

Deep Learning for Computer Vision

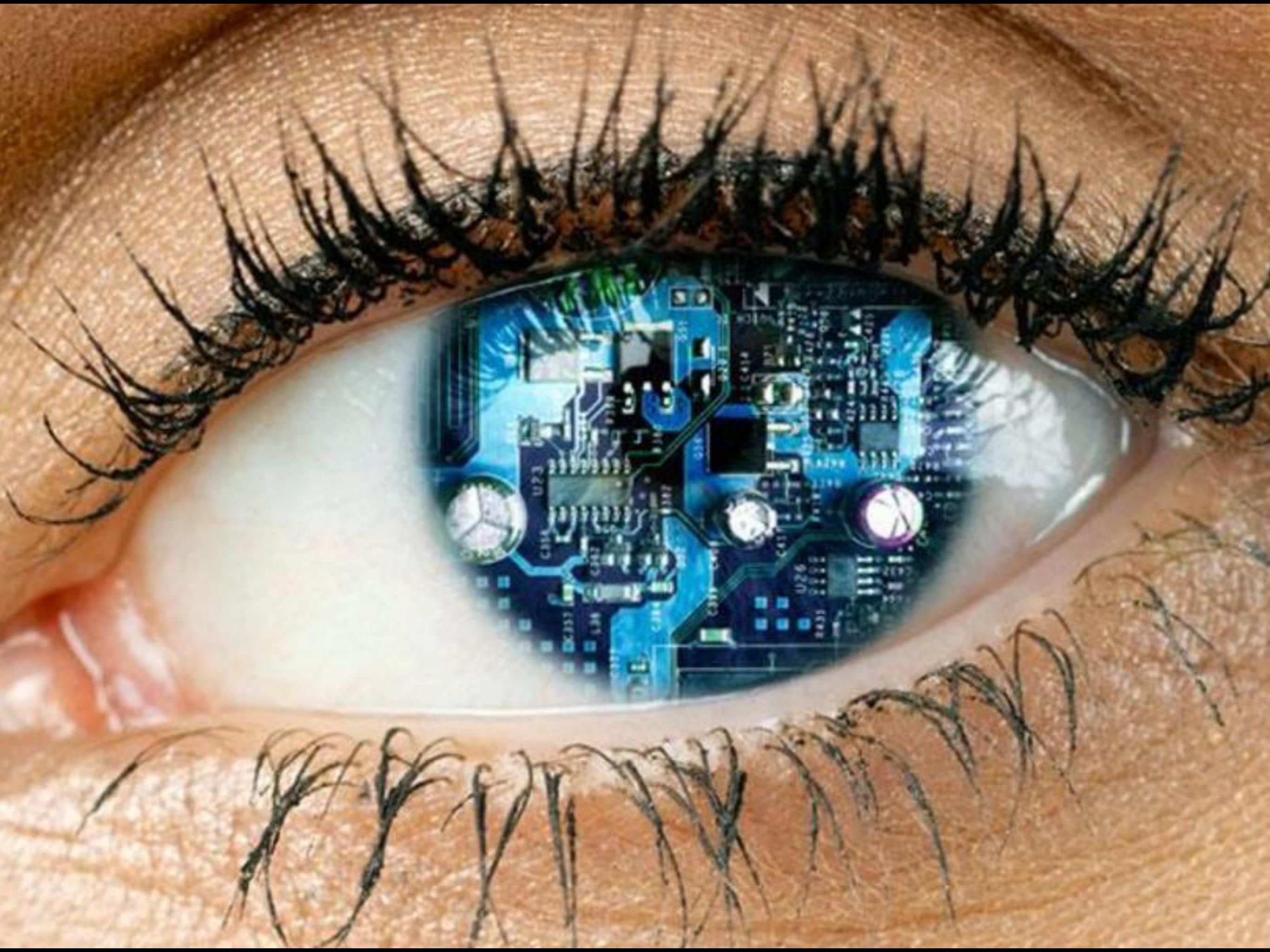
Lecture 1: An Introduction to Old-School Computer Vision

Peter Belhumeur

Computer Science
Columbia University

08	02	22	97	38	15	00	40	00	75	04	05	07	78	52	12	50	77	91	08
49	49	99	40	17	81	18	57	60	87	17	40	98	43	69	48	04	56	62	00
81	49	31	73	55	79	14	29	93	71	40	67	53	88	30	03	49	13	36	65
52	70	95	23	04	60	11	42	69	24	68	56	01	32	56	71	37	02	36	91
22	31	16	71	51	67	63	89	41	92	36	54	22	40	40	28	66	33	13	80
24	47	32	60	99	03	45	02	44	75	33	53	78	36	84	20	35	17	12	50
32	98	81	28	64	23	67	10	26	38	40	67	59	54	70	66	18	38	64	70
67	26	20	68	02	62	12	20	95	63	94	39	63	08	40	91	66	49	94	21
24	55	58	05	66	73	99	26	97	17	78	78	96	83	14	88	34	89	63	72
21	36	23	09	75	00	76	44	20	45	35	14	00	61	33	97	34	31	33	95
78	17	53	28	22	75	31	67	15	94	03	80	04	62	16	14	09	53	56	92
16	39	05	42	96	35	31	47	55	58	88	24	00	17	54	24	36	29	85	57
86	56	00	48	35	71	89	07	05	44	44	37	44	60	21	58	51	54	17	58
19	80	81	68	05	94	47	69	28	73	92	13	86	52	17	77	04	89	55	40
04	52	08	83	97	35	99	16	07	97	57	32	16	26	26	79	33	27	98	66
88	36	68	87	57	62	20	72	03	46	33	67	46	55	12	32	63	93	53	69
04	42	16	73	38	25	39	11	24	94	72	18	08	46	29	32	40	62	76	36
20	69	36	41	72	30	23	88	34	62	99	69	82	67	59	85	74	04	36	16
20	73	35	29	78	31	90	01	74	31	49	71	48	86	81	16	23	57	05	54
01	70	54	71	83	51	54	69	16	92	33	48	61	43	52	01	89	19	67	48

Find these pixels in
this image:



They aren't there.

08	02	22	97	38	15	00	40	00	75	04	05	07	78	52	12	50	77	91	08
49	49	99	40	17	81	18	57	60	87	17	40	98	43	69	48	04	56	62	00
81	49	31	73	55	79	14	29	93	71	40	67	53	88	30	03	49	13	36	65
52	70	95	23	04	60	11	42	69	24	68	56	01	32	56	71	37	02	36	91
22	31	16	71	51	67	63	89	41	92	36	54	22	40	40	28	66	33	13	80
24	47	32	60	99	03	45	02	44	75	33	53	78	36	84	20	35	17	12	50
32	98	81	28	64	23	67	10	26	38	40	67	59	54	70	66	18	38	64	70
67	26	20	68	02	62	12	20	95	63	94	39	63	08	40	91	66	49	94	21
24	55	58	05	66	73	99	26	97	17	78	78	96	83	14	88	34	89	63	72
21	36	23	09	75	00	76	44	20	45	35	14	00	61	33	97	34	31	33	95
78	17	53	28	22	75	31	67	15	94	03	80	04	62	16	14	09	53	56	92
16	39	05	42	96	35	31	47	55	58	88	24	00	17	54	24	36	29	85	57
86	56	00	48	35	71	89	07	05	44	44	37	44	60	21	58	51	54	17	58
19	80	81	68	05	94	47	69	28	73	92	13	86	52	17	77	04	89	55	40
04	52	08	83	97	35	99	16	07	97	57	32	16	26	26	79	33	27	98	66
88	36	68	87	57	62	20	72	03	46	33	67	46	55	12	32	63	93	53	69
04	42	16	73	38	25	39	11	24	94	72	18	08	46	29	32	40	62	76	36
20	69	36	41	72	30	23	88	34	62	99	69	82	67	59	85	74	04	36	16
20	73	35	29	78	31	90	01	74	31	49	71	48	86	81	16	23	57	05	54
01	70	54	71	83	51	54	69	16	92	33	48	61	43	52	01	89	19	67	48

When you strip away the
machinery of the human visual
system, you have to start at the
beginning...

What is vision?

10^{-12} meters

10^{-9}

10^{-6}

10^{-3}

10^0

10^3

1 nanometer

1000 nanometer

1 millimeter 1 meter

1 kilometer

Cosmic rays

X-rays

Gamma rays

Ultraviolet (UV)

Infrared (IR)

Radio

Broadcast band



Short Wavelengths

Long Wavelengths

Ultraviolet (UV)

Visible Light

Infrared (IR)

400 nanometers

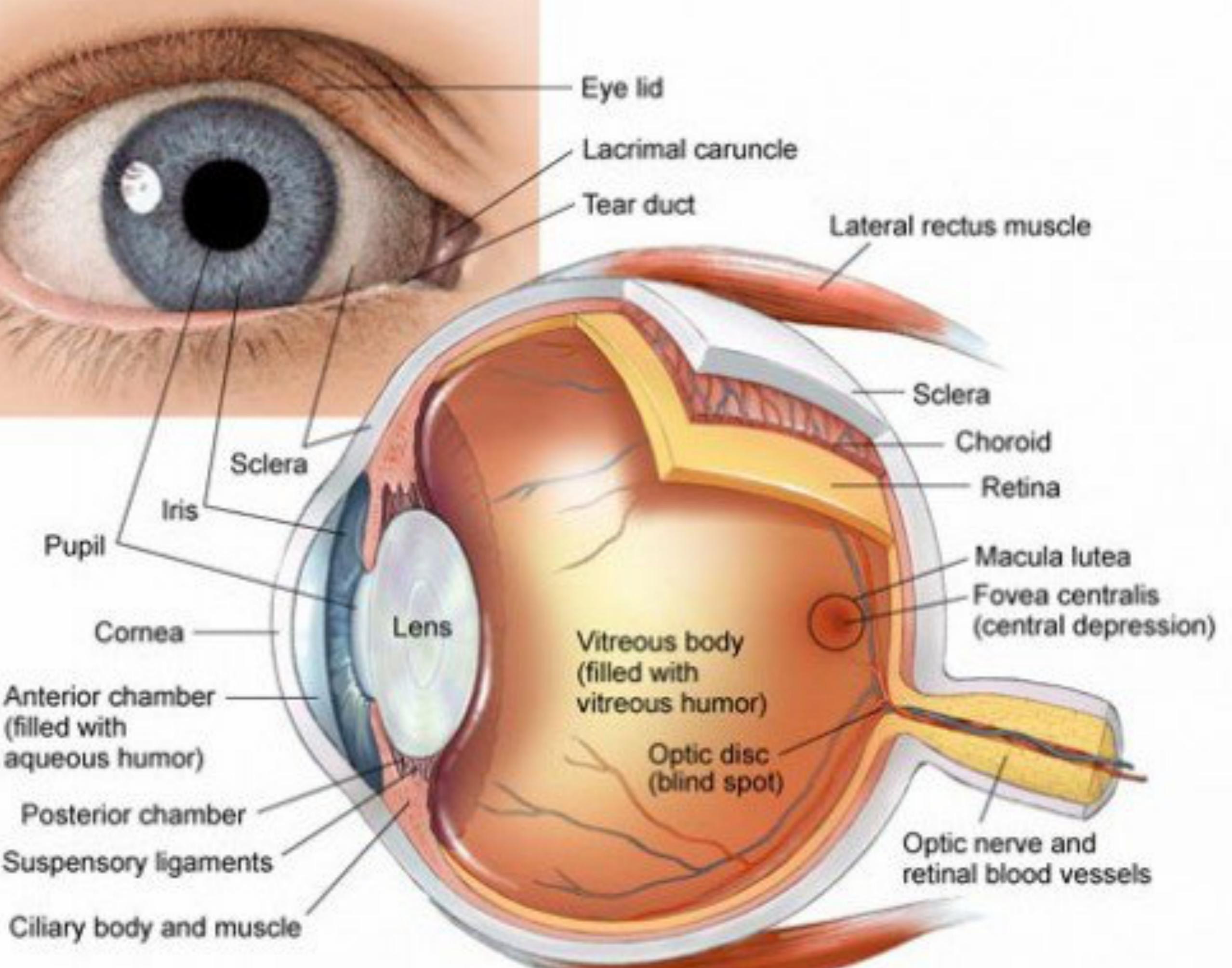
500 nanometers

600 nanometers

700 nanometers

Light

Visible light **travels in straight lines** and is not bent or diffracted by objects that we interact with like animals, buildings, furniture, etc.



