradiographic detection suffers from several limitations. Although the radioactive isotopes have a short half-life and low activity, disposal remains a concern. Because of their short half-life, a regular supply of reagents is required, and the reagents can not be stored for more than a few days. Autoradiography is labor intensive, requiring manual manipulation of film and, more importantly, manual interpretation of the resulting gel patterns.

Laser-Induced Fluorescence Detection

Advances in sequencing technology occurred between 1986 and 1987 when Smith, Ansorge, and Prober replaced the radioactive labels with fluorescence labels and replaced autoradiography with laser-induced fluorescence detection.[15-17] Laser-induced fluorescence (LIF) was particularly important because it allowed the direct reading of the electropherogram into a computer for automated sequence determination.

In Smith's automated sequencer,[15] primers labeled with different fluorescent dyes were used in the four sequencing reactions. The reaction products were pooled and separated in a single lane of a sequencing gel. Fluorescence was sequentially excited with two laser lines and detected in four spectral channels. The terminal nucleotide was identified based on the fluorescence signature of the associated fluorescent dye. Prober reported a more sophisticated version of a four-color sequencer based on the use of fluorescently labeled dideoxynucleotide chain terminator.[17] Since the fluorescence dye was associated with the termination event, only one sequencing reaction was required in the presence of the four chain-terminating nucleotides. Unfortunately, the dye-set used in Prober's method suffered from severe spectral overlap, which resulted in error-prone identification of sequence. Also, Prober's method used a simple two-color spectrometer to discriminate the spectrum from the four dyes. The ratio of fluorescence intensity in the red and blue spectral channel was used to determine the DNA sequence, which proved to suffer from poor accuracy. Finally, in Ansorge's method,[16] a single fluorescence dye was used for the four sequencing reactions. The reaction products were separated in four adjacent lanes of a sequencing gel and sequence identification was similar to the classic autoradiography technique.

Smith's, Prober's and Ansorge's fluorescence sequencer were commercialized by Applied Biosystems (ABI), Dupont, and Pharmacia, respectively. The Dupont sequencing technology was eventually sold to ABI, who replaced the original dye set with newer dyes that have wider spectral dispersion.

The set of four dye-labeled primers developed by ABI is widely used in DNA sequencing kit. The four fluorescent dyes are FAM (fluorescein), JOE (2',7'-dimethoxy-4',5'-dichloro-fluorescein), TAMRA (tetramethylrhodamine), and ROX (rhodamine χ). FAM and JOE are excited by 488 nm blue laser; TAMRA and ROX are excited by 514 nm green laser. The emission spectra of these four dyes were reported[18], and the maximum emission of FAM, JOE, TAMRA, and ROX occurs at 540 nm, 560 nm, 580 nm, and 610 nm, respectively.

The conventional slab gel systems operate with relatively thick gels, up to 400 μm, which result in high current operation during electrophoresis and unacceptable Joule heating at high electric fields. Therefore, these conventional slab gel systems can be operated only at low electric field, typically no more than 50 V/cm, which result in a long analysis time. More recent slab gel systems operate with thinner gels that can be operated safely at electric fields approaching 150 V/cm.[19] As an example of typical performance, the ABI model 377 instrument runs 96 lanes simultaneously on a slab gel to produce sequencing rates of 200 bases/h/lane or 20000 bases/h/slab.[20]

1.2.2. Capillary Gel Sequencer

In general, the excellent thermal properties of capillaries allows the use of higher electric field, which produce faster and higher resolution separations compared with slab gels. However, the co-migration of longer fragments limit the sequencing read-length at higher electric field,[21] which is crucial in large-scale sequencing efforts. It turns out that capillaries do not appear to provide a fundamental improvement in separation performance over ultrathin slab gels for DNA sequencing.[22] However, capillary gel electrophoresis does offer practical advantages.

First, it is possible to replace the separation medium between electrophoresis runs in capillaries to automate continuous operation of the sequencer. Second, the flexible capillaries are easily interfaced with a microtitre plate; sample loading and injection can be automated. Third, very dense arrays of capillaries can be generated for the development of large scale sequencers. The largest commercial slab gel system operates 96 sample channels simultaneously. It is difficult to generate larger numbers of channels on a slab gel. Especially, most current slab gel sequencers use a scanning technique for detection: as the number of channels increases, the duty cycle decreases and the signal-to-noise ratio of detection degrades. In contrast, large numbers of capillaries can be packaged in a small region that can be monitored simultaneously, significantly enhancing detection efficiency.

1.3 FLUORESCENCE DETECTION

There are two methods to detect the laser-induced fluorescence in capillary electrophoresis. One is the on-column detection where the fluorescence is detected before the sample band runs off the capillary tip.[11,12] In this method, each end of the capillary is immersed in a buffer reservoir, a small window is created near the detection end by burning off the polyimide coating outside the capillary. A laser beam is focused on the window of the capillary and fluorescence is collected from the same window. The light scattering at the capillary surfaces is significant in the on-column detection due to the change in refractive index at the interface between air, glass, and buffer solution. Especially in a multiple capillary array system, a critical issue is how to achieve the uniform illumination of all the capillaries without producing a large background light scatter signal.

Another method for LIF detection is the use of sheath-flow for post-column detection.[8-10] The post-column fluorescence detector based on the sheath-flow cuvette effectively eliminates the scattered light from capillary interfaces, and lowers the background signal to nearly Raman scatter limited conditions.[23,24]

A single capillary sheath-flow cuvette system developed by Dovichi's group is shown in Figure 1.1. The sheath-flow cuvette serves as a flow chamber as well as an optical detection window. The flow chamber is typically 200 μm square in cross section, 20 mm long, and has a 2 mm thick window. A 190 μm OD capillary is placed within the square chamber. As analyte elutes from the tip of the capillary, it passes as a narrow stream in the center of the square flow chamber, surrounded by a sheath stream consisting of conducting buffer. The sheath-flow is generated by gravity-driven siphon flow and enters the flow chamber from the top of the cuvette and goes to waste at the bottom. Flow rate is low, typically 0.15 mL/h. The laser beam is focused on the sample stream just beneath the capillary tip, and fluorescence is collected at right angles to the laser beam. Fluorescence is collected with a high efficiency microscope objective and imaged onto a photodetector. Bandpass filters are normally used for detection of a specific dye

The high optical quality quartz windows of the cuvette produce much less light scatter than does an on-column detector. Also by transferring the analyte to the moving sheath stream, the linear velocity and, hence the illumination time, is independent of fragment length. The post-column detector based on the sheath-flow cuvette results in high sensitivity. Separation of zeptomole (10^{-21}) quantities of fluorescently labeled amino acids by capillary zone electrophoresis and separation of zeptomole quantities of the products of a single dideoxynucleotide reaction have been achieved by the use of a single-spectral-channel fluorescence detector based on the cuvette.[23, 25, 26]

The object of this thesis was to build a multiple capillary instrument based on post-column detection using the sheath-flow cuvette system. Modification of the sheath-flow cuvette to accommodate a large number of capillaries and the design of an optical system which provides information of spectral profile are the subject of this thesis.

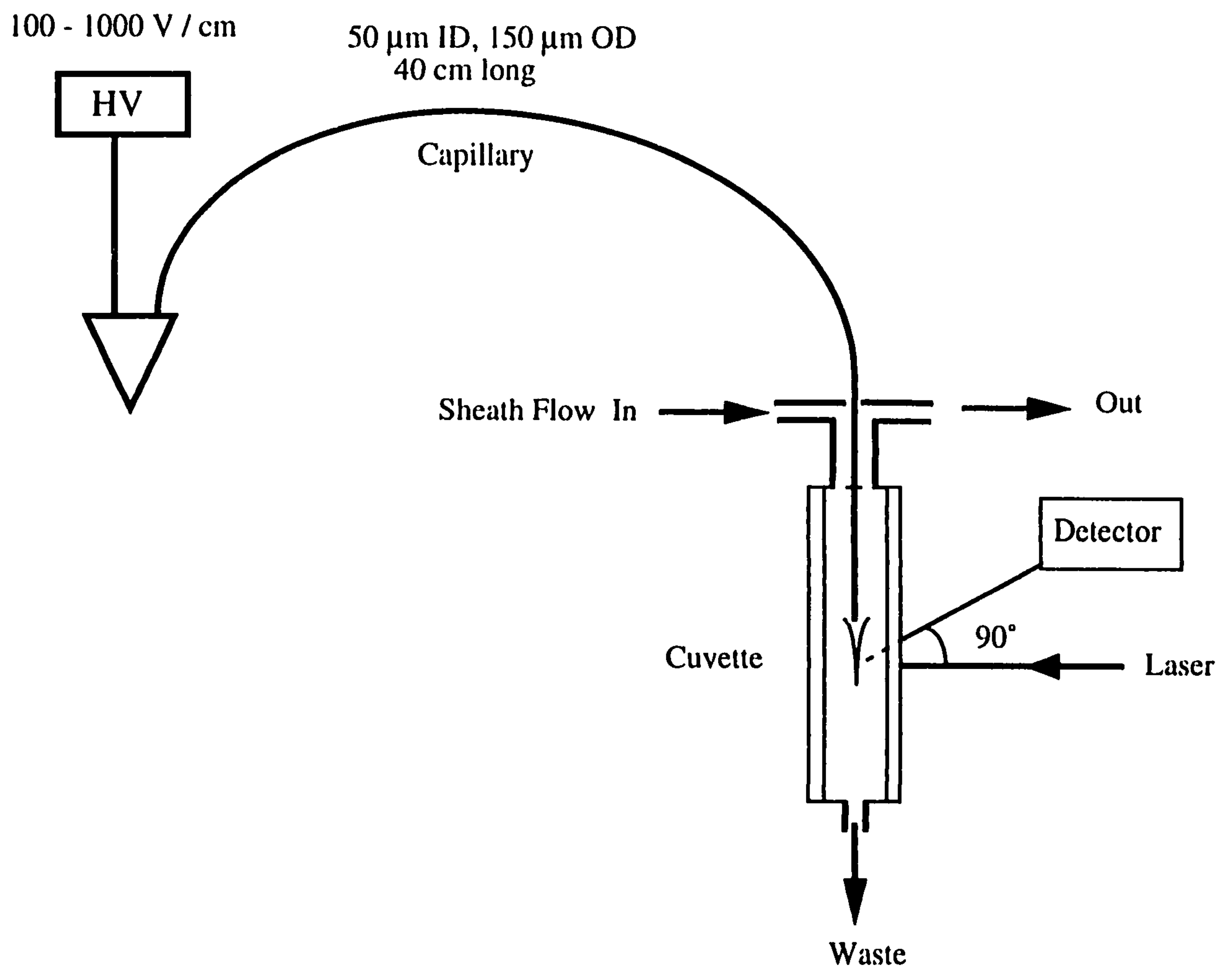


Figure 1.1 Sheath-flow cuvette for post-column detection

REFERENCES

1. A. Tiselius, Trans. Faraday Soc. 1937, 33, 524-531

2. J. W. Jorgensen, K. D. Lukacs, Science 1983, 222, 266-272

3. B. L. Karger, Nature 1989, 339, 641-642

4. H. J. Issaq, G. M. Janini, K. C. Chan, and Z. E. Rassi, Advances in Chromatography 1995, 35, 101-169

5. B. M. Olivera, P. Baine and N. Davidson, Biopolymers 1964, 2, 245-257

6. A. S. Cohen, D. R. Najarian, A. Paulus, A. Guttmann, J. A. Smith, B. L. Karger, Proc. Nat. Acad. Sci. U.S.A., 1988, 85, 9660-9663

7. N. J. Dovichi, in CRC Handbook of Capillary Electrophoresis, E. Landers (Ed.), CRC Press, Boca Raton, 1993, Chapter14, pp. 369-387

8. N. J. Dovichi, J. Z. Zhang, US Patent 5,439,578, 8 August 1995

9. N. J. Dovichi, J. Z. Zhang, US Patent 5,584,982, 17 December 1996

10. S. Takahashi, K. Murakami, T. Anazawa, and H. Kamdara, Anal. Chem. 1994. 66, 1021-1026

11. K. Ueno, E. S. Yeung, Anal. Chem. 1994, 66, 1424-1431

12. M. A. Quesada, S. Zhang, Electrophoresis 1996, 17, 1841-1851

13. A. M. Maxam, W. Gilbert, Proc. Natl. Acad. Sci. U. S. A., 1977, 74, 560-564

14. F. Sanger, S. Nicklen, A. R. Coulson, Proc. Natl. Acad. Sci. U. S. A. 1977. 74, 5463-5467

15. L. M. Smith, J. Z. Sanders, R. J. Kaiser, P. Hughes, C. Dodd, C. R. Connell, C. Heiner, S. B. Kent, L. E. Hood, Nature 1986, 321, 674-679

16. W. Ansorge, B. S. Sproat, J. Stegeman, C. Schwager, J. Biochem. Biophys. Meth. 1986, 13, 315-317

17. J. M. Prober, G. L. Trainor, R. J. Dam, F. W. Hobbs, C. W. Robertson, R. J. Zagursky, A. J. Cocuzza, M. A. Jensen, K. Baumeister, Science 1987, 238, 336-341

18. Applied Biosystems, Inc., Model 370 product literature

19. R. L. Brumley and L. M. Smith, Nuc. Acids Res. 1991, 19, 4121-4126

20. http://www2.perkin-elmer.com:80/ga/237205/237205.html

21. J. Yan, N. Best, J. Z. Zhang, H. Ren, R. Jiang, J. Hou, N. J. Dovichi, Electrophoresis 1996, 17, 1037-1045

22. N. J. Dovichi, Electrophoresis 1997, 18, 2748-2754

23. S. Wu and N. J. Dovichi, J. Chromatography, 1989, 180, 141-155

24. D. Y. Chen, H. P. Swerdlow, H. R. Harke, J. Z. Zhang, and N. J. Dovichi. J. Chromatography 1991, 559, 237-246

25. H. Swerdlow, S. Wu, H. Harke, N. J. Dovichi, J. Chromatogr.1990, 516, 61-67

26. (a) Y. F. Cheng, N. J. Dovichi, Science 1988, 242, 562-567 (b) Y. F. Cheng. S. Wu, D. Y. Chen, N. J. Dovichi, Anal. Chem. 1990, 62, 496-503

Chapter 2. DNA Sequencing in a 5-Capillary Instrument

2.1. INTRODUCTION

2.1.1 Characteristics of a DNA Sequencing System

Sequencing read-length and separation speed are two important criteria in evaluating a DNA sequencing run. Fast separation of DNA is the most obvious advantage of capillary gel electrophoresis over slab gel electrophoresis. The small inner diameter (ID) of the capillary allows efficient heat dissipation, which allows the use of high electric fields for fast separation.

Read-length, which is the number of bases of sequence determined per electropherogram, is probably the most important parameter in large-scale sequencing efforts.[1] Long read-lengths reduce the cost of finishing the sequence of an unknown piece of DNA in part because they minimize the number of samples that must be prepared for sequencing. Most importantly, long read-length minimizes computational costs to assemble a finished sequence from a randomly cloned template.[2]

An important phenomenon limits read-length. At the high electric fields used to produce rapid DNA separations, the spacing between peaks decreases for longer fragments. The mobility of long fragments reaches an asympotic limit, which determines the longest fragment that can be separated. This limit moves to shorter fragments as the electric field increases.[3] In cross-linked gels and at room temperature, separation at 800 V/cm resolves fragments 250 bases long in 7 min, while separation at 100 V/cm resolves fragments 650 bases long in 4 h.[4, 5] Modest electric fields are required to obtain long sequencing read-length at the expense of separation time.

2.1.2 Separation Matrices in Capillary Gel Electrophoresis

There are several sieving media employed for DNA sequencing. Cross-linked polyacrylamide is stable and viscous and is widely used in the slab gel DNA sequencer.

The use of cross-linked polymers in capillary electrophoresis leads to a serious limitation: the entire capillary must be replaced when the separation medium has degraded. This replacement can be quite tedious because of alignment constraints of the optical system with the narrow diameter capillaries. The use of low viscosity polymer makes it possible to replace the separation medium without replacing the capillary or realigning the optical system. Noncross-linked polyacrylamide is a polymer with low viscosity and is widely used in capillary electrophoresis.

Polymerization of Acrylamide

The polymerization of acrylamide is a free-radical reaction as shown in Figure 2.1.[6] Two reagents are used to initiate the chain extension reaction: tetramethylethylenediamine (TEMED) and ammonium persulfate (APS). TEMED loses a hydrogen and becomes a free-radical when it is oxidized by persulfate, the free-radical then reacts with acrylamide based on free-radical reaction. The TEMED radical can terminate the polymerization reaction as well as initiate it. Therefore, these two reagents have to be added into acrylamide at a proper ratio. The concentrations of these two reagents and the acrylamide determine the molecular weight of the polymer. The concentration of linear polyacrylamide is often labeled as %T.

$$\%T = W_{\text{Acrylamide}}\,(g) / V\,(mL) \times 100\% \qquad (2\text{-}1)$$

where $W_{Acrylamide}$ is the weight of acrylamide, and V is the total volume of the solution in mL.

Since the polymerization reaction of acrylamide is a free-radical reaction, anything having an unpaired electron may terminate the reaction and thus affect the chain length of the polymer. Oxygen is a paramagnetic species and is one factor which affects the chain length of polyacrylamide.

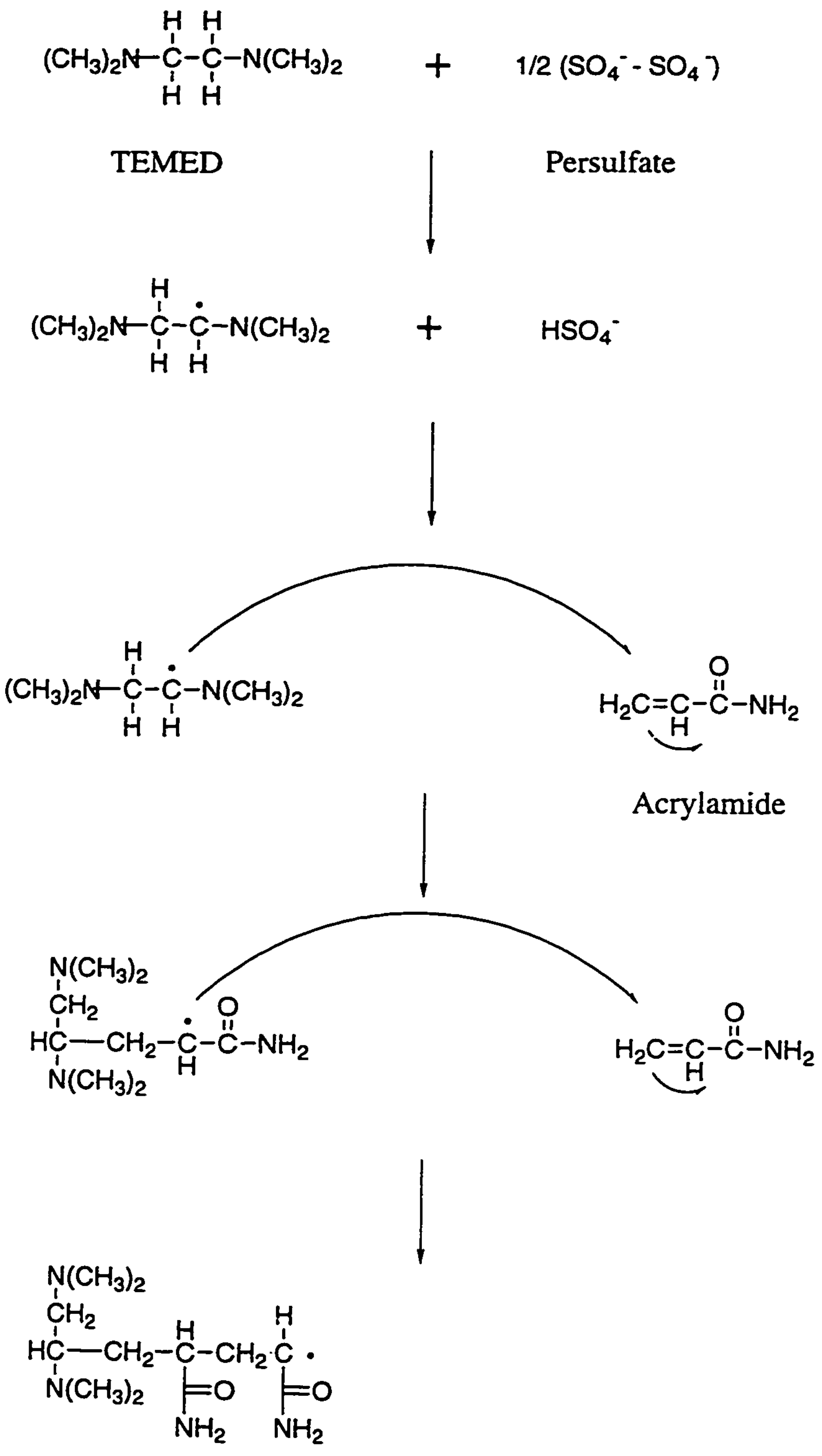


Figure 2.1 Polymerization of acrylamide

DNA Sequencing With Noncross-Linked Polymer

Noncross-linked polyacrylamide is popularly employed for DNA sequencing in capillary gel electrophoresis. Karger's group reported the first use of noncross-linked polyacrylamide for capillary electrophoresis separation of DNA.[7] Early reports on the use of noncross-linked polyacrylamide used a high concentration of polymer and high electric field. This combination resulted in short read-length. In one report, less than 200 bases of sequence were determined with 9%T polyacrylamide.[8] Pentoney and co-workers reported a DNA sequence in 10%T noncross-linked polyacrylamide at an electric field of 300 V/cm.[9] The sequence could not be determined for fragments longer than 300 bases. Similar results were reported by Mathies' group with 9%T noncross-linked polyacrylamide.[10]

The effect of the electric field on the co-migration of longer fragment in noncross-linked polyacrylamide has been investigated by Dovichi's group.[3] As in the cross-linked material, the onset of co-migration becomes more problematic at high electric field. Modest electric fields are required to obtain long sequencing read-length. Typically, a 200 V/cm electric field is used for 5%T noncross-linked polyacrylamide.

Other noncross-linked polymers are also employed for DNA sequencing. Noncross-linked polyethylene oxide is commercially available and has been used for DNA sequencing in Yeung's group.[11] The use of polyethylene glycol has been reported by Menchen,[12] who modified the polymer by end-capping it with a fluorocarbon tail. This material is commercially available from ABI and is used in their single capillary sequencer.

In this chapter, the separation conditions for DNA sequencing with noncross-linked polyacrylamide using a 5-capillary instrument are investigated.

2.2 EXPERIMENTAL

The first multiple capillary instrument built in this research group was a 5-capillary instrument.[13-15] This instrument is mainly used as a DNA sequencer. The four different base-termination reactions are performed with primers labeled by four different dyes: Fam for C-terminated fragments, Joe for A, Tamra for G, and Rox for T. The DNA sequencing fragments are separated by noncross-linked polyacrylamide in capillary gel electrophoresis according to the size of each fragment, and the terminated bases are determined by the maximum emission wavelengths of each dye.

2.2.1 The 5-Capillary DNA Sequencer

The schematic diagram of the 5-capillary instrument is shown in Figure 2.2. The five capillaries were aligned in an array in a rectangular sheath-flow cuvette. A 1 X TBE buffer was driven by gravity through the interstitial spaces between the capillaries. The buffer draws migrating DNA fragments from the tip of the capillary downstream. A laser beam excited fluorescence from the sample streams immediately downstream from the capillary exit. Fluorescence was collected at right angles to the laser beam with a 18 X 0.45 numerical aperture microscope objective (Melles Griot, CA, USA) and imaged onto an array of five 1.8 mm diameter GRIN lens (part number FCM-OF-100-063, NSG, Tokyo, Japan), which were coupled to 50-mm core optical fibers. The fibers were connected to an avalanche-photodiode single photon counting module (SPCM-100-PQ, EG&G Canada Ltd., Quebéc, Canada). The output of the photon counting modules was digitized by a Macintosh computer.

Two lasers, a He-Ne green laser (λ = 543.5 nm, Melles Griot, Carlsbad, CA, USA) operating at 1.5 mW and an air-cooled argon (Ar) ion blue laser (488 nm, model 2011-20SL, Uniphase, CA, USA) operating at 10 mW, were used for the excitation of the four dyes. The two beams were made collinear by use of a dichroic filter. To spectrally resolve fluorescence from the four dyes, a locally constructed four-channel filter wheel was placed between the collection optics and the GRIN lens. The 4-inch-diameter filter wheel was equipped with interference filters centered at 540, 560, 580, and 610 nm with a 10-nm bandwidth. To minimize interference of the green laser with the detection at 540 nm

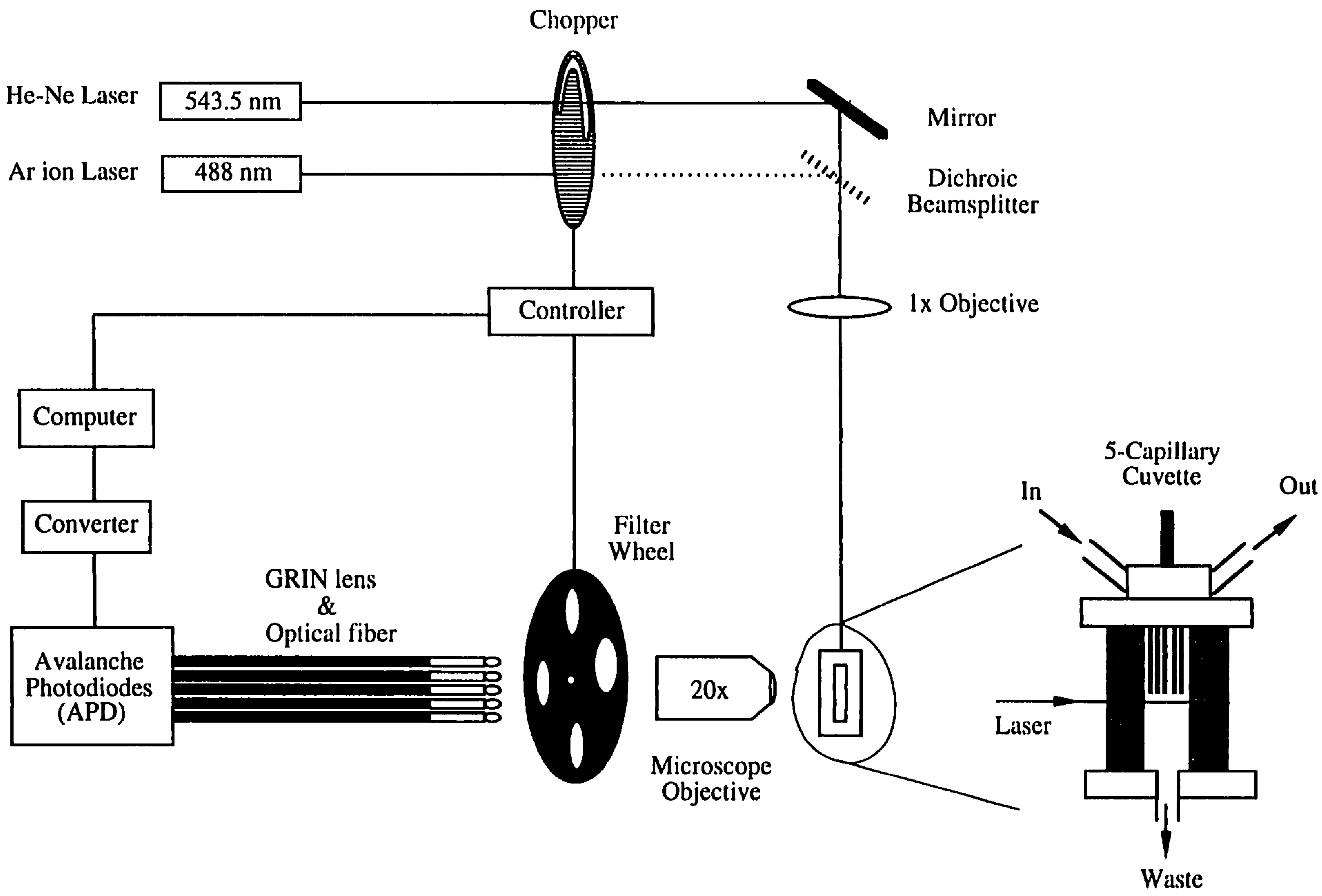


Figure 2.2 The 5-capillary instrument

and to minimize interference of Raman scatter generated by the blue laser with detection at 580 nm, a sector wheel was synchronized with the filter wheel and used to alternately transmit the blue and the green laser beams, both wheels rotated at 6 Hz. Excitation at 488 nm was synchronized with detection at 540 and 560 nm. Excitation at 543.5 nm was synchronized with detection at 580 and 610 nm. The wheels were synchronized by use of stepper motors and a signal control circuit.

2.2.2 Sample Preparation

All samples were four-color M13mp18 sequencing samples. A Sequitherm Long-Read Cycle Sequencing kit was used for cycle sequencing with fluorescently labeled primers (Applied Biosystems, Inc., -21M13, Fam-C, Joe-A, Tamra-G, Rox-T). Cycle sequencing was performed using 30 cycles on a PRT-100 Programmable Thermal Controller (MJ Research, Inc., Watertown, MA, U.S.A.); each cycle consists of 15 s at 95 °C and 90 s at 70 °C. Reaction products were pooled and immediately ethanol-precipitated. The dried pellet was re-suspended in 1.5 µL of formamide.

2.2.3 Preparation of Gel-Filled Capillary

The fused silica capillaries used in the 5-capillary DNA sequencer are 50 µm ID, 140 µm OD and typically 40 cm in length (Polymicro Technologies, Phoenix, AZ, USA). The capillary was first treated with a 2% [γ-(methacryloxy)propyl]trimethoxysilane solution prepared in 95% ethanol 5% water solution. One end of the capillary was connected to a water aspirator, and the other end was dipped into the silane solution. Once the capillary was filled with silane solution, the reaction was allowed to proceed for 20 min. Vacuum was then used to draw air through the capillary to flush the residual silane solution.

Polyacrylamide was prepared by mixing 5 mL of 5% acrylamide, 7 M urea, and 1 X TBE buffer which is a mixture of 89 mM tris(hydroxymethyl)aminomethane, 89 mM borate acid, and 2 mM EDTA at pH 8.5. The solution was typically degassed for

30 min under vacuum provided by a water aspirator. Polymerization was initiated by addition of 2 μL of TEMED and 20 μL of 10% APS. The solution was mixed and introduced into the capillary under vacuum. Polymerization occurs within the capillary and appears complete within 30 min; however, capillaries are typically stored overnight at room temperature before use.

The gel-filled capillaries were then inserted into the 5-capillary cuvette and glued together by epoxide on the top of the cuvette holder. DNA sample was typically injected into capillaries at 100 V/cm for 30 s. An electric field (typically 200 V/cm for 5%T) provided by the high voltage power supply (Model: CZE1000PN30, SPELLMAN High Voltage Electronics Corp., Plainview, NY, USA) was then applied across the capillary for electrophoresis. In high temperature experiments, the capillaries were encased in a Plexiglas box, connected to a temperature controller (AIR THERM, World Precision Instruments, Inc., Sarasota, FL, USA). This controller bathed the capillary in hot air with an accuracy of ± 0.5 °C.

2.3 RESULTS AND DISCUSSION

Separation conditions of DNA sequencing with noncross-linked denaturing polyacrylamide have been studied from four aspects: degassing method, gel age, gel concentration, and running temperature.

2.3.1 Degassing Conditions

As mentioned in Section 2.2.3, the polymerization reaction was performed in aqueous solution. The 5 mL 5%T solution was first degassed for 20 to 30 min, the polymerization reaction was then initiated by adding TEMED and APS. Here, two degassing methods — vacuum degas and Ar degas — were examined for degassing the monomer solution (5%T) before the initiation of the polymerization reaction. Resolution of the gels made by the two different methods is compared in Figure 2.3 using a four-color M13mp18 sequencing sample.

The gel used for Figure 2.3 a and c was made by bubbling Ar into the 5%T solution for 30 min followed by initiation of the polymerization reaction, whereas gel for Figure 2.3 b and d was made by applying a vacuum to the solution for 30 min followed by initiating the polymerization. Gels prepared by the two different methods were compared in the 5-capillary instrument. The sequencing sample was injected into the five capillaries at 100 V/cm field for 35 s, then the sequencing data from the five capillaries were monitored simultaneously via capillary electrophoresis at 200 V/cm electric field in 1 X TBE buffer.

The 101 bases sequencing data of M13mp18 template from 115 to 215 bases are compared in Figure 2.3. The vacuum degassed gel gives baseline resolution across this section of electropherogram (Figure 2.3 b and d). The Ar degassed gel provides a lower resolution especially for fragments longer than 180 bases. More bubble peaks are observed from an Ar degassed gel than a vacuum degassed gel, especially in a high temperature run. because gas solubility decreases at elevated temperatures. In Figure 2.3, four bubble peaks are produced in this 100 base sequencing run even at room temperature.

The purpose of degassing the gel is to reduce the concentration of oxygen in the solution because oxygen can terminate the polymerization reaction of acrylamide. Oxygen in the air dissolves into the solution until the dissolved oxygen and the oxygen in the air reach an equilibrium.

$$O_2\,(aq.) \Leftrightarrow O_2\,(gas) \qquad (2\text{-}2)$$

When vacuum is applied, the concentration of oxygen in the gas phase continuously decreases so that the equilibrium (2-2) moves to the right. As a result, the dissolved oxygen is continuously released from the solution and pumped out.

In the case of Ar degassing, the Ar gas forms a gas phase on the top of the solution, and this Ar gas phase isolates the solution from the air to keep oxygen in the air from

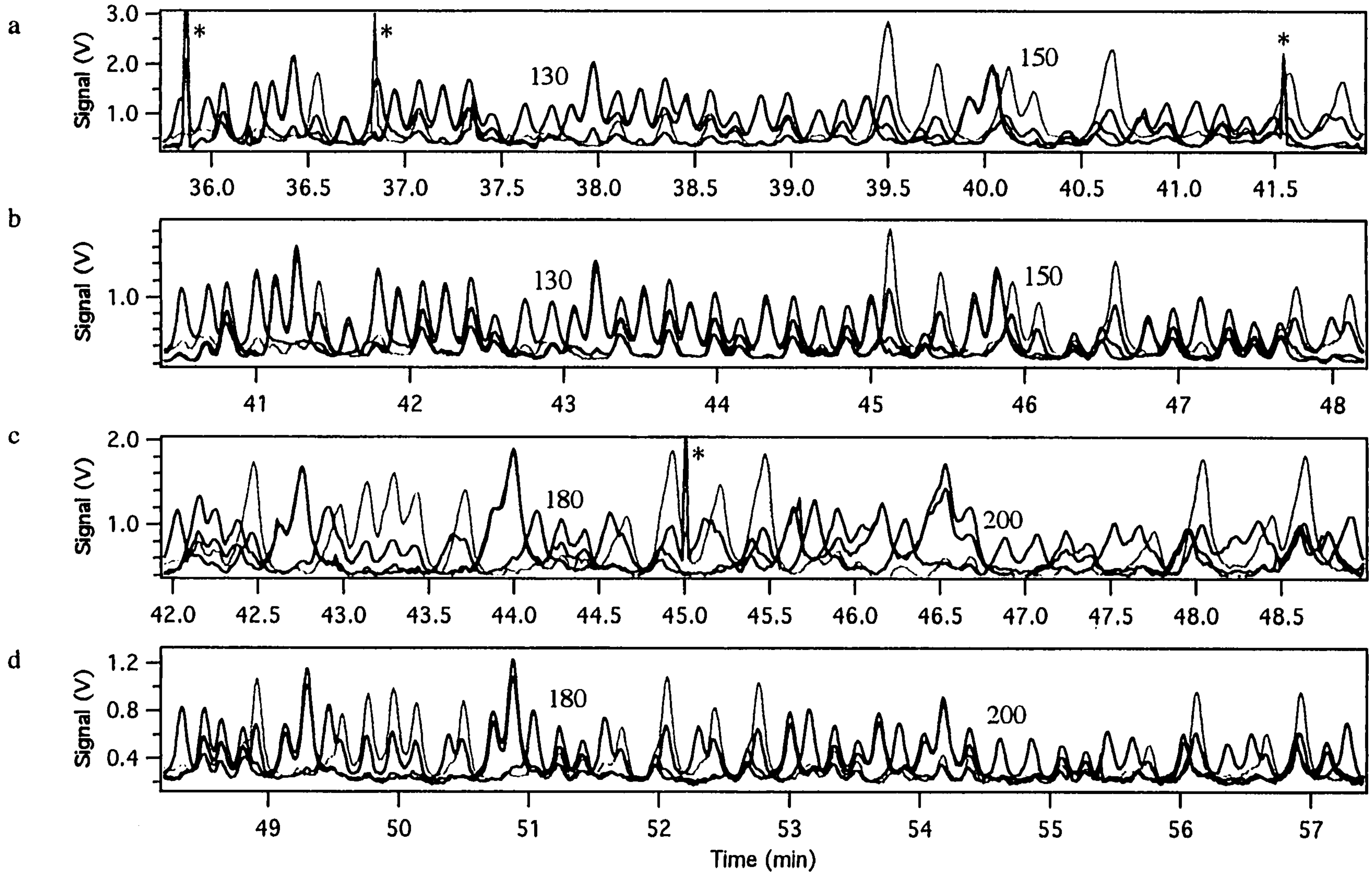


Figure 2.3 Effect of degas conditions on the resolution of gel
a,c) Ar degas 30 min b,d) vacuum degas 30 min * bubble peak
5% T running at 200 V/cm, sample: M13mp18, L= 41 cm

dissolving back into the solution. The equilibrium between the solution and the gas phase is the same as that shown in equation (2-2) where the gas phase on the right can be replaced with Ar. When the solution is bubbled with Ar, equilibrium is reached between the Ar bubbles and the solution. Fresh Ar continuously bubbled in forces the dissolved oxygen out of the solution.

With the same degassing time period (30 min), vacuum degassing is more efficient than Ar degassing as we can see from the different gel resolution shown in Figure 2.3. Compared with Ar degassing, vacuum degassing results in a relatively small amount of oxygen left in solution, which reduces free-radical chain termination, possibly resulting in formation of longer polymer fibers. Long polymer molecules appear to be important in efficient sieving of the DNA molecules.

2.3.2 Gel Age

As mentioned in Section 2.2.3, after the capillary is filled with gel, it is stored overnight and then used for DNA separation. This gel is called fresh gel. If the capillary is stored for one more day before use, the gel is called a 1-day old gel; and so on to count the gel age. The reproducibility of 5%T gel in terms of migration time of DNA fragments is compared for gels of different storage times in Figure 2.4. Data from 5 capillaries are plotted for fresh gels and data from 3 or 4 capillaries are plotted for other gels. The X-axis is the DNA fragment length (bases) and the Y-axis is the migration time. The data points plotted at 25 bases are for the primer peaks and the data points plotted at 900 bases are for the end peaks of the co-migrated long fragments. All runs are done at a 200 V/cm electric field.

It can be seen from Figure 2.4 that the aged gel is more reproducible than fresh gel in terms of migration time: the 3-day old gel is the most reproducible one compared with the gels that are less than 3 days old. Gels more than 100 days old were studied in this research group, and the same result was obtained.[16]

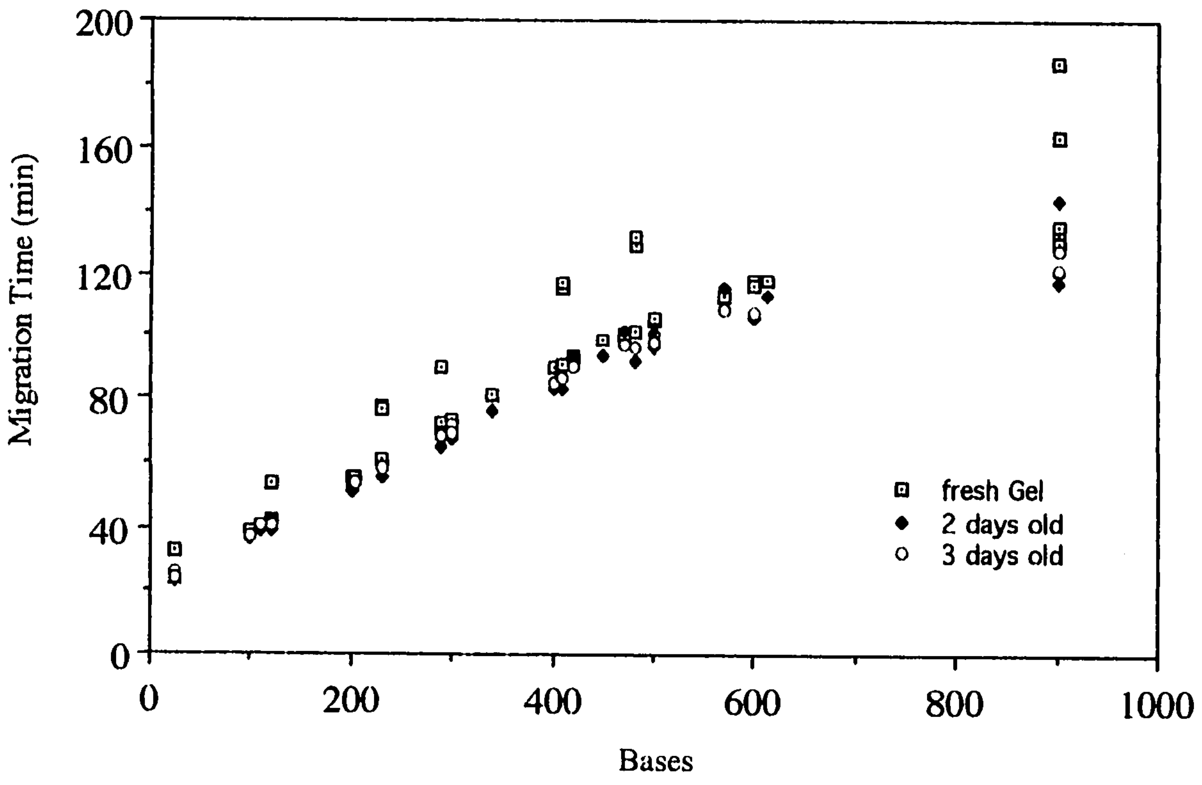


Figure 2.4 Effect of gel age on migration time reproducibility for 5% T

The migration time of DNA fragments with a certain length varies considerably in fresh gels because the ionic depletion at the injection end of the capillary is significant for fresh gels and the extent of ionic depletion from one capillary to the other may differ.

Ionic depletion during gel electrophoresis has been studied[17-21] and it occurs due to differences in the transport numbers of ions in free solution and in the gel.[20-21] The transport number is a measure of the fraction of the total electrical current carried by each ionic species.[22] The transport number for a given ion may change depending on its environment.[23] The DNA sequencing gel contains polyacrylamide, urea, and 1 X TBE buffer, whereas the running buffer and sheath-flow buffer contain only 1 X TBE. Current in the electrophoresis is mainly carried by the buffer ions which have different transport number in the free buffer solution and in the gel. The differences in transport number across the gel-buffer boundary lead to concentration differences as ions are pumped across the interface, causing depletion of ions at the buffer-gel interface at the injection end. The excess ions are pumped to the opposite end of the capillary, where they flush into the sheath-flow or the receiving reservoir; no change in ion concentration is observed at this end of the gel.[20, 21]

Anomalous mobility of DNA fragments near the negative electrode has been observed from a slab polyacrylamide gel; localized ionic depletion leads to a greater resistance thus a higher electric field near the negative electrode.[24] Ionic depletion is also demonstrated in capillary electrophoresis using noncross-linked polyacrylamide gels.[20] It was found that ionic depletion can lead to a significant voltage drop across the first few centimeters of the capillary, which leads to an electric field gradient within the capillary. The ionic depletion is significant for fresh gels, and the different extent of the ionic depletion in different capillaries contributes to the poor reproducibility of the migration time for DNA fragments of a certain length. This explains the variation of the migration time in fresh gels shown in Figure 2.4.

Figure 2.4 shows that the migration time of DNA fragments in the aged gel is more reproducible than that in the fresh gel. One possible reason is that the polymerization reaction of polyacrylamide is more complete in older gels than in younger gels. The polymerization kinetics of polyacrylamide has been studied by Caglio and Righetti, and it was reported that the polymerization is finished within 1 day.[25] So the change in gel reproducibility with age of the capillary is probably not due to the incomplete polymerization. It was reported that old gels generate a more stable current and a smaller ionic depletion.[20, 24] This might be the reason that the migration time of DNA is more reproducible in older gels than in younger gels as observed in Figure 2.4.

2.3.3 Running Temperature

Temperature is an important parameter in DNA sequencing. Joule heating introduces a temperature gradient which degrades resolution of the gel. Capillaries have efficient heat transfer due to their large surface-to-volume ratio so that Joule heating is generally not a problem in capillary gel electrophoresis. Operation of gels at elevated temperatures in a thermostatic environment is a different situation from Joule heating. High running temperatures actually have three advantages: increased separation speed, improved accuracy of base-determination and better resolution for longer fragments.

Operation of gels at elevated temperatures increases the separation speed. The viscosity of the medium decreases by about 2% /°C, which produces a proportional increase in mobility and decrease in separation time.[26, 27] The increased separation speed at elevated temperature allows a relatively lower electric field to be used to further increase the read-length.

Figure 2.5 presents the sequencing data obtained from an M13mp18 template separated with 5%T noncross-linked polyacrylamide at 150 V/cm and 60°C. The first readable sequence appears at 25 min, and 600 bases can be sequenced in 2 h at 60 °C rather than typically 3 h at room temperature.

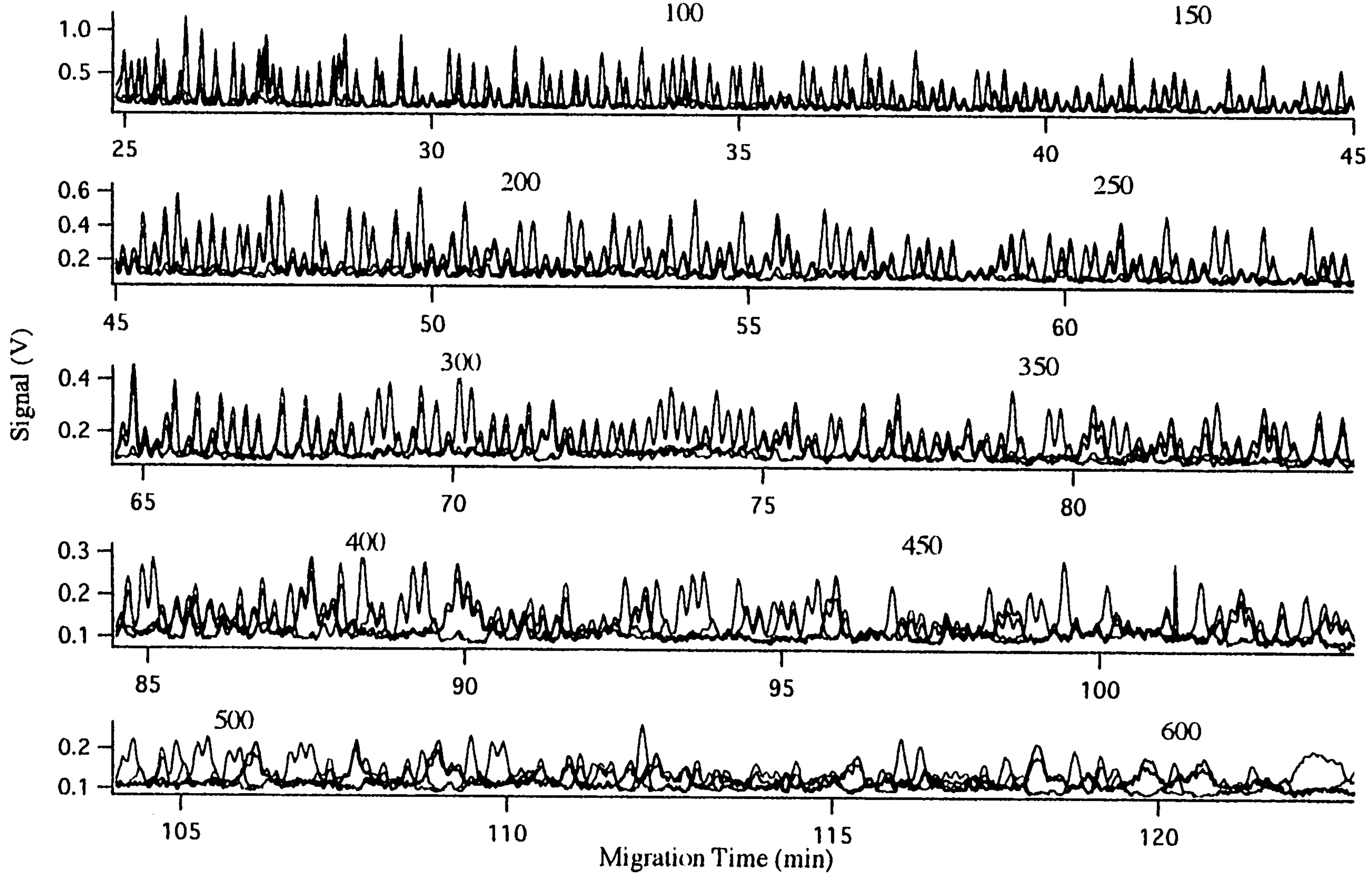


Figure 2.5 DNA sequencing at 60°C with 5% linear polyacrylamide 5%T running at 150 V/cm, L= 40 cm, sample: M13mp18

More importantly, separation at high temperature improves sequencing accuracy. Determinate errors occur in DNA sequencing due to the secondary structure of the single-stranded fragment, which can fold back on itself and self-hybridize to generate a hairpin structure. This structure migrates faster than relaxed DNA, resulting in overlapped electrophoretic peaks, which are called compression. Compressions are often observed in G C rich sequences because three hydrogen bonds can be formed between these two bases resulting in a relatively strong interaction. Denaturing reagent (7 M urea) in the gel helps to reduce formation of compressions, and formamide has been used as an additional denaturant in room temperature sequencing to further reduce but not eliminate the effects of compressions.[29-31] However, these compression can be relaxed fully through separation at elevated temperatures in the presence of 7 M urea.

Figure 2.6A presents a closeup of the region corresponding to bases 60 — 70 in M13mp18 at elevated temperatures. This region of the M13mp18 sequence is notorious for producing a compression in electrophoresis performed with conventional cross-linked polymers. The underlined sequence TACCGA for bases 63 — 68 are clearly separated at 60 °C and 70 °C, although base 63 T in the red channel overlaps with base 64 A in the green channel due to mobility shifts of small fragments. The mobility shifts are commonly observed in 5%T or more diluted gel and can only be avoided by using a higher %T gel (see next section). Operation of the capillary at 40°C leads to loss of resolution of peaks 66 and 67, which co-migrate. Also the two Cs of base 65 and 66 are barely resolved. Similar compression is observed in formamide-modified noncross-linked polyacrylamide.[30, 31] Finally, separation at 24°C results in a significant rearrangement of migration order, with base 67 G migrating faster than bases 64 A and 65 C. Bases 68 A and 64 A appear to co-migrate and bases 66 C and 65 C co-migrate as well. The compressions extend to bases 60 and 61, which are barely resolved.

Another compression region corresponding to bases 390 — 402 is shown in Figure 2.6B. At 60°C and 70°C, all peaks are evenly spaced and clearly resolved. At

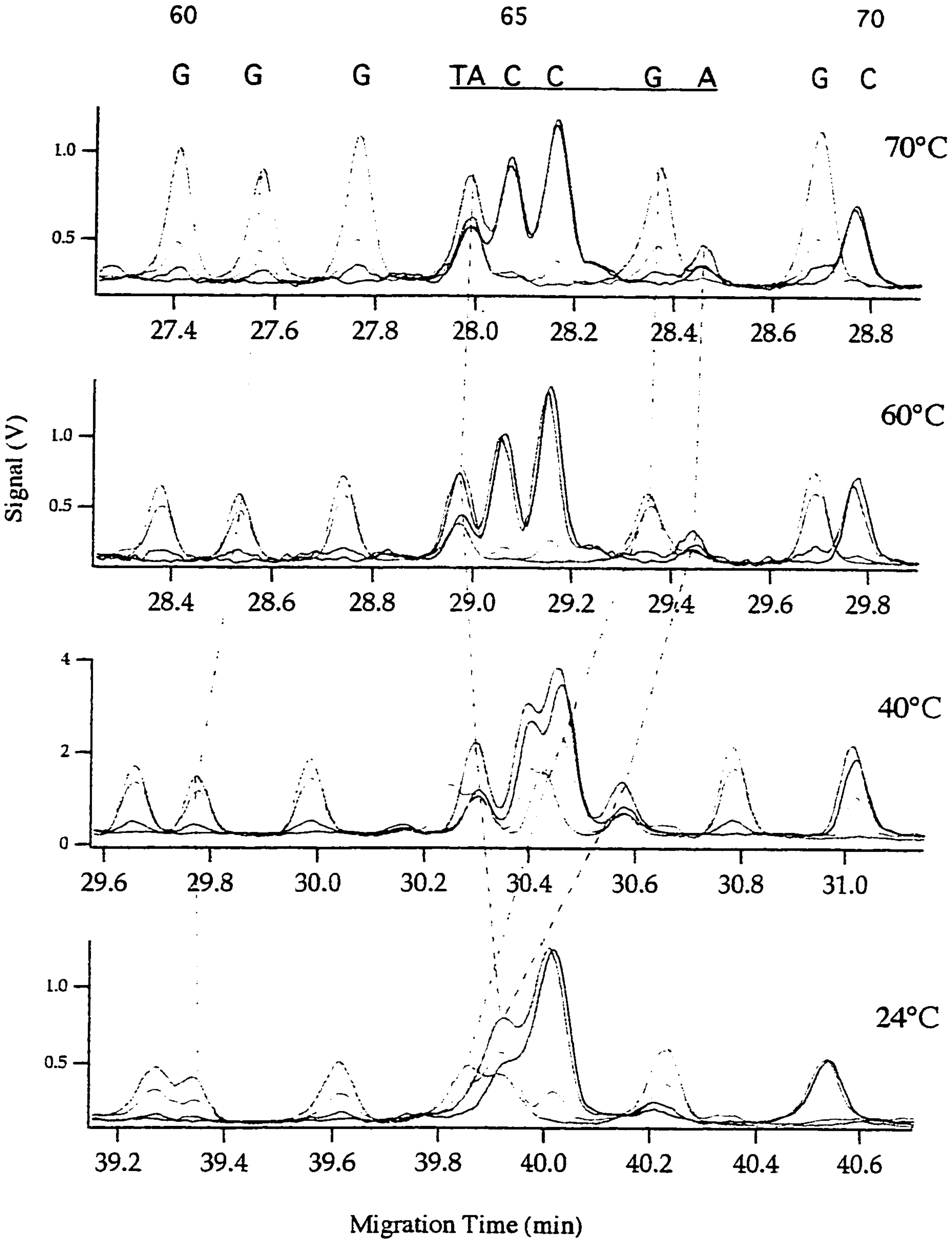


Figure 2.6A Effect of running temperature on compressions
5%T running at 150 V/cm

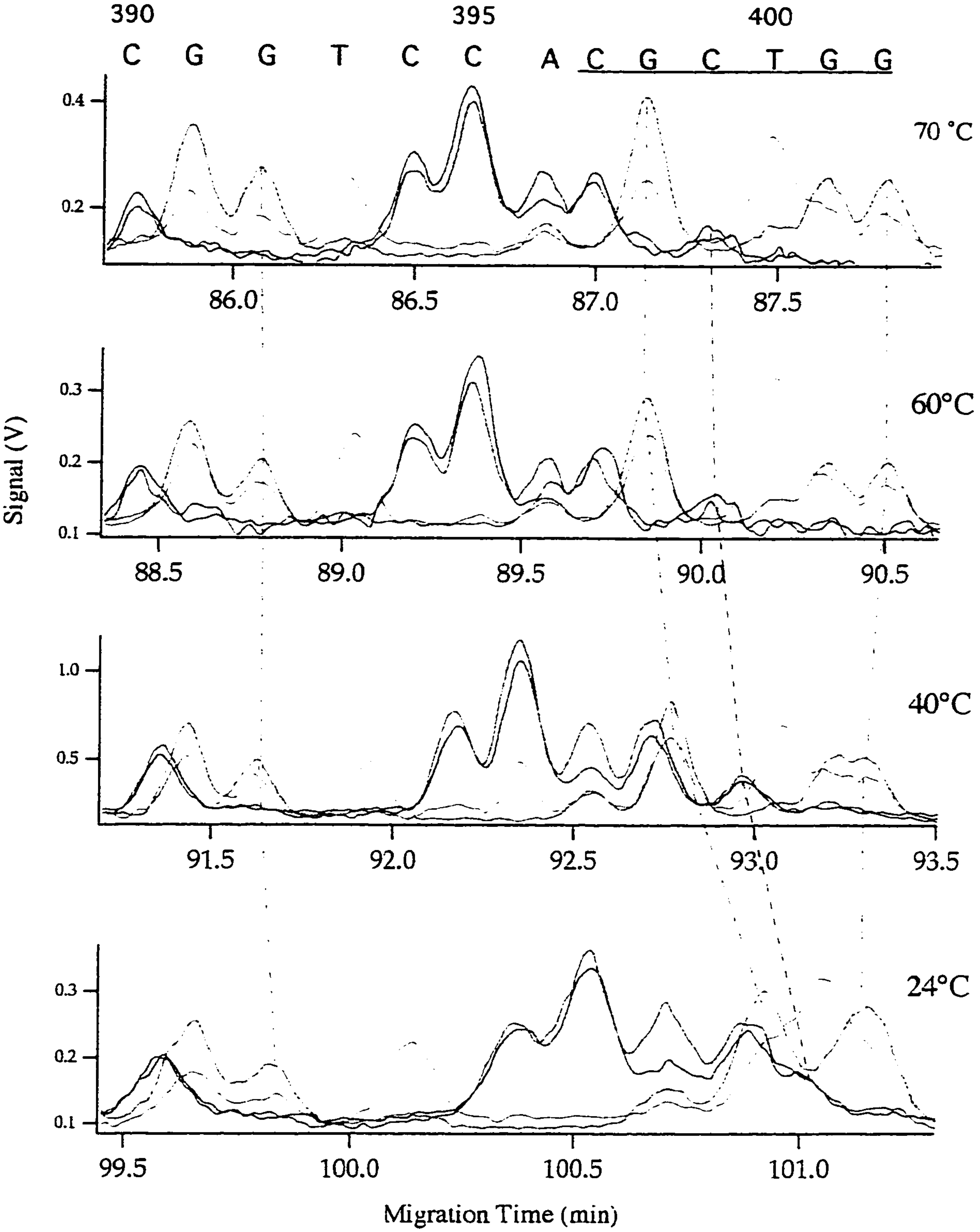


Figure 2.6B Effect of running temperature on compressions
5%T running at 150 V/cm

40°C, bases 397 C and 398 G are crowded together, and bases 401 G and 402 G are barely resolved. At 24°C, fragments of the underlined sequence migrate closely, two Gs of bases 401 and 402 co-migrate, and bases 399 C and 400 T co-migrate as well.

Compressions lead to systematic errors in sequencing, which apparently can only be avoided by operating the capillary at high temperatures. As we can see from Figure 2.6A and Figure 2.6B, operation of the capillary at 60°C or 70°C does not make much difference in resolution improvement; therefore, 60°C is high enough for sequence determination of M13 template.

Noncross-linked polyacrylamide at 60°C produces resolution of long sequencing fragments.[32] As we can see in Figure 2.5, the quintet and quartet of Ts at around 320 bases are baseline resolved. The doublet and triplet of Ts around 500 bases, which are usually not resolved at room temperature, are clearly resolved at 60 °C.

The activation energy of DNA separation in cross-linked[26,27] and non-cross-linked polyacrylamide[33] has been investigated in Dovichi's group. Unlike the behavior of the cross-linked material, the co-migration of longer fragments is less important at higher temperatures in noncross-linked polyacrylamide.[33] The longest resolvable fragment length extends to longer fragments with elevated temperatures. This enhanced separation can be explained in terms of an activated process, presumably associated with the passage of the single-stranded DNA fragments through pores in the gel. The noncross-linked polymer should be able to distort to allow passage of large DNA fragments. This distortion of the polymer may be driven by the electrostatic force of the DNA fragment. Larger, high-charged DNA fragments are driven more strongly by the electric field through the matrix, which results in distortion of the noncross-linked polymer. In contrast, a cross-linked gel will not be able to distort a significant amount without breaking the chemical bonds that cross-linked the gel fibers. DNA fragments must instead distort to pass through the pores, which requires more energy as the fragment length increases.

2.3.4 Gel Concentration

Several separation parameters affect the read-length and the separation time. The first effect is the electric field. There is a trade-off between separation time and read-length. High electric field results in a fast separation but short read-length, whereas low electric field results in good resolution for long fragments but takes a long time.[4,5] Typically a separation at 200 V/cm electric field in 5%T results in a good read-length (more than 500 bases) with reasonable separation time (2.5 h at room temperature).

The second effect is the running temperature. As discussed in the previous section, high temperature results in not only good resolution for long fragments, thus long read-length, but also fast separation. Therefore, relatively low electric field (150 V/cm for 5%T at 60°C) can be used to achieve long read-length and reasonable separation speed as shown in Figure 2.5.

The third effect is the concentration of the polymer. Separation in a high concentration of polymer takes a long time, compared with a low concentration of polymer at the same electric field. However, high %T noncross-linked polyacrylamide gives better resolution for short fragments, while low %T gives better resolution for long fragments. For an application involving short fragments and fast separation, a combination of high %T with high electric field and high running temperature is interesting. Here, separations in two different concentrations of noncross-linked polyacrylamide are compared using the 5-capillary sequencer.

The sample used is a four-color M13mp18 sequencing sample which is injected into capillaries at 100 V/cm for 35 s, and the running temperature is controlled thermostatically at 60°C. The separation in 5%T gel running at 150 V/cm is compared with that in 8%T gel running at 300 V/cm in Figure 2.7.

