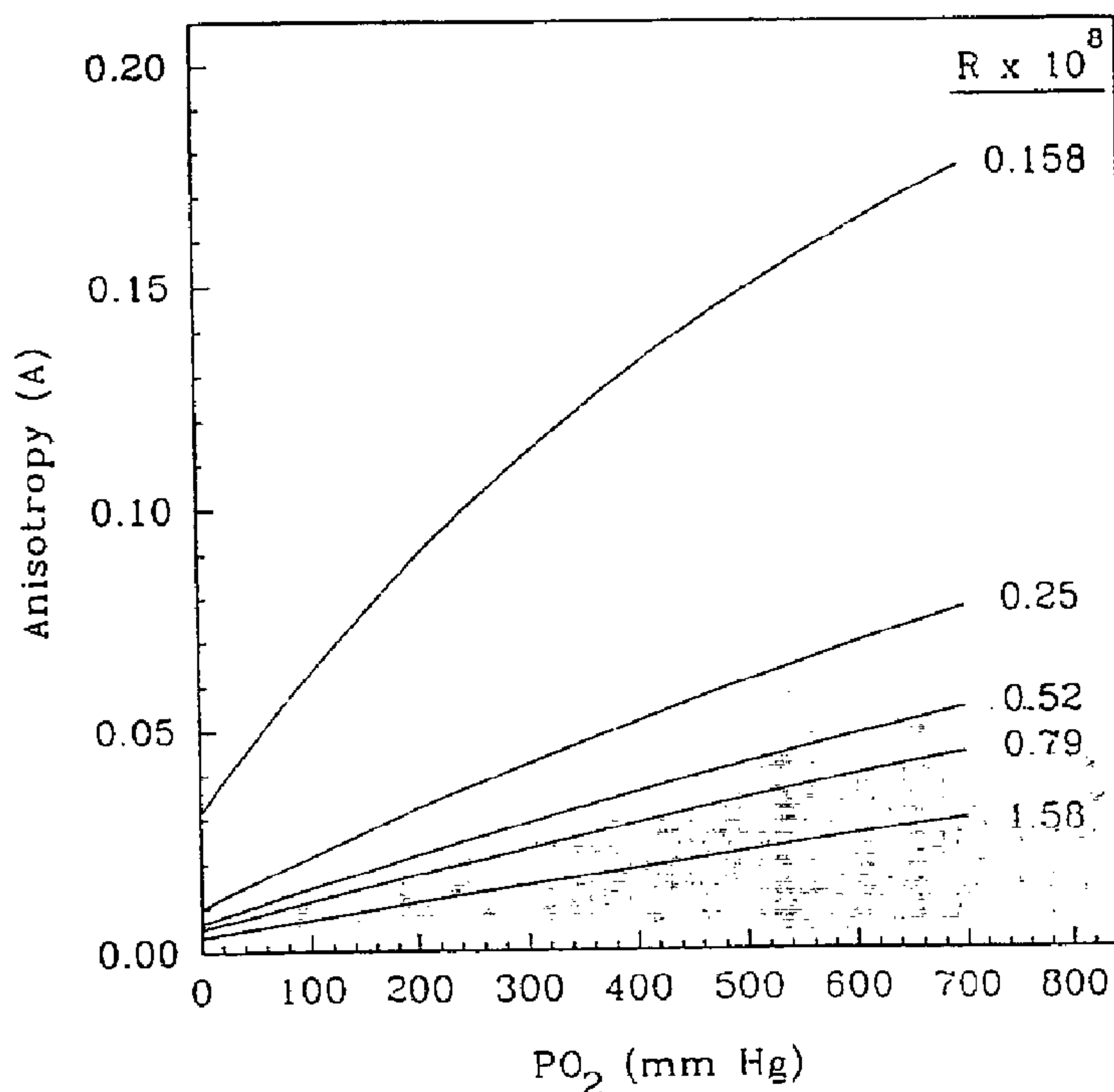




(86) Date de dépôt PCT/PCT Filing Date: 1995/04/19
(87) Date publication PCT/PCT Publication Date: 1995/11/02
(45) Date de délivrance/Issue Date: 2007/09/04
(85) Entrée phase nationale/National Entry: 1996/10/11
(86) N° demande PCT/PCT Application No.: US 1995/004772
(87) N° publication PCT/PCT Publication No.: 1995/028875
(30) Priorité/Priority: 1994/04/21 (US08/231,191)

(51) Cl.Int./Int.Cl. *G01N 21/64* (2006.01),
A61B 5/00 (2006.01), *A61B 6/00* (2006.01)
(72) Inventeur/Inventor:
ZUCKERMAN, RALPH, US
(73) Propriétaire/Owner:
ZUCKERMAN, RALPH, US
(74) Agent: FRENCH, DAVID J.

(54) Titre : DETECTION D'UN ANALYTE A PARTIR DE LA DUREE DE LUMINESCENCE A L'ETAT STATIONNAIRE
(54) Title: ANALYTE DETECTION FROM STEADY-STATE LUMINESCENCE LIFETIME



(57) Abrégé/Abstract:

Methods and apparatus determine analyte concentration in vivo and in vitro by the steady state determination of luminescence lifetime. A fluoro-phore that is quenched by the analyte is free to undergo Brownian rotation. The fluoro-phore (12) is irradiated with continuous linearly polarized light (24). Emitted luminescence is resolved into vector components parallel and perpendicular to the plane of polarization of the excitation light (36, 38). A mathematical function is employed which relates the luminescence anisotropy to quencher concentration. For analytes which do not quench excited states, a known quantity of the analyte is conjugated to a quencher molecule or energy transfer acceptor, and a competition reaction is set up in which labelled and unlabelled analytes compete for sites on a labelled carrier molecule. An empirically determined mathematical function is employed which relates luminescence anisotropy at the carrier label emission band to analyte concentration.



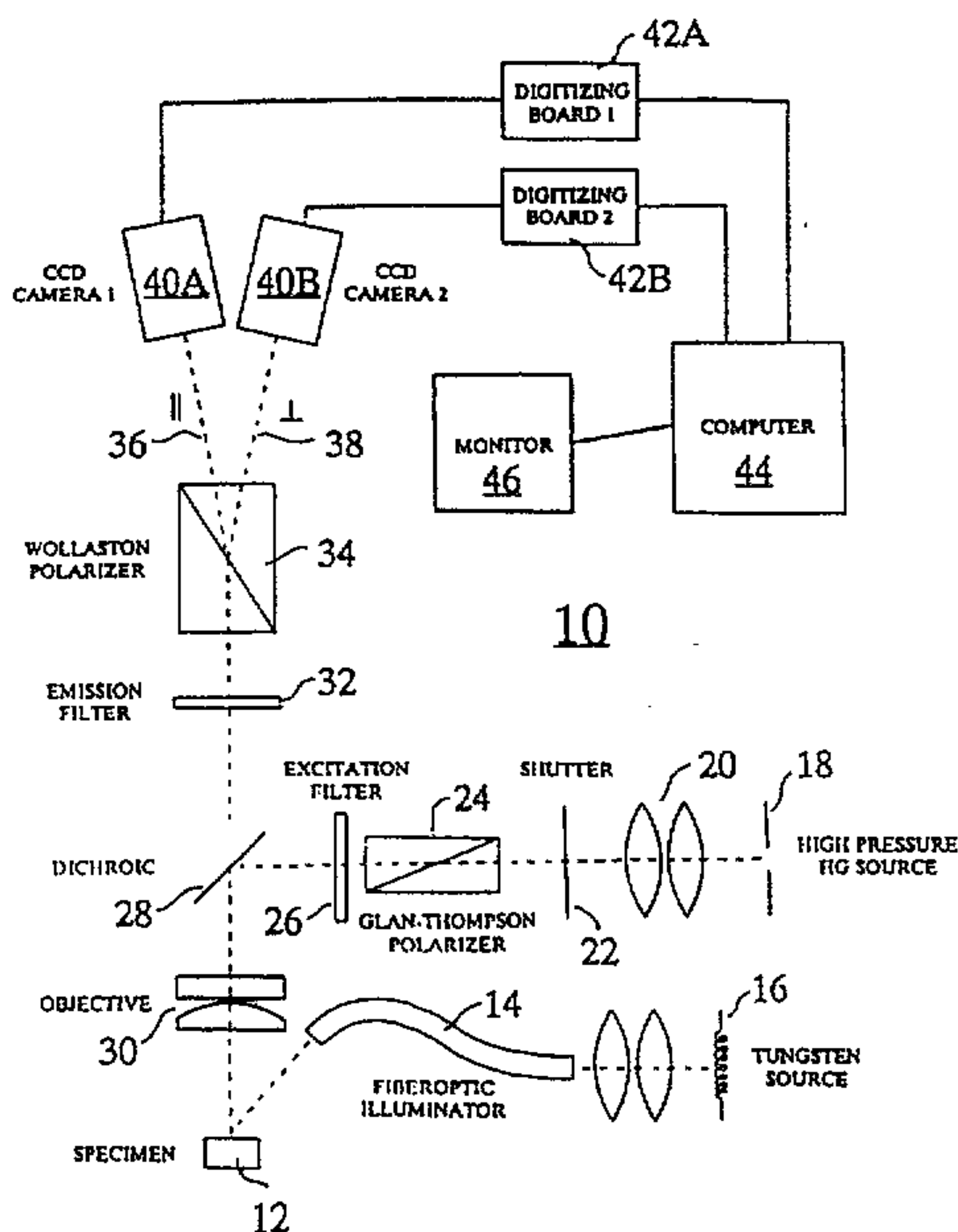
INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification ⁶ : A61B 5/00		A1	(11) International Publication Number: WO 95/28875
			(43) International Publication Date: 2 November 1995 (02.11.95)
(21) International Application Number: PCT/US95/04772 (22) International Filing Date: 19 April 1995 (19.04.95) (30) Priority Data: 08/231,191 21 April 1994 (21.04.94) US (60) Parent Application or Grant (63) Related by Continuation US 08/231,191 (CIP) Filed on 21 April 1994 (21.04.94)		(81) Designated States: AU, BB, BG, BR, BY, CA, CN, CZ, FI, HU, JP, KP, KR, KZ, LK, MG, MN, MW, MX, NO, NZ, PL, RO, RU, SD, SK, UA, US, VN, European patent (AT, BE, CH, DE, DK, ES, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, ML, MR, NE, SN, TD, TG). 2187735 Published <i>With international search report. Before the expiration of the time limit for amending the claims and to be republished in the event of the receipt of amendments.</i>	
(71)(72) Applicant and Inventor: ZUCKERMAN, Ralph [US/US]; 226 West Rittenhouse Square, Philadelphia, PA 19103 (US). (74) Agent: EILBERG, William, H.; 820 Homestead Road, P.O. Box 7, Jenkintown, PA 19046 (US).			

(54) Title: ANALYTE DETECTION FROM STEADY-STATE LUMINESCENCE LIFETIME

(57) Abstract

Methods and apparatus determine analyte concentration *in vivo* and *in vitro* by the steady state determination of luminescence lifetime. A fluoro-phore that is quenched by the analyte is free to undergo Brownian rotation. The fluoro-phore (12) is irradiated with continuous linearly polarized light (24). Emitted luminescence is resolved into vector components parallel and perpendicular to the plane of polarization of the excitation light (36, 38). A mathematical function is employed which relates the luminescence anisotropy to quencher concentration. For analytes which do not quench excited states, a known quantity of the analyte is conjugated to a quencher molecule or energy transfer acceptor, and a competition reaction is set up in which labelled and unlabelled analytes compete for sites on a labelled carrier molecule. An empirically determined mathematical function is employed which relates luminescence anisotropy at the carrier label emission band to analyte concentration.



ANALYTE DETECTION FROM STEADY-STATE LUMINESCENCE LIFETIME

BACKGROUND OF THE INVENTION

5 Molecular oxygen is a critical requirement for cellular function in animals, while
the protection from O_2 is required for the function of plant enzymes such as the
nitrogenases. Animals, and mammals in particular, are critically dependent upon the
continuous supply and utilization of O_2 in the processes which maintain life. This is
10 particularly true for high O_2 -consuming tissues such as neural tissues and muscle, both
striated and smooth. Interruption of oxygen delivery to these tissues, for times as brief
as minutes, can result in cell death and loss of functions critical to organ function.
Moreover, it is the oxygen concentration within tissues, rather than within blood or other
bodily fluids, which ultimately supports cellular function. Therefore, the measurement
of bodily fluid and tissue oxygen concentrations is of pivotal importance to clinical
15 medicine, with the compromise of oxygen delivery to tissues occurring in a host of
vascular diseases, such as arteriosclerosis, diabetes, sickle cell disease, and impaired
wound healing to name a few. It is therefore not surprising that numerous patents have
been issued for methods and devices which measure blood and bodily fluid oxygen
concentration, and this inventor's U.S. Patent No. 5,186,173 (hereinafter the Zuckerman
20 '173 patent) for the first measurement of tissue oxygen concentration *in vivo*.

 These patents, although each addressing problems involved in the noninvasive
determination of bodily fluid or tissue PO_2 , suffer from deficiencies which have
precluded their widespread application in clinical medicine. To achieve the requirement
of noninvasive measurement recent patents have turned to optical methods, which
25 involve the quenching of phosphorescence or fluorescence of a dye by dioxygen. For
example, patents such as that of U.S. Patent No. 4,476,870 (Peterson *et. al.*) have
developed catheter designs in which a fluorescent substance, such as perylene, is housed
within a sealed catheter which may be inserted into blood vessels and its fluorescence
quenching by O_2 in blood measured by use of fiber optics. This device, although
30 providing the ability to measure bodily fluid or blood PO_2 , suffers from two deficiencies.
First, the PO_2 of blood may be measured accurately only after the catheter probe is
externally calibrated prior to its insertion into a blood vessel; and second, the technique
provides no topographic information, as PO_2 is measured at a single locus, *viz.*, at the

probe tip. In U.S. Patent No. 4,810,655, Khalil *et. al.* purport to overcome the need for prior calibration of the catheter before its every usage by measuring phosphorescence lifetime instead of intensity. Although in theory a direct lifetime system should overcome the need for prior calibration, in practice, as shown in Table 4 of the Khalil *et. al.* patent, the phosphorescence lifetimes of the porphyrins employed change with light exposure, making calibration prior to use still necessary. In addition, the time-resolved direct lifetime system would be cumbersome to implement, and less precise than a steady-state approach. Vanderkooi and Wilson (U.S. Patent No. 4,947,850) developed a procedure based upon the determination of the phosphorescence lifetime of an O₂-sensitive probe substance, such as a metallo-porphyrin bound tightly to albumin, which phosphoresces on a timescale of fractions of a millisecond (10⁻³ sec), and whose phosphorescence lifetime is reduced (quenched) by dioxygen. Here the O₂-sensitive probe is a phosphorescing molecule which may be injected into the blood stream, thereby permitting topographic determination of the PO₂ of blood within the vasculature of an imaged tissue. However, due to self-quenching of the metallo-porphyrins, as well as the other probes described by Vanderkooi, these probe molecules cannot be used alone. That is, self-quenching results in probe concentration dependent changes in decay time which are greater than those induced by molecular oxygen. Since it is impossible to know the precise probe concentration in the blood, due to leakage at the injection site and variations in blood volume in different animals or humans, they must modify the probe to eliminate self-quenching. As stated in U.S. Patent No. 4,947,850 (col. 3, line 15) "porphyrins are preferably employed and said compositions are preferably admixed with proteinaceous compositions which bind with the phosphorescent composition.... Albuminous...compositions are preferred." As described in their publication Vanderkooi, J. *et. al.*, "An Optical Method for Measurement of Dioxygen Concentration Based Upon Quenching of Phosphorescence", *J. Biol. Chem.*, 262(12): 5476-5482 (April 1987), the technique will simply not work without albumin or some other large molecular mass protein bound to the probe molecule. When injected intravenously into the blood the large molecular mass protein (M.W.=67,000) prevents the probe molecule from passing through the small junctions or lipid membranes of the blood vessel wall, and thereby limits the measurement to oxygen concentration in the blood. Similarly, U.S. Patent No. 4,579,430 (Bille) discloses an invention that allows the determination of oxygen

saturation, percentage binding of O_2 to blood hemoglobin, within retinal vessels. In a host of diseases the oxygen saturation and oxygen concentration of the blood, as revealed by previous patents, remains normal although the tissue is believed to become hypoxic. Such diseases include diabetes, retinopathy of prematurity, and hypertensive and arteriosclerotic diseases. Similarly, during long surgeries, such as bypass surgery, carotid artery surgery and during prolonged intensive care, the oxygen saturation and/or concentration of the blood is carefully monitored and maintained, yet moderate to severe brain damage due to hypoxia has been known to occur. It is the oxygen concentration within tissues which is relevant to its functioning and health, and tissue oxygen concentration depends upon tissue oxygen consumption rate, blood velocity, and vessel caliber, in addition to blood oxygen concentration or saturation. Therefore, it is of paramount importance to be able to measure the PO_2 of tissue in space and time.

The noninvasive, topographic measurement of tissue PO_2 was addressed in the the Zuckerman '173 patent. In this patent, a highly lipid soluble probe substance, such as sodium pyrenebutyrate, is injected intravenously or intraperitoneally, or applied topically when appropriate. The lipid soluble, biocompatible probe substance leaves the blood and accumulates within the lipid bilayers of tissue cells. Here the probe is essentially the tissue itself, once pyrenebutyrate accumulates within the lipid bilayers of its cells. This and the digital imaging system detailed within the patent allows the first topographic determination of tissue oxygen concentration, which supports tissue health and function. The concentration of O_2 is determined in the Zuckerman '173 patent by the measurement of fluorescence intensity, according to the Stern-Volmer equation, written in terms of fluorescence intensities. The invention of the Zuckerman '173 patent, although allowing the first noninvasive determination of tissue PO_2 in space and time, suffers from two deficiencies which preclude its clinical application, and which presently limits its utility to research applications on laboratory animals. The most significant limitation results from the fact that fluorescence intensity is determined by both the concentration of pyrenebutyrate in space within tissues, which varies due to spatial differences in lipid composition of the cells within tissues, as well as by the spatial distribution of tissue oxygen concentration. To circumvent this problem, and to extract the oxygen concentrations at a plurality of locations, the fluorescence intensity at each locus is ratioed against the fluorescence intensity of the same location when the tissue

is brought to a PO_2 of 0 mm Hg (in the absence of oxygen). This may be accomplished by breathing the animal on 100% N_2 at the end of the experiment. Such a procedure cannot be applied clinically on humans as it would undoubtedly result in cell death, evidenced in brain damage or death. The second deficiency in the invention of the Zuckerman '173 patent resides in the optical filtering effects of blood on the fluorescence intensities at the emission wavelength of pyrenebutyrate. This may similarly be corrected in a laboratory situation by the separate measurement of the optical density of blood within the vasculature of the imaged tissue.