Distant Object



50 Meters

Nearby Object



25 Centimeters

Cornea -

Changes
in Lens
Shape

Cornea -

Sclera

Optic
Nerve

Retina

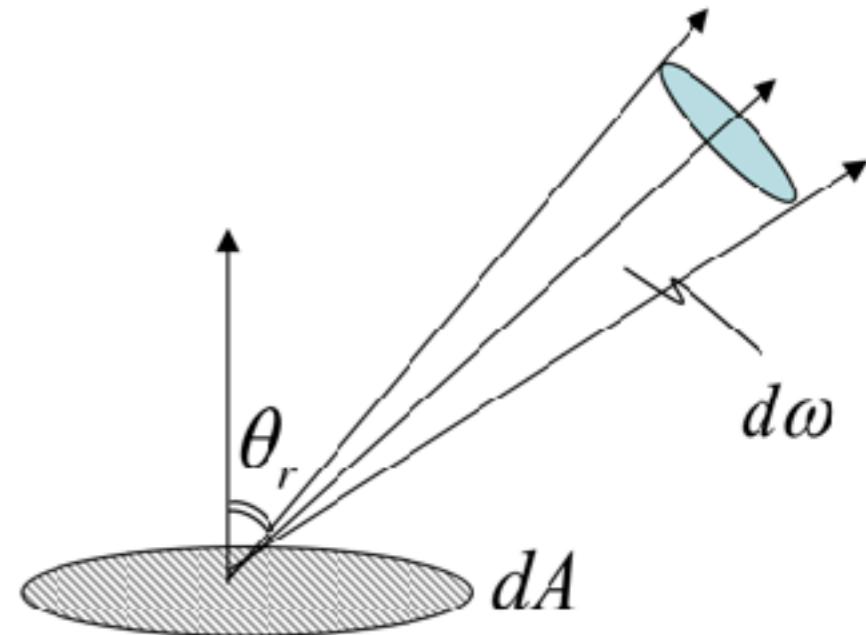
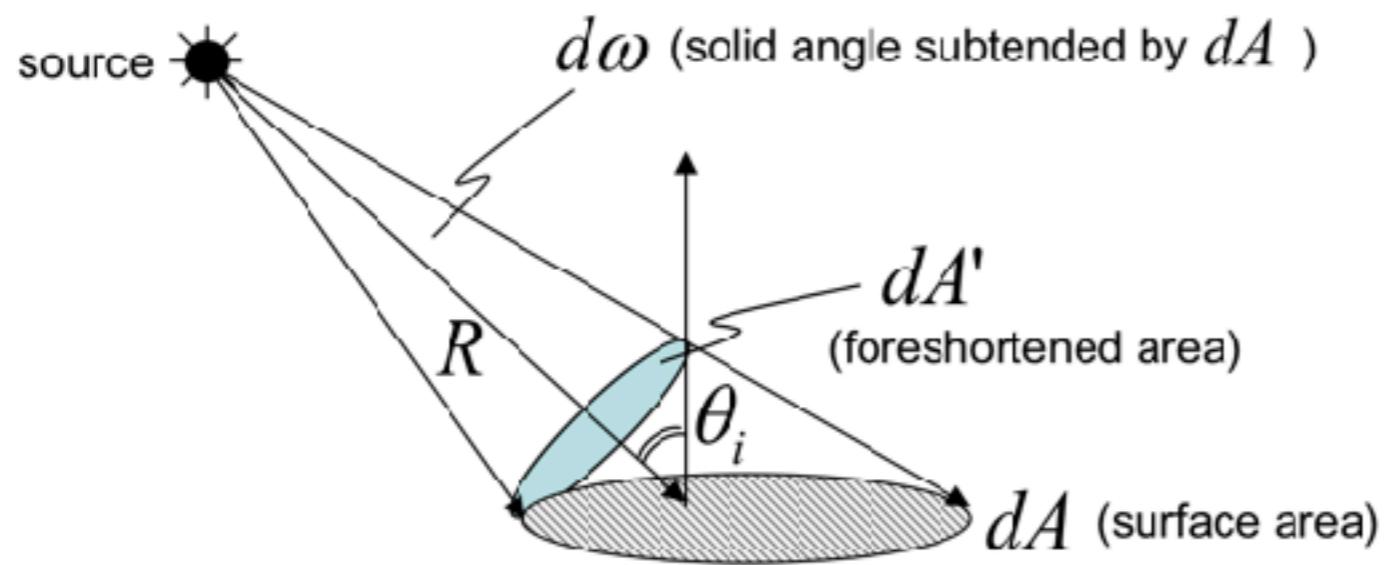
Vitreous
Humor

Optic
Nerve

So what can vision measure?

- **radiance** \approx how much light is reflected by the surface in a particular direction
- **radiance** = the radiant flux emitted by a *surface*, per unit solid angle per unit *projected area*

Radiometric Concepts



(1) Solid Angle : $d\omega = \frac{dA'}{R^2} = \frac{dA \cos \theta_i}{R^2}$ (steradian)

What is the solid angle subtended by a hemisphere?

(2) Radiant Intensity of Source : $J = \frac{d\Phi}{d\omega}$ (watts / steradian)

Light Flux (power) emitted per unit solid angle

(3) Surface Irradiance : $E = \frac{d\Phi}{dA}$ (watts / m²)

Light Flux (power) incident per unit surface area.

Does not depend on where the light is coming from!

(4) Surface Radiance (tricky) :

$$L = \frac{d\Phi}{(dA \cos \theta_r) d\omega}$$
 (watts / m² steradian)

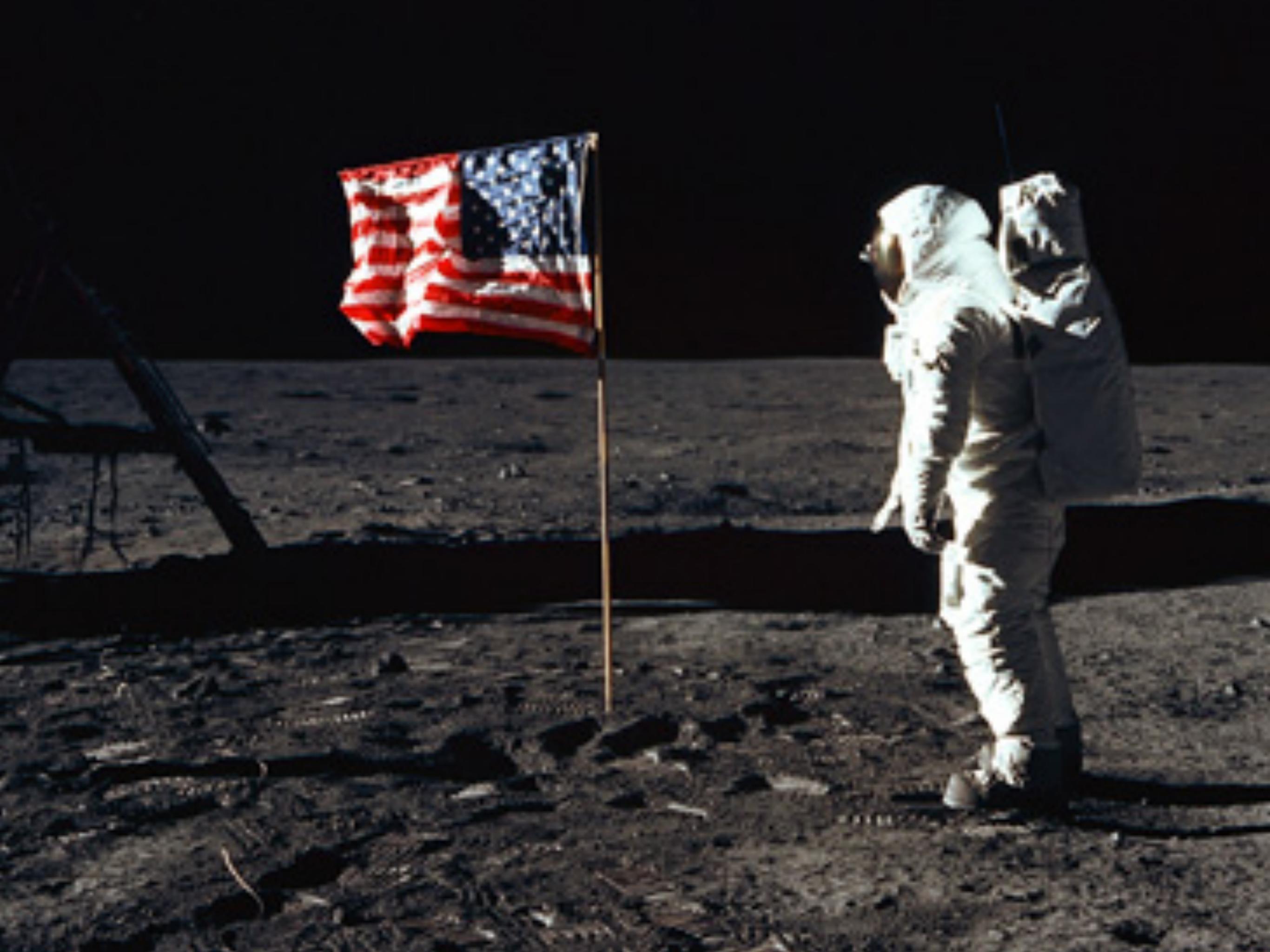
- Flux emitted per unit foreshortened area per unit solid angle.
- L depends on direction θ_r
- Surface can radiate into whole hemisphere.
- L depends on reflectance properties of surface.

So what can vision measure?

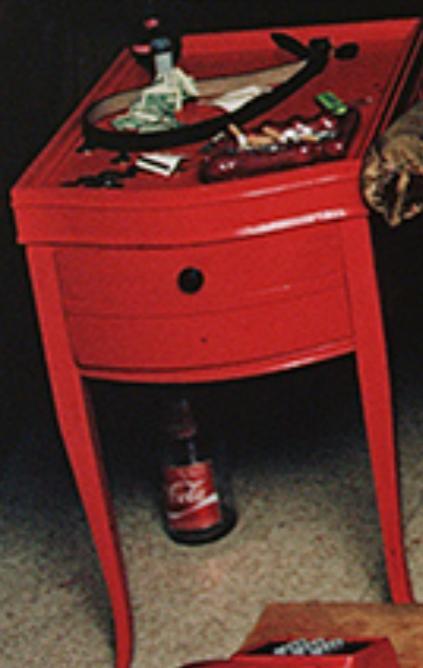
- so vision can measure the amount of light reflected back toward the eye by each point in the field of view as a function of time
- NOT size
- NOT weight
- NOT smell
- NOT temperature
- NOT chemical composition...