Two distinct advantages of 8%T gel over 5%T can be seen from Figure 2.7. One advantage is that more sequences of short fragments can be resolved by 8%T than 5%T. Fragments shorter than 39 bases are lost in a jumble of peaks after the primer peak in 5%T

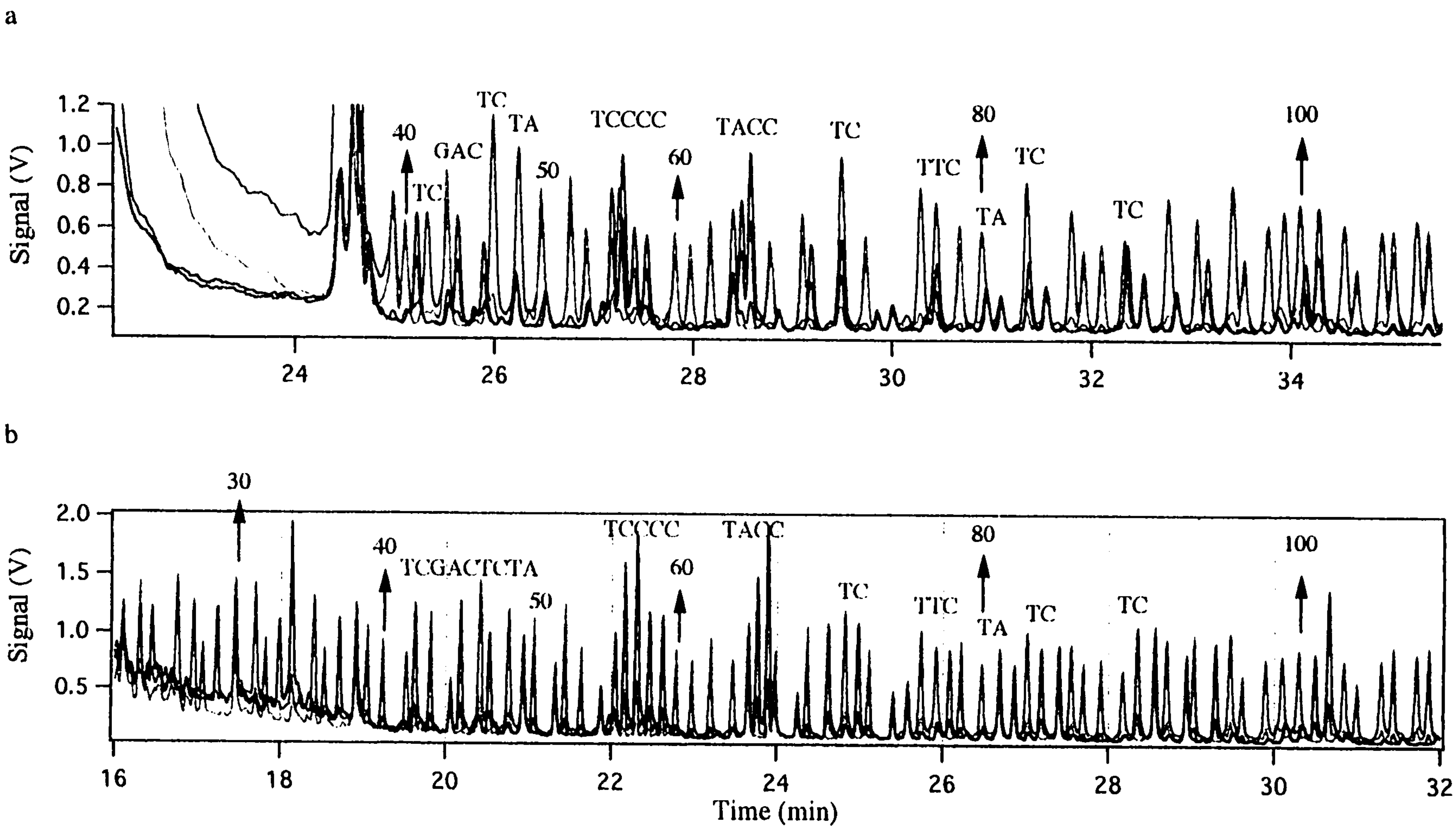


Figure 2.7 Effect of gel concentration on mobility shifts and resolution of short fragments
a) 5%T, running at 150 V/cm, 60°C b) 8%T, running at 300 V/cm, 60°C

gel (Figure 2.7 a), whereas fragments as short as 23 bases can be resolved by the 8%T gel (Figure 2.7 b).

Another advantage is that the mobility shifts of short fragments are avoided in the 8%T gel. As mentioned in the previous section, mobility shifts of short fragments are observed in 5%T gel for four-color sequencing with the ABI four-dye labeled primers, and the mobility shifts are most obvious for Rox-labeled T-terminated fragments. Separation of DNA fragments in 8%T gel avoids this problem. Figure 2.7 a shows that the peaks in the electropherogram are not evenly spaced for this early sequence from bases 40 to 110, and the T-terminated fragments migrate slower than their adjacent fragments at those locations labeled on the electropherogram. For sequence shorter than 60 bases long, the T-fragments are more than one base behind where they should be. For example, just after 40 bases, the sequence of "T C" is switched to "C T" in the electropherogram. Sequences of bases 46 and 47 "TC", 55 and 56 "TC" are switched as well. The mobility shifts of the T-fragments are corrected gradually when fragments get longer; at bases 80 and 81, the sequence appears correct but the two fragments migrate closer than they should be. The mobility shifts disappear after the sequence exceeds 100 bases.

The mobility shifts do not exist in the 8%T gel. As we can see from Figure 2.7 b, sequencing peaks in this section of electropherogram from bases 23 to 110 are evenly spaced. All the sequences appear correct in the labeled locations where mobility shifts are observed in 5%T (Figure 2.7 a).

The anomalous mobility shifts arise because the mobility of the dye-labeled primers depends on the particular dye used to label the primer. It appears that this mobility shift is not constant but instead it is most important for the smallest fragment, where the dye molecule makes the greatest proportional contribution to fragment mobility. Rox is relatively larger than the other dyes, so the mobility shifts are most obvious in Rox-labeled T-fragments. Polymers of higher concentrations have smaller pore sizes resulting in better resolution for small fragments so that the sequence can be resolved clearly in 8%T even with the abnormal mobility shifts of DNA fragments.

The 8%T gel running at 60°C and 300 V/cm electric field gives a very good resolution for short fragments as well as reasonable separation speed. Fragments of 100 bases migrate 30.3 min in a 40 cm capillary while the same fragments migrate 34.1 min in the 5%T gel running at 60°C and 150 V/cm (Figure 2.7).

However, the read-length of the 5%T at 60°C and 150 V/cm is longer than the 8%T at 60°C and 300 V/cm. The 5%T gel can resolve more than 600 bases as shown in Figure 2.5. But the 8%T gel at 60°C and 300 V/cm provides less resolvable length. As we can see from Figure 2.8, the 8%T gel provides baseline resolution up to 300 bases, and the resolution starts to degrade after 350 bases. Sequencing peaks are crowded together after 430 bases, and the end peak from the co-migrated long fragments appears at 90 min.

The relatively short read-length of the run in 8%T (shown in Figure 2.8) is a result of the combination of a high concentration of polymer and a high electric field. The fundamental restraint of electrophoresis — the co-migration of long fragments — limits the read-length in a high concentration polymer running at 300 V/cm.[7-10] As reported in the literature, less than 350 bases is determined even in 6%T at 300 V/cm.[34] In Figure 2.8, running the 8%T gel at 60°C helps to extend the read-length to 430 bases in less than 1.5 h.

The 5-capillary DNA sequencer has consistent performance for all five sample channels.[13] Here, four capillaries filled with 5% noncross-linked polyacrylamide are used to sequence a four-color M13 sample at 40°C and 150 V/cm. A section of the electropherogram from bases 238 to 346 are shown in Figure 2.9. The C between the quintet of Ts and quartet of Ts at about 75 min is 320 bases. Each capillary provides baseline resolution even at 320 bases.

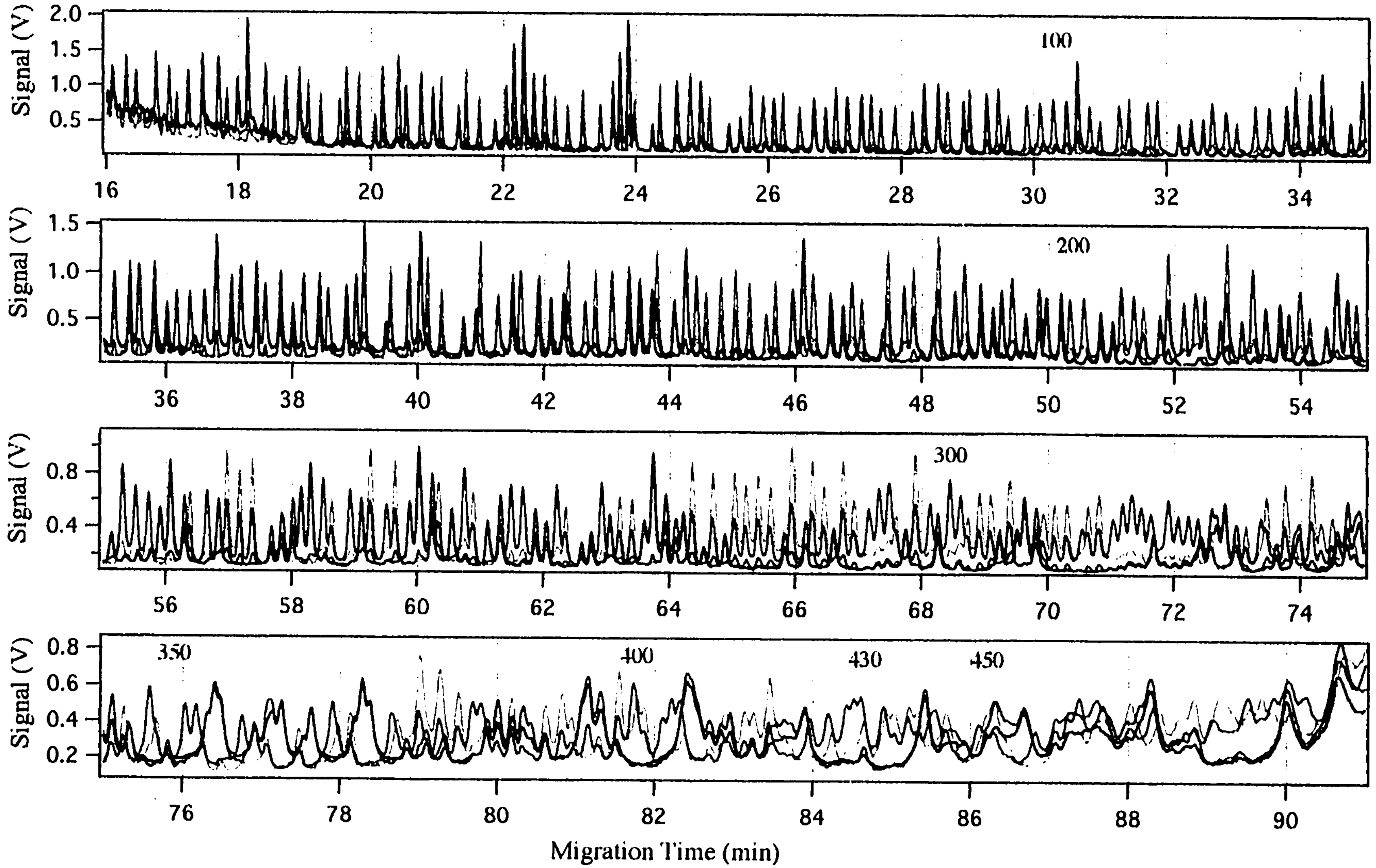


Figure 2.8 Electropherogram of four-color DNA sequencing in 8%T at 60°C sample: M13mp18, electric field: 300 V/cm, L = 40 cm

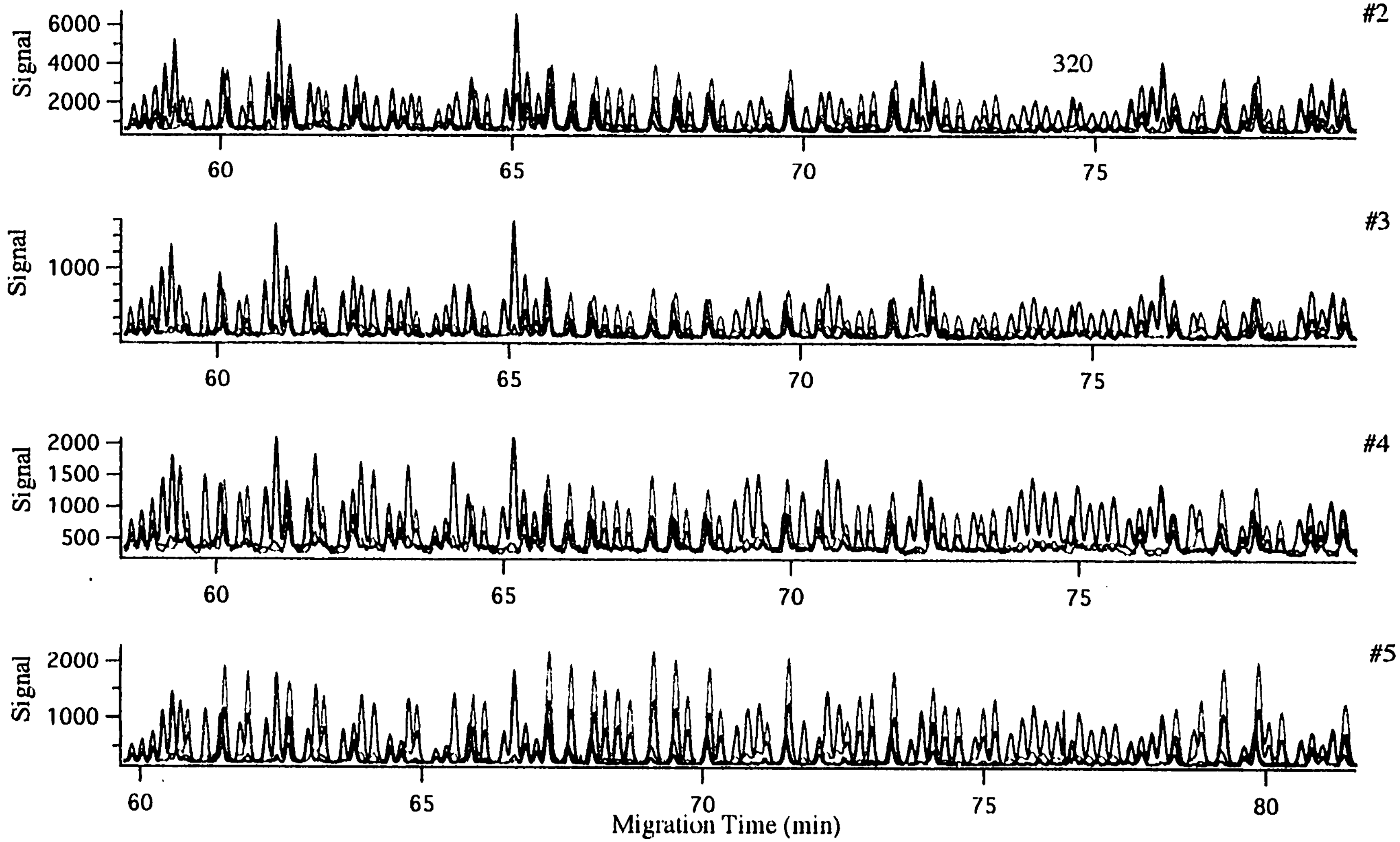


Figure 2.9 A section of electropherogram (238bp to 346bp) of 4 capillaries
5%T running at 150 V/cm, 40°C sample: M13mp18, L = 38.4 cm

2.4 CONCLUSIONS

Different separation conditions of DNA sequencing have been compared simultaneously in the 5-capillary DNA sequencer. In terms of gel preparation, the vacuum degassing works better than Ar degassing if the gel is polymerized in aqueous solution. Older gels are more reproducible in terms of migration time than younger gels; 3-day old gels give reasonable resolution and reproducibility. High temperatures result in fast separation, improved base-calling accuracy, and better resolution for longer fragments.

5%T noncross-linked polyacrylamide running at 150 V/cm and 60°C gives good read-length and reasonable separation speed; up to 600 bases can be resolved in 2 h. Separation in 8%T noncross-linked polyacrylamide at 300 V/cm and 60°C results in good resolution for short fragments: fragment as short as 23 bases can be resolved. This combination of high concentration of polymer (8%T), high electric field (300 V/cm), and high running temperature (60°C) results in a reasonable read-length and separation speed: 430 bases is resolved in less than 1.5 h. Mobility shifts of small fragments are observable in 5%T, but can be avoided by using 8%T gel.

The successful performance of the 5-capillary instrument proves that it is possible to build a sensitive multiple capillary instrument based on the sheath-flow system for post-column detection. In order to be comparable with commercial multi-lane slab gel systems, a capillary electrophoresis system with more channels is required. In response to the demand for a high sample throughput capillary electrophoresis system, a 32-capillary instrument with the ability to supply spectral information was designed and built. The design and performance of this 32-capillary instrument is discussed in the following chapters.

REFERENCES

1. B. Barrell, FASEB J. 1991, 5, 40-45

2. J. M. Claverie, Genomics 1994, 23, 575-581

3. J. Yan, N. Best, J. Z. Zhang, H. Ren, R. Jiang, J. Hou, and N. J. Dovichi. Electrophoresis 1996, 17, 1037-1045

4. M. J. Rocheleau, N. J. Dovichi, J. Microcolumn Sep. 1992, 4, 449-453

5. J. A. Luckey, L. A. Smith, Anal. Chem. 1993, 65, 2841-2850

6. T. Tanaka, Scientific American 1981, 244(1), 124-138

7 D. N. Heiger, A. S. Cohen, B. L. Karger, J. Chromatogr. 1990, 516, 33-48

8. S. Carson, A. S. Cohen, A. Belenkii, M. C. Ruiz-Martinez, J. Berka, B. L. Karger, Anal. Chem. 1993, 65, 3219-3226

9. S. L. Pentoney, K. D. Konrad, W. Kaye, Electrophoresis 1992, 13, 467-474

10. X. C. Huang, M. A. Quesada, R. A. Mathies Anal. Chem. 1992, 64, 2149-2154

11. E. N. Fung, E. S. Yeung, Anal. Chem. 1995, 67, 1913-1919

12. S. Menchen, B. Johnson, M. A. Winnik, B. Xu, Electrophoresis 1996, 17, 1451-1459

13. J. Z. Zhang, PhD thesis, Department of Chemistry, University of Alberta, DNA Sequencing by Single and Multiple CGEs, 1994, p156

14. N. J. Dovichi, J. Z. Zhang, US Patent 5,439,578, 8 August 1995

15. N. J. Dovichi, J. Z. Zhang, US Patent 5,584,982, 17 December 1996

16. D. Fiegeys, N. J. Dovichi, J. Chromatogr. A 1995, 717, 105-111

17. M. Spencer, Electrophoresis 1983, 4, 36-41

18. M. Spencer, Electrophoresis 1983, 4, 41-45

19. M. Spencer, J. M. Kirk, Electrophoresis 1983, 4, 46-52

20. D. Fiegeys, A. Renborg, N. J. Dovichi, Electrophoresis 1994, 15, 1512-1517

21. H. Swerdlow, K. E. Dew-Jager, K. Brady, R. Grey, N. J. Dovichi, R. Gestland, Electrophoresis 1992, 13, 475-483

22. J. Goodisman, Electrochemistry: Theoretical Foundations, Quantum and Statistical Mechanics, Thermodynamics, the Solid State, John Wiley & Sons, New York 1987, pp. 33-35

23. K. Izutsu, T. Nakamura, and T. Yamashita, J. Electroanalytical Chem. 1988, 256, 43-50

24. G. W. Slater, P. Mayer, G. Drouin, Analysis Magazine 1993, 21, M25-M28

25. S. Caglio, P. G. Righetti, Electrophoresis 1993, 14, 997-1003

26. H. Lu, E. Arriaga, D.Y. Chen, H. Starke, N. J. Dovichi, J. Chromatogr. 1994, 680, 497-501

27. H. Lu, E. Arriaga, D.Y. Chen, D. Figeys, N. J. Dovichi, J. Chromatogr. 1994, 680, 503-510

28. D. Y. Chen, H. P. Swerdlow, H. R. Harke, J. Z. Zhang, N. J. Dovichi, J. Chromatogr. 1991, 559, 237-246

29. M. J. Rocheleau, R. J. Grey, D. Y. Chen, H. R. Harke, N. J. Dovichi, Electrophoresis 1992, 13, 484-486

30. S. L. Pentoney, K. D. Konrad, W. Haye, Electrophoresis 1992, 13, 467-474

31. M. C. Ruiz-Martinez, J. Berka, A. Belenkii, F. Foret, A. W. Miller, B. L. Karger, Anal. Chem. 1993, 65, 2851-2858

32. J. Z. Zhang, Y. Fang, J. Y. Hou, H. J. Ren, R. Jiang, P. Roos, and N. J. Dovichi, Anal. Chem., 1995, 67, 4589-4593

33. Y. Fang, J. Z. Zhang, J. Y. Hou, H. Lu, and N. J. Dovichi, Electrophoresis 1996, 17, 1436-1442

34. M. Nelson, J. L. Van Etten, R. Grabherr, B.L. Karger, Nucleic Acid Res. 1992, 20, 1345-1348

Chapter 3. Optical Design of 32-Capillary Spectrometer

3.1 INTRODUCTION

The most popular DNA sequencing technique employs ABI four-dye chemistry. That is the so-called four-color DNA sequencing method, where each of the four termination fragments is labeled with a different dye and thus is represented by specific spectral emission wavelengths. Therefore, optical design to discriminate the four dyes becomes a major factor in the DNA sequencing instrument performance. Recently, multiple capillary four-color instruments have drawn more and more attention because of their potential for high sample throughput. Most of these systems have capillaries arranged in parallel in order to form a one-dimensional array. Our objective was to build a sensitive instrument with high sample throughput and spectral information.

A number of capillary array systems have been reported in the literature.[1-3, 6, 7, 10, 11, 15, 17] The systems fall into two classes according to the method used for acquiring signal from multiple capillaries: those that use optical scanning and those that use optical imaging.

3.1.1 Scanning Technique

There have been two scanning detector systems for capillary array electrophoresis. The first was reported in 1990, when Zagursky and McCormick[1] replaced a slab gel in a commercial sequencer with a linear array of capillaries. The instrument's detector scanned across the capillaries, recording the fluorescence signal sequentially from each capillary. A more recent design came from Mathies' group, using a confocal microscope in their scanner.[2,3] The confocal microscope reduces light scatter at the capillary wall, improving the signal-to-noise ratio.

These scanning systems are similar to those employed in commercial slab-gel sequencers. They all suffer from a fundamental limitation. Because the detector evaluates each capillary sequentially, the duty cycle is low. For example, in a 100 capillary instrument, each capillary is probed for 1% of the time. To first approximation, the fluorescence intensity from each capillary will be reduced by the duty cycle. In a 100 capillary instrument, the fluorescence signal will be reduced to 1% of the value generated in a single capillary instrument.

3.1.2 Imaging Technique

Scanning systems suffer from a poor duty cycle, particularly when dealing with large numbers of capillaries. The duty cycle problem is eliminated by using discrete detectors to monitor fluorescence from each capillary. Because each capillary is continuously monitored, there is no loss in duty cycle.

Before discussing the development of multiple capillary instruments, the development of a four-color detection system in a single capillary instrument should be examined.

Single-Capillary Four-Color System

The first high-speed, four-color capillary electrophoresis system for DNA sequencing was reported by Smith's group in 1990.[4] This system was developed for a single capillary with on-column detection. A multi-line argon (Ar) ion laser was used to illuminate a narrow capillary simultaneously with light at 488 and 514 nm. Fluorescence was collected at right angles from the capillary. A set of beam splitters was used to direct the fluorescence to a set of four photomultiplier tubes (PMTs), each equipped with a band-pass spectral filter, and fluorescence was recorded simultaneously in the four spectral channels.

The use of four narrow bandpass (DF10) filters allows discrimination of the four dyes by selectively detecting their maximum emission. It is well known that the 488 nm laser line produces a strong water Raman band around 580 nm, which interferes with the detection of Tamra labeled fragments. In Smith's system, both laser lines are present in the excitation of the four dyes, thus the water Raman band interferes with fluorescence detection.

In 1991, Dovichi's group developed a four-color system based on time-multiplexed fluorescence detection.[5] Instead of the use of beamsplitters and filters, a filter wheel with four 10 nm bandpass filters was rotated to sequentially transmit fluorescence intensity in four spectral bands to a single PMT. Two lasers were used to excite the four dyes, and the two were alternatively chopped by a chopper wheel. The filter wheel was synchronized with the chopper wheel to ensure that appropriate filters were in line with an appropriate laser. For example, detection at 540 nm or 560 nm was synchronized with the excitation at 488 nm.

In the filter wheel system, water Raman interference during the detection of Tamra-labeled fragments is eliminated, because the 488 nm blue laser is blocked when Tamra is being excited by the 543.5 nm green laser. This detector design offers the advantage of low component count and simple alignment as compared to the beam-splitter arrangement based on four PMTs.[4] The detector was first built for a single capillary instrument with on-column detection, and later developed into a sensitive 5-capillary instrument with sheath-flow for post-column detection.[6-7]

Also in 1991, another sequencer was reported for the ABI fluorescently labeled primer system. Gesteland's group used a spectrograph and charge-coupled device (CCD) camera to spectrally resolve fluorescence from all four dyes.[8] In this on-column detection system, fluorescence was monitored simultaneously from all four dyes, which improves the signal-to-noise ratio (S/N) compared with time-multiplex detection. With the use of a commercial spectrometer equipped with a 150 line grating, spectral information is available. The multi-wavelength detection provides better analytical selectivity than detection with four bandpass filters.

Four-color detection systems are continuously being developed and also applied to multiple capillary systems.

Multiple Capillary Array Four-Color System

In 1992, Smith's group developed a four-color detection system based on an assembly of a wedge prism. [9] This optical detection system is designed for ultrathin slab gel DNA sequencing with up to 18 channels and can be applied to multiple capillary systems as well.

The optics for four-color detection included a four-element interference filter and a four-component wedge prism sandwiched in-between two camera lenses. Each element of the filter assembly has a 10 nm bandpass filter which transmitted an optimized emission region of one dye. Following the filter assembly was a four-component wedge prism, in which each wedge was aligned with one of the four filter elements. Thus, each of the four selected wavelength regions was diverted angularly in a different direction. On the cooled CCD detector, four rows of fluorescent bands were obtained; each row corresponds to one spectral channel of all the separation channels. This design provides theoretically the same duty cycle as the filter wheel design,[5-7] but without any moving parts in the system. However water Raman interference will be present if two laser lines are used for excitation, because the laser is not chopped.

Unlike the post-column detection with sheath-flow cuvette,[6,7,11] it is difficult to illuminate all the channels simultaneously and uniformly with a point-focusing laser for on-column detection. In Smith's design,[9] a cylindrical lens was used to focus the laser into a line, then the beam was directed into the gel at Bruiser's angle (about 34° from horizontal) to minimize reflection loss. The line-focused laser beam illumination provides uniform and simultaneous illumination of all the channels, but this illumination method spreads the laser power over a large area. For example, if the laser power is spread over 100 capillaries, each capillary only benefits, on average, from less than 1% of the total laser power. If the excitation power is not high enough to photobleach a significant fraction of the dyes, and assuming shot noise limited detection, the S/N decreases roughly as the inverse square root of the laser power. Thus, illumination of a larger area of gel with the same laser power will decrease the S/N.

At the same time as Smith, Kambara developed a similar optical system based on filters and image-splitting prisms,[10] but used sheath-flow for post-column detection.[11] In their 20-capillary sheath-flow system, relatively large ID capillaries were used to load a sufficient amount of analyte for detection because of poor collecting efficiency. The wide-bore capillaries lead to large inter-capillary spacing, which decreased the detection efficiency further. As a result, both the optical and the heat-transfer properties of the system were not ideal.[12]

The 5-capillary instrument developed in Dovichi's group used a different design of sheath-flow for post-column detection.[6,7] Five capillaries with 50 μm ID and 140 μm OD were aligned in an array in a sheath-flow cuvette, laser beam was focused into a point and shone through the sample streams from the capillary array in the sheath-flow region. the schematic diagram of the 5-capillary instrument is shown in Figure 2.2. A filter wheel, which was synchronized with a chopper wheel, was employed for sequential transmission of the four dyes. A He-Ne green laser (543.5 nm) and an Ar ion blue laser (488 nm) were used for excitation. The two lasers were alternately chopped by the chopper wheel, recombined by a dichroic beamsplitter, and then focused onto the 5-capillary sheath-flow cuvette. Fluorescence emission from the sample streams of the 5 capillary tips was collected by a microscope objective, transmitted by a proper filter, then sent to the five avalanche photodiodes through a combination of GRIN lenses and optical fibers.

Focusing the laser beam onto the sheath-flow region instead of a window on the capillary produces much less light scatter and allows the simultaneous and uniform illumination of all the capillaries in an array, which in turn results in high sensitivity of the system. The detection limit (3σ) of this 5-capillary instrument was 130 ± 30 molecules of fluorescein injected into capillaries.[13a] As described in chapter 2, with this instrument, more than 600 bases can be sequenced at 60°C with 5% linear polyacrylamide in 2 h.[14]

Yeung's group reported a 100-capillary array system with on-column detection in 1994.[15] In order to eliminate the water Raman interference on the four-color detection, they used a two-window excitation method.[16] The 488 nm and 514 nm laser beams were line-focused and directed onto two separate windows on the 100-capillary array from both sides of detection optics. Fluorescence was collected at right angles to the capillaries and imaged onto a CCD camera. The use of the CCD camera ensured that all the capillaries were monitored simultaneously. However, the water Raman interference is no longer a problem because of the recent availability of ABI BIG DYES which allows single laser line excitation.

Quesada and Zhang reported the use of fiber optic illumination and detection.[17] An Ar ion laser beam was split into eight separate fibers to illuminate eight capillaries. Fluorescence was collected by a second bundle of fiber optics, dispersed with an imaging spectrograph, and detected with a CCD camera. This system took advantage of the redundant spectral information generated with the camera/spectrograph combination, and was reminiscent of the system reported by Gesteland for single capillary sequencing.

3.1.3 The 32-Capillary Spectrometer

Our 32-capillary spectrometer uses a more convenient cuvette design for post-column detection and a dispersing prism for spectral information. Figure 3.1 illustrates the preliminary design of the instrument.

The four-color detection system includes: a short wavelength cutoff filter (Scott Glass Technologies INC., Duryea, PA, USA) for rejecting scattered excitation light and passing emitted fluorescence light; two camera lenses (55 mm f/1.4, Nikon Canada): the first one for collecting and collimating fluorescence light and the second one for imaging the spectra onto detector; a dispersing prism (F2 equilateral, 75 x 75 x 75 x 50 mm,

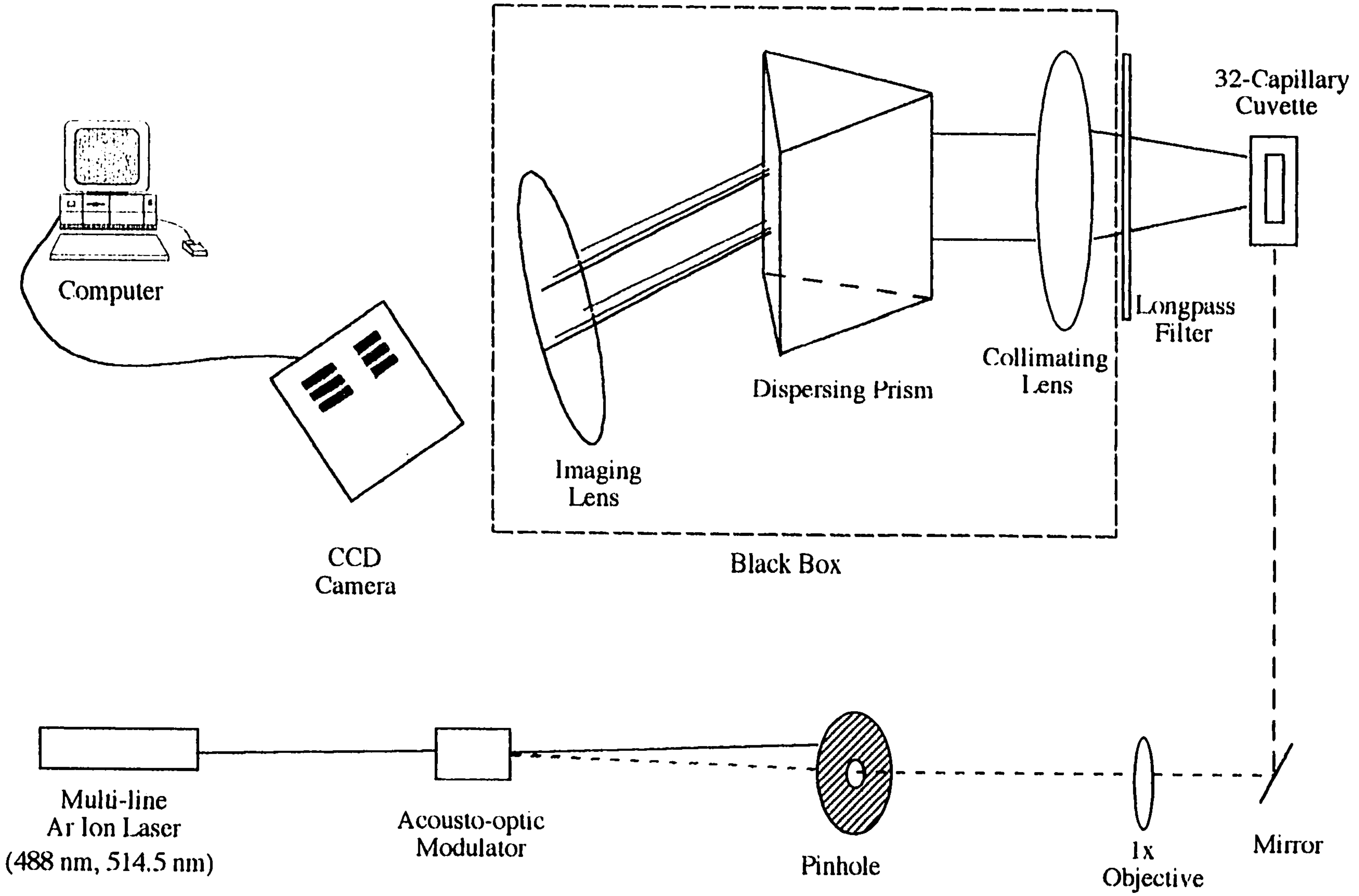


Figure 3.1 The preliminary design of 32-capillary spectrometer

OptoSigma, Santa Ana, CA, USA) for spectral dispersion; a cooled CCD camera (KAF1400, PXL, Photometrics Inc., Tucson, AZ, USA) for simultaneously monitoring the spectra from all 32 capillaries.

The 32 capillaries are held into a flat array in the locally-designed sheath-flow cuvette, which has some features to help align all the capillary tips to the same level. A multi-line Ar ion laser beam is focused by a 1x objective onto the cuvette, and illuminates the sample streams from all the capillaries simultaneously and uniformly. An acousto-optical modulator is used to block the laser beam instead of a camera shutter.

Elements of the instrument are discussed sequentially from the laser to the detector in more detail in the following sections.

3.2 CUVETTE DESIGN AND MANUFACTURE

3.2.1 Cuvette Design

Design Considerations

Sheath flow cuvettes are flow-through detection cells for post-column detection, as described in Chapter 1. For a linear array multiple capillary instrument, several considerations have to be taken into account to design the cuvette for an optimum laser-induced fluorescence (LIF) detection of a linear array of capillaries.

First of all, it is necessary to hold capillaries tightly in the direction perpendicular to the plane of the capillary array, so that a flat array is maintained in order for the laser to intersect the stream-flow from all 32 capillaries simultaneously.

Second, along the direction of the laser line, the distance between each capillary in the linear array has to be even and sufficient to ensure a uniform sheath flow between capillaries. A uniform sheath-flow is important for obtaining reproducible fluorescent spots after the capillaries are replaced from run to run. Otherwise, detection optics would have to be re-aligned each time capillaries are replaced. This is crucial for instruments with point detectors, such as the 5-capillary (Figure 2.2) and 16-capillary[18a] instruments. For the 32-capillary spectrometer, it is not as big a problem because software alignment is utilized instead of optics alignment. (Software alignment will be discussed in chapter 4.)

Finally, the optimum post-column position for excitation and detection of fluorescence is just beneath the capillary ends.[19] Therefore, all capillary ends must be vertically aligned so that the laser beam can be focused right below all the capillaries at once.

The first multiple capillary cuvette developed in Dovichi's group was a 5-capillary cuvette used in the 5-capillary instrument (Figure 2.2). This cuvette is a 4 mm x 4 mm x 20 mm quartz window with a 150 μm x 750 μm x 20 mm hole in the center. The opening hole serves as the capillary holder and sheath flow chamber.[13b] Typically, it uses 140 μm OD, 50 μm ID capillaries.

This design gives little control over the three points mentioned above, and those concerns become more important when the number of capillaries in the array increases to 16 or 32 in order to achieve high sample throughput. Dr. John Crabtree from Dovichi's group designed a 16-capillary cuvette through micromachining techniques.[18b] Features like channels as capillary guides and V-shaped capillary stops were added to the cuvette to obtain more control of the capillary array. The same idea and techniques were utilized to design and manufacture the 32-capillary cuvette for this instrument. Basically, two identical cuvette pieces with etched channels and capillary stops were glued together face-to-face, with a 145 ± 3 μm glass spacer between them on each side. Details will be discussed in the next section.

Figure 3.2 shows a 225 μm-spaced 32-capillary sheath-flow cuvette. The cuvette is made of Pyrex glass. 32 channels with center-to-center spacing of 225 μm are etched on the glass to serve as capillary guides; capillary guides are meant to aid capillary insertion and to space capillaries evenly. Typically, standard capillaries with 150 μm OD and 50 μm ID are used in this cuvette, the 225 μm spacing means capillaries are separated by a distance equal to half a capillary OD. With the center aligned to each channel, V-shaped

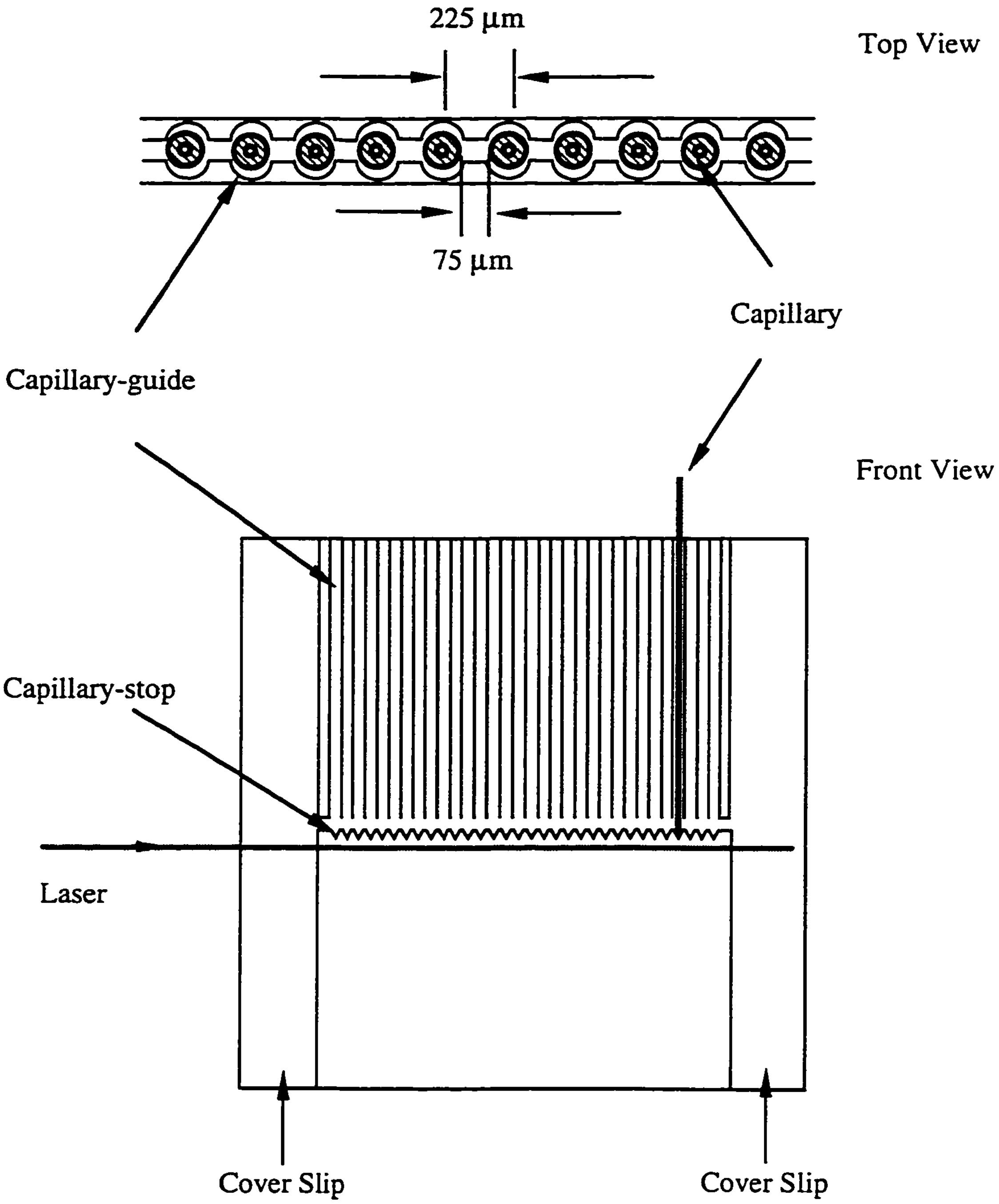


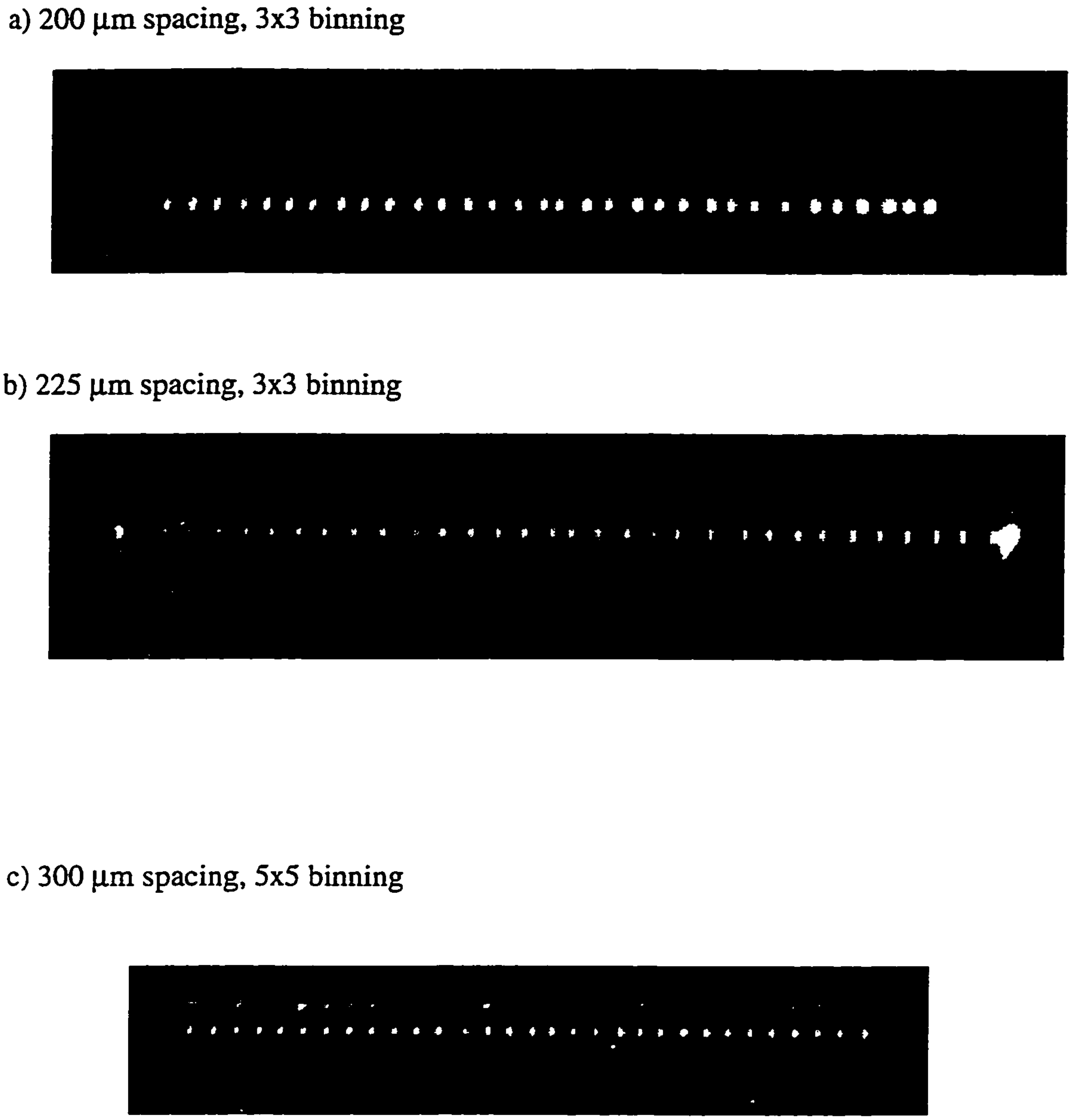
Figure 3.2 The 32-capillary sheath-flow cuvette

capillary stops are etched at a location 100 μm below the capillary guides. Capillary stops help to center the capillaries and hold each capillary end at the same height. The laser beam shines through the 32 capillary streams about 50 to 100 μm below the capillary stops.

Effect of Spacing Between Capillaries on Sheath-Flow

Three cuvettes with different spacing between capillary guides were made and tested by continuously flowing 10^{-7} M rhodamine 6G (R6G) through the capillaries. Figure 3.3 shows images of 32 fluorescent spots. The spacing between the capillary guides of these three cuvettes are 200 μm, 225 μm, and 300 μm in Figure 3.3 a, b, and c, respectively. The images are captured by a CCD camera with 3 x 3 binning for Figure 3.3 a and b, and 5 x 5 binning for Figure 3.3 c. (The binning parameter will be introduced in Section 3.3.5.) Space between adjacent capillaries is 50 μm (one-third of a capillary OD) in a 200-μm-spaced cuvette. The space increases to 75 μm (half of a capillary OD) in a 225-μm-spaced cuvette; and to 150 μm (one capillary dimension) in a 300-μm-spaced cuvette.

The best result is achieved with the 300-μm-spaced cuvette (Figure 3.3 c). All the fluorescent spots are evenly spaced, indicating an even sheath flow in the cuvette. For the 225-μm-spaced cuvette (Figure 3.3 b), the fluorescent spots are mostly spaced evenly except for a couple of spots in the center. In the 200-μm-spaced cuvette (Figure 3.3 a), the fluorescent spots are bigger and closer to each other on the right side than on the left side, indicating much slower sheath flow on the right side. One possible reason of this uneven flow is that the cover slip used on the right might be thinner than that on the left, but this shouldn't have a large effect since there were no problems in inserting capillaries into the channels on the right side. Most likely, it is because the spacing between capillaries is too small. Although capillary stops in the cuvette centers each capillary, the capillary is still capable of several μm's movement.[18c] If two capillaries each moves 5 μm away from



a) 200 µm spacing, 3x3 binning

b) 225 µm spacing, 3x3 binning

c) 300 µm spacing, 5x5 binning

Figure 3.3 Effect of spacing between capillaries on sheath-flow

each other in this 200-µm-spaced cuvette, then the sheath flow volume between the two capillaries will be 1.5 times greater than if the two capillaries move 5 mm towards each other. But this sheath flow difference reduces to 1.1 in a 300-µm-spaced cuvette.

The advantage of a larger spacing cuvette is that it results in a more even sheath flow. The drawback is that the width of the detection window increases. The detection window of a 300-µm-spaced cuvette is 9.6 mm, while the detection window of a 225-µm-spaced cuvette is 7.2 mm. The extended window results in difficulty in fluorescence detection and poor collection efficiency.[12] Also, the 300-µm-spaced cuvette is too big for the available CCD chip with 1 to 1 magnification, as shown in Figure 3.3 c, only 31 spots are detected by the CCD camera. For these reasons, the 225 µm-spaced cuvette is selected to serve as the detection chamber for this instrument.

3.2.2 Cuvette Manufacture

The cuvette is made using the facilities in the Alberta Microelectronic Center. A mask of the cuvette with detail design is made with a circuit drawing program called L-Edit (Tanner Research Inc., Pasadena, CA), and then sent to Precision Photomask in Québec for electronical mask fabrication. After the mask is ready, the cuvette is fabricated with micromachining techniques, which include deposition of a metal film, patterning of the mask by photolithography, wet etching of glass, gluing, cutting and polishing of the cuvette.[18]

Manufacture of Cuvette Pieces

A Pyrex glass plate is selected as the substrate. After it is cleaned, two metal layers (Cr layer followed by an Au layer) are deposited on the plate by vacuum evaporation. Then, a layer of photoresist is applied on top of the metal layers. After that, the mask is patterned onto the substrate by UV exposure. In the subsequent photoresist development step, the photoresist is washed away at those locations designed to be etched.

The patterned substrate is then treated with a mixture of 37% HCl and 70% HNO_3 (3 to 1 volumetric ratio) to etch the metal layers at those locations without the protection of photoresist. At this point, only those places designed not to be etched are covered by two metal layers with a photoresist layer above; the other areas are raw glass without any covering. Then, the substrate is treated with buffered oxide etchant (a mixture of 40% NH_4F and 49% HF with a volumetric ratio of 10:1) to etch the glass. The etching speed at room temperature is about 1 μm per hour. Typically, glass is etched by 25 μm for the cuvette pieces.

At the same time as the glass is etched downwards, it is also etched sideways underneath the covering layers of metal and photoresist. The ratio of horizontal to vertical etch depth is called aspect ratio, which is about 1:1 for an annealed Pyrex plate. The aspect ratio has to be considered in the design of the mask in order to obtain the right size in the final product. After the glass is etched to the desired depth, the substrate is washed with water to remove the wet etching solution. Finally, the remaining photoresist and metal layers are washed away with acetone and acid. At this stage, the etched cuvette pieces are ready for assembly.

Cuvette Assembly

The cuvette is assembled by gluing two identical etched cuvette pieces together with a thin glass spacer piece (cover slip) on each side. One edge of the cover slip is polished and this edge is placed inside the cuvette. The four pieces are aligned under a microscope and glued together with a UV curing glue. After being exposed to 366 nm UV radiation from an 18 W UV lamp (model UVGL-58, UVP Inc., San Gabriel, CA) for 30 minutes, the glued cuvette is heat cured at 50°C for 12 hours. After that, the cuvette is cut to the right size with a wafer dicing saw. The edge that the laser beam will be focused through is polished with diamond pastes.

In order to ensure that the cuvette holds the capillary array tightly to obtain a flat array, the average thickness of the cover slip used is 145 ± 3 μm for the use of 150 μm OD capillaries. Ideally, the laser travels through the cover slip, enters the solution in the detection region, exciting fluorescence from each capillary stream, and then exits through the other cover slip. The typical spot size of the laser beam is about 50 to 100 μm after being focused by a 1X microscope objective. In an ideal situation, the laser should never touch the two etched cuvette pieces within the detection region. However, the second cover slip reflects a portion of the laser beam back along a path different from the incident beam, and also unfocuses the laser beam. Due to this effect, the fluorescence of the cuvette glass can be detected. Fortunately, the glass fluorescence is weak and is in the far red spectral range. Therefore, it should not be a problem in the spectral detection process. This will be discussed further in Figure 3.20.

3.3 OPTICAL CONSIDERATIONS

3.3.1 Camera Shutter

The CCD camera comes with a mechanical shutter. The function of a shutter is to remain open when the camera is taking a picture, and to remain closed when the image data is transferred out to the computer. When the camera is in the data transfer (readout) mode, the mechanical shutter blocks the background light, so that no additional charge builds up on the CCD chip during the data transfer process. But during the experiment, the shutter often fails to close or open completely, especially when a picture is taken from the top of the optical table. This is possibly because the gravity adds friction to the shutter and keeps it from opening and closing, especially when the camera is tilted.

An alternative to a mechanical shutter is to use an acousto-optic modulator to block the laser beam. Acousto-optic (AO) components are commonly used in laser equipment for electronic control of the intensity and position of a laser beam. An acousto-optic device consists of two components: an AO driver and an AO modulator. Radio frequency (RF) output of the AO driver acts as the RF input of the AO modulator.

An acousto-optical modulator works on the photoelastic effect or piezooptic effect, which is the change in refractive index caused by pressure (mechanic stress or acoustic waves) in a crystal.[20] The RF generator drives a piezoelectric transducer to produce an acoustic wave, and the piezoelectric transducer is in contact with the acoustic-wave supporting crystal. When an acoustic wave is launched into the optical medium (crystal), it generates a refractive index wave that behaves like a sinusoidal grating. With an incident laser beam passing through, this grating will diffract the laser beam into several orders. When the incident laser beam is aligned to intercept the acoustic waves at the Bragg angle, a large fraction of the output beam is periodically diffracted into the first order at the acoustic frequency. The Bragg angle (θ_B) is given by:

$$\theta_B = \arcsin(\lambda/2\lambda_s) \qquad (3\text{-}1)$$

where λ is the wavelength of the laser beam, while λ_S is the acoustic wavelength in the crystal.

The efficiency with which the incident beam is diffracted into the first order is controlled by the RF power and the sixth power of the refractive index, so material with high refractive index is preferred for the crystal media. In other words, because acoustic waves introduce periodic changes in refractive index with periodic changes in optical pathlength, the crystal acts like a grating, diffracting light into different orders. It turns out that when the incident beam intercepts with acoustic waves at the Bragg angle, the first order beam has the highest efficiency. This angle is used to block the laser beam.

The acousto-optical modulator used in the spectrometer as a laser shutter is made of 80 MHz flint glass (model N24080, AO driver model N21080-1AME, Newport Electro-Optics Systems, Inc. Melbourne, FL, USA). The trigger signal from the CCD camera is used to drive the AO driver so that the AO modulator is synchronized with the CCD camera.

3.3.2 Laser

As shown in Figure 2.2, the 5-capillary instrument uses two lasers as excitation sources. One is a 543.5 nm green He-Ne laser, the other is a 488 nm blue Ar ion laser. The two laser lines are alternately chopped by a chopper wheel, which is synchronized with the filter wheel to align the proper laser in line with the proper filter. After the chopper wheel, the two laser lines are combined by a dichroic beamsplitter and directed to the cuvette.

The major advantage of using two alternately chopped lasers as excitation sources is that the Raman scattering of one laser line does not affect the detection of fluorescence excited by the other laser line, such as the interference of the water Raman band induced by the 488 nm laser on the detection of the Tamra fluorescence induced by the 543.5 nm laser. By blocking the blue laser when the green laser is being used for excitation, this interference caused by the water Raman band is eliminated. One disadvantage of the use of two lasers is that it is almost impossible to recombine the two laser lines completely. Extreme care has to be taken to adjust the angle of the dichroic beamsplitter, in order to recombine the two laser lines. Otherwise, each laser line excites fluorescence at a different location of cuvette, and this will decrease the sensitivity of the instrument due to misalignment of the sample stream, laser lines and point detectors.

Unlike the 5-capillary instrument, the 32-capillary spectrometer only uses one laser (model 532A, Omnichrome, Melles Griot, Carlsbad, CA, USA) as the excitation source. It is a multi-line Ar ion laser, which produces 6 oscillation laser lines (five blues and one green); two major lines are the 488 nm blue line and the 514.5 nm green line. Advantage of the use of one multi-line laser instead of two single-line lasers is that the two laser lines are always combined, which is almost impossible to achieve in the 5-capillary instrument. Also, it is more economical to buy one multi-line laser than two single-line lasers.

The output laser power is adjustable from 15 to 50 mW. An experiment has been done to measure the power of both laser lines. As shown in Figure 3.4 a, a 515DF20 filter is placed between the laser and the acousto-optic modulator to filter out the blue lines. As discussed in Section 3.3.3, the wavelength band transmitted by an interference filter is sensitive to the incident angle of the incoming laser beam. When the incident laser beam is normal to the plane of the filter surface, all of the green laser line will go through the filter and be directed onto the AO modulator, then onto the cuvette; all of the blue line is reflected back to the laser cavity, resulting in an unstable oscillation condition in the laser cavity. In order to keep the reflected blue line from going back to the cavity, and to measure the power of the reflected blue line as well, the filter has to be tilted to direct the reflected blue line away from the incident laser beam, and also to leave enough space between incident and reflected beam to fit the power meter sensor. In this case, a small portion of the green line is also reflected.

As shown in Figure 3.4 a, an optical power meter (power meter: model 835, Sensor: model 818-UV, Newport Corporation, Fountain Valley, CA, USA) is placed at position 1 to measure the power of the blue laser line with a wavelength setting at 490 nm. It is then placed at position 2 to measure the power of the green laser line with a wavelength setting at 520 nm. Finally, it is placed at position 3 to measure the green line power after it goes through the modulator.

The results from the power meter are plotted against the power setting of the laser in Figure 3.4 b. The green laser line is weaker than the blue line. At the laser power setting of 20 mW, the measured power of the blue line is 20 mW, while that of the green line is only 3.8 mW, which is less than one-fifth of the power of the blue line. When the laser power is set at 40 mW, the power of the blue line reads as 39.4 mW, while that of green line reads as 9.3 mW, which is about one-fourth of that of the blue line's power. Actually, the laser is at the optimized oscillation condition of the blue laser lines. However, although the power of the green line is weaker than that of blue line, the excitation efficiency of

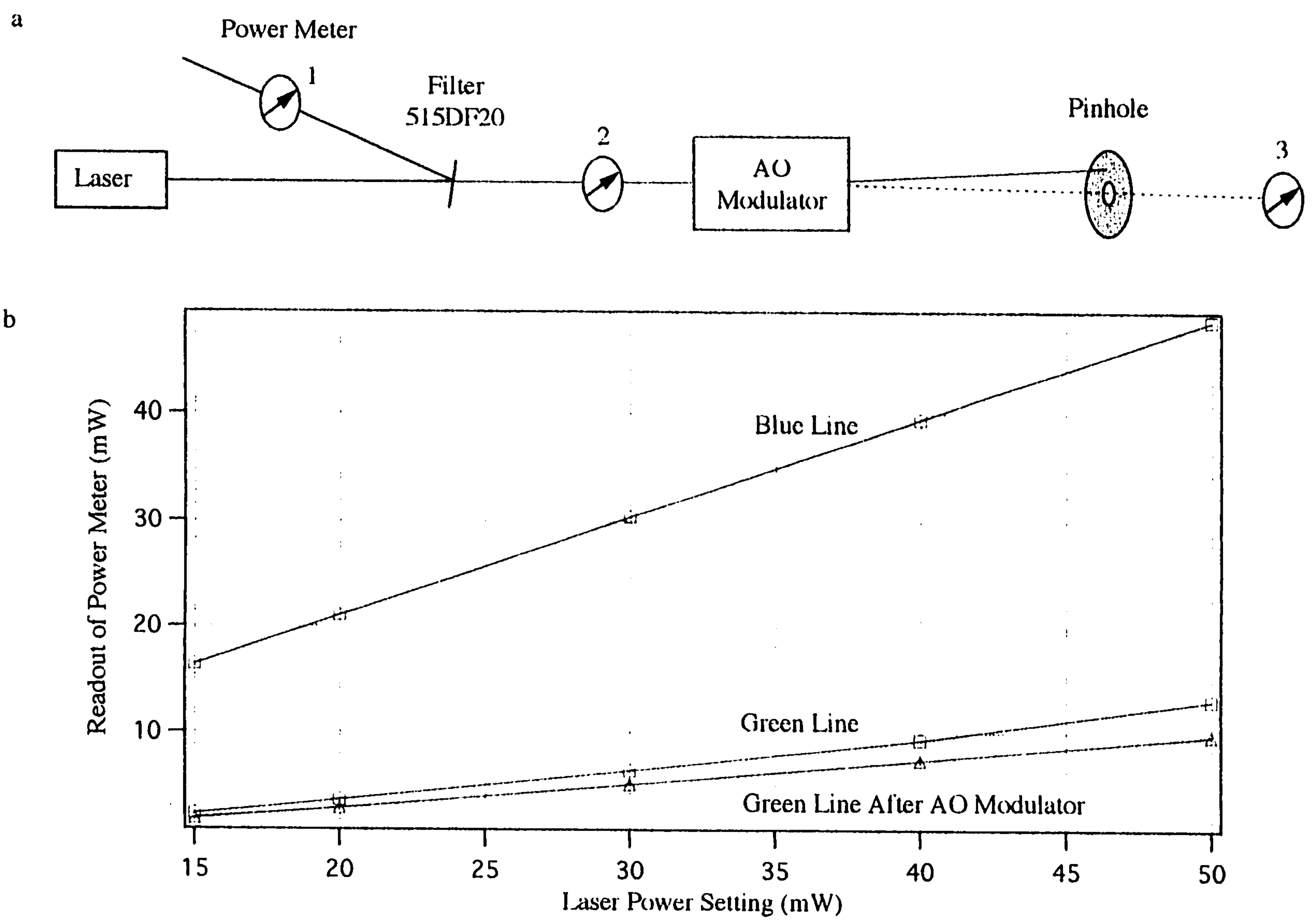


Figure 3.4 Power of the green laser line and the blue laser line

the green line is higher than that of blue line, so that the weaker green line does not cause a large decrease in sensitivity during four-color DNA sequencing with the ABI four-dye labeled primers.

When the green laser beam travels through the AO modulator, the modulator works like a grating and diffracts light into various orders. The first order is used for excitation. By carefully adjusting the incident angle of light to the crystal, the power of the light in the first order can be maximized and allowed to go through the pinhole. Calculated from the data shown in Figure 3.4, the loss of the green laser power caused by the modulator is about 20%.

3.3.3 Filters

There are two types of optical filters: cutoff filter and bandpass filter.

A cutoff filter is usually an absorption filter. There are two types of cutoff filters: longpass (LP) and shortpass (KP). A longpass filter allows a high transmission in the longwave region and a low transmission in the shortwave region. A shortpass filter allows low transmission in the longwave region and a high transmission in the shortwave region. A cutoff filter is classified by its cutoff position (λ_c), which is the wavelength at which the internal spectral transmittance is 50%. For example, a LP530 filter is a longpass filter, and it cuts off at $\lambda_c = 530$ nm.

The detection filter of the 32-capillary spectrometer is a colored glass cutoff filter (Schott Glass Technologies Inc., Duryea, PA, USA). Unlike an interference bandpass filter, a colored glass filter is not sensitive to the incident angle of the fluorescence light and does not have to be placed in parallel light. So the sharp cutoff filter rather than a bandpass filter is used in front of the light collection lens in order to reject laser light while passing fluorescence.

A bandpass (BP) filter is typically specified by the center wavelength at maximum transmittance (λ_m) when incident light is normal to the plane of the filter, and the bandwidth passed through the filter ($\Delta\lambda$) which is the full width at half maximum (FWHM) at the curve of transmittance verse wavelength. For instance, a 580DF30 filter passes light with a center wavelength of λ_m = 580 nm and a bandpass width of $\Delta\lambda$ = 30 nm. Therefore, only light with wavelengths in the 580 nm ± 15 nm range passes through the filter.