Both of the deficiencies in the device and method of the Zuckerman '173 patent may be obviated by the direct measurement of fluorescence lifetime (decay) instead of fluorescence intensity. Measurement of fluorescence lifetime at a given PO_2 and within a given tissue, once determined in a laboratory calibration of the instrument, would obviate the need to bring the tissue to a PO_2 of 0 mm Hg in the clinic, as the fluorescence decay constant is independent of pyrenebutyrate concentration, and is similarly unaffected by the absorbance of blood within the vasculature. However, fluorescence lifetime occurs on the timescale of nanoseconds (10^{-9} sec), and in addition to the cumbersome and expensive apparatus required directly to measure such short fluorescence lifetimes at a plurality of locations, would require intense pulsed laser excitation (pulse duration <20 nanoseconds) of the tissue at the ultraviolet excitation wavelength to achieve usable signal-to-noise ratios. Since the quantum efficiency of pyrenebutyrate and other fluorescent probe molecules is generally less than 0.5, photon absorption which is not converted into emitted radiation by fluorescence would be converted to heat, thereby causing tissue damage. Tissue damage subsequent to an intense laser pulse, used in a time-resolved direct fluorescence lifetime determination, is the major drawback to the use of this approach *in vivo*. Herein resides the need for a method for the topographic measurement of oxygen concentration *in vivo* by a steady-state, rather than time-resolved, procedure which provides an indirect determination of fluorescence lifetime.

In view of the deficiencies of the prior art, it would be desirable to have a method which can be applied in numerous forms and which allows the determination of bodily fluid, blood, and tissue PO_2 in space and time by an indirect determination of fluorescence lifetime. Such a method would allow the construction of an O_2 -sensitive

catheter, which may be inserted into a blood vessel, and which does not require cumbersome external calibration prior to its every usage. Similarly, such a method would, by intravenous injection of a biocompatible lipid soluble fluorescent dye, and by the application of digital image processing techniques, allow the topographic determination of tissue PO_2 , as well as the determination of blood PO_2 within the vasculature of the imaged tissue. By extension, the application of optical serial sectioning methodologies would allow the first determinations of tissue and blood PO_2 distributions in three dimensions, tomographically within a volume of imaged tissue. Most importantly, such a procedure would be eminently suitable for clinical applications involving diagnosis and treatment of vascular, metabolic and other diseases in humans. Moreover, a methodology for the indirect, steady-state determination of luminescence lifetimes would be equally applicable to the measurement of the concentrations of numerous other substances of biologic and physical importance, in addition to dioxygen, in time and space.

OBJECTS OF THE INVENTION

Accordingly, it is a general object of this invention to provide an apparatus and methods for the determination of the concentration and distribution in space of numerous substances of biologic and physical import *in vivo* and *in vitro* by the steady-state determination of luminescence lifetime.

It is a further object of the invention to provide a method and apparatus for the *in vivo* topographic determination of tissue and bodily fluid oxygen concentration or PO_2 within an imaged tissue, as well as a method and apparatus for measuring blood or bodily fluid PO_2 with a fiber optic catheter which overcomes the disadvantages of the prior art.

It is a further object of this invention to provide an *in vivo* method and apparatus for determining tissue and bodily fluid oxygen concentration using a fluorescent probe substance whose fluorescence lifetime is quenched by molecular oxygen, and oxygen concentration or PO_2 is determined by measuring the fluorescence anisotropy of the fluorescent probe substance.

It is another object of this invention to provide an *in vivo* method and apparatus for determining tissue and bodily fluid oxygen concentration which is accurate, non-intrusive and provides reproducible results.

SUMMARY OF THE INVENTION

These and other objects of this invention are achieved by providing an apparatus and methods for the determination of the concentration and distribution of numerous substances of biologic and physical import, *in vivo* and *in vitro*, by means of the steady-state determination of luminescence lifetime. In the general instance, a fluorophore whose excited state is quenched by the substance in question is free to undergo Brownian rotation alone or when conjugated to a carrier molecule within a medium of suitable viscosity. The analysis medium is irradiated with continuous linearly polarized light at a wavelength strongly absorbed by the fluorophore. The emitted fluorescence is resolved into its vector components parallel and perpendicular to the plane of polarization of the excitation light, thereby permitting the calculation of the fluorescence anisotropy of the irradiated specimen. The concentration of the quencher is determined by applying a mathematical function which relates the fluorescence anisotropy of the fluorophore to the concentration of the quencher. For the determination of the concentration of substances which do not themselves quench excited states a quantity of the substance under measurement is conjugated to a quencher molecule or an energy transfer acceptor molecule and a competition reaction is set up in which the luminescently labelled substance and the unlabelled substance within the sample compete for sites on a labelled carrier molecule. To the extent that the concentration of unlabelled substance in the sample increases and displaces labelled material on the carrier molecule the luminescence lifetime of the carrier fluorophore will increase. The fluorescence anisotropy is measured at the emission band of the luminescent label on the carrier molecule and the concentration of the substance under measurement is determined by applying an empirically determined mathematical function which relates fluorescence anisotropy to the concentration of the substance being measured.

In the specific instance of the quencher dioxygen the objects of the invention are met by providing an apparatus and methods for the *in vivo* topographic determination of tissue and bodily fluid oxygen concentration or PO_2 within an imaged tissue, as well as a method for measuring blood or bodily fluid PO_2 with a fiber optic catheter. In the first method, a lipid soluble, biocompatible fluorescent probe substance is administered to an animal body and accumulates within the lipid bilayers of its tissue cells. In the second method, the fluorescent probe substance is conjugated to a large molecular mass

protein which causes it to be retained within a bodily fluid such as blood. In the third method describing a catheter design, in which PO_2 is measured at the catheter tip, the tip contains the fluorescent probe substance dissolved in a viscous nonpolar solvent. In all methods described, a fluorescent probe substance is preferred whose fluorescence lifetime is quenched by molecular oxygen, and oxygen concentration or PO_2 is determined by measuring the fluorescence anisotropy of the fluorescent probe substance. The bodily fluid or tissue containing the biocompatible fluorescent probe substance, or the fluorescent probe substance in a nonpolar solvent within a catheter tip, is irradiated with continuous linearly polarized ultraviolet light at a wavelength strongly absorbed by the fluorophore. The emitted fluorescence is resolved into its vector components parallel and perpendicular to the plane of polarization of the excitation light, thereby permitting the calculation of the fluorescence anisotropy of the irradiated specimen. Tissue and/or bodily fluid PO_2 of the imaged specimen, or the PO_2 of a fluid at the tip of a sealed catheter, is determined by applying a mathematical function which relates the fluorescence anisotropy of an O_2 -quenchable fluorophore to oxygen concentration or partial pressure.

BRIEF DESCRIPTION OF THE DRAWINGS

Other objects and many attendant features of this invention will become readily appreciated as the same becomes better understood by reference to the following detailed description when considered in connection with the accompanying drawings wherein:

Fig. 1 is a theoretical plot of the fluorescence anisotropy of an O_2 -quenchable fluorophore dissolved in a nonpolar solvent as a function of the oxygen partial pressure (PO_2) of the fluorophore-containing solution. A family of curves is derived from equation 6 for the range of Brownian rotational motions (R) known to occur in the lipids of living tissue cells. Fluorescence anisotropy increases monotonically with increasing PO_2 at all rotational rates.

Fig. 2 is a schematic illustration of an imaging apparatus used to determine the topographic distribution of tissue and bodily fluid PO_2 within an imaged specimen.

Fig. 3 is an apparatus of the present invention used to measure the oxygen concentration or PO_2 of a fluid contacting the tip of a sealed fiber optic catheter.

Fig. 4 is a schematic illustration of a data processor (of microprocessor design) which may be used to implement the calculations necessary to determine PO₂ at the catheter tip. MUX is a multiplexer; T is temperature, and G is an empirical constant used to correct for the dichroic mirror's transmission efficiency in the parallel and perpendicular planes for linearly polarized light. I₁ and I₂ are the resolved intensities of fluorescence emission respectively parallel and perpendicular to the plane of polarization of the excitation light.

Fig. 5 is a graphical illustration of PO₂ levels measured parallel (upper) and perpendicular (lower) to a retinal arteriole within the retina of an isolated and perfused bovine eye. The longitudinal PO₂ drop parallel to the arteriole is fit (solid line) by the indicated theoretical function, while the PO₂ measured perpendicular to the arteriole is fit (solid curve) by the Krogh mathematical model. R² is the coefficient of determination.

Exhibit A is the source code excerpted from "Software for the Acquisition, Manipulation and Analysis of Video Images", Copyright 1993, Biometric Imaging, Inc..

DETAILED DESCRIPTION OF PREFERRED EMBODIMENTS

The present invention addresses several problems inherent in the prior art. The invention is based upon the well-accepted physical principle that absorption and emission of photons by fluorophores occur via electric dipole transition moments which have well-defined orientations within the molecular frame of the fluorophore. In a nonpolar solvent or lipid bilayer containing randomly oriented fluorophores only suitably oriented molecules can be excited by linearly polarized light, thus creating a nonequilibrium orientation of excited and fluorescing molecules. This anisotropy, and its rotational relaxation, are reflected in the polarization of the emitted light. The case of time-resolved fluorescence anisotropy induced by a pulse of linearly polarized light may be described by the following equation:

$$A(t) = A_0 e^{-6\bar{R}t} \quad (1)$$

where A is fluorescence anisotropy, t is time of observation after pulsed excitation, $\bar{R}$ is mean molecular rotation time in radians/sec, and A₀ is the fluorescence anisotropy in the "frozen" state, in the absence of Brownian rotation. However, if fluorescence anisotropy

is measured with steady-state, continuous excitation, A is an average of the time-resolved decay over time weighted by the decay of intensity:

$$A(\tau) = \frac{\int_0^{\infty} e^{-t/\tau} A(t) dt}{\int_0^{\infty} e^{-t/\tau} dt} = \frac{1}{\tau} \int_0^{\infty} e^{-t/\tau} A(t) dt \quad (2)$$

where τ is the fluorescence lifetime of the fluorophore.

Carrying out the transform results in the Perrin equation:

$$A = \frac{A_0}{1 + 6\bar{R}\tau} \quad \text{or} \quad \frac{A_0}{A} = 1 + 6\bar{R}\tau \quad (3)$$

which for an ideal spherical molecule becomes:

$$\frac{A_0}{A} = \frac{1 + R_g T}{\eta V} \tau \quad (4)$$

where R_g is the gas constant, T is temperature, η is viscosity, and V is the fluorophore's hydrodynamic volume.

The present invention considers, for the first time, the case of an O_2 -quenched fluorophore, such as sodium pyrenebutyrate, accumulated within a nonpolar solvent or lipid bilayer, such as those found in all tissue cells in the body. The fluorescence lifetime of the fluorophore may be related respectively to oxygen concentration or partial pressure by the following forms of the Stern-Volmer relation:

$$\tau = \frac{\tau_0}{1 + K_D[O_2]} \quad \text{or} \quad \frac{\tau_0}{\tau} = 1 + \alpha K_D PO_2 \quad (5)$$

where α is the Bunsen solubility coefficient, K_D is the dynamic quench constant, and τ_0 is fluorescence lifetime in the absence of O_2 . By combining equations 3 and 5, we derive a mathematical relationship which formally relates oxygen partial pressure to steady-state fluorescence anisotropy:

$$PO_2 = \frac{A_0 - A(6\bar{R}\tau_0 + 1)}{\alpha K_D(A - A_0)} \quad (6)$$

Equation 6, therefore, is the central and novel concept of the present invention, and demonstrates that steady-state fluorescence anisotropy measurements may be used as a precise procedure, which may be implemented in simple configurations, to determine (i) the PO_2 of tissues in space and time subsequent to the accumulation of an O_2 -quenchable fluorophore within the lipid bilayers of its tissue cells, (ii) the topographic distribution of blood PO_2 , subsequent to the accumulation of the fluorophore within the lipid bilayers of red blood cells, or for a fluorophore conjugated to a protein injected into the blood stream and (iii) the PO_2 at the tip of an insertable catheter whose tip contains a fluorophore such as pyrene in a nonpolar solvent or lipid such as mineral or paraffin oils, separated from the sample by an O_2 -permeable membrane.

The mathematical relation expressed in equation 6 is plotted in Figure 1 for the range of Brownian rotational motions known to occur in the lipids of tissue cells. As shown in Fig. 1, steady-state fluorescence anisotropy increases in a systematic way with increasing values of PO_2 at all rotational rates. In other words, as PO_2 is raised the fluorescence lifetime of the fluorophore is shortened (equation 5), thereby reducing the angle swept out by the fluorophore during its lifetime. Fluorescence anisotropy, therefore, increases with increasing PO_2 . The other variables which would affect fluorescence anisotropy in such a situation, in addition to PO_2 , are temperature and viscosity of the lipid bilayer. However, it is well known that in the case of an imaged tissue and/or vasculature *in vivo*, the body superbly regulates these variables. Similarly, a catheter with an O_2 -sensitive tip of small volume will quickly come into equilibrium with the temperature of the blood within a blood vessel into which it is inserted. Moreover, as demonstrated in equation 4, small variations in body or catheter tip temperature encountered in some disease states may be compensated for precisely by concomitant temperature measurement and the application of this equation to the calculation of PO_2 from steady-state fluorescence anisotropy measurements.

The present invention has all of the advantages of a direct fluorescence lifetime system, without the need for intense, pulsed laser excitation and its potentially damaging effects on the imaged specimen. In addition, the cumbersome and expensive equipment required to measure fluorescence decays in the 40-135 nanosecond range are avoided. Similarly, since steady-state, continuous fluorescence anisotropy determinations are the time average of a huge number of fluorescence decay events, such steady-state

measurements must be more precise than any time-resolved procedure. The present invention determines PO_2 by measurement of steady-state fluorescence anisotropy, while in U.S. Patent No. 4,476,870 (Peterson) as well as Zuckerman '173 patent, fluorescence intensity is the variable measured. Furthermore, the present invention differs from the invention disclosed in U.S. Patent No. 4,810,655 (Khalil *et. al.*) and that of U.S. Patent No. 4,947,850 (Vanderkooi) in that both patents determine PO_2 from direct measurements of the lifetimes of phosphorescent substances. Phosphorescing substances have much longer, and therefore easier to measure, lifetimes than the preferred fluorescing substances employed in this invention. In the present invention, an indirect measure of fluorescence lifetime, viz. fluorescence anisotropy, is employed in simple-to-implement configurations to provide measurements of tissue and bodily fluid PO_2 *in vivo*, without the limitations of the prior art.

Although presented above for the specific case of a methodology for the determination of oxygen concentration or partial pressure, the mathematical derivation which relates the concentration of the quencher dioxygen to fluorescence anisotropy may be rewritten for the general case of any fluorophore whose excited state may be quenched, and the concentration of the quencher thereby determined. Considering the more general case, equation 5 may be rewritten for any fluorophore whose fluorescence is quenched by a substance according to the Stern-Volmer relation, as shown below:

$$\tau = \frac{\tau_0}{1 + K_D[Q]} \quad (7)$$

where $[Q]$ is the concentration of the quencher of a given fluorophore. Similarly, equation 6 may then be rewritten for the relationship of fluorescence anisotropy to quencher concentration as follows:

$$[Q] = \frac{A_0 - A(6\bar{R}\tau_0 + 1)}{K_D(A - A_0)} \quad (8)$$

Consequently the present invention, although presented for the specific case of the quencher dioxygen, should be viewed as a general methodology for the determination of the concentration of any quencher of excited states. In practice, the resolution of the technique may be optimized for a given fluorophore and quencher by adjusting the

viscosity of the medium, and therefore the rotational velocity of the fluorophore, to the fluorescence lifetime of the fluorophore in the absence of quencher. This may be accomplished for a lipid soluble fluorophore by choosing a lipid of appropriate viscosity, or for a nonlipid soluble fluorophore by conjugating the fluorophore to a carrier molecule of sufficient hydrodynamic volume in a medium whose viscosity is manipulated by adjusting glycerol concentrations in a glycerol-water mixture or sucrose concentration in a sucrose-water medium. Although not meant to be an inclusive list of the classes of fluorophores and quenchers to which the invention may be applied, one might consider the steady-state measurement of pH for the fluorophore fluorescein, or a fluorescein derivative, whose fluorescence lifetime is quenched by protons. Similarly, chloride concentrations may be determined using the same steady-state methodology employing the fluorophore quinine, or the concentration of iodide determined using the fluorophore γ -pyrenebutyric acid. It will then be obvious to one skilled in the art that the general methodology presented herein may be applied for the detection of a host of ion and other quencher concentrations by selecting the appropriate fluorophore-quencher combination and, by manipulating viscosity, thereby adjusting the rotational velocity of the fluorophore to the range of fluorescence lifetimes encountered over the desired quencher concentration range.

In addition to the application of the present invention to the determination of the concentrations of quenchers of excited states, the methodology may be further extended to encompass the determination of the concentration of any substance of interest in a sample. In this embodiment of the invention, a fluorophore is conjugated to larger carrier molecule which has a high affinity for the the substance in question. Similarly, a quantity of the substance to be measured is conjugated to a quencher of the excited state of the carrier fluorophore. The fluorescently-labelled substance along with the sample, containing a quantity of the unlabelled substance to be determined, is added to the analysis mixture, resulting in a competition for sites on the carrier molecule. This competition will displace quencher-labelled molecules from the carrier, thereby increasing the fluorescence lifetime of the carrier fluorophore. An increase in the lifetime of the carrier fluorophore will in turn reduce the fluorescence anisotropy of the sample measured at the emission wavelength of the carrier fluorophore. Therefore, in this embodiment of the invention fluorescence anisotropy will decrease with increasing

sample concentrations of the substance whose concentration is to be determined. In other words, as the concentration of the substance under analysis is raised the number of quenchers on the carrier is reduced, thus increasing the fluorescence lifetime of the carrier fluorophore (equation 7), thereby increasing the angle swept out by the carrier fluorophore during its lifetime. Fluorescence anisotropy, therefore, decreases with increasing concentration of the substance under measurement.

An interesting choice for the two fluorophores employed in this methodology would represent energy donor and acceptor molecules capable of Förster energy transfer. The basic paradigm of Förster energy transfer involves the excitation of the donor molecule with light of a wavelength lying within the absorption region of the donor. A small amount of the excitation energy is given up to the surroundings in the form of vibrational energy of the molecule (transition to the lowest vibrational state of the electronically excited molecule); the remainder can be given off either through emission of fluorescent light or by energy transfer to the acceptor molecule. The efficiency of the energy transfer depends upon the quantum yield of the donor, the degree of overlap between the emission spectrum of the donor and the absorption spectrum of the acceptor, and the orientation and distance between the donor and acceptor. Depending upon the extent to which the above conditions are met, the approximation of the acceptor and donor molecules in space will result in a shortening of the fluorescence lifetime of the donor fluorophore. When donor-acceptor pairs are used according to the method described above, fluorescence anisotropy measured at the emission wavelength of the donor will decrease with increasing concentration of the measured substance in the sample. A significant advantage of employing Förster energy transfer donor-acceptor pairs is the ability to shift the absorption band of the donor to longer wavelengths, thereby permitting the use of less expensive light sources such as light emitting diodes (LEDs) and laser diodes.