But from vision we can infer an awful lot
based on innate or learned experience...













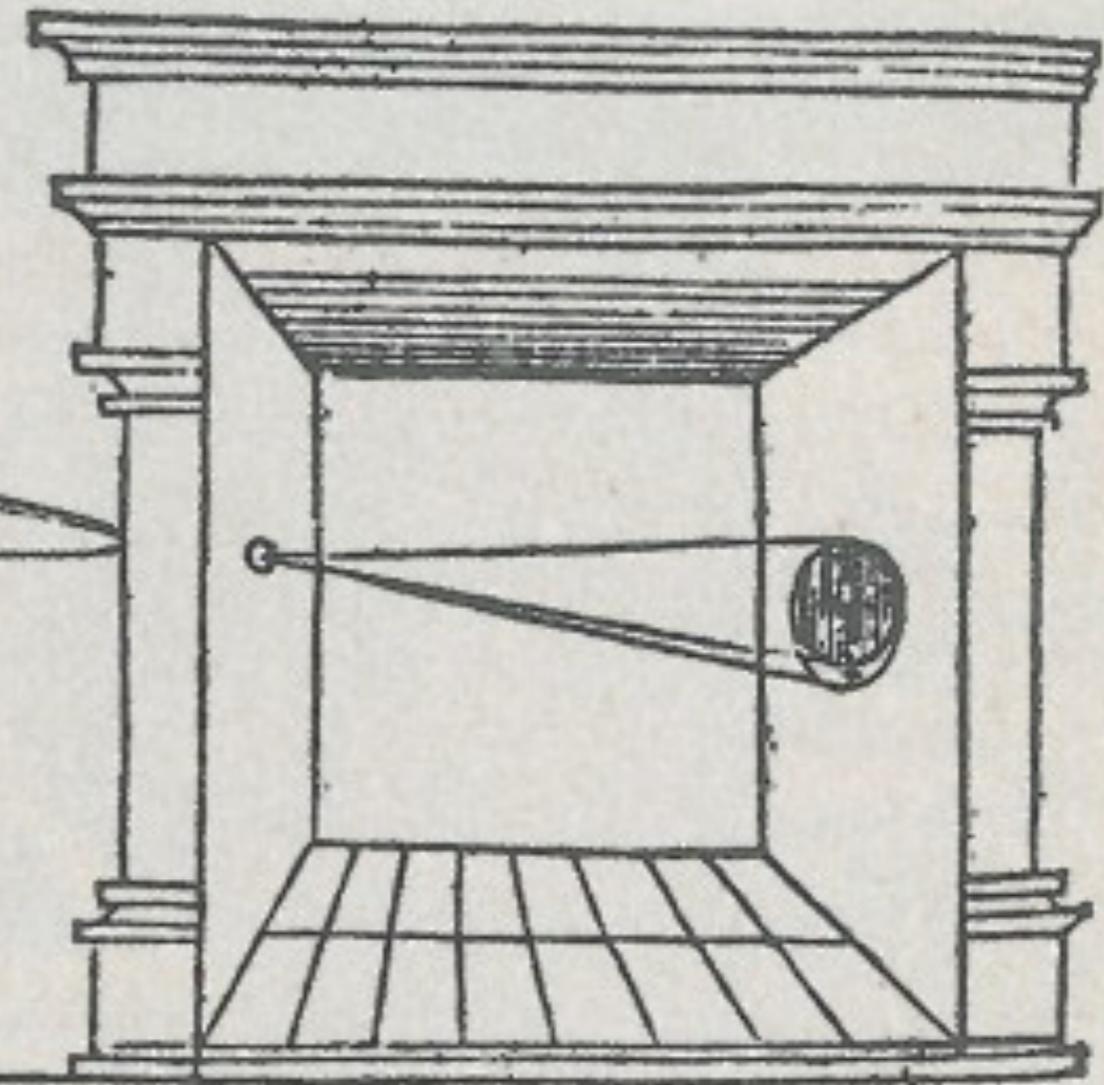
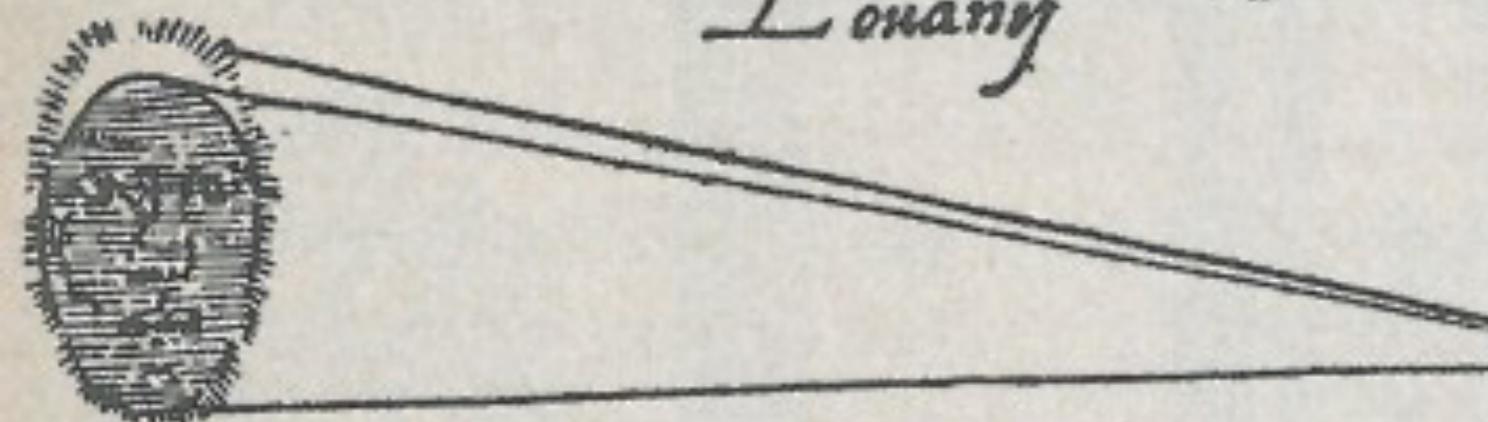
But what about computers?



Leonardo da Vinci's
Codex Atlanticus (1490)

Camera Obscura

*Solis deliquium Anno Christi
1544. Dic 24 Januarij
Louanijs*



Camera Obscura

Gemma Frisius' 1545 De Radio Astronomica and Geometrica

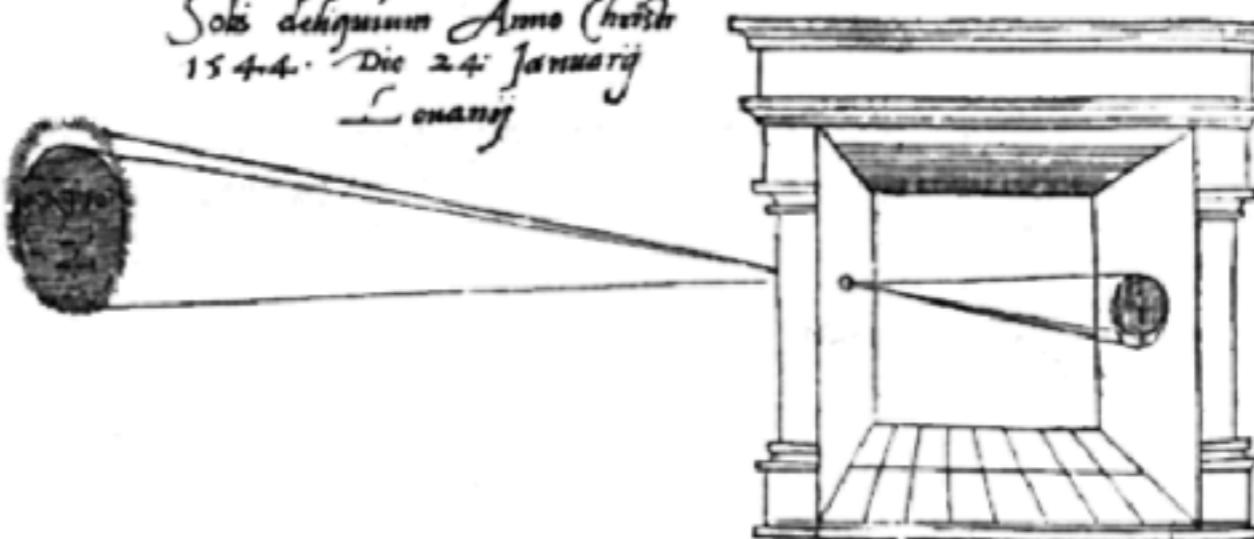


Camera Obscura (1490)

[Next ➔](#)

illum in tabula per radios Solis, quam in cœlo contin-
git: hoc est, si in cœlo superior pars deliquiū patiatur, in
radiis apparebit inferior deficere, ut ratio exigit optica.

*Soli deliquium Anno Christi
1544. Die 24 Januarij
Louvain*



*Sic nos exacte Anno .1544. Louvaini eclipsim Solis
obseruauimus, inuenimusq; deficere paulo plus q̄ dex-*

A camera obscura is a dark box with a small hole in the front. Light passes through this hole and lands on the back of the box, casting an upside-down version of the scene. Although this idea had been understood by scholars for hundreds of years, Leonardo da Vinci (1452-1519) was the first to write a detailed description of the camera obscura in his *Atlantic Codex* in 1490. An astronomer named Gemma Frisius (1508-1555) was the first to draw a camera obscura around 1545.

**RECEPTIVE FIELDS, BINOCULAR INTERACTION
AND FUNCTIONAL ARCHITECTURE IN
THE CAT'S VISUAL CORTEX**

By D. H. HUBEL AND T. N. WIESEL

*From the Neurophysiology Laboratory, Department of Pharmacology
Harvard Medical School, Boston, Massachusetts, U.S.A.*

(Received 31 July 1961)

What chiefly distinguishes cerebral cortex from other parts of the central nervous system is the great diversity of its cell types and interconnexions. It would be astonishing if such a structure did not profoundly modify the response patterns of fibres coming into it. In the cat's visual cortex, the receptive field arrangements of single cells suggest that there is indeed a degree of complexity far exceeding anything yet seen at lower levels in the visual system.

In a previous paper we described receptive fields of single cortical cells, observing responses to spots of light shone on one or both retinas (Hubel & Wiesel, 1959). In the present work this method is used to examine receptive fields of a more complex type (Part I) and to make additional observations on binocular interaction (Part II).

This approach is necessary in order to understand the behaviour of individual cells, but it fails to deal with the problem of the relationship of one cell to its neighbours. In the past, the technique of recording evoked slow waves has been used with great success in studies of functional anatomy. It was employed by Talbot & Marshall (1941) and by Thompson, Woolsey & Talbot (1950) for mapping out the visual cortex in the rabbit, cat, and monkey. Daniel & Whitteridge (1959) have recently extended this work in the primate. Most of our present knowledge of retinotopic projections, binocular overlap, and the second visual area is based on these investigations. Yet the method of evoked potentials is valuable mainly for detecting behaviour common to large populations of neighbouring cells; it cannot differentiate functionally between areas of cortex smaller than about 1 mm^2 . To overcome this difficulty a method has in recent years been developed for studying cells separately or in small groups during long micro-electrode penetrations through nervous tissue. Responses are correlated with cell location by reconstructing the electrode tracks from histological material. These techniques have been applied to

Hubel and Wiesel's 1962
single cell recordings in
the cat's visual cortex

Visual Cortex

Mapping receptive fields

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
PROJECT MAC

Artificial Intelligence Group
Vision Memo. No. 100.