All the bandpass filters used in the instrument are interference filters. For an interference filter, the center wavelength λ_m is sensitive to the incident angle of the incoming light. If the incident light is not normal to the filter surface, the transmitted center wavelength will be different from the specified center wavelength. For a Fabry-Perot type of interference filter, the new transmitted center wavelength is related to the center wavelength at normal incidence (λ_m) and the incident angle (θ) by:

$$\lambda = \lambda_m\,(1 - (\sin^2\theta\,/\,n^2)\,)^{1/2} \qquad (3\text{-}2)$$

where n is the refractive index of the dielectric material in the filter. To use an interference filter to pass the designated wavelengths, it is necessary to place the filter in parallel light with the incident angle of incoming light normal to the filter surface.

As mentioned in Section 3.3.2, among the six oscillating wavelengths (five blue and one green) in the multi-line Ar ion laser, the two lines with the strongest intensity are the 488 nm blue line and the 514.5 nm green line. In those experiments involving single line excitation, a narrow bandpass filter is used to allow a single laser line to pass through. with the other laser lines reflected back. The most often used laser filters are a 488NB3 filter which passes 488 nm ± 1.5 nm blue line, and a 514.5NB3 filter which passes 514.5 nm ± 1.5 nm green line.

3.3.4 Lenses

Lens Specification

There are a couple of important parameters for a lens: f-number, numerical aperture and field of view. The f-number is defined by:

$$\text{f-number} = f / \phi \qquad (3\text{–}3)$$

where f is the focal length, and ϕ is the clear aperture diameter of the lens, as shown in Figure 3.5 a. For instance, the clear aperture diameter of a 55 mm f/1.4 lens is 55/1.4 = 39.3 mm.

Numerical aperture is associated with the amount of light collected from a point source by the lens:

$$\text{N.A.} = n \sin\theta = \phi / 2f \qquad (3\text{-}4)$$

where n is the refractive index of the medium in the object or image space (usually 1 for air). θ is the largest collectable angle (Figure 3.5 a) Combining the above two equations, N.A. is related to f-number by:

$$\text{f-number} = 1/ (2\ \text{N.A.}) \qquad (3\text{-}5)$$

both N.A. and f-number represent the collection efficiency of a lens, where a larger N.A. or smaller f-number indicates better collection efficiency.

Field-of-view describes the actual view window of the lens and relates to the angle at which the lens is able to see the object. One point worthy of mentioning is that the N.A. of an optical fiber describes a different angle than it does in a lens, as shown in Figure 3.5 b. The N.A. of an optical fiber represents the cone of light accepted by the fiber. It actually describes the field-of-view of the fiber.

In general, it is easy to efficiently detect light from a compact object, but it is difficult to detect light from an extended object. For example, high magnification microscope objectives have high numerical apertures and efficiently detect fluorescence from a localized region. In contrast, low magnification objectives have poor collection efficiency but have a wide field-of-view. As it has been described in section 3.2.1, the total width of the 32 capillary array in the designed cuvette is 7.2 mm. In this case, a microscope objective is not efficient enough to collect the fluorescence from the extended capillary array; therefore, a relatively large-diameter lens is needed for good light-collection efficiency.

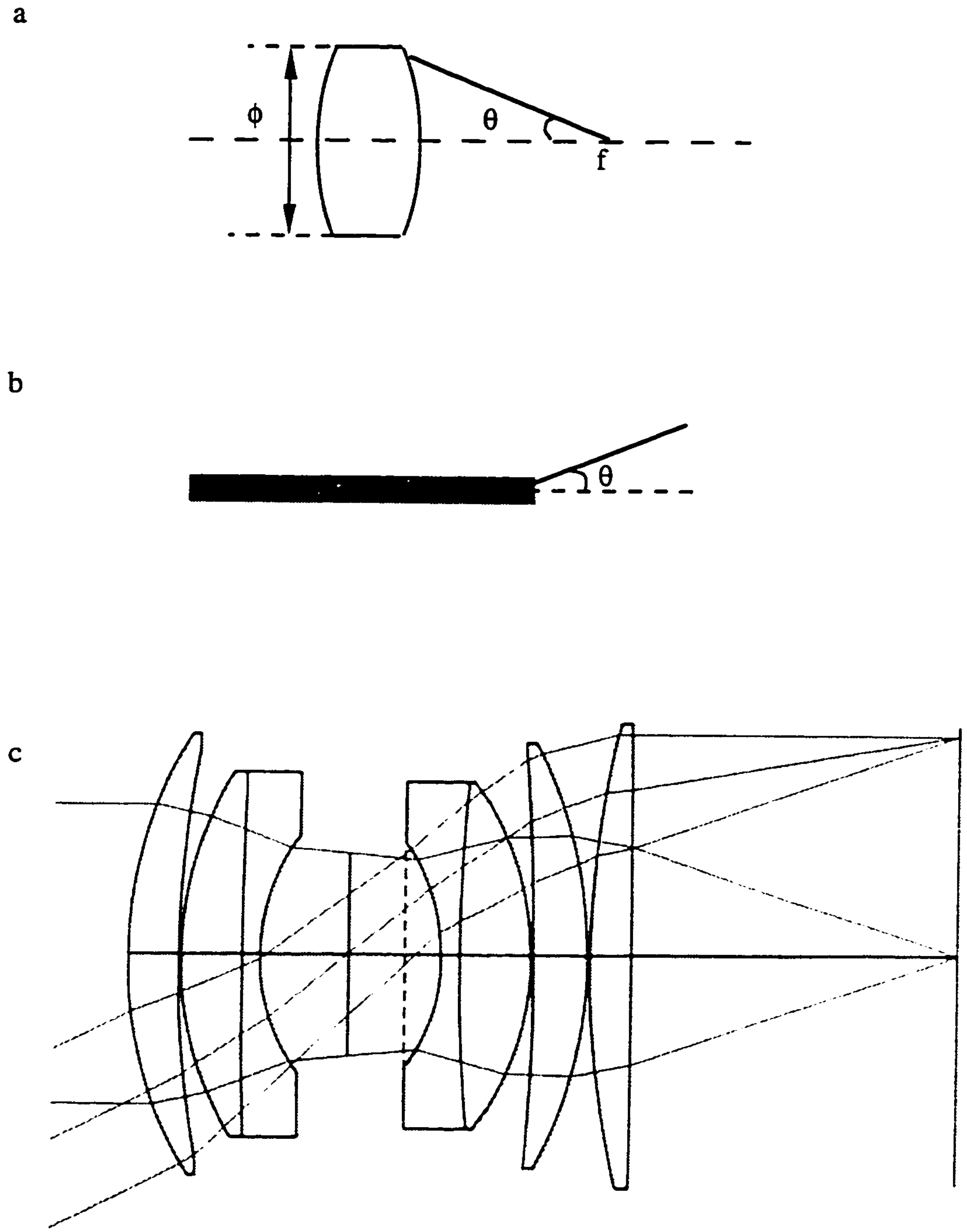


Figure 3.5 a) conventional lens
b) optical fiber
c) 50 mm f/1.8 SLR camera lens

Camera Lenses

A camera lens is typically a double gauss lens. Figure 3.5 c illustrates a 50 mm f/1.8 SLR camera lens,[21] which is the basic lens supplied with a 35 mm single-lens reflex (SLR) camera. The lens consists of a nearly symmetrical arrangement of elements about a central stop. Surrounding this stop are two "achromats" with the flint element facing the stop, such that its surface is concave toward the stop. Additional elements are generally added in the front and rear to reduce the f-number and correspondingly will improve the aperture. The complex arrangement of elements in a camera lens provides clear imaging, less spherical aberration, and relatively achromatic characteristics of the lens. A camera lens also offers a larger observation field compared to a microscope objective, and can image a large number of capillaries. In these experiments, 35 mm camera lenses (focal length 55 mm, aperture f/1.4, Nikon, Japan) were used to achieve good collection efficiency and less aberration.

In order to apply a dispersion device to disperse the spectra with high spectral resolution, the fluorescence light from the 32-capillary array needs to be collimated. A pair of lenses is arranged to achieve this purpose (Figure 3.1). The first lens (a collimating lens) collects and collimates the fluorescence light from 32 capillaries; the second lens (an imaging lens) re-focuses the collimated light and images the spectra onto the detector. A dispersing prism is placed in-between the two lenses, and disperses the spectra from all 32 capillaries.

As a general rule in imaging optical systems, the amount of light collected varies inversely as the square of demagnification. Assuming that a system is shot noise limited, this rule leads to a linear decrease in signal-to-noise ratio with demagnification.[9] For this reason, two identical camera lenses (55mm, f/1.4) are chosen to produce a system magnification of 1 in an ideal situation; the imaged area is 7.2 mm in width, giving the same width of spectra array on the CCD chip. The first camera lens is positioned with its focal point coincident with the excitation laser line in the sheath flow cuvette, thereby collecting and collimating the emitted fluorescence. The collimated fluorescence passes through a dispersing prism, which disperses the fluorescent spectra along a direction perpendicular to the fluorescent spot array. The second camera lens collects this light and produces images of the spectra on the surface of the CCD.

3.3.5 Detector

A two-dimensional CCD camera is an efficient detector for simultaneously monitoring the spectra of all 32 capillaries. When a CCD chip is exposed, light strikes the silicon chip, and a photon-induced charge is accumulated in each potential well or pixel. Within the linear response range of the CCD chip, the total amount of charge accumulated in each potential well is proportional to the product of light intensity and exposure time. A pixel can be saturated if it is exposed to a large amount of photons which exceeds its linear range, then the electrons may overflow along the charge transfer direction into adjacent pixels, an effect called smearing. After exposure, the stored electrons in each potential well are read out through a charge transfer process. When the camera finishes the readout of the whole image, it is ready for another exposure.

An important parameter of the CCD camera is the hardware binning parameter. Binning takes place at the readout. For example, in a 2x4 binning, charges of 2 pixels in serial and 4 pixels in parallel are added together at the output node. In other words, data of every 2 columns and 4 rows in the CCD chip are added together as one output data point. Binning results in a decrease of readout noise and an increase of frame readout speed, but a loss of image resolution. Reasonable binning is often utilized to reduce frame data size, with respect to the required data transfer speed and image resolution.

The 32-capillary spectrometer utilizes a PXL CCD camera (KAF1400-G1, Photometrics Ltd., Tucson, AZ, USA) as a detector. The PXL camera has good pixel resolution where the physical size of one pixel is only 7 μm square; the small pixel size provides good spectral resolution for the spectrometer. There are 1316 pixels in serial (column) and 1034 pixels in parallel (row). A built-in electronics system within the camera makes digital output available to be directly sent to computer through a SCSI cable. This feature of the camera makes data acquisition free of extra electronics amplification and transaction, but subject to software design.

3.4 DISPERSION DEVICE

Either a grating or a dispersing prism can serve as a dispersing device for this spectrometer. The drawback of a grating is the presence of higher order spectral bands, but it would not produce spatial interference for this spectrometer because the separation system is a one-dimensional capillary array. However, a dispersion prism is preferred in order to directly apply the designed optical system to our next generation instrument, which has a 2-dimensional separation system.[22] In this 2-D instrument, free space for spectral dispersion is only 800 μm, and if a grating is utilized as the dispersion device, the higher order spectral band from capillaries in the first row will overlap with that from subsequent rows. Evaluation of a dispersing prism is discussed below.

3.4.1 Concepts Related to a Dispersing Prism

Deviation

When light travels from one medium to another, Snell's law applies. In a typical dispersing prism, as shown in Fig. 3.7, a light ray strikes the first surface at an incident angle ϕ_1, and is refracted downward, making a refraction angle φ_1; then, it strikes the second surface at angle ϕ_2, and is refracted downward again at angle φ_2. The total deviation (δ) of the ray is given by:

$$\delta = (\phi_1 - \varphi_1) + (\varphi_2 - \phi_2) \tag{3-6}$$

if *A* is the apex angle of the prism, from the geometry of Figure 3.6, it can be seen that:

$$D = 180^\circ - A \tag{3-7}$$

$$\varphi_1 = 180^\circ - D - \phi_2 = A - \phi_2 \tag{3-8}$$

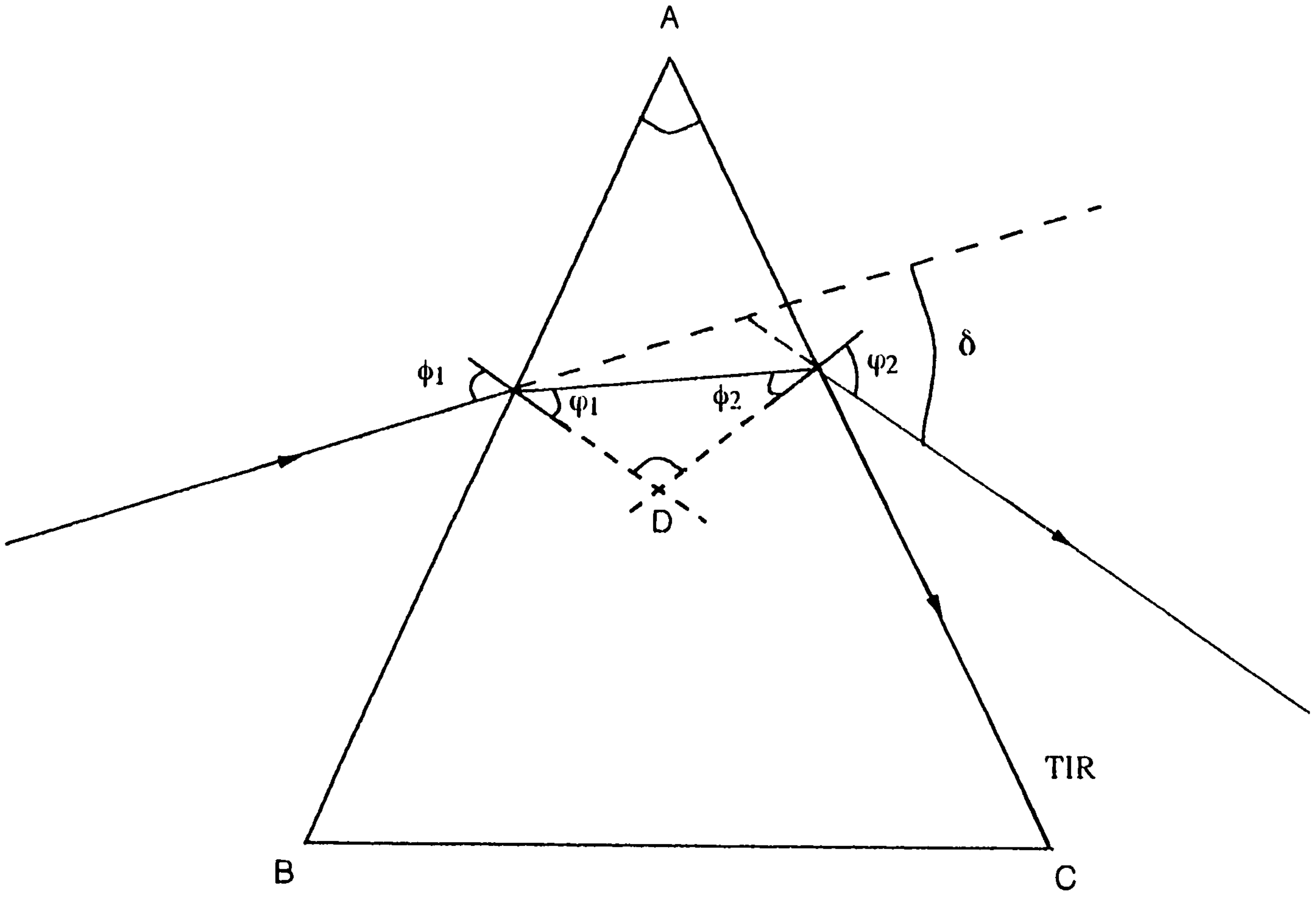


Figure 3.6 Dispersion caused by a prism
TIR: total internal reflection δ: deviation

according to Snell's law:

$$1 \sin \phi_1 = n \sin \varphi_1 \qquad (3\text{-}9)$$

$$n \sin \phi_2 = 1 \sin \varphi_2 \qquad (3\text{-}10)$$

Combining equations (3-6), (3-8), (3-9), (3-10), the total angular deviation can be expressed as:

$$\delta = \phi_1 - A + \arcsin [(n^2 - \sin^2\phi_1)^{1/2}\sin A - \cos A \sin\phi_1] \qquad (3\text{-}11)$$

where the deviation is a function of the refractive index (n) of the prism material, and hence of wavelength (λ). According to equation (3-11), at the same incident angle ϕ_I, an increase of n will result in an increase of deviation δ. For most materials, the refractive index is higher for short wavelength (blue light) than for longer wavelength (red light), therefore the deviation angle will be greater for blue light than for red light. This is how a prism works as a dispersion device.

Total Internal Reflection (TIR)

A critical condition arises, as shown in Figure 3.6, when angle $\varphi_2 = 90°$ and the incident light undergoes total internal reflection (TIR). Substituting this critical condition into equation (3-10), total internal reflection occurs when:

$$\phi_2 \geq \arcsin(1/n) \qquad (3\text{-}12)$$

Combining equations (3-12), (3-10), and (3-8), the incident angle at which TIR just occurs can be expressed as:

$$\phi_{TIR} = \arcsin (n \sin (A - \arcsin(1/n))) \qquad (3\text{-}13)$$

therefore when $\phi_1 \leq \phi_{TIR}$, light is totally internally reflected, so no light will come out of the second surface of the prism.

Minimum Deviation (δ_{min})

Deviation is at a minimum level when the ray passes symmetrically through the prism.[23, 24] Thus, at minimum deviation (δ_{min}), the incident and refraction angles have the following relationship:

$$\phi_1 = \varphi_2 \qquad (3\text{-}14)$$

$$\varphi_1 = \phi_2 \qquad (3\text{-}15)$$

substituting equation (3-8) into (3-15), we get:

$$\varphi_1 = \phi_2 = A/2 \qquad (3\text{-}16)$$

Combining (3-16) and (3-9), the incident angle at δ_{min} is related to *n* and *A* by:

$$\phi_{\delta_{min}} = \text{arc sin (n sin }(\frac{A}{2})) \qquad (3\text{-}17)$$

A short C program, whose source code is listed in appendix III, was written to calculate the critical incident angle for total internal reflection and the incident angle of minimum deviation for various prisms, using equations (3-13) and (3-17). Table 3.1 shows part of the calculation results:

Table 3.1 Critical incident angle for total internal reflection and incident angle for minimum deviation at λ = 546.1 nm for four different prism apex angles (A) and two glass materials (BK7, F2)

material	BK7				F2			
A	15°	30°	45°	60°	60°	45°	30°	15°
ϕ_{TIR}	-42.1°	-17.1°	5.8°	29.3°	37.5°	11.4°	-13.1°	-39.4°
$\phi_{\delta_{min}}$	11.4°	23.1°	35.5°	49.4°	54.3°	38.4°	24.9°	12.2°

A counter-clockwise incident angle is defined as positive, and a clockwise incident angle is defined as negative.

Reflection Loss

The total light energy is conserved as a ray passes through the prism:

$$E_i = E_a + E_r + E_t \tag{3-18}$$

where E_i is the energy of the incident ray, E_a is the absorption loss, E_r is the reflection loss, and E_t is the energy of the output ray.

When the incident angle is close to the critical condition, reflection loss E_r increases and reaches a maximum when total internal reflection occurs. Once this happens, E_t equals zero and no light comes directly out of the prism.

At minimum deviation, the passage of a ray through the prism is symmetrical, allowing the largest diameter beam to pass through the prism.[24] In other words, the prism has its greatest effective aperture, and produces the smallest amount of reflection loss at the minimum deviation condition.

To minimize reflection loss and therefore obtain the best sensitivity of the spectrometer, the condition of minimum deviation is preferred, if the dispersion is large enough.

Angular Dispersion

The variation of the deviation angle (δ) with wavelength (λ), $d\delta/d\lambda$, is the angular dispersion of the prism. At a constant incident angle ϕ_1, angular dispersion can be re-written into two parts:

$$\frac{d\delta}{d\lambda} = \frac{d\delta}{dn} \cdot \frac{dn}{d\lambda} \tag{3-19}$$

the first part $d\delta/dn$ depends on the geometry of the optical arrangement, such as the incident angle ϕ_1 and the apex angle of the prism A. The second part $dn/d\lambda$ depends on the dispersive power of the prism material. The effect of the geometry of the optical arrangement on dispersion ($d\delta/dn$) is discussed in more detail below to explain the choice of prism and working conditions. From equation (3-11), $d\delta/dn$ can be expressed as:[24]

$$d\delta/dn = n \sin A\ [\ (n^2 - \sin^2\phi_1)\ (1-u^2)]^{-1/2}$$

$$u = (n^2 - \sin^2\phi_1)^{1/2} \sin A - \sin\phi_1 \cos A \qquad (3\text{-}20)$$

A program in C (source code in Appendix III) was written to calculate the value of dδ/dn at various incident angles (ϕ_1) for prisms with different apex angles and materials. The calculated results are plotted in Figure 3.7 and Figure 3.8.

Figure 3.7 plots the variation of dδ/dn with ϕ_1 for BK7 prisms with apex angles of 15°, 30°, and 60°. The value of dδ/dn has been calculated at 632.8 nm (squares) and 501.7 nm (diamonds). Several trends can be seen from this plot:

First of all, dδ/dn is independent of wavelength (λ), only if ϕ_1 is far away from the incident angle for total internal reflection (ϕ_{TIR}) to occur. For example, in the curve of the 60° prism, when $\phi_1 > 45°$, diamond markers overlap completely with square markers, suggesting that value of dδ/dn is the same for both wavelengths. When $\phi_1 < 45°$, the value of dδ/dn starts to differ at these two wavelengths, and the difference increases when ϕ_1 approaches ϕ_{TIR}. At the incident angle of 30° (0.8° away from ϕ_{TIR}), dδ/dn at 501.7 nm is 41% larger that at 632.8 nm.

Secondly, dδ/dn, therefore angular dispersion, increases when the incident angle approaches ϕ_{TIR}, and the largest dispersion is obtained at ϕ_{TIR}. For instance, with the 30° prism, the ϕ_{TIR} is -17.1°. When ϕ_1 varies from 85° to 5°, dispersion remains relatively constant. But when $\phi_1 < 5°$, dispersion starts to increase obviously. Following the transition region which has a relatively mild increase (i.e. ϕ_1 from 5° to -15°), there is a rapid increase in the dispersion (dδ/dn) when ϕ_1 is very close to ϕ_{TIR} (i.e. ϕ_1 from -15° to -17.1°).

Finally, a prism with a larger apex angle provides a larger dispersion. As described in Section 3.4.1, if the dispersion is large enough, an incident angle of minimum deviation is preferred due to its minimized reflection losses and maximized effective aperture. In Figure 3.7, the cross marker on each curve indicates the angle at which minimum deviation

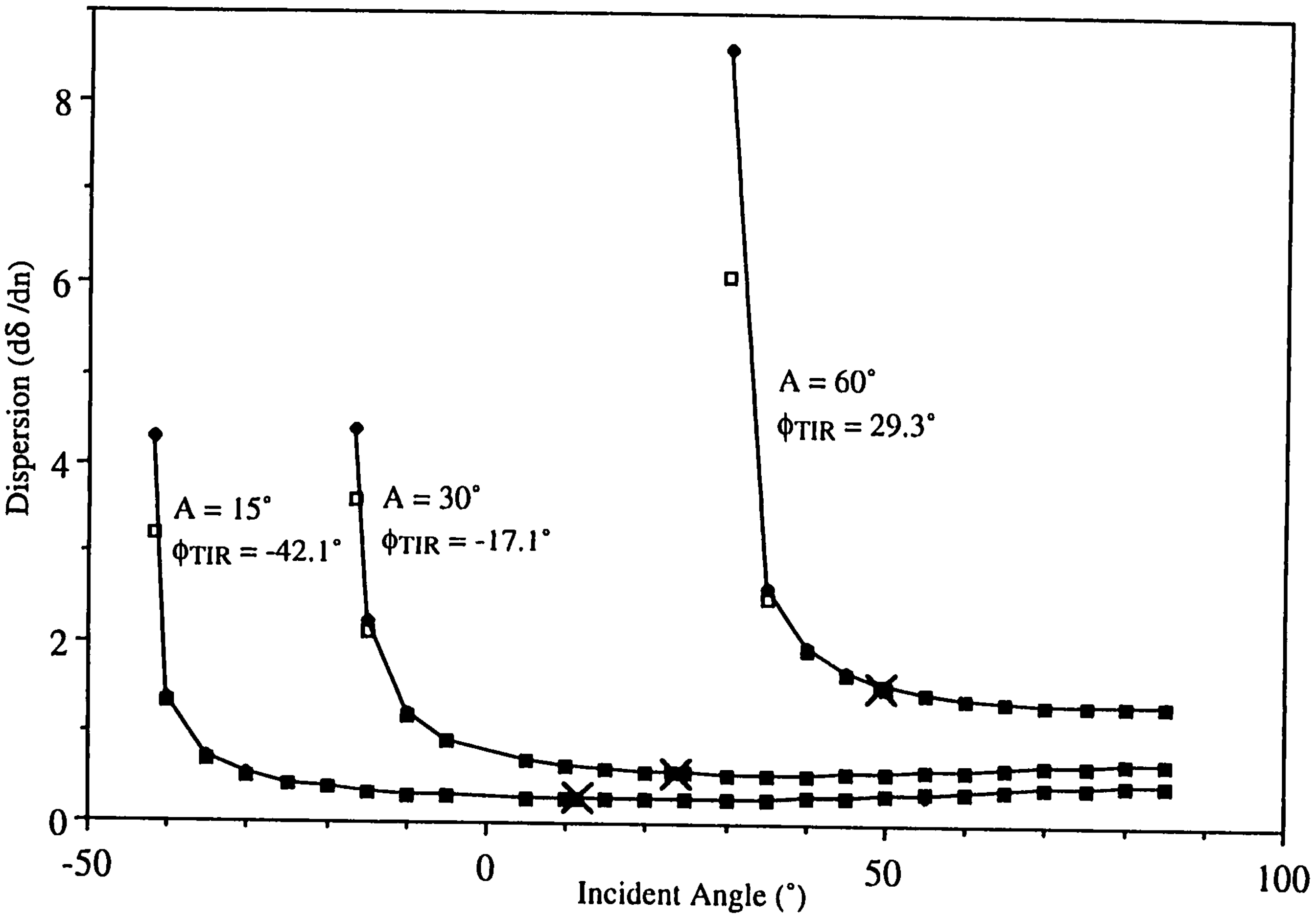


Figure 3.7 Effect of incident angle on dispersion for three BK7 prisms
ϕ_{TIR}: incident angle at which total internal reflection occurs
A: apex angle of the prism
□: λ = 632.8 nm ◆: λ = 501.7 nm ✕: minimum deviation

(δ_{min}) occurs. Comparing the dispersion at the angle of δ_{min} among the three prisms, the 60° prism gives the largest value of $d\delta/dn$, while the 15° prism gives the smallest.

Another factor affecting dispersion is the prism material. F2 and BK7 are two kinds of glass commonly used for commercially available prisms. F2 glass has a larger resolving power than BK7. Thus $dn/d\lambda$, the second part in equation (3-19) of angular dispersion, is larger for a F2 prism than for a BK7 prism.

The effect of these two glass materials on the first part of the angular dispersion equation ($d\delta/dn$) is also investigated. Variation of $d\delta/dn$ with ϕ_1 for these two materials in a 60° prism is shown in Figure 3.8. Similar to Figure 3.7, the value of $d\delta/dn$ is plotted at two wavelengths, where squares represent 632.8 nm while diamonds represent 501.7 nm, and cross markers indicate the δ_{min}. It can be seen that F2 gives a larger value of $d\delta/dn$ than BK7 does at the incident angle of δ_{min}.

The effects of the incident angle (ϕ_1), the apex angle of prism (A), and the glass material on dispersion have been calculated and discussed above. These results suggest that an F2 equilateral prism could be a good candidate as a polychromator for a sensitive spectrometer. To confirm this idea, and to investigate the best optical conditions, computer modeling with a ray-tracing program was performed as discussed in the following section.

3.4.2 Consideration of Optical Arrangement

Ray Tracing

Commercial ray tracing software (Beam Three Optical Ray Tracer, Stellar Software, Berkeley, CA, USA) was used to further model the proposed dispersing optics — two camera lenses aligned with the object sides facing each other and with a prism in-between — which is the structure enclosed in the black box in Figure 3.1.

There are three tables for values to be entered by the user in this software: an optical table, a ray table and a media table. In the optical table, one defines the vertex axis, angle, curvature, and refractive index (or material) of each optical surface, then lists each surface

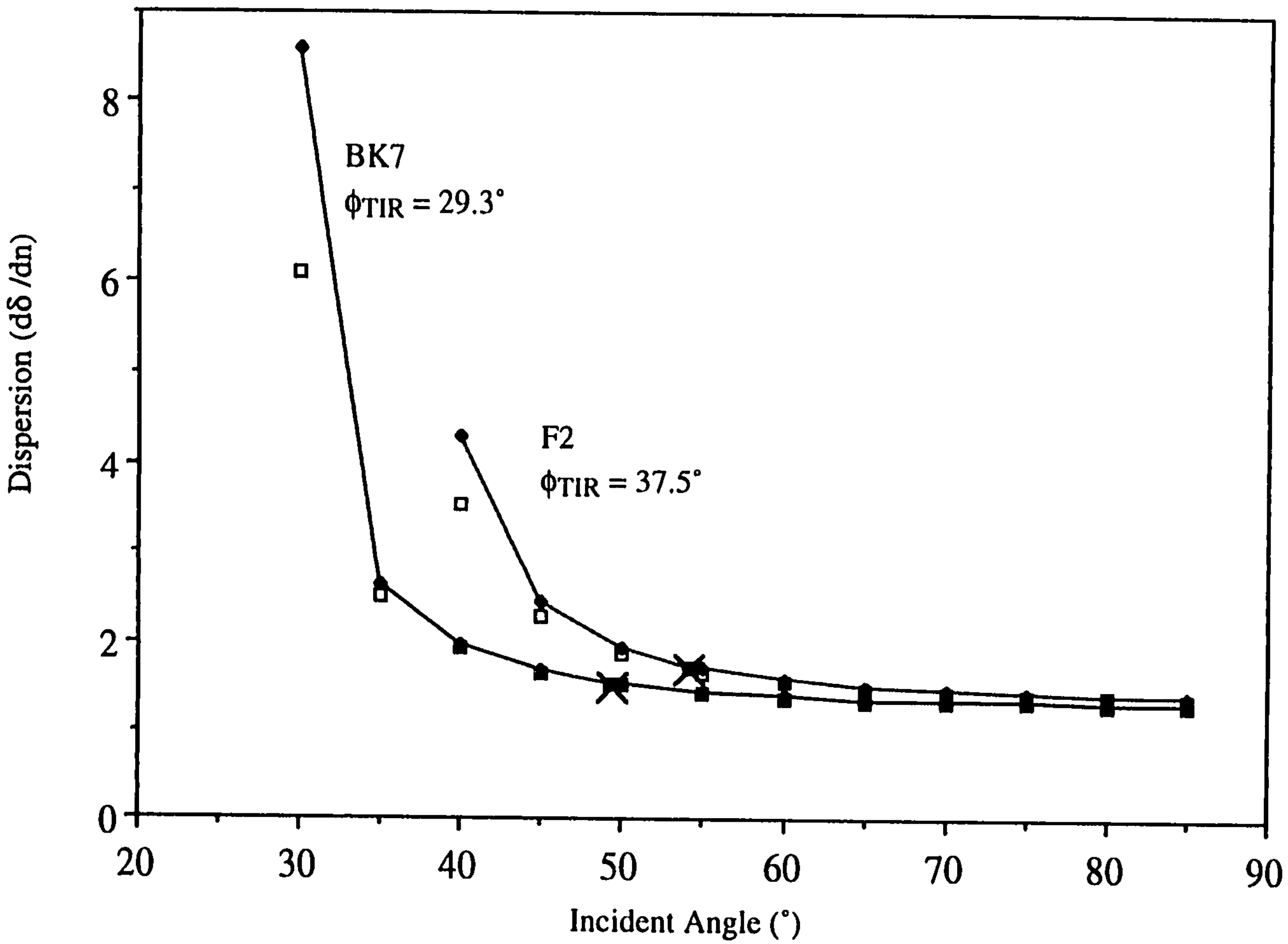


Figure 3.8 Effect of prism material on dispersion for equilateral prism (A = 60°)
ϕTIR: number shown is the value at 546.1 nm

□: λ = 632.8 nm ◆: λ = 501.7 nm ✕: minimum deviation in the sequence by which rays will encounter them. In the ray table, the wavelength and the direction of rays to be tracked are defined. In the media table, the refractive index of each material at every wavelength of interest is given. The software automatically traces each ray by calculating its path through the model, lists the results in the ray table, draws the layout of the optical system, and plots the location of each ray on the final surface.

In most of the following computer modelings for optical arrangements, two plano-convex lenses are used instead of two camera lenses to simplify the modeling process. As previously mentioned, the first lens (collimating lens) collects and collimates light from an object; the collimated light goes through the prism and is dispersed by its wavelength; the second lens (imaging lens) images the spectral dispersed light onto the detector plane. Rays with a different wavelength enter the imaging lens at a different angle, and therefore are imaged at a different location on the detector. A typical ray tracing result is shown in Figure 3.9.

Final Dispersion

Figure 3.9 a shows the optical layout for the modeling. Light propagates along the Z-axis and is dispersed by the prism along the X-axis; the ZX-plane is the ray propagation plane. This modeling assumes that the two plano-convex lenses are achromatic, the refractive index of the two lenses is 1.5167 and the curvature of the convex surface is 0.0387. The prism is a BK7 prism with an apex angle (A) of 30°. Four rays, starting at the same point but going in two different directions, are used in all modeling. In each direction, there are two rays with a different wavelength (red: 632.8 nm, blue: 532.0 nm). The incident angle (ϕ_1) at which rays enter the first surface of the prism is 20°. The numbers within parenthesis in Figure 3.9 a indicate the vertex axis of each surface on the ZX-plane.

The resultant final dispersion is shown in (b). Y_f and X_f are the axis of the detector plane, and dispersion is along the X_f axis. The difference in terms of X_f (ΔX_f) between

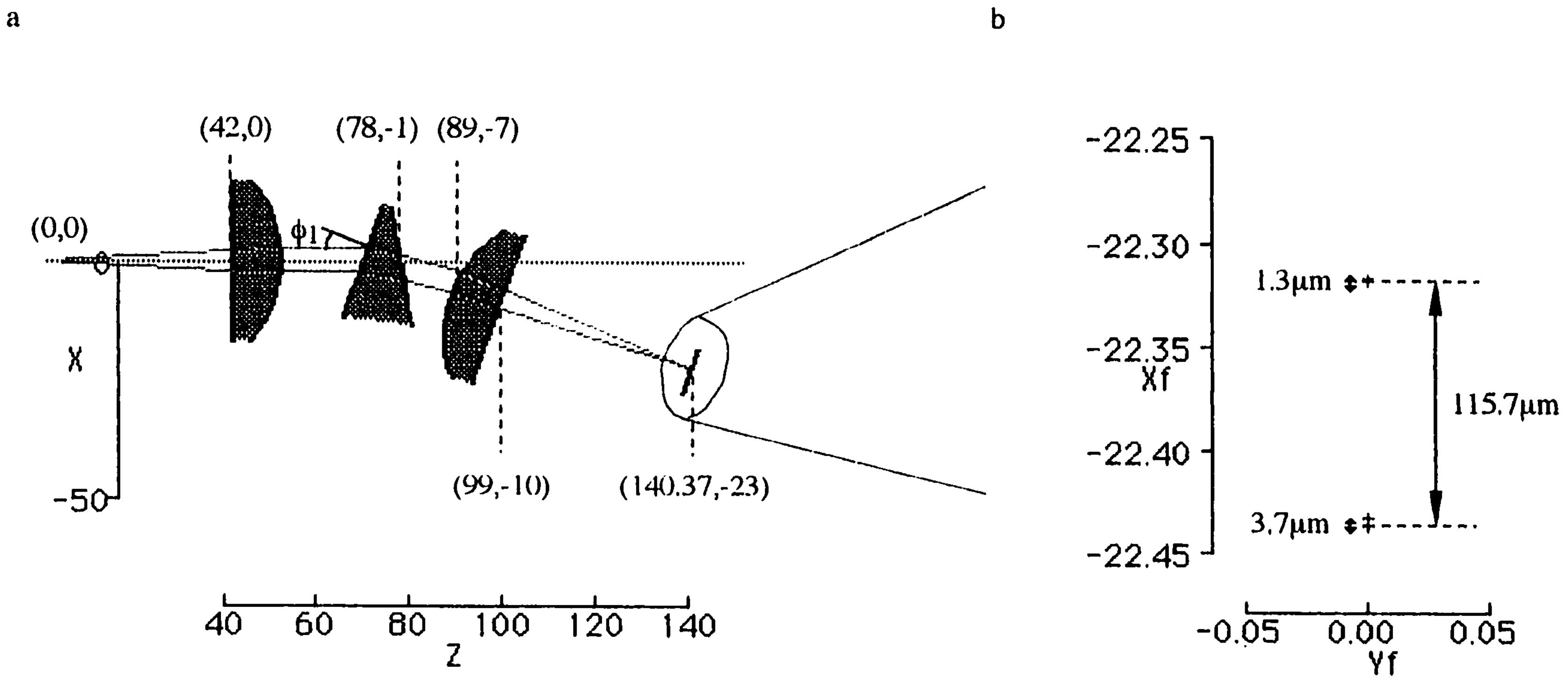


Figure 3.9 Optical system for ray tracing
a) optical layout prism: A = 30°, BK7 plano-convex lenses: n = 1.5167
b) resulted dispersion on detector plane
incident angle to prism: ϕ_1= 20°, red: 632.8 nm blue: 532.0 nm

the average location of red cross markers and that of blue cross markers is the final dispersion achieved by this optical system, 115.7 μm in this case. The final dispersion is different from the angular dispersion (represented by dδ/dn in most of the discussions above). Final dispersion describes the dispersion after the imaging lens, and is the distance between the images of two different wavelengths on the detector after the collimated rays with the same wavelength have been focused into a point. On the other hand, angular dispersion describes the dispersion power of the prism alone without taking the effect of the imaging lens into account.

ΔX_f between the cross markers of the same color (1.3 μm for red, 3.7 μm for blue) reflects the focal power of the system. If it is perfectly focused, all the cross markers of the same color will be in exactly the same position. Ray-tracing shows that rays with different wavelengths (about 100 nm apart in most cases) have different focal distances, which is sensible because they reach the imaging lens at different angles.

By rotating the prism, and accordingly adjusting the location and angle of the second lens and detection plane of the optical system shown in Figure 3.9 a, a curve showing the change in final dispersion with the incident angle is calculated by ray-tracing and plotted in Figure 3.10 b. The variation of dδ/dn with incident angle from the calculations discussed in the last part of Section 3.4.1 is shown in Figure 3.10 a, which is similar to the curve in the middle of Figure 3.7, except that the diamond markers here represent a different wavelength. Comparing the curve in Figure 3.10 a with that in Figure 3.10 b, it can be seen that both curves have a similar shape, suggesting that both angular dispersion and final dispersion change with the incident angle similarly. Results of ray tracing confirm the calculations done in Section 3.4.1: dispersion exponentially decreases as the incident angle becomes further away from ϕ_{TIR}. Again, the cross marker on each curve of Figure 3.10 indicates the minimum deviation condition. The final dispersion at the minimum deviation condition achieved with two plano-convex lenses and a BK7 30° prism is approximately 115 μm (Figure 3.10 b).

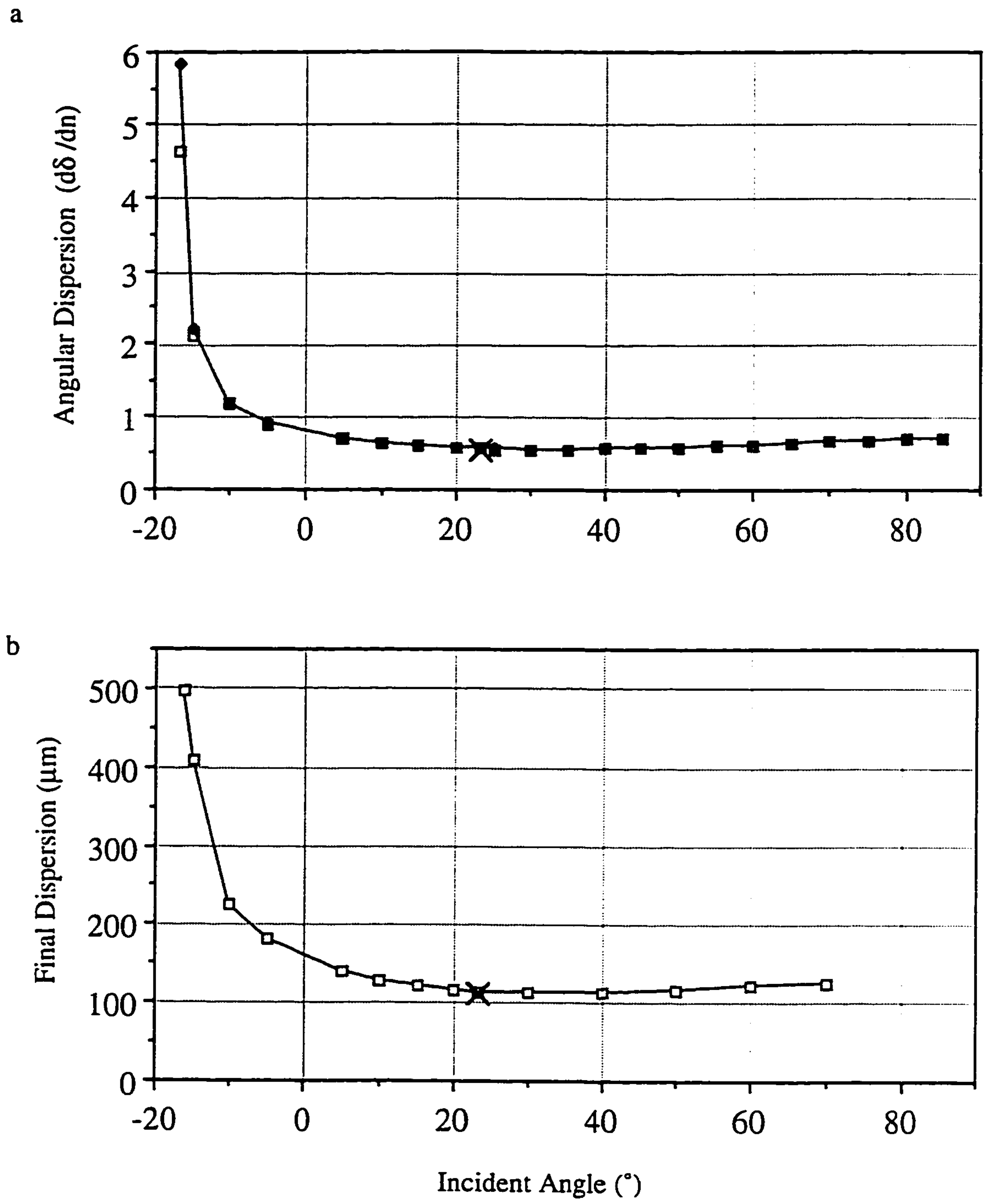


Figure 3.10 a) angular dispersion by calculation
□ : $\lambda = 632.8$ nm, ◆ : $\lambda = 532.0$ nm
b) final dispersion by ray tracing
(between 632.8 nm and 532.0 nm)
prism: BK7, A = 30° ✕ : minimum deviation

Effect of Imaging Lens on Final Dispersion

With the powerful ray tracing tool Beam Three, useful optical considerations can be made before the actual experiments. All of the following modelings use the same plano-convex lens pair as in Figure 3.9. Wavelength of the rays, Apex (A) and material of the prism, and the incident angle (ϕ_1) at which rays enters the first surface of the prism are specified in each figure.

It has been proven by ray tracing that the distance between the first lens and the prism has no effect on final dispersion. But the distance between the second lens and the prism does affect the final dispersion. The closer the imaging lens is to the prism, the larger the final dispersion. Figure 3.11 shows an example. A 30° BK7 prism is placed at a -10° incident angle of light to the prism, and this angle falls in the transition region of the curves in Figure 3.10. Two cases are compared here. In Figure 3.11 a, the distance between the prism and the imaging lens is 14.4 mm, while the resultant final dispersion is 260.4 µm over a 100 nm spectral range. In Figure 3.11 b, the imaging lens is placed further away at 90.4 mm, and the resulting final dispersion is 217.3 µm. The distance in Figure 3.11 a is 16% of that in Figure 3.11 b, but results in a 20% increase in the final dispersion.

Another investigation has been done with the same optical system in Figure 3.11 but at a 20° incident angle (ϕ_1) which is close to the minimum deviation condition of the prism. The results are listed in Table 3.2., where again two cases are compared. The distance between the prism and the imaging lens in case (a) is 12.5mm, which is 19.6% of the 63.9 mm in case (b). The resulting final dispersion is 115.7 µm in (a), which is 1.7% larger than the 113.8 µm in (b). At this incidence, the effect is not obvious.

A similar investigation has been done with the similar optical system in Figure 3.11 but for the prism of interest — the F2 equilateral prism at its minimum deviation angle. As listed in Table 3.2, a distance of 12.2 mm in case (a) results in a final dispersion of 627.8 µm, while a distance of 72.3 mm in case (b) results in a final dispersion of 591.5 µm. The

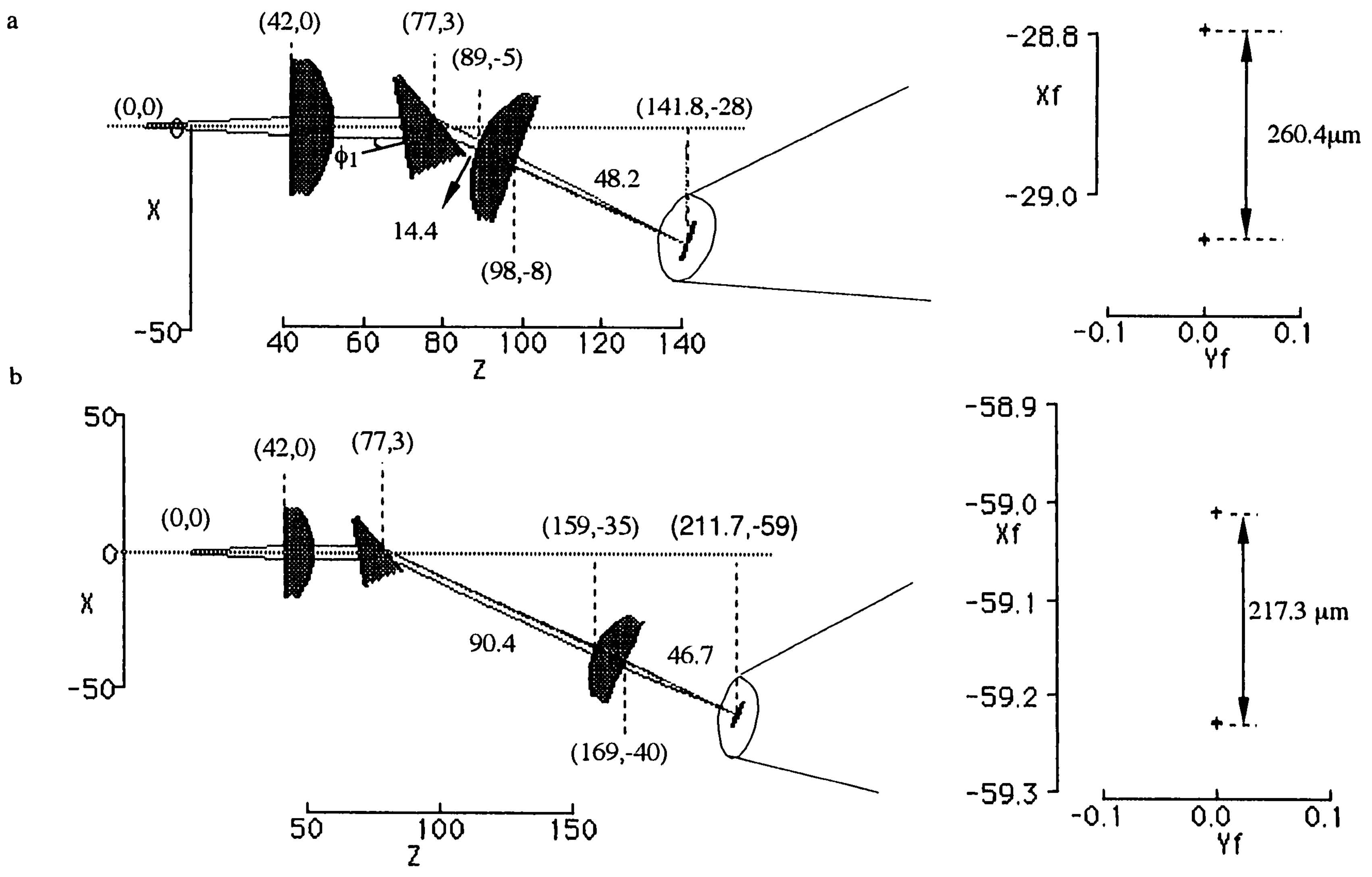


Figure 3.11 Effect of distance between prism and imaging lens a) 14.4 mm, b) 90.4 mm on dispersion
ϕ_1: -10°   prism: 30°, BK7   red: 632.8 nm, blue: 532.0 nm

distance in (a) is 16.9% of that in (b), but resulting in a 6% larger final dispersion over a 100 nm spectral range than in (b).

Table 3.2 Final dispersions at various distances between the prism and the imaging lens for two prisms at different incident angles

prism		ϕ_I	distance (prism — 2nd lens)	final * dispersion
apex	material			
30°	BK7	-10°	a) 14.4 mm	260.4 μm
			b) 90.4 mm	217.3 μm
		20°	a) 12.5 mm	115.7 μm
			b) 63.9 mm	113.8 μm
60°	F2	54.1°	a) 12.2 mm	627.8 μm
			b) 72.3 mm	591.5 μm

* the dispersions cover a 100 nm spectral range from 532.0 nm to 632.8 nm

In summary, the distance between the prism and the imaging lens has a large effect on dispersion when the incident angle is close to ϕ_{TIR}, but it has a relatively small effect at the incident angle for minimum deviation. A smaller distance between the prism and the imaging lens results in a larger dispersion.

The imaging lens is fixed with the CCD camera with a certain distance in-between, so they are always parallel to each other and forms the detector. The angle at which the color dispersed beam enters the detector is another factor to be considered. Figure 3.12 shows the effect of this angle on final dispersion and focusing. The angle at which each surface intercepts with the XY-plane is called the pitch angle of the surface. The pitch angles of the imaging lens and detector are -25°, -10°, and 0° in Figure 3.12 a, b, and c, respectively. Rays are approximately perpendicular to the surface of the imaging lens in

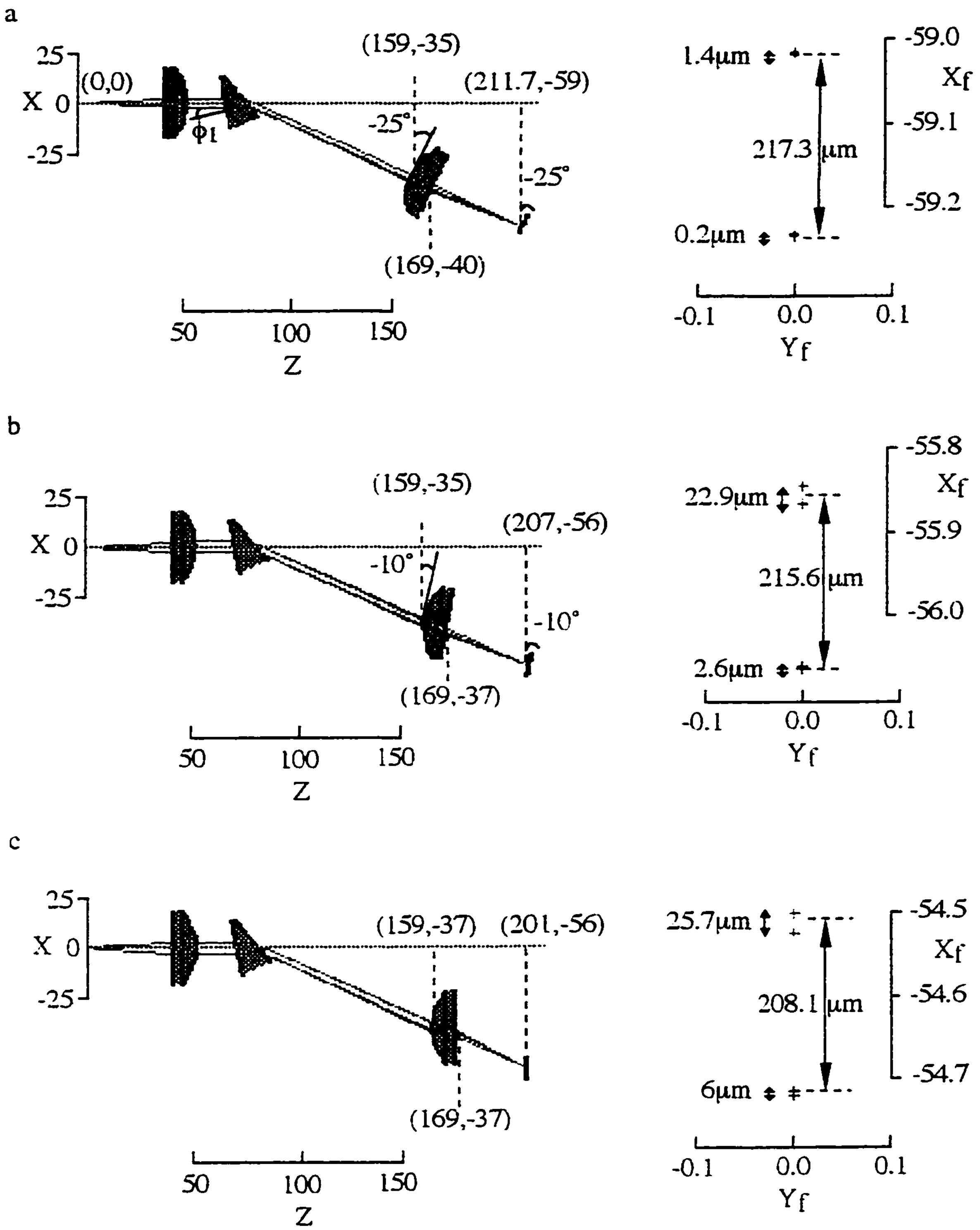


Figure 3.12 Effect of angle of imaging lens on focusing and dispersion
imaging lens a) perpendicular to rays, pitch -25°,
b) pitch -10°, c) pitch 0°
prism: BK7, 30° ϕ_1: -10° red: 632.8 nm, blue 532.0 nm

Figure 3.12 a. This is referred to as the perpendicular condition. Figure 3.12 b and Figure 3.12 c show progressive deviation from the perpendicular condition.

The perpendicular condition is the best condition for focusing. As shown in Figure 3.12 a, rays of both wavelengths are focused when they strike onto the imaging lens approximately vertically. Focusing becomes a problem in Figure 3.12 b and c. For instance, instead of being 1.4 µm apart as in Figure 3.12 a, the two rays of 632.8 nm (red) are 22.9 µm apart and 25.7 µm apart in Figure 3.12 b and c, respectively.

However, the effect of the perpendicular condition on final dispersion is not dramatic, as the difference in the final dispersion between these three cases depicted in Figure 3.12 is within 5%.

Through this modeling, a conclusion can be reached: color-dispersed beams will be focused best when they strike the imaging lens at an angle close to 90°. It is more difficult to focus when beams strike the imaging lens at an angle that deviates from 90°.

There is another factor affecting the final dispersion — the type and material of the imaging lens, which will be discussed in more detail in Chapter 6. All of the above ray tracings were made with a pair of plano-convex lenses with the curvature of the convex surface of 0.0387, and the refractive index of the lens was assumed to be 1.5167 for all wavelengths. These parameters give a focal length of 50 mm of the plane-convex lens. Actually, two 55 mm f/1.4 SLR camera lenses will be used in the designed spectrometer. Ray tracing with camera lenses was done to estimate the amount of dispersion to be expected in the spectrometer. Because the lens data of a 50 mm f/1.8 SLR camera lens is available,[21] this lens was used in the ray tracing to model the actual optical system more closely.

Figure 3.13 compares the modeling results using two plano-convex lenses (Figure 3.13 a) with that of using two 50 mm f/1.8 camera lenses (Figure 3.13 b). The camera lens is a complex lens as there are 13 surfaces required to describe the 50 mm f/1.8 lens. It is difficult to rotate all the surfaces of the second camera lens to make it

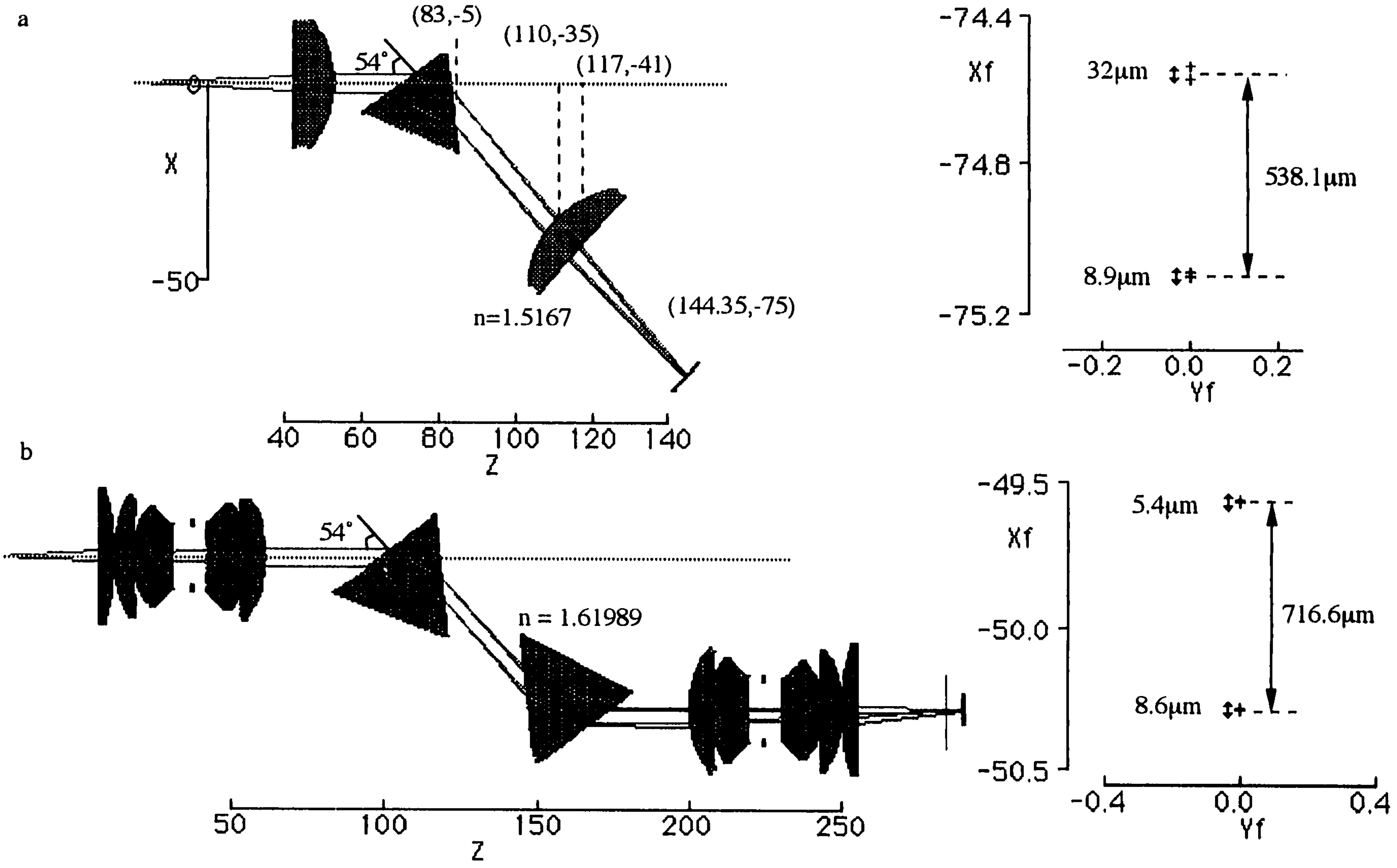


Figure 3.13 Effect of imaging lens a) plano-convex lens b) camera lens on dispersion
prism: F2, 60° ϕ_1: 54° (minimum deviation) red:643.9 nm, blue:546.1 nm

perpendicular to the rays, as is done in the plano-convex lens modeling (Figure 3.13 a). In order to simplify the modeling process, a fake prism is added in-between the dispersing prism and the second camera lens to make the rays coming from the dispersing prism parallel to the Z-axis. The fake prism has the same geometric structure as the dispersing prism, except that it is arranged upside down and it has the same refractive index of 1.61989 for rays of all wavelengths.

With the F2 equilateral prism and at the incident angle for minimum deviation, a final dispersion of 716.6 µm is achieved with the two-camera-lens system over a nearly 100 nm spectral range from 546.1 nm to 643.9 nm. Meanwhile, a dispersion of 538.1 µm is achieved with the simplified plano-convex lenses system. Also, it can be seen that the camera lenses provide better focusing than the plano-convex lenses.

Similar modeling has been done for different type of prisms by replacing the dispersing prism and fake prism with the corresponding prisms. The final dispersion at the minimum deviation condition calculated with ray-tracing for each prism is listed in Table 3.3.

Table 3.3 Final dispersion at the minimum deviation condition to be expected over a 100 nm spectral range from 546.1 nm to 643.9 nm for various prisms

material	BK7		F2		
apex	30°	60°	30°	45°	60°
final dispersion at δ_{min}	117 µm	310 µm	240 µm	405 µm	716 µm

As can be seen from Table 3.3, a F2 prism has better resolving power, resulting in a much larger final dispersion than a BK7 prism. At the minimum deviation condition, a 60° F2 prism provides a final dispersion of 716 µm, while a 45° F2 prism provides a dispersion of 405 μm. Both of them could be used as the polychromator of the spectrometer. The 60° prism was chosen because of its availability.

The spectral range of interest for four-color DNA sequencing with the conventional four-dye chemistry is about 110 nm from 530 nm to 640 nm. Ray tracing results show that with the proposed spectrometric system (two camera lenses aligned face-to-face with a F2 equilateral prism in-between), a final dispersion of more than 700 μm can be achieved at the minimum deviation condition of the prism. The spectra of the whole emission range of the four dyes will be spread over approximately 100 pixels of the CCD chip, because the pixel size of the CCD is 7 μm.

Results of the calculations and ray-tracing suggest that the F2 equilateral prism can provide enough dispersion at the minimum deviation condition for four-color DNA sequencing. In the next section, the prism is experimentally tested with the actual setup.