The embodiment of the invention employing energy transfer donor-acceptor pairs will be illustrated by two nonlimiting examples. Consider first a detection system designed to detect the quantity of a hormone (protein). A polyclonal antibody raised against the protein is labelled with FITC (fluorescein isothiocyanate), and thus serves as the donor, while a quantity of the protein is labelled with malachite green isothiocyanate, thereby providing a suitable acceptor. These labelled molecules, along with the

unlabelled protein within the sample, are admixed in a glycerol-water buffer to provide the appropriate viscosity. The mixture is illuminated with linearly polarized light at the absorption wavelength for FITC, and the fluorescence anisotropy of the sample determined at the emission wavelength of FITC (donor emission). As the quantity of the unlabelled protein (hormone) in question increases in the sample it will compete off protein labelled with acceptor fluorophores, thereby increasing donor fluorescence lifetime. Therefore, the fluorescence anisotropy at the FITC emission band (donor wavelength) will decrease systematically as a function of increasing unlabelled protein concentration in the sample. As a second example, consider a system designed to measure blood glucose levels. In this case one might employ concanavalin A (ConA) as the carrier molecule, which has a high affinity for glucose, labelled with Cascade Blue as the donor fluorophore and a quantity of fluorophore-labelled glucose such as 6-NBD-glucosamine (6-(N-(7-nitrobenz-2-oxa-1,3-diazol-4-yl)amino)-6-deoxyglucose) as the acceptor. In this system, glucose in the blood will compete off fluorescently labelled glucose thereby resulting in a decrease in the fluorescence anisotropy measured at the Cascade Blue emission band. Although two nonlimiting examples have been provided it will be obvious to one skilled in the art that the basic method disclosed herein may be adapted to determine the concentrations of a host of substances such as saccharides, DNA fragments, hormones, drugs, and immunoglobulins.

The following figures depict embodiments of the present invention for the topographic measurement of tissue PO_2 and the topographic determination of the distribution of blood PO_2 within the vasculature of an imaged tissue. In addition, a catheter design, which does not require calibration prior to every usage is also disclosed. It would be readily appreciated by one skilled in the art that in view of this disclosure, however, that other specimens, biologic or physical, might be assessed with respect to their concentrations and distributions in space by variations of the designs herein disclosed using the method of the present invention.

As shown below, fluorescence anisotropy is operationally defined in the general case, and formally described in the specific case for the quencher dioxygen, by the

following equation:

$$A = \frac{I_{\parallel}G - I_{\perp}}{I_{\parallel}G + I_{\perp}} = \frac{A_0(\alpha K_D PO_2 + 1)}{6\bar{R}\tau_0 + \alpha K_D PO_2 + 1} \quad (9)$$

5 where $I_{\parallel}$ and $I_{\perp}$ are the intensities of fluorescence emission with their electric vectors respectively parallel and perpendicular to that of the linearly polarized exciting radiation, and G is an empirical correction factor used to correct for the dichroic mirror's transmission efficiency in the parallel and perpendicular planes, and where additional symbols used in the formal definition of anisotropy are as described above.

10 Fig. 2 is a schematic of an apparatus 10 designed for the topographic mapping of tissue and blood PO_2 in an imaged tissue. An O_2 -quenchable, biocompatible probe substance, such as sodium pyrenebutyrate in a suitable solvent or incorporated in liposomes, is injected intravenously into the host organism. The tissue 12 is illuminated with nonpolarized visible light by a fiberoptic illuminator 14 utilizing a tungsten source
15 16. The radiant energy of a high pressure mercury bulb 18 is gathered by a collector lens 20, is shuttered by a shutter 22, and passes through a Glan-Thompson polarizer 24 (Ealing, Inc.). The UV light is spectrally shaped by an excitation filter 26 (340 nm peak, 25 nm half bandpass) and is then reflected by a dichroic mirror 28 (Omega Optical) which reflects wavelengths <400 nm through an objective lens 30 to the imaged tissue.
20 Emitted fluorescence (wavelengths >400 nm) from the excited tissue 12 is gathered by the objective lens 30 and passes through the dichroic mirror 28, and through an emission filter 32, which passes wavelengths from 400-420 nm to a Wollaston prism polarizer 34 which resolves the emitted fluorescence into its linearly polarized components parallel 36 and perpendicular 38 to the plane of excitation polarization. The vector components
25 ($I_{\parallel}$ and $I_{\perp}$) are respectively and simultaneously detected by the CCD (charge coupled devices) chips of two video cameras 40A and 40B (e.g. Xybion model 250). It is obvious to one skilled in the art that alternative optical detectors with sufficient spatial resolution, such as slow scan chilled CCD cameras, SIT or ISIT tube cameras, or photodiode arrays (not shown) would also be suitable for the detection of the two-
30 dimensional distributions of the parallel and perpendicular components of the emitted fluorescence. The outputs of the two video cameras are digitized by two digitizing boards 42A and 42B (such as sold by Imaging Technologies or under the designation

model DT3851 by Data Translation) within a microcomputer 44, (e.g., an IBM® or clone computer, preferably having a processing chip operating at 33 or 66 MHz). Such a device is sufficient for the task and for subsequent image processing prior to display on the monitor 46.

5 The computer utilizes the software which is produced and copyrighted by Biometric Imaging Inc. The source code for that software is attached hereto and allows the variation of the slopes and offsets at each pixel location within the CCD array to be corrected by a correction file, thereby ensuring uniform responsivity across the detector array. Similarly, the software package allows pixel intensities within the array to be
10 quantified, filtered, averaged, and anisotropy to be calculated at a plurality of locations by means of equation 9 and converted to a two dimensional representation of tissue PO₂ by the application of equation 6 at each pixel locus. The present prototype apparatus allows the gathering of more than 300,000 values of tissue PO₂ in space within approximately 33 milliseconds. The values for the constants in equation 6 are
15 determined during calibration experiments which involve the simultaneous measurement of tissue PO₂ by an oxygen microelectrode and by the indirect lifetime (anisotropy) system.

Should the autofluorescence of the tissue at the emission wavelength show some degree of anisotropy, due to the presence of endogenous fluorophores with short
20 fluorescence lifetimes (< 20 nanoseconds), and which are therefore not quenched by physiologic levels of O₂, then such "auto-anisotropy" can be imaged and saved to disk prior to the injection of the biocompatible, O₂-quenchable fluorescent probe substance. In this case, subsequent to the injection of the O₂-quenchable probe substance, the total anisotropy (A_{total}) can be related to its component anisotropies ("auto-anisotropy" or
25 A_{tissue}, and the O₂-dependent anisotropy, A_{probe}) by the following equation:

$$A_{total} = \frac{a(A_{tissue}) + b(A_{probe})}{a + b} \quad (10)$$

where a+b=1. The O₂-dependent anisotropy can then be separated from the total
30 anisotropy by equation 11, where A(PO₂) is the PO₂-dependent anisotropy, and a is a constant determined empirically in experiments in which tissue PO₂ is measured

simultaneously by an O₂ microelectrode and by optical measurements of fluorescence anisotropy.

$$A_{\text{probe}} \text{ or } A(\text{PO}_2) = \frac{A_{\text{total}} - a(A_{\text{tissue}})}{1 - a} \quad (11)$$

This correction can be rapidly and conveniently implemented on a pixel-by-pixel basis by programs within the software package.

An immediate advantage of the present procedure over the invention of the Zuckerman '173 patent, which depends upon the measurement of fluorescence intensities rather than anisotropy, should be evident to those skilled in the art. That is, in that patent, tissue PO₂ measurements can only commence after the O₂-quenchable fluorescent probe substance has reached equilibrium (stable) concentrations within the imaged tissue since fluorescence intensity in space and time is related to the concentration of probe in the tissue as well as to tissue PO₂. In the indirect lifetime system herein depicted, fluorescence anisotropy, which is insensitive to probe concentration, rather than intensity is the determinant of tissue PO₂, thereby allowing PO₂ measurements to commence as soon as sufficient probe substance has accumulated to provide suitable signal-to-noise levels. Similarly, the anisotropy method is insensitive to the time-dependent degradation of the probe substance by metabolism, thereby permitting tissue PO₂ to be determined topographically for extended time periods.

It should be pointed out at this juncture that pseudocoloring programs are contained in the software package of Exhibit A to allow optical maps of tissue PO₂ to be depicted in two dimensional space by assigning different hues to different values of tissue PO₂, with an accompanying scale which would allow rapid interpretation of the distribution of tissue PO₂ by the clinician. The programs written to implement the indirect lifetime system can proceed essentially in "real time", thereby allowing the clinician to view immediately the effects of treatment modalities designed to increase tissue oxygenation (e.g. laser treatment employed to increase retinal tissue PO₂ used frequently in the treatment of diabetic retinopathy).

The power of the indirect lifetime system in imaging tissue PO₂ can be extended from two to three dimensions by the implementation of optical serial sectioning either by confocal techniques or by the application of digital deconvolution programs, such as those contained in Exhibit A. In this manner a topographic view of tissue PO₂ can be

effected, allowing tissue and blood PO_2 to be imaged in three dimensions within volumes of tissue. Such three-dimensional information could be conveniently pseudocolored, and by means of three-dimensional reconstruction algorithms, be depicted on the monitor screen. Optical serial sectioning and 3-D reconstruction would allow the clinician to
5 detect and treat abnormal regions of tissue PO_2 located at various depths within the imaged tissue.

Fig. 3 shows an apparatus 50 constructed in accordance with the present invention to implement the indirect fluorescence lifetime system in a sealed catheter design which does not require calibration prior to its every usage. As in the previous embodiment the
10 radiant energy of a high pressure mercury bulb 52 is gathered by a collector lens 54, is shuttered via shutter 56, and passes through a Glan-Thompson polarizer 58 (such as sold by Ealing, Inc. under the designation of catalog number 34-5223). The UV light is spectrally shaped by an excitation filter 60 (340 nm peak, 25 nm half bandpass) and is then reflected by a dichroic mirror 62 (such as sold by Omega Optical under the
15 designation 400DCLP) which reflects wavelengths <400 nm through an objective lens 64. A single mode polarization-preserving glass fiber 66 (such as sold by Ealing, Inc., under the designation model HB450) is fixed at the focal point of the objective lens. The core diameter of such fibers is typically <10 microns, thereby permitting an exceedingly narrow catheter to be designed. In the case of a narrow submillimeter
20 catheter, the linearly polarized light passes directly to the O_2 -sensitive tip 68, whereas for larger diameter catheters the glass fiber 66 is mounted at the focal point of a collimating lens 70 which is used to provide uniform illumination of the catheter tip 68. The sealed O_2 -sensitive tip 68 contains a quartz plate 72 which separates the optical components from the nonpolar viscous medium 74 in the tip 68. The catheter tip 68
25 contains a nonpolar viscous medium 74 such as paraffin or mineral oils in which pyrenebutyric acid or pyrene 76 is dissolved. Pyrene has a fluorescence lifetime approximately ten times that of pyrenebutyric acid or its salt form, thereby increasing the precision of the measurement, with an accuracy of PO_2 measurement of <1 mm Hg possible. The O_2 -quenchable probe substance dissolved in a nonpolar viscous solvent is
30 sealed within the catheter by an O_2 permeable membrane 78 such as polyethylene. A micro-thermistor 80 is also incorporated into the tip 68 of the catheter to allow simultaneous measurement of temperature, thus permitting temperature corrections to

be performed according to the relation described in equation 4. Since optics are reversible, the linearly polarized fluorescence emission from the probe tip 68 returns along the single mode polarization-preserving glass fiber and is collected by the objective lens 64. Wavelengths >400 nm pass through the dichroic mirror 62 and wavelengths from 400-420 nm pass through an emission filter 82 to a Wollaston prism polarizer 84, which resolves the emitted fluorescence into its linearly polarized components parallel 86A and perpendicular 86B to the plane of excitation polarization. The vector components ($I_{\parallel}$ and $I_{\perp}$) are respectively and simultaneously detected by two optical detectors 88A and 88B, which can be photodiodes or photomultiplier tubes.

Since the catheter design provides PO_2 measurement at a single locus, viz. at the catheter tip, anisotropy can be calculated from equation 9 and related to PO_2 according to equation 6, by means of a simple microprocessor design 90. Such a generic microprocessor design is illustrated in Fig. 4. Moreover, once the relations and constants employed in these equations are determined, they may be fixed and employed in every catheter manufactured, and each catheter need not be individually calibrated at the factory, or calibrated prior to its use in a clinical situation. This represents the considerable advantage of the present catheter design over previous designs which require constant recalibration in the factory as well as in the field.

EXAMPLE

To measure the topographic distribution of tissue PO_2 in retinal tissue, an imaging apparatus was set up as follows and as schematically illustrated in Fig. 2. A bovine eye was obtained from a local slaughterhouse and transported on ice to the laboratory. The isolated bovine eye was set up for normothermic arterial perfusion according to the procedures of de Coe, Zonnenberg and Trap (*Current Eye Research*, 12(4), 293-301, 1993), using oxygenated serum-free MEM (minimal essential medium) which was initially supplemented with $108 \mu\text{M}$ sodium pyrenebutyrate. After one hour of perfusion with MEM containing sodium pyrenebutyrate, the biocompatible, fluorescent probe substance reached equilibrium concentration in the retinal tissue, and the eye was subsequently perfused with oxygenated MEM alone during the period of time in which measurements of tissue PO_2 were performed. Corneal refraction was negated by a custom fundus lens made of UV transmitting glass (Optical Industries), and the retina was imaged first in visible light and a region of the retina selected which contained a retinal arteriole. Once

proper orientation was established visible light was discontinued, and the retinal tissue was illuminated with linearly polarized ultraviolet light (UV). The radiant energy of a high pressure mercury bulb was gathered by a collector lens, was shuttered, and passed through a Glan-Thompson polarizer (Ealing, Inc.). The UV light was spectrally shaped by an excitation filter (340 nm peak, 25 nm half bandpass) and was then reflected by a dichroic mirror (Omega Optical) which reflects wavelengths <400 nm through an objective lens to the imaged retinal tissue. Emitted fluorescence (wavelengths >400 nm) from the excited tissue was gathered by the objective lens and passed through the dichroic mirror, and through an emission filter, which passed wavelengths from 400-420 nm to a Wollaston prism polarizer which resolved the emitted fluorescence into its linearly polarized components parallel and perpendicular to the plane of excitation polarization. The vector components ($I_{\parallel}$ and $I_{\perp}$) were respectively and simultaneously detected by the CCD (charge coupled devices) chips of two video cameras (Xybion model 250).

The vector components of fluorescence emission parallel and perpendicular to the plane of excitation polarization were digitized by the previously identified two digitizing boards within the previously identified computer operating at 66 MHz, and stored to hard disk (not shown). The computer software of Exhibit A was run on the computer to allow the variation of the slopes and offsets at each pixel location within the CCD arrays to be corrected by a correction file in the software, thereby ensuring uniform responsivity across the detector array. Similarly, the software allowed anisotropy to be calculated at a plurality of locations by means of equation 9 and converted to a two dimensional representation of tissue PO_2 by the application of equation 6 at each pixel locus. This procedure allowed the gathering of more than 300,000 values of retinal tissue PO_2 in space within approximately 33 milliseconds. The values for the constants used in equation 6 were previously determined during calibration experiments which involved the simultaneous measurement of tissue PO_2 by an oxygen microelectrode and by the indirect lifetime (anisotropy) system. To test the validity and precision of the system and the resultant optical map of retinal tissue PO_2 , PO_2 gradients were measured both parallel and perpendicular to the retinal arteriole, as the mathematical functions describing these relationships can be predicted from oxygen diffusion mathematics, and are well established in the literature. The results of these measurements are shown in

Fig. 5. Oxygen diffusion mathematics predicts that the longitudinal PO_2 drop in the tissue parallel to the arteriole should follow a linear relationship, whose slope is directly proportional to the oxygen consumption rate of the tissue (M) and inversely proportional to the velocity (V) of oxygen-containing solution within the arteriole times the square of the arteriolar diameter (D). As shown in the upper part of Fig. 5, the tissue PO_2 measurements along the length of the arteriole conform excellently to this well-established linear relationship, with a coefficient of determination (R^2) for a linear fit of 0.96 obtained; *i.e.*, 96% of the variance in the data is accounted for by the expected linear fit. Similarly, the PO_2 gradient perpendicular to an arteriole should conform to a monotonically decreasing function which is described mathematically by the Krogh model. In Fig. 5 (lower) a Krogh model fit has been applied to the data with a goodness of fit (R^2) of 0.99; *i.e.*, 99% of the variance in the data is accounted for by the expected mathematical relation. The data of Fig. 5, then, demonstrate that the indirect fluorescence lifetime system provides a noninvasive and valid method for evaluating tissue PO_2 in space and time.

While the present invention has been described in conjunction with preferred embodiments and illustrative examples, one skilled in the art after reading the foregoing specifications will be able to effect various changes, substitutions, and other alterations to the methods set forth herein. It is therefore intended that the protection granted by Letters Patent hereon be limited only by the definition contained in the appended claims and equivalents thereof.

Source code excerpted from "Software for the Acquisition, Manipulation
and Analysis of Video Images", © 1993, Biometric Imaging, Inc.

Please note that page numbers in this excerpt do not correspond to those in full document.

```
/* initial grabber */

#include <stdio.h>
#include <dos.h>

#define BASEMEM 0xD000 /* memory base address */
#define RB 0x300      /* pcplus register base address */

void ini_lut(void);
void ini_reg(void);

main()
{
int i;

    ini_reg();
    ini_lut();

    ini_reg();
    ini_lut();      /* why does it need twice ? */
}

void ini_lut(void)
{
int j;
unsigned char i;

    outp(RB+1,0x00);
    for(j=0;j<=255;j++)
    {
        i=j;
        outp(RB+2,i); outp(RB+3,i); }

    outp(RB+1,0x01);
    for(j=0;j<=255;j++)
    {
        i=j;
        outp(RB+2,i); outp(RB+3,i); }

    outp(RB+1,0x02);
    for(j=0;j<=255;j++)
    {
        i=j;
        outp(RB+2,i); outp(RB+3,i); }

    outp(RB+1,0x03);
    for(j=0;j<=255;j++)
    {
        i=j;
        outp(RB+2,i); outp(RB+3,i); }
}

void ini_reg(void)
{
```

```

outp(RB,0x16); outp(RB+1,3); outp(RB+2,0);
outp(RB+3,0); outp(RB+4,0x36);outp(RB+5,0x02);
outp(RB+6,0); outp(RB+7,0); outp(RB+8,0x40);
outp(RB+9,0); outp(RB+10,0);      outp(RB+11,0);

```

```

}

```

```

/* clean the screen */

```

```

#include <stdio.h>

```

```

#include <dos.h>

```

```

#define BP 0x3F8      /* for com1. baseport register address*/

```

```

#define CONREG 0x300  /* pcplus register base address */

```

```

main()

```

```

{

```

```

int i,j,k,line;

```

```

unsigned int stat,ini_stat;

```

```

unsigned char huge *mem;

```

```

unsigned char huge *mem_keep;

```

```

    mem=mem_keep=(unsigned char huge *)MK_FP(0xD000,0);

```

```

    ini_stat=0x16;

```

```

    for(i=0;i<8;i++)

```

```

    {

```

```

        stat=(i<<5)&0xe0;

```

```

        stat |=ini_stat;

```

```

        outp(CONREG,stat);

```

```

        line=64;

```

```

        if(i==7) line=32;

```

```

        for(j=0;j<line;j++)

```

```

        {

```

```

            for(k=0;k<640;k++)

```

```

            {

```

```

                *mem=0;

```

```

                mem++;

```

```

            }

```

```

            mem+=(1024-640);

```

```

        }

```

```

        mem=mem_keep;

```

```

    }

```

```

}

```

```

/* This program is used to get integrating image from CCD camera.