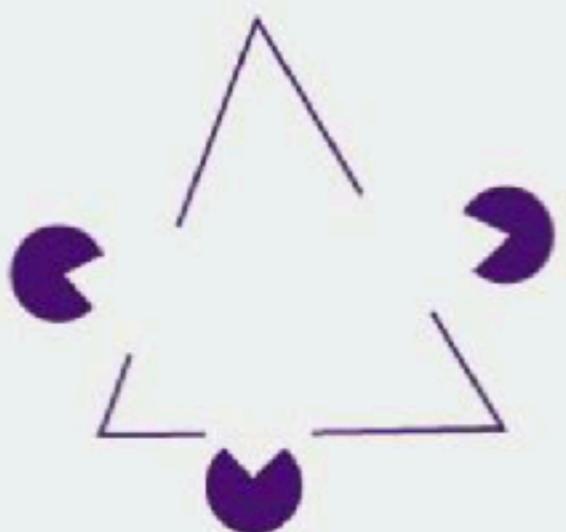
July 7, 1966

THE SUMMER VISION PROJECT

Seymour Papert

The summer vision project is an attempt to use our summer workers effectively in the construction of a significant part of a visual system. The particular task was chosen partly because it can be segmented into sub-problems which will allow individuals to work independently and yet participate in the construction of a system complex enough to be a real landmark in the development of "pattern recognition".

VISION



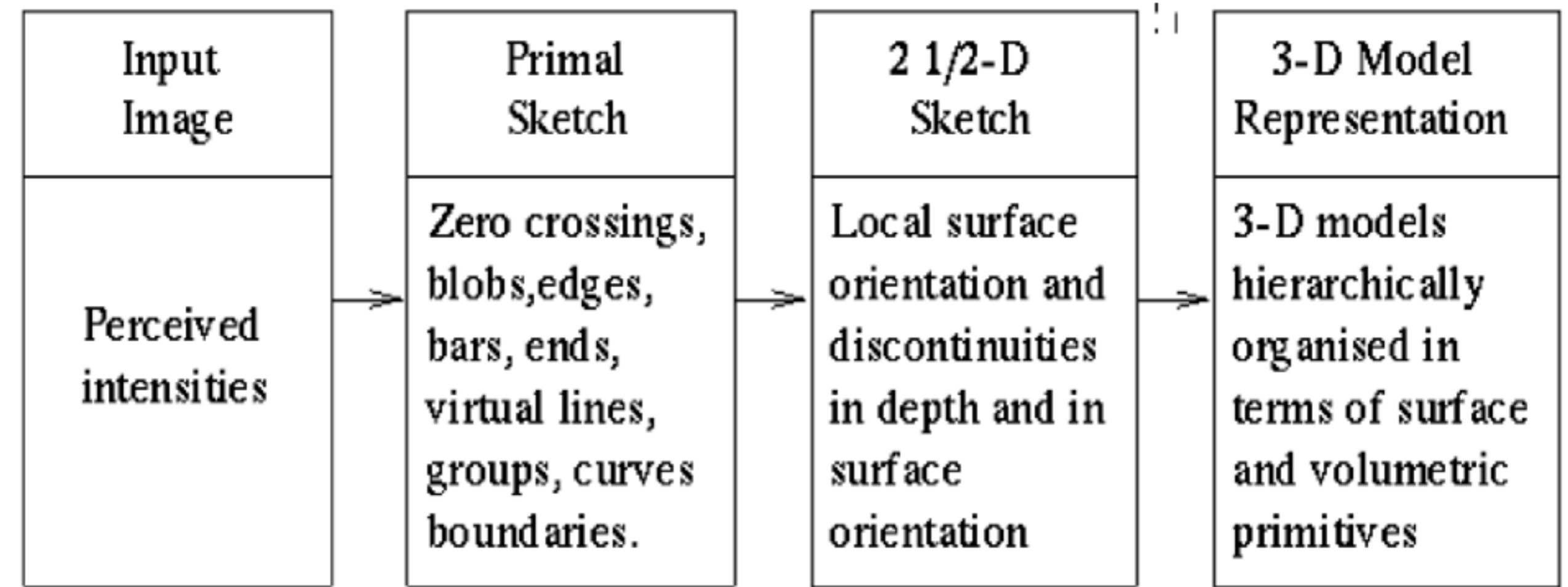
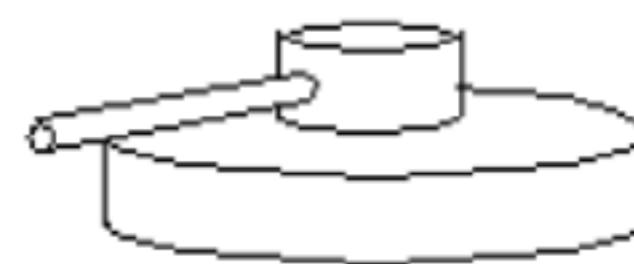
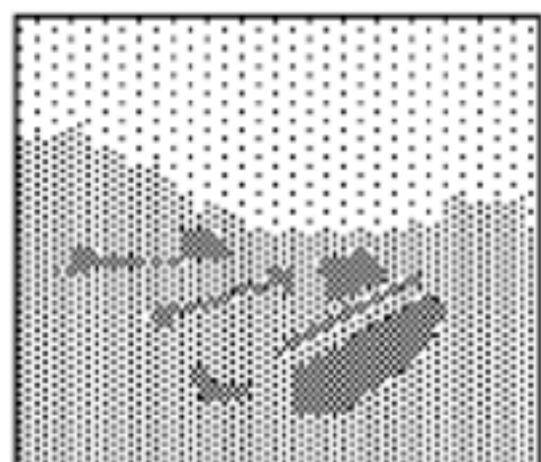
David Marr

FOREWORD BY
Shimon Ullman

AFTERWORD BY
Tomaso Poggio

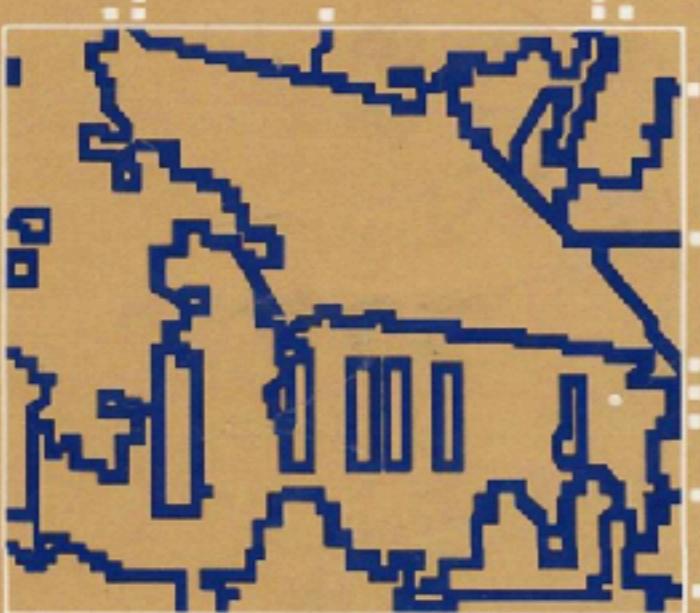
Vision from a 1970s
neuroscience perspective

input image edge image 2¹/₂-D sketch 3-D model



COMPUTER VISION

DANA H. BALLARD • CHRISTOPHER M. BROWN



Vision from a 1982
computer science perspective

(3.57) as closely as possible and also is locally smooth [Horn and Schunck 1980]. In this case as well, the Laplacians of the two velocity components, $\nabla^2 u$ and $\nabla^2 v$, can measure local smoothness.

Again using the method of Lagrange multipliers, minimize the flow error

$$E^2(x, y) = (f_x u + f_y v + f_t)^2 + \lambda^2[(\nabla^2 u)^2 + (\nabla^2 v)^2] \quad (3.58)$$

Differentiating this equation with respect to u and v provides equations for the change in error with respect to u and v , which must be zero for a minimum. Writing $\nabla^2 u$ as $u - u_{av}$ and $\nabla^2 v$ as $v - v_{av}$, these equations are

$$(\lambda^2 + f_x^2)u + f_x f_y v = \lambda^2 u_{av} - f_x f_t \quad (3.59)$$

$$f_x f_y u + (\lambda^2 + f_y^2)v = \lambda^2 v_{av} - f_y f_t \quad (3.60)$$

These equations may be solved for u and v , yielding

$$u = u_{av} - f_x \frac{P}{D} \quad (3.61)$$

$$v = v_{av} - f_y \frac{P}{D} \quad (3.62)$$

where

$$P = f_x u_{av} + f_y v_{av} + f_t$$

$$D = \lambda^2 + f_x^2 + f_y^2$$

To turn this into an iterative equation for solving $u(x, y)$ and $v(x, y)$, again use the Gauss-Seidel method.

Algorithm 3.4: Optical Flow [Horn and Schunck 1980].

$k = 0$.

Initialize all u^k and v^k to zero.

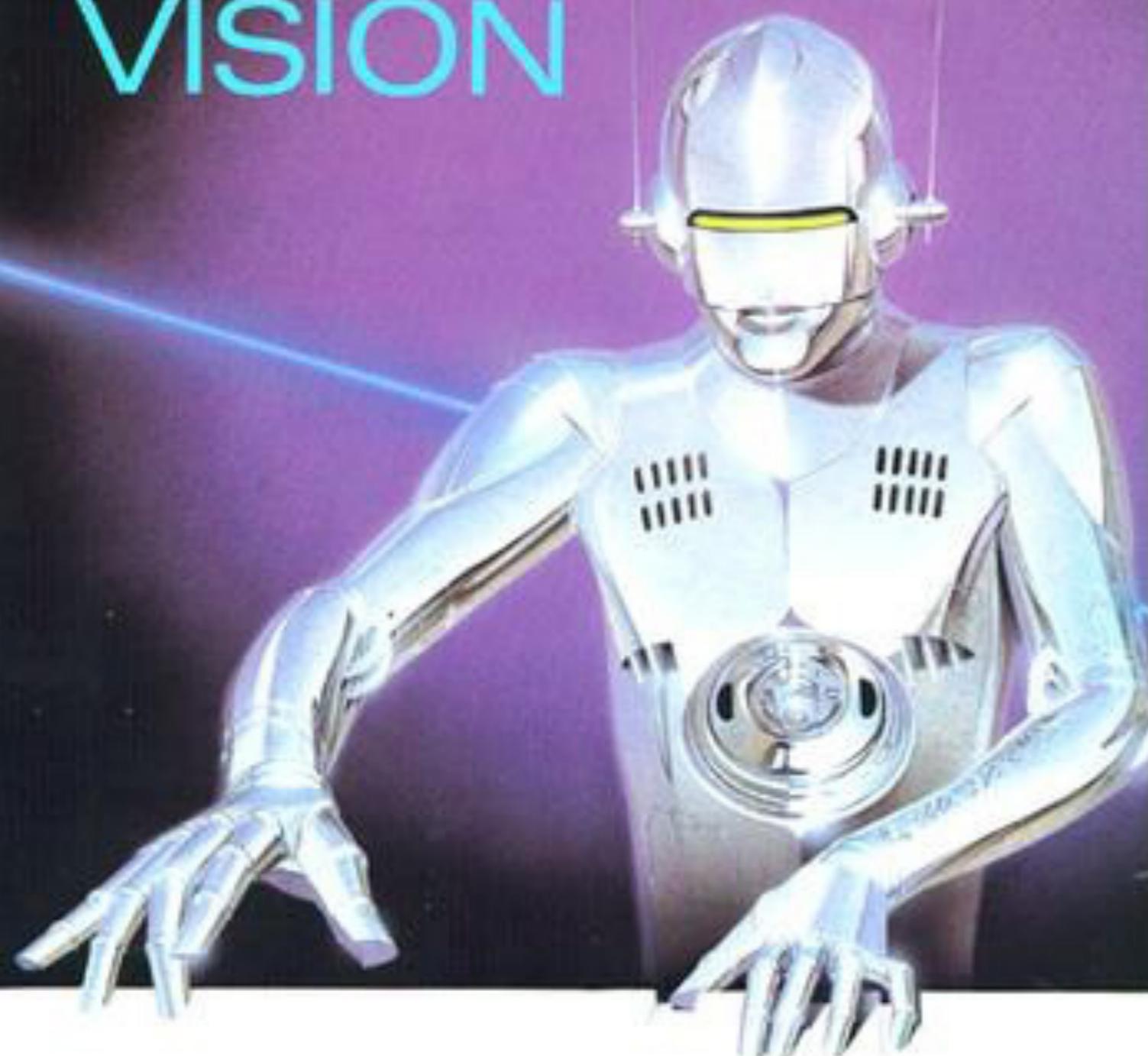
Until some error measure is satisfied, do