3.5 PRELIMINARY STUDY OF THE RESOLVING POWER OF F2 EQUILATERAL PRISM

As predicted by the calculations and ray-tracings in the previous section, with an F2 equilateral prism, it is possible to achieve both sufficient dispersion and minimized optical loss. This is achieved by operating the prism at the optical condition of minimum deviation, which occurs when collimated light strikes the surface of the prism at an incident angle of 54.3° at λ = 546.1 nm. In this section, the optical system is further tested to confirm the predicted results.

3.5.1 Experimental

All the experiments have been done with the setup shown in Figure 3.14 with dyes continuously injected through the capillaries. Commercial software IP-Lab was used to operate the CCD camera and to analyze the image data.

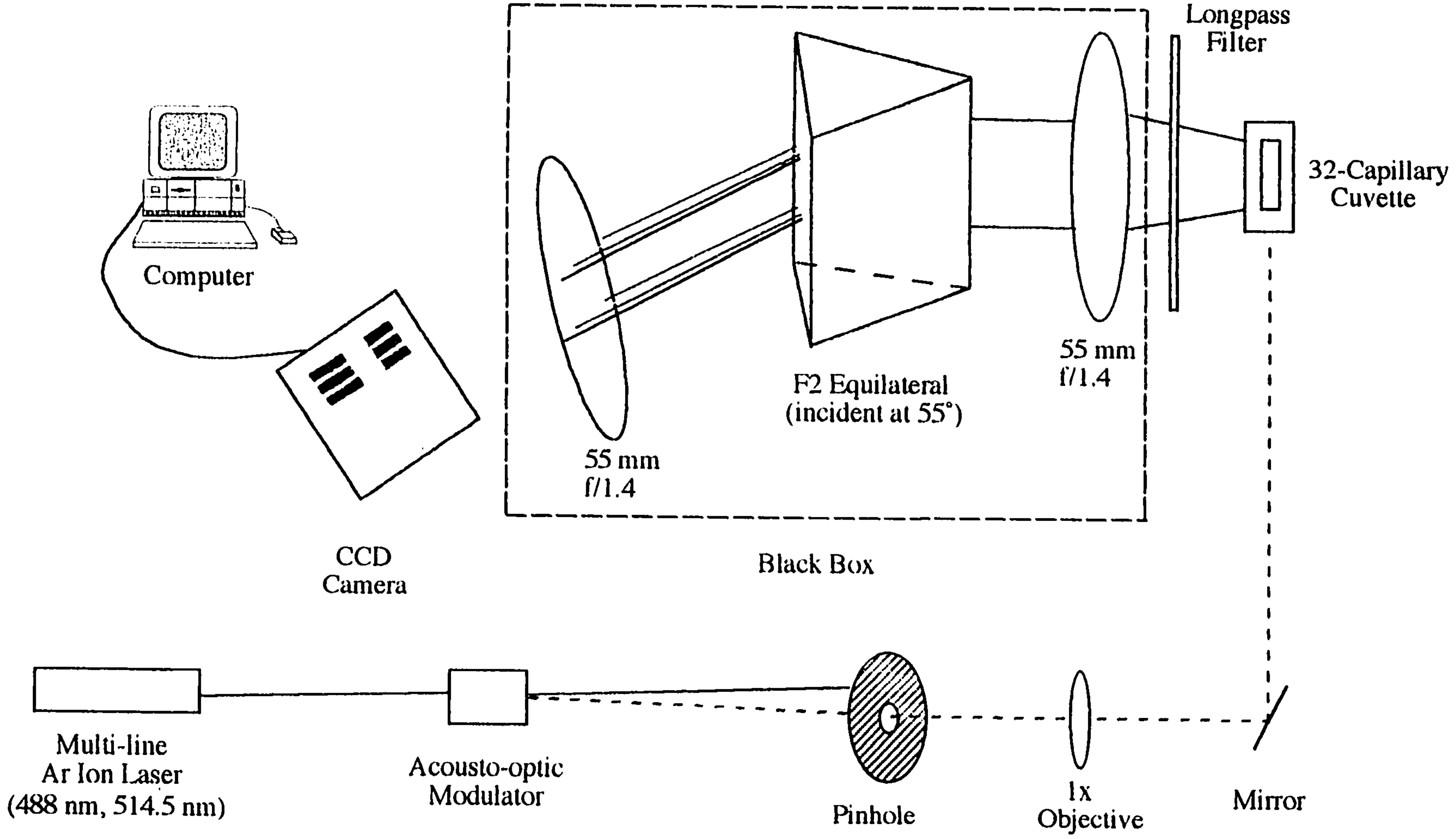


Figure 3.14 The 32-capillary spectrometer

Orientation of Cuvette

In most instruments in our group, the cuvette stands on its end as shown in Figure 2.2. This orientation allows the capillaries to be replaced without removing the cuvette, therefore the alignment is not disturbed. In order to disperse the spectra of all 32 capillaries in a direction other than the direction of the capillary array, while also considering the easiest way to hold the prism, the cuvette is arranged to lie on its side, as shown in Figure 3.15. Sheath-flow buffer flows into the cuvette from the inlet below the capillary bundle, and is pushed through the cuvette by pressure from the sheath flow reservoir, and then goes to a waste container at the end of the cuvette. Outlet of the sheath-flow above the capillary bundle can be used to purge bubbles from the incoming sheath-flow and is closed during the run.

The sideways cuvette orientation is actually better than the standing-up orientation in Figure 2.2 in terms of being able to get rid of bubbles. Bubbles tend to go up after getting in the cuvette through the sheath-flow or the capillaries. If a cuvette stands on its end as in Figure 2.2, higher pressure is needed to push the bubbles all the way down to the waste outlet than if a cuvette lies on its side (Figure 3.15).

As shown in Figure 3.14, the laser beam is deflected by the acousto-optical modulator, and the first order of the deflected laser beam passes through the pinhole. It is then focused by a 1X microscope objective and shines through the cuvette from bottom to top, inducing 32 fluorescent spots. The first camera lens collects fluorescence at the back of the cuvette, and then collimates the light. The prism is fixed onto a rotationable stand, so that the incident angle of fluorescent light to the prism can be easily changed and approximately measured. This rotator provides an angular precision of half a degree, and gives a reasonably accurate measurement of changes in angle. But the initial angle position is estimated by eye-aligning one edge of the prism with the optical table, so the absolute angle read from the rotator may not be accurate. The spectrum from every capillary is

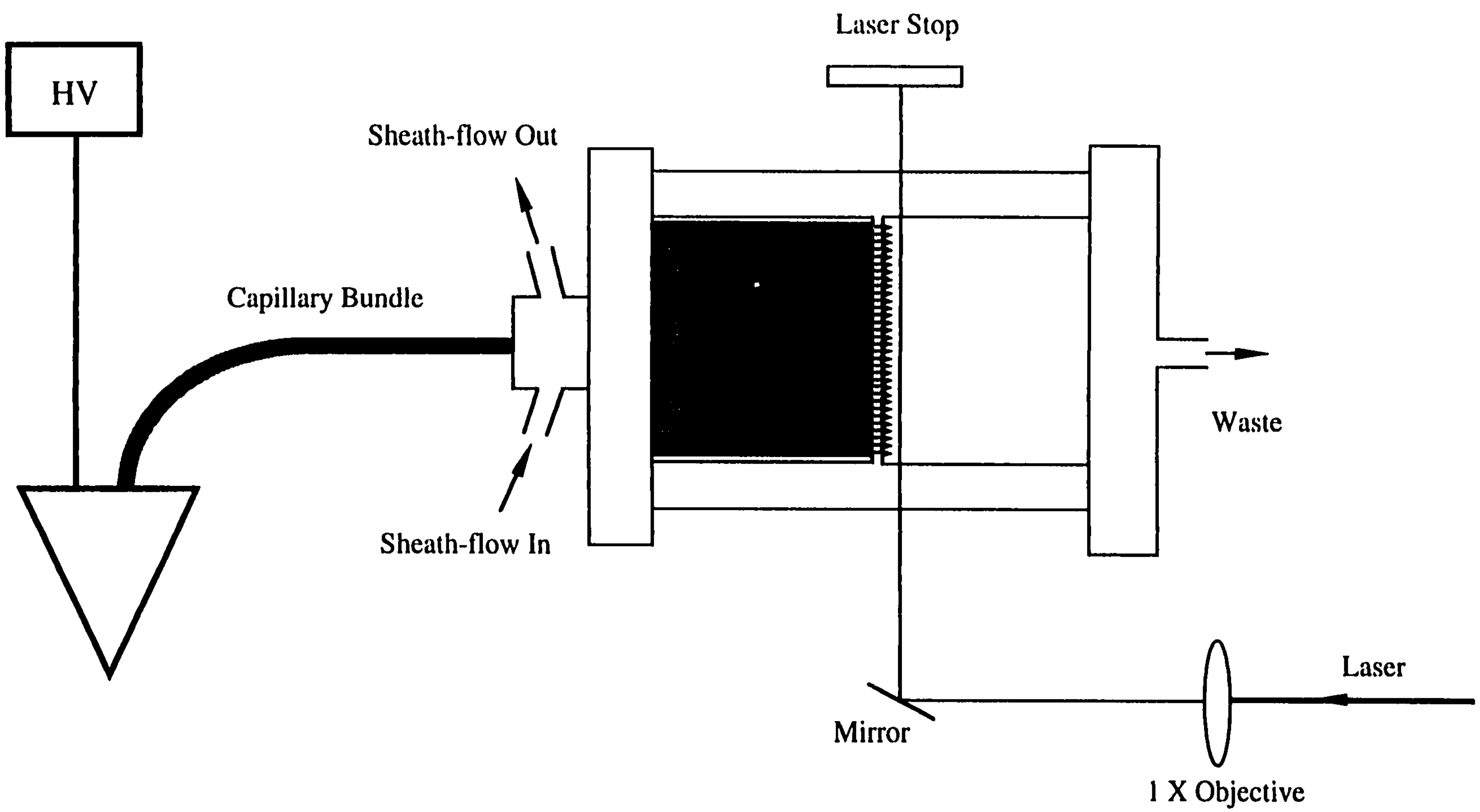


Figure 3.15 Sheath-flow cuvette lying on its side

dispersed horizontally by the prism, and then imaged onto the CCD chip by the second camera lens.

3.5.2 Results and Discussion

Spectral resolution

In order to determine the spectral resolution of the system and the exact position of the fluorescent bands and the scattered laser lines, which are needed for the data acquisition software design, spectral images were taken at approximately the minimum deviation condition of the prism (about 55° incident angle) in Figure 3.16.

In Figure 3.16 a, two laser lines (488 nm and 514.5 nm) are used for excitation and the image is taken without any detection filter. The resultant image has two scattered laser lines followed by fluorescent bands. After blocking the blue line of the multi-line laser with a 514.5NB3 filter, the first line image in Figure 3.16 a disappears, and the image becomes that in Figure 3.16 b. This indicates that the first line image in Figure 3.16 a must be the scattering of the blue laser line (488 nm), and the second line image is the scattering of the green laser line (514.5 nm). Both lines disappear after using a 530 nm long-pass detection filter (Figure 3.16 c). These three images are taken with different exposure times in order to avoid saturation of the CCD chip and also to achieve similar image intensity. The exposure times are 5 ms, 200 ms, and 400 ms for Figure 3.16 a, b, and c, respectively.

The spectral difference between the two scattered laser lines is 26 nm (514 nm - 488 nm). The two scattered laser lines are separated from each other by approximately 40 pixels (Figure 3.16 a). The spectral resolution of the dispersion optics at about 500 nm and an incident angle of about 55° can be calculated as: 40 pixels / 26 nm = 1.54 pixels/nm, which is 10.8 μm/nm (1.54 pixel/nm * 7 μm) since the pixel size of the CCD is 7 μm. Because the dispersion of a prism is non-linear and is less in the red region than in the blue region, the spectral resolution in the fluorescence region is smaller than

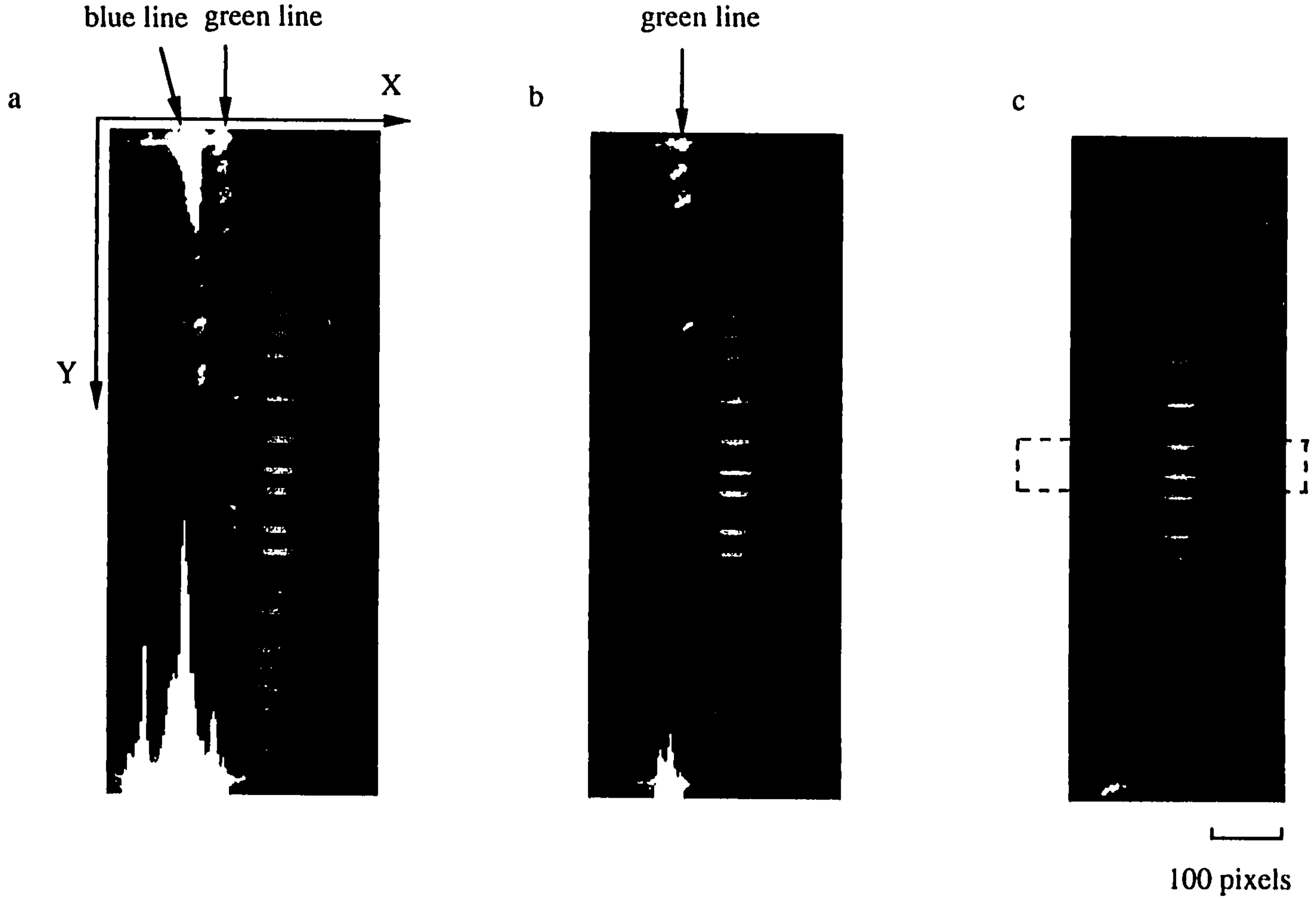


Figure 3.16 Spectra of R6G at an incident angle of ~55°
a) 488 nm + 514.5 nm excitation without detection filter, CCD: 5 ms exposure
b) 514.5 nm excitation without detection filter, CCD: 200 ms exposure
c) 514.5 nm excitation with LP530 filter for detection, CCD: 400 ms exposure
the two capillaries in the dashed rectangular window will be further discussed in Figure 3.18

1.54 pixels/nm. This experiment also proves that the spectral axis is from blue to red as one moves from left to right on the image.

From Figure 3.16, we can see that there is an optical distortion in the system: the spectral images are not straight but a little bowed from the edge to the center of the cuvette. The reason for this optical distortion is due to the prism in the optical train.[25] This distortion is important for the data acquisition software design and has to be taken into account to correctly locate each spectral band. (See Chapter 4.)

Perspective View of Spectra

The spectral data can be examined differently in 3-dimensional perspective view. Several concepts have to be introduced before we take a closer look at the data of the two capillaries enclosed in the dashed rectangle in Figure 3.16 c. As shown in Figure 3.17, assuming that the fluorescence band (wavelength) is along the X-axis, the capillary array is aligned along the Y-axis, and the intensity is recorded in the Z direction, the angle which the viewing direction forms with the XY plane is called the Attitude (θ). The angle which the projection of the viewing direction on the XY plane forms with the Y-axis is called the Direction (φ). A perspective view angle is often labeled as (θ,φ).

Figure 3.18 shows perspective views at a (60°, 60°) view angle of spectra of R6G for the two selected capillaries in Figure 3.16 c. The spectral data in Figure 3.18 is shown in the same order as in Figure 3.16. Spectra under multi-line excitation without a detection filter are shown in Figure 3.18 a. Spectra under a single green line excitation are shown in Figure 3.18 b (without a detection filter) and Figure 3.18 c (with a LP530 filter). The frontmost peak in each of the three graphs of Figure 3.18 corresponds to the lower band of the two selected capillaries in Figure 3.16 c.

In Figure 3.18, the light intensity is recorded as a peak and also encoded with color. With the help of the color-encoded intensity, we can clearly see the spectral

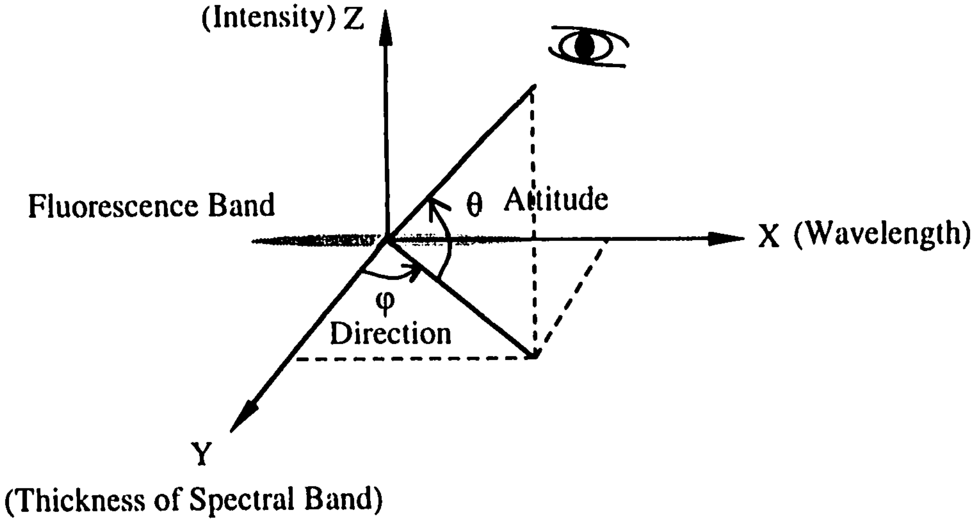


Figure 3.17 Diagram of perspective view angle (Attitude angle θ, Direction angle φ)

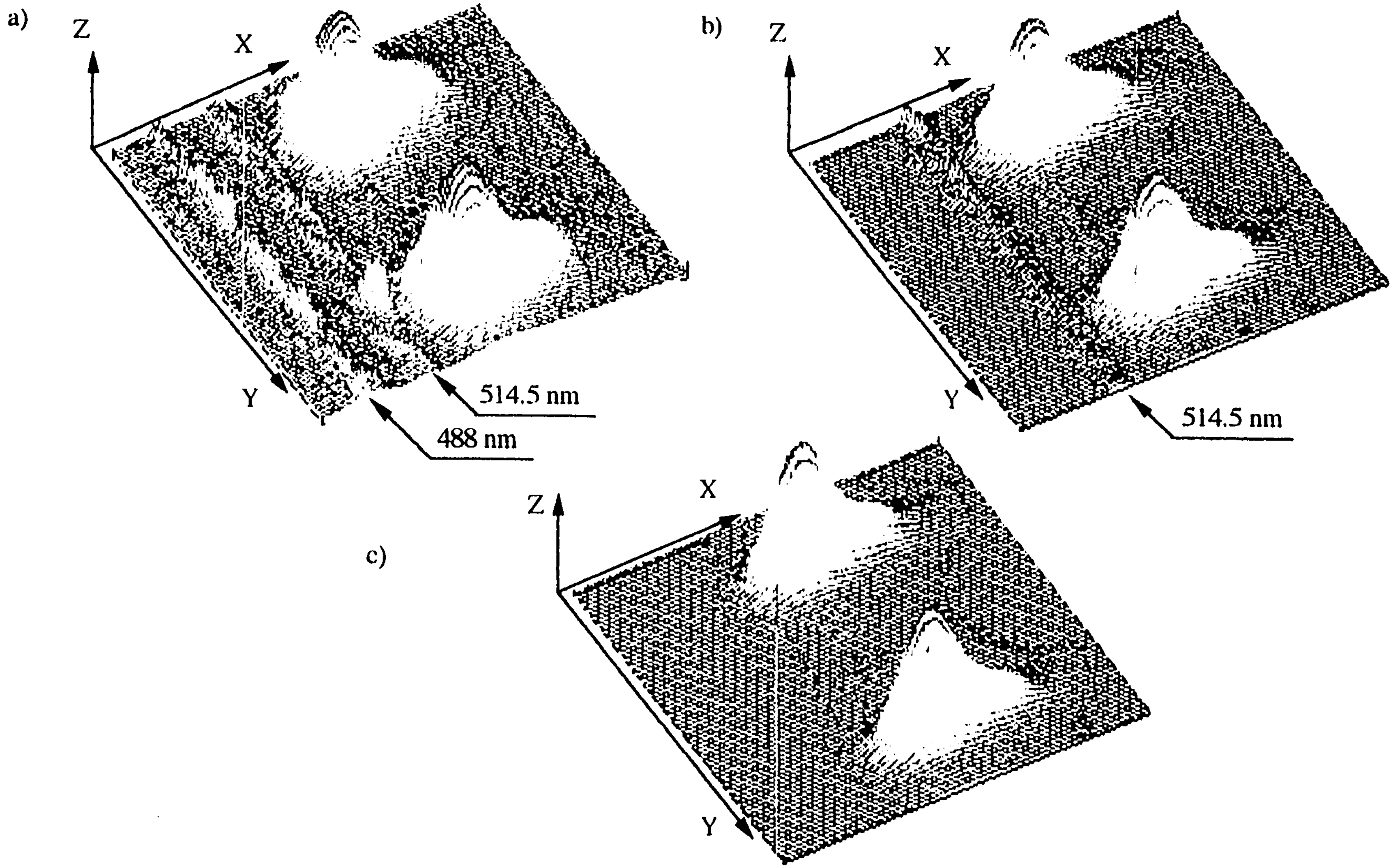


Figure 3.18 Spectra of R6G for two selected capillaries in Figure 3.16 at view angle: (θ = 60°, φ = 60°)
excitation: a) 488 nm + 514.5 nm b, c) 514.5 nm
detection: a, b) without filter c) with LP530 filter

locations of the Rayleigh scattering of the laser lines, the fluorescent spectra of R6G, and the change of spectra in the presence or absence of a detection filter.

Comparing the perspective data (Figure 3.18) with the image data (Figure 3.16), with the addition of the Z-axis and the color-encoded intensity, the perspective data shows spectral information more clearly and directly than the image data. Details such as the spectral shape and the variation of spectral intensity with band thickness are clearly shown in the perspective view (Figure 3.18). Figure 3.16 and Figure 3.18 also demonstrate the ability of the detection filter to efficiently minimize the scattering background by allowing only those wavelengths within the designated spectral range to pass through the filter.

Identification of Dyes by Band Location

One major advantage of a multiple capillary instrument is that the comparison of different samples and/or different electrophoresis conditions can be done simultaneously. Here, 15 out of 32 capillaries are used to see the difference in the spectra of two different dyes, fluorescein and R6G (10^{-7} M). As shown in Figure 3.19, capillaries 1, 3 and 11 to 15 are continuously injected with R6G; capillary 2, and 4 to 10 are continuously injected with fluorescein. The image is taken by the CCD camera with 2 x 2 binning.

In Figure 3.19 a, the incident angle to the prism is 55° which is an optical condition close to the minimum deviation condition — the proposed working condition of the spectrometer. At this condition, the spectra are condensed. The spectral band locations of fluorescein and R6G are shifted relative to each other, where the spectra for fluorescein shifts to the blue region, and R6G to the red. The two dyes can be easily distinguished by their different locations of spectral bands. This difference provides one way to identify different dyes.

Decreasing the incident angle to about 40°, as shown in Figure 3.19 b, dispersion increases as expected. The difference in the spectral band locations of the two dyes relative to each other is still observable. This difference disappears when the incident angle

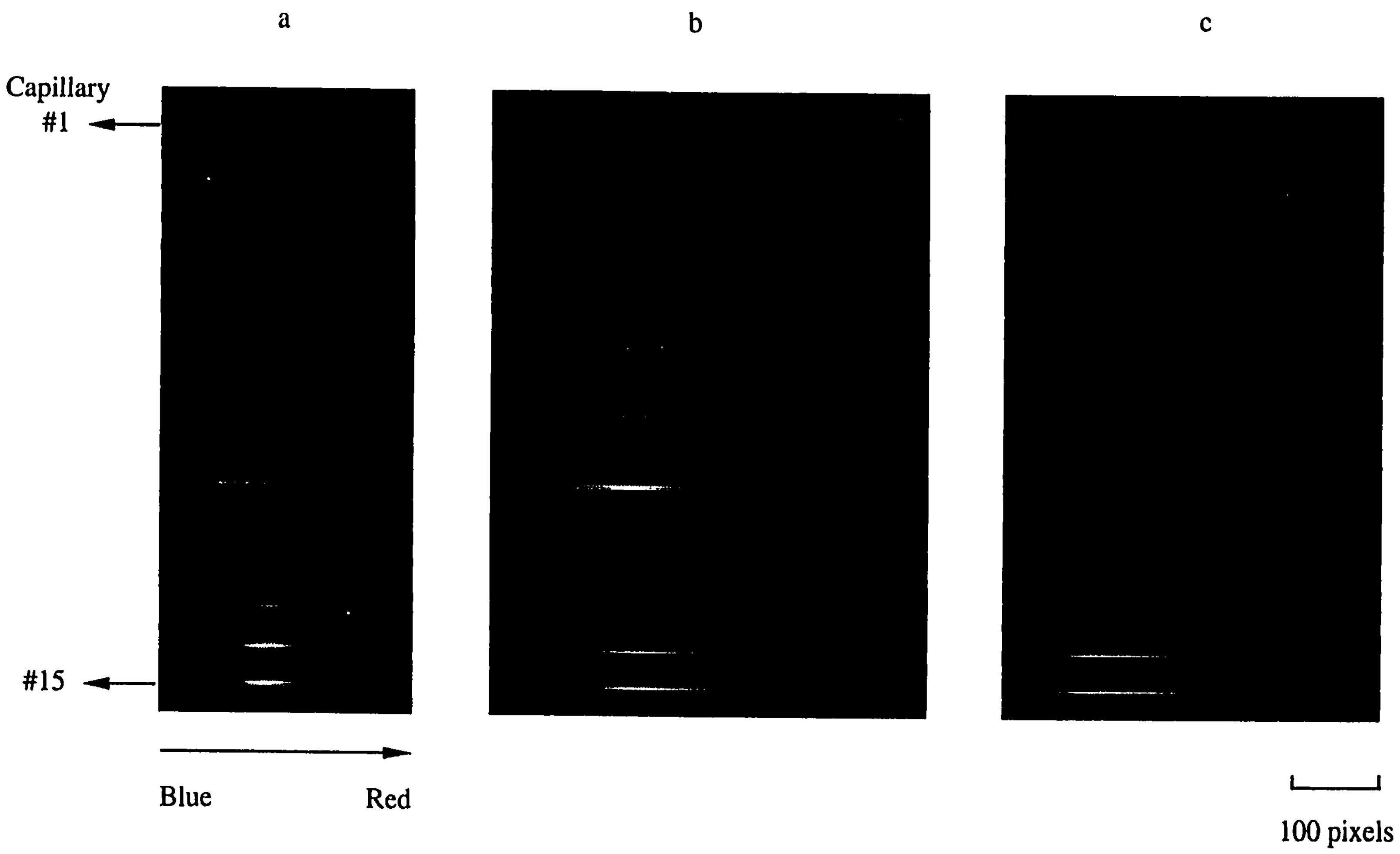


Figure 3.19 Spectra of R6G (#1, 3, 11-15) and fluorescein (#2, 4-10) at different incident angles a) 55°, b) 40°, c) 39.5° excitation: 488 nm, detection filter: LP515

decreases half a degree more in Figure 3.19 c, it seems that there is no longer much difference between the spectra of the two dyes. This is because part of the short wavelength region of the fluorescein spectra has undergone total internal reflection.

According to earlier calculations, ϕ_{TIR} decreases while wavelength increases: the ϕ_{TIR} is 36.9° at 632.8 nm, 37.5° at 546.1 nm, and 37.8° at 514.5 nm. In Figure 3.19, a 515 nm long-pass filter is used for detection, so the total internally reflected spectral wavelengths of the fluorescein spectra in Figure 3.19 c must be longer than 515 nm. Therefore the incident angle in Figure 3.19 c must be smaller than 37.8°. Considering the spectral difference between fluorescein and R6G, the wavelengths undergoing total internal reflection should be between 515 nm and 540 nm. So the actual incident angle in Figure 3.19 c should be between 37.8° to 37.5°, instead of the measured angle 39.5°. As mentioned before, the absolute angle read from the prism rotator may not be accurate. In this case, it is about 2° more than its true value. This experiment provides a way to confirm the observed value of the incident angle.

Identification of Dyes by Emission Maximum

Four ABI dyes, which are often used for DNA sequencing, have been investigated near the minimum deviation condition. Two pixels along the Y-axis of Figure 3.17 at the center of the image band have been binned together for every X pixel. The resulting spectra are plotted in Figure 3.20. All the dyes are excited by a 488 nm blue line, so the spectrum of Rox is weak due to the inefficient excitation. The vertical axis is the light intensity, and it has different scales in the five spectra shown in Figure 3.20. Emission spectra[26] of Fam, Joe, Tamra and Rox overlap with each other, and the wavelength for maximum emission (λ_{max}) for each dye is known. The λ_{max} has been labeled on each measured spectral peak in Figure 3.20.

From the measured spectral data of the four dyes in Figure 3.20 a - d, it can be seen that the whole spectral range of interest occupies about 90 pixels, from the 250th pixel to the 340th pixel on the CCD camera. The number of pixels between peak maxima can also be read: approximately 17 pixels between 540 nm and 560 nm, 13 pixels between 560 nm and 580 nm, 18 between 580 nm and 610 nm. So, it should not be difficult to distinguish these four dyes through binning the pixels around the maximum emission of each dye.

The peak width of a separation peak of DNA fragments in capillary gel electrophoresis is about 10 s, so a minimum data acquisition speed of 1 Hz is required to give approximately 10 data points per peak. A DNA sequencing run often lasts about 2.5 h. If the whole spectra is saved as it is without binning, as in Figure 3.20, there will be 90 data points for each capillary in a single frame (one exposure). If the data acquisition speed is 1 Hz (1 frame per second), there will be 90 points x 1 Hz x 2.5 h x 3600 s/h = 810,000 data points for one capillary in a complete run; that is 25,920,000 data points for all 32 capillaries. If each data point is saved as 2-byte (16-bit), then a DNA run for the system will occupy 51.84 MB of the computer memory. This data size is too large for the data transfer system to handle. Binning is the only way to reduce the data size at the expense of spectral resolution. Our in-house software bins each spectrum into 10 data points and saves them in an array. So data size is reduced to 10 x 1 Hz x 2.5 h x 3600 s/h x 2 byte x 32 = 5.76 MB for a complete DNA run.

Because of binning, the spectral resolution will not be as good as shown in Figure 3.20. But we should still be able to distinguish the 4 dyes by finding a particular channel to represent a specific dye. This can be done by searching data in the 10 channels to find an appropriate channel where one specific dye has a relatively higher intensity than the others. More details will be discussed in Chapters 4 and 5.

Another interesting phenomenon was observed in this experiment. At wavelengths longer than 610 nm, there is an additional broad band. To identify this broad band, a background spectrum (Figure 3.20 e) was measured without injecting any dye into the capillary. The circle on the background spectrum (Figure 3.20 e) labels the location of the

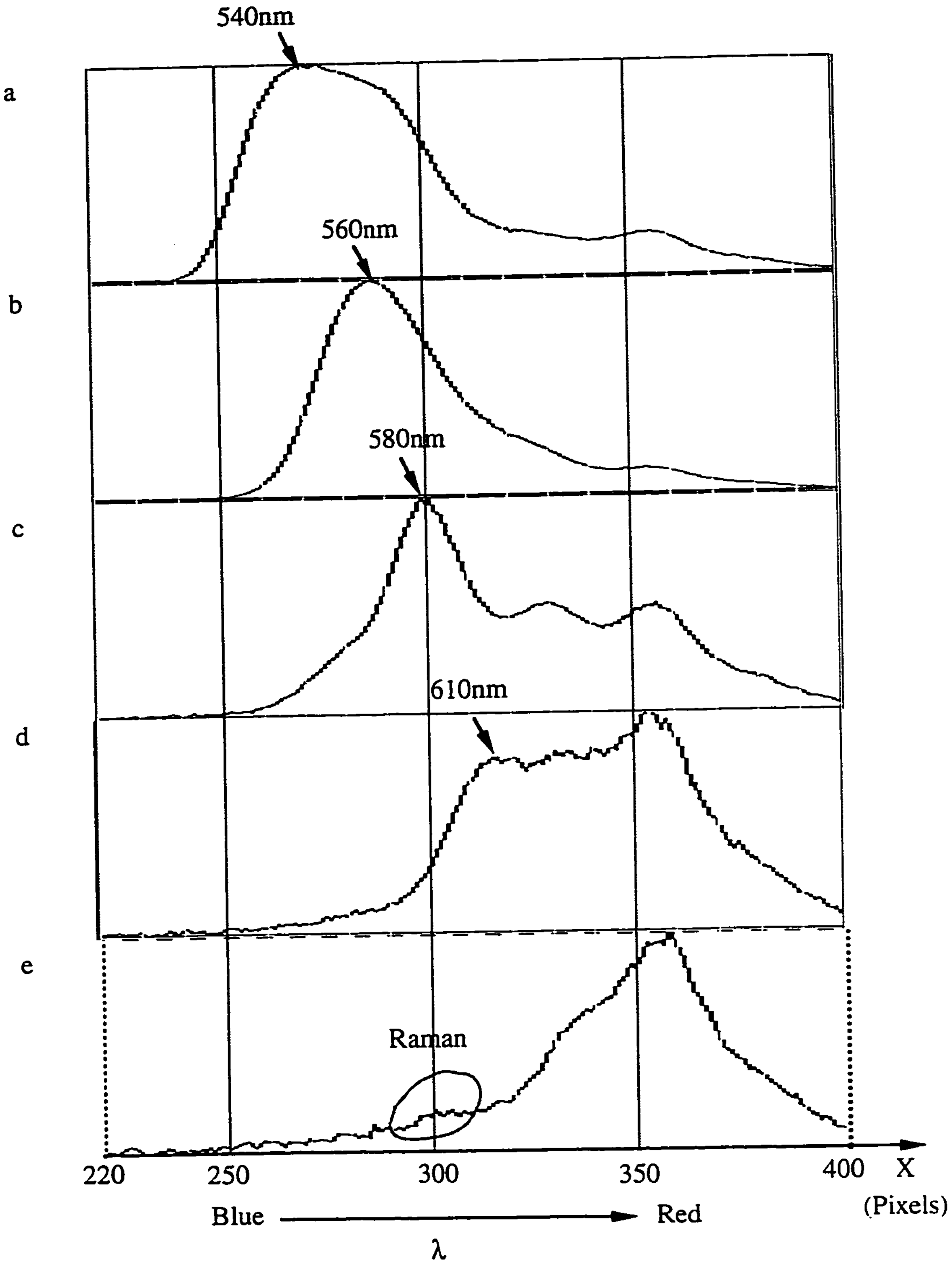


Figure 3.20 Spectra of a) Fam, b) Joe, c) Tamra, d) Rox, e) background
excitation: 488 nm detection: LP515
prism incident angle ~55°

water Raman band at around 580 nm induced by the 488 nm laser line. The broad band in the far red region must be the fluorescence of the cuvette glass.

As described in Section 3.2.2, after the laser travels through the sheath-flow region, it is partly reflected back at an angle slightly different from the incident angle and shines onto the cuvette glass due to the imperfectly polished edge of the cover slip. The reflected portion of the laser light is weaker than the input laser line, but strong enough to induce fluorescence from the glass, producing the broad band of the background spectra in Figure 3.20 e. The explanation is confirmed by moving the cuvette in and out relative to the laser line. When the laser is misaligned with the capillaries, the input laser beam is deliberately shone onto the cuvette glass, and the intensity of the background spectra peak increases by about 30%.

Fortunately, the fluorescence band of glass is not in the spectral range of interest, since it starts to appear after emission maximum of Rox. The intensity of the background fluorescence is not very strong. If Rox is efficiently excited with the 515 nm green line, there is not much interference at all. Also, the interference can be excluded by the use of a bandpass filter or software binning which will be discussed in Chapter 4.

Final Version of Dispersing Optics

Experimental results in this section coincide with the calculated results from theory and the ray tracing results in Section 3.4, confirming that with the F2 equilateral prism and camera lenses, the minimum deviation condition can be used to build the spectrometer.

The schematic of the final optical arrangement is shown in Figure 3.21. A cutoff filter is placed after the cuvette to block the Rayleigh scatter of the laser lines. The first camera lens collects the fluorescent light from all 32 capillaries, and collimates it. The collimated light strikes on the first surface of the prism at an incident angle of about 55°, and color-dispersed light comes out of the second surface of the prism in such a manner that the light passage is approximately symmetric. Then, the second camera lens images

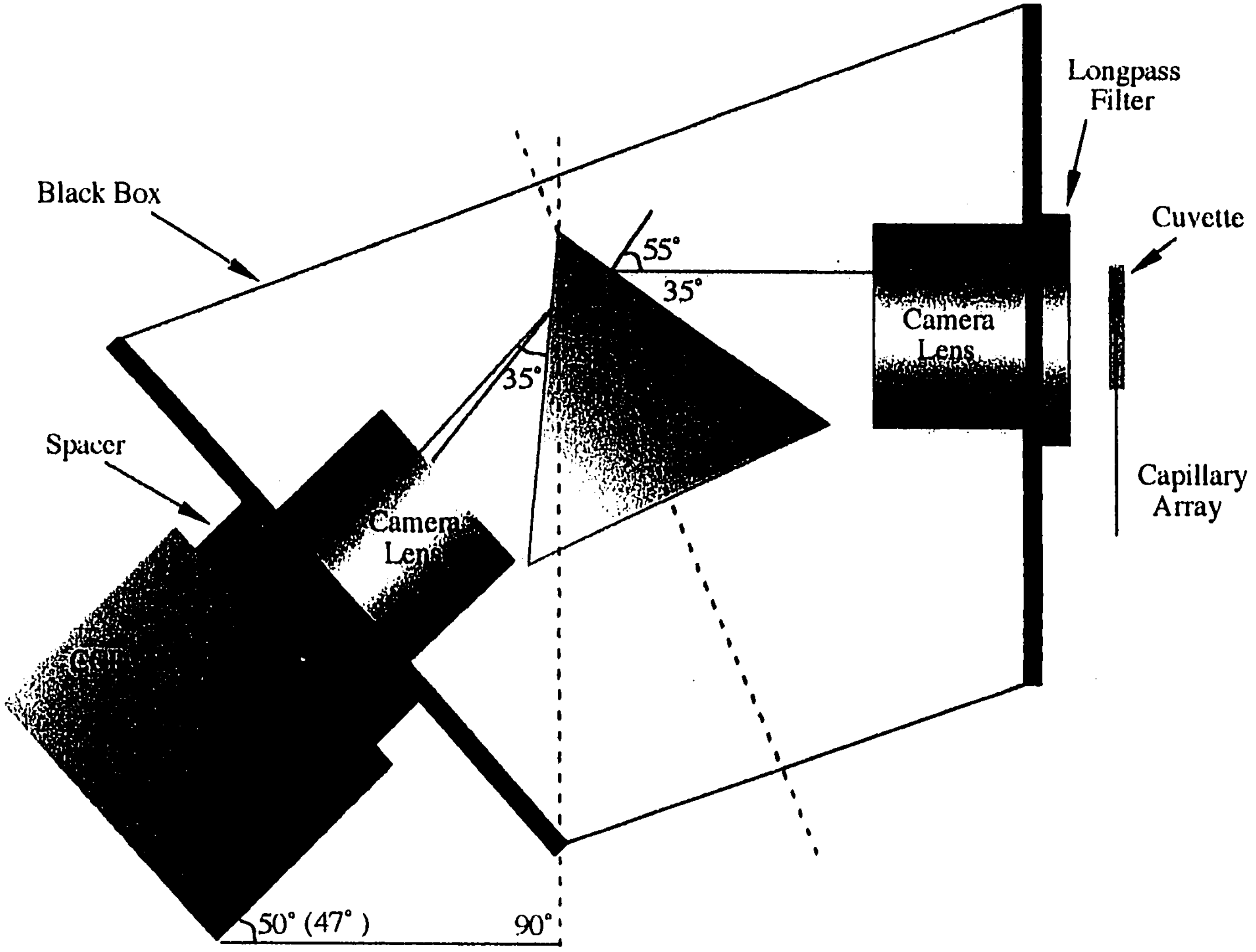


Figure 3.21 Schematic diagram of the final optical arrangement

the spectra of all 32 capillaries onto a CCD chip. This design of the dispersing optics has been successfully applied to our next generation instrument which has a 2-dimensional separation system.

At the left bottom corner of Figure 3.21, the angle formed by the stand of the CCD camera and horizontal axis is 50° in an ideal symmetric situation. But the best focus actually occurs at 47°, suggesting that the light passage is not perfectly symmetric, possibly because the actual optical condition is not exactly the one of minimum deviation.

As mentioned in Section 3.3.5, the intensity of light has been converted into a digital signal by the built-in electronics system of the CCD camera, so the data acquisition is free of extra electronic development. A well designed data acquisition software is essential to read out each frame of data and save it in the computer. Software design related to data acquisition and data analysis is discussed in the next chapter.

REFERENCES

1. R. J. Zagursky, R.M. McCormick, BioTechniques 1990, 9, 74-79
2. X. C. Huang, M.A. Quesada, and R.A. Mathies, Anal. Chem. 1992, 64, 2149-2154
3. X. C. Huang, M.A. Quesada, and R.A. Mathies, Anal. Chem. 1992, 64, 967-972
4. J. A. Luckey, H. Drossman, A. J. Kostichka, D. A. Mead, J. D'Cunha, T. B. Norris, and L. M. Smith, Nucleic Acids Res., 1990, 18, 4417-4421
5. H. Swerdlow, J. Z., Zhang, D. Y. Chen, H. R. Harke, R. Grey, S Wu, and N. J. Dovichi, Anal. Chem. 1991, 63, 2835-2841
6. N. J. Dovichi, J. Z. Zhang, US Patent 5,439,578, 8 August 1995
7. N. J. Dovichi, J. Z. Zhang, US Patent 5,584,982, 17 December 1996
8. A. E. Karger, J. M. Harris and R. F. Gestland, Nucleic Acids Res., 1991, 19, 4955-4962

9. A. J. Kostichka, M. L. Marchbanks, R. L. Brumley, Jr., H. Dossman and L. M. Smith, Bio/Technology 1992, 10, 78-81

10. H. Kambara, K. Nagai, and S. Hayasaka, Bio/Technology 1991, 9, 648-651

11. S. Takahashi, K. Murakami, T. Anazawa, and H. Kambara, Anal. Chem. 1994, 66, 1021-1026

12. N. J. Dovichi, Electrophoresis 1997, 18, 2748-2754

13. J. Z. Zhang, PhD thesis, Department of Chemistry, University of Alberta, DNA Sequencing by Single and Multiple CGEs, 1994, a) p113, b)p68

14. J. Z. Zhang, Y. Fang, J. Y. Hou, H. Ren, R. Jiang, P. Roos, N. J. Dovichi. Anal. Chem. 1995, 67, 4589-4593

15. K. Ueno, E. S. Yeung, Anal. Chem. 1994, 66, 1424-1431

16. S. Carson, A. S. Cohen, A. Belenkii, M. C. Ruiz-Martinez, J. Berka, and B. L. Karger, Anal. Chem. 1993, 65, 3219-3226

17. M. A. Quesada, S. Zhang, Electrophoresis 1996, 17, 1841-1851

18. J. Crabtree, PhD thesis, Department of Chemistry, University of Alberta, a high throughput multicapillary DNA sequencer, 1997. a) p99, b)198, c)227

19. Y. F. Cheng, S. Wu, D. Y. Chen and N. J. Dovichi, Anal. Chem. 1990, 62, 496

20. J. D. Ingle, S.R. Crouch, Spectrochemical Analysis, P45

21. M. Laikin, Lens Design, Marcel Dekker, Inc., New York, 1991, p68-69

22. J. Z. Zhang, N. J. Dovichi, US Patent 5,567,294, 22 October 1996

23. W. J. Smith, Modern Optical Engineering, The Design of Optical Systems, McGraw-Hill, Inc., 1996, p75

24. L. Levi, Applied Optics, A Guide to Optical System Design/Volume 1, John Wiley & Sons, Inc., New York, 1968, P365

25. E.J. Meehan, Optical Methods of Analysis, Interscience Publishers, New York, 1964, p2818

26. Applied Biosystems, Inc., Model 370 product literature

Chapter 4. Software Design for 32-Capillary Spectrometer

4.1 DATA ACQUISITION SOFTWARE FOR CCD CAMERA

4.1.1 Introduction to C Programming

A computer performs a programmed procedure fast and automatically. The purpose of software with a user interface is to accept the user's instructions and to have the computer perform a sequence of procedures with accurate timing and high speed.

A program (source code) is written by a programmer and consists of a sequence of instructions for the computer. Programs are usually written in high-level languages with C language being a popular one. Machine language (object code) is the only language that computers understand directly. Machine languages generally consist of strings of numbers (ultimately reduced to 1s and 0s) that instruct computers to perform their most elementary operations one at a time.[1, 2]

Compilers are the translator programs that convert high-level language programs into machine language in order to let the computer understand the instructions written by the programmer. There are various C compilers for different operating systems. The computer used for all the programming in this chapter is a Macintosh, and the C compiler is Symantec C++ in the Think C development environment (Symantec Corporation, Cupertino, CA, USA).

A function or procedure is an element of a program, and is designed to accomplish a specific task. There are many pre-written functions or procedures available for use; a collection of these existing functions is called a library. Libraries involved in the data acquisition software for the CCD camera are the C standard library,[1, 2] a Macintosh operation library (Mac library),[3] and a PVCAM library.[4] The last one is an ANSI C library created by Photometrics, Inc. (Tucson, AZ, USA) to abstract and simplify the interface for the CCD camera; using this library allows the programmer to have access to the full power of a Photometrics CCD camera, without needing to worry about details of hardware interface.[4]

A complex program typically consists of many functions. A C program has to have a main function, and it can have as many function prototypes which are programmer self-defined functions.[2] All the function prototypes have to be defined in the header file of the program. The data acquisition programs for CCD can be divided into three parts: resource files, header files, and function files. (See Appendix I.)

The data acquisition (DAQ) software named "Cam" is written in C, using the C library PVCAM provided by Photometrics to access the Photometrics camera system. All the user interface objects, such as menus, windows, buttons, icons, dialog boxes, etc., are created in resource files in ResEdit (version 2.1, Apple Computer Inc., Reading, MA, USA) and used in the programs through Macintosh procedures.[3]

The source code is listed in Part I of Appendix I. Here only the design and the function of the software are discussed.

4.1.2 Format of the Spectral Data from 32 Capillaries

Original Data

The DAQ software should be able to control the CCD camera, transfer data from the CCD and write data to a computer hard disk efficiently. As discussed in Section 3.5.2, the spectrum of each capillary has to be binned to save memory and speed up the data acquisition. There are two types of binning: hardware binning and software binning. Hardware binning takes place at the camera's readout node, where the data in corresponding pixels are added together before being read out to the computer as one data point. Software binning is a data manipulating process after all the image has been transferred out to the computer. It is not controlled by the CCD hardware.

Here, both binning methods are employed to reduce the data size. Each spectrum is divided into 10 portions and the data from each portion is binned into 1 data point, so that each spectrum is represented as 10 data points or 10 bands of wavelengths; in other words, each spectrum is saved in the form of a 10-element array with each element of the array representing the light intensity for one band of wavelengths. Data from one frame (one exposure or shot) contains the data from all 32 capillaries. Because each capillary has 10 data points per frame, one frame of data is 320 data points. The original data is arranged to save on disk as binary integers in the following way:

As shown in Figure 4.1, the first 10 data points (0 to 9) is the spectrum data for capillary #1; the second 10 data points (10 to 19) for capillary #2; ...; the 32nd 10 data points (310 to 319) for capillary #32. The first 320 data points (0 to 319) compose the first frame. The first frame of data is written to disk, followed by the second frame of data, then the third frame, and so on until the final frame. The data type of each data point is an unsigned 32-bit integer, which requires 4 bytes. So, each capillary has 40 bytes of data per frame, and the data size of each frame is 1.28 kB (40 bytes x 32).

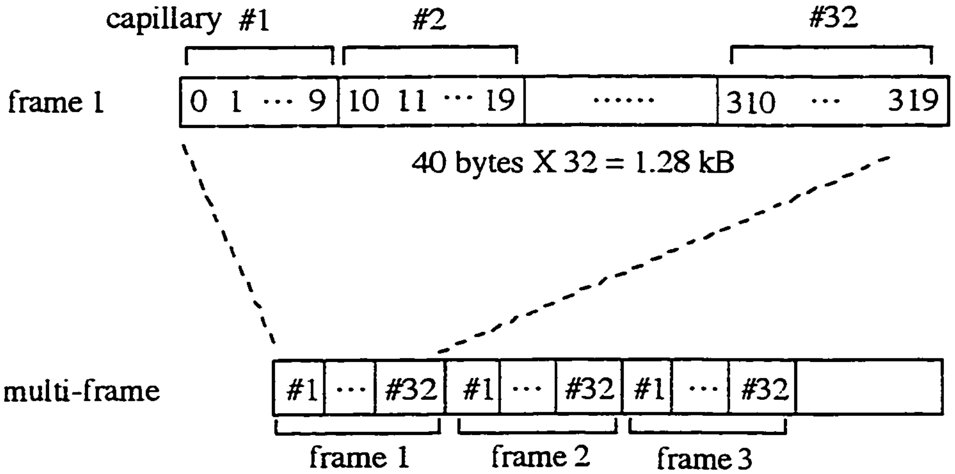


Figure 4.1 The format of the spectral data saved in memory

As we can see from Figure 4.1, the original data from all the capillaries is interleaved. If the original data is directly loaded into Igor Pro for analysis, there will be 320 waves, and every 10 waves belongs to one capillary. For instance, the 10 waves from wave 0 to wave 9 make up of the spectrum for capillary #1; the 10 waves from 230 to 239 forms the spectrum of capillary #24, and so on. The original data file is typically big. For example, the data size for a 10 minute run with a 1.5 Hz frame rate will be: 1.28 kB/frame x 600 s x 1.5 frames/s, or 1.152 MB. The big file and the large number of waves slow down the data analysis process, and a large amount of RAM is required to run the Igor Pro program for data analysis.

Split Data

The original data format described above is the easiest way to save data in the DAQ process, but it is not convenient for data analysis. If we split the original file into 32 individual files, each of which contains the data from one capillary, then the data size of one individual file for a 10 minute run at 1.5 Hz is reduced to only 36 kB. After the data is loaded in Igor Pro there are only 10 waves, each wave representing one band of wavelengths of the spectrum. To achieve this, a C program called "Cut32" was written to re-write the data of each capillary separately onto disk. The source code of "Cut32" is listed in part II of Appendix I.

One way to re-organize the original data is to open a new file, read out the data for one capillary and write it into a new file. After finishing this for one capillary, open another new file, read out and write data for another capillary. This is done 32 times, once for each capillary. According to Figure 4.1, in the original data file, the starting data point in each frame of data for different capillaries can be calculated as:

$$(f - 1)\ 1280 + 40\ (n - 1) \text{ bytes} \qquad (4\text{-}1)$$

In equation (4-1), *n* is the capillary number, *f* is the frame number. Following each starting data point, there are 40 bytes of data belonging to the same capillary, For example, for capillary #2 (n = 2), the first frame data (f = 1) starts at the 40th byte, the second frame data will start at the 1320th byte. To read out the data of capillary #2 and write it into an individual file, we can set the read marker first at the 40th byte and read out the following 40 bytes of data and write it to disk. We then move the marker 1280 bytes away from the first starting point to the 1320th byte, and read the second 40 bytes and write it to disk. This is repeated until we reach the end of the original file.

There is an alternative way to do the same job. Create 32 new files, read 1280*i (i is an integer) bytes of data each time, then write the first 40 bytes into file1 for capillary #1; the second 40 bytes to file2 for capillary #2; and so on. After finishing 32 cycles (writing each 1280 bytes), the cycle starts over again to write file1. The read-write cycle continues until the end of the original file is reached.

The second method runs faster than the first one, because the computer must read through the whole original file only once and not 32 times as in the first method. The "Cut32" program uses the second method, creating 32 different files within one folder, reading the original data in, and rewriting the data from each capillary into an individual file.

4.1.3 Multiple Exposure Regions

To use disk space efficiently, we want to write only the actual data and discard the useless data in those regions in between the spectral bands. There are two ways to achieve this:

One is to use the whole CCD chip or part of it as a single region for exposure. Take a frame of data, let the user read out the region for each spectral band and input the obtained axis for software binning. A program can be designed to write only the data from those pixels where the spectral bands are located, according to the region that the user inputs. But the process for the user to locate the accurate location of each spectral band by reading pixels on screen can be tedious.

The other is to take advantage of multiple-region exposure, where the CCD chip is divided into different regions and only those selected regions are treated as active by the camera hardware. In this case, the whole CCD chip is exposed to the light, but only the designated exposure regions are read out by the CCD hardware. A program can be designed to let the user define the multiple exposure regions easily. The user can select the spectra covered regions by drawing rectangles on screen and these rectangles can be passed to the CCD camera as individual exposure regions.

The second method takes advantage of the effective readout. Only data in the regions covered by the spectra thus selected by the user are recorded; data in those regions not occupied by the spectra is not readout and transferred to the computer and thus not saved. The data transfer time decreases compared with the first method, because in the first case the useless data in those regions not covered by spectra has to be read out and transferred from the CCD camera to the computer as the useful data. Also, defining useful regions by drawing rectangles is direct and easy to do compared with reading the regions pixel by pixel on screen.

The "Cam" uses the multiple regions exposure method. Typically, about 1.5 Hz DAQ speed can be achieved with an exposure time of 400 ms, and about 2.2 Hz with a 200 ms exposure time.

4.1.4 User Interface

There are two user interface windows in this software, an alignment panel and a data display panel. The alignment panel (Figure 4.2 - 4.4) has two functions: to display and update the image of the spectra, and to allow the user to edit the multiple exposure regions. The data display panel (Figure 4.5) displays one spectral channel of the data from one capillary.

Alignment Panel

Figure 4.2 shows the alignment panel. The large rectangle at the top is the image view window. It displays the image of the region on CCD chip from 0 to 1100 in the serial direction and from 0 to 300 in the parallel direction, where the image of spectral bands is located. Below the display window, there are two rows of icons with numeric labels on them. There are 32 icons in total, each representing one capillary.

The exposure time used for the image collection is 200 ms, and the image seen on screen is updated after every exposure without being written to disk. This view window provides a coarse reference for the user to align the optical system by looking at the brightness of all the spectral bands.

In Figure 4.2, half of the 32 capillaries are continuously injected with fluorescein, and another half are injected with R6G. So two sets of fluorescent bands can be seen in the view window. The images of fluorescent bands are shown horizontally, not vertically as in the figures of Chapter 3. Actually, the capillary array is arranged vertically and dispersion is arranged horizontally as in Chapter 3. Here, the CCD camera has been rotated 90° to make the display fit the screen better. This is just a matter of computer display, the optical system used is otherwise the same as shown in Figure 3.15 and Figure 3.21.

As shown in Figure 4.2, in the image view window we can see both edges of the cuvette. The edge on the right has strong light scattering: that is where the laser light enters the cuvette. The left edge through which the laser beam leaves the cuvette has much less scattering. Between the two edges, there are the spectra dispersed by the prism.

The polyimide coating outside each capillary fluoresces due to the illumination of the scattered laser beam, since the laser beam shines through the solution less than 100 μm beneath the capillary tips. In the experiment shown in Figure 4.2, the top part of the cuvette where the capillary array is located is covered by black tape to block the fluorescence and scattering from the capillaries themselves. Above the individual spectral bands we can still see a relatively wide band of light all the way across the whole cuvette.

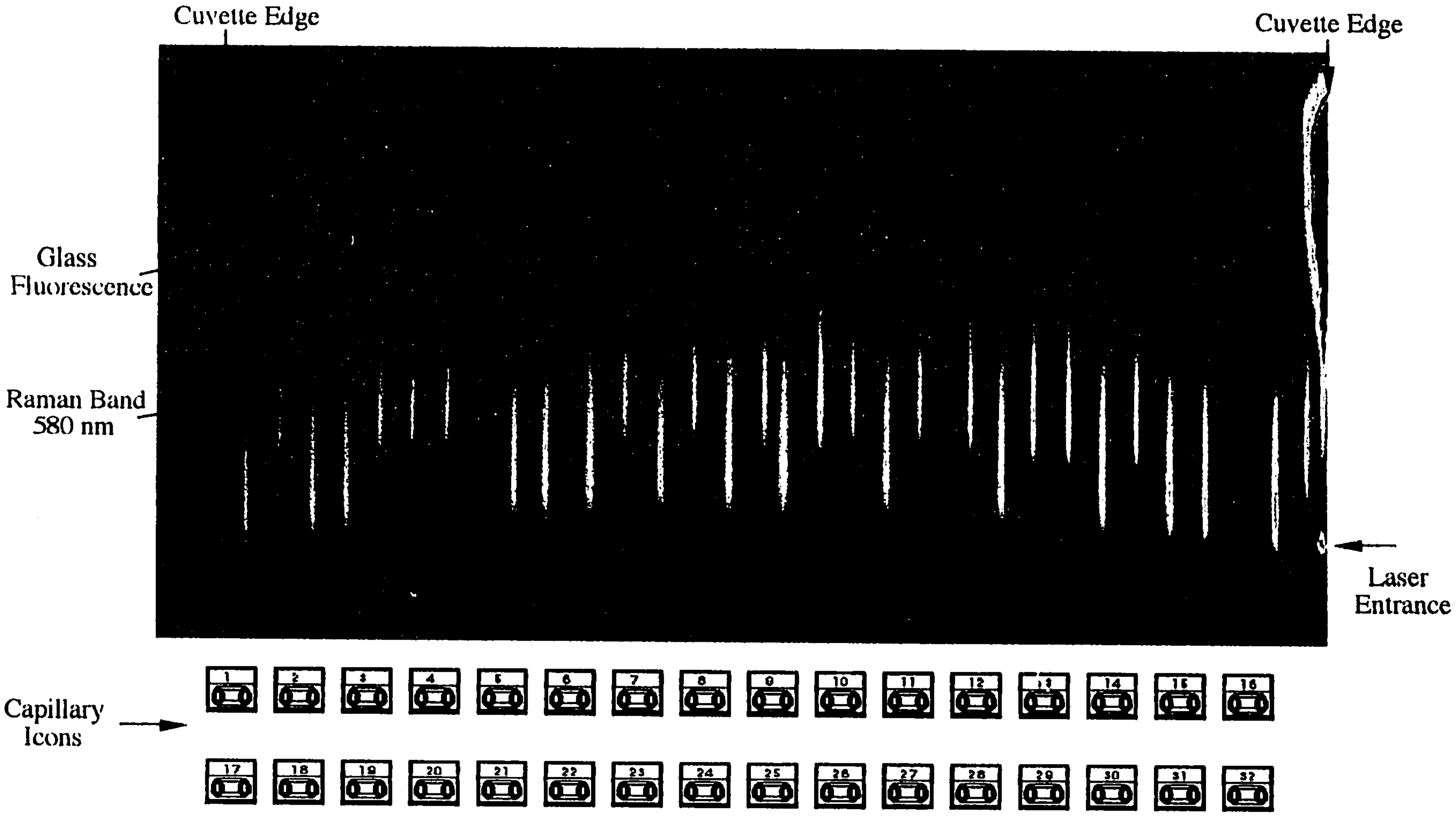


Figure 4.2 Alignment panel with 10^{-7} M dye solution in 10 mM borate
dye: fluorescein for capillaries #1,3,4,9-11,13,15,17,20,23,26,28-31
R6G for the other 16 capillaries
detection: LP515, excitation: 488 nm

The location of laser entrance is the spectral location of 488 nm; the relative narrow band in the middle of the fluorescent bands is the Raman scattering band around 580 nm from the laser beam; the wide band above the individual spectrum in the far red region is the fluorescence of cuvette glass, which is the background peak we have seen in Figure 3.20.

If the top part of the cuvette where the capillary array is located is not covered by black tape, images of the capillaries will be seen. This is shown in Figure 4.3 and Figure 4.4, in which all 32 capillaries were continuously injected with fluorescein.

The alignment panel allows the user to edit each exposure region by drawing a rectangle to fit each spectral band, and the software is designed to remember the regions from the last time when the "Cam" was running. To edit the exposure regions, the user presses the command key () and the period key (.) at the same time. The image will then stop updating and the last image before the keydown event will be displayed on the view window. Rectangles from the last time when "Cam" was running will be automatically loaded into memory and displayed.

As shown in Figure 4.4, to edit the exposure region of capillary #26, the user clicks on the capillary icon labeled 26 first, the icon will turn red telling the user that capillary #26 is ready for editing. The user would then start to draw a rectangle which fits the outline of the spectral band. After editing the exposure region for this capillary, the user can click on the capillary icon for the next capillary to be edited, and the icon of #26 will turn black at the same time as the corresponding icon turns red, and the user may edit the designed capillary region. This is done until the user is satisfied with all the exposure regions. Then by simultaneously pressing command () and 'Q', the edited rectangles will be saved for future use and the program goes to the next panel, the data display panel.