```

```

Integrate signal is a positive impulse sent from port 0x3FC (modem
control register, DTR&RTS ON ),the snap signal from camera is sent
to port 0x3FD (modem status register )
*/

```

```

#include <stdio.h>

```

```

#include <dos.h>

```

```

#include <time.h>

```

```

#include <string.h>

```

```

#include <alloc.h>
#include <dir.h>

#define BASEMEM 0xD000 /* memory base address */
#define BP 0x3F8 /* for com1. baseport register address*/
#define RB 0x300 /* pcplus register base address */

main()
{
int m,j,test_times,x1,y1,x2,y2,th,tm,ts,av1,av2,img_num,re_openfile,time_delay;
float i,ii,sample_value,test_value,av,zeit_interval,zeit;
char imname[30],datname[30],s1[30],s2[30],s3[30],c,overwr;
FILE *fp,*datfile;
unsigned char *pixel,n,value;
unsigned char huge *mem;
long zz,zzz;
time_t anf_zeit,jetzt_zeit;
int odd,even,k;
struct fblk fblk;

printf("The datafile saves the information about your images,");
do{
overwr='y';
printf("\ninput the datafile name--> ");
scanf("%s",datname); printf("\n");
if(findfirst(datname,&fblk,0)==0)
{
printf("--- Warning: the file already exists, overwrite it?(y) ");
overwr=getche();
}
}while(overwr!='y');

printf("\nThe name of your image has two attributes,\n");
printf("the first is the NAME (e.g. 'myimage') given by user; \n");
printf("the second is the IMAGE ORDER,(1,2,3,...) determined by the program.\n");
printf("The image sequence will be myimage1,myimage2,.....\n");
printf("input image name--> ");
scanf("%s",s1); printf("\n");

datfile=fopen(datname,"wt");
/* set output signal low */

mem=(unsigned char huge *)MK_FP(BASEMEM,0);

outp(BP+4,0);

/* initiate grabber's registers */

/*
inigrab();
*/

c="";

do{
if(c==3) exit();
printf("\n-----\n");

```

```

/* image grab */

value=inp(RB+5);
outp(RB+5,value|0xc0); /* for(j=0;j<10000;j++){}; */

/* get area of interested */

printf("if ready, type 'f' to freeze image and get AOI:");
while((c=getche())!='f') { if(c==3) exit(); };
printf("\n");
outp(RB+5,0x82); /* do snap */

getaoi(&x1,&y1,&x2,&y2);
x1++; y1++; x2--; y2--;
area_ave(x1,y1,x2,y2,&av);
printf("the area size is %d X %d. average value is %f.\n",
      x2-x1+1,y2-y1+1,av);
printf("AIO is OK?(y)");

}while((c=getche())!='y');

fprintf(datfile,"AOI size:(%d x %d); rectangular position: [(%d,%d),(%d,%d)].\n",
x2-x1+1,y2-y1+1,x1-1,y1-1,x2+1,y2+1);
printf("\n");

img_num=0;
time_delay=0;

do{

/* inigrab();
*/
outp(BP+4,0);
value=inp(RB+5);
outp(RB+5,value|0xc0); /* do grab again */

/* input integrateduration */

printf("-----\n");
if(time_delay==0)
{
printf("Delay time(sec.): ");
scanf("%f",&zeit); printf("\n");
time_delay=1;
}
else
{
printf("delay time: %f sec., change it?(y): ",zeit);
if(getche()=='y')
{
printf("\ndelay time(sec.): ");
scanf("%f",&zeit);
}
printf("\n");
}

i=zeit;
ii=i;
test_times=0;

```

```

do{

i=ii;
outp(RB,0x16); value=inp(RB+5); outp(RB+5,value|0xc0);

/* output integrating impulse */

/* printf("integrating.....");
i*=25000;
*/
i*=100000;
while(i>0)
{
    outp(BP+4,3);          /* set output signal high */
    i--;
}
outp(RB+5,0x82);
for(i=0;i<10000;i++){          /* for 386 i<1000 */

outp(BP+4,0);          /* set output signal low */

/* wait for grab signal */

while(inp(BP+6)==1) {      };

/* get test value, if integrate not successful, do again */

outp(RB,((5<<5)&0xe0) | 0x16);
test_value=100;
test_times++;
}while(test_value<10 && test_times<=2.);

/* store image into disk*/
show_time(&th,&tm,&ts,&jetzt_zeit);
area_ave(x1,y1,x2,y2,&av);
rectangular_show(x1-1,y1-1,x2+1,y2+1,255);
av=(2*av<255)? 2*av:255;
av1=av;          av*=100;          av2=av;          av1*=100;
av1=av2-av1;  av2/=100;          av/=100;
outp(RB,(((1<<5)&0xe0)|0x16));

/*      for(k=0;k<40;k++)
{
    zz=1024;          zz*=k;          zz+=10;
    printf("%d ",*(mem+zz+10));
}
printf("\n");
*/
odd=even=0;
for(k=0;k<50;k++)
{
    odd+=*(mem+k);          even+=*(mem+k+1024);
}

if(odd>even)  printf("-----this is the ODD field image.\n",odd,even);
else          printf("-----this is the EVEN field image.\n",odd,even);
printf("-----average value is %d.%d, store image?(y) ",av2,av1);

```

```

    if(getche()=='y')
    {
        img_num++;
        printf("\n");
        if(img_num==1)      anf_zeit=jetzt_zeit;
        zeit_interval=jetzt_zeit-anf_zeit;
        zeit_interval/=60;
        do{
            overwr='y';
            itoa(img_num,s2,10);
            strcpy(imname,s1);
            strcat(imname,s2);
            re_openfile=0;
            printf("the name of image %d is < %s >, change it(y): ",
                img_num,imname);
            if(getche()=='y')
            {
                printf("\nnew name: ");
                scanf("%s",imname);
            }
            printf("\n");
            if(findfirst(imname,&ffblk,0)==0)
            {
                printf("--- Warning: the file already exists, overwrite it?(y) ");
                overwr=getche();
            }
        }while(overwr!='y');

        fp=fopen(imname,"wb");
        im_store(fp,x1,y1,x2,y2);
        fclose(fp);
        if(odd>even)
            fprintf(datfile,"-- %s: mean_value(%f), time(%f min.), odd field.\n",
                imname,av,zeit_interval);
        else
            fprintf(datfile,"-- %s: mean_value(%f), time(%f min.), even field.\n",
                imname,av,zeit_interval);
    }

    printf("\n");
    printf("Press q to quit, any other key to continue.\n");

}while(getche()!='q');

fclose(datfile);
printf("\n");

}

inigrab()
{
    outp(RB,0x16); outp(RB+1,3); outp(RB+2,0);
    outp(RB+3,0); outp(RB+4,36);      outp(RB+5,0x02);
    outp(RB+6,0); outp(RB+7,0); outp(RB+8,0x40);
    outp(RB+9,0); outp(RB+10,0);      outp(RB+11,0);
}

```

```
#define CONREG 0x300
```

```
getaoi(x1,y1,x2,y2)
int *x1,*y1,*x2,*y2;
{
float xx1,yy1,xx2,yy2;
```

```
start_end_point(&xx1,&yy1,&xx2,&yy2);
```

```
*x1=xx1;      *y1=yy1;      *x2=xx2;      *y2=yy2;
rectangular_show(*x1,*y1,*x2,*y2,255);
```

```
}
```

```
rectangular_show(x1,y1,x2,y2,br)
int x1,y1,x2,y2,br;
{
line_show(x1,y1,x2,y1,br);
line_show(x2,y1,x2,y2,br);
line_show(x2,y2,x1,y2,br);
line_show(x1,y1,x1,y2,br);
}
```

```
area_ave(x1,y1,x2,y2,av)
int x1,y1,x2,y2;
float *av;
{
int i,j,x,y,k,start_line,end_line,start_blc,end_blc,
start_y_offset,end_y_offset,line_offset;
unsigned char huge *mem;
unsigned char huge *mem_keep;
unsigned char ini_stat,stat;
unsigned char p;
long zz,zzz,testv1,testv2;
```

```
mem=(unsigned char huge *)MK_FP(BASEMEM,0);
x=x2-x1+1;
start_blc=y1/64;          end_blc=(y2+1)/64;
start_y_offset=y1%64;     end_y_offset=(y2+1)%64;
mem_keep=mem+x1;
zz=start_y_offset;      zz*=1024;      zz+=x1;
mem +=zz;
```

```
/* fine the data line */
testv1=testv2=0;
for(i=0;i<x;i++)
{
testv1+=*(mem+i);    testv2+=*(mem+1024+i);
}
line_offset=(testv1>testv2)? 0:1;
```

```
ini_stat=0x16;      zzz=zz=0;
for(i=start_blc;i<=end_blc;i++)
{
stat=(i<5)&0xe0;
stat |=ini_stat;
```

```

    outp(CONREG,stat);

    if(i==start_blc)
        start_line=start_y_offset+line_offset;
    else start_line=0+line_offset;
    if(i==end_blc) end_line=end_y_offset;
    else end_line=64;

    for(j=start_line;j<end_line;j+=2)
    {
        for(k=0;k<x;k++)
        {
            zzz+=*mem;  zz++;
            mem++;
        }
        mem+=(1024-x);
    }
    mem=mem_keep;
}

*av=zzz;  *av/=zz;
}

point_show(xp,yp,br)
int xp,yp,br;
{
    unsigned char huge *mem;
    int block,y_offset;
    unsigned char status;
    long zz;

    mem=(unsigned char huge *)MK_FP(BASEMEM,0);
    block=yp/64;  y_offset=yp%64;
    zz=1024;  zz*=y_offset;  zz+=xp;  mem+=zz;
    status=((block<5)&0xe0)|0x16;
    outp(CONREG,status);  *mem=br;
}

line_show(x1,y1,x2,y2,br)
int x1,y1,x2,y2,br;
{
    /* start(small address)----->end(big address) */
    int p,xx,yy,q,aa,delta_x,delta_y;
    float qq,dx,dy;
    long zzz,zz;

    zz=y1;  zz*=1024;  zz+=x1;
    zzz=y2;  zzz*=1024;  zzz+=x2;
    if(zz>zzz)
    {
        xx=x1;  x1=x2;  x2=xx;
        yy=y1;  y1=y2;  y2=yy;
    }
    delta_x=x2-x1;  delta_y=y2-y1;

```

```
/*printf("delta_x: %f %d %d ,delta_y: %f %d %d.\n",delta_x,x2,x1,delta_y,y2,y1);
*/
```

```
if(delta_x==0)
    for(p=0;p<=delta_y;p++)        point_show(x1,y1+p,br);
if(delta_y==0)
    for(p=0;p<=delta_x;p++)        point_show(x1+p,y1,br);
if(delta_x!=0 && delta_y!=0)
{
    aa=(delta_x>0)? 1:-1;  dx=delta_x;  dy=delta_y;
    if(aa*delta_x >= delta_y)
        for(p=0;p<=aa*delta_x;p++)
        {
            qq=(dy*p)/(dx*aa)+0.5;        q=qq;
            point_show(x1+aa*p,y1+q,br);
        }
    else
        for(p=0;p<delta_y;p++)
        {
            qq=(dx*p)/(aa*dy)+aa*0.5;        q=qq;
            point_show(x1+q,y1+p,br);
        }
}
```

```
/*    printf("--draw show ok.\n");
*/
}
}
```

```
#define CONREG 0x300
```

```
start_end_point(x1,y1,x2,y2)
float *x1,*y1,*x2,*y2;
{
    unsigned char huge *cross_buf;
    char b[4];
    int a,i,j,x,y,xx,yy,kx,ky,sx,sy,ex,ey;
```

```
x=320; y=240;
```

```
cross_buf=(unsigned char huge *)calloc(41,1);
```

```
for(i=0;i<2;i++)
{
    xx=x;  yy=y;
    do{
        if(i==0)        printf("current position---(%1d,%1d).\r",xx,yy);
        if(i==1)
        {
            printf("current position---(%1d,%1d), AOI size (%1d,%1d).\r",
                xx,yy,xx-sx-1,yy-sy-1);
            if(xx-sx-1<0 || yy-sy-1<0)
            {
                sound(600);    delay(100);    nosound();
            }
        }
    }
}
```

```

    kx=xx; ky=yy;
    move_cross(kx,ky,cross_buf);
    a=getch();
    if(a==56)    yy--;
    if(a==57)    yy-=10;

    if(a==50)    yy++;
    if(a==49)    yy+=10;

    if(a==52)    xx--;
    if(a==55)    xx-=10;

    if(a==54)    xx++;
    if(a==51)    xx+=10;

    if(a==48)    {    xx=320;    yy=240;    }

    if(a>=48 && a<=57) recover_cross(kx,ky,cross_buf);
}while(a!=43);
if(i==0)
{
    sx=xx; sy=yy; point_show(sx,sy,255);
    *x1=xx;    *y1=yy;
    printf("\nup-left point(%d,%d).\n",sx,sy);
}
if(i==1)
{
    ex=xx; ey=yy; point_show(ex,ey,255);
    *x2=xx,    *y2=yy;
    printf("\nlow-right point(%d,%d).\n",ex,ey);
}
}

move_cross(xp,yp,cross_buf)
int xp,yp;
unsigned char huge *cross_buf;
{
    int i,xx,yy;
    unsigned char huge *mem;
    unsigned char huge *mem_keep;
    int block,y_offset;
    unsigned char status;
    long zz;

    mem=mem_keep=(unsigned char huge *)MK_FP(BASEMEM,0);
    for(i=-10;i<=10;i++)
    {
        yy=yp+i;    xx=xp;
        block=yy/64;    y_offset=yy%64;
        zz=1024;    zz*=y_offset;    zz+=xx;    mem+=zz;
        status=((block<<5)&0xe0)|0x16;
        outp(CONREG,status);
        *cross_buf=*mem;    *mem=255;    cross_buf++;
        mem=mem_keep;
    }
}

```

```

mem=mem_keep;
block=yp/64;  y_offset=yp%64;
status=((block<<5)&0xe0)|0x16;
outp(CONREG,status);
for(i=-10;i<=10;i++)
{
    mem=mem_keep;    xx=xp+i;
    zz=1024;    zz*=y_offset;  zz+=xx;    mem+=zz;
    if(i!=0)
    {
        *cross_buf=*mem;    *mem=255;    cross_buf++;
    }
}

```

```

recover_cross(xp,yp,cross_buf)
int xp,yp;
unsigned char huge *cross_buf;
{
    int i,xx,yy;
    unsigned char huge *mem;
    unsigned char huge *mem_keep;
    int block,y_offset;
    unsigned char status;
    long zz;

```

```

mem=mem_keep=(unsigned char huge *)MK_FP(BASEMEM,0);
for(i=-10;i<=10;i++)
{
    mem=mem_keep;
    yy=yp+i;    xx=xp;
    block=yy/64;  y_offset=yy%64;
    zz=1024;    zz*=y_offset;  zz+=xx;    mem+=zz;
    status=((block<<5)&0xe0)|0x16;
    outp(CONREG,status);
    *mem=*cross_buf;    cross_buf++;
}

```

```

mem=mem_keep;
block=yp/64;  y_offset=yp%64;
status=((block<<5)&0xe0)|0x16;
outp(CONREG,status);
for(i=-10;i<=10;i++)
{
    mem=mem_keep;
    xx=xp+i;
    zz=1024;    zz*=y_offset;  zz+=xx;    mem+=zz;
    if(i!=0)
    {
        *mem=*cross_buf;    cross_buf++;
    }
}

```

```

im_store(fp,x1,y1,x2,y2)
FILE *fp;
int x1,y1,x2,y2;

```

```

{
int i,j,x,y,k,start_line,end_line,start_blc,end_blc,
    start_y_offset,end_y_offset;
unsigned char huge *mem;
unsigned char huge *mem_keep;
unsigned char ini_stat,stat;
unsigned char p;
long zz;
char ix[8],iy[8];

    mem=(unsigned char huge *)MK_FP(BASEMEM,0);
    x=x2-x1+1;    y=y2-y1+1;
    itoa(x,ix,10);    itoa(y,iy,10);
    for(i=0;i<8;i++)    fputc(ix[i],fp);
    for(i=0;i<8;i++)    fputc(iy[i],fp);
    start_blc=y1/64;    end_blc=(y2+1)/64;
    start_y_offset=y1%64;    end_y_offset=(y2+1)%64;
    mem_keep=mem+x1;
    zz=start_y_offset;    zz*=1024;    zz+=x1;
    mem +=zz;
    ini_stat=0x16;
    for(i=start_blc;i<=end_blc;i++)
    {
        stat=(i<<5)&0xe0;
        stat |=ini_stat;
        outp(CONREG,stat);

        if(i==start_blc)    start_line=start_y_offset;
        else    start_line=0;
        if(i==end_blc)    end_line=end_y_offset;
        else    end_line=64;
        for(j=start_line;j<end_line;j++)
        {
            for(k=0;k<x;k++)
            {
                p=*mem;    fputc(p,fp);
                mem++;
            }
            mem+=(1024-x);
        }
        mem=mem_keep;
    }
printf("----- a %d X %d image is saved.\n",x,y);
}

#include <time.h>

show_time(th,tm,ts,jetzt_zeit)
int *th,*tm,*ts;
time_t *jetzt_zeit;
{
    struct tm *timeptr;
    time_t secsnow;

    time(&secsnow);    *jetzt_zeit=secsnow;

```

```

timeptr=localtime(&secsnow);
printf("-----current time is %02d:%02d:%02d\n",
       timeptr->tm_hour,timeptr->tm_min,timeptr->tm_sec);
*th=timeptr->tm_hour;
*tm=timeptr->tm_min;
*ts=timeptr->tm_sec;
}

#include "ypdef.h"
#include <stdio.h>
#include <string.h>
#include <dos.h>
#include <alloc.h>

#define BASEMEM 0xD000
#define CONREG 0x300

main()
{
char imname[30],outname[30],calname1[30],calname2[30];
unsigned char huge *image;
unsigned char huge *stdimage1;
unsigned char huge *stdimage2;
long zz;
int x,y,i,j,pp,field;
float ave0,ave1,ave2,ave3,p0,p1,p2,k,b;

do{
printf("1. image with one field.\n");
printf("2. image with both fields.\n");
scanf("%d",&field);    printf("\n");
}while(field!=1 && field!=2);
printf("image name: ");
scanf("%s",imname);    printf("\n");
printf("calibration image 1: ");
scanf("%s",calname1);  printf("\n");
printf("calibration image 2: ");
scanf("%s",calname2);  printf("\n");

/*
zz=y;  zz*=x;
image=(unsigned char huge *)farcalloc(zz,1);
stdimage1=(unsigned char huge *)farcalloc(zz,1);
stdimage2=(unsigned char huge *)farcalloc(zz,1);
if(image==NULL || stdimage1==NULL || stdimage2==NULL )
{    printf("fail to allocate.\n");    exit(1); }

*/
stdimage1=(unsigned char huge *)read_image(calname1,&x,&y);
image_show(stdimage1,0,0,x,y);
stdimage2=(unsigned char huge *)read_image(calname2,&x,&y);
image_show(stdimage2,50,50,x,y);
image=(unsigned char huge *)read_image(imname,&x,&y);
image_show(image,100,100,x,y);

ave1=ave2=0;