$$u^k = u_{av}^{k-1} - f_x \frac{P}{D}$$

$$v^k = v_{av}^{k-1} - f_y \frac{P}{D}$$

As Horn and Schunck demonstrate, this method derives the flow for two time frames, but it can be improved by using several time frames and using the final solution after one iteration at one time for the initial solution at the following time frame. That is:

ROBOT VISION



Vision from a 1986
electrical eng. perspective

for some m . This is the *Poisson* distribution. We can calculate the average number liberated in time T as follows:

$$\sum_{n=1}^{\infty} ne^{-m} \frac{m^n}{n!} = m e^{-m} \sum_{n=1}^{\infty} \frac{m^{n-1}}{(n-1)!}.$$

But

$$\sum_{n=1}^{\infty} \frac{m^{n-1}}{(n-1)!} = \sum_{n=0}^{\infty} \frac{m^n}{n!} = e^m,$$

so the average is just m . We show in exercise 2-18 that the variance is also m . The standard deviation is thus \sqrt{m} , so that the ratio of the standard deviation to the mean is $1/\sqrt{m}$. The measurement becomes more accurate the longer we wait, since more electrons are gathered. Again, the ratio of the "signal" to the "noise" only improves as the square root of the average number of electrons collected, however.

To obtain reasonable results, many electrons must be measured. It can be shown that a Poisson distribution with mean m is almost the same as a Gaussian distribution with mean m and variance m , provided that m is large. The Gaussian distribution is often easier to work with. In any case, to obtain a standard deviation that is one-thousandth of the mean, one must wait long enough to collect a million electrons. This is a small charge still, since one electron carries only

$$e = 1.602192\ldots \times 10^{-19} \text{ Coulomb.}$$

Even a million electrons have a charge of only about 160 fC (femto-Coulomb). (The prefix *femto-* denotes a multiplier of 10^{-15} .) It is not easy to measure such a small charge, since noise is introduced in the measurement process.

The number of electrons liberated from an area δA in time δt is

$$N = \delta A \delta t \int_{-\infty}^{\infty} b(\lambda) q(\lambda) d\lambda,$$

where $q(\lambda)$ is the quantum efficiency and $b(\lambda)$ is the image irradiance in photons per unit area. To obtain a usable result, then, electrons must be collected from a finite image area over a finite amount of time. There is thus a trade-off between (spatial and temporal) resolution and accuracy.

A measurement of the number of electrons liberated in a small area during a fixed time interval produces a result that is proportional to the irradiance (for fixed spectral distribution of incident photons). These measurements are quantized in order to read them into a digital computer.

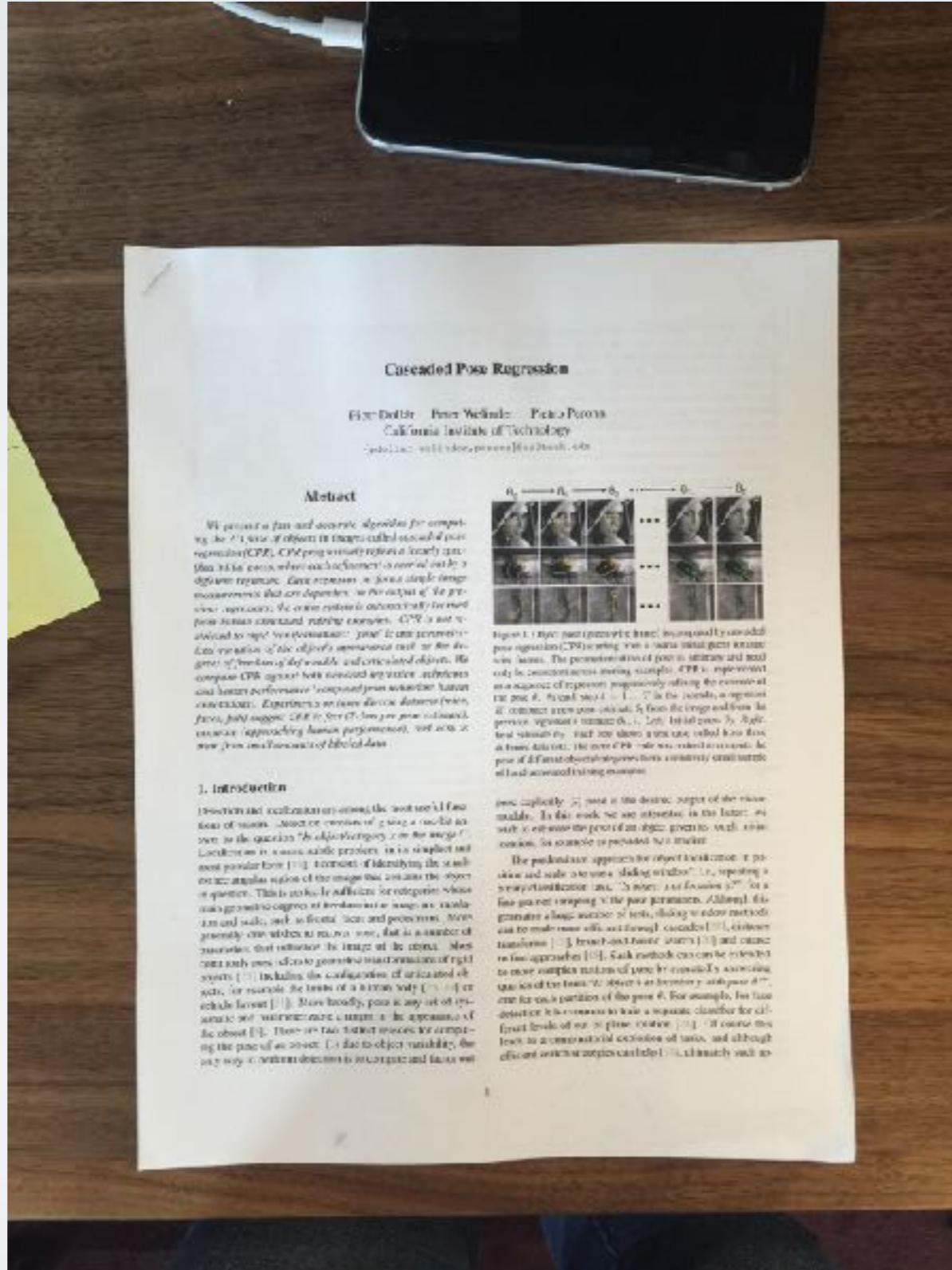
Computer Vision Algorithm: Document Scanner

DOC SCANNER

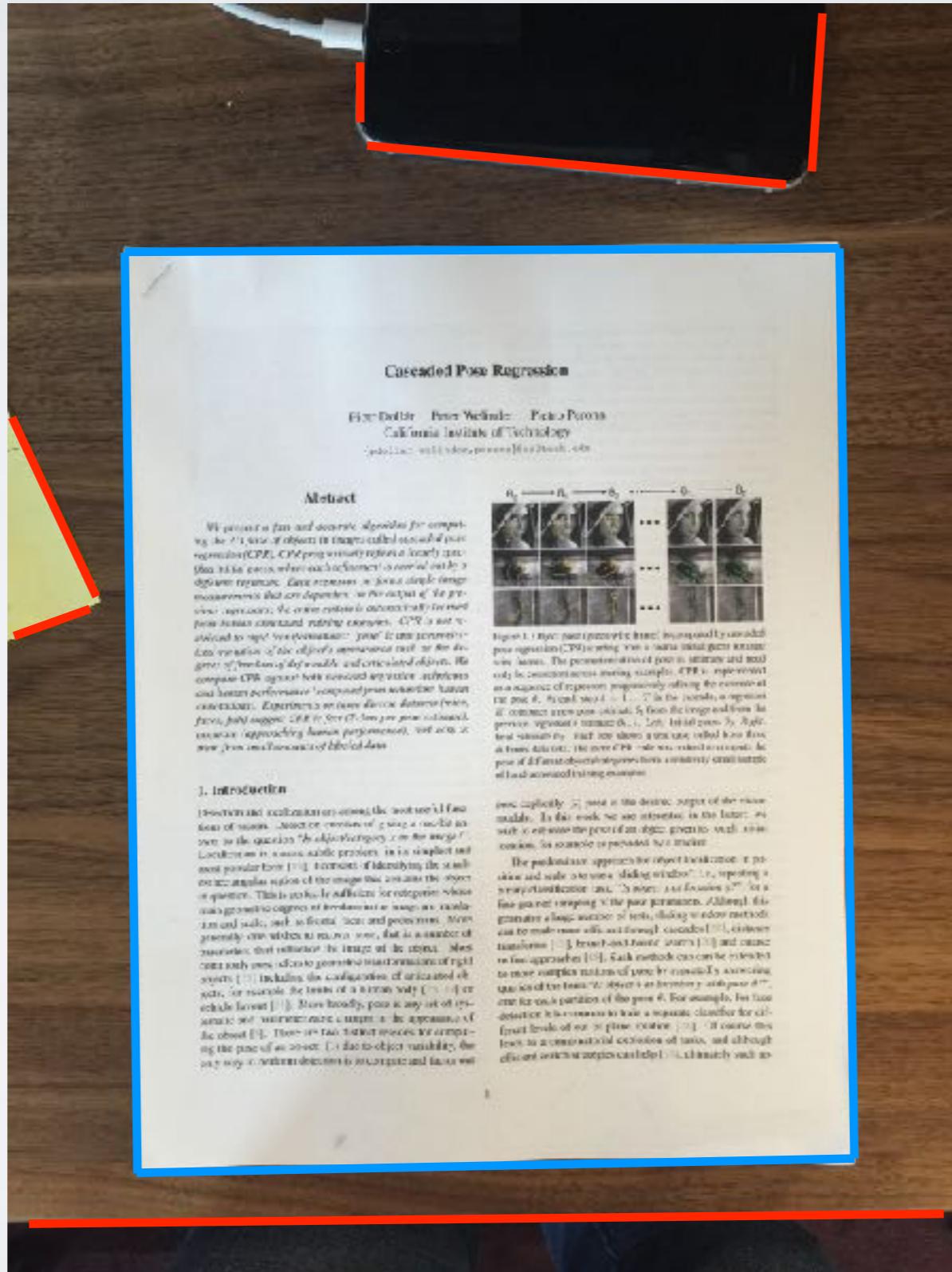
ML TEAM @ DROPBOX

[Dropbox Tech Blog](#)

DOCUMENT DETECTION



DOCUMENT DETECTION



Cascaded Pose Regression

Han-Dol Oh¹ · Peter Welinder² · Pedro Perona¹
California Institute of Technology
<http://vision.csail.mit.edu/paper/CPR.pdf>

Abstract

We present a fast and accurate algorithm for computing the 2D pose of objects in images called cascaded pose regression (CPR). CPR progressively refines a loosely specified initial guess where each refinement is computed on a digitized template. Each iteration is based on simple image measurements that are dependent on the output of the previous iteration; the entire process is automatically learned from human annotated training examples. CPR is not restricted to rigid objects, and it can handle both the orientation and location of the object's unimposed rest or the degrees of freedom of deformable and articulated objects. We compare CPR against both cascaded regression methods and human performance (computed from subjective human evaluations). Experiments on three diverse datasets (books, faces, and chairs) suggest CPR is 5x faster than prior state-of-the-art, while approaching human performance, and scales linearly with the number of labeled data.