Data Display Panel

Figure 4.5 shows the data display panel. This part of the program is designed to acquire the data in all 10 spectral channels from all 32 capillaries and save all the data to

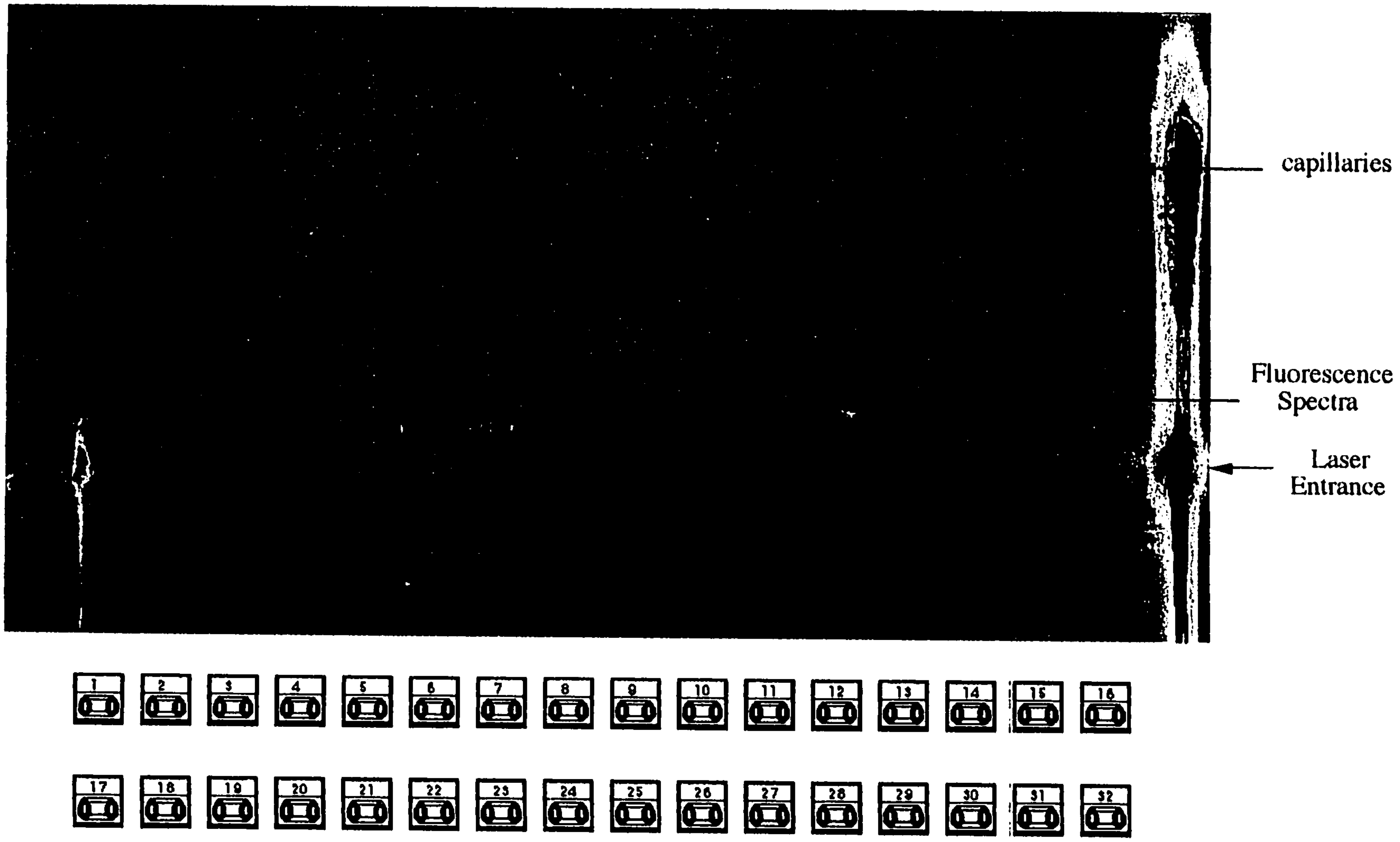


Figure 4.3 Alignment panel with visible capillary images
dye: 10^{-8} M fluorescein solution in 5 mM SDS 10 mM borate
excitation: 488 nm detection: LP515

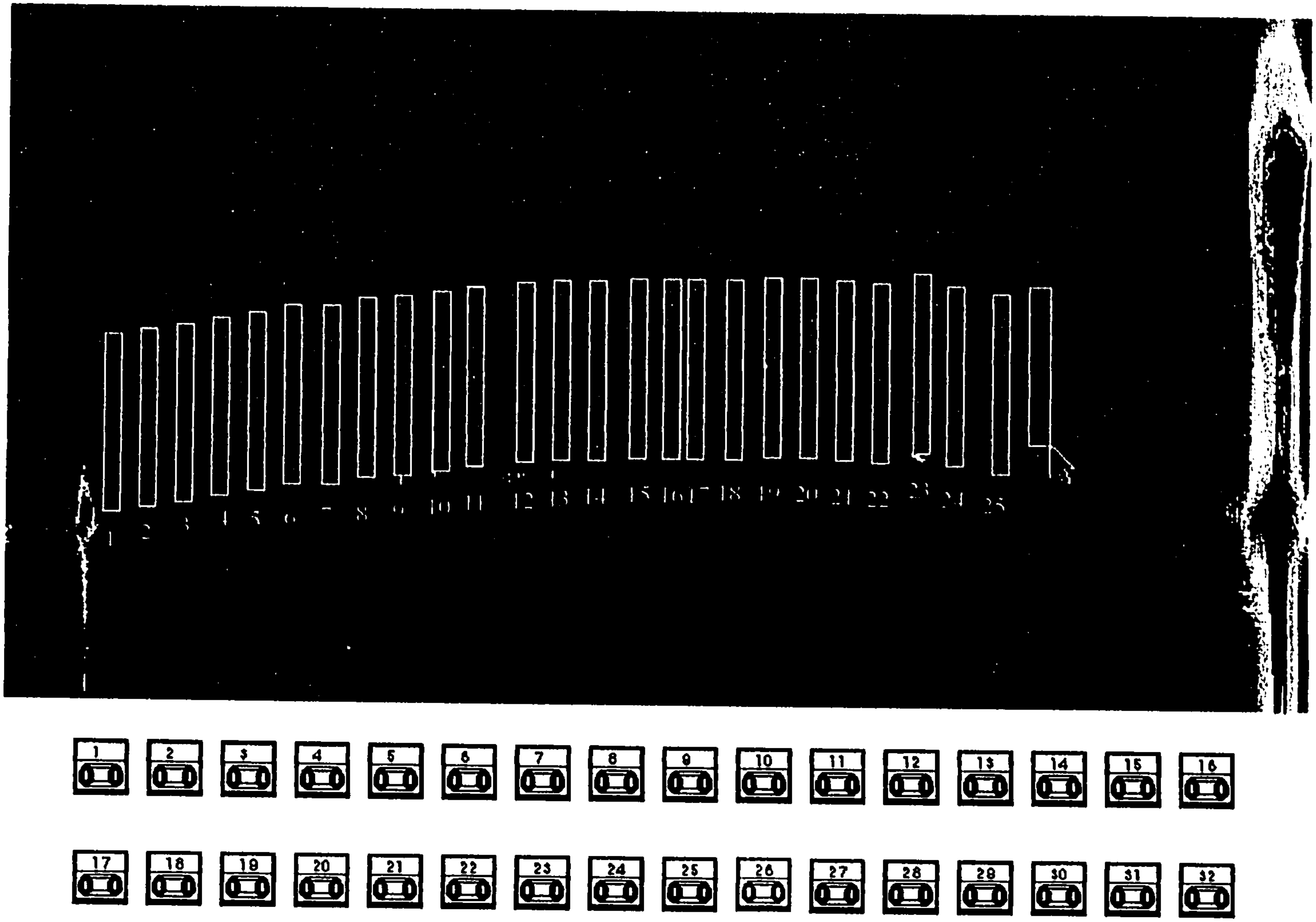


Figure 4.4 Selection of multiple exposure regions
dye: fluorescein 10^{-8} M in 5 mM SDS 10 mM borate
excitation: 488 nm detection:LP515

disk while the user monitors the fifth spectral channel from one capillary at a time. Again, there are 32 capillary icons at the bottom of the display window representing the 32 capillaries. Four buttons on the right top of the window are the controls for baseline up and down, data zoom in and out.

The user can select any capillary to monitor by clicking on the corresponding capillary icon, and the selected icon turns red. In Figure 4.5, capillary #26 is being monitored. If the user switches to another capillary, the newly selected capillary icon turns red and the previous one turns black. The data display continues until the lines reach the right edge of the display window, when the old trace is erased and data display continues from the left edge of the window.

The user can also enlarge or shrink the data display scale, by clicking on the button labeled Z (zoom) or S (shrink), each click will change the display scale by a factor of two; and the baseline can be shifted up or down by clicking the button labeled Δ (up) or ∇ (down), each click shifting the baseline by 50 pixels on the screen.

In the left top corner above the display window, there are two menus which allow the user to quit the "Cam" application from the "File" menu or to change the exposure time from the "Exposure(ms)" menu. "Cam" also recognizes '' 'Q' as a command to quit the application.

4.2 DATA ACQUISITION PROGRAM IN LABVIEW FOR PMT

4.2.1 Introduction to LabVIEW Programming

LabVIEW is a program development application, like C. However, LabVIEW is different from C in one important respect. C uses a text-based language to create programs as lines of code, while LabVIEW uses a graphical programming language, G, to create programs in a block diagram form.[5, 6]

LabVIEW is a general-purpose programming system, but it also includes libraries of functions and development tools designed specifically for data acquisition and

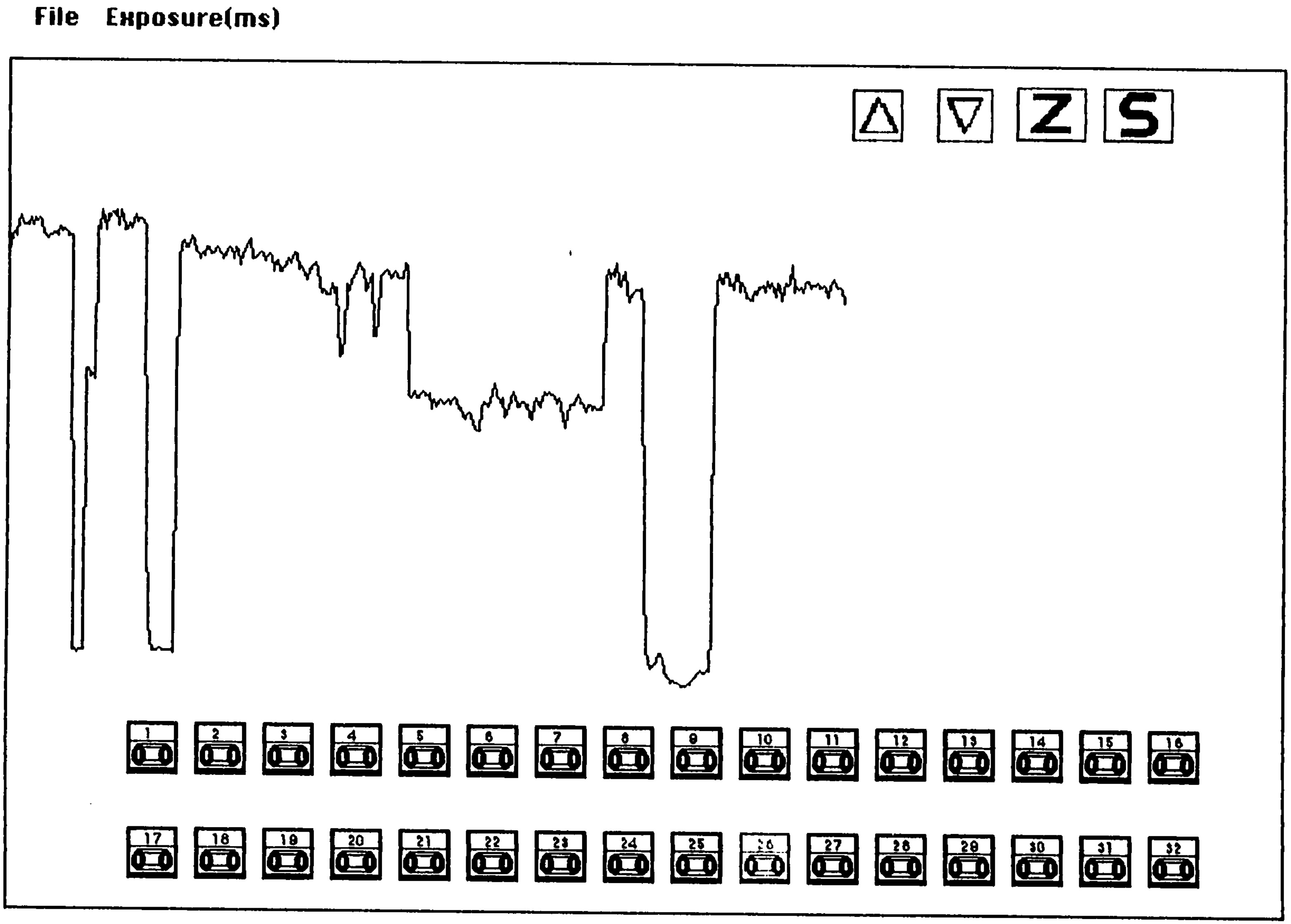


Figure 4.5 Data display panel

instrument control. LabVIEW programs are called virtual instruments (VIs), they are identical to functions from conventional language program. VIs have an interactive user interface, a source code equivalent, and accept parameters from higher level VIs.[6]

The interactive user interface of a VI is called the front panel, where users can enter data using the mouse and keyboard and then view the results on the screen. The VI receives instruction from a block diagram, which the programmer constructs in G by "wiring" together the objects that send or receive data, perform specific functions, and control the flow of execution. The block diagram is the graphical source code for the VI.

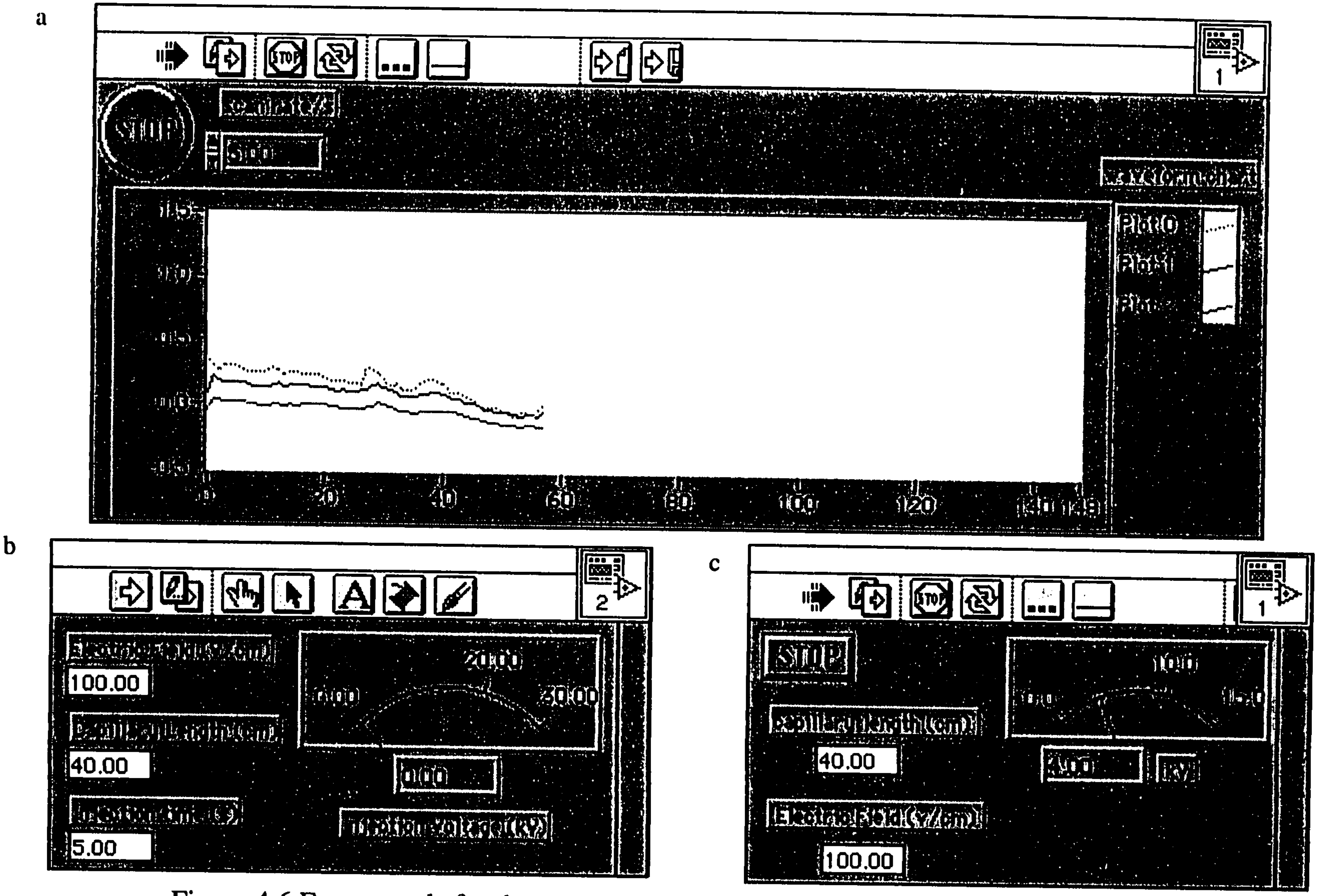
A control flow method is used in the execution of a conventional language (such as C) program. In that program, instructions are executed in the sequence in which they are written in the program, except when specific commands (such as "if" or "while loop") move the point of execution to another location in the program. Graphic orientated data flow, rather than flow control, is the principle that governs LabVIEW program execution. Stated simply, a node executes only when data arrives at all its input terminals; the node supplies data to all of its output terminals when it finishes executing; the data passes immediately from source to sink (destination) terminals of the node.

4.2.2 Data Acquisition Program

Data Acquisition and Display

As discussed in the next chapter, the light emission property of the cuvette has been tested with a PMT coupled to a 60X microscope objective. The data acquisition interface board used in this experiment is a NB-MIO-16XL from National Instruments (Austin, TX, USA). The interface board has 8 analog inputs and 2 analog outputs. The data acquisition program has to read data from the analog inputs, write it to disk, and display it.

Figure 4.6 shows the front panels for three applications, a) is the data display panel, which controls the data acquisition and displays the acquired data; b) is the injection panel, which controls the injection voltage and the injection time; c) is the high voltage



100.00

40.00

5.00

Capillary Length (cm)

40.00

Electric Field (v/cm)

100.00

Figure 4.6 Front panels for data acquisition in multi-channel application
a) data display panel, b) injection panel, c) high voltage panel

panel, which controls the running voltage during an electrophoretic run. These three applications provide single wavelength data collection, and automatic voltage control.

First, let's look at the data acquisition and display. The data display VI actually plays two roles: display the analog input data and write the data to disk. In the front panel (Figure 4.6 a), the user can enter the data collection speed (scan rate/s) which is 5 Hz in this particular application; the user can also stop the data collection by clicking on the "STOP" button in the left top corner. Data from three channels is displayed on the screen.

The block diagram of data display VI is shown in Figure 4.7 where there are 10 objects as noted in parenthesis. Once the user starts to run the application, object 1 opens a file for data writing, and object 2 sets up the data collection configuration. In this case (Figure 4.7), data from 5 channels (channel 0 to 4 on the DAQ interface board) are collected. Object 3 reads the user-entered scan rate and starts data acquisition. The big rectangular box in the center is a while loop. If the user does not click on the "STOP" button on the front panel (Figure 4.6 a), the while loop is 'true', and the program will keep executing the instructions inside the while loop. Once the user clicks on the "STOP" button, the condition of the while loop becomes 'false', the program then leaves the while loop and executes the command inside the box labeled "False", where object 8 stops the reading of analog input data and object 9 closes the file the data was being written into.

Data display and writing to disk are done in the while loop. Object 4 reads out the analog input data from the DAQ board and transfers input data to binary data for output. The output data goes to two places, one to object 5 for writing to disk, and the other to objects 6 and 7, which display the data in a waveform chart.

The programmer can selectively display one or more channels and one or more scans out of the total 5 scans per second. To do this, one needs to change the input values for object 6: "scan" (initial scan number), "Lscan" (length of scans), "ch" (initial channel number), and "Lch" (length of channels). As we can see from the input value of object 6 in

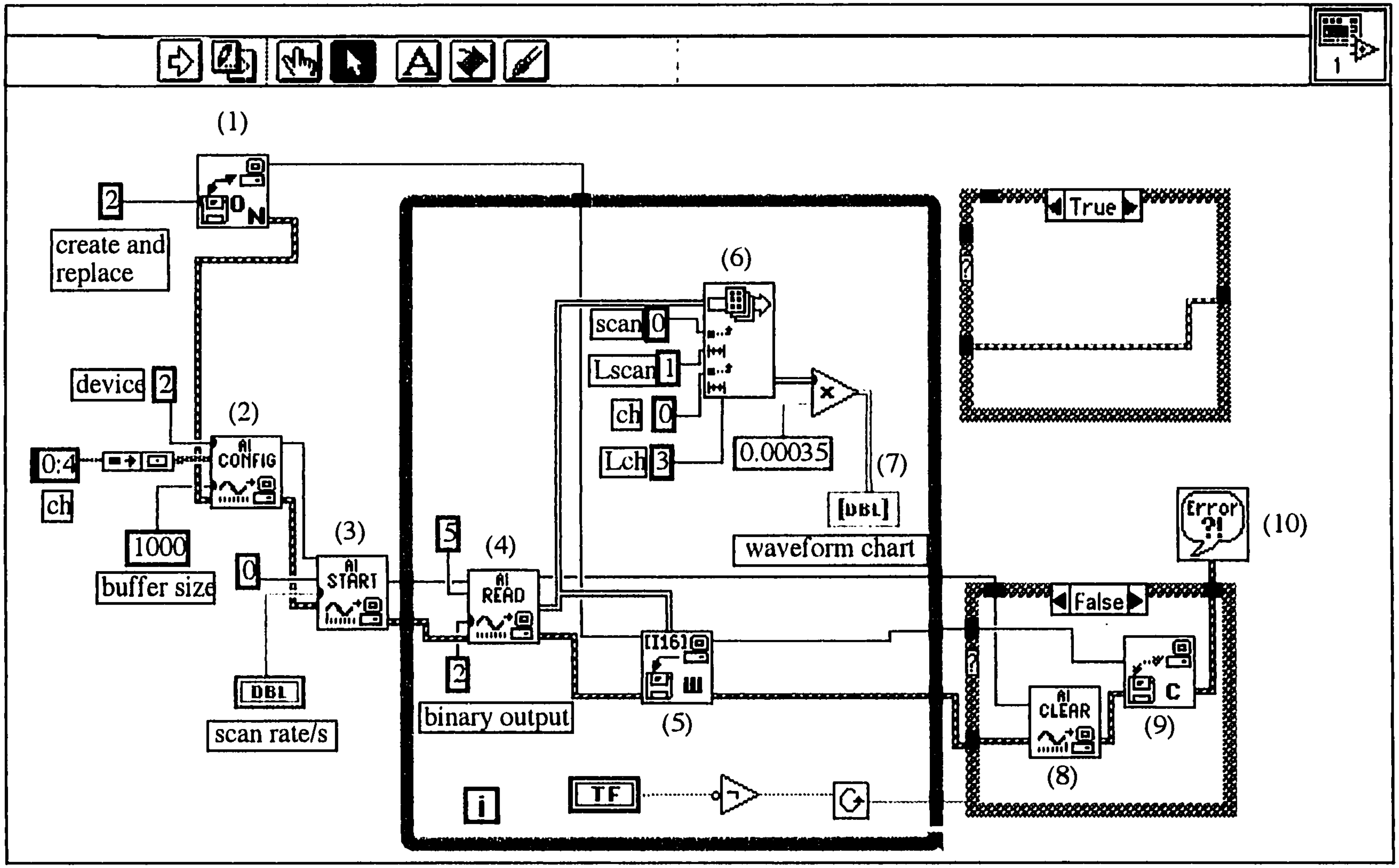


Figure 4.7 The block diagram of the data display panel in Figure 4.6 a

Figure 4.7, this VI is designed to display data of the first scan each second, and only the first three channels are displayed.

Voltage Control

The front panels of two voltage control VIs are shown in Figure 4.6 b and 4.6 c. The injection VI (Figure 4.6 b) allows the user to enter an electrical field, a capillary length, and an injection time. The applied voltage is displayed in the indicator. The purpose of an injection voltage control is to apply high voltage only during the defined injection period (for example, 5 s in Figure 4.6 b).

The 5 s time control is done by using a sequence shown in Figure 4.8. In step 0, the high voltage required by the user is supplied to the first channel (channel 0) of the analog output on the DAQ interface board (device 2); this analog output controls the high voltage power supply for the separation system. The value of the voltage is also displayed on the indicator of the front panel. Execution then goes to step 1, where the computer inner-clock starts to count for 5000 ms (5 s). Once the 5 s are over, 0 volts is sent to the high voltage power supply (in step 2) and the corresponding indicator as well.

Figure 4.6 c shows the front panel of a high voltage control VI. After the user enters the value for the capillary length and the electric field and runs the VI, the high voltage is applied. For instance, in the application shown in Figure 4.6 c, a 4 kV voltage is entered in by the user (40 cm x 100 V/cm) and this voltage is applied to the electrophoresis system. Once the user clicks on the "STOP" button in the left top corner of the high voltage panel, the voltage will go to zero instantly. This is done by using a while loop, as shown in Figure 4.9. When the user runs the program before clicking on the "STOP" button, the condition of the while loop is 'true', so the high voltage will be constantly supplied from the analog output channel 0. Once the user clicks on the "STOP" button, the condition of the while loop becomes 'false', the program leaves the while loop and executes the commands in the box labeled "False", and thus the high voltage is turned off.

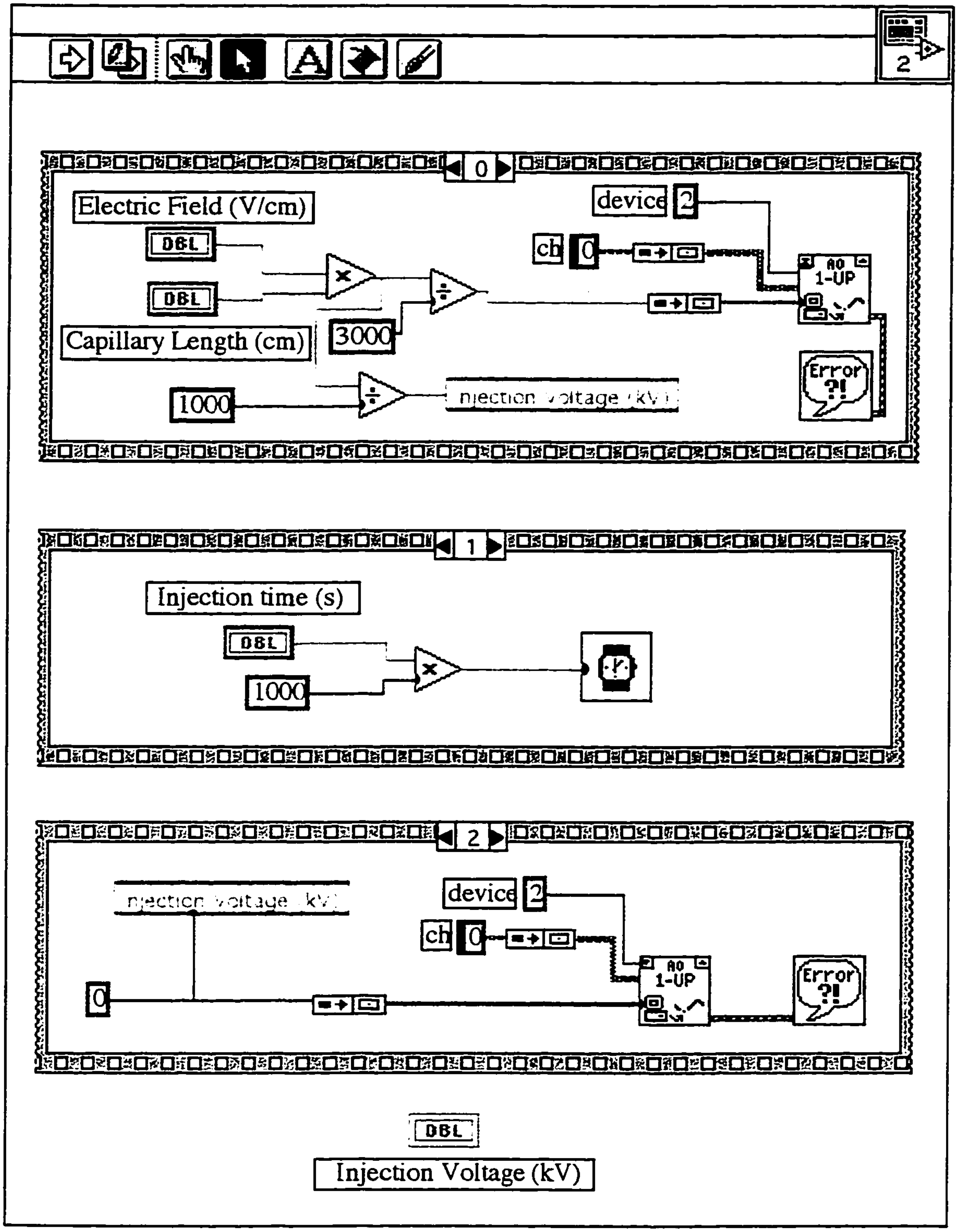


Figure 4.8 The block diagram of the injection panel in Figure 4.6 b

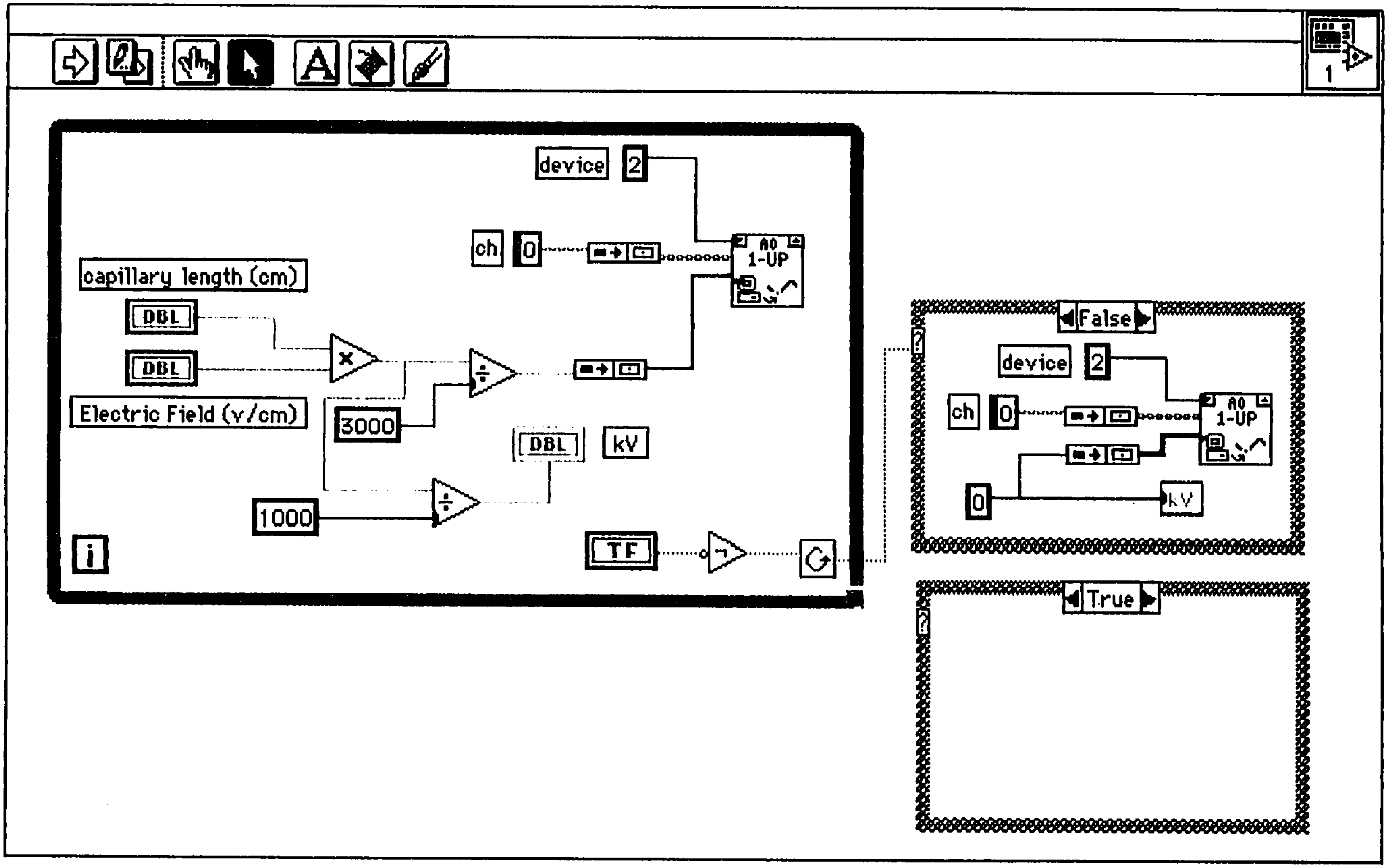


Figure 4.9 The block diagram of the high voltage panel in Figure 4.6 c

4.3 PROGRAMMING IN IGOR PRO FOR DATA ANALYSIS

4.3.1 Introduction to Igor Programming

High sample throughput is the main advantage of a multiple capillary instrument. The 32-capillary spectrometer analyzes 32 samples simultaneously, so 32 sets of data are obtained from one experiment. This high data throughput requires fast data analysis.

Igor Pro (WaveMetrics, Inc., Lake Oswego, Oregon, USA) is widely used for data analysis. Most often, it is used to display data in wave forms (line plot). Igor's user interface provides both the conventional Macintosh approach using "menus" and "dialog" boxes, and a parallel path using command line operations.[7] Commands can be executed by typing into the command line and pressing the return key. Alternatively, commands can be incorporated into procedures. There are two main types of procedures, "Macros" and "Functions". A programmer incorporates Igor commands into procedures in a specific window called "procedure window", then one can run the procedures via the command line or the Macro menu.

There are several differences between a Function and a Macro. A Function is pre-compiled so that it works like a single command and runs much faster than a Macro which has to be interpreted every time when it is called. There are also some limitations in a Function. For example, a Macro can use "prompt" to ask the user to enter variables. but a Function can not.

Igor's programming language is much like other conventional procedural languages such as C or Pascal. It is a text-based language, and uses flow control including conditionals and loops.

4.3.2 Procedures for Fast Data Analysis

The purpose of all the procedures here is to analyze data from multiple capillaries automatically. Macros can be combined together into new procedures to achieve different purposes. Here, only four examples are discussed. Source codes for the four procedures are listed in Appendix II.

Procedure 1: Layout of the Whole Electropherogram of Each Capillary

Sometimes it is necessary to look at the whole electropherogram from each capillary in one run, especially when 32 different samples are injected. This Macro is able to print out a layout of single-color data for all 32 capillaries, and it can be easily modified to display four-color data. In this Macro, the electropherogram from each capillary is divided into three sections and is placed in a one-page layout. So, if the user wishes to print out data from all 32 capillaries, there will be 32 pages, each page listing the whole electropherogram for one capillary.

To use the Macro, load the single color data for all 32 capillaries into Igor first. then call this Macro. In a DNA sequencing electropherogram, the first peak to appear is a large primer peak, then the narrow DNA sequencing peaks appear; the end of the run is indicated by a large end peak which is the signal from the co-migrated large DNA fragments. Those DNA sequencing peaks in between the primer peak and the end peak contain the data of interest.

Once a user runs the Macro, it prompts the user to enter "File" (file name), "DAQ_speed" (data collection speed in Hz), "t_primer" (migration time of the primer peak), and "t_end" (migration time of the end peak). Then a graph of the electropherogram will be printed out automatically with the file name and the capillary number labeled on it. The user can select which and how many capillaries to be printed out by changing the initial value of variable "m" and its terminal value in the condition brackets of the while loop.

Procedure 2: Layout of One Section of the Electropherogram for All 32 Capillaries

To compare the performance of each capillary (each separation channel), it is useful to compare one section of the electropherogram of all 32 capillaries. To do so manually is time-consuming and tedious; therefore, this procedure gives an example to have the computer do the repetitive job quickly. The procedure consists of one Macro and three Functions. It prints out a section of a single color electropherogram from all 32 capillaries in two selectable ways: one is 4 capillaries on each page with a layout format of 4 rows x 1 column, 8 pages in total for 32 capillaries; another is 16 capillaries on each page with a layout format of 4 x 4, 2 pages in total.

To run this procedure, one can load the data file into Igor first, then from the Macros menu, select "printout_32_1color", a window then appears to ask for several parameters: "DAQ_speed" (data collection speed in Hz), "Start_time" (start of the electropherogram in minutes to be displayed), "End_time" (end of the electropherogram in minutes), "Filename" (name of the file), and "caps_per_page" (how many capillaries to be displayed on one page; there are two choices: 4 or 16). After these parameters are entered, the selected section of the electropherogram of all 32 capillaries is printed out in either the 4 x 1 layout on 8 pages or the 4 x 4 layout on 2 pages. Figure 4.10 shows a typical example of a 4 x 4 layout of 16 capillaries produced by this procedure. The data analysis process is much easier and quicker in comparing the separation performance from capillary to capillary if this procedure is used.

Procedure 3: Calculation of Limit of Detection

Limit of Detection (LOD) is an important parameter for an analytical instrument. Typically, an experiment to measure a LOD takes approximately 10 minutes, but it can take several hours to calculate the LODs for all 32 capillaries manually. Writing a procedure to have the computer do the calculations makes the whole process much quicker and more accurate.

This procedure consists of four Macros, one for electropherogram display, two for LOD calculation, and another one for graph layout. To use the procedure, one can load the data file into Igor, use Macro "display8waves()" to display the electropherogram of 8

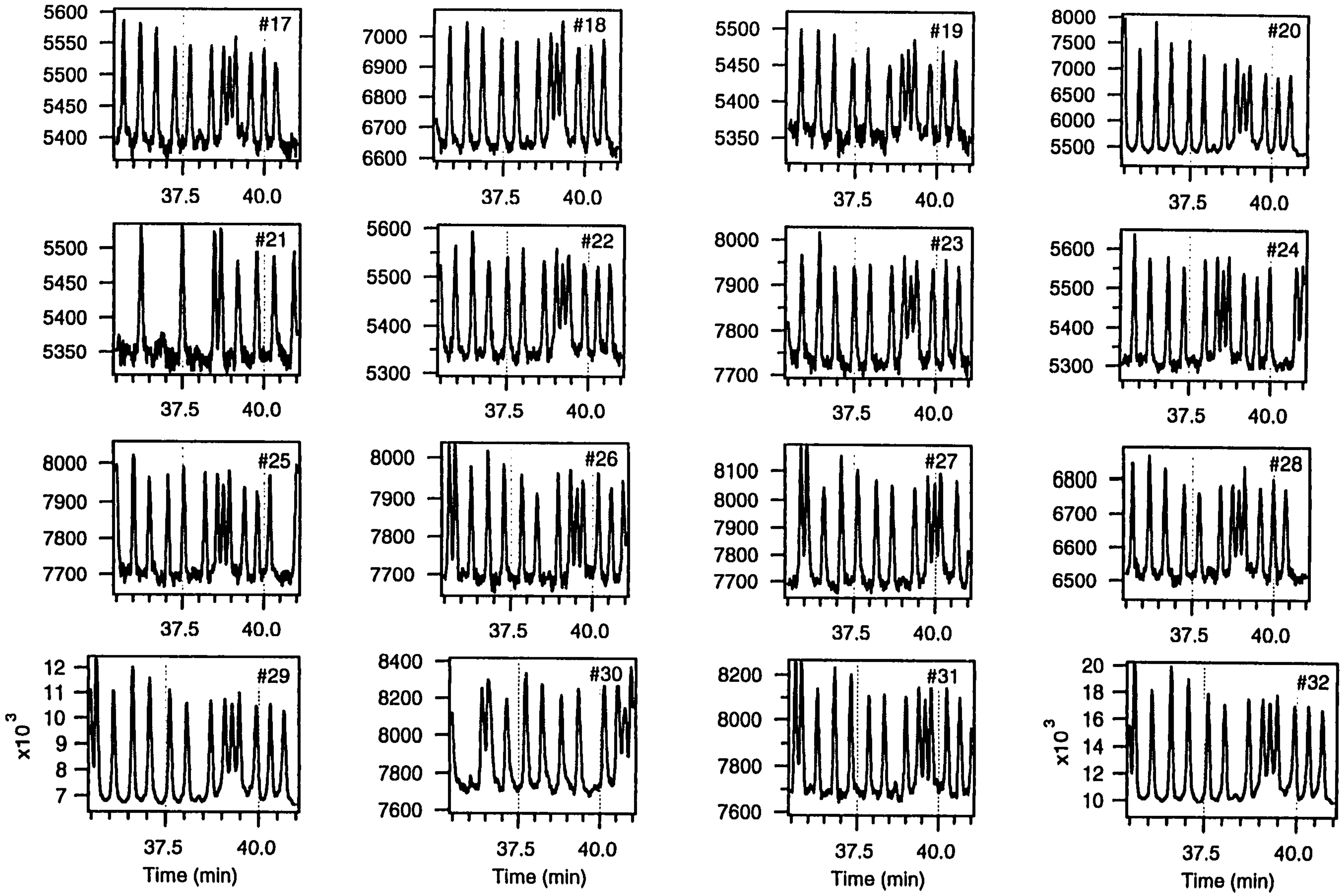


Figure 4.10 Layout of T-terminated DNA sequencing data from 16 capillaries

capillaries, the user can choose which eight capillaries to be displayed by entering different values of the parameter "j": 0 to display capillaries #1 to #8, 1 to display #9 to #16; ..., and so on. Then, by placing cursors, which appear in the Info window of each graph, on the curve of one graph and calling Macro "LOD()" or "LOD_peakheight()", the LOD for the electropherogram is calculated and printed on the graph. This is then repeated for all 8 graphs. After calculating all the LODs, the Macro "layout8x1()" can be called to print the 8 electropherograms with calculated LOD results on each graph.

The whole process can only be done semi-automatically, because the user needs to place the two cursors on the specific location on the graph. Macro LOD() is written to calculate LOD for the electropherograms without bubble peaks near the analyte peak. The user needs to place cursor A on the top of electrophoretic peak, and place cursor B on the baseline far behind the analyte peak, then call the Macro LOD(). The computer will then automatically calculate the standard deviation of the baseline, measure the peak height and then calculate the LODs.

Macro LOD_peakheight() is written for those electropherograms with bubble peaks near the analyte peak. In these cases, the user needs to measure the peak height and migration time of the analyte peak first, then choose a portion of the baseline that does not include bubble peaks and place both cursors on that region of the baseline for a calculation of standard deviation. After calling the LOD_peakheight(), the Macro will ask for "Hp" (peak height) and "t_mig" (migration time of the analyte in seconds), then calculate both the LOD in concentration and in molecules, and print them on the graph.

Before using these two LOD calculation Macros, some constants need to be changed to fit the experimental conditions, these changes are done in the source code where they are labeled with triple question marks. For example, the injection time, injection voltage, running voltage, length of capillary, and the concentration of the dye used for the measurement all vary from experiment to experiment and need to be set to correct values for the LOD calculation.

Procedure 4: Image Plot for Spectral Change with Time

All of the above procedures work on a line graph of the electropherogram, which shows the variation of light intensity with time. In the line plot format of an electropherogram, different spectral channels have to be specified by different lines distinguished by color as shown in Figure 4.11 a and b. As discussed in Section 4.1.2, in the 32-capillary spectrometer, each capillary has 10 spectral channels. If we want to look at the data of every spectral channel simultaneously, it is not practical to show all 10 spectral channels on one line plot.

An image plot shows the electropherogram in a different way. As shown in Figure 4.11 c and d, the X-axis is still migration time (the frame number can be converted to seconds if divided by data collection speed), and the Y-axis is not the light intensity but the spectral channel. Light intensity is encoded by the brightness of the spectral bands. The Image plot provides a better way to monitor the change of all the spectral channels with time.

As mentioned in Section 3.5.2, different dyes can be discriminated by their maximum emission wavelength or by the location of spectral bands. Figure 4.11 shows, as an example, the image plot (the spectral bands) and the line plot (the maximum emission) of one short section of an electropherogram. Here, data from two capillaries are provided. The two image plots (c and d) are produced by calling this procedure. Obviously, the procedure supports the handling of data from multiple capillaries.

Because of the Raman scattering, the background level in each of the 10 spectral channels is different. The background level for each channel has to be normalized, in order to get a good image plot showing the true spectral image of the dye. The procedure allows the user to correct the background level of each spectral channel. It consists of two Macros, each of which prompts the user for input parameters and then calls a Function. Macro "DisplayAllWaves()" displays the waves of all 10 spectral channels for one capillary. The user is able to examine different capillaries by entering different values of

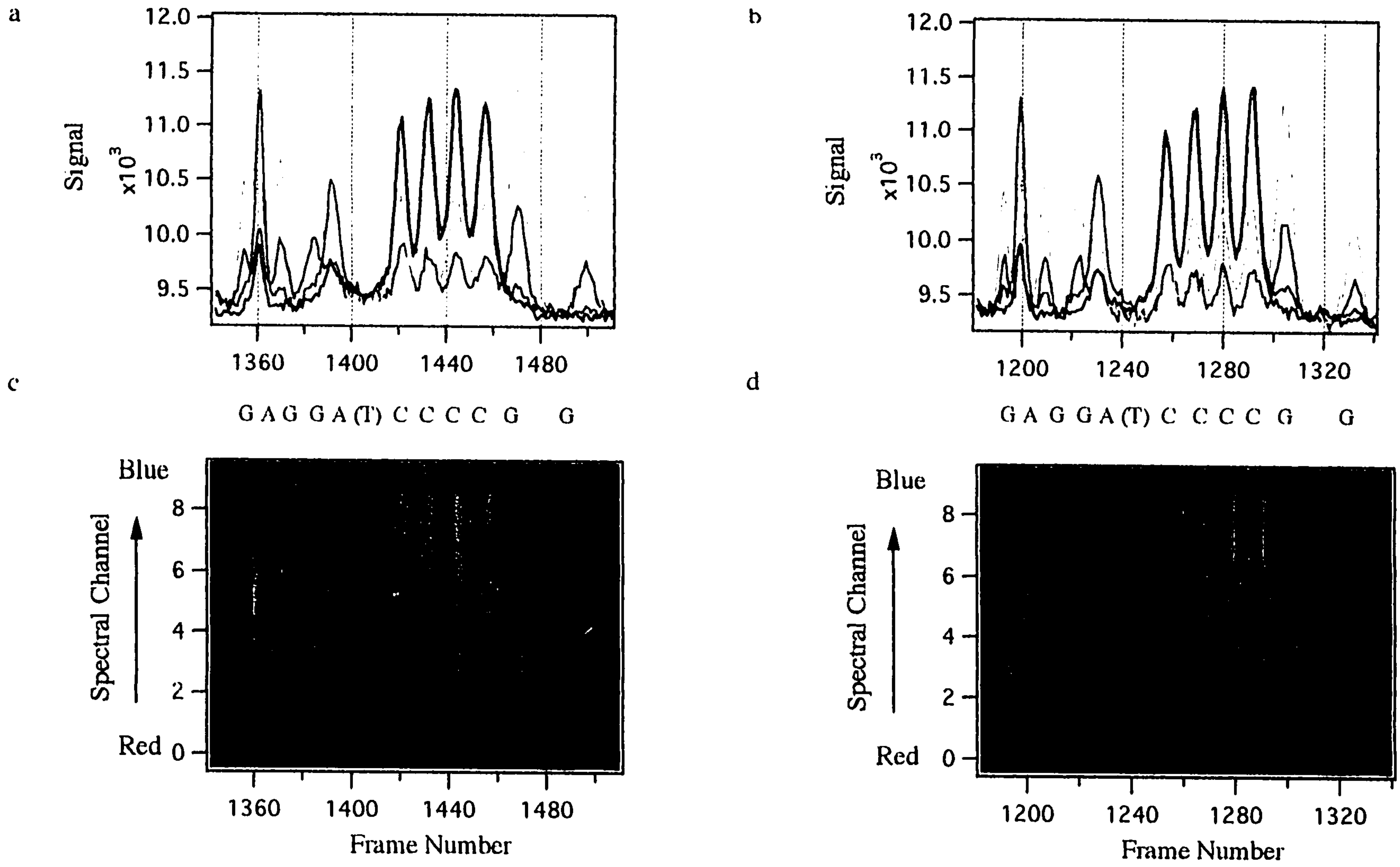


Figure 4.11 Four-color DNA sequencing data for capillaries #13 (a,c) and #9 (b,d)
a,b) line plot, T-red (w1), G-yellow(w3), A-green(w6), C-blue (w8)
c,d) image plot

"n" as instructed. After the 10 waves are displayed, the user records the background level of each wave (spectral channel), then calls "Macro plot_image()". The background level of each wave is then corrected according to the value the user enters, and an image plot of the electropherogram like those in Figure 4.11 is displayed.

This chapter discussed the software design of both the data acquisition and data analysis for the 32-capillary spectrometer. The next chapter will discuss the sensitivity and preliminary applications of the instrument. Both data acquisition programs discussed here in Section 4.1 and Section 4.2 are used for data collection in the various experiments of the next chapter. Igor procedures, such as the ones discussed here in Section 4.3, are also used for the data analysis. In general, programs discussed in this chapter have been successfully applied in the actual experiments and data analysis described in the next chapter.

REFERENCES

1. D. Mark, Learn C on the Macintosh, 2nd ed.; Addoson-Wesley Publishing Company, Don Mills, ON 1995
2. H. M. Deitel, P. J. Deitel, C How to Program, Prentice Hall, Inc. Englewood Cliffs, NJ 1992
3. Apple® Inside Macintosh Volume I, II, III, Apple Computer Inc., Addison Wesley Publishing Company, Inc., Reading, MA 1995
4. Professional Programming Kits, Photometrics Ltd., Tucson, AZ 1995
5. L. K. Wells, J. Travis, LabVIEW for Everyone: Graphical Programming Made Even Easier, Printice Hall PTR, upper Saddle River, NJ 1997
6. LabVIEW® for Macintosh User Manual, National Instruments Corporation, Austin, TX 1994
7. Igor Pro 3 Programming and Reference Manual, 3 ed., WaveMetrics, Inc., Lake Oswego, OR 1996

Chapter 5. Performance of the 32-Capillary Spectrometer

5.1 INTRODUCTION

5.1.1 Limit of Detection

Limit of detection (LOD) is an important measurement of an analytical instrument. It indicates the lowest detectable concentration of an analyte at a specific confidence level on the instrument.

The LOD is measured by injecting a dye solution into open-tube, non-treated capillaries and is calculated through the peak height and noise level of the electropherogram. The LOD can be expressed in terms of concentration or number of molecules. Using the 3-standard deviation definition of a limit of detection, LOD in concentration (LOD_C) is related to the peak height (H_p), standard deviation of background (σ) and concentration of the injected dye (C_{dye}) as follows:

$$LOD_C = (3\sigma / H_p)\, C_{dye} \qquad (5\text{-}1)$$

LOD in the number of molecules (LOD_m) is related to the peak height (H_p), standard deviation of background (σ) and the number of molecules injected into the capillary (N_{dye}):

$$LOD_m = (3\sigma / H_p)\, N_{dye} \qquad (5\text{-}2)$$

where N_{dye} can be calculated from the following equation:

$$N_{dye} = V_{inj}\, C_{dye}\, (6.02 \text{ x } 10^{23}) \qquad (5\text{-}3)$$

where V_{inj} is the volume of the dye being injected into the capillary, and it can be estimated by the following equation:[1]

$$V_{inj} = (\pi\, r^2\, L)\, [(t_{inj}\, E_{inj}) / (t_{mig}\, E_{el})] \qquad (5\text{-}4)$$

where L is the length of the capillary, r is the inner radius of the capillary (25 µm), t_{inj} is the injection time, E_{inj} is the injection field, t_{mig} is the migration time of the dye, and E_{el} is the electrophoretic field.

An Igor procedure was written to quickly calculate the LOD_C and LOD_m according to the above equations. This procedure was discussed in Section 4.3.2 and is used to calculate LODs in all cases here.

5.1.2 Fluorescence Labeling of Peptide

A few dyes, such as FQ (3-(2-furoyl)quinoline-2-carboxaldehyde), CBQ (3-(p-carboxybenzoyl)quinoline-2-carboxaldehyde), FITC (fluorescein isothiocyanate), can be utilized to label a peptide. FQ or CBQ does not fluoresce until it reacts with a free amino group of a peptide,[2] so labeling a peptide with FQ or CBQ should yield a low background. However, FITC-labeling gives high background because the free dye itself fluoresces.

The peptide is dissolved in water and diluted to a concentration of 10^{-4} M, the FQ or CBQ dye is dissolved in methanol to a final concentration of 10 mM, and potassium cyanide (KCN) aqueous solution is diluted into 25 mM borate buffer to a final concentration of 25 mM. Then the labeling reaction is performed as follows: Mix 4 μL of the peptide solution with 5 μL dye solution into a 0.6 mL microcentrifuge tube (Fisher Scientific, Pittsburgh, PA, USA), and warm the mixture in a 65°C thermal water bath for 2 min using a "dry bath incubator" (Fisher Scientific, Pittsburgh, PA, USA), then add in 1 μL KCN solution to the mixture and continuously incubate the reaction mixture at the 65°C water bath for 15 min. After incubation, the labeling reaction is terminated by addition of 90 μL running buffer, which is 10 mM borate and 5 mM SDS buffer (pH 9.2).

5.1.3 Steps to Evaluate the Instrument

The spectrometer has three major systems: (1) separation system — the sheath-flow cuvette and the high voltage supply, (2) optical detection system — the two camera lenses and the CCD camera, and (3) light dispersion system — the prism. Performance of the three parts has been investigated individually.

Firstly, in the separation system (32-capillary array in the cuvette), the performance of the cuvette was investigated with a single-capillary detection system, which establishes the sheath-flow conditions for the cuvette. Secondly, the sensitivity of the optical detection system, which is the performance of two camera lenses and the CCD camera without the prism, was studied. Lastly, the overall performance of the spectrometer was evaluated.

5.2 EXPERIMENTAL

5.2.1 Evaluation of the Separation System

LOD by PMT

To test the sheath-flow and the light scattering properties of the 32-capillary cuvette, a detection system of a single capillary instrument was used to measure the LOD of one capillary in the 32-capillary cuvette. In this case, all 32 capillaries were inserted into the cuvette, but only one capillary was monitored. The schematic experimental setup is shown in Figure 5.1. A 515DF20 bandpass filter was used to let the green laser line pass for excitation of fluorescence. A 60x microscope objective was used to efficiently collect fluorescence from the monitored capillary and to image the fluorescence onto a pinhole which was used as a spatial filter to remove the scattered excitation light. The fluorescence light then passed through a 560DF40 bandpass filter to further remove the scattered excitation light. Intensity of the fluorescence was then measured by a PMT (Hamamatsu, USA) biased at 1100 V.

The dye used in this LOD measurement is R6G which is diluted into a 10 mM borate 5 mM SDS buffer solution to a final concentration of 4.7 x 10^{-11} M. The R6G solution was injected into a 39.5-cm long capillary at 30 V/cm for 5 s. The capillary electrophoresis was performed at an electric field of 450 V/cm in the 10 mM borate 5 mM SDS buffer. The excitation wavelength of this dye is 528 nm and the maximum emission wavelength in methanol is 550 nm.[3] Data acquisition was performed with a LabVIEW DAQ board (NB-MIO-16XL, National Instruments, Austin, TX, USA). The software

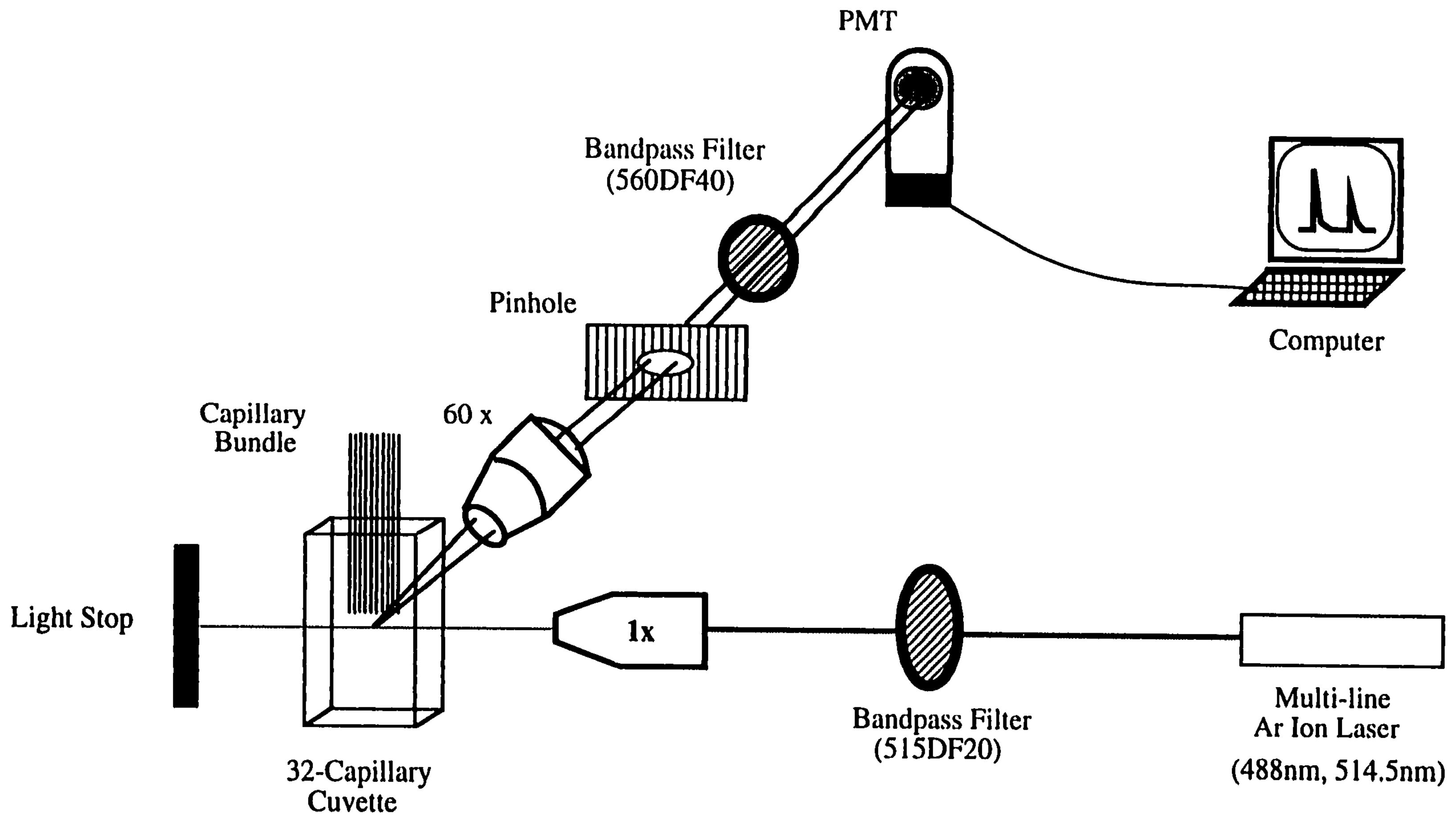


Figure 5.1 Instrument setup for performance of 32-capillary cuvette

described in Section 4.2 was used for data collection. The software was able to monitor three data channels simultaneously, but only one channel was used for this application.

5.2.2 Performance of the Optical Detection System

LOD by CCD Without Spectral Dispersion

The detection optics of the 32-capillary spectrometer, which include the two camera lenses and the CCD camera, were compared with those for a single capillary instrument, which include the microscope objective and the PMT shown in Figure 5.1. The schematic diagram of the instrument for this measurement is shown in Figure 5.2. This setup differs from that shown in Figure 3.14 in that the prism is not in place here, and the two camera lenses are aligned with the object side facing each other with a 1.5-inch diameter 580DF30 bandpass filter in between. The first camera lens collects the fluorescence from the cuvette and collimates the light; the second lens re-images the fluorescent light onto the CCD camera. In this manner, we can determine the sensitivity of the detection optics of the spectrometer, and also the sensitivity change following incorporation of a prism. This will be discussed in more detail in Section 5.3.1.

The LOD was measured using R6G as well as fluorescein in 10 mM borate and 5 mM SDS solution. Capillary electrophoresis was performed using 10 mM borate and 5 mM SDS buffer as the running buffer.

In the case of R6G, only one of the 32 capillaries was used for the measurement. R6G of 4.7 x 10^{-11} M was injected into the 41.5-cm long capillary at 1 kV for 5 s, and capillary electrophoresis was then performed at 18 kV electric field. The 514.5 nm green laser line was used for excitation of R6G, and a 1.5-inch 580DF30 filter was used for detection.