```

```

for(j=0;j<y;j++)
    for(i=0;i<x;i++)
        {
            zz=j;  zz*=x; zz+=i;
            ave1+=*(stdimage1+zz);
            ave2+=*(stdimage2+zz);
        }
ave1/=x;    ave1/=y;
ave2/=x;    ave2/=y;
if(field==1)
    {    ave1*=2;    ave2*=2;    }
printf("the average value of cal-image 1: %f.\n",ave1);
printf("the average value of cal-image 2: %f.\n",ave2);
if(ave1==ave2)
    {
        printf("same average value.\n");
        exit();
    }
ave0=ave3=0;
for(j=0;j<y;j++)
    for(i=0;i<x;i++)
        {
            zz=j;  zz*=x; zz+=i;
            p0=*(image+zz);    ave3+=p0;
            p1=*(stdimage1+zz);
            p2=*(stdimage2+zz);
            k=(p1-p2)/(ave1-ave2);    b=p2-k*ave2;
            if(k!=0)    p0=(p0-b)/k+0.5;
            ave0+=p0;
            pp=p0;
            if(pp>=255)    pp=255;
            if(pp<=0)    pp=0;
            point_show(i+150,j+150,pp);
        }
ave0/=x;    ave0/=y;
ave3/=x;    ave3/=y;
printf("the average value of the image to be calibrated: %f.\n",ave3);
printf("the average value of the calibrated image: %f.\n",ave0);
printf("\n");

printf("save image?(y): ");
if(getche()=='y')
    {
        printf("\noutput image name: ");
        scanf("%s",outname);    printf("\n");
        save_image(outname,150,150,150+x-1,150+y-1);
    }

printf("\n=====END\n");

}

/*.....subtract image2 from image1 .....*/

#include "ypdef.h"
#include <stdio.h>

```

```

#include <dos.h>
#include <alloc.h>
#include <dir.h>

#define BASEMEM 0xD000
#define CONREG 0x300

main ()
{
char imgname1[50],imgname2[50],outname[50],overwr;
unsigned char huge *image;
unsigned char huge *board;
unsigned char huge *p;
int choice,c,i,j,x,y,ws,wrongname,pp,ppp;
long size,zz;
int n;
float s,s1,s2,s3;
struct fblk fblk;

/* read an image from disk into memory */

printf("%lu bytes free in ram.\n",farcoreleft());
printf("      Image1 - Image2      \n");
printf(" Polarization= ----- X Scale_factor  \n");
printf("      Image1 + Image2      \n");
printf("\n");
printf("the names of image1, image2, and scale_factor: ");
scanf("%s %s %f",imgname1,imgname2,&s);
printf("\n");

image=(unsigned char huge *)read_image(imgname1,&x,&y);
image_show(image,0,0,x,y);
farfree((unsigned char far *)image);

p=(unsigned char huge *)farcalloc(1,1);
image=(unsigned char huge *)read_image(imgname2,&x,&y);
for(j=0;j<y;j++)
    for(i=0;i<x;i++)
    {
        zz=j;  zz*=x; zz+=i;
        get_point_value(i,j,p);  pp=*p;
        ppp=*(image+zz);
        s1=pp; s2=ppp;          s3=s*(s1-s2)/(s1+s2)+0.5;
        pp=s3;
        if(pp>255)  pp=255;          if(pp<0)  pp=0;
        point_show(i,j,pp);
    }
printf("save image?(y): ");
if(getche()=='y')
{
do{
overwr='y';
printf("\noutput image name: ");
scanf("%s",outname);  printf("\n");
if(findfirst(outname,&fblk,0)==0)
{

```

```

        printf("--- Warning: the file already exists, overwrite it?(y) ");
        overwr=getche();
    }
    }while(overwr!='y');
    printf("\n");
    save_image_screen(outname,0,0,x-1,y-1);
}

save_image_screen(imname,x1,y1,x2,y2)
char imname[50];
int x1,y1,x2,y2;
{
    int wrongname,i,j,x,y,k,start_line,end_line,start_blc,end_blc,
        start_y_offset,end_y_offset;
    unsigned char huge *mem;
    unsigned char huge *mem_keep;
    unsigned char ini_stat,stat;
    unsigned char p;
    long zz;
    FILE *fp;
    char ix[8],iy[8];

    do{
        wrongname=0;
        if((fp=fopen(imname,"wb"))==NULL)
        {
            wrongname=1;
            printf("wrong name! try again: ");
            scanf("%s",imname);    printf("\n");
        }
    }while(wrongname==1);
    mem=(unsigned char huge *)MK_FP(BASEMEM,0);
    x=x2-x1+1;    y=y2-y1+1;
    itoa(x,ix,10);    itoa(y,iy,10);
    for(i=0;i<8;i++)    fputc(ix[i],fp);
    for(i=0;i<8;i++)    fputc(iy[i],fp);

    start_blc=y1/64;    end_blc=(y2+1)/64;
    start_y_offset=y1%64;    end_y_offset=(y2+1)%64;
    mem_keep=mem+x1;
    zz=start_y_offset;    zz*=1024;    zz+=x1;
    mem +=zz;
    ini_stat=0x16;
    for(i=start_blc;i<=end_blc;i++)
    {
        stat=(i<<5)&0xe0;
        stat |=ini_stat;
        outp(CONREG,stat);

        if(i==start_blc)    start_line=start_y_offset;
        else    start_line=0;
        if(i==end_blc)    end_line=end_y_offset;
        else    end_line=64;
        for(j=start_line;j<end_line;j++)
        {

```

```

        for(k=0;k<x;k++)
        {
            p=*mem;    fputc(p,fp);
            mem++;
        }
        mem+=(1024-x);
    }
    mem=mem_keep;

}

printf("----- a %d X %d image %s is saved.\n",x,y,imname);
}

#include "ypdef.h"
#include <stdio.h>
#include <alloc.h>
#include <dos.h>
#include <math.h>

#define BASEMEM 0xD000
#define CONREG 0x300

#define XO 320
#define YO 240

main()
{
    char imname[30],outname[30];
    unsigned char huge *image;
    int i,j,x,y,t1,t2;
    long size,*hist;

    hist=(long *)calloc(256,sizeof(long));
    printf("%lu bytes free.\n",farcoreleft());
    printf("image name: ");
    scanf("%s",imname);    printf("\n");
    ini_reg();
    ini_lut();
    image=(unsigned char huge *)read_image(imname,&x,&y);
    image_show(image,XO-x/2,YO-y/2,x,y);
    histogram_image(image,x,y,hist);
    /*
    for(j=0;j<480;j++)
        for(i=533;i<567;i++) point_show(i,j,0);
    grey_level_show();
    show_hist_vertical(hist);    */
    move_cursor_equalize(image,x,y,&t1,&t2,hist);
    equalize_mem(image,x,y,t1,t2);
    printf("\nsave image?(y): ");
    if(getche()=='y')
    {
        printf("\noutput equalized image name: ");
        scanf("%s",outname);    printf("\n");
        save_image(outname,XO-x/2,YO-y/2,XO-x/2+x-1,YO-y/2+y-1);
    }
}

```

```

printf("\n");
}

#define CONREG 0x300

move_cursor_equalize(image,x,y,t1,t2,hist)
unsigned char huge *image;
int x,y,*t1,*t2;
long *hist;
{
    unsigned char huge *cross_buf;
    char b[4];
    int a,i,j,t,xx,yy,kx,ky,sx,sy,ex,ey,k1,k2;
    float tt;

    cross_buf=(unsigned char huge *)calloc(41,1);
    xx=610;      yy=240;
    printf("move cursor to get threshold.");

    do{
        for(k2=0;k2<480;k2++)
            for(k1=533;k1<639;k1++) point_show(k1,k2,0);
        grey_level_show();
        show_hist_vertical(hist);

        printf("\n");
        for(i=0;i<2;i++)
        {
            if(i==0)      printf("move cursor to LOW cut-off grey level.\n");
            else          printf("move cursor to HIGH cut-off grey level.\n");
            do{
                tt=255; tt*=yy; tt/=480;      t=tt;
                printf("current threshold--- %d .\r",t);
                kx=xx; ky=yy;
                save_move_cross(kx,ky,cross_buf);
                a=getch();
                if(a==56)      yy--;
                if(a==57)      yy-=10;

                if(a==50)      yy++;
                if(a==49)      yy+=10;

                if(a==48)      yy=240;

                recover_cross(kx,ky,cross_buf);
            }while(a!=43);
            if(i==0)      *t1=yy;
            else          *t2=yy;
            printf("\n");
        }
        tt=255;      tt*=*t1;      tt/=480;      *t1=tt;
        tt=255;      tt*=*t2;      tt/=480;      *t2=tt;
        equalize_screen(image,x,y,*t1,*t2);
        printf("\nOK?(y): ");
    }while(getche()!='y');
}

```

}

save_move_cross(xp,yp,cross_buf)

int xp,yp;

unsigned char huge *cross_buf;

{

int i,xx,yy;

unsigned char huge *mem;

unsigned char huge *mem_keep;

int block,y_offset;

unsigned char status;

long zz;

mem=mem_keep=(unsigned char huge *)MK_FP(BASEMEM,0);

for(i=-10;i<=10;i++)

{

yy=yp+i; xx=xp;

block=yy/64; y_offset=yy%64;

zz=1024; zz*=y_offset; zz+=xx;

mem+=zz;

status=((block<<5)&0xe0)|0x16;

outp(CONREG,status);

*cross_buf=*mem; *mem=255; cross_buf++;

mem=mem_keep;

}

mem=mem_keep;

block=yp/64; y_offset=yp%64;

status=((block<<5)&0xe0)|0x16;

outp(CONREG,status);

for(i=-10;i<=10;i++)

{

mem=mem_keep; xx=xp+i;

zz=1024; zz*=y_offset; zz+=xx;

mem+=zz;

if(i!=0)

{

*cross_buf=*mem; *mem=255; cross_buf++;

}

}

}

recover_cross(xp,yp,cross_buf)

int xp,yp;

unsigned char huge *cross_buf;

{

int i,xx,yy;

unsigned char huge *mem;

unsigned char huge *mem_keep;

int block,y_offset;

unsigned char status;

long zz;

mem=mem_keep=(unsigned char huge *)MK_FP(BASEMEM,0);

for(i=-10;i<=10;i++)

{

mem=mem_keep;

yy=yp+i; xx=xp;

```

        block=yy/64;    y_offset=yy%64;
        zz=1024;        zz*=y_offset;    zz+=xx;                mem+=zz;
        status=((block<<5)&0xe0)|0x16;
        outp(CONREG,status);
        *mem=*cross_buf;    cross_buf++;
    }
    mem=mem_keep;
    block=yp/64;    y_offset=yp%64;
    status=((block<<5)&0xe0)|0x16;
    outp(CONREG,status);
    for(i=-10;i<=10;i++)
    {
        mem=mem_keep;
        xx=xp+i;
        zz=1024;        zz*=y_offset;    zz+=xx;                mem+=zz;
        if(i!=0)
        {
            *mem=*cross_buf;    cross_buf++;
        }
    }
}

grey_level_show()
{
    int i,j,greyscale;
    float g;

    for(j=0;j<480;j++)
    {
        g=j;    g*=255;                g/=480;                greyscale=g;
        for(i=610;i<640;i++)
        {
            point_show(i,j,greyscale);
        }
    }

    number_display(0,578,6,7,9,255);    line_show(567,0,571,0,255);
                                         line_show(606,0,609,0,255);

    number_display(3,578,60,7,9,255);
    number_display(1,587,60,7,9,255);

                                         line_show(567,60,571,60,255);
                                         line_show(606,60,609,60,255);

    number_display(6,578,120,7,9,255);
    number_display(3,587,120,7,9,255);

                                         line_show(567,120,571,120,255);
                                         line_show(606,120,609,120,255);

    number_display(9,578,180,7,9,255);
    number_display(5,587,180,7,9,255);

                                         line_show(567,180,571,180,255);
                                         line_show(606,180,609,180,255);

    number_display(1,578,240,7,9,255);
    number_display(2,587,240,7,9,255);
    number_display(7,596,240,7,9,255);

                                         line_show(567,240,571,240,255);
                                         line_show(606,240,609,240,255);

    number_display(1,578,300,7,9,255);
    number_display(5,587,300,7,9,255);

```

```

number_display(9,596,300,7,9,255);
                                line_show(567,300,571,300,255);
                                line_show(606,300,609,300,255);

number_display(1,578,360,7,9,255);
number_display(9,587,360,7,9,255);
number_display(1,596,360,7,9,255);
                                line_show(567,360,571,360,255);
                                line_show(606,360,609,360,255);

number_display(2,578,420,7,9,255);
number_display(2,587,420,7,9,255);
number_display(3,596,420,7,9,255);
                                line_show(567,420,571,420,255);
                                line_show(606,420,609,420,255);

number_display(2,578,474,7,9,255);
number_display(5,587,474,7,9,255);
number_display(5,596,474,7,9,255);
                                line_show(567,479,571,479,255);
                                line_show(606,479,609,479,255);
}

equalize_screen(image,x,y,t1,t2)
unsigned char huge *image;
int x,y,t1,t2;
{
    int i,j,p,t;
    long zz;
    float k,b,w;

/*          function w=[255/(t2-t1)] *(b-t1)          */

    if(t1>t2) {      t=t1;  t1=t2; t2=t;  }
    k=255;           k/=(t2-t1);

    for(j=YO-y/2;j<YO-y/2+y;j++)
    {
        for(i=XO-x/2;i<XO-x/2+x;i++)
        {
            zz=j-(YO-y/2); zz*=x; zz+=(i-(XO-x/2));
            p=*(image+zz);
            if(p<=t1)      point_show(i,j,0);
            if(p>=t2)      point_show(i,j,254);
            if(p>t1 && p<t2)
            {
                b=p-t1;      b*=k;  p=b;
                point_show(i,j,p);
            }
        }
    }

    printf("\npress any key to display new color scale.\n");
    getch();
    equalized_grey_level_show(t1,t2);
}

equalize_mem(image,x,y,t1,t2)

```

```

unsigned char huge *image;
int x,y,t1,t2;
{
int i,j,t,p;
long zz;
float k,w,b;

if(t1>t2)      {      t=t1;  t1=t2; t2=t;  }
k=255;        k/=(t2-t1);

for(j=0;j<y;j++)
{
for(i=0;i<x;i++)
{
zz=j;  zz*=x; zz+=i;
p=*(image+zz);
if(p<=t1)      *(image+zz)=0;
if(p>=t2)      *(image+zz)=254;
if(p>t1 && p<t2)
{
w=p-t1;          w*=k;  p=w;
*(image+zz)=p;
}
}
}
}

```

```

#define BASEMEM 0xD000 /* memory base address */
#define RB 0x300      /* pcplus register base address */

```

```

ini_lut(void)
{
int j;
unsigned char i,ii;

```

```

/* initialize bank 0 to linear */

```

```

outp(RB+1,0x00);
for(j=0;j<=255;j++)
{
i=j;
outp(RB+2,i); outp(RB+3,i); }

```

```

outp(RB+1,0x01);
for(j=0;j<=255;j++)
{
i=j;
outp(RB+2,i); outp(RB+3,i); }

```

```

outp(RB+1,0x02);
for(j=0;j<=255;j++)
{
i=j;
outp(RB+2,i); outp(RB+3,i); }

```

```

outp(RB+1,0x03);
for(j=0;j<=255;j++)
{
i=j;

```

```
    outp(RB+2,i); outp(RB+3,i); }
```

```
/* initialize bank 4 to bit-plane-oriented */
```

```
outp(RB+1,0x82);
for(j=0;j<=255;j++)
{
    i=j;    outp(RB+2,i);
    ii=0;
    if(i<64 && i>0)                ii=128;
    if(i>=64 && i<=128)    ii=128-2*(i-64);
    if(i==255)                ii=128;
    /*
    if( (i & 0x01)==1)    ii=36;
    if( (i & 0x02)==2)    ii=72;
    if( (i & 0x03)==3)    ii=108;
    if( (i & 0x04)==4)    ii=144;
    if( (i & 0x05)==5)    ii=180;
    if( (i & 0x06)==6)    ii=216;
    if( (i & 0x07)==7)    ii=252;
    */
    outp(RB+3,ii);
}
```

```
printf("\n");
```

```
outp(RB+1,0x81);
for(j=0;j<=255;j++)
{
    i=j;    outp(RB+2,i);
    ii=0;
    if(i<64 && i>0)                ii=i*2;
    if(i>=64 && i<192)    ii=128;
    if(i>=192)                ii=128-2*(i-192);
    if(i==254)                ii=0;
    if(i==255)                ii=128;
    /*
    if( (i & 0x08)==0x08)    ii=36;
    if( (i & 0x10)==0x10)    ii=72;
    if( (i & 0x18)==0x18)    ii=108;
    if( (i & 0x20)==0x20)    ii=144;
    if( (i & 0x28)==0x28)    ii=180;
    if( (i & 0x30)==0x30)    ii=216;
    if( (i & 0x38)==0x38)    ii=252;
    */
    outp(RB+3,ii);
}
printf("\n");
```

```
outp(RB+1,0x80);
for(j=0;j<=255;j++)
{
    i=j;    outp(RB+2,i);
    ii=0;
    if(i>128 && i<192)    ii=2*(i-128);
    if(i>=192)                ii=128;
    if(i==254)                ii=220;
    /*
```

```

        if( (i & 0x40) == 0x40) ii=85;
        if( (i & 0x80) == 0x80) ii=170;
        if( (i & 0xc0) == 0xc0) ii=255;
        */
        outp(RB+3,ii);
    }
}

ini_reg(void)
{

    outp(RB,0x16); outp(RB+1,3); outp(RB+2,0);
    outp(RB+3,0); outp(RB+4,0x36);outp(RB+5,0x02);
    outp(RB+6,0); outp(RB+7,0); outp(RB+8,0x40);
    outp(RB+9,0); outp(RB+10,0); outp(RB+11,0);

}

/* display a number at (x,y), size is (size_x X size_Y) */

number_display(number,x,y,size_x,size_y,brightness)
int number,x,y,size_x,size_y,brightness;
{
    int hx,hy,i,j,b;

    b=brightness; hx=size_x/2; hy=size_y/2;

    switch(number)
    {
    case 0: line_show(x-hx,y-hy,x+hx,y-hy,b);
            line_show(x-hx,y-hy,x-hx,y+hy,b);
            line_show(x-hx,y+hy,x+hx,y+hy,b);
            line_show(x+hx,y-hy,x+hx,y+hy,b);
            break;

    case 1: line_show(x,y-hy,x,y+hy,b);
            break;

    case 2: line_show(x-hx,y-hy,x+hx,y-hy,b);
            line_show(x+hy,y-hy,x+hx,y,b);
            line_show(x+hy,y,x-hx,y,b);
            line_show(x-hx,y,x-hx,y+hy,b);
            line_show(x-hx,y+hy,x+hx,y+hy,b);
            break;

    case 3: line_show(x-hx,y-hy,x+hx,y-hy,b);
            line_show(x+hx,y-hy,x+hx,y+hy,b);
            line_show(x-hx,y,x+hx,y,b);
            line_show(x-hx,y+hy,x+hx,y+hy,b);
            break;

    case 4: line_show(x-hx,y-hy,x-hx,y,b);
            line_show(x-hx,y,x+hx,y,b);
            line_show(x+hx,y-hy,x+hx,y+hy,b);
            break;
    }
}