1. Introduction

Detection and localization are among the most used functions of vision. Most often, one is trying to find an object in the question “Is the object category *x* in the image?” Localization is a common subtask problem, it is simplest and most popular form [1–4]. It consists of identifying the subregion in an image that contains the object in question. This is not always sufficient for recognition, whose main goal is to categorize local features in images, such as color, texture and scale, and to identify them and process them. More generally, one wishes to know more, but it is a number of annotations that enhance the image of the image. These can only now allow to generate information of rigid objects [5–7] including the configuration of articulated objects, for example the joints of a human body [5–7] or robotic layout [8–10]. More broadly, pose is any set of geometric and kinematic constraints that attempt to fit appearance of the object [11]. These are two further issues we are computing the pose of an object. One due to object variability, the other way to handle detection is to compute and have well-

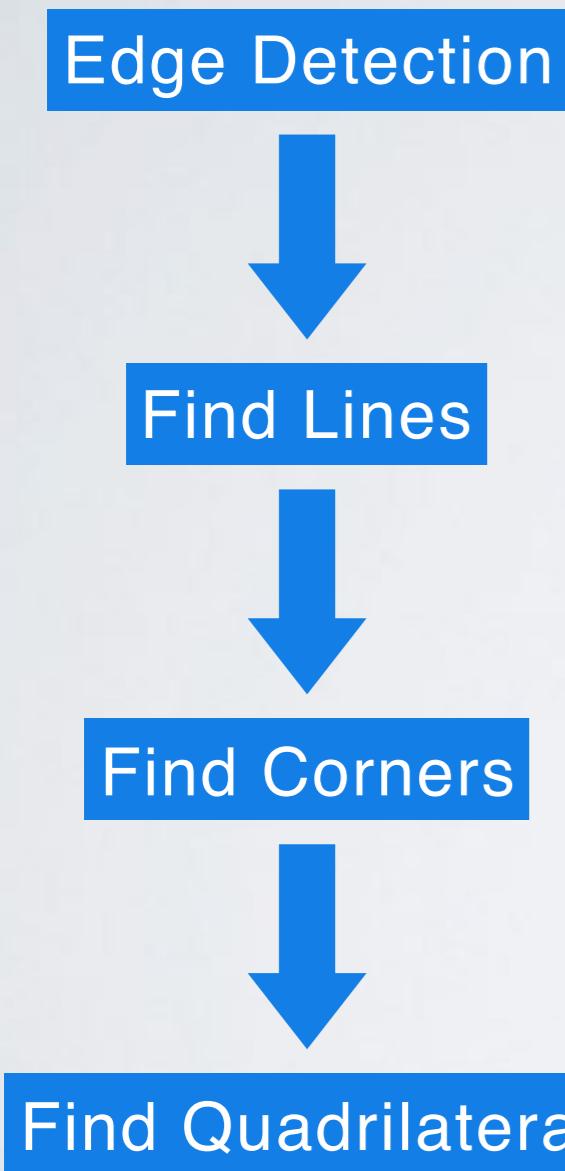


Figure 1: Our pose regression framework sequentially cascades pose regression (CPR) starting from a coarse initial guess (bottom row). The performance ratio of pose in accuracy and recall only by concatenating training samples. CPR is implemented as a sequence of regressions progressively refining the estimated pose θ_1 , θ_2 and θ_3 ($\theta = 1 \dots 7$). In the middle, a sequence of concatenated training samples. From the original pose θ_0 (top row), the previous regression estimates $\theta_1, \dots, \theta_6$. In addition, the next CPR module will be initialized with the last pose of all former observations plus some constantly small initial of local annotated training examples.

pose explicitly. θ_1 pose is the desired output of the main module. In this work we are interested in the latter, as such to estimate the pose of an object given its visual representation, for example represented by a model.

The predominant approach for object localization, or position and scale detection, is sliding window, i.e., repeating a search-and-detect-and-classify loop. It is a fine grained mapping of the pose parameters. Although this generates a large number of tests, sliding window methods can be made more efficient through cascaded [12], coarse-to-fine [13], branch-and-bound [14] and coarse-to-fine approaches [15]. Such methods can be extended to more complex notions of pose by gradually increasing quality of the function ψ : object's appearance and pose θ^* , and for each portion of the pose θ . For example, for hand detection it is common to train a separate classifier for different levels of our 3D pose feature [16]. Of course this leads to a computational explosion of tasks, and although efficient optimization techniques can help [17], ultimately such ap-

STEP BY STEP



STEP BY STEP

Edge Detection



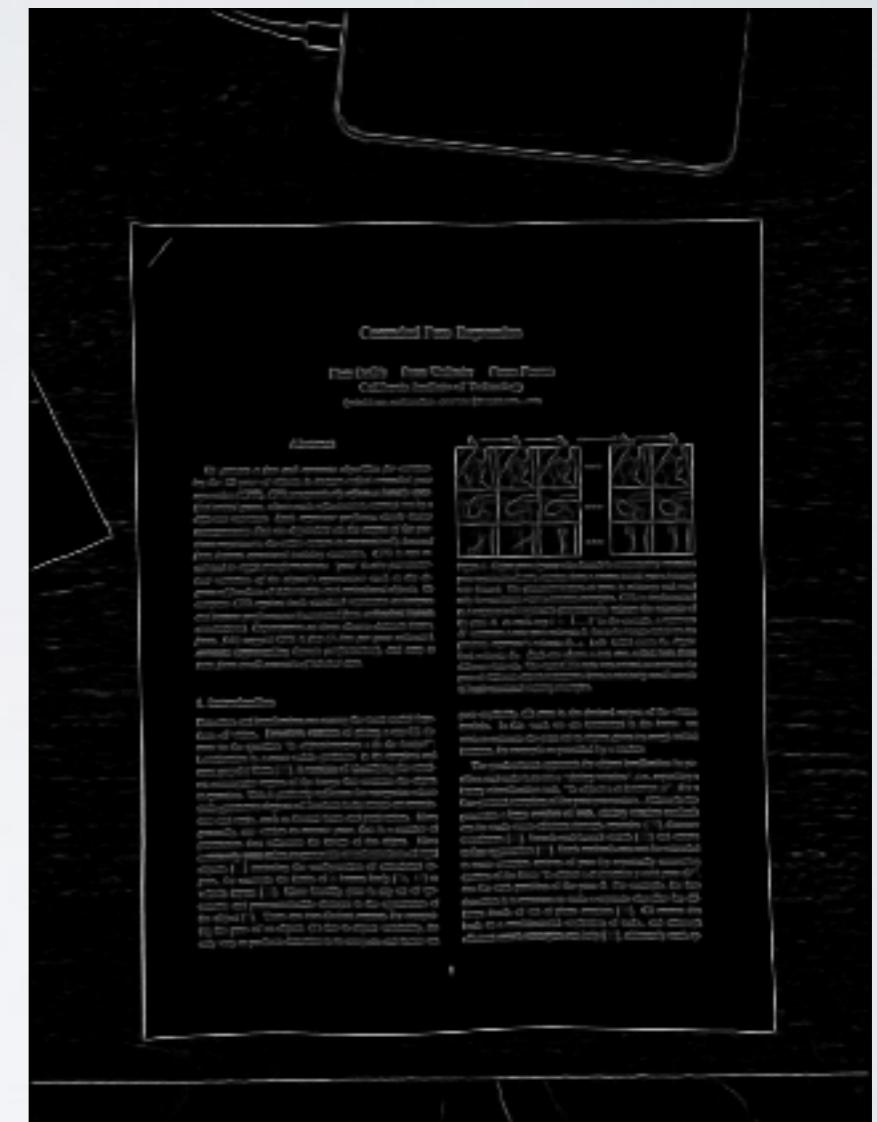
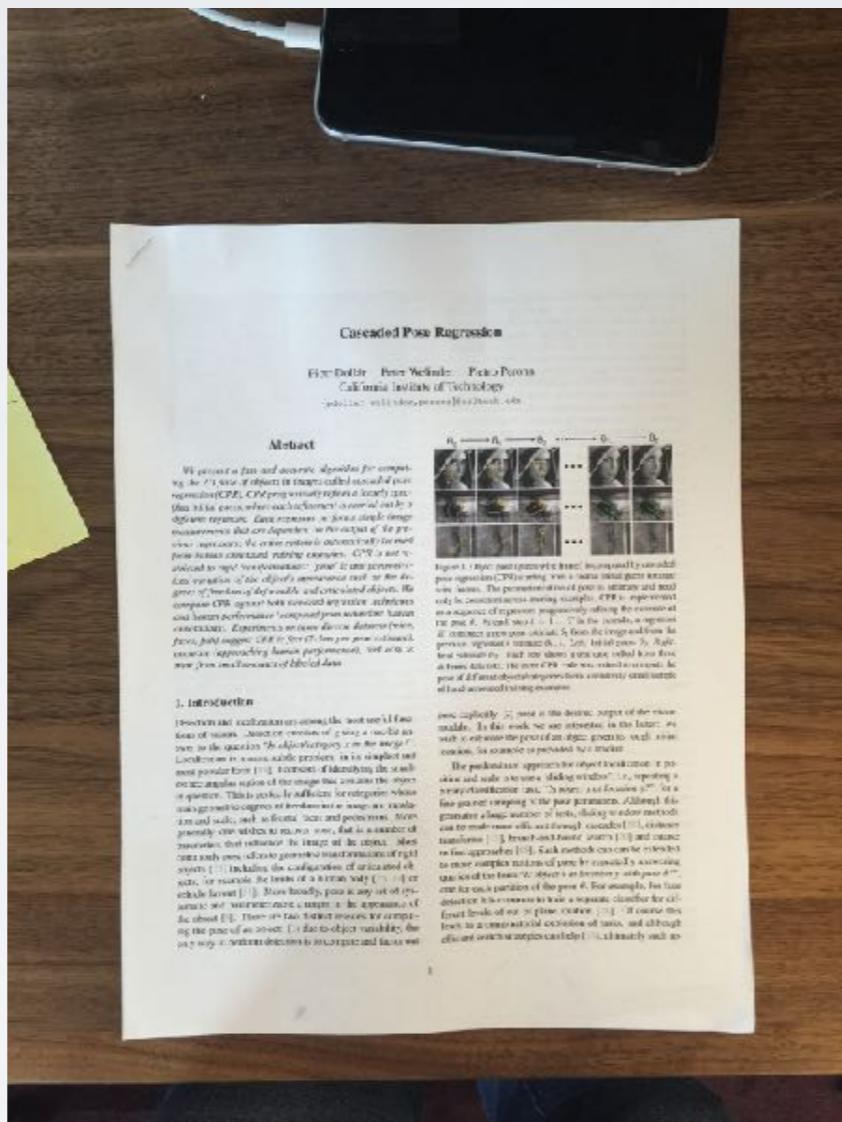
Find Lines