In the case of fluorescein, all 32 of the 41-cm long capillaries were injected with 6 x 10^{-11} M fluorescein solution at 1kV for 5 s, and then capillary electrophoresis was

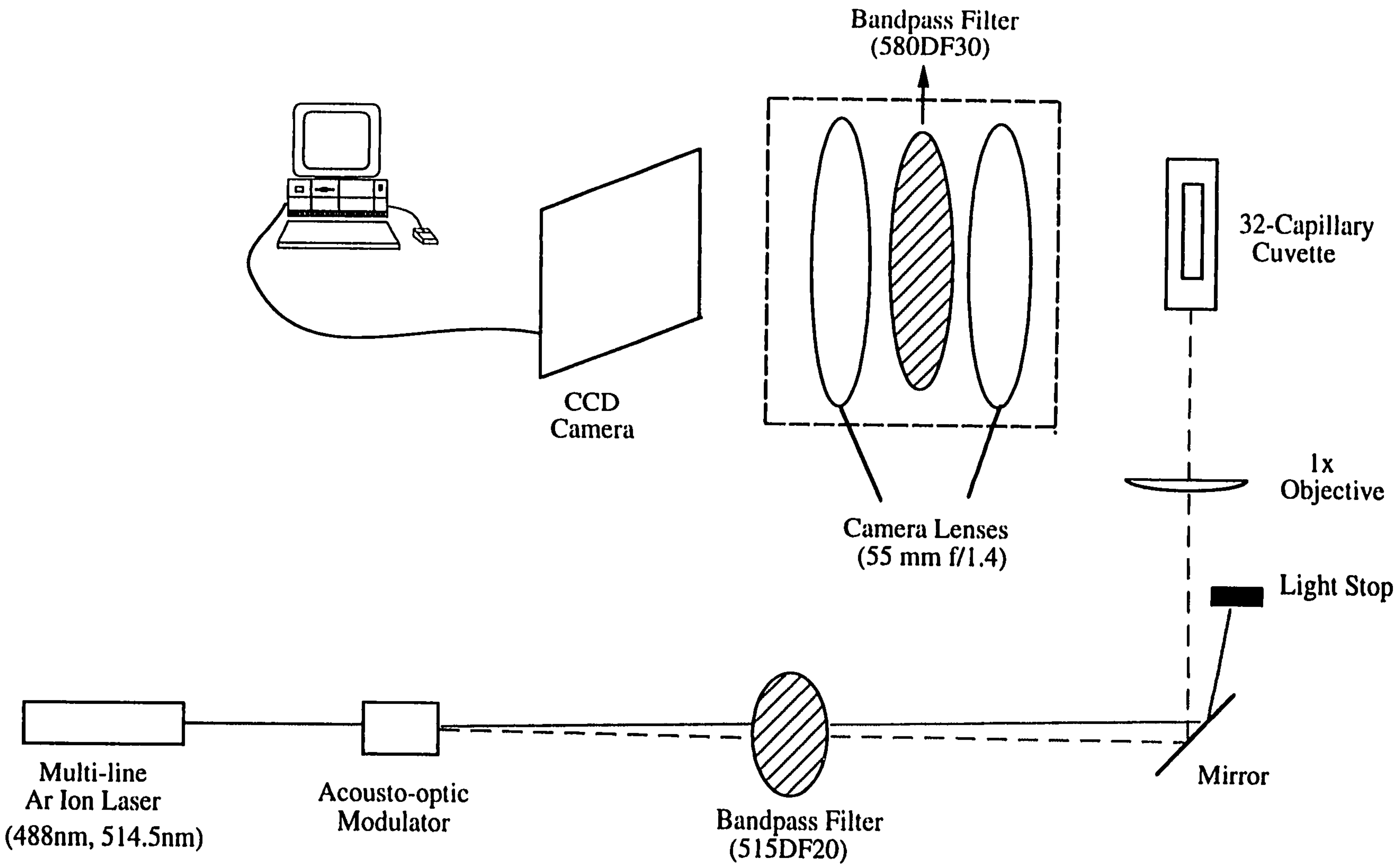


Figure 5.2 Instrument setup for evaluation of camera lenses and CCD camera

performed at 12 kV electric field. The 488 nm blue laser line was used for excitation, and a 1.5-inch diameter 530DF30 filter was employed for detection of the fluorescence.

In both cases, Darren Lewis's data acquisition software was used for data collection.[4] This software integrates the light intensity in each area on the CCD chip illuminated by the fluorescent spot. The output data has the data type of unsigned 32-bit. The exposure time of the CCD camera was 200 ms, and the DAQ speed was 3.75 Hz.

Single-Color DNA Sequencing

The setup shown in Figure 5.2 was used for single-color DNA sequencing. DNA sequencing fragments were separated using 5% linear polyacrylamide in 7 M urea and 1 X TBE buffer at room temperature. The treatment of capillaries is similar to that in Section 2.2.3, the 32 capillaries were first treated with a 2% [γ-(methacryloxy)propyl] trimethoxysilane solution in 95% ethanol for 20 min. Then the capillaries were filled with the freshly made 5%T polyacrylamide through the application of vacuum. The DNA sample used was the primer-labeled T-terminated M13mp18 cycle sequencing sample, where the primer was labeled with Tamra and the sequencing products were dissolved in formamide. DNA was electrokinetically injected into the 40 cm long capillaries at 4125 V for 17 s, and then the sample vial was replaced with 1xTBE buffer for electrophoresis at 8 kV.

The 514.5 nm green laser line was used for excitation of Tamra, and a 580DF30 1.5-inch diameter bandpass filter was utilized for detection of the fluorescence from Tamra. The same data acquisition software as used in the LOD measurement above was used for the single-color DNA sequencing.[4] The size of the single-color data is much smaller than that of the four-color data, because there is only one data point per frame for each capillary due to the lack of the spectral information. The exposure time of the CCD camera was 500 ms, and the DAQ speed was 1.72 Hz.

5.2.3 Detectability and Application of the 32-Capillary Spectrometer

After testing the detectability of the fluorescence emission device (the cuvette) and the optical detection system (the two camera lenses and the CCD camera), the dispersing prism was put in place. Detectability of the spectrometer was evaluated by LOD analysis. Performance of the spectrometer was further investigated by peptide analysis and DNA sequencing. The schematic diagram of the 32-capillary spectrometer is shown in Figure 3.14. The DAQ software "Cam" discussed in Section 4.1 was used for data collection for all the experiments in this section.

LOD With Spectral Dispersion

The LOD of the spectrometer was measured by 1.2 x 10^{-10} M fluorescein solution in 10 mM borate 5 mM SDS buffer. The instrument setup is shown in Figure 3.14 where the cutoff filter behind the cuvette is a LP515 filter and the multi-line laser is filtered with a 488NB3 filter to let the 488 nm blue line pass through. This setup differs from Figure 5.2 by the addition of the dispersing prism; therefore, the spectrum of the fluorescence emission is obtained. The capillaries used in the measurement were 39.1 cm long. The fluorescein was injected into all 32 capillaries at 3 kV for 5 s, then the 10 mM borate 5 mM SDS buffer was used as the running buffer for capillary electrophoresis at 15 kV. The exposure time of the CCD camera was 200 ms, and the DAQ speed was 2.3 Hz.

Peptide Analysis

FQ and CBQ (Molecular Probes, INC., Eugene, OR, USA) were used to label two peptides: bradykinin and Ile-Ser-bradykinin (Sigma-Aldrich Canada, Ltd. Oakville, ON). The dye-labeled peptides were analyzed with 41.5 cm capillaries in 10 mM borate 5 mM SDS buffer using open-tube capillary electrophoresis. The labeled-peptide products were injected into the capillary at 4 kV for 5 s. The sample valve was then replaced with 10 mM borate 5 mM SDS buffer for separation at 12 kV. The exposure time of the CCD camera was 200 ms, and data was collected at 2.3 Hz.

The instrument setup is shown in Figure 3.14, where a LP530 filter was placed after the cuvette and before the detection optics to cut off the light with a wavelength less than 530 nm and a 488NB3 filter was used to let the blue laser through for excitation of fluorescence.

Four-Color DNA Sequencing

The ABI four-dye (Fam, Joe, Tamra, Rox) labeled M13mp18 four-color cycle sequencing sample was separated with 5% linear polyacrylamide in 7 M urea and 1 X TBE buffer (pH 8.5) at room temperature. The sample preparation was similar to that described in Section 2.2.2. The preparation of the 40-cm long gel-filled capillaries was the same as that in Section 2.2.3. DNA was injected into the capillaries at 4 kV (100 V/cm electric field) for 50 s. After the injection, the injection ends of the capillaries were transferred to a 1 X TBE buffer container, and a 8 kV Voltage (200 V/cm field) was then applied to the capillaries for separation.

Both 488 nm and 514.5 nm laser lines were used for excitation of the four dyes. The LP530 cutoff filter was used for detection. The exposure time of the CCD camera was 400 ms, and the data was collected at 1.54 Hz

5.3 RESULTS AND DISCUSSION

5.3.1 Limit of Detection

Sheath Flow Rate

Before measuring the LOD, the optimum sheath-flow rate was determined using the setup shown in Figure 5.1. Flow rate in the 32-capillary cuvette is typically faster than that in the single-capillary cuvette or the 5-capillary cuvette, because of the designed space between capillary channels. (see Section 3.2.1.) In an open-tube experiment, the peak width is typically a couple of seconds. If the sheath-flow is too fast, the period when the dye stays at the detection region is too short, and the sensitivity of the measurement decreases. But if the sheath-flow is too slow, the electrophoretic peak becomes wide and distorted; therefore, an optimum flow rate is important during any measurement.

The sheath-flow rate is regulated by the height difference (Δh) between the sheath-flow buffer reservoir and the waste container. The suitable sheath-flow rate is determined by comparing the peak height and peak width of each electrophoretic peak at different *Δh*, and the peak width is represented by the full width at half maximum (FWHM) of the peak.

The experiment was done by injecting a 10^{-8} M R6G solution at an electric field of 37.5 V/cm for 5 s, then a 225 V/cm electric field is applied with a 10 mM borate and 5 mM SDS buffer solution as the running buffer. The measurement was done with four height differences (Δh), and the results are shown in Table 5.1.

Table 5.1 Effect of the sheath-flow rate on the electrophoretic peak

Δh (cm)	0.5	1.0	1.5	2.5
FWHM (s)	2.70	2.19	2.10	2.07
peak height (points)	161	139	115	100

As shown in Table 5.1, when Δh increases, the peak height decreases as expected. If there is no distortion of the electrophoretic peak from sheath-flow, the FWHM of the peak should not vary much when the sheath-flow rate changes, since the same separation conditions are applied from one measurement to another in this experiment. According to the results shown in Table 5.1, the peak width does decrease 19% from 2.7 s to 2.19 s when Δh increases from 0.5 cm to 1 cm, indicating that the sheath-flow is too slow at 0.5 cm height difference between the sheath-flow reservoir and the waste container. When

Δh is increased to 1.5 cm, the peak width decreases 4% more to 2.1 s. However, the peak width changes only 1% when Δh increases further from 1.5 to 2.5 cm, indicating that there is no distortion of the peak from the sheath-flow at the flow rates of Δh 1.5 cm and the increase of Δh above 1.5 cm only reduces the detection time of the fluorescent band. Therefore, the optimum sheath-flow rate can be set at Δh 1.5 cm for the separation system of this 32-capillary spectrometer.

For all the LOD measurement discussed below, Δh was set at 1.5 cm, and the injection end was set at the same height as the waste to eliminate siphoning. Figure 5.3 and Figure 5.4 show a series of LOD measurement at different detecting conditions. The LOD_c and the LOD_m listed on the graph were calculated using the Igor procedure 3 discussed in Section 4.3.2. All the electropherograms were smoothed by a Savitzky-Golay 9 points second-order filter.[5, 6]

LOD Without Spectral Dispersion

Figure 5.3 shows that the LOD of the CCD detection system in the spectrometer is comparable to that of the PMT detection system in a single capillary instrument. The comparison was performed in two experiments.

First, performance of the 32-capillary cuvette was studied using a detection system shown in Figure 5.1. This PMT detection system examines only one capillary in the cuvette. R6G is diluted in 10 mM borate 5 mM SDS buffer solution to a final concentration of 4.7 x 10^{-11} M. Injection of 1.1 nL of this R6G solution into the capillary gives a peak shown in Figure 5.3 a. The LOD_c calculated for this measurement is 3.6 x 10^{-12} M, and LOD_m is about 2400 molecules.

Next, the detection system was replaced with the one shown in Figure 5.2. The CCD detection system can monitor all 32 capillaries simultaneously, but in this measurement only one capillary is being injected with R6G. The peak shown in Figure 5.3 b comes from the injection of 0.8 nL 4.7 x 10^{-11} M R6G solution through

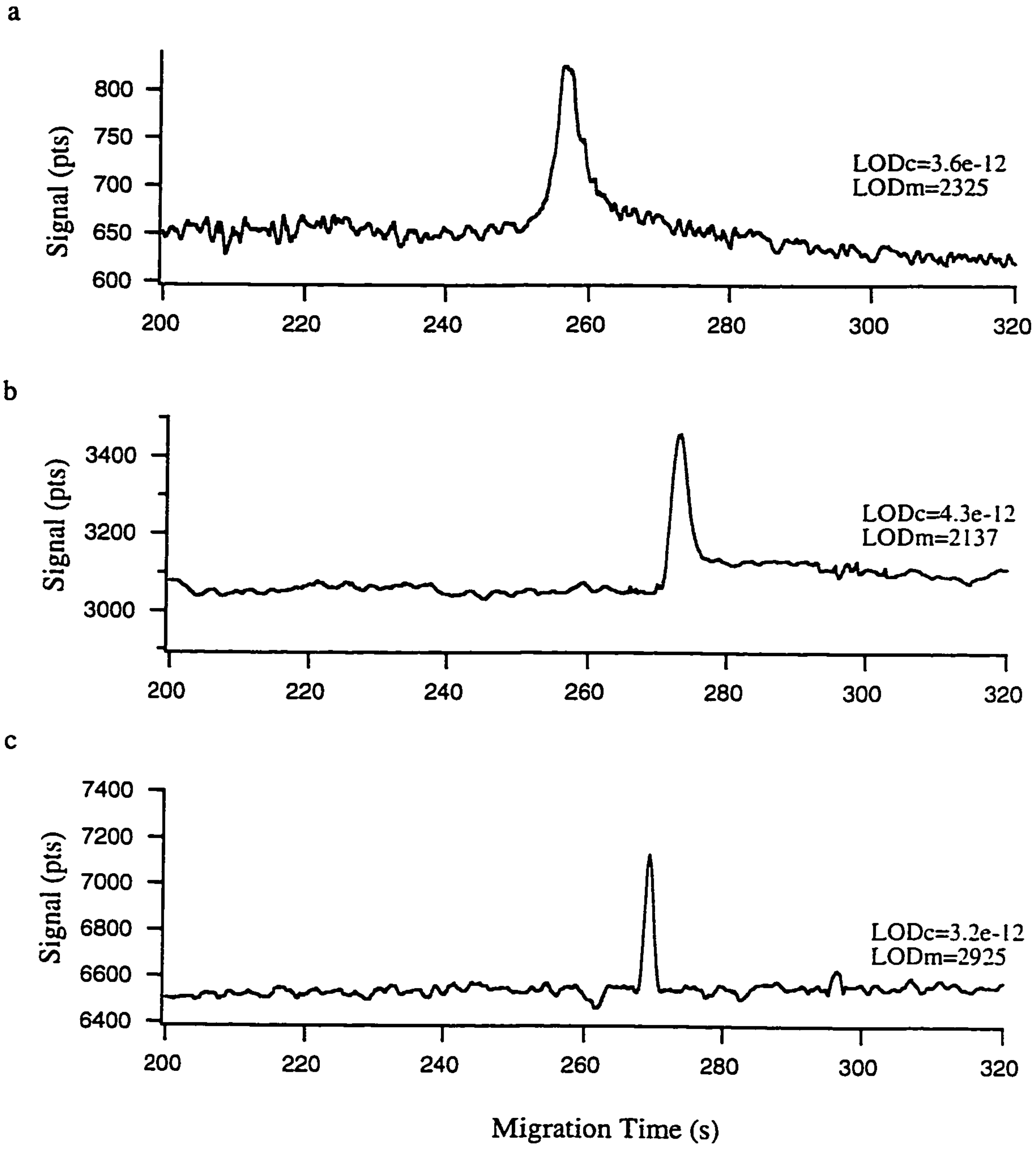


Figure 5.3 LOD measured by a) PMT and b, c) CCD
buffer: 10 mM borate 5 mM SDS
a) injection of 1.1 nL, 4.7 X 10^{-11} M R6G
b) injection of 0.8 nL, 4.7 X 10^{-11} M R6G
c) injection of 1.5 nL, 6 X 10^{-11} M Fluorescein

the capillary. The LOD_c from this measurement is 4.3 x 10^{-12} M, and LOD_m is approximately 2200 molecules.

Comparing the limits of detection (Figure 5.3 a and b) of the two different detection systems in Figures 5.1 and 5.2, one can conclude that the two detection systems are comparable to each other in terms of detectability. Examining the electrophoretic peaks of R6G in Figure 5.3 a and b, we can see that the peaks are broad and tailing, which indicates that R6G tends to stick to the capillary wall due to electrostatic force.

Fluorescein is negatively charged at the pH 9.2 of borate SDS buffer, so there should be less interaction between the negatively charged ions and the negatively charged capillary wall. This makes fluorescein a better dye relative to R6G for LOD measurement, so the experiment is repeated with a fluorescein solution (Figure 5.3 c) rather than a R6G solution (Figure 5.3 b). The instrument setup used is shown in Figure 5.2, where the detection filter is replaced with a 530DF30 filter (1.5 inch diameter), and the multi-line laser is filtered with a 488NB3 filter instead of the 515DF20 filter.

1.5 nL of the 6 x 10^{-11} M was injected into each of the 32 capillaries, giving a typical result shown in Figure 5.3 c. The LOD_m from this measurement is about 3000 molecules, and the LOD_c is 3.2 x 10^{-12} M. Comparing Figures 5.3 b with c, we can see that the electrophoretic peak of fluorescein is narrower and more symmetric than the peak of R6G, which confirms that fluorescein has less interaction with the capillary wall.

The LODs from the three measurements in Figure 5.3 are close to each other within the same order of magnitude, indicating that the detectability of the CCD camera with the two camera lenses is as good as that of the PMT with the microscope objective.

LOD With Spectral Dispersion

Figure 5.4 shows how the LOD is affected by addition of the prism. Figure 5.4 a is the same graph as Figure 5.3 c, which indicates the efficiency of the light collection (two camera lenses) and detection (CCD) system of the spectrometer. Figure 5.4 b and c is measured with the spectrometer setup shown in Figure 3.14.

Injection of 3.6 nL of 1.2 x 10^{-10} M fluorescein solution into each of the 32 capillaries gives a typical example shown in Figures 5.4 b and c.

As mentioned in Section 4.1, each spectrum dispersed by the prism is binned into 10 spectral channels (w0 to w9) with the red most part of the spectrum in the first spectral channel (w0). The maximum emission wavelength of fluorescein measured at pH 9 is 514 nm,[3a] and is pushed to a longer wavelength in our measurement because a LP515 filter is used for detection. The measured maximum emission turns out to be in the 9th spectral channel (w8), of which the electropherogram is shown in Figure 5.4 b. The calculated LOD_c of this spectral channel is 1.1 x 10^{-11} M, and the LOD_m is approximately 25200 molecules.

Figure 5.4 c shows the resultant electropherogram after all the 10 spectral channels are added together. The calculated LOD_c is 6.4 x 10^{-12} M and the LOD_m is about 14400 molecules. Compared with the w8 spectral channel (Figure 5.4 b), the LOD is improved by 2-fold after the superimposing of all spectral channels (Figure 5.4 c). Also, Figure 5.4 c should be able to show the final light collection efficiency of the 32-capillary spectrometer because the peak corresponds to the intensity of all emission wavelengths finally reaching the CCD camera. Therefore, by comparing the LODs in Figures 5.4 a and c, we should be able to see the sensitivity loss caused by the prism. The data shows that adding the prism does degrade the LOD_c of the instrument by 2-fold, possibly due to the reflective losses of the fluorescence light caused by the prism.

As shown in Figures 5.4 b and c, with fluorescein as the fluorophore and measured in capillary zone electrophoresis, the typical LOD_c of the 32-capillary spectrometer for one spectral channel is 1.1 x 10^{-11} M, while that for the superimposing of all spectral channels is 6.4 x 10^{-12} M. The LOD of the 32-capillary spectrometer can be compared with the reported multiple capillary instruments.

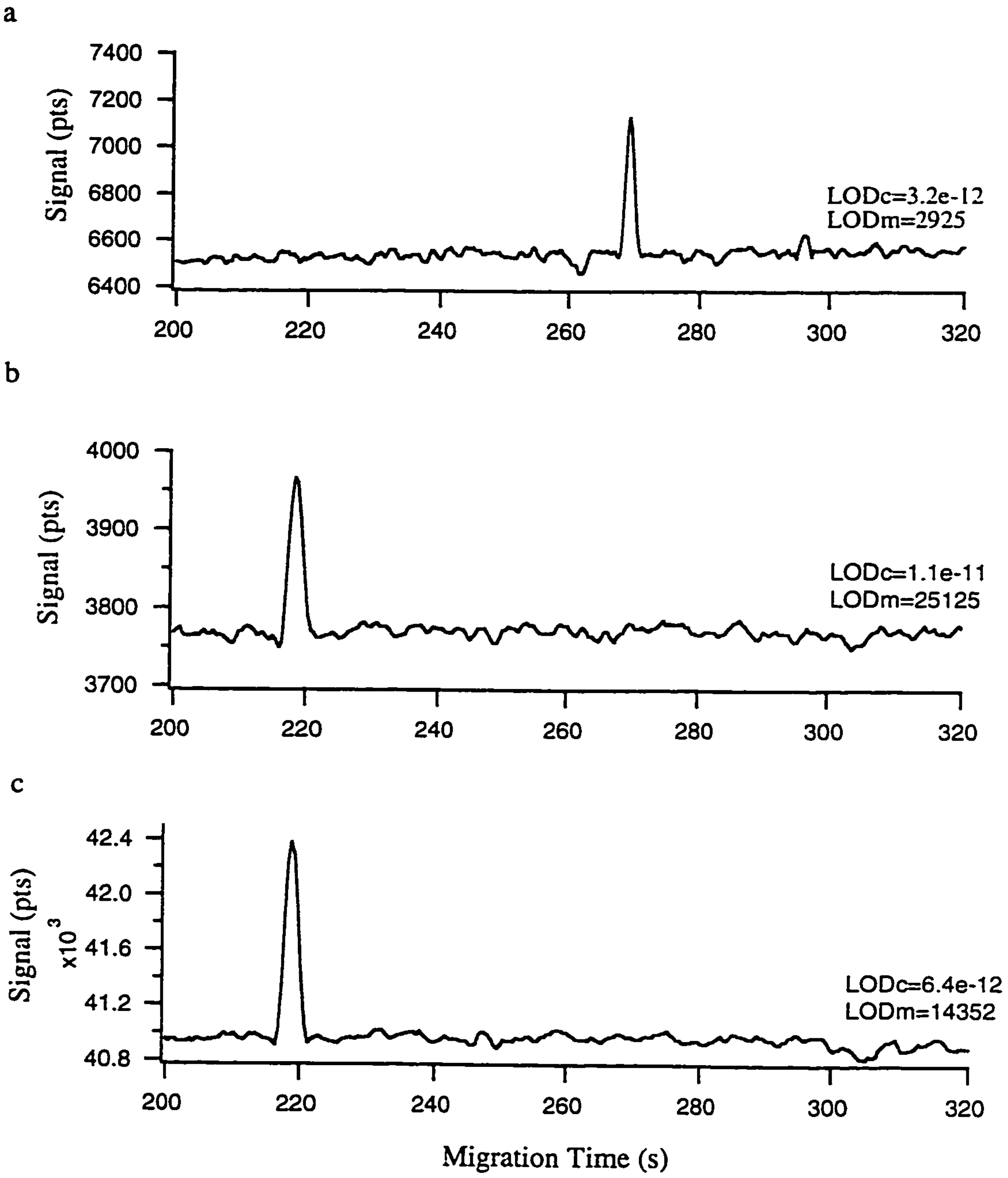


Figure 5.4 LOD measured a) without prism and b,c) with prism detector: CCD, buffer: 10 mM borate 5 mM SDS

a) injection of 1.5 nL, 6 X 10^{-11} M fluorescein

b) injection of 3.6 nL, 1.2 X 10^{-10} M fluorescein, w8

c) after ten spectral channels are added together

Yeung's group reported a 100-capillary array instrument with on-column detection.[7] With fluorescein as the fluorophore and measured in open-tube electrophoresis, they reported a LOD_C of 4 x 10^{-10} M without color-dispersion of the fluorescence light, which is approximately two orders of magnitude worse than our 32-capillary spectrometer.

Quesada reported an 8-capillary instrument[8] where optical fibers were used for laser illumination and fluorescence detection and an imaging spectrograph was used for fluorescence dispersion. With Fam-primer as the fluorophore and LOD measured in 4% linear polyacrylamide, a LODc of 1.5 x 10^{-11} M was obtained at a signal-to-noise ratio of 2.

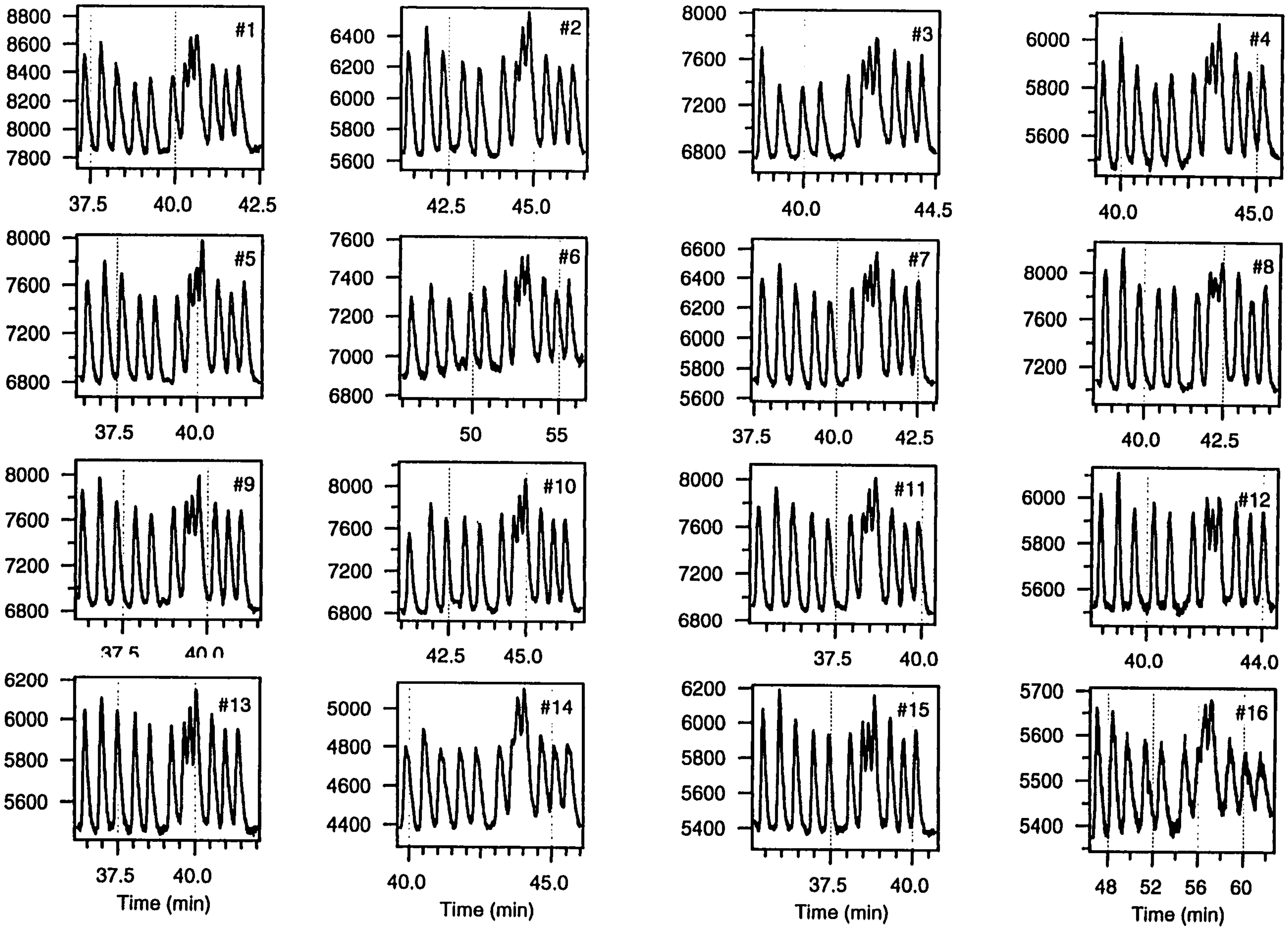
Zhang's 5-capillary instrument [9] is more sensitive due to its better light collection efficiency than the 32-capillary spectrometer. The 5 capillaries are aligned in a close array so that the total detection region is only 0.7 mm rather than 7.2 mm in the 32-capillary spectrometer. The LOD of the 5-capillary instrument is 2 x 10^{-13} M for fluorescein.

5.3.2 Performance of All 32 Sample Channels

Single-Color DNA Sequencing

As discussed above, the spectrometer's detection optics without the prism (setup shown in Figure 5.2) is sensitive enough to be comparable to a PMT detection system (setup shown in Figure 5.1). In order to further characterize the performance of all 32 sample channels, single-color DNA sequencing was done without dispersing the fluorescence spectra. The instrument setup used for this experiment is shown in Figure 5.2.

The resultant electropherograms are shown in Figure 5.5. In this case, two samples from different batches were used for sample injection: the first 16 capillaries are all injected from the same sample, while the second 16 capillaries are all injected from another sample. The Igor procedure 2 discussed in Section 4.3.2 was used to display the data


#1
8800
8600
8400
8200
8000
7800
37.5
40.0
42.5
#2
6400
6200
6000
5800
5600
42.5
45.0
#3
8000
7600
7200
6800
40.0
44.5
#4
6000
5800
5600
40.0
45.0
#5
8000
7600
7200
6800
37.5
40.0
#6
7600
7400
7200
7000
6800
50
55
#7
6600
6400
6200
6000
5800
5600
37.5
40.0
42.5
#8
8000
7600
7200
40.0
42.5
#9
8000
7600
7200
6800
37.5
40.0
#10
8000
7600
7200
6800
42.5
45.0
#11
8000
7600
7200
6800
37.5
40.0
#12
6000
5800
5600
40.0
44.0
#13
6200
6000
5800
5600
37.5
40.0
Time (min)
#14
5000
4800
4600
4400
40.0
45.0
Time (min)
#15
6200
6000
5800
5600
5400
37.5
40.0
Time (min)
#16
5700
5600
5500
5400
48
52
56
60
Time (min)

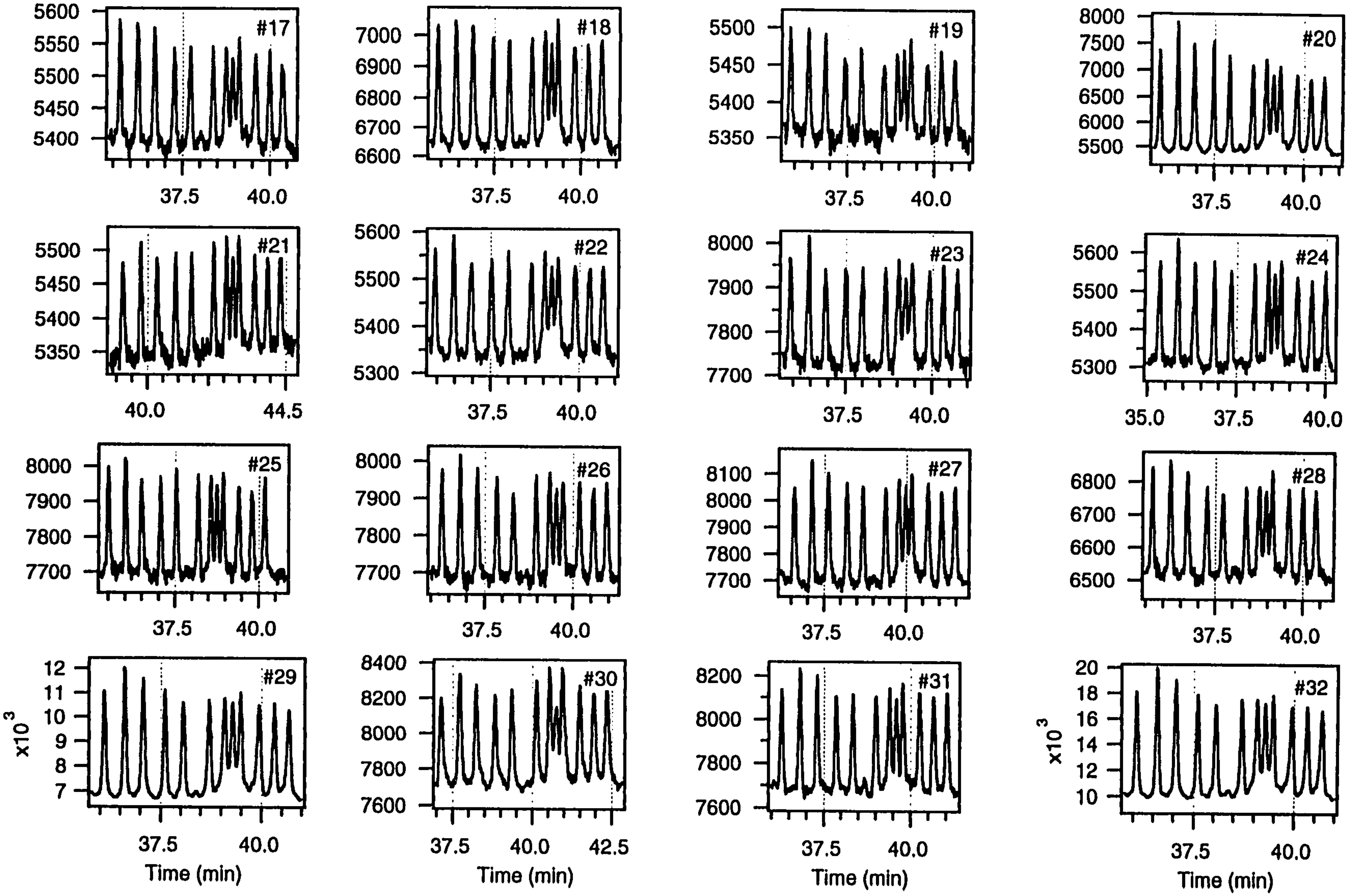


Figure 5.5 Single-color DNA sequencing for 32-capillaries (sequence shown corresponds to bases 80 to 107)

shown in Figure 5.5. The section of the electropherogram around 100 bases is chosen to compare the performance of each of the 32 capillaries. We can see from Figure 5.5 that the instrument has fairly consistent performance for all 32 capillaries.

Figure 5.6 shows an example of a complete DNA sequencing run from one capillary. With 5% linear polyacrylamide and running at 200 V/cm, separation up to 600 bases is achieved in less than 2 h.

5.3.3 Application of the 32-Capillary Spectrometer

One of the most important advantages of the 32-capillary spectrometer is its ability to provide spectral information, which helps to study the emission spectrum of a dye and discriminate different dyes based on their spectral properties. Here, two potential applications of the spectrometer are investigated.

Peptide Analysis

The electropherogram of each of the three dye-labeled peptides (FQ-bradykinin, CBQ-bradykinin, and CBQ-Ile-Ser-bradykinin) is shown in Figure 5.7. The 8th spectral channel (w7) is utilized to display the electropherogram. As illustrated in Figure 5.7, FQ-labeled bradykinin gives one clean peak, while CBQ-labeled bradykinin and Ile-Ser-bradykinin each gives two peaks. Both bradykinin and Ile-Ser-bradykinin have only one reactable free amino group, so multiple labeling should not be taking place. Therefore, only one peak should be present in CBQ-labeling as well as the FQ-labeling. This is not observed for CBQ-labeled product, possibly because the CBQ dye itself is not pure. As shown in Figure 5.7, it takes a longer time for CBQ-Ile-Ser-bradykinin to migrate to the detection region than CBQ-bradykinin, which makes sense because the latter peptide is two amino acids shorter than the former one. The migration time of FQ-bradykinin is close to that of CBQ-Ile-Ser-bradykinin. These two peptides were later analyzed as a mixture to see if it is possible to distinguish them.

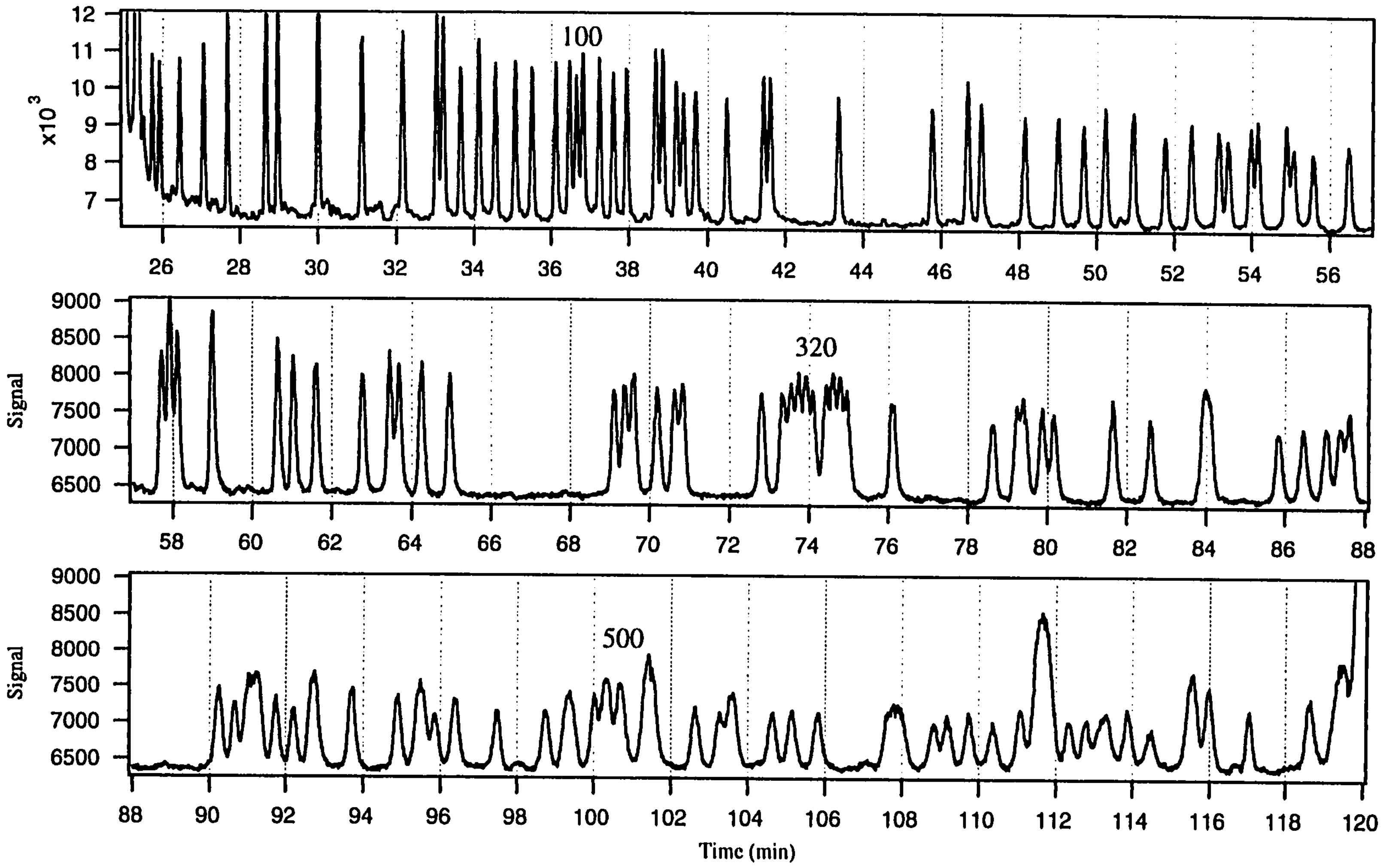


Figure 5.6 Single-color DNA sequencing from one capillary
sample: Tamra-labeled T-terminated M13mp18
gel :5%T linear polyacrylamide

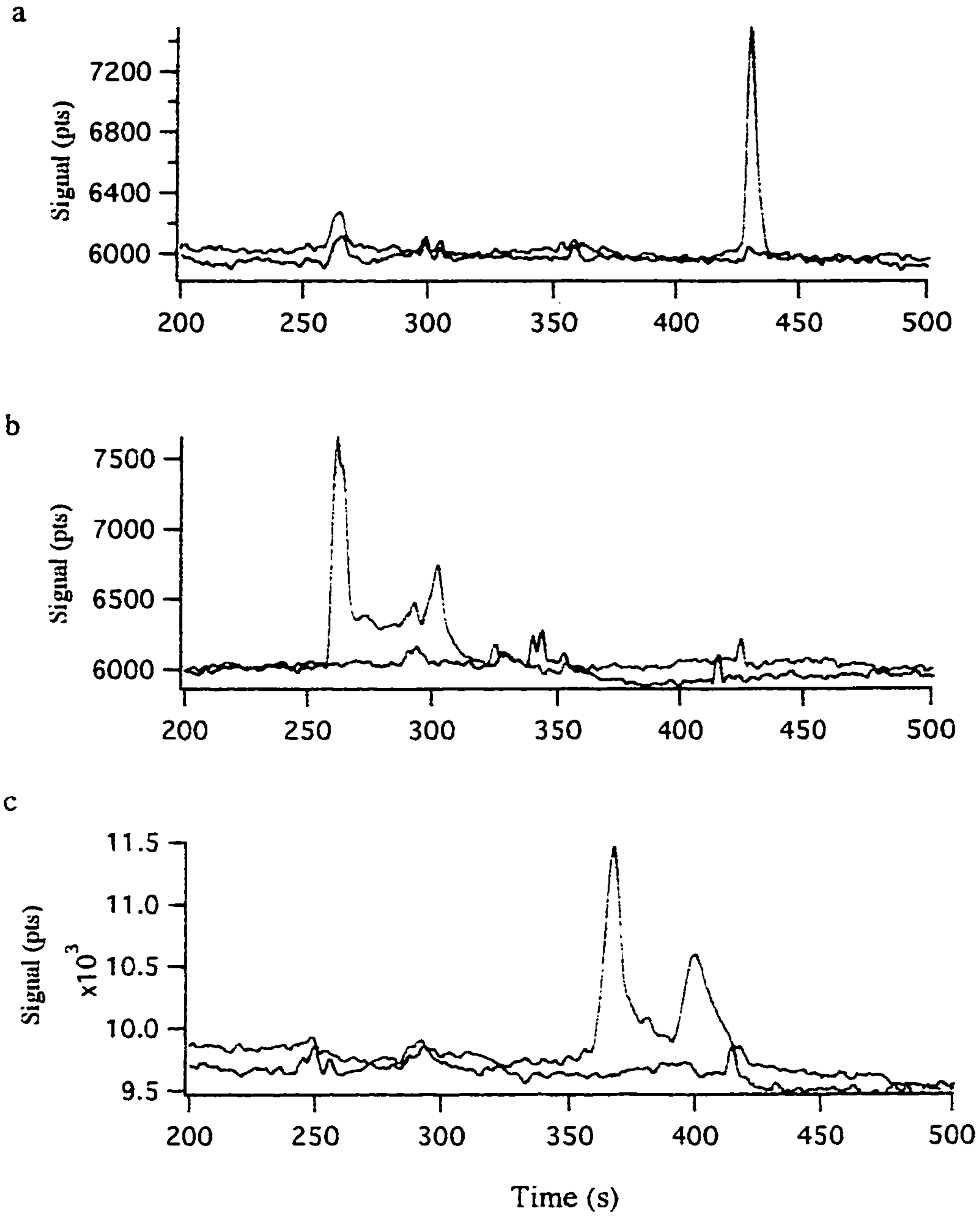


Figure 5.7 Electropherograms (w7) of dye-labeled peptides (red)
(green line on each graph is the blank)
a) FQ-bradykinin
b) CBQ-bradykinin
c) CBQ-Ile-Ser-bradykinin

In order to study the spectral profile of the three dye-labeled peptides, the detected spectral range was first specified by two popular dyes: fluorescein and Rox, then the spectra of the three dye-labeled peptides were examined. Figure 5.8 shows the image plots of the spectra, which is an application of Igor procedure 4 from Section 4.3.2. The image plots in Figure 5.8 b and c show that the FQ-bradykinin has a different spectral location than CBQ-bradykinin and CBQ-Ile-Ser-bradykinin; the spectrum of FQ shifts more to the red than CBQ.

In order to examine the spectral profiles more clearly, the corresponding 3-dimensional display of the image plots in Figures 5.8 b and c are produced in MATLAB (The MathWorks, Inc., Natick, MA, USA) and shown in Figures 5.9 a and b where the Z-axis is the light intensity; the X-axis is the spectral channel with channel 1 representing the redmost wavelength and channel 10 representing the bluemost wavelength; the Y-axis shows the frame sequence. The frame sequence 0 in Figure 5.9 a and b corresponds to the 400th frame in Figure 5.8 b and c, and sequence 400 in Figure 5.9 a and b corresponds to the 800th frame in Figure 5.8 b and c.

The view angles in Figure 5.9 have the same definition as that in Figure 3.17. The spectra in Figure 5.9 a and b are viewed at an Attitude (θ) of 50° and Direction (φ) of 75°, whereas the spectra in Figure 5.9 c and d are plotted at Attitude of 80° and Direction of 30°. The last signal in Figure 5.9 a is from the FQ-bradykinin and its enlarged mesh plot is shown in Figure 5.9 c at a different view angle (80°, 30°). The set of signals in Figure 5.9 b is from CBQ-Ile-Ser-bradykinin and its enlarged mesh plot is shown in Figure 5.9 d. Only 100 frame sequences are plotted in both Figures 5.9 c and d in order to show the spectra clearly.

The set of signals on the left side of Figure 5.9 a is from CBQ-bradykinin. Comparing this with the signals in Figure 5.9 b, we can see that the spectral profiles of CBQ-bradykinin and CBQ-Ile-Ser-bradykinin are similar. The two amino-acids difference between the two peptides does not change the emission profile of the labeled product.

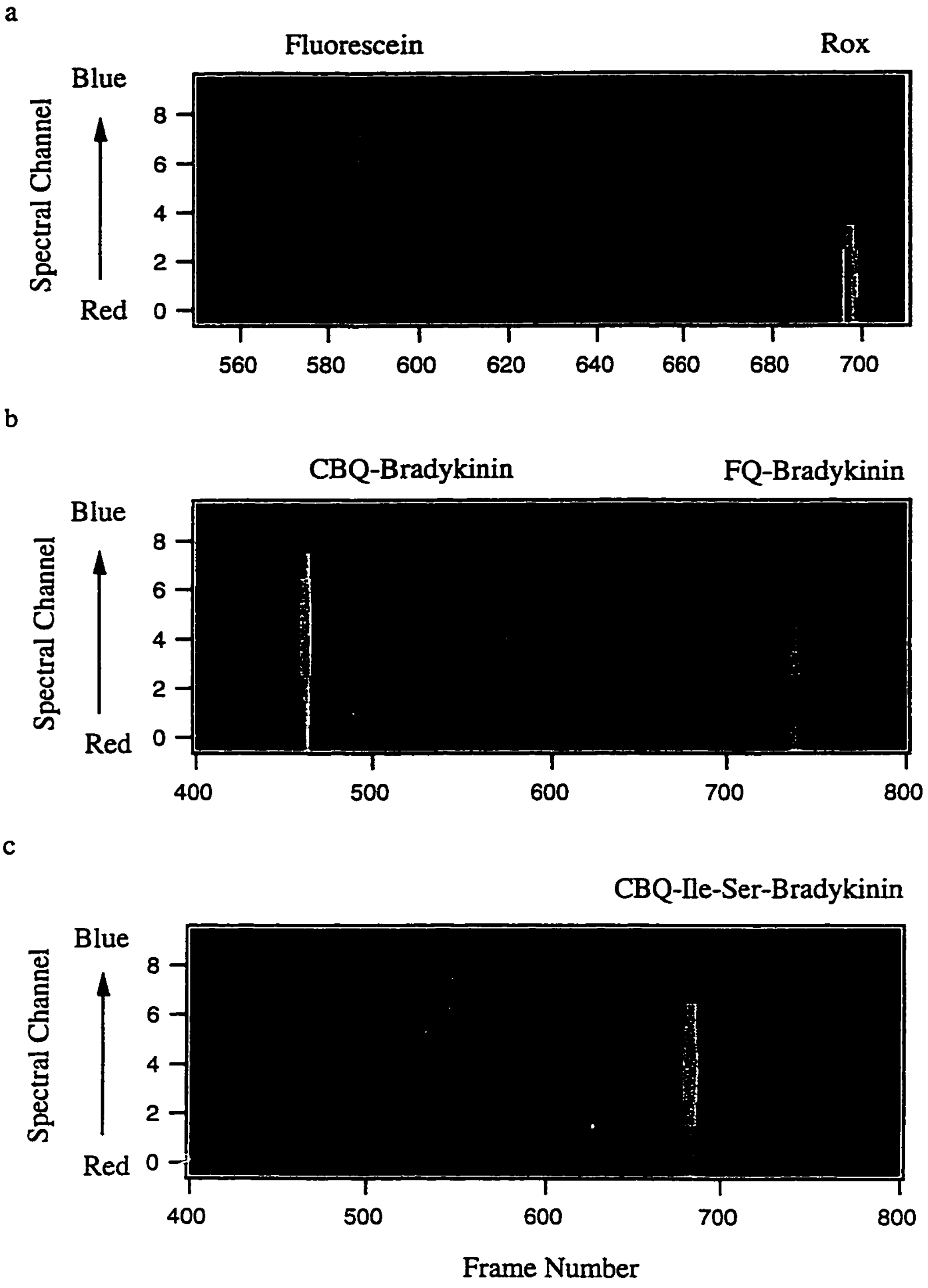


Figure 5.8 Image plots of spectra of fluorescein, Rox, and peptides
a) left: fluorescein, right: Rox
b) left: CBQ-bradykinin, right: FQ-bradykinin
c) CBQ-Ile-Ser-bradykinin

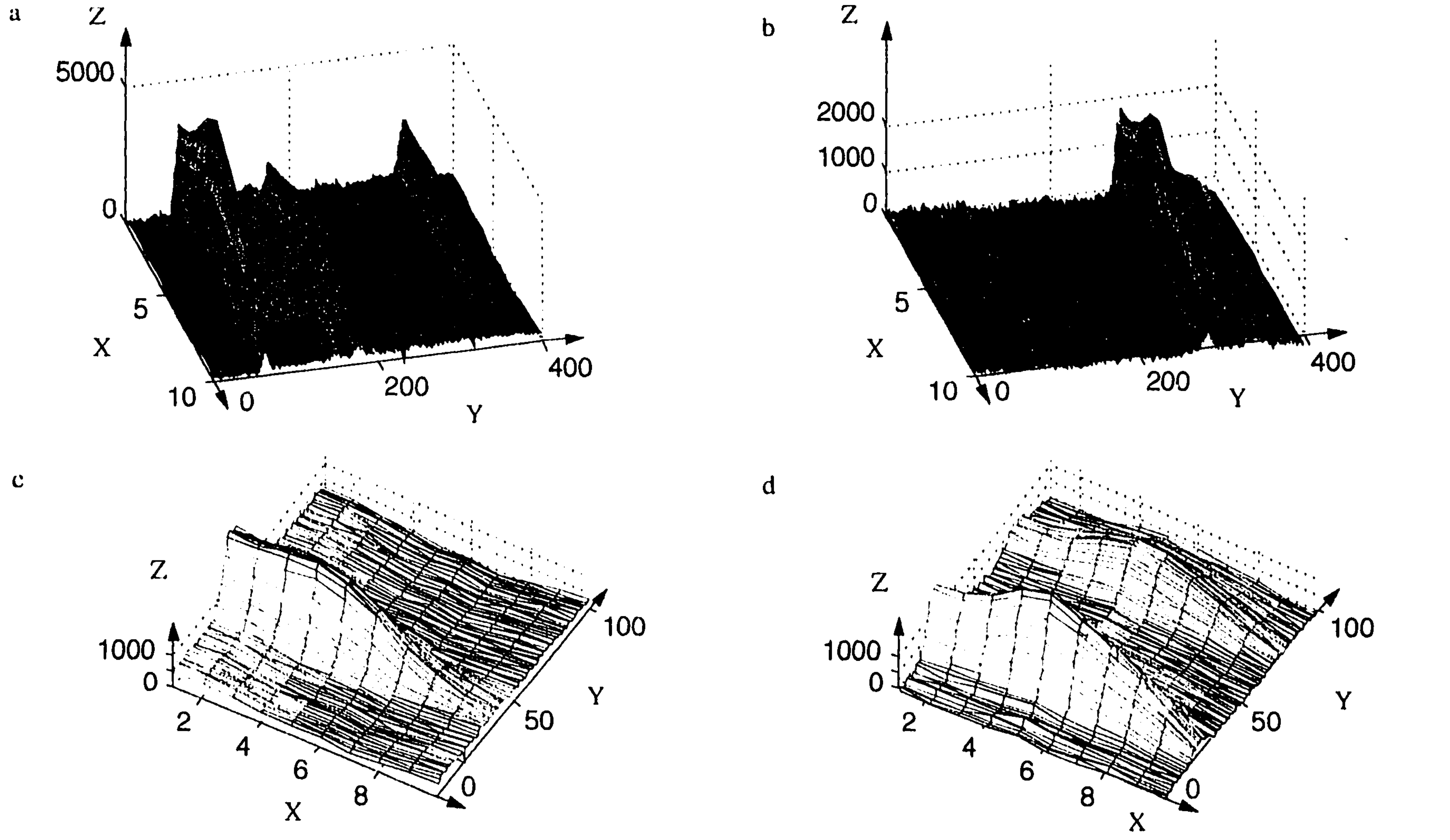


Figure 5.9 The 3D-view of the spectra of dye-labeled peptides at view angles of a,b) (50°, 75°) and c,d) (80°, 30°)
a) left: CBQ-bradykinin, right: FQ-bradykinin b) CBQ-Ile-Ser-bradykinin
c) enlargement of FQ-bradykinin d) enlargement of CBQ-Ile-Ser-bradykinin

The spectra of FQ and CBQ, shown in Figure 5.9 c and d respectively, are broad and overlap each other. The spectral shapes of the two dyes differ from each other. The spectrum of CBQ (Figure 5.9 d) is relatively more symmetric than the spectrum of FQ (Figure 5.9 c), and the maximum emission wavelength of CBQ is at the 5th spectral channel (w4). More spectral information in the red region will be needed in order to locate the maximum emission wavelength of FQ. However, its maximum emission does shift to red compared with CBQ.

From Figures 5.8 b and c we can see that the migration time of FQ-bradykinin is close to that of the small peak of CBQ-Ile-Ser-bradykinin. To examine the spectral resolving ability of the spectrometer, FQ-bradykinin and CBQ-Ile-Ser-bradykinin were mixed together and injected into one capillary for separation. The relative amounts of these two samples had to be adjusted to get a relative equal intensity of the fluorescence signal from the two dyes. The volume ratio of the labeling product of CBQ to that of FQ was 9:1 because the excitation of FQ is more favorable than CBQ at 488 nm excitation, since the optimized excitation wavelength of FQ is 486 nm and that of CBQ is 466 nm.[3b] The resultant electropherogram is shown in Figure 5.10.

Although the spectra of these two dyes overlap and the migration time of these two dye-labeled peptides are close to each other, the two peptides can still be distinguished by taking advantage of the spectral information provided by the 32-capillary spectrometer. As shown in Figure 5.10, spectral channel w1 and w2 (the 2nd and 3rd spectral channel in Figure 5.9) are binned together and plotted in red, while w6 and w7 (the 7th and 8th spectral channel in Figure 5.9) are binned together and plotted in blue. From Figure 5.9 c and d, we know that FQ has more spectral information in the red region than CBQ. This coincides with Figure 5.10: the red channel has a higher signal than the blue channel at the location of the electrophoretic peak of FQ-bradykinin, whereas both the red channel and the blue channel have the same signal level at the location of CBQ-Ile-Ser-bradykinin.

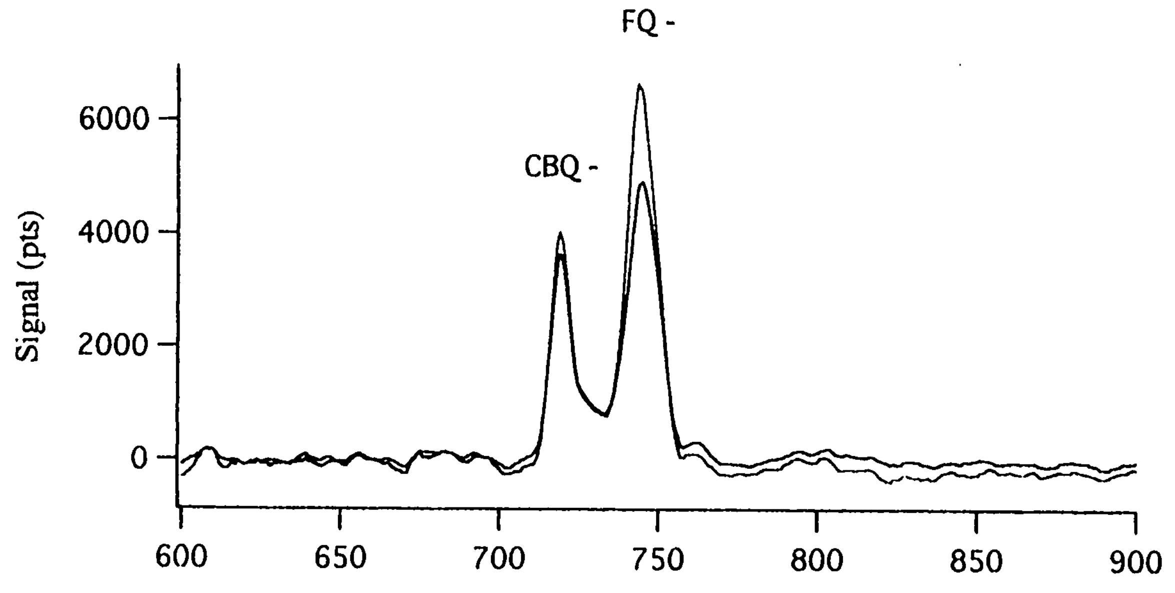


Figure 5.10 Spectral resolution for two dyes (red: w1+w2, blue: w6+w7)
sample: CBQ-Ile-Ser-bradykinin + FQ-bradykinin (9:1)
data shown has been smoothed by Savitzky-Golay function (2nd order 9 points)

The spectral information provided by the 32-capillary spectrometer can help to distinguish these two different dyes even though their spectra are broad and overlapping. This characteristic of the spectrometer can be helpful for peptide analysis. FQ-labeling is widely used in peptide and protein analysis by capillary electrophoresis with laser-induced fluorescence detection (CE-LIF).[10] In CE, identification of an unknown sample relies on the match of the migration time of a standard with that of an unknown, which is the same principle used in chromatography. Because the migration time in CE is not generally as reproducible as that in chromatography, it is not always accurate to use external standards to identify peptides. It is better to use internal standards with a different label from the unknown, such as CBQ-standard and FQ-unknown. The different spectral profile of the two dyes can be used to distinguish the standards from the unknowns. A calibration curve of the migration time of two different dye-labeled standards needs to be made, since the same peptide labeled with a different dye may have a different migration time.

In the actual application, FITC instead of CBQ may be used with FQ for labeling of the standards and unknowns because there is much less spectral overlap between FITC- and FQ-labeled products than between CBQ- and FQ-labeled products. The unlabelled FITC also fluoresces but only results in one separate fluorescence peak.

The above investigation of the dye-labeled peptide identification proves that the 32-capillary spectrometer provides an efficient and convenient method for peptide or protein separation and identification., which is an important CE research application.

Four-Color DNA Sequencing

The 32-capillary spectrometer is versatile in terms of application compared with the commercial slab gel DNA sequencer. It can be used for peptide or protein analysis with capillary zone electrophoresis as well as DNA analysis with capillary gel electrophoresis. An example of the application of four-color DNA sequencing is shown below.

Figure 5.11 shows a section of electropherograms from 16 capillaries. The DNA sequence of this section is GAGGATCCCCG from 50 to 60 bases. Four spectral channels are selected to represent the four terminated fragments, each of which is labeled with a different dye. Spectral channel w1 in red for Rox-labeled T-terminated fragments; w3 in yellow for Tamra-labeled G; w6 in green for Joe-labeled A; w8 in blue for Fam-labeled C. As can be seen from Figure 5.11, four different bases can be distinguished from each other by comparing the light intensity in each of the four spectral channels.

As discussed in Section 3.5.2, different dyes can be identified by the emission maximum as well as the spectral band location. Figure 5.12 confirms this statement by showing both the line plot of the four-color electropherogram and the image plot of the spectral location. The signal of T-terminated fragments in this sample is too low to be detected, leaving a space between base 54 A and 56 C. The line plot (Figure 5.11 a) and the image plot (Figure 5.11 b) confirms the DNA sequence.