```

```

case 5: line_show(x-hx,y-hy,x+hx,y-hy,b);
        line_show(x-hx,y-hy,x-hx,y,b);
        line_show(x-hx,y,x+hx,y,b);
        line_show(x+hx,y,x+hx,y+hy,b);
        line_show(x-hx,y+hy,x+hx,y+hy,b);
        break;

case 6: line_show(x-hx,y-hy,x+hx,y-hy,b);
        line_show(x-hx,y-hy,x-hx,y+hy,b);
        line_show(x-hx,y,x+hx,y,b);
        line_show(x-hx,y+hy,x+hx,y+hy,b);
        line_show(x+hx,y,x+hx,y+hy,b);
        break;

case 7: line_show(x-hx,y-hy,x+hx,y-hy,b);
        line_show(x+hx,y-hy,x+hx,y+hy,b);
        break;

case 8: line_show(x-hx,y-hy,x+hx,y-hy,b);
        line_show(x-hx,y,x+hx,y,b);
        line_show(x-hx,y+hy,x+hx,y+hy,b);
        line_show(x-hx,y-hy,x-hx,y+hy,b);
        line_show(x+hx,y-hy,x+hx,y+hy,b);
        break;

case 9: line_show(x-hx,y-hy,x+hx,y-hy,b);
        line_show(x-hx,y,x+hx,y,b);
        line_show(x-hx,y+hy,x+hx,y+hy,b);
        line_show(x-hx,y-hy,x-hx,y,b);
        line_show(x+hx,y-hy,x+hx,y+hy,b);
        break;

```

```

}
}

```

```

histogram_image(image,x,y,hist)

```

```

unsigned char huge *image;

```

```

int x,y;

```

```

long *hist;

```

```

{

```

```

int i,j,z;

```

```

long zzz;

```

```

for(i=0;i<256;i++) *(hist+i)=0;

```

```

for(j=0;j<y;j++)

```

```

    for(i=0;i<x;i++)

```

```

    {

```

```

        zzz=j; zzz*=x; zzz+=i;

```

```

        z=*(image+zzz);

```

```

        *(hist+z)+=1;

```

```

    }

```

```

}

```

```

show_hist_vertical(hist)

```

```

long *hist;

```

```

{

```

```

int i,ii,j,a,d;
float max,rate,g,gg;
double buf;

max=0; /* ii=0 */
for(i=0;i<256;i++)
    if(*(hist+i)>max) { max=*(hist+i); /* ii=i; */ }
rate=34/max; a=max;
/*
printf("max=%d, grey level=%d.\n",a,ii);
*/
line_show(566,0,566,470,255);
g=479; g/=255;
for(i=0;i<256;i++)
{
    buf=*(hist+i); buf*=rate;
    if(buf>0 && buf<2) a=2;
    else a=buf;
    gg=i*g; d=gg;
    line_show(566-a,d,567,d,255);
}

/* get digits from a integer, nd is the number of the digits */

integer_to_digit(integer,nd,d)
int integer,nd,*d;
{
    int i,n;

    n=integer;

    for(i=0;i<nd;i++)
    {
        n/=10; n*=10; *(d+i)=integer-n; integer/=10; n/=10;
    }
}

equalized_grey_level_show(t1,t2)
int t1,t2;
{
    int i,j,greys,d[3],t;
    float g,dt;

    dt=t2-t1; dt/=8;
    for(j=0;j<480;j++)
    {
        g=j; g*=255; g/=480; greys=g;
        for(i=610;i<640;i++)
        {
            point_show(i,j,greys);
        }
    }
    for(j=0;j<480;j++)
        for(i=532;i<610;i++) point_show(i,j,0);
    line_show(567,0,567,480,255);
}

```

```

t=t1;
integer_to_digit(t,3,&d[0]);
number_display(d[2],578,6,7,9,255);
number_display(d[1],587,6,7,9,255);
number_display(d[0],596,6,7,9,255);

line_show(567,0,571,0,255);
line_show(606,0,609,0,255);

for(i=60;i<=420;i+=60)
{
j=i/60;
t=t1+j*dt;
integer_to_digit(t,3,&d[0]);
number_display(d[2],578,i,7,9,255);
number_display(d[1],587,i,7,9,255);
number_display(d[0],596,i,7,9,255);

line_show(567,i,571,i,255);
line_show(606,i,609,i,255);

}

/*
t=t1+2*dt;
integer_to_digit(t,3,&d[0]);
number_display(d[0],578,120,7,9,255);
number_display(d[1],587,120,7,9,255);
number_display(d[2],596,120,7,9,255);

line_show(567,120,571,120,255);
line_show(606,120,609,120,255);

t=t1+3*dt;
integer_to_
number_display(9,578,180,7,9,255);
number_display(5,587,180,7,9,255);

line_show(567,180,571,180,255);
line_show(606,180,609,180,255);

number_display(1,578,240,7,9,255);
number_display(2,587,240,7,9,255);
number_display(7,596,240,7,9,255);

line_show(567,240,571,240,255);
line_show(606,240,609,240,255);

number_display(1,578,300,7,9,255);
number_display(5,587,300,7,9,255);
number_display(9,596,300,7,9,255);

line_show(567,300,571,300,255);
line_show(606,300,609,300,255);

number_display(1,578,360,7,9,255);
number_display(9,587,360,7,9,255);
number_display(1,596,360,7,9,255);

line_show(567,360,571,360,255);
line_show(606,360,609,360,255);

number_display(2,578,420,7,9,255);
number_display(2,587,420,7,9,255);
number_display(3,596,420,7,9,255);

line_show(567,420,571,420,255);
line_show(606,420,609,420,255);

*/
t=t2;
integer_to_digit(t,3,&d[0]);

```

```

number_display(d[2],578,474,7,9,255);
number_display(d[1],587,474,7,9,255);
number_display(d[0],596,474,7,9,255);

line_show(567,479,571,479,255);
line_show(606,479,609,479,255);

}

/* the PSF is a image file */
/* maximum neighbor: I=c2 [O(j) - c1 max(s1*O(j-1), s1*O(j+1))] */
/*
convolution: I=H*O;
NO! deconvolution: O(n)=O(n-1)-a(I-O(n-1)*H);
*/
/* modified maxnb2.c, after convolution, the convolved image will be saved
into hard disk, the memory requirement can be reduced, the saved convolved
images will be named after original image with wild card '.cv1' and '.cv2' */

#include "ypdef.h"
#include <stdio.h>
#include <alloc.h>
#include <dos.h>
#include <math.h>

#define XO 320
#define YO 240

/*
void deconvolution(double huge *im, double huge *ob, int x, int y,
double huge *h, int hx, int hy);
*/
void convolution(unsigned char huge *im, double huge *ob, int x, int y,
double huge *h, int hx, int hy);
void matrix_sub(double huge *a, double huge *b, double huge *c, int x, int y);
void matrix_scale(double huge *a, int x, int y, float c);
void readdat_double(char datfile[30], double huge *c, long size);
void savedat_double(char datfile[30], double huge *c, long size);

void image_show_double(double huge *img, int sx, int sy, int x, int y);

main()
{
char cname[30],uname[30],lname[30],hname1[30],hname2[30],outname[30],
convname1[30],convname2[30];
unsigned char huge *im1;
unsigned char huge *im2;
unsigned char huge *im3;
double huge *im;
double huge *h1;
double huge *h2;

double max1,max2,weight1,weight2;
float kx,ky,xb,yb,pp1,pp2,c1,c2;
unsigned int xx,yy;
int hx,hy,x,y,i,j,redo,pp,p1,p2;
char sv;
long zz,size;

```

```
unsigned char huge *psf1;
unsigned char huge *psf2;

printf("%lu bytes free in ram.\n",farcoreleft());
printf("center, upper, lower image names: ");
scanf("%s %s %s",cname,uname,lname);      printf("\n");
printf("up/low PSF image name: ");
scanf("%s %s",hname1,hname2);             printf("\n");

im1=(unsigned char huge *)read_image(uname,&x,&y);
image_show(im1,0,0,x,y);

psf1=(unsigned char huge *)read_image(hname1,&hx,&hy);
psf2=(unsigned char huge *)read_image(hname2,&hx,&hy);

size=x;      size*=y;

im=(double huge *)farcalloc(size,sizeof(double));

/*
ob=(double huge *)farcalloc(size,sizeof(double));
*/

size=hx;      size*=hy;
h1=(double huge *)farcalloc(size,sizeof(double));
h2=(double huge *)farcalloc(size,sizeof(double));
for(zz=0;zz<size;zz++)
    {      *(h1+zz)=*(psf1+zz);  *(h2+zz)=*(psf2+zz);  }
weight1=weight2=0;
for(zz=0;zz<size;zz++)
    {
        if(*(h1+zz)<0) *(h1+zz)*=-1;
        if(*(h2+zz)<0) *(h2+zz)*=-1;
        weight1+=*(h1+zz);
        weight2+=*(h2+zz);
    }
printf("\nweight=%lf.\n",weight1);
for(zz=0;zz<size;zz++)
    {
        *(h1+zz)/=weight1;      *(h2+zz)/=weight2;
        printf("%lf ",*(h1+zz));
    }
/*
if( im==NULL || ob==NULL)
    {
        printf("no enough memory.\n");
        exit();
    }
else
    printf("%lu bytes allocated.\n",size);
*/

printf("\n");
for(j=0;j<y;j++)
    for(i=0;i<x;i++)
        {
```

```

        zz=j;  zz*=x; zz+=i;
        *(im+zz)=*(im1+zz);
    }
    convolution(im1,im,x,y,h1,hx,hy);
/*
    image_show_double(im,0,y,x,y);
*/
    strcpy(convname1,uname);    strcat(convname1,".cv1");
    save_image(convname1,XO-x/2,YO-y/2,XO-x/2+x-1,YO-y/2+y-1);
    im2=(unsigned char huge *)read_image(lname,&x,&y);

    image_show(im2,x,0,x,y);
    for(j=0;j<y;j++)
        for(i=0;i<x;i++)
        {
            zz=j;  zz*=x; zz+=i;
            *(im+zz)=*(im2+zz);
        }
    convolution(im2,im,x,y,h2,hx,hy);

    strcpy(convname2,lname);    strcat(convname2,".cv2");
    save_image(convname2,XO-x/2,YO-y/2,XO-x/2+x-1,YO-y/2+y-1);
    farfree((double far *)im);
    im3=(unsigned char huge *)read_image(cname,&x,&y);
    image_show(im3,2*x,0,x,y);

    do{
        printf("\ninput c1, c2: ");    scanf("%f %f",&c1,&c2);    printf("\n");
        for(j=0;j<y;j++)
            for(i=0;i<x;i++)
            {
                zz=j;  zz*=x; zz+=i;
                p1=*(im1+zz); p2=*(im2+zz);
                pp1=(p1>p2)? p1:p2;  pp1*=c1;
                pp2=*(im3+zz);
                pp2-=pp1;    pp2*=c2;
                if(pp2<0)    pp2=0;
                if(pp2>255)    pp2=255;
                pp=pp2;
                point_show(XO-x/2+i,YO-y/2+j,pp);
            }
        printf("OK?(y) ");
    }while(getche()!='y');

    printf("\nsave image?(y) ");
    if(getche()=='y')
    {
        printf("\noutput image name: ");
        scanf("%s",outname);    printf("\n");
        save_image(outname,XO-x/2,YO-y/2,XO-x/2+x-1,YO-y/2+y-1);
    }
    printf("\n");
}

/*
    deconvolution:

```

$$O(n) = O(n-1) - a(I - O(n-1) * H);$$

```

void deconvolution(double huge *im, double huge *ob, int x, int y,
    double huge *h, int hx, int hy)
{
    int i,j,p,k1,k2;
    long zz,z;
    double huge *c;

    zz=x;  zz*=y;
    c=(double huge *)calloc(zz,sizeof(double));

    for(j=0;j<y;j++)
        for(i=0;i<x;i++)
            {
                zz=j;  zz*=x;  zz+=i;
                *(ob+zz)=*(im+zz);
            }
    for(i=0;i<1;i++)
        {
            printf("\n---%d\n",i);
            convolution(c,ob,x,y,h,hx,hy);
            image_show_double(c,0,0,x,y);
            matrix_sub(im,c,c,x,y);
            image_show_double(c,x+10,0,x,y);
            matrix_scale(c,x,y,1.2);
            matrix_sub(ob,c,ob,x,y);
            matrix_scale(ob,x,y,0.9);
            image_show_double(ob,2*(x+10),0,x,y);
        }
    farfree((double far *)c);
}

*/
/*
    convolution:    IM=OB*H;

                    IM and OB have the same dimension (x,y),
                    H's dimension is (hx,hy), if hx or hy is even
                    number, it will be minus 1.

*/

void convolution(unsigned char huge *img, double huge *ob, int x, int y,
    double huge *h, int hx, int hy)
{
    int hhx,hhy,i,j,k1,k2,yy,p;
    double c,c1,c2;
    long z1;

    hhx=(hx-1)/2;  hhy=(hy-1)/2;
    z1=hhy;        z1*=hx;        z1+=hhx;
    h+=z1;

    if(y>36)        yy=y/3;

```

```

else          yy=y;
for(i=0;i<yy;i++)    printf("_");    printf("\r");
for(j=0;j<y;j++)
{
    if(y==yy)        printf("+");
    else              if(j==(j/3)*3)    printf("+");
    for(i=0;i<x;i++)
    {
        z1=j;    z1*=x; z1+=i;
        c=0;
        for(k2=-hhy;k2<=hhy;k2++)
            for(k1=-hhx;k1<=hhx;k1++)
                if((j+k2)>=0 && (j+k2)<y && (i+k1)>=0 && (i+k1)<x)
                {
                    c1=(ob+z1+k2*x+k1);
                    c2=(h+k2*hx+k1);
                    c+=(c1*c2);
                }

        p=c;
        *(img+z1)=p;
        point_show(XO-x/2+i,YO-y/2+j,p);
    }
}

```

```

void matrix_sub(double huge *a, double huge *b, double huge *c, int x, int y)
{
    int i,j;

    for(j=0;j<y;j++)
        for(i=0;i<x;i++)
        {
            *c=*a-*b;    a++;    b++;    c++;
        }
}

```

```

void matrix_scale(double huge *a, int x, int y, float c)
{
    int i,j;

    for(j=0;j<y;j++)
        for (i=0;i<x;i++)
            { (*a)*=c;    a++; }
}

```

```
#define SPEC_RANG 16
```

```

void savedat_double(char datfile[30], double huge *c, long size)
{
    FILE *fp;
    double *buf;
    int fnum,bufsize,i,j;
    char fname[30],s0[30];

    bufsize=SPEC_RANG*16;
    fnum=size/bufsize;
}

```

```

if(NULL==(buf=(double *)calloc(bufsize,sizeof(double))))
{
    printf("no enough memory, exit.\n");    exit(1);
}
for(i=0;i<fnum;i++)
{
    for(j=0;j<bufsize;j++)
    {
        *(buf+j)=*c;        c++;
    }
    itoa(i,s0,10);
    strcpy(fname,datfile);
    strcat(fname,s0);
    if((fp=fopen(fname,"wb"))==NULL)
    {
        printf("data file open failure.\n");
        exit(1);
    }
    fwrite((double *)buf,sizeof(double),bufsize,fp);
    fclose(fp);
}
free((double *)buf);
}

void readdat_double(char datfile[30], double huge *c, long size)
{
    FILE *fp;
    double *buf;
    int fnum,bufsize,i,j;
    char fname[30],s0[30];

    bufsize=SPEC_RANG*16;
    fnum=size/bufsize;
    if(NULL==(buf=(double *)calloc(bufsize,sizeof(double))))
    {
        printf("no enough memory, exit.\n");    exit(1);
    }
    for(i=0;i<fnum;i++)
    {
        itoa(i,s0,10);
        strcpy(fname,datfile);
        strcat(fname,s0);
        if((fp=fopen(fname,"rb"))==NULL)
        {
            printf("data file open failure.\n");
            exit(1);
        }
        fread((double *)buf,sizeof(double),bufsize,fp);
        fclose(fp);
        for(j=0;j<bufsize;j++)
        {
            *c=*(buf+j);
            c++;
        }
    }
    free((double *)buf);
}

```

```

void image_show_double(double huge *img, int sx, int sy, int x, int y)
{
    int k1,k2,p;
    long zz;

    for(k2=0;k2<y;k2++)
        for(k1=0;k1<x;k1++)
        {
            zz=k2; zz*=x; zz+=k1;
            p=*(img+zz); if(p<0) p=-p; if(p>255) p=255;
            point_show(sx+k1,sy+k2,p);
        }
}

/*.....image division, can save results as float number .....*/

#include "ypdef.h"
#include <stdio.h>
#include <dos.h>
#include <alloc.h>
#define BASEMEM 0xD000
#define CONREG 0x300

main ()
{
    char imgname1[50],imgname2[50],outname[50],ffile[50];
    unsigned char huge *image;
    unsigned char huge *board;
    unsigned char huge *p;
    int choice,c,i,j,x,y,ws,wrongname,pp,ppp;
    long size,zz;
    int n;
    float s,p1,p2,p3;
    FILE *fp;
    /* read an image from disk into memory */

    printf("%lu bytes free in ram.\n",farcoreleft());
    printf("div.=(image1/image2)*scale_factor.\n");

    printf("the names of image1, image2, scale_factor: ");
    scanf("%s %s %f",imgname1,imgname2,&s);          printf("\n");
    printf("do you want to save division results as float number:(y) ");
    if(getche()=='y')
    {
        printf("\nfloat data file name: ");
        scanf("%s",ffile);          printf("\n");
    }
    fp=fopen(ffile,"wb");

    image=(unsigned char huge *)read_image(imgname1,&x,&y);
    printf("display image1.\n");
    image_show(image,0,0,x,y);
    farfree((unsigned char far *)image);
    p=(unsigned char huge *)farcalloc(1,1);
    image=(unsigned char huge *)read_image(imgname2,&x,&y);
    for(j=0;j<y;j++)

```

```

        for(i=0;i<x;i++)
        {
            zz=j;  zz*=x; zz+=i;
            get_point_value(i,j,p);  pp=*p; p1=pp;
            ppp=*(image+zz);          p2=ppp;
            p3=(p1/p2);
            fwrite(&p3,sizeof(float),1,fp);
            p3*=s;  pp=p3;
            if(pp>255)  pp=255;
            point_show(i,j,pp);
        }

fclose(fp);
printf("save image?(y): ");
if(getche()=='y')
{
    printf("\noutput image name: ");
    scanf("%s",outname);  printf("\n");
    save_image_screen(outname,0,0,x-1,y-1);
}

save_image_screen(imname,x1,y1,x2,y2)
char imname[50];
int x1,y1,x2,y2;
{
    int wrongname,i,j,x,y,k,start_line,end_line,start_blc,end_blc,
        start_y_offset,end_y_offset;
    unsigned char huge *mem;
    unsigned char huge *mem_keep;
    unsigned char ini_stat,stat;
    unsigned char p;
    long zz;
    FILE *fp;
    char ix[8],iy[8];

    do{
        wrongname=0;
        if((fp=fopen(imname,"wb"))==NULL)
        {
            wrongname=1;
            printf("wrong name! try again: ");
            scanf("%s",imname);  printf("\n");
        }
    }while(wrongname==1);
    mem=(unsigned char huge *)MK_FP(BASEMEM,0);
    x=x2-x1+1;  y=y2-y1+1;
    itoa(x,ix,10);  itoa(y,iy,10);
    for(i=0;i<8;i++)  fputc(ix[i],fp);
    for(i=0;i<8;i++)  fputc(iy[i],fp);

    start_blc=y1/64;          end_blc=(y2+1)/64;
    start_y_offset=y1%64;     end_y_offset=(y2+1)%64;
    mem_keep=mem+x1;
    zz=start_y_offset;  zz*=1024;  zz+=x1;
    mem +=zz;

```

```

ini_stat=0x16;
for(i=start_blc;i<=end_blc;i++)
{
    stat=(i<<5)&0xe0;
    stat |=ini_stat;
    outp(CONREG,stat);

    if(i==start_blc)        start_line=start_y_offset;
        else start_line=0;
    if(i==end_blc) end_line=end_y_offset;
        else end_line=64;
    for(j=start_line;j<end_line;j++)
    {
        for(k=0;k<x;k++)
        {
            p=*mem;        fputc(p,fp);
            mem++;
        }
        mem+=(1024-x);
    }
    mem=mem_keep;
}

printf("----- a %d X %d image %s is saved.\n",x,y,imname);
}

#include <stdio.h>
#include <alloc.h>
#include <dos.h>

#define BP 0x3F8        /* for com1. baseport register address*/
#define CONREG 0x300    /* pcplus register base address */

unsigned char huge *read_image(char imname[30], int *x, int *y);
void image_show(unsigned char huge *im_ad, int sx, int sy, int x, int y);

main()
{
    int x,y,sx,sy;
    char imname[30];
    unsigned char huge *mem;

    printf("%lu bytes free.\n",farcoreleft());
    printf("input image file name, upper-left corner (sx,sy):");
    scanf("%s %d %d",imname,&sx,&sy);        printf("\n");
    mem=(unsigned char huge *)read_image(imname,&x,&y);
    image_show(mem,sx,sy,x,y);
    printf("\n");
}

```

1. A method of measuring concentration of an analyte in a medium, the method comprising the steps of:

- a) introducing, into the medium, a probe substance which becomes luminescent when irradiated and whose luminescence is quenched by the analyte in the medium,
- b) irradiating the medium with linearly polarized light so as to cause the probe substance to become luminescent,
- c) resolving luminescence emitted by the probe substance into vector components parallel and perpendicular to a plane of polarization of said linearly polarized light,
- d) calculating a luminescence anisotropy, as a function of said vector components, in space or in time, of the irradiated medium, and
- e) applying a mathematical function which relates said luminescence anisotropy to said concentration of the analyte in said medium, the mathematical function being in the form of an equation.