Find Corners



Find Quadrilateral



ML-Based Detector

STEP BY STEP

Edge Detection



Find Lines



Find Corners



Find Quadrilateral



STEP BY STEP

Edge Detection



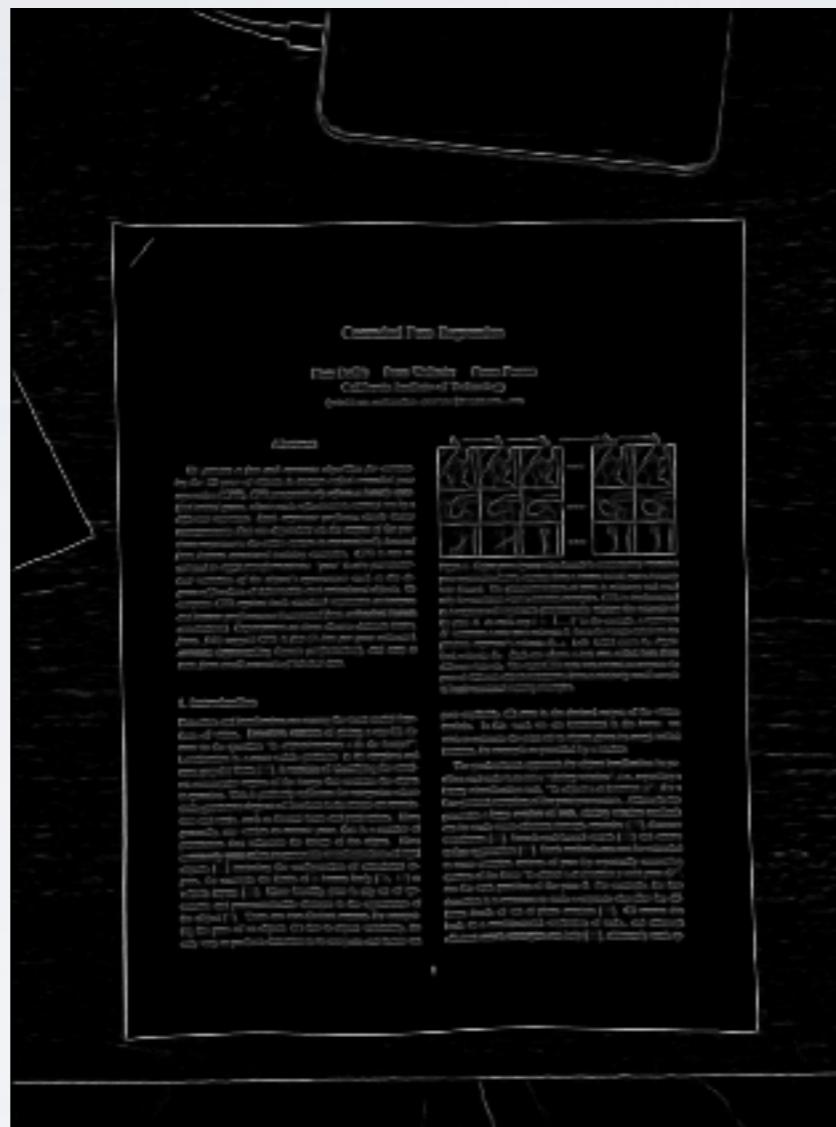
Find Lines



Find Corners



Find Quadrilateral



STEP BY STEP

Edge Detection



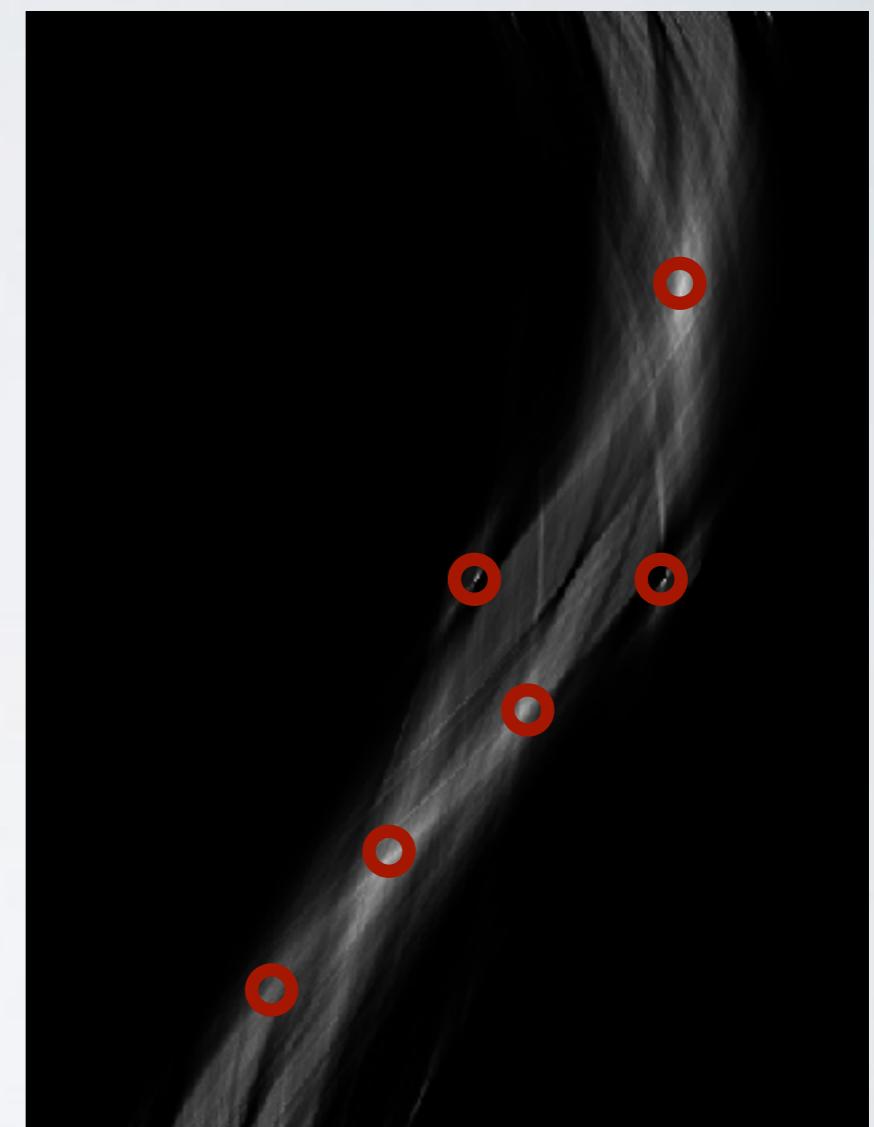
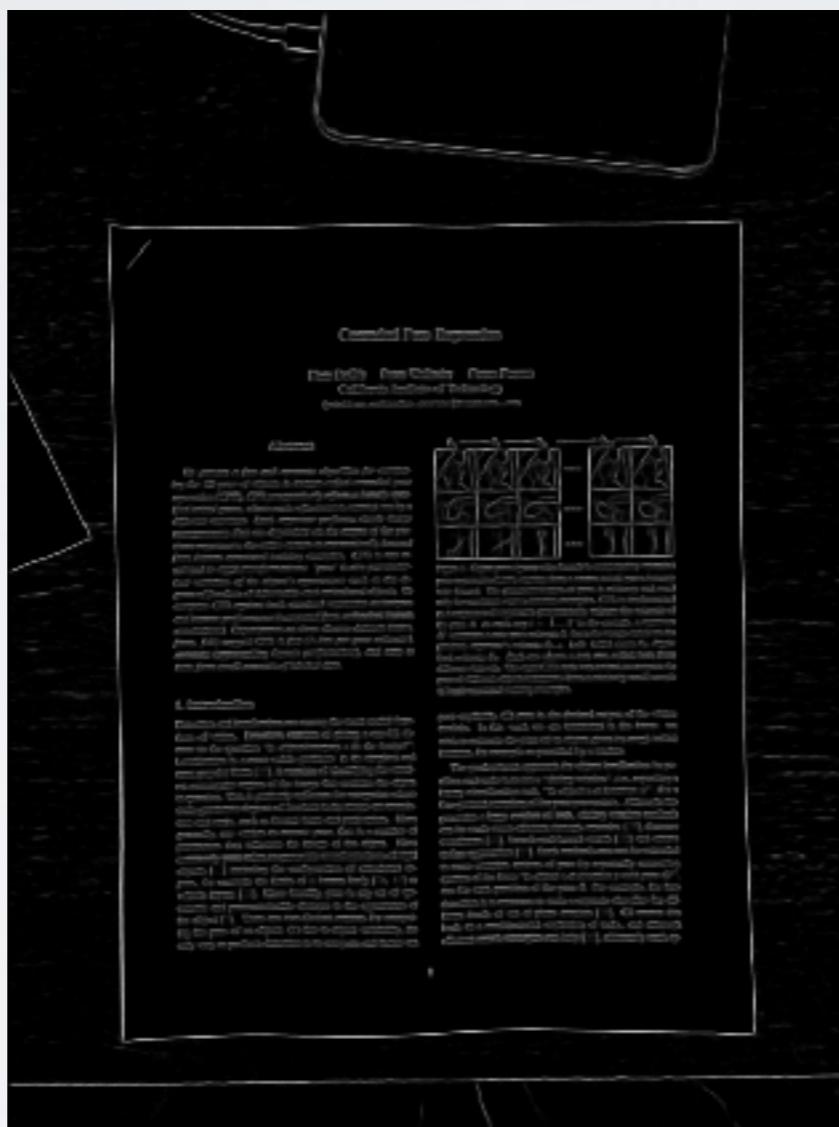
Find Lines