5.4 CONCLUSIONS AND FUTURE DIRECTIONS

The 32-capillary spectrometer was successfully designed and developed. The advantage of this spectrometer is that it provides high sample throughput as well as spectral information such as emission profile. The evaluation of the performance of the instrument including LODs and applications were investigated. It was proven that the spectrometer may be used as a DNA sequencer as well as a peptide or protein analyzer. The optical design with the dispersing prism works well, as demonstrated by the data in Chapter 3 and this chapter, and has been successfully applied to our next generation instrument which has a 2-dimensional separation system.[11]

The optical detection and spectral dispersion system of the instrument have been fully developed. The separation system, however, can be further developed to make it more convenient. For example, a gel refill system[12] for refilling the sieving media in the 32 capillaries will speed up the DNA sequencing application, and an interface for fitting the

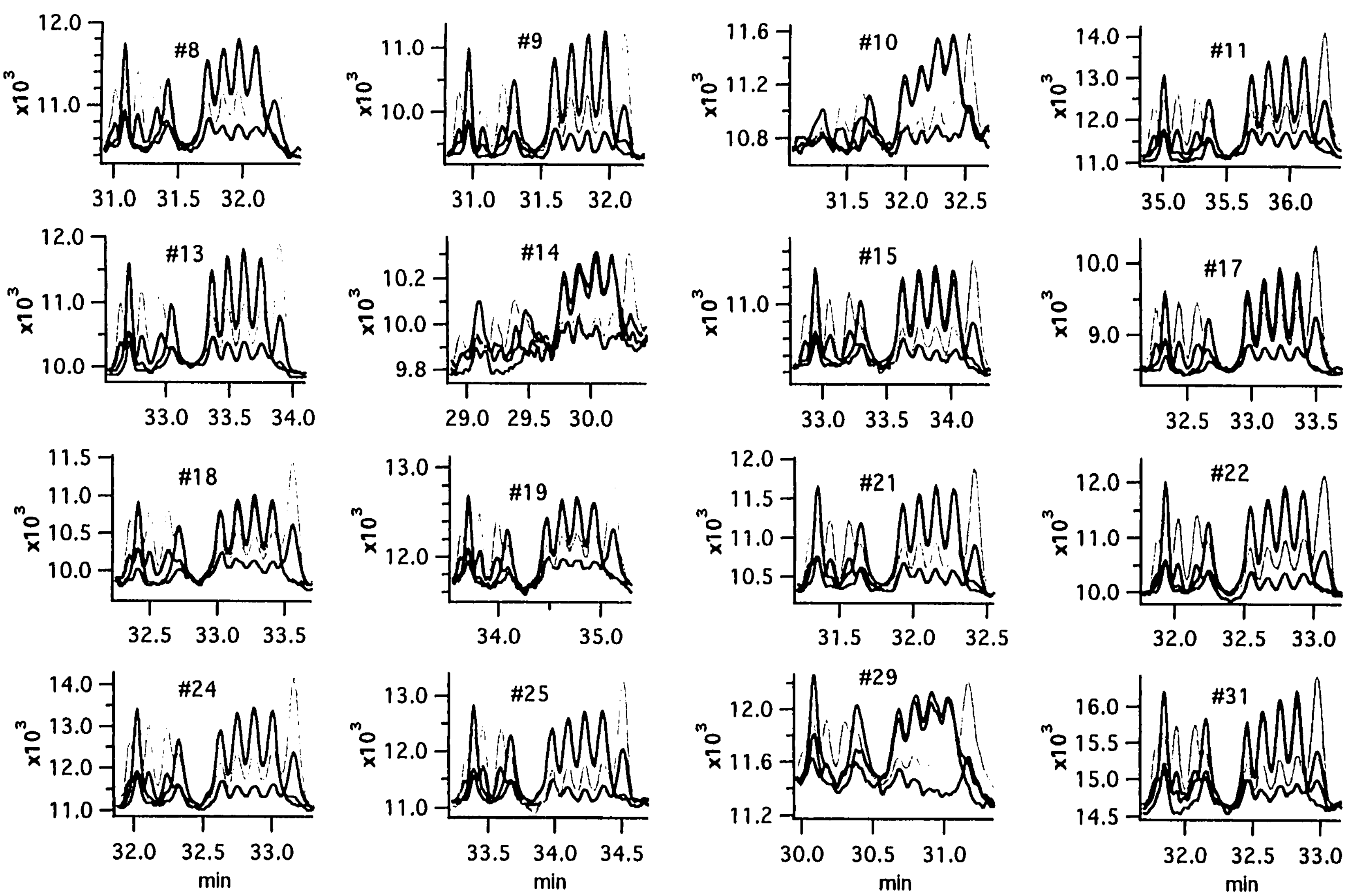


Figure 5.11 Four-color DNA sequencing with prism: electropherograms of 16 capillaries at room temperature

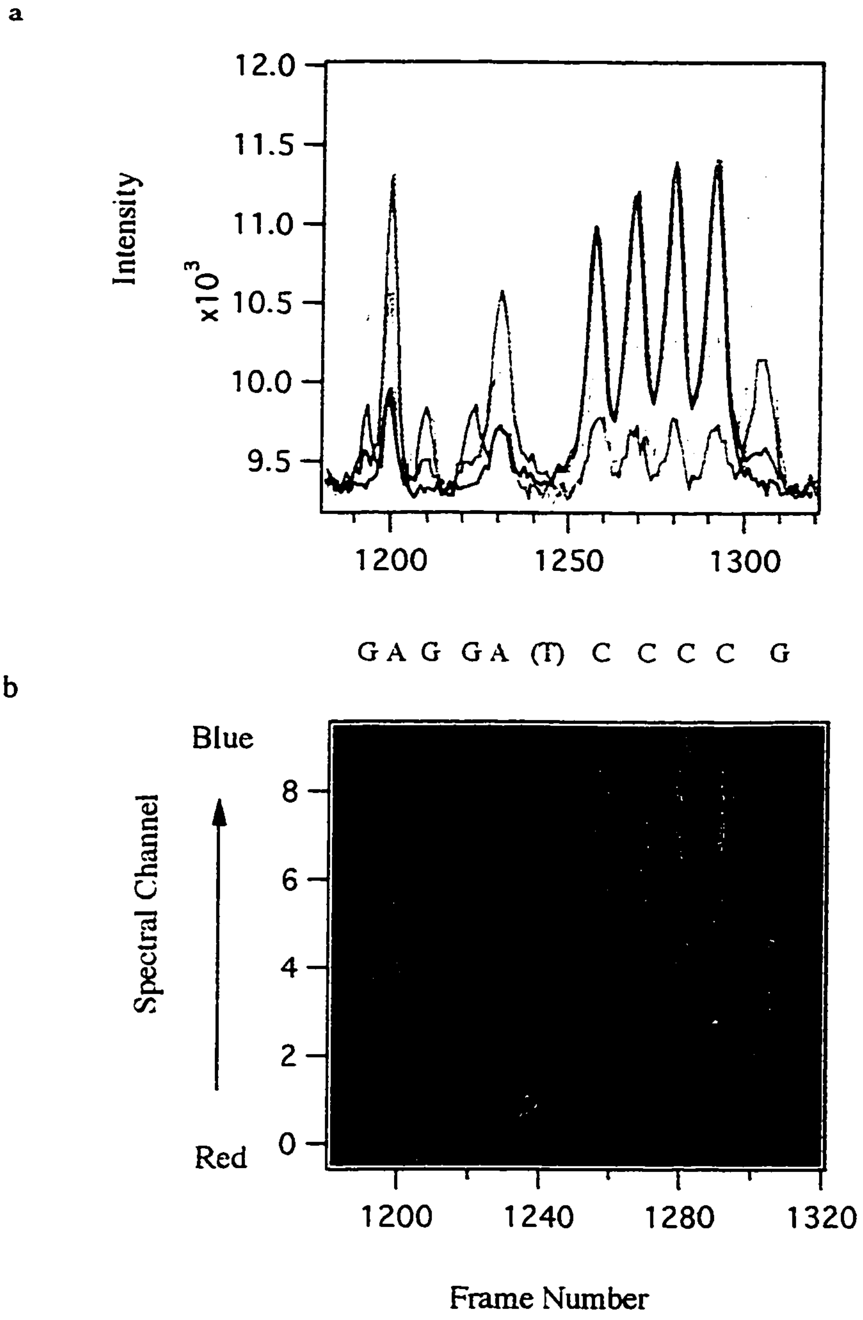


Figure 5.12 Four-color DNA sequencing from one capillary
a) electropherogram
T-Rox-red: w1 G-Tamra-yellow: w3
A-Joe-green: w6 C-Fam-blue: w8
b) image plot of the spectra

32 capillaries with a microtitre plate will need to be designed for automatic injection of multiple samples.

The data acquisition software can also be further developed to have more controls. such as the control of the high voltage power supply and the control of the laser power adjustment. The data display panel (Figure 4.5) supports the data display of one spectral channel from one user-selected capillary. It will be more convenient if data from multiple spectral channels from one or more capillaries can be displayed simultaneously.

REFERENCES

1. X. Huang, M. J. Gordon, and R. N. Zare, Anal. Chem., 1988, 60(4), 375
2. E. A. Arriaga, Y. Zhang, N. J. Dovichi, Analytica Chimica Acta, 1995. 299. 319-326
3. Product Catalogue, Molecular Probe, 1994, a) p85, b) p39
4. D. F. Lewis, Ph.D. thesis, Department of Chemistry, University of Alberta, Instrumentation for the Analysis of Biological Molecules, in progress.
5. Igor Pro 3 Programming and Reference Manual, 3 ed., WaveMetrics, Inc., Lake Oswego, OR 1996
6. A. Savitzky and M. J. E. Golay, Anal. Chem., 1964, 36, 1627-1639
7. K. Ueno, E. S. Yeung, Anal. Chem. 1994, 66, 1424-1431
8. M. A. Quesada, S. Zhang, Electrophoresis 1996, 17, 1841-1851
9. J. Z. Zhang, PhD thesis, Department of Chemistry, University of Alberta, DNA Sequencing by Single and Multiple CGEs, 1994, a) p113, b)p68
10. D. M. Pinto, E. A. Arriaga, D. Craig, J. Angelova, N. Sharma, H. Ahmadzadeh, and N. J. Dovichi, Anal. Chem.1997, 69, 3015-3021
11. J. Z. Zhang, N. J. Dovichi, US Patent 5,567,294,22 October 1996
12. S. Bay, PhD thesis, Department of Chemistry, University of Alberta, Construction and Evaluation of a Multiple Capillary DNA Sequencer, 1998

Chapter 6. Design of a Miniature Fluorescence Spectrometer

6.1 INTRODUCTION

Following the successful design of the 32-capillary fluorescence spectrometer, a design for a miniature fluorescence spectrometer is proposed. As can be seen from Chapters 3 and 5, a spectrometer provides useful spectral information which allows the distinguishing of different dyes through data analysis rather than a change of hardware (such as detection filter). This broadens the application of a spectrometer.

6.1.1 Miniature Spectrometer

High sample throughput is a major advantage of a multiple capillary instrument. On the other hand, large number of capillaries produce difficulty for the detection optics to efficiently collect the fluorescence from all the sample channels. For example, a relatively large prism and large lenses must be used in order to see all 32 capillaries. A 2-dimensional detector, such as a CCD camera, is also needed to record the spectral information from the 32-capillary array. These large detection optics make the instrument setup not portable.

A single capillary instrument provides a lower sample throughput. However, much smaller optics can be used to efficiently detect the fluorescence from a single capillary instead of multiple capillaries. The use of small optics make the instrument compact and portable, and this is the main advantage of a miniature spectrometer. The optical detecting system can be separated from the fluorescence collecting optics by using an optical fiber to transfer the fluorescence light from the cuvette to the spectrometer. The use of an optical fiber makes the instrument even more flexible and portable for outdoor monitoring. Also, detection of one spectrum requires only a one-dimensional detector such as a photodiode array which is less expensive than a CCD camera.

There is one miniature spectrometer from Ocean Optics, Inc. (Dunedin, Florida, USA) commercially available.[1] This miniature spectrometer measures the absorption spectrum of a solution. A light source of continuous wavelength is introduced to the solution through an optical fiber with a low N.A. The light beam passes through the solution, which absorbs some of the wavelengths, and enters the second optical fiber which has a high N.A. and is aligned with the first one. The second optical fiber leads the collected light to the detection optics, which is a set of mirrors, a grating and a photodiode array. The detection optics and the electronic circuits for data conversion are located in a compact box. The measured spectrum can be displayed by connecting the output of the detection circuitry to a computer. The absorbance and other information about the solution can also be displayed by using the software accompanying the miniature spectrometer. Comparing this miniature spectrometer with a conventional spectrophotometer, the miniature one is portable and easy to use, providing not only the absorbance but also the absorption spectrum.

A miniature fluorescence spectrometer would be useful for the monitoring of environmental pollutants and biological weapons. The object of this chapter is to describe the design of a miniature fluorescence spectrometer using the dispersion optics similar to those of the 32-capillary spectrometer. Before discussing this design, several optical pieces need to be introduced.

6.1.2 Optical Fiber

Optical fibers or fiber optics are experiencing greater use in spectroscopy to transfer light between various points. The reason for this increasing use is that optical fibers are mechanically flexible, which allows light to be transmitted over curved paths and long distances. This allows remote monitoring in hazardous environments since the more delicate components (for example, the detection system and signal processing electronics) can be far away from the monitoring site. Over long distances, the absorption and scattering losses in the fiber can be significant.

Light experiences total internal reflection inside the fiber. The material of the core in the center of the fiber has a higher refractive index than the material of the clad which surrounds the core. When light enters the fiber at certain angles, it experiences total internal refraction at the interface between the core and the clad, so light will propagate along the fiber in the core. The N.A. of an optical fiber represents the cone of light accepted by the fiber as shown in Figure 3.5 b, and it actually describes the field of view of the fiber. Core diameters are typically 50 to 600 μm and the refractive index of the clad is typically 0.02 to 0.2 less than the core. This gives a value for the N.A. in the range of 0.1 to 0.5.

6.1.3 GRIN Lens[2]

A gradient refractive index (GRIN) lens is different from a standard optical lens principally in the way it focuses light. A conventional lens consists of a homogeneous material with a single refractive index and focuses light due to the refraction at the curved surface(s) of the lens. A GRIN lens is made of a material with gradient refractive index, and it focuses light by a continuous refraction of light inside the lens.

A GRIN lens is usually made in a cylindrical shape, and the refractive index in the cylinder varies continuously as a function of the radius about the central axis. In the center of the lens, the refractive index is the highest, and it gradually decreases along the lens radius, and reaches the lowest value at the edge of the lens. The continuous refraction of light inside the lens and the discrete refraction at the lens/air interface cause light to travel through the lens in a sinusoidal fashion.[2]

An important parameter of a GRIN lens is the pitch number, one pitch is defined as the length of the lens required for light of a certain wavelength to finish one sine wave cycle in its path. Therefore, a GRIN lens of 0.25 pitch or 0.75 pitch is able to transform a collimated beam into a point spot and visa versa, so this type of GRIN lens is often used to convert a large image into a smaller one in cases when it is coupled with an optical fiber.

6.1.4 Focal Length of Conventional Lenses

The focal length of a conventional lens can be calculated from the lens shape and the index of refraction of the lens material.[3] According to the Newtonian form of "the lens formula", the focal length of a lens in air can be expressed as:

$$\frac{1}{f} = (n - 1)\left(\frac{1}{r_1} - \frac{1}{r_2}\right) + \frac{(n - 1)^2}{n}\frac{t_c}{r_1 r_2} \qquad (6\text{-}1)$$

where f is the focal length, n is the refractive index, r_1 and r_2 are the radii of the two surfaces, and t_c is the center thickness. For a thin lens, $t_c \equiv 0$, and for a plano lens either r_1 or r_2 is infinite. In either case the second term of the above equation vanishes, and the familiar Lens Maker's Formula is left as follows:

$$\frac{1}{f} = (n - 1)\left(\frac{1}{r_1} - \frac{1}{r_2}\right) \qquad (6\text{-}2)$$

for a plano-convex lens, r_2 is infinite, so

$$f = \frac{r_1}{n - 1} = \frac{1}{(n - 1)\, c_1} \qquad (6\text{-}3)$$

where c_1 is the curvature of the curved surface, and

$$c_1 = \frac{1}{r_1} \qquad (6\text{-}4)$$

for a symmetric lens, $r_1 = - r_2$. With center thickness constrained, the following equation can be derived from equation (6-1):

$$r_1 = (n - 1)\, f\, [\, 1 + (1 - \frac{t_c}{nf})^{\frac{1}{2}}\,] \qquad (6\text{-}5)$$

equation (6-5) can be rewrite to:

$$\frac{1}{f} = \frac{n - 1}{r_1}\, [\, 1 + (1 - \frac{t_c}{nf})^{\frac{1}{2}}\,] \qquad (6\text{-}6)$$

6.2 THE MINIATURE FLUORESCENCE SPECTROMETER

To achieve the flexibility in the light collection and light detection, an optical fiber can be used to transfer fluorescence to the detection optics. A similar design of the dispersion optics of 32-capillary spectrometer can be used in the miniature one for spectral dispersion. Figure 6.1 shows a general design of the miniature spectrometer.

Fluorescence from the capillary stream is collected by a lens labeled (1) in Figure 6.1 and imaged onto a GRIN lens (2), which is connected to the inlet of the optical fiber (3). The GRIN lens (2) focuses the fluorescence image from the lens (1) further into a small image which fits the core of the optical fiber. The light then travels along the optical fiber (3). Another GRIN lens (4) is connected to the outlet of the optical fiber and collimates the fluorescence. The parallel light is then color-dispersed by a prism (5), and the spectrum is imaged onto a photodiode array (7) by an imaging lens (6).

The optics can be divided into four parts: fluorescence collection, light transfer, color dispersion and spectral detection. Each part will be discussed in the following sections.

6.2.1 Fluorescence Excitation and Collection

The flow chamber could be a sheath-flow cuvette with a single capillary for the introduction of the sample solution. A syringe pump in the case of cytometry or an electric field in the case of capillary electrophoresis can be used to inject sample solution into the capillary. A laser can be introduced into the flow chamber through an optical fiber with a low N.A., exciting fluorescence just below the capillary tip. The fluorescence is then collected at a right angle to the laser beam.

There are two ways to get the fluorescence into the optical fiber. One is to use a lens (1) to collect the fluorescence and image it onto the fiber, as shown in Figure 6.1. A high magnification microscope objective can serve as this lens and collect the fluorescence efficiently. A 0.25 pitch or a 0.75 pitch GRIN lens (2) can be coupled to the optical

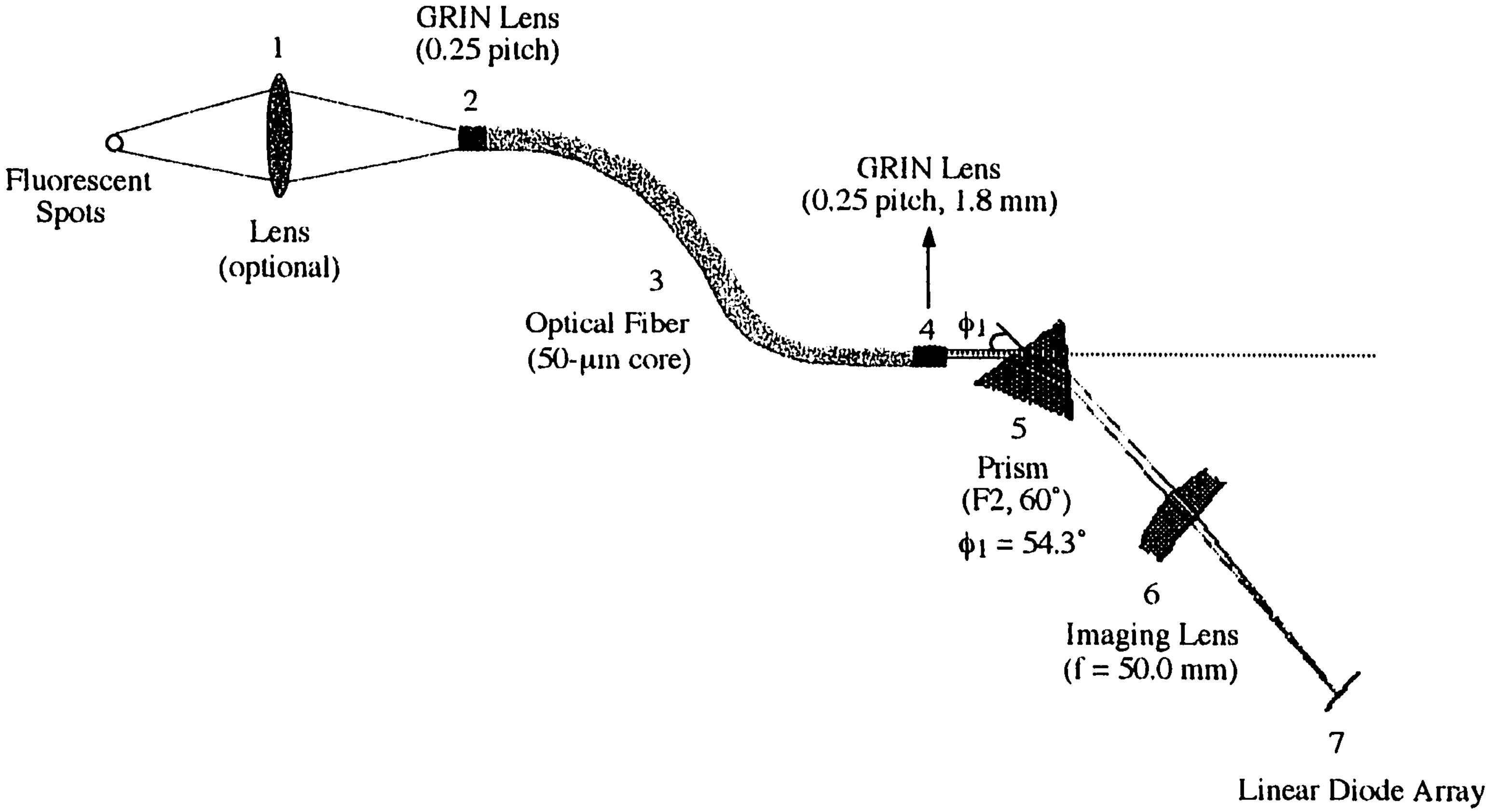


Figure 6.1 Design of the miniature fluorescence spectrometer

fiber (3) in order to focus the relatively large image produced by the objective further into a small image, which fits the size of the core of the optical fiber.

Another way to collect the fluorescence is to directly couple the light into the optical fiber without using the lens (1) in Figure 6.1. Unlike a conventional lens, the N.A. of an optical fiber is not an alternative term for the light collecting power of the fiber from a localized region. Therefore, in this case a fiber with high N.A. does not help much in improving the efficiency of light collection from a fluorescent spot. In order to collect as much light as possible, the optical fiber (3) has to be placed as close as possible to the fluorescent spot. Adding a 0.25 pitch GRIN lens (2) to the fiber will also help to collect fluorescence more efficiently.

6.2.2 Light Transfer

The collected fluorescence light experiences total internal reflection inside the optical fiber. The core size of the fiber can be 50 μm, since the ID of a standard capillary is 50 μm. The outlet of the optical fiber is a point light source, and the fluorescence light has to be collimated in order that a prism can be used for color dispersion. A GRIN lens with a 0.25 pitch 1.8 mm in diameter (4 in Figure 6.1) can be coupled to the outlet of the optical fiber for this purpose.

6.2.3 Color Dispersing

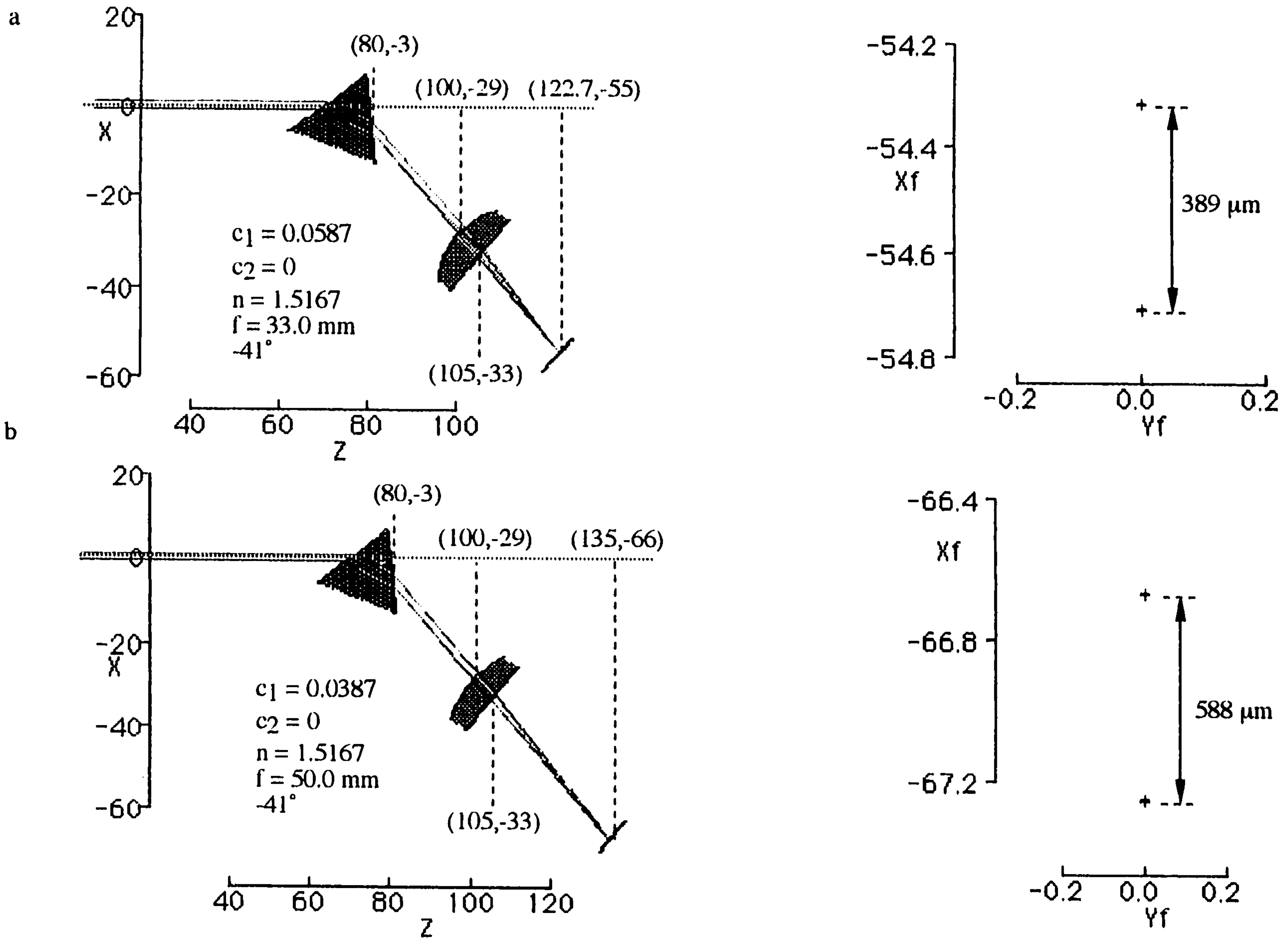
Basically, the same approach as in the 32-capillary spectrometer is applied to the miniature spectrometer for color dispersion. The fluorescence light is collimated first, and then goes through a F2 equilateral prism which is operated at the minimum deviation condition. The prism disperses the spectrum according to wavelength. The color-dispersed light is then imaged by an imaging lens onto a photo-detector.

The miniature spectrometer has only one sample channel, which makes the detection optics much smaller compared with those in the 32-capillary spectrometer. As shown in Figure 6.1, rather than two camera lenses, a GRIN lens (4) can serve as the collimating lens and a microscope objective (6) can be used as the imaging lens. A diode array (7) rather than a CCD camera is good enough for spectral recording, and the size of the prism (5) can be much smaller as well.

As discussed in Section 3.4.2, final dispersion depends on the location, as well as the shape and the material of the imaging lens. Here the last two effects are investigated with the ray tracing program Beam Three.

Figure 6.2 shows the results of the modeling. The labels on the graphs have similar meanings to those in Figure 3.9. Additional information about the imaging lens (c_1, c_2, n, f, and pitch angle) are added for each diagram. "c_1" is the curvature of the first surface of the lens; "c_2" is the curvature of the second surface; "n" is the refractive index of the lens material; "f" is the focal length of the lens; the pitch angle of the lens surface (-41°) is the angle at which the surface intercepts with the XY-plane. Two collimated rays 1.8 mm apart are used as the incident light to the prism in order to model the optical conditions after the GRIN lens (4) in Figure 6.1. Each of the incident rays contains two wavelengths: a 632.8 nm red and a 532 nm blue. In all cases of Figure 6.2 a to d, the prism and the imaging lens are placed at exactly the same locations and orientations, the only difference is that the curvature, the lens type, or the refractive index of the imaging lens differs from one case to another.

Figure 6.2 a and b shows the effect of lens curvature on dispersion. A lens with less curvature, and therefore a longer focal length, produces a larger dispersion. A plano-convex lens made of the same material is used in both cases. To the right of each optical layout, the resultant final dispersion on the detector plane is shown. In the approximately 100 nm spectral range from 532 nm to 632.8 nm, the lens with 0.0587 curvature gives a final dispersion of 389 μm (Figure 6.2 a), whereas a curvature of 0.0387 gives a 588 μm dispersion in Figure 6.2 b. The focal length (f) is calculated from the curvature (c_1) and refractive index (n) of each lens using equation (6-3). The imaging lens in Figure 6.2 b


a
20
0
X
-20
-40
-60
(80,-3)
(100,-29)
(122.7,-55)
(105,-33)
$c_1 = 0.0587$
$c_2 = 0$
n = 1.5167
f = 33.0 mm
-41°
40
60
80
100
Z
-54.2
-54.4
Xf
-54.6
-54.8
389 μm
-0.2
0.0
0.2
Yf
b
20
0
-20
X
-40
-60
(80,-3)
(100,-29)
(135,-66)
(105,-33)
$c_1 = 0.0387$
$c_2 = 0$
n = 1.5167
f = 50.0 mm
-41°
40
60
80
100
120
Z
-66.4
Xf
-66.8
-67.2
588 μm
-0.2
0.0
0.2
Yf

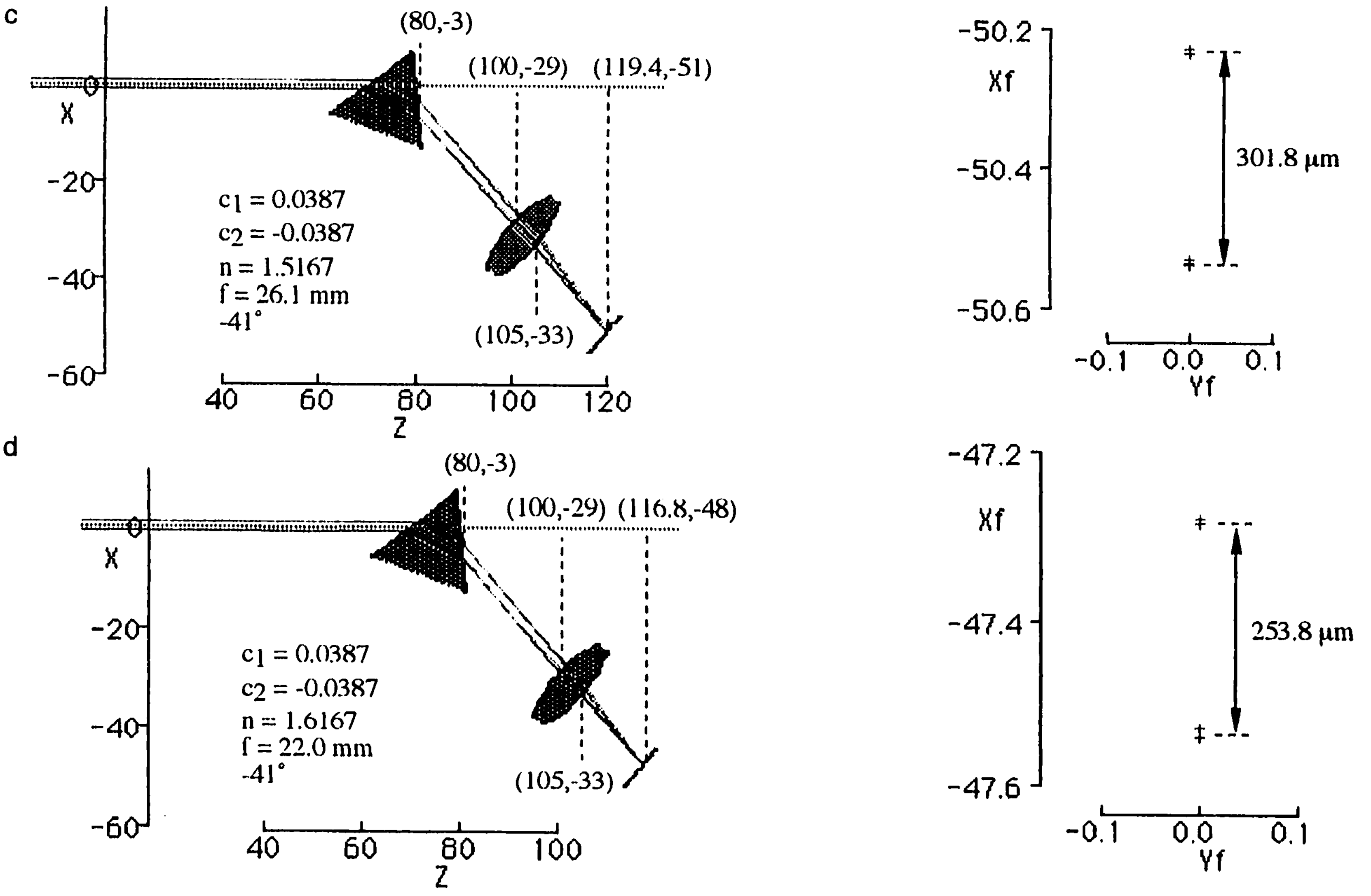


Figure 6.2 Effect of curvature (a,b), lens type (b,c), reflective index (c,d) of the imaging lens on dispersion prism: F2, 60° at minimum deviation condition red: 632.8 nm blue: 532.0 nm

has a focal length of 50 mm and gives 51% more dispersion than the lens with the 33 mm focal length in Figure 6.2 a.

The effect of the lens type on dispersion is shown in Figure 6.2 b and c. The imaging lens in Figure 6.2 b is a plano-convex lens, while the lens in Figure 6.2 c is a symmetric lens. Both lenses have the same refractive index and curvature. The focal length of the symmetric lens was calculated using equation (6-6) and is given on the diagram. Comparing Figure 6.2 b with Figure 6.2 c, we can see that the different lens type gives a different focal length, and therefore results in a different dispersion. The symmetrical lens has a shorter focal length (f = 26.1 mm) than the plano-convex lens (f = 50 mm), and results in less final dispersion (301.8 μm) than the plano-convex lens (588 μm) over the 100 nm spectral range.

The lens material and therefore the refractive index also affects the final dispersion. As shown in Figure 6.2 c and d, the lens in Figure 6.2 d has a higher refractive index of 1.6167 than the 1.5167 refractive index in Figure 6.2 c. According to equation (6-6), the higher refractive index results in a smaller focal length of 22.0 mm in Figure 6.2 c, which in turn gives a smaller final dispersion of 253.8 μm.

The Lens Maker's Formula (equation 6-2) shows that the lens material and the lens shape define the focal length of a lens. The above ray-tracing results indicate that the lens with a longer focal length gives a larger final dispersion for the spectrometer. The resultant dispersion in the spectral range from 532 nm to 632.8 nm for each case in Figure 6.2 is listed in Table 6.1.

If a lens with a 50 mm focal length is used as the imaging lens (6) in the miniature spectrometer shown in Figure 6.1, an approximate 590 μm final dispersion should be obtained. However, the above ray-tracing is done with a simple lens, in the case of a complex lens the resulting dispersion may be a little different. For instance, the plano-convex lens and the camera lens in Figure 3.13 have the same focal length but result in different final dispersions.

Table 6.1 Summary of the ray-tracing results of Figure 6.2

	a	b	c	d
f (mm)	33.0	50.0	26.1	22.0
final dispersion (μm)	389	588	301.8	253.8

6.2.4 Spectral Detection

A one-dimensional photodiode array can be used as the detector. Since the fluorescent spot is 50 μm, a diode array with an element size of 50 μm is the choice for use. The EG&G RETICON's catalogue[4] lists a relatively inexpensive linear diode array which belongs to the linear A series and has 64 elements of a 50 x 50 μm size. For a 590 μm dispersion between 532 nm and 632.8 nm, the spectrum will cover approximately 12 elements of this diode array.

6.2.5 Conclusion

The optical system of a relatively inexpensive miniature fluorescence spectrometer has been designed. As shown in Figure 6.1, the fluorescence is collected by a microscope objective (1) and imaged onto a GRIN lens (2). The 0.25 pitch GRIN lens (2) is connected to an optical fiber (3) with a 50 μm core size, and focuses the fluorescence image further into a smaller image which fits the optical fiber. The fluorescence light passes through the fiber and then is collimated by a 0.25 or 0.75 pitch GRIN lens 1.8 mm in diameter (4). The collimated light is then dispersed by a 20 mm F2 equilateral prism (5), which is operated at its minimum deviation condition (a 54.3° incident angle at 546.1 nm light). The color dispersed light is then focused by a lens (6) with a focal length of 50 mm which images the spectrum onto a 64-element linear A series photodiode array (7). With this design, the 100 nm spectrum from 532.0 nm to 632.8 nm will cover approximately 12 elements of the diode array detector.

REFERENCES

1. Ocean Optics, Inc., Dunedin, Florida, http://www.OceanOptics.com
2. SELFOC Product Guide SPG0295, NSG America, Inc., Somerset, New Jersey, 1995 p28
3. Optical Guide 4, MELLES GRIOT, Ontario, Canada, 1-5
4. Product Summary, EG&G RETICON, Sunnyvale, California

Appendix I. Data Acquisition Software

Part I. SOURCE CODE OF "CAM"

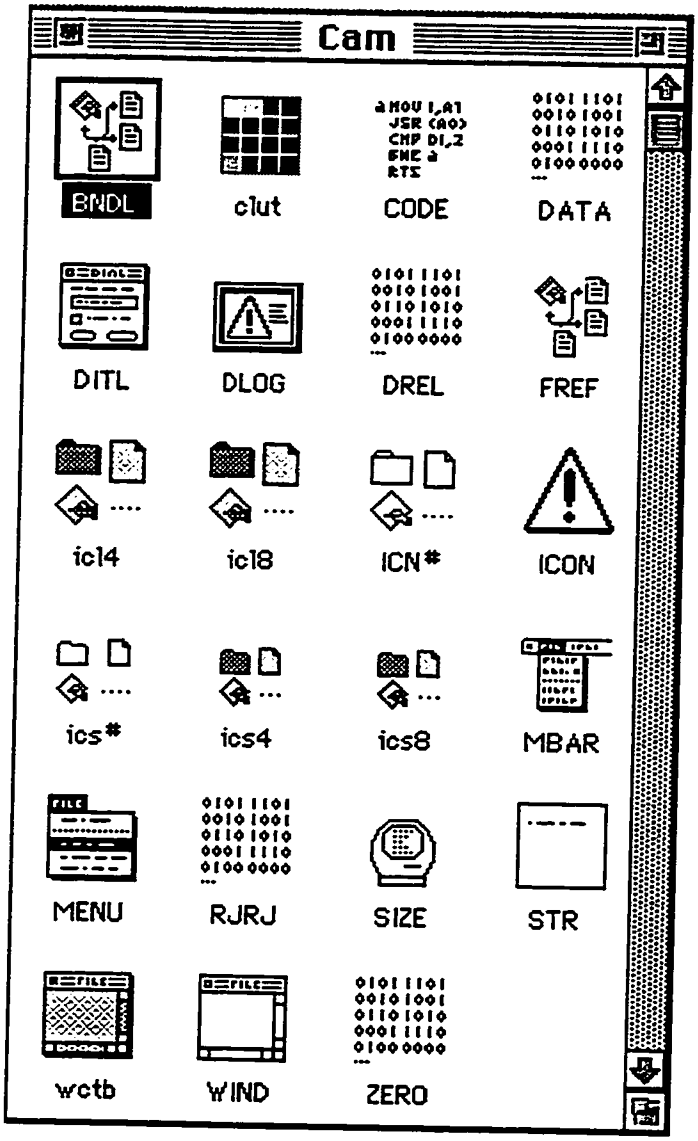
I-1. Header File:

ccd.h

```
#include "master.h"
#include "pvcam.h"
#include <stdio.h>
#include <Dialogs.h>
#include <Files.h>
#include <Folders.h>          /* for the FindFolder function */
#include <OSUtils.h>
#include <Windows.h>

#define       kcamErrDialogID     128
#define       kOSErrDialogID      129
#define       kDialogOK           1
#define       kDataDisWinID       128
#define       kpixImageWinID      129
#define       kcapIconWinID       130
#define       kbackgroundID       131
#define       kMenuBarID          128
#define       kFileMenu           128
#define       kExposureMenu       129
#define       kFileQuit           1
#define       kExposure1          1
#define       kExposure2          2
#define       kExposure3          3
#define       kExposure4          4
#define       kGreyClutID         128
#define       kColorClutID        129
#define       kCapIconID          131
#define       kzoomInIconID       132
```

```
#define     kshrinkIconID       133
#define     kUpIconID           134
#define     kDownIconID         135

void        errCam              (int16 hcam);
int16       initCam             (void);
void        uninitCam           (int16 hcam);
void        dataAcquisition     (int16 hcam);
void        osError             (OSErr error, Str255 message);
void        createFile          (SFReply reply);
int16       openFile            (SFReply reply);
void        writeFile           (int16 reNum, SFReply reply, long count,
                                 Ptr dataBuf);
void        closeFile           (int16 refNum, SFReply reply);
void        displayData         (int16 datapoint, uns16 dataValue);
void        doEvent             (EventRecord *event, int *change_exp,
                                 float *scalingFactor, int *baseShift,
                                 int *displayCap);
int         doMenu              (long selectmenu);
Boolean     shtrSetup           (int16 hcam, uns16 Hz);
rgn_ptr     exposureRgn         (void);
rgn_ptr     focusRgn            (int16 hcam);
void        setPixelMap         (PixMapHandle pixMap, short pmWidth,
                                 short pmHeight, CTabHandle myCTable,
                                 uns8_ptr data8);
Boolean     displayImage        (int16 hcam);
Boolean     make8Bits           (uns32 total_pix, uns8_ptr data8, uns16_ptr data16);
void        releaseDataMem      (uns8_ptr data8, uns16_ptr data16);
void        releasePixMem       (PixMapHandle pixMap,CTabHandle myCTable);
void        handleMouseDown     (EventRecord *eventPtr, int *mouseCode);
void        doEvent2            (EventRecord *eventPtr, int *mouseCode,
                                 int *keyCode);
```

I-2. Resource Files:



I-3. Function Files:

1. main.c
2. initialization.c
3. showImage.c
4. dataAcquisition.c
5. dataFile.c
6. event.c
7. menu.c
8. showerr.c

1. main.c

```
#include "ccd.h"

main()
{       int16   Hcam;
        long    mem;
        InitGraf (&qd.thePort);
        InitWindows ();
        FlushEvents (everyEvent,0);
        InitMenus ();
        InitDialogs (0L);
        InitCursor ();

        Hcam = initCam ();
        if ( pl_cam_check (Hcam) )
        {
                if (displayImage (Hcam))
                {       dataAcquisition (Hcam); }
                pl_cam_close (Hcam);
        }
        else
        {       SysBeep(30);
                SysBeep(30);
        }
}
```

2. initialization.c

```
/* initilize and unintialize the library and hardware (the camera) for expoure
 * function prototypes:          int16    initCam (void)
 *                               void uninitCam (int16 hcam)
 */

#include "ccd.h"

int16   initCam (void)
{
        int16           hcam ;
        char            cam_name[CAM_NAME_LEN];
        Boolean         error;
        int16           err = 0;
        char            msg[ERROR_MSG_LEN];
        int16           total;
        //char_ptr      chip_name;
        int16           tmp, tmp_hi, tmp_lo;
        int16           tmp_set = -2700;

/*initialize library */
        if ( !pl_pvcam_init() )
        {       errCam(hcam);
                return (int16)800;
        }

/* initialize the buffer function */
        if ( !pl_buf_init() )
        {       errCam(hcam);
                return (int16)800;
        }

/* initialize the data acquisition functions */
        if ( !pl_exp_init_seq() )
        {       errCam(hcam);
                return (int16)800;
        }

/* total number of cameras in the system */
        if ( !pl_cam_get_total(&total))
        {       errCam(hcam);
                return (int16)800;
        }

/* camera name (PXL) */
        if ( !pl_cam_get_name(0, cam_name))
        {       errCam(hcam);
                return (int16)800;
        }

/* open PXL */
        if ( !pl_cam_open(cam_name, &hcam, OPEN_EXCLUSIVE))
        {       errCam(hcam);
                return (int16)2000;
        }
/* diagnose system and subsystem */
        if ( !pl_cam_get_diags(hcam))
        {       errCam(hcam);
                return hcam;
        }
/* set up camera for exposure: temperature, clear_cycle, shutter, pmode */
        if ( !pl_ccd_set_tmp_setpoint(hcam,tmp_set))          /* set camera temperature */
        {       errCam(hcam);
                return hcam;
        }
        if ( !pl_ccd_set_clear_cycles(hcam,5))          /* clear 5 times */
        {       errCam(hcam);
                return hcam;
        }
        if ( !pl_ccd_set_clear_mode (hcam, CLEAR_PRE_EXPOSURE))
        {       errCam(hcam);
                return hcam;
        }
        if ( !pl_shtr_set_open_mode (hcam, OPEN_PRE_EXPOSURE))
        {       errCam(hcam);
                return hcam;
        }
        if ( !pl_shtr_set_open_dly (hcam, 0))
        {       errCam(hcam);
                return hcam;
        }
        if ( !pl_shtr_set_close_dly (hcam,0))
        {       errCam(hcam);
                return hcam;
        }
        if ( !pl_ccd_set_pmode (hcam,PMODE_MPP))
        {       errCam(hcam);
                return hcam;
        }
        return hcam;
}



void uninitCam (int16 hcam)
{       pl_cam_close (hcam);

/* uninitialize the buffer, data acquisition functions and library */
        if ( !pl_buf_uninit() )
        {       errCam(hcam);
        }
        if ( !pl_exp_uninit_seq() )
        {       errCam(hcam);
        }
        if ( !pl_pvcam_uninit() )
        {       errCam(hcam);
        }
}
```

3. showImage.c

```
/* function prototypes handleing the alignment panel:
*       Boolean     displayImage (int16 hcam)
*       void        releaseDataMem (uns8_ptr data8, uns16_ptr data16)
*       void        releasePixMem (PixMapHandle pixMap, CTabHandle myCTable)
*       Boolean     make8Bits (uns32 total_pix, uns8_ptr data8, uns16_ptr data16)
*       void        setPixelMap (PixMapHandle  pixMap, short pmWidth,
*                           short pmHeight, CTabHandle cTable, uns8_ptr data8)
*/

#include "ccd.h"

Rect            gdataRect[32];
rgn_ptr         gexp_rgn = nil;

Boolean displayImage (int16 hcam)
{
        long    fmem;
        OSErr               memErr, error;
        Str255              message = "\pOutOfMem";
        uns16               exp_ser_size, exp_par_size;
        uns32               exp_pix;
        uns16               exp_total = 1, rgn_total = 1;
        uns32               exp_bytes;
        uns8_ptr            data8 = nil;
        uns16_ptr           data16 = nil;
```

```
CTabHandle          myCTable = nil;
PixMapHandle        pixMap = nil;
EventRecord         event;
Rect                clipRect;
RgnHandle           savedClip = nil, outlineRgn, tempRgn;
GrafPtr                     savedPort = nil;
WindowPtr           background;
Rect                capRect[32], imageRectG, stringRect[32];
Handle              capIcon[32];
Str255              capNum;
Point               mouseLoc;
long                top, bottom, left, right;
StringHandle        topHdl[32], bottomHdl[32], leftHdl[32], rightHdl[32];
int                 labelh, labelv, xstart, ystart, j,i,xfinish, yfinish ;
int                 mouseCode = 100, keyCode = 0;
Boolean             go;
Boolean             inDrag = false;
short               activeRect = -1;
int16               status;
uns32               byte_cnt;
Boolean             quit;
uns32               Exp_time;

quit = false;
go = false;
Exp_time = 200;         /* set default exp_time */
fmem = FreeMem();
gexp_rgn =(rgn_ptr)NewPtr ((Size)2*sizeof(rgn_type));
memErr = MemError();
if (memErr != noErr)
{       osError( memErr, message);
        return false;
}

/* define rgn on ccd chip */
gexp_rgn[0].s1 = 0;
```

```
        gexp_rgn[0].p1 = 0;
        gexp_rgn[0].s2 = 1100;
        gexp_rgn[0].p2 = 400;
        gexp_rgn[0].sbin = 1;
        gexp_rgn[0].pbin = 1;

        if (!pl_exp_setup_seq (hcam,exp_total, rgn_total, gexp_rgn,
                                TIMED_MODE,Exp_time, &exp_bytes) )
        {       errCam (hcam);
                DisposePtr ((Ptr)gexp_rgn);
                return false;
        }

/* calculate total number of pixels used in one exposure */
        exp_ser_size = (uns16)( (gexp_rgn->s2 - gexp_rgn->s1 +1)/gexp_rgn->sbin );
        exp_par_size = (uns16)( (gexp_rgn->p2 - gexp_rgn->p1 +1)/gexp_rgn->pbin );
        exp_pix = (uns32)(exp_ser_size) * (uns32)(exp_par_size);

/* allocate memory for storage of auto-scaled image */
        data8 = (uns8_ptr) NewPtr((Size)(exp_pix));
        memErr = MemError();
        if (memErr != noErr)
        {       osError( memErr, message);
                DisposePtr ((Ptr)gexp_rgn);
                return false;
        }

/* allocate memory for pixel_stream (data), each pixels takes 2 bytes *
 *      exp_bytes = exp_pix * (uns32)2; this value is a return in setup_seq */

        data16 = (uns16_ptr) NewPtr ((Size)exp_bytes);
        memErr = MemError();
        if (memErr != noErr)
        {       osError( memErr, message);
                DisposePtr ((Ptr)gexp_rgn);
                DisposePtr ((Ptr)data8);
                return false;
        }

/* get CTable from resourse, move it Hi in the heap, lock it */
        myCTable = GetCTable (kColorClutID);
        if (myCTable == nil)
        {
                printf (" cannot find CTable from resourse\nclick to go on\n");
                while(!Button())
                {}
                DisposePtr ((Ptr)gexp_rgn);
                releaseDataMem (data8, data16);
                return false;
        }
        MoveHHi ((Handle)myCTable);
        HLock ((Handle)myCTable);

/* allocate pixel map */
        pixMap = NewPixMap();
        memErr = MemError();
        if (memErr != 0)
        {       osError (memErr, message);
                DisposePtr ((Ptr)gexp_rgn);
                releaseDataMem (data8, data16);
                HUnlock((Handle)myCTable);
                myCTable = nil;
                return false;
        }
        MoveHHi ((Handle)pixMap);
        HLock ((Handle)pixMap);

/* setup pix map, make it point to the scaled data from exposure (data8) */
        setPixelMap (pixMap, exp_ser_size, exp_par_size, myCTable, data8);
        background = GetNewWindow (kbackgroundID, 0, (WindowPtr)-1L);
        ShowWindow (background);
        SetPort (background);
```

```
/*Set rectangles for the 32 capillary icons*/
        for(i = 0; i < 32; i++)
        {
                if (i <16)
                {
                        SetRect (&capRect[i], (short)(i*40+55), (short)505,
                                        (short)(i*40+85), (short)535);
                }
                else
                {       SetRect (&capRect[i], (short)((i-16)*40+55), (short)565,
                                        (short)((i-16)*40+85), (short)595);
                }
        }

/*Draw the capillary icons in the panel window*/
        for(i = 0; i < 32; i++)
                {
                        capIcon[i] = GetIcon(kCapIconID);
                        PlotIcon (&capRect[i], capIcon[i]);
                        if (i <16)
                        {       MoveTo(i*40+65,515);
                        }
                        else
                        {       MoveTo ( (i-16)*40 +65, 575);
                        }
                        TextSize (7);
                        NumToString ((i+1),capNum);
                        DrawString (capNum);
                }

        SetRect (&imageRectG, 0,72,830,472);
        FrameRect (&imageRectG);

/* set up clipping region of display window   (imageWin) */
        savedClip = NewRgn ();
        memErr = MemError();
        if (memErr != 0)
        {       osError (memErr, message);
                DisposePtr ((Ptr)gexp_rgn);
                releaseDataMem (data8, data16);
                releasePixMem(pixMap, myCTable);
                return false;
        }
        clipRect = imageRectG;
        SetRectRgn(savedClip, clipRect.left, clipRect.top, clipRect.right, clipRect.bottom);

/************************ start exposure loop *********************/
        while (!quit)
        {
                status = 10;
                FlushEvents;
                GetNextEvent (everyEvent,&event);
                if (event.what == keyDown)
        {       if ((event.modifiers & cmdKey) &&
                    ((char)(event.message &charCodeMask) == '.'))
                        quit = true;
        }

/* fill data16 buffer with valid data from ccd exposure */
        if (!pl_exp_start_seq (hcam, data16))
        {       errCam (hcam);
                DisposeRgn (savedClip);
                DisposPtr ((Ptr)gexp_rgn);
                releaseDataMem (data8, data16);
                releasePixMem(pixMap, myCTable);
                return false;
        }

        /* check status, then transfer data from camera controller to Mac */
        while ( (status != READOUT_COMPLETE) && (status != READOUT_FAILED) )
        {
                if (!pl_exp_check_status (hcam, &status, &byte_cnt))
                {       errCam (hcam);
                        DisposeRgn (savedClip);
                        DisposPtr ((Ptr)gexp_rgn);
                        releaseDataMem (data8, data16);
                        releasePixMem(pixMap, myCTable);
                        return false;
                }
        }

/* fill data8 buffer with valid data, and it is overwritten in every loop */
        if (!make8Bits (exp_pix, data8, data16))
        {       DisposeRgn (savedClip);
                DisposPtr ((Ptr)gexp_rgn);
                releaseDataMem (data8, data16);
                releasePixMem(pixMap, myCTable);
                return false;
        }
        ObscureCursor();        /* hide cursor untill it is moved */

/* copy data from pixel map to display window */
        CopyBits ((BitMapPtr)(*pixMap), &(background->portBits),
                        &((*pixMap)->bounds), &imageRectG, srcCopy, savedClip);
}
/********************* end of exposure loop ***********************/

/* Load the string resource containing the default area rectangles   *
 * 1st 32 resourse IDs from 128 to 159 saves the top coordinate of 32 rectangles,
 * 2nd 32IDs save the bottom cordinates.
 *      top=128+i, bottom=160+i, right=192+i, left=224+i  (i= 1 to 32)      */

ForeColor (yellowColor);        /* set foreground color to yellow*/
for (i = 0; i<32; i++)
{       /* get four cordinates of each rect */
        topHdl[i] = GetString (128+i); /* get string from resourse */
        HLock((Handle)topHdl[i]);
        StringToNum (*topHdl[i], &top);
        HUnlock((Handle)topHdl[i]);
        gdataRect[i].top = top;

        bottomHdl[i] = GetString (160+i);
        HLock((Handle)bottomHdl[i]);
        StringToNum (*bottomHdl[i], &bottom);
        HUnlock((Handle)bottomHdl[i]);
        gdataRect[i].bottom = bottom;

        rightHdl[i] = GetString (192+i);
        HLock((Handle)rightHdl[i]);
        StringToNum (*rightHdl[i], &right);
        HUnlock((Handle)rightHdl[i]);
        gdataRect[i].right = right;

        leftHdl[i] = GetString (224+i);
        HLock((Handle)leftHdl[i]);
        StringToNum (*leftHdl[i], &left);
        HUnlock((Handle)leftHdl[i]);
        gdataRect[i].left = left;

        FrameRect (&gdataRect[i]);

/* label each rectangle with capillary number */
        labelh = gdataRect[i].left;
        labelv = gdataRect[i].bottom + 10;
        MoveTo (labelh,labelv);
        NumToString (i+1, capNum);
        DrawString (capNum);

        SetRect (&stringRect[i], labelh-2, labelv-12, labelh+17, labelv+5);
}
ForeColor (blackColor);  /* reset foreground color to black */

FlushEvents;
```

```
tempRgn = NewRgn ();
outlineRgn = NewRgn ();
if (tempRgn == nil | outlineRgn == nil)
{       SysBeep (30);
        memErr = MemError();
        osError( memErr, message);
        DisposeRgn (savedClip);
        DisposPtr ((Ptr)gexp_rgn);
        releaseDataMem (data8, data16);
        releasePixMem(pixMap, myCTable);
        return false;
}

go = false;

/********************** start selection of exp_rgn loop*******************/
while (!go)
{
        GetNextEvent (everyEvent, &event);
        doEvent2 (&event, &mouseCode, &keyCode);
        if (keyCode == -1)
        {       go = true;
        }
        else if (event.what == mouseDown && (mouseCode>=0) && (mouseCode < 32))
        {
                activeRect = mouseCode;
                ForeColor (redColor);
                PlotIcon (&capRect[activeRect], capIcon[activeRect]);
                if (activeRect <16)
                {       MoveTo(activeRect*40+65,515);
                }
                else
                {       MoveTo ( (activeRect-16)*40 +65, 575);
                }
                TextSize(7);
                NumToString((activeRect+1),capNum);
                DrawString(capNum);
                ForeColor (blackColor);
        }

        if (event.what == mouseDown && PtInRect(event.where, &imageRectG))
        {
                xstart = event.where.h & 0xFFFE;
                ystart = event.where.v & 0xFFFE;
                inDrag = true;
        }

        if (inDrag && (activeRect >= 0) && (activeRect < 32))
        {
                xfinish = event.where.h & 0xFFFE;
                yfinish = event.where.v & 0xFFFE;

/* limit the rectangle inside the image display window */
                if (xfinish > imageRectG.right)
                        xfinish = imageRectG.right & 0xFFFE;
                if (yfinish > imageRectG.bottom-15)
                yfinish = (imageRectG.bottom - 15) & 0xFFFE;
                InvertRgn (outlineRgn);

        /* make sure the height of rectangle = 100, let user define the width */
                SetRect (&gdataRect[activeRect], (short)xstart, (short)ystart, (short)xfinish,
                                (short)(ystart+140));
                RectRgn (outlineRgn, &gdataRect[activeRect]);
                CopyRgn (outlineRgn, tempRgn);
                InsetRgn (tempRgn, 1, 1);
                DiffRgn (outlineRgn, tempRgn, outlineRgn);
                InvertRgn (outlineRgn);
        }

/* If we are drawing rectangle and let the mouse up end the drawing */
        if (event.what == mouseUp && inDrag && (activeRect>=0) && (activeRect<32))
        {
                labelh = gdataRect[activeRect].left;
                labelv = gdataRect[activeRect].bottom + 15;
        //      SetRect(&stringRect[activeRect], labelh-2, labelv-12, labelh+17, labelv+5);
                CopyBits ((BitMapPtr)(*pixMap), &(background->portBits),
                        &((*pixMap)->bounds), &imageRectG, srcCopy, savedClip);
                ForeColor (yellowColor);
                for (i=0; i<32; i++)
                {
                        FrameRect (&gdataRect[i]);
                        labelh = gdataRect[i].left;
                        labelv = gdataRect[i].bottom + 10;
                        MoveTo (labelh,labelv);
                        NumToString (i+1, capNum);
                        DrawString (capNum);
                }
                ForeColor (blackColor);
                PlotIcon (&capRect[activeRect], capIcon[activeRect]);
                if (activeRect <16)
                {       MoveTo(activeRect*40+65,515);
                }
                else
                {       MoveTo ( (activeRect-16)*40 +65, 575);
                }
                TextSize(7);
                NumToString((activeRect+1),capNum);
                DrawString(capNum);
                inDrag = false;
                SetEmptyRgn(outlineRgn);
                activeRect = -1;
        }
}
/******************** end of exp_rgn selection loop ******************/

/* save gdataRect to resourse containing the default area rectangle */
for (j = 0; j< 32; j++)
```

```
{
        NumToString (gdataRect[j].bottom, *bottomHdl[j]);
        HNoPurge((Handle)bottomHdl[j]);
        ChangedResource((Handle)bottomHdl[j]);
        error = ResError();
        if ( error != 0)
                osError (error,message);
        else
                WriteResource((Handle)bottomHdl[j]);
        HPurge ((Handle)bottomHdl[j]);

        NumToString(gdataRect[j].right, *rightHdl[j]);
        HNoPurge((Handle)rightHdl[j]);
        ChangedResource((Handle)rightHdl[j]);
        error = ResError();
        if ( error != 0)
                osError (error,message);
        else
                        WriteResource((Handle)rightHdl[j]);
                HPurge ((Handle)rightHdl[j]);

                NumToString(gdataRect[j].top, *topHdl[j]);
                HNoPurge((Handle)topHdl[j]);
                ChangedResource((Handle)topHdl[j]);
                error = ResError();
                if ( error != 0)
                        osError (error,message);
                else
                        WriteResource((Handle)topHdl[j]);
                HPurge ((Handle)topHdl[j]);

                NumToString(gdataRect[j].left, *leftHdl[j]);
                HNoPurge((Handle)leftHdl[j]);
                ChangedResource((Handle)leftHdl[j]);
                error = ResError();
                if ( error != 0)
```

```
                    osError (error,message);
                else
                    WriteResource((Handle)leftHdl[j]);
                HPurge ((Handle)leftHdl[j]);
        }

        /* transfer gdataRect[i] to the exp_rgn[i] on ccd chip          *
         * gexp_rgn on ccd: (0,0), (1100,300)                          *
         * image is copied to imageRectG (0,72), (830,472)             *
         * 1100/830 = 1.3253, +0.5 to make sure >0.5 round to 1        */

        for (i = 0; i<32; i++)
        {
                gdataRect[i].top = gdataRect[i].top - 72;
                gdataRect[i].left = (int)(gdataRect[i].left * 1.3253 + 0.5);
                gdataRect[i].bottom = gdataRect[i].bottom - 72;
                gdataRect[i].right = (int)(gdataRect[i].right * 1.3253 + 0.5);
        }
/* dispose window, rect, and handle related to capIcon and datarect */
        DisposeWindow (background);

/* release memery of rgn_array(gexp_rgn), pixel_stream(data16), data_buffer */
        DisposeRgn (outlineRgn);
        DisposeRgn (tempRgn);
        DisposeRgn (savedClip);
//      DisposPtr ((Ptr)gexp_rgn);
        releaseDataMem (data8, data16);
        releasePixMem(pixMap, myCTable);
//      uninitCam ( hcam);

        return true;
}



/*********** function protocol to dispose data buffer and pixel map ************/
```

```
void releaseDataMem (uns8_ptr data8, uns16_ptr data16)
{
        DisposPtr ((Ptr)data16);
        DisposPtr ((Ptr)data8);
}



void releasePixMem (PixMapHandle pixMap,CTabHandle myCTable)
{
        HUnlock ((Handle)myCTable);
        HUnlock ((Handle)pixMap);
        DisposePixMap (pixMap);
        myCTable = nil;
}



/********** function prototype to make a lookup table for a 8-bit display ************/

Boolean         make8Bits (uns32 total_pix, uns8_ptr data8, uns16_ptr data16)
{
        uns32           n;
        uns16           min,max;
        float           scale;
        uns8_ptr        scale_lut = nil;
        uns16           value;
        Str255          message = "\pOutOfMem";
        OSErr           error = 0;

/* if the pixel is saturated, data16[n] = 4095, make data16[n] = 0 to make it black */
        for (n = 1; n < total_pix; n++)
        {
                if (data16[n] >= 500)  /* make it black, if data16[n] >= 500 */
                        data16[n] = 0;          /* good for 10-8 M allignment solution*/
        }

/* find the min and max intesity in the image */
        min = max = data16[0];
        for (n = 1; n < total_pix; n++)
        {
                if (data16[n] < min)
                        min = data16[n];
                else if (data16[n] > max)
                        max = data16[n];
        }

/* calculate scaling factor */
        if (min < max)
                scale =(float) (255.)/(float)(max - min);
        else
                scale = 1;

        if (total_pix <= max)
        {       for (n= 0; n < total_pix; n++)
                        data8[n] = (uns8)((float)(data16[n] - min)*scale);
        }
        else
        {       scale_lut = (uns8_ptr)NewPtr((Size)max-(Size)min+(Size)1);
                error = MemError();
                if (error != 0)
                {       osError (error, message);
                        return false;
                }

        /*make a lookup table for 256-level display from min upto max intesity */
                for (n= min; n < max+1; n++)
                        scale_lut[n-min] = (uns8)((float)(n - min)*scale);

        /* scale data in all the pixels into 8 bits, using lookup table */
                for (n = 0; n < total_pix; n++)
                {
                        value = (uns16)(data16[n] - min);
                        data8[n] = scale_lut[value];
                }
        }
        DisposePtr((Ptr)scale_lut);
        return true;
}




/************ function protocol to set up pixel map **************/

void    setPixelMap (PixMapHandle  pixMap, short pmWidth, short pmHeight,
                                   CTabHandle cTable, uns8_ptr data8)
{
/* address pixMapHandle (pixMap) to data buffer which handle data from exposure */
        (*pixMap)->baseAddr = (Ptr)data8;
        (*pixMap)->rowBytes = 0x8000 | pmWidth;

/* set exp_rgn (after binned) Rect of CCD chip to pixel map boundary Rect */
        SetRect(&(*pixMap)->bounds, 0, 0, pmWidth, pmHeight);
        (*pixMap)->pmVersion = 0;
        (*pixMap)->packType = 0;
        (*pixMap)->packSize = 0;
        (*pixMap)->hRes = 72;
        (*pixMap)->vRes = 72;
        (*pixMap)->pixelType = 0;
        (*pixMap)->pixelSize = 8;
        (*pixMap)->cmpCount = 1;
        (*pixMap)->cmpSize = 8;
        (*pixMap)->planeBytes = 0;
        DisposCTable ((*pixMap)->pmTable);
        (*pixMap)->pmTable = cTable;
        (*pixMap)->pmReserved = 0;
        return;
```

```
}
```