2. The method of Claim 1, further comprising the step of calculating a luminescence lifetime of the probe substance according to the following equation:

$$\tau = \frac{\tau_0}{1 + K_D [Q]}$$

wherein τ is the luminescence lifetime, $[Q]$ is the concentration of the analyte, K_D is a dynamic quench constant, and τ_0 is a luminescence lifetime in the absence of the analyte.

3. The method of Claim 1, further comprising the step of selecting the equation as follows:

$$[Q] = \frac{A_0 - A(6\bar{R}\tau_0 + 1)}{K_D(A - A_0)}$$

wherein A is fluorescence anisotropy, defined as

$$A = \frac{I_{\parallel}G - I_{\perp}}{I_{\parallel}G + 2I_{\perp}}$$

wherein $I_{\parallel}$ and $I_{\perp}$ are intensities of luminescence emission with their
 5 electric vectors respectively parallel and perpendicular to the linearly
 polarized light of step (b), G is an empirical correction factor used to
 correct for transmission efficiency in parallel and perpendicular planes,
 $\bar{R}$ is mean molecular rotation time in radians/sec, A_0 is fluorescence an-
 isotropy in a "frozen" state, in the absence of Brownian rotation, K_D is
 10 a dynamic quench constant, τ_0 is a luminescence lifetime in the absence of
 quencher, and [Q] is the concentration of the analyte which is to be
 determined.

4. The method of Claim 1, further comprising the step of determin-
 ing a topographic distribution of said analyte concentration in said
 15 medium.

5. The method of Claim 1, wherein the probe substance is disposed
 in an optical sensor which comprises a tip and a thermistor, the method
 further comprising the step of correcting anisotropy measurements for
 temperature of the tip of the sensor according to the following equation:

$$\frac{A_0}{A} = \frac{1 + R_g T}{\eta V} \tau$$

wherein R_g is a gas constant, T is temperature, η is viscosity, V is
 hydrodynamic volume of the probe substance, A is luminescence anisotropy,
 A_0 is a luminescence anisotropy in a "frozen" state, in the absence of
 25 Brownian rotation, and τ is a fluorescence lifetime.

6. The method of Claim 1, further comprising the step of determin-
 ing tomographically a topographic distribution of said analyte concentra-

tion within the medium by means of optical serial sectioning methodologies.

7. A method of measuring concentration of an analyte in a medium, the method comprising the steps of:

- 5 a) conjugating probe molecules to a carrier which has an affinity for the analyte, and adding the carrier to the medium, wherein the probe molecules are chosen such that they become luminescent when irradiated,
- b) adding to the medium a quantity of the analyte conjugated with a substance selected from the group consisting of a quencher of lumines-
- 10 cence energy of the probe molecules and an energy transfer acceptor of luminescence energy of the probe molecules,
- c) irradiating the medium with linearly polarized light so as to cause the probe molecules to become luminescent,
- d) resolving luminescence emitted by the probe molecules into vec-
- 15 tor components parallel and perpendicular to a plane of polarization of said linearly polarized light,
- e) calculating a luminescence anisotropy, as a function of said vector components, in space or in time, of the irradiated medium, and
- f) applying an empirically determined mathematical function which
- 20 relates said luminescence anisotropy to said concentration of analyte in said medium.

8. The method of Claim 7, wherein the luminescence anisotropy (A) of step (e) is defined as:

$$A = \frac{I_{\parallel} G - I_{\perp}}{I_{\parallel} G + 2I_{\perp}}$$

25

wherein $I_{\parallel}$ and $I_{\perp}$ are intensities of luminescence emission with their electric vectors respectively parallel and perpendicular to the

linearly polarized light of step (c), and G is an empirical correction factor used to correct transmission efficiency in parallel and perpendicular planes.

9. The method of Claim 7, further comprising the step of determining a topographic distribution of said analyte concentration in said medium.

10. The method of Claim 7, wherein the probe molecules are disposed in an optical sensor which comprises a tip and a thermistor, the method further comprising the step of correcting anisotropy measurements for temperature of the tip of the sensor by means of an empirically determined mathematical function.

11. The method of Claim 7, further comprising the step of determining tomographically a topographic distribution of said analyte concentration within the medium by means of optical serial sectioning methodologies.

12. A method of measuring concentration of an analyte in a medium, the method comprising the steps of:

a) conjugating probe molecules to a carrier which has an affinity for the analyte, and adding the carrier to the medium, wherein the probe molecules are chosen such that they become luminescent when irradiated,

b) adding to the medium a known quantity of the analyte conjugated with a substance selected from the group consisting of a quencher of luminescence energy of the probe molecules and an energy transfer acceptor of luminescence energy of the probe molecules,

c) adding to the medium a sample to be analyzed which contains an unknown quantity of the analyte,

d) irradiating the medium with linearly polarized light so as to cause the probe molecules to become luminescent,

e) resolving luminescence emitted by the probe molecules into vector components parallel and perpendicular to a plane of polarization of said linearly polarized light,

f) calculating a luminescence anisotropy, as a function of said vector components, in space or in time, of the irradiated medium, and

g) applying an empirically determined mathematical function which relates said luminescence anisotropy to said concentration of the unknown quantity of analyte in said medium.

13. The method of Claim 12, wherein the luminescence anisotropy (A) of step (f) is defined as:

$$A = \frac{I_{\parallel}G - I_{\perp}}{I_{\parallel}G + 2I_{\perp}}$$

wherein $I_{\parallel}$ and $I_{\perp}$ are intensities of luminescence emission with their electric vectors respectively parallel and perpendicular to the linearly polarized light of step (d), and G is an empirical correction factor used to correct transmission efficiency in parallel and perpendicular planes.

14. The method of Claim 12, further comprising the step of determining a topographic distribution of said analyte concentration in said medium.

15. The method of Claim 12, wherein the probe molecules are disposed in an optical sensor which comprises a tip and a thermistor, the method further comprising the step of correcting anisotropy measurements for temperature of the tip of the sensor by means of an empirically determined mathematical function.

16. The method of Claim 12, further comprising the step of determining tomographically a topographic distribution of said analyte concen-

tration within the medium by means of optical serial sectioning methodologies.

17. Apparatus for measuring a concentration of an analyte, comprising:

5 a) means for directing polarized light to an optical sensor,

b) the optical sensor comprising means for housing a probe substance which becomes luminescent when irradiated with light, the probe substance being chosen such that its luminescence is quenched by the analyte,

10 c) means for resolving said luminescence, emitted by the probe substance, into vector components parallel and perpendicular to a plane of polarization of said polarized light, and for detecting intensities of said vector components, and

d) means for processing the intensities obtained from the resolving and detecting means, the processing means comprising means for calculating said concentration of the analyte based on values of said intensities.

18. The apparatus of Claim 17, wherein the optical sensor contains a viscous medium, the probe substance and the viscous medium being separated from the analyte by a membrane permeable to the analyte.

19. The apparatus of Claim 17, wherein the calculating means comprises means for calculating analyte concentration according to the following equation:

$$[Q] = \frac{A_0 - A(6\bar{R}\tau_0 + 1)}{K_D(A - A_0)}$$

wherein A is fluorescence anisotropy, defined as

$$A = \frac{I_{\parallel} G - I_{\perp}}{I_{\parallel} G + 2I_{\perp}}$$

wherein $I_{\parallel}$ and $I_{\perp}$ are intensities of luminescence emission with their electric vectors respectively parallel and perpendicular to the polarized light originating from the directing means, G is an empirical correction factor used to correct for transmission efficiency in parallel and perpendicular planes, $\bar{R}$ is mean molecular rotation time in radians/sec, A_0 is fluorescence anisotropy in a "frozen" state, in the absence of Brownian rotation, K_D is a dynamic quench constant, τ_0 is a luminescence lifetime in the absence of quencher, and $[Q]$ is the concentration of the analyte which is to be determined.

20. The apparatus of Claim 17, wherein the analyte is disposed in a medium, the apparatus further comprising means for determining a topographic distribution of said analyte concentration in said medium.

21. The apparatus of Claim 17, wherein the optical sensor comprises a tip and a thermistor, the apparatus further comprising means for correcting anisotropy measurements for temperature of the tip of the sensor according to the following equation:

$$\frac{A_0}{A} = \frac{1 + R_g T}{\eta V} \tau$$

wherein R_g is a gas constant, T is temperature, η is viscosity, V is hydrodynamic volume of the probe substance, A is luminescence anisotropy, A_0 is a luminescence anisotropy in a "frozen" state, in the absence of Brownian rotation, and τ is a fluorescence lifetime.

22. Apparatus for measuring concentration of an analyte, comprising:

- a) means for directing polarized light to an optical sensor,
- b) the optical sensor comprising means for housing a carrier with a

high affinity for the analyte, the carrier being conjugated to probe molecules that become luminescent when irradiated, the optical sensor also housing a known quantity of the analyte conjugated to a substance selected from the group consisting of a quencher of luminescence energy of the probe molecules and an energy transfer acceptor of luminescence energy of the probe molecules,

c) the optical sensor being disposed in a medium which contains an unconjugated form of the analyte,

d) means for resolving said luminescence, emitted by the carrier probe molecules, into vector components parallel and perpendicular to a plane of polarization of said polarized light, and for detecting intensities of said vector components, and

e) means for processing the intensities obtained from the resolving and detecting means, the processing means comprising means for calculating a concentration of the unconjugated form of the analyte contacted by the optical sensor, based on values of said intensities.

23. The apparatus of Claim 22, wherein the optical sensor contains a viscous medium containing the carrier and the conjugated analyte, the carrier and the conjugated analyte being separated from the unconjugated form of the analyte by a membrane permeable to the analyte in its unconjugated form but impermeable to the conjugated analyte.

24. The apparatus of Claim 22, the apparatus further comprising means for determining a topographic distribution of said analyte concentration in said medium.

25. The apparatus of Claim 22, wherein the optical sensor comprises a tip and a thermistor, the apparatus further comprising means for correcting anisotropy measurements for temperature of the tip of the sensor by an empirically determined mathematical function.

26. The apparatus of Claim 22, the apparatus further comprising means for determining a spatial distribution of said analyte concentration within the medium.

27. Apparatus for measuring concentration of an analyte disposed in
5 a medium, the apparatus comprising:

a) means for directing polarized light towards the medium,

b) the medium including a probe substance which becomes luminescent when irradiated with light, the probe substance being chosen such that its luminescence is quenched by the analyte,

10 c) means for resolving said luminescence, emitted by the probe substance, into vector components parallel and perpendicular to a plane of polarization of said polarized light, and for detecting intensities of said vector components, and

d) means for processing the intensities obtained from the resolving
15 and detecting means, the processing means comprising means for calculating said concentration of the analyte based on values of said intensities.

28. The apparatus of Claim 27, wherein the calculating means comprises means for calculating said analyte concentration according to the
20 following equation:

$$[Q] = \frac{A_0 - A(G\bar{R}\tau_0 + 1)}{K_D(A - A_0)}$$

wherein A is fluorescence anisotropy, defined as

$$A = \frac{I_{||}G - I_{\perp}}{I_{||}G + 2I_{\perp}}$$

25 wherein $I_{||}$ and $I_{\perp}$ are intensities of luminescence emission with their

electric vectors respectively parallel and perpendicular to the polarized light originating from the directing means, G is an empirical correction factor used to correct for transmission efficiency in parallel and perpendicular planes, $\bar{R}$ is mean molecular rotation time in radians/sec, A_0 is a fluorescence anisotropy in a "frozen" state, in the absence of Brownian rotation, K_D is a dynamic quench constant, τ_0 is a luminescence lifetime in the absence of quencher, and $[Q]$ is the concentration of the analyte which is to be determined.

29. The apparatus of Claim 27, the apparatus further comprising means for determining a topographic distribution of said analyte concentration in the medium.

30. The apparatus of Claim 27, further comprising means for correcting anisotropy measurements for temperature according to the following equation:

$$\frac{A_0}{A} = \frac{1 + R_g T}{\eta V} \tau$$

wherein R_g is a gas constant, T is temperature, η is viscosity, V is hydrodynamic volume of the probe substance, A is luminescence anisotropy, A_0 is a luminescence anisotropy in a "frozen" state, in the absence of Brownian rotation, and τ is fluorescence lifetime.

31. The apparatus of Claim 27, the apparatus further comprising means for determining tomographically a topographic distribution of said analyte concentration within the medium by means of optical serial sectioning methodologies.

32. Apparatus for measuring concentration of an analyte disposed in a medium, the apparatus comprising:

- a) means for directing polarized light towards the medium,
- b) the medium including carriers conjugated with probe molecules

which become luminescent when irradiated with light, the medium also including a known quantity of the analyte conjugated with substances selected from the group consisting of a quencher of luminescence energy of the probe molecules and an energy transfer acceptor of luminescence energy of the probe molecules,

c) means for resolving said luminescence, emitted by the probe molecules, into vector components parallel and perpendicular to a plane of polarization of said polarized light, and for detecting intensities of said vector components, and

d) means for processing the intensities obtained from the resolving and detecting means, the processing means comprising means for calculating said concentration of the analyte based on values of said intensities.

33. The apparatus of Claim 32, wherein the calculating means comprises means for calculating said analyte concentration according to an empirically determined mathematical function.

34. The apparatus of Claim 32, the apparatus further comprising means for determining a topographic distribution of said analyte concentration in the medium.

35. The apparatus of Claim 32, further comprising means for correcting anisotropy measurements for temperature of the medium according to an empirically determined function.

36. The apparatus of Claim 32, further comprising means for determining tomographically a topographic distribution of said analyte concentration within the medium by means of optical serial sectioning methodologies.