4. dataAcquisition.c

```
/* function prototype handling the data display panel:
*                   void dataAcquisition (int16 hcam)
 */

#include "ccd.h"

extern  Rect            gdataRect[32];
extern  rgn_ptr         gexp_rgn;
uns32                   gExp_time;
Boolean                 gQuit;

void dataAcquisition (int16 hcam)
{
        OSErr               memErr;
        Str255              message = "\pOutOfMem";
        uns16               exp_ser_size, exp_par_size;
        uns32               exp_pix = 0;
        uns16               exp_total = 1, rgn_total = 32;
        uns32               exp_bytes;
        uns16_ptr           data16 = nil;
        int16               hbuf;
        Str255              prompt = "\pSave as:";
        Str255              defaultName = "\pCamData";
        SFReply             sfReply;
        Point               filePoint;
        int16               RefNum = 0;
        WindowPtr           win = nil;
        EventRecord         event;
        Handle              menubar;
        Rect                upRect, downRect, zoomInRect, shrinkRect, capRect[32];
        Handle              upIcon, downIcon, zoomInIcon, shrinkIcon, capIcon[32];
        Str255              capNum;
        int16               status;
        uns32               byte_cnt;
        int16               himg;
        uns16_ptr           output_data;
        long                fMem,time;
        float               scale = 100;
        int                 baseline = 350;
float               scalingFactor = 1;
int                 change_exp = 0;
int                 baseShift = 0;
int                 test1 = 0, test2=0, displayCap = 9;    /* display cap#10 as default */
Point               pen;
uns32               data_value[32][10];    /* data_value is a 32x10 memeber array*/
int16               dataPoint = 0;
int                 m,n, k, i, step;
uns16               exp_width[32], exp_height[32], m_sec;

gQuit = false;
gExp_time = 400;        /* set default exp_time */
gexp_rgn = (rgn_ptr) NewPtr ( (Size)(32*sizeof(rgn_type)) );
memErr = MemError();
if (memErr != noErr)
{       osError( memErr, message);
        return;
}

for(i = 0; i<32; i++)
{
        gexp_rgn[i].s1 = gdataRect[i].left - 2;
        gexp_rgn[i].p1 = gdataRect[i].top - 2;
        gexp_rgn[i].s2 = gdataRect[i].right + 2;
        gexp_rgn[i].p2 = gdataRect[i].bottom + 2;
        gexp_rgn[i].sbin = 2;
        gexp_rgn[i].pbin = 2;
        exp_width[i] = (uns16)( (gexp_rgn[i].s2 - gexp_rgn[i].s1 +1) / gexp_rgn[i].sbin );
        exp_height[i] = (uns16)( (gexp_rgn[i].p2 - gexp_rgn[i].p1 +1) /gexp_rgn[i].pbin );
        exp_pix += (uns32)(exp_width[i]*exp_height[i]);
}
if (!pl_exp_setup_seq (hcam,1, 32, (rgn_const_ptr)gexp_rgn, TIMED_MODE,
                        gExp_time, &exp_bytes) )
{       SysBeep(60);
        errCam (hcam);
        DisposePtr ((Ptr)gexp_rgn);
        return;
}

/* allocate memory for pixel_stream (data), each pixels takes 2 bytes        *
 * actually, pl_exp_setup_seq() return the exact number in &exp_bytes        */

//exp_bytes = exp_pix * (uns32)2;
data16 =(uns16_ptr) NewPtr ((Size)exp_bytes);
memErr = MemError();
if (memErr != noErr)
{       osError( memErr, message);
        DisposePtr ((Ptr)gexp_rgn);
        return;
}
output_data = (uns16_ptr) NewPtr ((Size)exp_bytes);
memErr = MemError();
if (memErr != noErr)
{       osError( memErr, message);
                DisposePtr ((Ptr)gexp_rgn);
                DisposePtr ((Ptr)data16);
                return;
}

/* allocate memory for data buffer */
if ( !pl_buf_alloc ( &hbuf, (int16)exp_total, (int16)PRECISION_UNS16, (int16)rgn_total,
                        (rgn_const_ptr)gexp_rgn) )
{       errCam (hcam);
```

```
        DisposePtr ((Ptr)gexp_rgn);
        DisposePtr ((Ptr)data16);
        DisposPtr ((Ptr)output_data);
        return;
}

/* open file for data writing */
filePoint.h = 100;
filePoint.v = 200;
SFPutFile (filePoint, prompt, defaultName,nil, &sfReply);

/* If error happens, file cannot be opened, openFile return Refum = -1. If user cancels,
 * RefNum = 0. If File opened successfully, RefNum = 7992 */

if (sfReply.good)
{       createFile ( sfReply);
        RefNum = openFile (sfReply);
}
if (RefNum == -1)
        return;

menubar = GetNewMBar (kMenuBarID);
SetMenuBar (menubar);
DrawMenuBar();

/* open data display window */
win = GetNewWindow (kbackgroundID, nil, (WindowPtr)-1L);
if (win != nil)
{
        ShowWindow (win);
        SetPort (win);
        MoveTo (0,80);
}
else
{
        printf ("could not find data display window\nclick to go on\n");
```

```
    while (!Button() )
    {}
}

SetRect (&upRect, 450, 20,490, 60);
upIcon = GetIcon(kUpIconID);
PlotIcon (&upRect, upIcon);

SetRect (&downRect, 500, 20,540, 60);
downIcon = GetIcon(kDownIconID);
PlotIcon (&downRect, downIcon);

SetRect (&zoomInRect, 550, 20,590, 60);
zoomInIcon = GetIcon(kzoomInIconID);
PlotIcon (&zoomInRect, zoomInIcon);

SetRect (&shrinkRect, 600, 20,640, 60);
shrinkIcon = GetIcon(kshrinkIconID);
PlotIcon (&shrinkRect, shrinkIcon);

/*Set rectangles for the 32 capillary icons*/
for(i = 0; i < 32; i++)
{
    if (i <16)
    {
        SetRect(&capRect[i], (short)(i*40+55), (short)505, (short)(i*40+85),
            (short)535);
    }
    else
    {       SetRect(&capRect[i], (short)((i-16)*40+55), (short)565,
            (short)((i-16)*40+85), (short)595);
    }
}

/*Draw the capillary icons in the panel window*/
for(i = 0; i < 32; i++)
```

```
{
        capIcon[i] = GetIcon(kCapIconID);
        PlotIcon (&capRect[i], capIcon[i]);
        if (i <16)
        {       MoveTo(i*40+65,515);
        }
        else
        {       MoveTo ( (i-16)*40 +65, 575);
        }
        TextSize (7);
        NumToString ((i+1), capNum);
        DrawString (capNum);
}
MoveTo (0, 350);

/************************* start data acquisition loop *********************/
while (!gQuit)
{       dataPoint += 1;

/* initialize 2D-array data_value[32][10] to 0 */
        for (i = 0; i<32; i++)
        {       for (k = 0; k<10; k++)
                        data_value[i][k] = 0;
        }

        status = 10;
        test1 = displayCap;
        GetNextEvent (everyEvent,&event);

/* set up new sequence, if exp_time changes (doEvent returns 1, otherwise 0)        *
 * when gExp_time changes, exp_bytes doesn't change, so, I don't think             *
 * it is necessary to re-allocate memory for data16, output_data and hbuf          */

        doEvent (&event, &change_exp, &scalingFactor, &baseShift, &displayCap);
        if (change_exp == 1)
        {
```

```
DisposPtr ((Ptr)data16);
DisposPtr ((Ptr)output_data);
if (!pl_buf_free(hbuf))
{       errCam (hcam);
}

if (!pl_exp_setup_seq (hcam,1, 32, (rgn_const_ptr)gexp_rgn,
                        TIMED_MODE,gExp_time, &exp_bytes) )
{       errCam (hcam);
        if ((RefNum != 0) && (RefNum != -1))
                closeFile (RefNum,sfReply);
        DisposPtr ((Ptr)gexp_rgn);
        return;
}

data16 =(uns16_ptr) NewPtr ((Size)exp_bytes);
memErr = MemError();
if (memErr != noErr)
{       osError( memErr, message);
        DisposePtr ((Ptr)gexp_rgn);
        if (!pl_buf_free(hbuf))
        {       errCam (hcam);
        }
        return;
}

output_data = (uns16_ptr) NewPtr ((Size)exp_bytes);
memErr = MemError();
if (memErr != noErr)
{       osError( memErr, message);
        DisposePtr ((Ptr)gexp_rgn);
        if (!pl_buf_free(hbuf))
        {       errCam (hcam);
        }
        DisposePtr ((Ptr)data16);
        return;
        }

        if (!pl_buf_uninit())
        {
            errCam (hcam);
            return;
        }

        if (!pl_buf_init())
        {
            errCam (hcam);
            return;
        }

        if ( !pl_buf_alloc ( &hbuf, (int16)exp_total, (int16)PRECISION_UNS16,
                    (int16)rgn_total, (rgn_const_ptr)gexp_rgn) )
        {   errCam (hcam);
            DisposePtr ((Ptr)gexp_rgn);
            return;
        }
    }

/* handle scale zoom in or zoom out */
    if (scalingFactor != 1)
    {
        scale = scale * scalingFactor;
        scalingFactor = 1;
    }

/* handle the baseline shift up and down */
    if (baseShift != 0)
    {
        baseline = baseline + baseShift;
        baseShift = 0;
    }
```

```
GetPen (&pen);

if (displayCap != test1)
{
        test2 = test1;
        ForeColor (redColor);
        PlotIcon (&capRect[displayCap], capIcon[displayCap]);
        if (displayCap <16)
        {       MoveTo(displayCap*40+65,515);
        }
        else
        {       MoveTo ( (displayCap-16)*40 +65, 575);
        }
        TextSize (7);
        NumToString ((displayCap+1),capNum);
        DrawString (capNum);
        ForeColor(blackColor);
        MoveTo (pen.h, pen.v);
}

if (displayCap == test1)
{
        ForeColor (blackColor);
        PlotIcon (&capRect[test2], capIcon[test2]);
        if (test2 <16)
        {       MoveTo(test2*40+65,515);
        }
        else
        {       MoveTo ( (test2-16)*40 +65, 575);
        }
        TextSize (7);
        NumToString ((test2+1),capNum);
        DrawString (capNum);
        MoveTo (pen.h, pen.v);
}
FlushEvents;
```

```
    if (!pl_exp_start_seq (hcam, data16))
    {   errCam (hcam);
        if ((RefNum != 0) && (RefNum != -1))
            closeFile (RefNum,sfReply);
        DisposPtr ((Ptr)gexp_rgn);
        DisposPtr ((Ptr)data16);
        DisposPtr ((Ptr)output_data);
        if (!pl_buf_free(hbuf))
        {   errCam (hcam);
        }
        return;
    }

/* check status, then transfer data from camera controller to Mac */
    time = 0;
    while ( (status != READOUT_COMPLETE) && (status != READOUT_FAILED) )
    {
        if (!pl_exp_check_status (hcam, &status, &byte_cnt))
        {   errCam (hcam);
            SysBeep(30);
            if (RefNum != 0)
                closeFile (RefNum,sfReply);
            DisposPtr ((Ptr)gexp_rgn);
            DisposPtr ((Ptr)data16);
            DisposPtr ((Ptr)output_data);
            if (!pl_buf_free(hbuf))
            {   errCam (hcam);
            }
            return;
        }
        time++;
        if (time >= 4096)
        {   if (RefNum != 0)
                closeFile (RefNum,sfReply);
            DisposPtr ((Ptr)gexp_rgn);
            DisposPtr ((Ptr)data16);
            DisposPtr ((Ptr)output_data);
            if (!pl_buf_free(hbuf))
            {   errCam (hcam);
            }
            SysBeep(30);
            SysBeep(30);
            SysBeep(30);
            SysBeep(30);
            return;
        }
    }

/* finish sequence, transfer data from camera controller to hbuf in Mac */
    if (!pl_exp_finish_seq (hcam, data16, hbuf) )
    {   if (RefNum != 0)
            closeFile (RefNum,sfReply);
        DisposPtr ((Ptr)gexp_rgn);
        DisposPtr ((Ptr)data16);
        if (!pl_buf_free(hbuf))
        {   errCam (hcam);
        }
        errCam (hcam);
    }

/* the following "for" structure handles the data buffer including software binning, write
        and display data */
    for (i = 0; i<32; i++)
    {
        if (!pl_buf_get_img_handle (hbuf,(int16)0, (int16)i, &himg))
        {   if ((RefNum != 0) && (RefNum != -1))
                closeFile (RefNum,sfReply);
            DisposPtr ((Ptr)gexp_rgn);
            DisposPtr ((Ptr)data16);
            if (!pl_buf_free(hbuf))
            {   errCam (hcam);
            }
            errCam (hcam);
        }

    /* output_data[] is transfered to RAM as an array */
        if (!pl_buf_get_img_ptr (himg, (void_ptr)&output_data))
        {
            if ((RefNum != 0) && (RefNum != -1))
                closeFile (RefNum,sfReply);
            DisposPtr ((Ptr)gexp_rgn);
            DisposPtr ((Ptr)data16);
            if (!pl_buf_free(hbuf))
                errCam (hcam);
            errCam (hcam);
        }

/* add the data_array elements together, extract 10 data_points out */
        step = (int)((exp_height[i] - 2) / 10 + 0.5);
        for (k = 0; k < 10; k++)
        {
            for (n = step*k +1; n < step*(k+1) +1; n++)
            {   for (m = 1; m< exp_width[i]-1; m++)
                {data_value[i][k] += output_data[exp_width[i]*(n-1) + m];
                }
            }
        }
    }

/* write data to disk, closeFile then reopen it to force data to be written to disk */
    if (sfReply.good)
    {   if ((RefNum != 0) && (RefNum != -1))
        {   writeFile (RefNum, sfReply, 40*32, (Ptr)data_value);
                closeFile (RefNum, sfReply);
        }
        RefNum = openFile (sfReply);
    }
```

```
/* if use the following line of codes, scaling occurs once mouse in the rect without
 * mousedown
 *      GetNextEvent (mouseDown, &event);
 *      if ( PtInRect (event.where, &zoomInRect) )
 *              scale = scale * 0.5;
 *      else if ( PtInRect (event.where, &shrinkRect) )
 *              scale = scale *2;
 */

/* display data , win 350 x 820 */
        if (win != nil)
        {       if ((dataPoint % 820) == 0)
                {       EraseRect (&win->portRect);
                        PlotIcon (&upRect, upIcon);
                        PlotIcon (&downRect, downIcon);
                        PlotIcon (&zoomInRect, zoomInIcon);
                        PlotIcon (&shrinkRect, shrinkIcon);
                        for(i = 0; i < 32; i++)
                        {
                                PlotIcon (&capRect[i], capIcon[i]);
                                if (i <16)
                                {       MoveTo(i*40+65,515);
                                }
                                else
                                {       MoveTo ( (i-16)*40 +65, 575);
                                }
                                TextSize (7);
                                NumToString ((i+1),capNum);
                                DrawString (capNum);
                        }
                        MoveTo (0, baseline - data_value[displayCap][4]/scale);
                        dataPoint = 0;
                }
                if (dataPoint < 820)
                {
```

```
            LineTo (dataPoint,  baseline - data_value[displayCap][4]/scale);
        }
    }
}
/******************** end of data acquisition loop **********************/

/* release memery of rgn_array(gexp_rgn), pixel_stream(data16), data_buffer */
    if ((RefNum != 0) && (RefNum != -1))
    closeFile (RefNum,sfReply);
    DisposeMenu  ( GetMHandle (kFileMenu));
    DisposeMenu  ( GetMHandle (kExposureMenu));
    DisposPtr ((Ptr)gexp_rgn);
    DisposPtr ((Ptr)data16);
    DisposePtr ((Ptr) output_data);
    if (!pl_buf_free (hbuf))
        errCam (hcam);
    DisposeWindow  (win);
    return;
}
```

5. dataFile.c

```
/* functions prototypes handling files for data write-to-disk:
 *          void    createFile (SFReply reply)
 *          int16   openFile (SFReply reply)
 *          void    writeFile (int16 refNum, SFReply reply, long count, Ptr dataBuf)
 *          void    closeFile (int16 refNum, SFReply reply)
 */

#include        "ccd.h"

void createFile (SFReply reply)
{       OSType          creator = 'RJRJ';
        OSErr           error;
```

```
         Str255          message = "\pFileNotCreate";

/* when the file doesn't exist, FSDelete returns with error (filenotfound) but in case of
**replacing, it will delete both fork of the file and create a new file under the same fName
**and vRefNum(replace the old file)  *****/

         FSDelete (reply.fName, reply.vRefNum);
         error = Create (reply.fName, reply.vRefNum, creator, 'Data');
         if (error != 0)
                  osError (error, message);
         return;
}



int16 openFile (SFReply reply)
{        OSErr           error;
         Str255          message = "\pFileNotOpen";
         int16           refNum;

         error = FSOpen (reply.fName, reply.vRefNum, &refNum);
         if(!error)
                  return refNum;
         else
         {        osError (error, message);
                  return (int16)-1;
         }
}



void writeFile (int16 refNum, SFReply reply, long count, Ptr dataBuf)
{        Str255 message = "\pFileNotWrite";
         OSErr  error;

         SetFPos (refNum, fsFromLEOF, (long)0);
```

```
        error = FSWrite (refNum, &count, dataBuf);
        if (error != 0)
                osError (error, message);
//      FlushVol (reply.fName, refNum);     *it doesn't help to write file to disk*
        return;
}



void closeFile (int16 refNum, SFReply reply)
{       OSErr  error;
        Str255 message = "\pFileNotClose";

        FlushVol (reply.fName, refNum);
        error = FSClose (refNum);
        if (error != 0)
                        osError (error, message);
        return;
}
```

6. event.c

```
/* function prototypes handling events:
 *              void    doEvent (EventRecord *event, int *change_exp,
 *                                      float *scalingFactor, int *baseShift, int *displayCap)
 *              void    doEvent2 (EventRecord *eventPtr, int *mouseCode, int *keyCode)
 *              void    handleMouseDown (EventRecord *eventPtr, int *mouseCode)
 */

#include "ccd.h"

void doEvent (EventRecord *event, int *change_exp, float *scalingFactor, int *baseShift,
                int *displayCap)
{
```

```
char            key;
WindowPtr       mdown_window;

switch (event->what)
{
        case autoKey:
        case keyDown:
                key = (char) (event->message & charCodeMask);
                if ( event->modifiers & cmdKey)
                *change_exp = doMenu ( MenuKey ( key));
        break;

        case mouseDown:
                switch ( FindWindow (event->where, &mdown_window) )
                {
                        case inMenuBar:
                                *change_exp = doMenu (MenuSelect (event->where));
                        break;

                        case inContent:
                        {
                                int     i;
                                Rect    zoomInRect, shrinkRect, upRect, downRect,
                                        capRectG[32];
                                /* shrink 5 pixels for each icon rect */
                                SetRect (&upRect, 455, 25,485, 55);
                                SetRect (&downRect, 505, 25,535, 55);
                                SetRect (&zoomInRect, 555, 25,585, 55);
                                SetRect (&shrinkRect, 605, 25,635, 55);

                                for (i = 0; i<32; i++)
                                {
                                        if (i <16)
                                        {SetRect (&capRectG[i], (short)(i*40+55),
                                                (short)505, (short)(i*40+85), (short)535);
                                        }
```

```
                    else
                    {SetRect (&capRectG[i], (short)((i-16)*40+55),
                          (short)565, (short)((i-16)*40+85), (short)595);
                    }
                }

                for (i = 0; i<32; i++)
                {
                    if ( PtInRect(event->where, &capRectG[i]) )
                        *displayCap = i;
                }

                if ( PtInRect (event->where, &downRect) )
                    *baseShift = 50;
                else if ( PtInRect (event->where, &upRect) )
                    *baseShift = -50;
                else if ( PtInRect (event->where, &zoomInRect) )
                    *scalingFactor = 0.5;
                else if ( PtInRect (event->where, &shrinkRect) )
                    *scalingFactor = 2;
            }
        break;
    }
    break;
}
}



/**** function prototype which returns multiple values *******/

void    doEvent2 (EventRecord *eventPtr, int *mouseCode, int *keyCode)
{
    switch (eventPtr->what)
    {
        case keyDown:
                if ( (eventPtr->modifiers & cmdKey) &&
                        ((char)(eventPtr->message & charCodeMask) == 'q') )
                *keyCode = -1;          /* return value to keyCode */
            break;

            case mouseDown:
                handleMouseDown (eventPtr, mouseCode);
                /* 100 for NotInIconWin, 0 - 31 for cap */
            break;
        }
}



void handleMouseDown (EventRecord *eventPtr, int *mouseCode)
{
        Rect    capRectG[32];
        int     i = 100, j = 100;

        for (i = 0; i<32; i++)
        {
            if (i <16)
            {
                SetRect(&capRectG[i],(short)(i*40+55),(short)505,
                                (short)(i*40+85),(short)535);
            }
            else
            {   SetRect(&capRectG[i],(short)((i-16)*40+55),(short)565,
                                (short)((i-16)*40+85),(short)595);
            }
        }
        for (i = 0; i<32; i++)
        {
            if ( PtInRect(eventPtr->where, &capRectG[i]) )
            {
                j = i;
                }
        }

        *mouseCode = j; /* return Value to mouseCode */
}


/*** mouseCode:
                capillary icon:         0 to 31
                not in capIconRect:     100
 *** eventCode
                go:     -1
                not in capIconRect:     100 ***/
```

7. menu.c

```
/* function prototype handles menu operations:
 *              int     doMenu (long selectmenu)
 */

#include        "ccd.h"

extern Boolean          gQuit;
extern uns32            gExp_time;

int     doMenu (long selectmenu)
{
        int                     menuID;
        int                     itemID;
        int                     changeExp = 0;

        menuID = HiWord ( selectmenu);
        itemID = LoWord ( selectmenu);
        switch ( menuID)                /* If no selection is made menuID is nil */
```

```
    {
        case kFileMenu:
            if (itemID == kFileQuit)
                gQuit = true;
            changeExp = 0;
        break;

        case kExposureMenu:
            switch (itemID)
            {   case kExposure1:
                    gExp_time = 100;
                    changeExp = 1;
                break;

                case kExposure2:
                    gExp_time = 200;
                    changeExp = 1;
                break;

                case kExposure3:
                    gExp_time = 300;
                    changeExp = 1;
                break;

                case kExposure4:
                    gExp_time = 400;
                    changeExp = 1;
                break;
            }
        break;
    }
    HiliteMenu (0);         /* after selected, dehilite the menu bar */
    return changeExp;
}
```

8. showerr.c

```
/* show error code and error message if an error occurs
 * function prototypes:
 *              void    errCam (int16 hcam)
 *              void    osError (OSErr error, Str255 message)
 */

#include "ccd.h"

void errCam (int16 hcam)
{
        short           item;
        DialogPtr       dialog;
        Str255          errCode;
        OSErr           resErr;
        Str255          resourse_error;
        int16           cam_err;
        char            msg[ERROR_MSG_LEN];
        Str255          message;
        int             i;

        cam_err = pl_error_code();
        pl_error_message (cam_err,msg);

/* close camera,Ifound hcam == 0 if camera is successfully opened */
        if (hcam == 0)
                pl_cam_close (hcam);
        NumToString (cam_err,errCode);

/* transfer char array msg[] to Str255  */
        for (i = ERROR_MSG_LEN-1; i >= 0; i--)
        {       message[i+1] = msg[i];
                if (msg[i] == 0)
                        message[0] = i+1;
        }
        ParamText (errCode, message, nil, nil);
        dialog = GetNewDialog ( kcamErrDialogID, nil, (WindowPtr)-1);
        SysBeep (30);
        while (item != kDialogOK)
        {       ModalDialog (nil,&item);
        }
        DisposDialog (dialog);
}




void    osError (OSErr error, Str255 message)
{
        short           item;
        DialogPtr       dialog;
        Str255          errorCode;
        OSErr           resErr;

        if (error != noErr)
        {
                NumToString (error, errorCode);
                ParamText ( message, errorCode, nil,nil);

        /* bring the dialog to the front, show it */
                dialog = GetNewDialog ( kOSErrDialogID, nil, (WindowPtr)-1);
                SysBeep (30);

                while (item != kDialogOK)
                {
                        ModalDialog (nil,&item);
                }
                DisposDialog (dialog);
        }
}
```

Appendix I. Data Acquisition Software

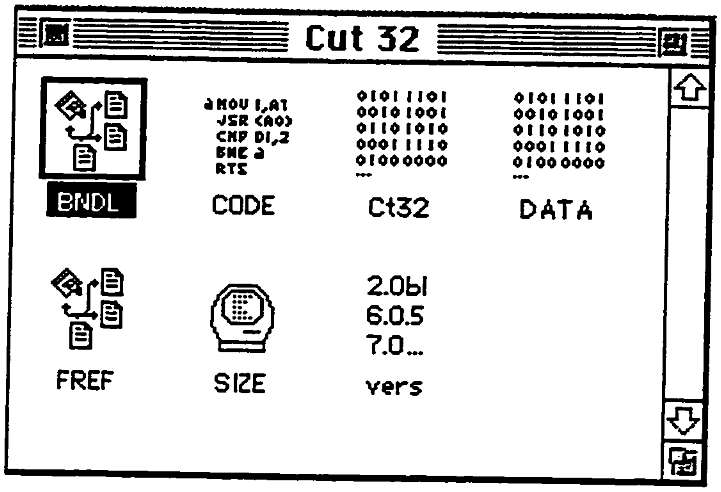
Part II. SOURCE CODE OF "CUT32"

II-1. Header File

Split 32.h

```
#include <Files.h>
#include <StandardFile.h>

#define        kCreator       'RJRJ'          /* the created files have this creator    */
#define        kMinBlockSize         2560
#define        kMinDataSize          1280
#define        kSleepTime            0        /* a tenth of a second for other apps     */

/* function prototype in Main.c */
                    void    main            (void);
                    OSErr   Initialize      (void);
                    OSErr   doEvent         (short mask, unsigned long sleep,
                                             RgnHandle mouseRgn);
            pascal  OSErr   oappHandler     (const AppleEvent *theAE, AppleEvent *reply,
                                             long refCon);
            pascal  OSErr   quitHandler     (const AppleEvent *theAE, AppleEvent *reply,
                                             long refCon);
            pascal  OSErr   pdocHandler     (const AppleEvent *theAE,
                                            AppleEvent *reply, long refCon);
            pascal  OSErr   odocHandler     (const AppleEvent *theAE, AppleEvent *reply,
                                             long refCon);
/* function prototype in Split 32.c */
                    OSErr   Create32Files   (FSSpec sourceSpec, ScriptCode script,
                                             OSType type, FSSpec* newFile);
                    OSErr   HandleFile      (FSSpec sourceSpec);
                    OSErr   CleaveFile      (FSSpec sourceSpec, FSSpec* destSpec);
                    void    TransferData    (void);
                    OSErr   WriteFiles      (FSSpec* destSpec);
```

II-2. Resource Files



II-3. Function Files

1. Main.c
2. Split 32.c

Main.c

```
/* function prototypes:
 *      void    main    (void)
 *      OSErr  Initialize        (void)
 *      OSErr  doEvent           (short mask, unsigned long sleep, RgnHandle mouseRgn)
 *      pascal  OSErr oappHandler    (const AppleEvent *theAE, AppleEvent *replyAE,
 *                                    long refCon)
 *      pascal  OSErr quitHandler    (const AppleEvent *theAE, AppleEvent *reply,
 *                                    long refCon)
 *      pascal  OSErr pdocHandler    (const AppleEvent *theAE, AppleEvent *reply,
 *                                    long refCon)
 *      pascal  OSErr odocHandler    (const AppleEvent *theAE, AppleEvent *reply,
 *                                    long refCon)
 */

#include        "Split 32.h"            /* prototypes and constants    */

Ptr             gDataIn, gDataOut;      /* pointers to the data buffers */
long            gInSize, gOutSize;      /* sizes of the data buffers    */
Boolean         gQuit;

void    main (void)
{
        StandardFileReply       reply;
        FSSpec                  newFile[32];            /* the FSSpecs for our new files*/
        OSErr                   error;
        EventRecord             event;

        if (Initialize () == noErr)
        {
                do {
                        doEvent (everyEvent, kSleepTime, nil);
                } while  (!gQuit);
        }
}



OSErr  Initialize (void)
{
        OSErr                   error;
        long                    total, blockSize;       /* size of file buffer */

        InitGraf (&qd.thePort);
        InitFonts ();
        FlushEvents (everyEvent, 0);
        InitWindows ();
        InitMenus ();
        TEInit ();
        InitDialogs (0L);
        InitCursor ();

        gQuit = false;
//      PurgeSpace (&total, &blockSize);                /* find how much free mem */
//      blockSize = 1902L * 1024;                       /* allocate buffers */
//      blockSize -= 2048;                              /* save some for the program  */
        blockSize = 32767;
        blockSize = (blockSize / kMinBlockSize) * kMinBlockSize;

        gDataIn = NewPtr (blockSize);
        error = MemError();
        if (gDataIn && !error && blockSize > 0)
        {
                gInSize = blockSize / 2;
                gDataOut = gDataIn + gInSize;
                gOutSize = gInSize / 32;                /* let the user choose any file  */
                error = AEInstallEventHandler (kCoreEventClass, kAEOpenApplication,
                        NewAEEventHandlerProc (oappHandler), 0, false);

                if (!error)
                error = AEInstallEventHandler (kCoreEventClass, kAEOpenDocuments,
                                NewAEEventHandlerProc (odocHandler), 0, false);

                if (!error)
                error = AEInstallEventHandler (kCoreEventClass, kAEPrintDocuments,
                                NewAEEventHandlerProc (pdocHandler), 0, false);

                if (!error)
                error = AEInstallEventHandler (kCoreEventClass, kAEQuitApplication,
                                NewAEEventHandlerProc (quitHandler), 0, false);
        }
        else
        {
                if (!error)
                        error = 1;                      /* pass back some error value */
        }
        return error;                                   /* pass any failure back        */
}



OSErr doEvent (short mask, unsigned long sleep, RgnHandle mouseRgn)
{
        EventRecord                     theEvent;
        OSErr                           error;

        WaitNextEvent (everyEvent, &theEvent, sleep, mouseRgn);
        switch (theEvent.what)
        {
                case updateEvt:
                        BeginUpdate ((WindowPtr) theEvent.message);
```

```
                    EndUpdate ((WindowPtr) theEvent.message);
                break;

                case keyDown:
                case autoKey:
                    if ((theEvent.message & charCodeMask)>>8 == 'Q')
                        gQuit = true;
                break;

                case kHighLevelEvent:
                    error = AEProcessAppleEvent (&theEvent);
                    if (error)
                        if (AEInteractWithUser (kNoTimeOut, nil, nil) == noErr)
                        {       /* show an error message        */
                        }
                    break;
        }
        return error;
}



pascal OSErr oappHandler (const AppleEvent *theAE, AppleEvent *replyAE, long refCon)
{
        OSErr                           error;
        StandardFileReply       sfReply;

        error = noErr;                          /* use Pack3 to choose a file   */
        do {    StandardGetFile (nil, -1, nil, &sfReply);
                if (sfReply.sfGood)             /* process the user's file      */
                        error = HandleFile (sfReply.sfFile);
                else
                        gQuit = true;           /* quit if Cancel is choosen    */
        } while (!gQuit && !error);
        return error;
}
```

```
pascal OSErr quitHandler (const AppleEvent *theAE, AppleEvent *reply, long refCon)
{
        gQuit = true;
}
pascal OSErr pdocHandler (const AppleEvent *theAE, AppleEvent *reply, long refCon)
{       /* since I can't handle this I'll do nothing      */
        return noErr;
}



pascal OSErr odocHandler (const AppleEvent *theAE, AppleEvent *reply, long refCon)
{
        OSErr           error;
        long            actualSize;
        DescType        dummyType;
        AEKeyword       missed;
        AEDesc          theList;
        long            numFiles, fileCount;
        short           numErrors = 0;
        OSErr           firstError = noErr;

/* the direct object parameter is the list of aliases        */
        error = AEGetParamDesc (theAE, keyDirectObject, typeAEList, &theList);
        if (error)
                return error;

/* Get the "missed keyword" attribute from the AppleEvent. (Safety check)  */
        error = AEGetAttributePtr(theAE, keyMissedKeywordAttr, typeKeyword,
                        &dummyType, (Ptr)&missed, sizeof(AEKeyword), &actualSize);

/* If the descriptor isn't found, then we got the required parameters. */
        if (error != errAEDescNotFound)
        {       error = errAEEventNotHandled;
                return error;
        }
/* get number of files in list    */
        error = AECountItems(&theList, &numFiles);
        if (error)
        {       return error;
        }
/* open each file - keep track of errors */
        for (fileCount = 1; fileCount <= numFiles; fileCount++)
        {
                FSSpec          theFile;
                long            actualSize;
                AEKeyword       dummyKeyword;
                DescType        dummyType;
                WindowPtr       newWindow = nil;

                /* get the alias for the nth file, convert to an FSSpec */
                error = AEGetNthPtr(&theList, fileCount, typeFSS, &dummyKeyword,
                        &dummyType, (Ptr) &theFile, sizeof(FSSpec), &actualSize);
                if (error)
                        return error;
                error = HandleFile (theFile);   /* process the user's file */

                if (error)          /* count file opening errors */
                {
                        ++numErrors; /* keep count */
                        if (numErrors == 1)
                                firstError = error;       /* save first error */
                }
        }

/* set best error message */
        if (numErrors)            /* at least one happend */
                error = firstError;
        AEDisposeDesc(&theList);      /* dispose what we allocated */
        gQuit = true;
        return error;
}
```

Split 32.c

```
/* function prototypes:
 *      OSErr  HandleFile       (FSSpec sourceSpec)
 *      OSErr  Create32Files    (FSSpec sourceSpec, ScriptCode script, OSType type,
 *                               FSSpec* newFile)
 *      OSErr  CleaveFile       (FSSpec sourceSpec, FSSpec* destSpec)
 *      void   TransferData     (void)
 *      OSErr  WriteFiles       (FSSpec* destSpec)
 */

#include        "Split 32.h"

extern Ptr      gDataIn, gDataOut;              /* pointers to the data buffers  */
extern long     gInSize, gOutSize;              /* sizes of the data buffers     */

OSErr  HandleFile (FSSpec sourceSpec)           /* works on a FSSpec             */
{
        FSSpec          newFile[32];            /* the FSSpecs for our new files*/
        OSErr           error;
        FInfo           fndrI;                  /* data on our source file       */

        error = FSpGetFInfo(&sourceSpec, &fndrI);
        if (!error)                             /* use system script             */
                error = Create32Files (sourceSpec, smSystemScript,
                                fndrI.fdType, newFile);        /* make new files*/
        if (!error)
                error = CleaveFile (sourceSpec, newFile);
        if (!error)
                error = FlushVol(nil, sourceSpec.vRefNum);
        return error;
}
```

```
OSErr  Create32Files  (FSSpec sourceSpec, ScriptCode script, OSType type,
                       FSSpec* newFile)
{
        OSErr           error;          /* function returns           */
        short           i, j;           /* local indexing variables   */
        long            dirID;          /* the ID of new directory    */
        char            name[64];

        for (j = 0; j < 64; j++)        /* copy name string temporarily */
                name[j] = sourceSpec.name[j];       /* into file 0's space to make a */
        name[0] += 2;                               /* new FSSpec for a folder    */
        if (name[0] > 31)                           /* clip name if too long      */
        {
                name[name[0] - 2]  = '...';
                name[0] = 31;
        }
        name[name[0] - 1]  = ' ';                   /* append ' ƒ' at end         */
        name[name[0]]  = 'ƒ';                       /* let the OS make the FSSpec */
        error = FSMakeFSSpec (sourceSpec.vRefNum, sourceSpec.parID,
                (ConstStr255Param)name, &newFile[0]);
        if (error == fnfErr)                        /* fail if the folder exists  */
                error = FSpDirCreate (&newFile[0], script, &dirID);
        else
                error = fidExists;                  /* not quite the right error  */
        if (!error)
        {
                for (i = 0; i < 32; i++)            /* repeat for 32 files        */
                {
                        for (j = 0; j < 64; j++)    /* copy name string           */
                                name[j] = sourceSpec.name[j];
                        name[0] += 3;
                        if (name[0] > 31)           /* clip name if too long      */
                        {
                                name[name[0] - 3]  = '...';
                                name[0] = 31;
                        }
                        name[name[0] - 2]  = ' ';       /* append digits at end          */
                        name[name[0] - 1]  = 48 + ((i + 1) / 10);
                        name[name[0]]  = 48 + ((i + 1) % 10);
                        if (i < 9)                      /* space instead of leading zero*/
                                name[name[0] - 1] = ' ';/* let the Toolbox make the Spec*/
                        error = FSMakeFSSpec (sourceSpec.vRefNum, dirID,
                                        (ConstStr255Param)name, &newFile[i]);
                        if (error == fnfErr)            /* ensure file does not exist*/
                                error = FSpCreate (&newFile[i], kCreator, type, script);
                        else
                                i = 32;                 /* stop creating files on error  */
                        if (error)
                                i = 32;                 /* stop creating files on error  */
                }
        }
        return error;
}



OSErr CleaveFile (FSSpec sourceSpec, FSSpec* destSpec)
{
OSErr                   error, error2;
short                   sourceRef;              /* refNum for our source file  */
long                    sourceSize, bytesRead, tmpInSize;

bytesRead = 0;
error = FSpOpenDF (&sourceSpec, fsRdPerm, &sourceRef);
if (!error)
{
        error = SetFPos (sourceRef, fsFromStart, 0L);
        if (!error)
        {
                error = GetEOF(sourceRef, &sourceSize);
                if (!error)
                {
                        while (sourceSize > bytesRead)
                        {       /* loop 'til file has been read   */
                               doEvent (everyEvent-highLevelEventMask, kSleepTime, nil);
                                if (gInSize > sourceSize - bytesRead)
                                        gInSize = sourceSize - bytesRead;
                                tmpInSize = gInSize;
                                error = FSRead (sourceRef, &tmpInSize, gDataIn);
                                if (!error)
                                {
                                        bytesRead += gInSize;
                                        gInSize = (gInSize / kMinDataSize) * kMinDataSize;
                                        TransferData ();
                                        gOutSize = gInSize / 32;
                                        error = WriteFiles (destSpec);
                                }
                        }
                }
        }
        error2 = FSClose (sourceRef);       /* close the source file          */
        if (!error)                         /* if no previous error occured*/
                error = error2;             /* return any closing error       */
}
return error;
}



void TransferData (void)                    /* move data within our buffer          */
{
        short   j, k;                       /* copy data between buffers  */
        long    i;
        long    *dataOut, *dataIn;
        short   numSamples;

        dataIn = (long*)gDataIn;
        dataOut = (long*)gDataOut;
```

```
    numSamples = gInSize / kMinDataSize;
    for (i = 0; i < numSamples; i++)
    {
        for (j = 0; j < 32; j ++)
        {
            for (k = 0; k < 10; k++)
            {
                dataOut[j*numSamples*10 + i*10 + k]
                    = dataIn[i*kMinDataSize/4 + j*10 + k];
            }
        }
    }
}



OSErr WriteFiles (FSSpec* destSpec)
{
    OSErr                   error, error2;
    short                   destRef;
    short                   i;          /* general indexing varable     */
    long                    tgOutSize;
    Ptr                     buffer;

    for (i=0; i<32; i++)               /* open each of the 32 files    */
    {
        error = FSpOpenDF (&destSpec[i], fsWrPerm, &destRef);
        if (!error)
        {
            error = SetFPos (destRef, fsFromLEOF, 0L);
            if (!error)                 /* move to the end of the file  */
            {                           /* and write the buffer         */
                tgOutSize = gOutSize;
                buffer  = gDataOut + i * gOutSize;
                error = FSWrite (destRef, &tgOutSize, buffer);
            }
```

```
                    error2=FSClose (destRef);
                    if (!error)                 /* close file before finishing     */
                              error = error2;           /* and pass the first error          */
          }
     }
     return error;
}
```

Appendix II. Igor Procedures

Procedure 1. Layout of Whole Electropherogram from Each Capillary

```
// Macros:      print_1cap_32p_1color(File, DAQ_speed, t_primer, t_end)

// ??? marks:   parameter may need to be changed according to the application
// File:        Filename(eg: 1/23/97)
// DAQ_speed: data collection speed
// t_primer:    time at which the primer comes out
// t_end:       time at which the end peak comes out
// each electropherogram of each capillary is divided into 3 parts.
// t_primer — (t_primer+graph period/3) — (t_primer+graph period*2/3) — t_ end

#include <strings as lists>
#include "SetAxisRange"

Macro print_1cap_32p_1color(File, DAQ_speed, t_primer, t_end)
        string File
        Variable t_primer
        Variable t_end
        Variable t_inc
        string Filename
        string waves
        string onewave
        Variable DAQ_speed
        Variable t1
        Variable t2
        Variable m=16
        Variable i

        waves = WaveList("*",";","")
        t_inc = (t_end-t_primer)/3
        do
                onewave = GetStrFromList(waves,m,";")
```

```
                    SetScale/P x 0,1/(60*DAQ_speed),"",$onewave
                    Smooth/s=2 9,$onewave             //smooth S-G 2nd order 9 points    ???
                    i = 0
                    do
                            display $onewave
                            Label left "Signal"
                            Label bottom "Time (Min)"
                            t1 = t_primer + i*t_inc
                            t2 = t_primer +(i+1)*t_inc
                            SetAxis bottom t1,t2
                            SetAxisRange()
                            i+=1
                    while (i <3)
                    TileWindows/O=1
                    layout/C=1 graph0, graph1, graph2
                    Tile/A=(3,1)
                    Filename = "#"+ num2str(m+1)+ "    "+"File: "+File
                    Textbox/F=0/X=85/Y=5 Filename
                    PrintLayout /D Layout0
                    DoWindow /K graph0
                    DoWindow /K graph1
                    DoWindow /K graph2
                    DoWindow /K Layout0
                    m+=1
            while(m<32)
End
```

Procedure 2. Layout of One Section of the Electropherogram for all 32 Capillaries

```
// Macros:        printout_32_1color(DAQ_speed,Start_time, End_time, Filename,
//                                        caps_per_page)
// Functions:     displaywaves(Hz, j,T1,T2,caps_layout)
//                layoutgraphs(File_name, cap_layout)         Allgraph(a)
// to printout one section of the Electropherogram  of all 32 caps in two ways
//              1) 4  caps each page  (layout4X1), total 8 pages
//              2) 16 caps each page (layout4X4 ), total 2 pages

// Start_time:   start time of electropherogram (min.)
// End_time:     End time of electropherogram (min.)
// DAQ_speed:           data collection speed
// caps_per_page:       enter 16  for 16cap X 2page printout
// caps_per_page:       enter 4   for 4cap X 8pages printout

// printout 32 single color DNA sequencing Data

#include <strings as lists>
#include "SetAxisRange"

Macro printout_32_1color(DAQ_speed,Start_time, End_time, Filename,caps_per_page)
    string Filename
    Variable DAQ_speed
    Variable Start_time
    Variable End_time
    Variable caps_per_page
    Prompt caps_per_page, "caps/page, enter 4 for 4capX8pages layout, 16 for 16X2"

    Variable m=0, n=0

    do
        if (caps_per_page == 4)
            do

        displaywaves(DAQ_speed,n,Start_time,End_time,caps_per_page)
                TileWindows/O=1
                layoutgraphs(Filename,caps_per_page)
                n+=1
            while(n<8)
            break
        endif
```

```
                if( caps_per_page == 16)
                        do

                displaywaves(DAQ_speed,m,Start_time,End_time,caps_per_page)
                                TileWindows/O=1
                                layoutgraphs(Filename,caps_per_page)
                                m+=1
                        while (m<2)
                endif
        while(0)
End



Function displaywaves(Hz, j,T1,T2,caps_layout)
        Variable Hz
        Variable j
        Variable T1
        Variable T2
        Variable caps_layout        // #of caps/layout
        Variable i=0
        String onewave
        String allwaves
        String capname

        allwaves=WaveList("*",":","")        //make a list of all the waves' name

        do
                onewave = GetStrFromList(allwaves,caps_layout*j+i,":")    // onewave is a
        pointer
                display $onewave
                SetScale/P x 0,1/(60*Hz),"",$onewave
                Label left "Signal"
                Label bottom "Time (Min)"
                Smooth/s=2 9,$onewave                //smooth S-G 2nd order 9 points    ???
                capname = "#"+num2str(caps_layout*j+i+1)
```

```
            TextBox /F=0/X=0/Y=0 capname
            SetAxis bottom T1,T2
            SetAxisRange()
            i+=1
        while (i < caps_layout)
End



Function layoutgraphs(File_name, cap_layout)
        String File_name
        Variable cap_layout
        String graphnames
        String onegraph
        String lay = "layout/C=1 "
        Variable k=0

        lay =lay + Allgraph(cap_layout)
        Execute lay

        do
            if (cap_layout == 4)
                Execute "Tile/A=(4,1)"
                break
            endif
            if (cap_layout == 16)
                Execute "Tile/A=(4,4)"
                break
            endif
        while(0)

        Textbox/F=0/X=50/Y=0 File_name
        Execute "PrintLayout /D Layout0"

        //  kill all the graphs
        graphnames = WinList("Graph*",";","WIN:1")
```

```
	do
		onegraph = GetStrFromList(graphnames,k,";")
		DoWindow /K $onegraph
		k+=1
	while (k < cap_layout)

	DoWindow /K layout0
End



Function/S  Allgraph(a)
	Variable a
	Variable b
	string graphnames0
	string graphs

	b = a-1
	graphnames0 = WinList("Graph*",";","WIN:1")	// another way to make the
	string, then Execute the string
	graphs=  GetStrFromList(graphnames0,b,";")
	do
		b = b-1
		graphs = graphs +"," + GetStrFromList(graphnames0,b,";")
	while (b>=0)
	return graphs
End
```

Procedure 3: Calculation of the Limit of Detection

```
// Macros:	display8waves()
//		LOD()
//		LOD_peakheight()
//		layout8X1()
```

```
// It is good for LOD calculation without siphening
// LOD() for electrophoregram without bubble peaks near the analyte peak
//      to use, place csr A on the peak top, csr B on the baseline far right to A
// LOD_peakheight() for electrophoregram with bubble peaks near the analyte peak
//      to use, record Hp & t_mig first, place csr A &B on the baseline for sdev
// Hz:  Data collection speed, j=0(#1to#8), 1(#9 to #16), 2(#17 to #24), 3(#25 to #32)

#include <strings as lists>
#include "SetAxisRange"

Macro diplay8waves(Hz,j)   // j= 0,1,2,3        ???
        Variable i=0, j
        Variable Hz
        String wavenames
        String onewave
        String capname

        wavenames=WaveList("*",":","")
        do
                onewave = GetStrFromList( wavenames,8*j+i,":")
                display $onewave
                SetScale/P x 0,1/Hz,"",$onewave
                Label left "Signal"
                Label bottom "Time (Sec)"
                Smooth/s=2 9,$onewave      //smooth S-G 2nd order 9 points   ???
                SetAxis bottom 200,450      // ???
                SetAxisRange()
                capname = "#"+num2str(8*j+i+1)
                TextBox /F=0/X=90/Y=0 capname
                ShowInfo
                i = i + 1
        while (i <8)
        TileWindows/O=1
End
```

```
//LOD calculation, this program calculate LOD of the top graph

Macro LOD()
        Variable peak_height
        Variable t_mig
        Variable t_inj = 6          // ???
        Variable V_inj=1000         // ???
        Variable V_run=12000        // ???
        Variable C_dye=6e-11        // (mol/litre)  ???
        Variable L=41                // ???
        Variable Nmolec
        Variable Vol_inj,Vol_cap
        Variable LODconc,LODmolec
        String LODprint

        // pcsr = pts in X axis, xcsr = s in X axis
        // place csrA at the peak top, csrB on the baseline right to csrA   !!!

        WaveStats /R=(xcsr(A)+15,xcsr(B)) $CsrWave(B)
        peak_height=$CsrWave(A)[pcsr(A)]-$CsrWave(A)[pcsr(A)+20]
        t_mig=xcsr(A)

        //print $CsrWave(A)[pcsr(A)]
        //print $CsrWave(A)[pcsr(A)+40]
        //print $CsrWave(A)[pcsr(A)+20]

        Vol_cap=(1.96e-8)*L       // (litre)
        Vol_inj=(t_inj*V_inj)/(t_mig*V_run)*Vol_cap
        Nmolec=Vol_inj*C_dye*(6.02e23)          //Numer of molecule injected
        LODconc=3*V_sdev/peak_height*C_dye
        LODmolec=3*V_sdev/peak_height*Nmolec
        LODprint= "LODc="+num2str(LODconc)+"   "+"LODm="+num2str(LODmolec)
        TextBox /F=0/Y=0 LODprint
End
```

```
// If bubble peak after the analyte peak, use this one
// put csr A &B on the baseline for std calculation, input peakheight(Hp) & migration time

macro LOD_peakheight(Hp, t_mig)
        Variable Hp
        Variable t_mig
        Variable t_inj = 5              // ???
        Variable V_inj=1000             // ???
        Variable V_run=18000            // ???
        Variable C_dye=4.76e-11         // (mol/litre)  ???
        Variable L=41.5         // ??? cm
        Variable Nmolec
        Variable Vol_inj,Vol_cap
        Variable LODconc,LODmolec
        String LODprint

        WaveStats /R=(xcsr(A),xcsr(B)) $CsrWave(B)
        Vol_cap=(1.96e-8)*L             // (litre)
        Vol_inj=(t_inj*V_inj)/(t_mig*V_run)*Vol_cap
        Nmolec=Vol_inj*C_dye*(6.02e23)          //Numer of molecule injected
        LODconc=3*V_sdev/Hp*C_dye
        LODmolec=3*V_sdev/Hp*Nmolec
        LODprint= "LODc="+num2str(LODconc)+"   "+"LODm="+num2str(LODmolec)
        TextBox /F=0/Y=0 LODprint
End



// This program layout 8 graphs, then kill them

Macro layout8x1()
        string graphnames,onegraph
        Variable i=0

        Layout/C=1 Graph0,Graph1,Graph2,Graph3,Graph4,Graph5,Graph6,Graph7
        Tile/A=(8,1)
```

```
    Textbox/F=0/X=25 /Y=0"Filename:3/28/97-14"     // ???
    PrintLayout /D layout0

    //  kill all the graphs
    graphnames = WinList("Graph*",";","WIN:1")  //make a list of all graphs (names
    start with" Graph")
    do
        onegraph = GetStrFromList(graphnames,i,";")
        //print GetStrFromList(graphnames,0,";")
        DoWindow /K $onegraph
        i+=1
    while (i<8)

    // kill the layout
    DoWindow /K layout0
end
```

Procedure 4. Image Plot for Spectral Change with Time

```
// Macros:      DisplayAllWaves(n)
//              plot_image(b0,b1,b2,b3,b4,b5,b6,b7,b8,n)
// Functions:   Function display_all_waves(j)
//              ImageGraph(j)

// j: file number.         enter j=0, for w0-w9;        j=1  for w10-w19...
// To use,  load file , call Macro DisplayAllWaves,
//        record background of each wave (w9 as standard), then, call Macro plot_image

#pragma rtGlobals=1            // Use modern global access method.
#include <strings as lists>
#include "SetAxisRange"

Macro DisplayAllWaves(n)
    Variable n
```

```
        Prompt n, "file number (0 for w0-w9; 1 for w10-w19)"
        display_all_waves(n)
end



Function display_all_waves(j)
        Variable j
        String waveNames, onewave
        Variable i

        wavenames=WaveList("*",":","")
        onewave = GetStrFromList( wavenames,10*j,":")
        display $onewave                              // $ make a string to a wave
        Label left "Signal"
        SetAxis bottom 1000,1600            //??? (X Axis 1000-1600pts)
        i=1
        do
                onewave = GetStrFromList( wavenames,10*j+i,":")
                appendToGraph $onewave
                Smooth/s=2 9,$onewave      //??? (smooth S-G 2nd order 9 points)
                i = i + 1
        while (i <10)

        SetAxisRange()
        ShowInfo
end



Macro plot_image(b0,b1,b2,b3,b4,b5,b6,b7,b8,n)
        Variable b0,b1,b2,b3,b4,b5,b6,b7,b8,n
        Prompt n,"file number (0 for w0-w9; 1 for w10-w19)"
        Prompt b0, "Background of wave 0:"
        Prompt b1, "Background of wave 1:"
        Prompt b2, "Background of wave 2:"
        Prompt b3, "Background of wave 3:"
        Prompt b4, "Background of wave 4:"
        Prompt b5, "Background of wave 5:"
        Prompt b6, "Background of wave 6:"
        Prompt b7, "Background of wave 7:"
        Prompt b8, "Background of wave 8:"

        String wavenames, onewave, background
        Variable i

// correct background of all waves

        wavenames=WaveList("*",":","")
        i=0
        do
                onewave = GetStrFromList( wavenames,10*n+i,":")
                background = "b"+num2str(i)
                $onewave = $onewave-$background
                i=i+1
        while (i<9)              // w9 or w19 ... is standard, no offset
        SetAxisRange()

        ImageGraph(n)
end



Function ImageGraph(j)
        Variable j
        String wavenames, onewave
        Variable i

        wavenames=WaveList("*",":","")
        onewave = GetStrFromList( wavenames,10*j,":")
        Redimension/N=(6200,1) $onewave           //??? (total data point 6200)
        Display;AppendImage $onewave
        SetAxis bottom 1000,1600                    //???  (X Axis 1000-1600pts)
        ModifyImage $onewave ctab={9000,12000,Grays,0}

        i=1
        do
                onewave = GetStrFromList( wavenames,10*j+i,":")
                Redimension/N=(6200,1) $onewave            //??? (total data point 6200)
                AppendImage $onewave
                SetScale/P y i,1,"",$onewave
                ModifyImage $onewave ctab={9000,12000,Grays,0}
                                                //??? (image contrast)
                i=i+1
        while (i<10)
end
```

Appendix III. C Programs for Prism Calculation

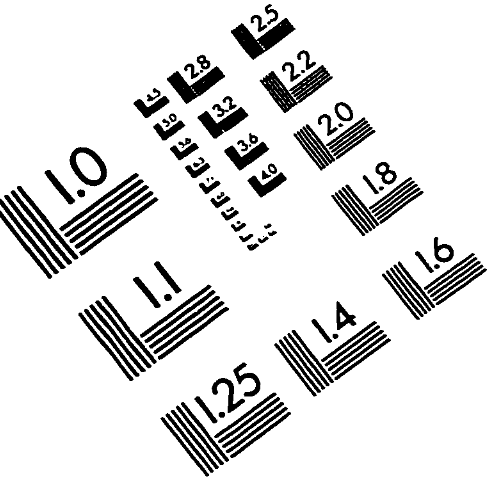
Calculation of Incident Angle for TIR and δmin

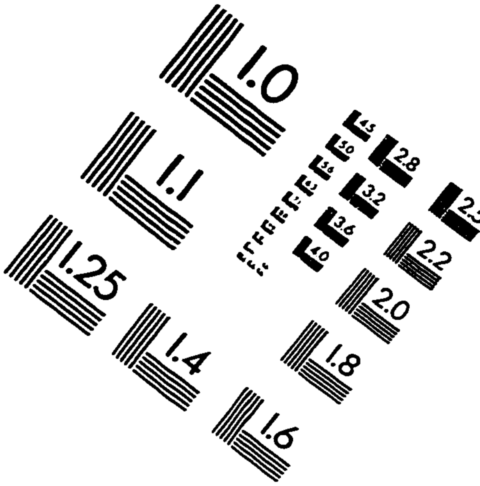
```
/* this program calculate the incident angle (φ) for total internal reflection( TIR)
 * and for minimum deviation (δmin), based on n = 1.517 (BK7).
 * value of n can be changed according to the material of prism or wavelength
 * program will ask for apex angle of prism in degrees
*/

#include <stdio.h>
#include <math.h>

main()
{       /*** TIR: the incident angle for TIR;  min: the incvident angle for δmin **/
        double    n, A=0, TIR = 0, B=0, min;

        n = 1.517;      /** change according to material and λ **/

        printf ("Enter angle of prism based on degrees A= ");
        scanf ("%lf", &A);

        while (A != 0)
        {
                A = A / 180 * 3.14159;
//              B = asin(0.5)/3.14159*180;
//              printf("arcsin0.5 = %f\n",B);
//              printf ("sin(A) = %f\n",sin(A));

                TIR = asin(n*sin(A-asin(1/n)));
                TIR = TIR / 3.14159 * 180;

                min = asin(n*sin(A/2) );
                min = min / 3.14159 * 180;
                printf ("refractive index n = 1.517\n");
        //      printf ("Apex angle of prism A = %f\n", A/3.14159*180);
                printf ("Incident angle for TIR = %f\n", TIR);
                printf ("incident angle for minimum deviation = %f\n", min);
                printf ("\n");
                printf ("Enter angle of prism based on degrees A = ");
                scanf ("%lf", &A);
        }
        printf ("calculation terminated by user\n");
}
```

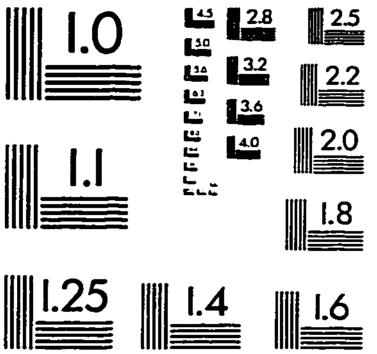
Calculation of dδ/dn at various incident angles

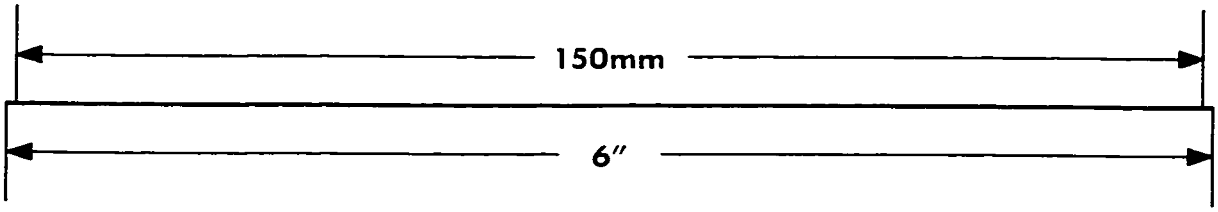
```
/* This program calculates dδ/dn (1st term in angular dispersion)
 *       for incident angle 5 to 85 degrees for various prisms of Apex and material
 * the results will be automatically saved in the file called "data"
 * program will ask for apex angle and material of the prism
 */

#include <stdio.h>
#include <math.h>
#define size 17

main()
{
        double          A, u=0, x=0, y=0, m=0, dmD=0;

/*** D: dδ/dn;  dD: (dδ/dn) at λ1 - (dδ/dn) at λ2;   mD: dδ/dn at minimum deviation ***/
        double          D[size][2], dD[size], mD[2];
        double          I[size] = {5, 10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 65, 70,
                                        75, 80, 85};
        int             i, j;
int             material;
double           n[2] = {0,0};
Boolean         Quit = false;
FILE            *f1;

f1 = fopen("data", "w");        /* filename = data, only "write" permission */
printf ("enter 1 for F2, 2 for BK7. prism material = ");
scanf ("%d", &material);
printf ("material = %d", material);

switch (material)
{       case 1:
                n[0] = 1.61656;
                n[1] = 1.62969;
        break;

        case 2:
                n[0] = 1.51509;
                n[1] = 1.52130;
        break;

        default:
                printf ("wrong material,try again ");
                Quit = true;
        break;
}

if (!Quit)
{       printf ("\nEnter angle of prism A =  ");
        scanf ("%lf", &A);
        printf ("A= %lf",A);
        A = A / 180 * 3.14159;
        m = 1/ sin(A/2);
        for (i=0; i<=1; i++)
                mD[i] = 2/sqrt(m * m - n[i] * n[i]);
        dmD = mD[1] -mD[0];
        for (j=0; j<= size - 1; j++)
            {       I[j] = I[j] /180 * 3.14159;
            }
        for (i=0; i <= 1; i++)
        {
            for (j=0; j<= size - 1; j++)
            {       x = n[i] * n[i] - sin( I[j] ) * sin( I[j] );
                    u = sin (A) * sqrt(x) - sin (I[j]) * cos (A);
                    y = x * (1- u * u);
                    D[j][i] = n[i] * sin( A )/sqrt(y);
            }
        }
        for (j=0; j<= size - 1; j++)
                    dD[j] = D[j][1] - D[j][0];
        if (material == 1)
            fprintf (f1, " F2, A = \t%10f\n", A/3.14159*180);
        else if (material == 2)
            fprintf (f1, " BK7, A = \t%10f\n", A/3.14159*180);
        else
            fprintf (f1, "wrong material, ignore all the calculated results\n");

        fprintf (f1,"\nD1 - 632.8nm - n1 = %10f\nD2 - 501.7nm - n2 = %10f\n",
                    n[0],n[1]);
        fprintf (f1,"D = dDeviation / dn\nmD = d(min. Deviation)/ dn\n
                    dD = D2 - D1\n\n");
        fprintf (f1, "%10s%10s%10s\n%10f\t%10f\t%10f\n\n",
                    "mD1", "mD2", "dmD", mD[0], mD[1], dmD);
        fprintf (f1,"%10s%14s%14s%14s\n", "I[j]", "D1", "D2", "dD");
        for (j=0; j<= size -1; j++)
        {
            fprintf(f1,"%10f\t%10f\t%10f\t%10f\n", I[j]/3.14159*180, D[j][0],
                    D[j][1], dD[j]);
        }
    }
    fclose(f1);
}
```

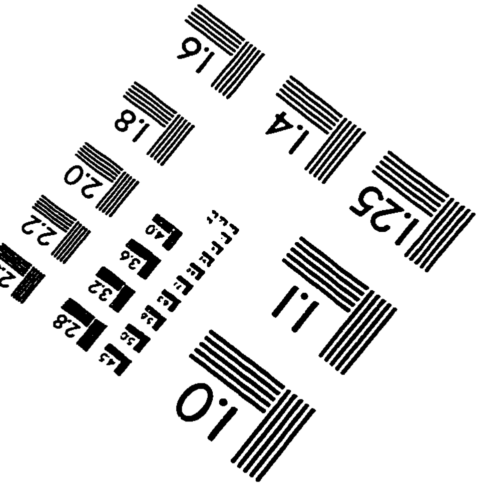
IMAGE EVALUATION TEST TARGET (QA-3)











APPLIED IMAGE, Inc
1653 East Main Street
Rochester, NY 14609 USA
Phone: 716/482-0300
Fax: 716/288-5989