1 / 5

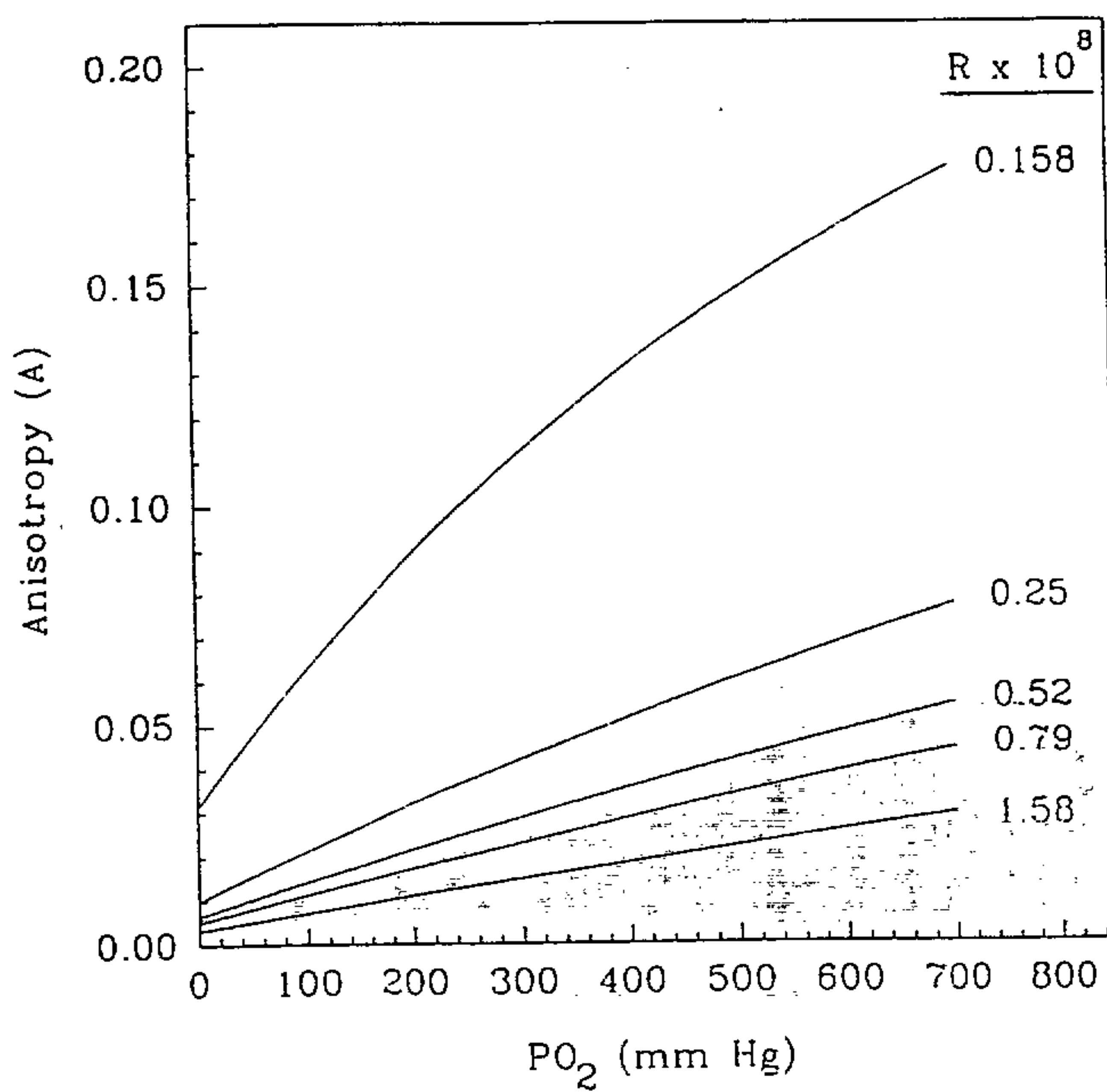


Figure 1

2/5

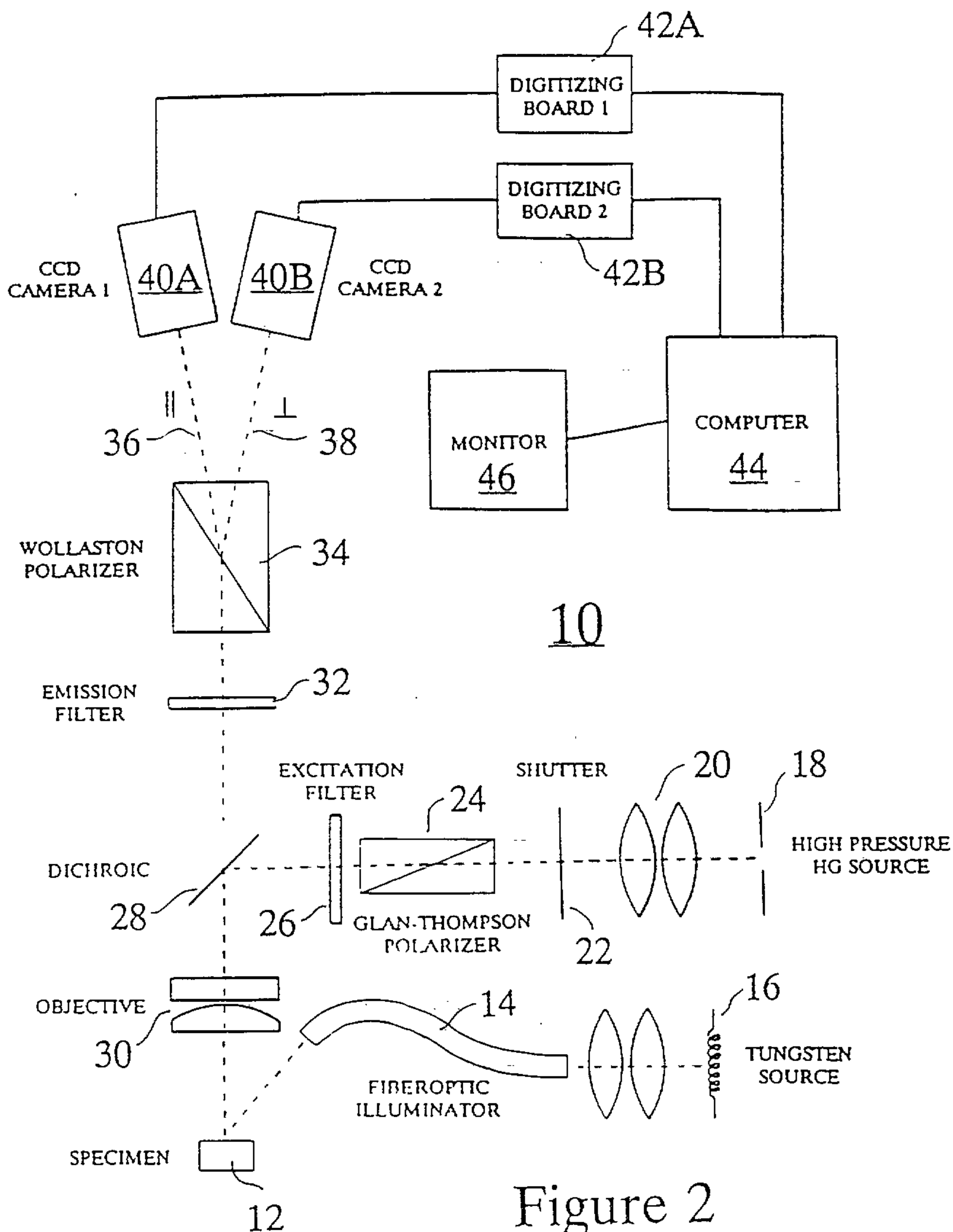


Figure 2

3/5

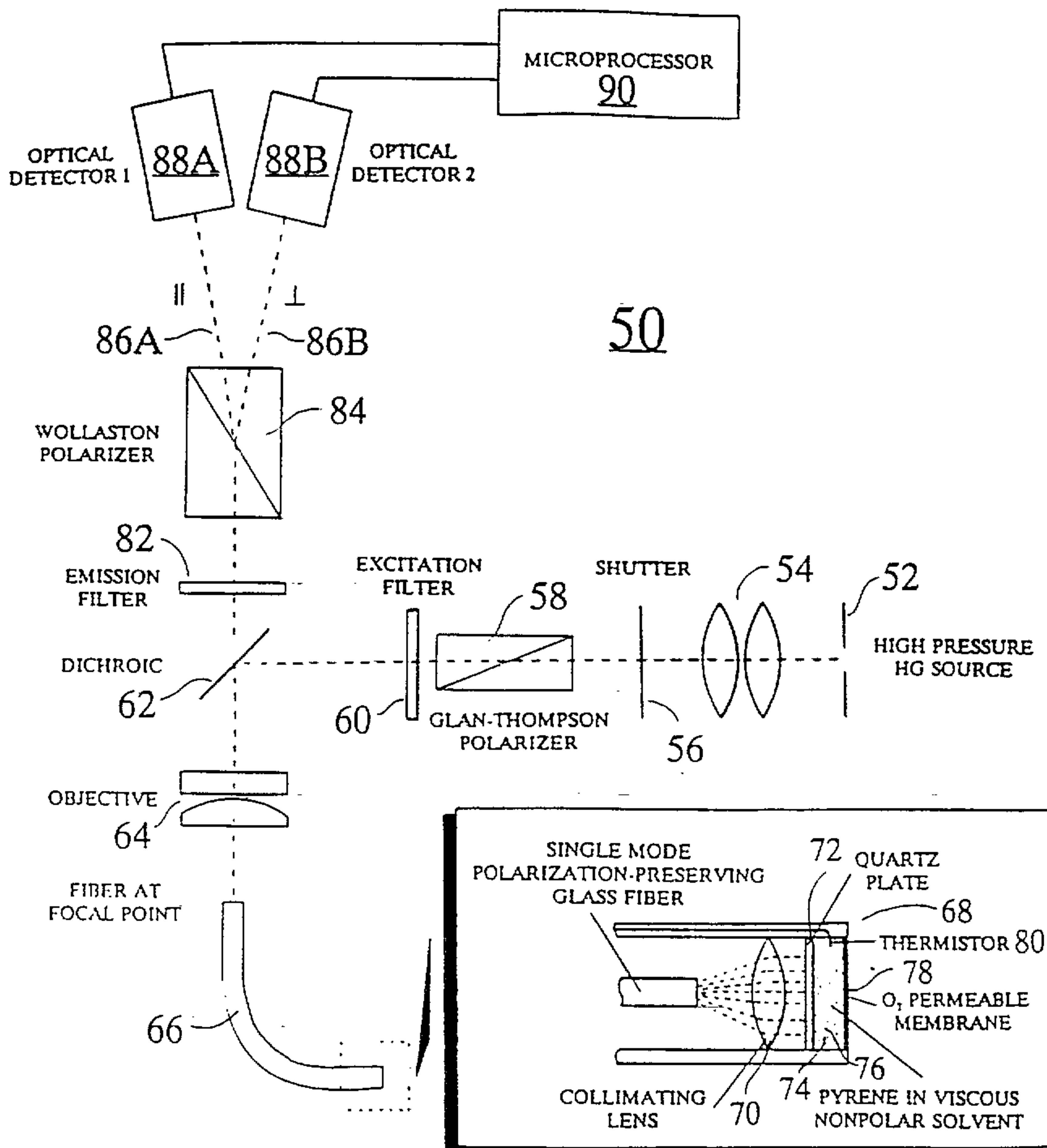


Figure 3

4/5

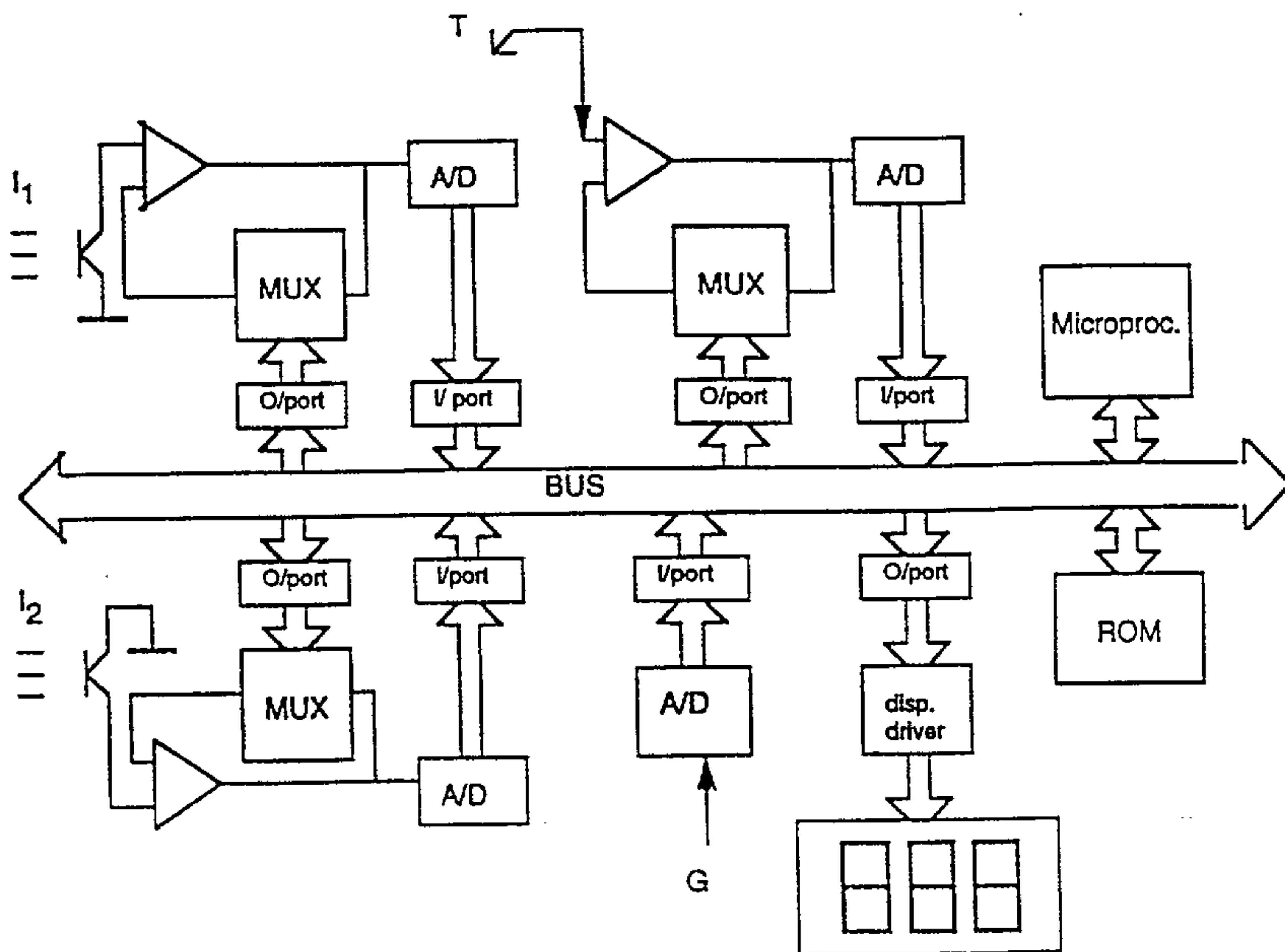


Figure 4

5/5

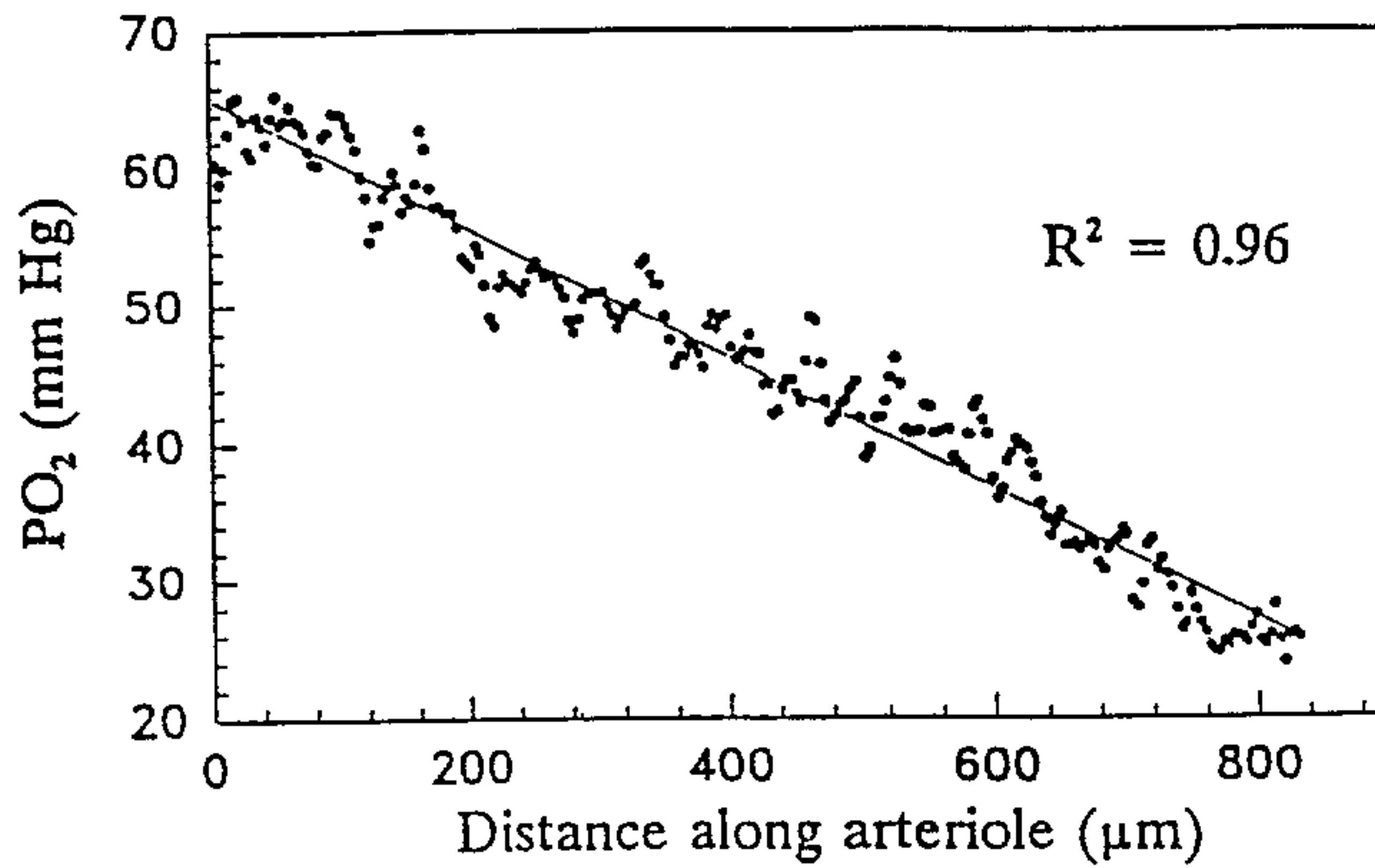


Figure 5A

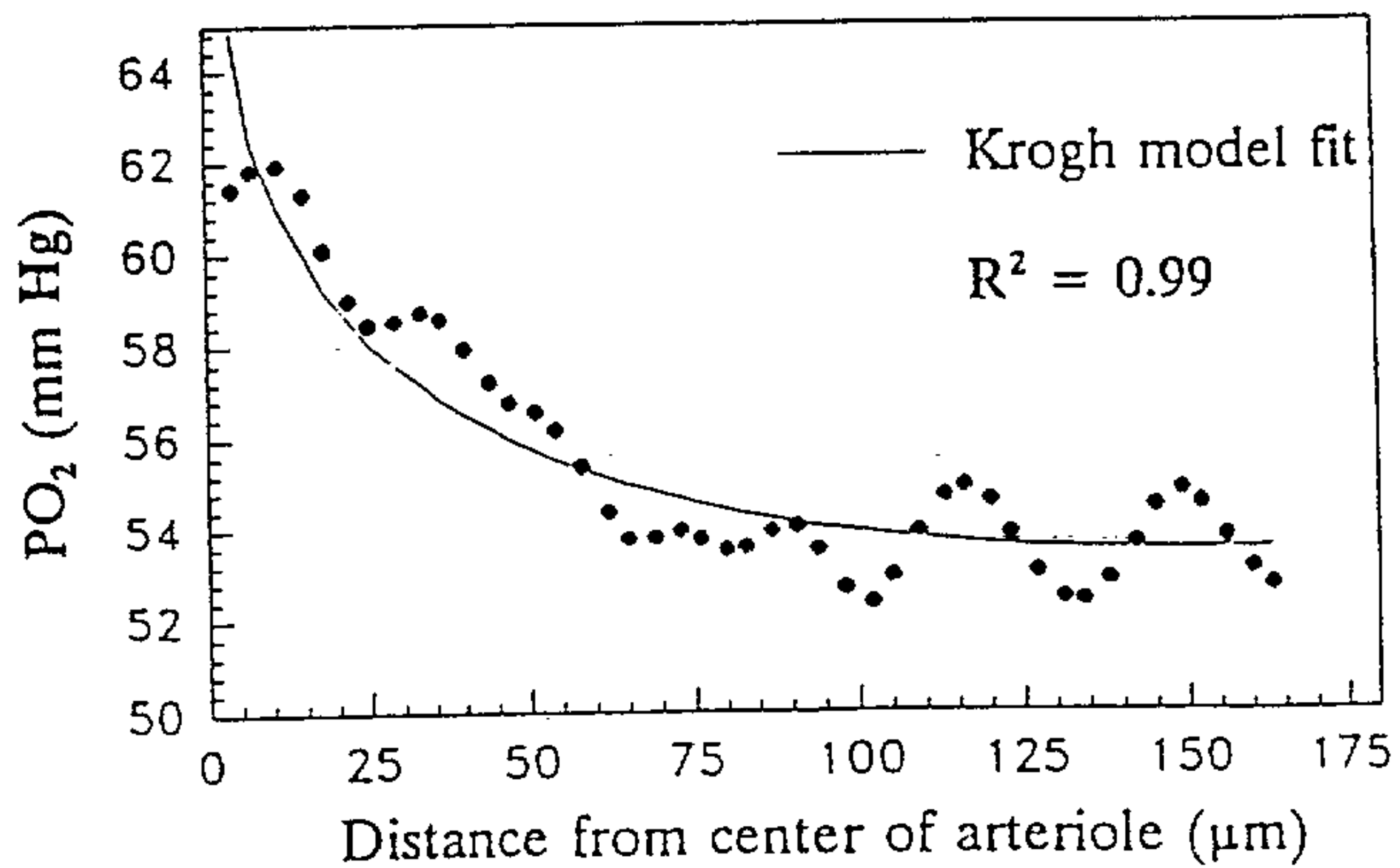


Figure 5B