Find Corners



Find Quadrilateral



STEP BY STEP

Edge Detection



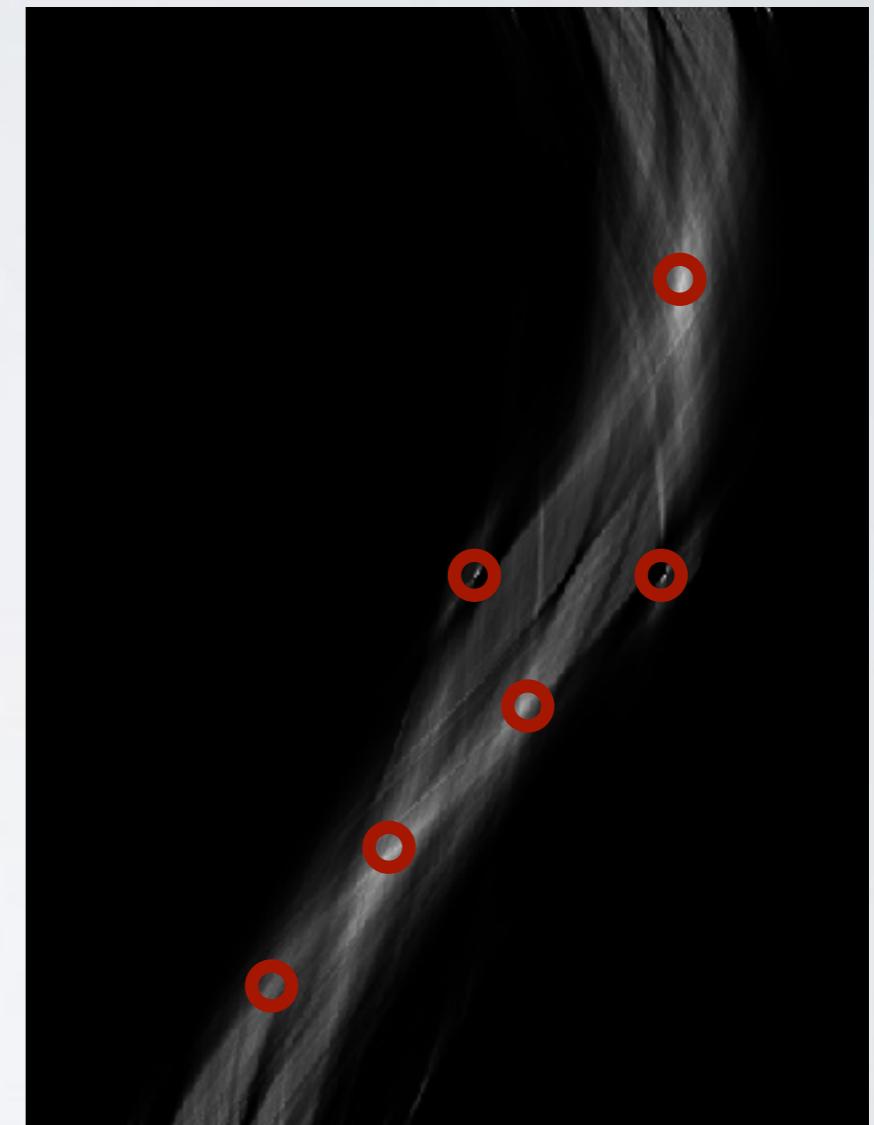
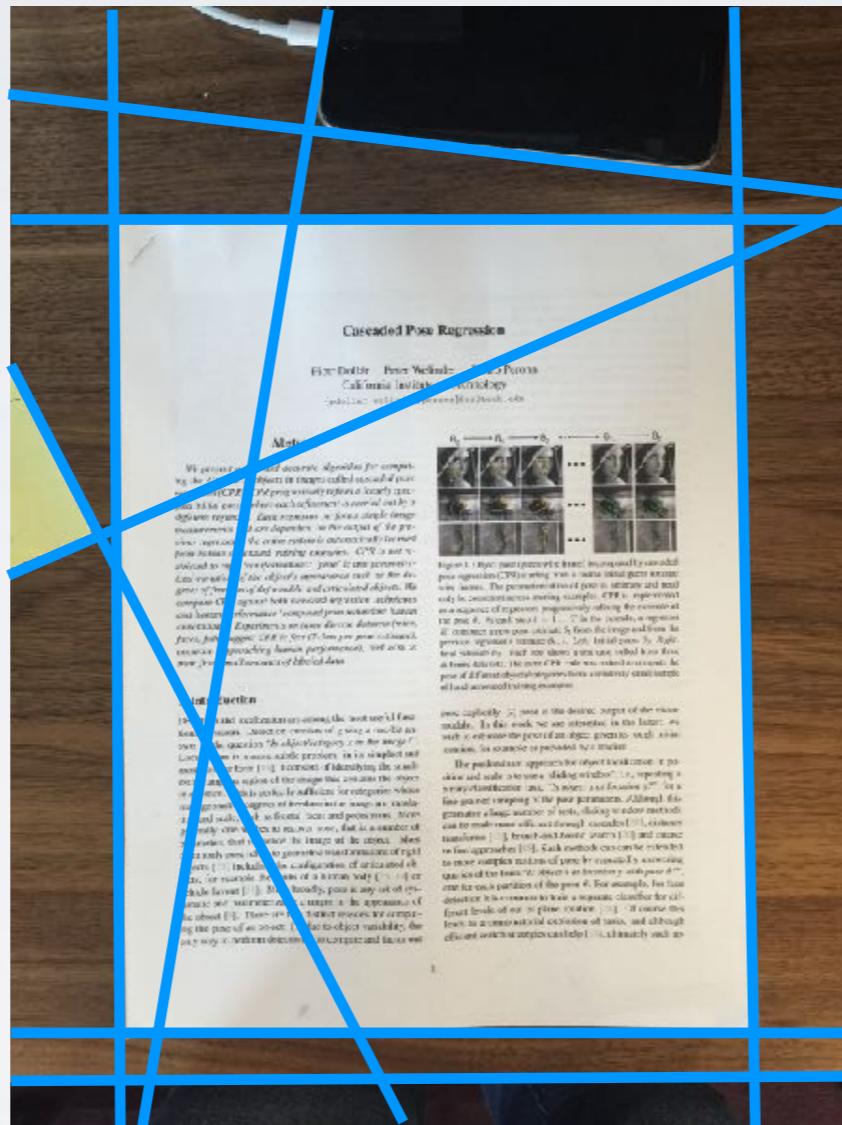
Find Lines



Find Corners



Find Quadrilateral



STEP BY STEP

Edge Detection



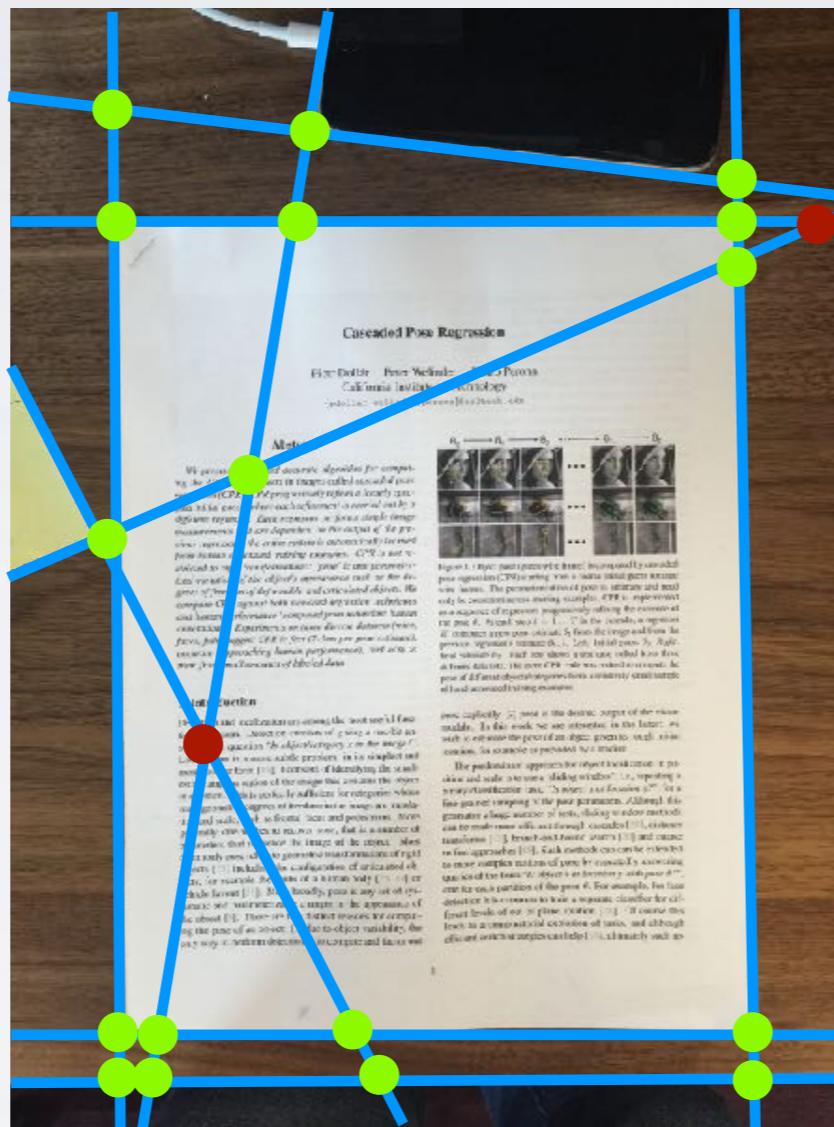
Find Lines



Find Corners



Find Quadrilateral



STEP BY STEP

Edge Detection



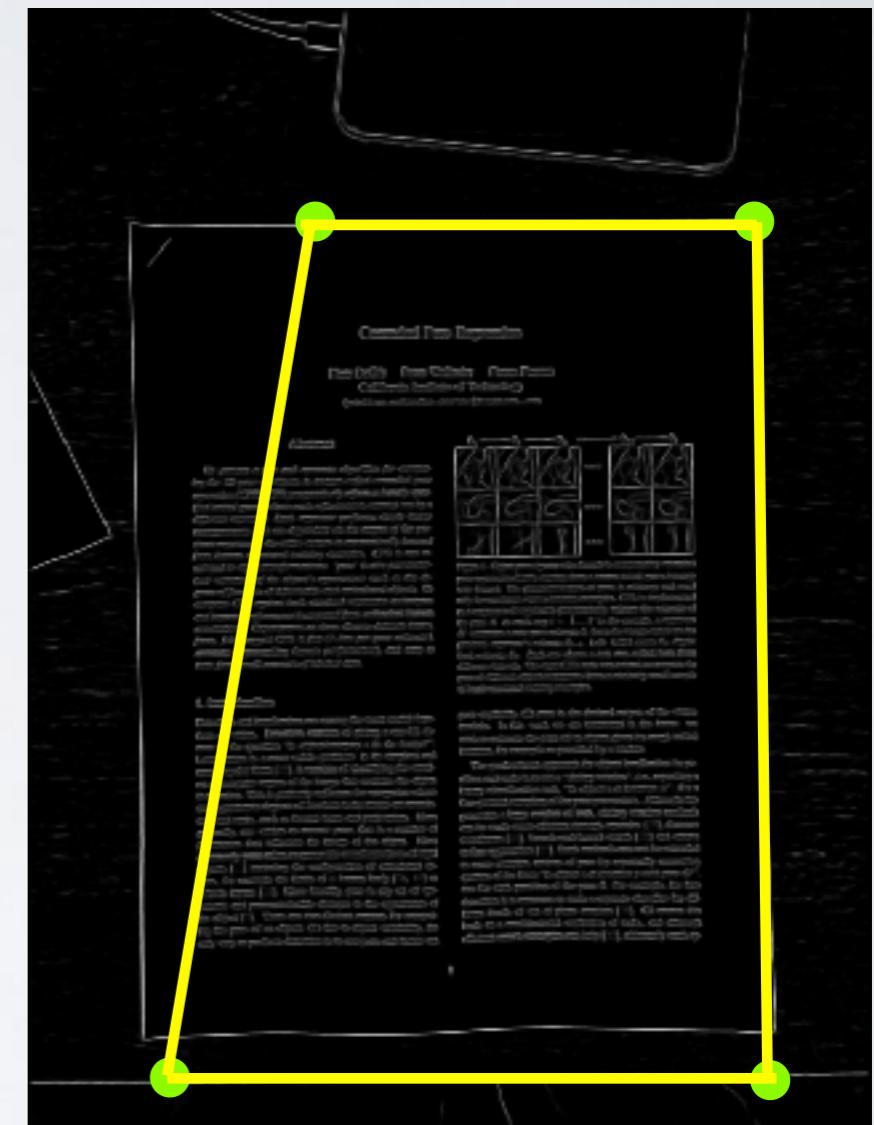
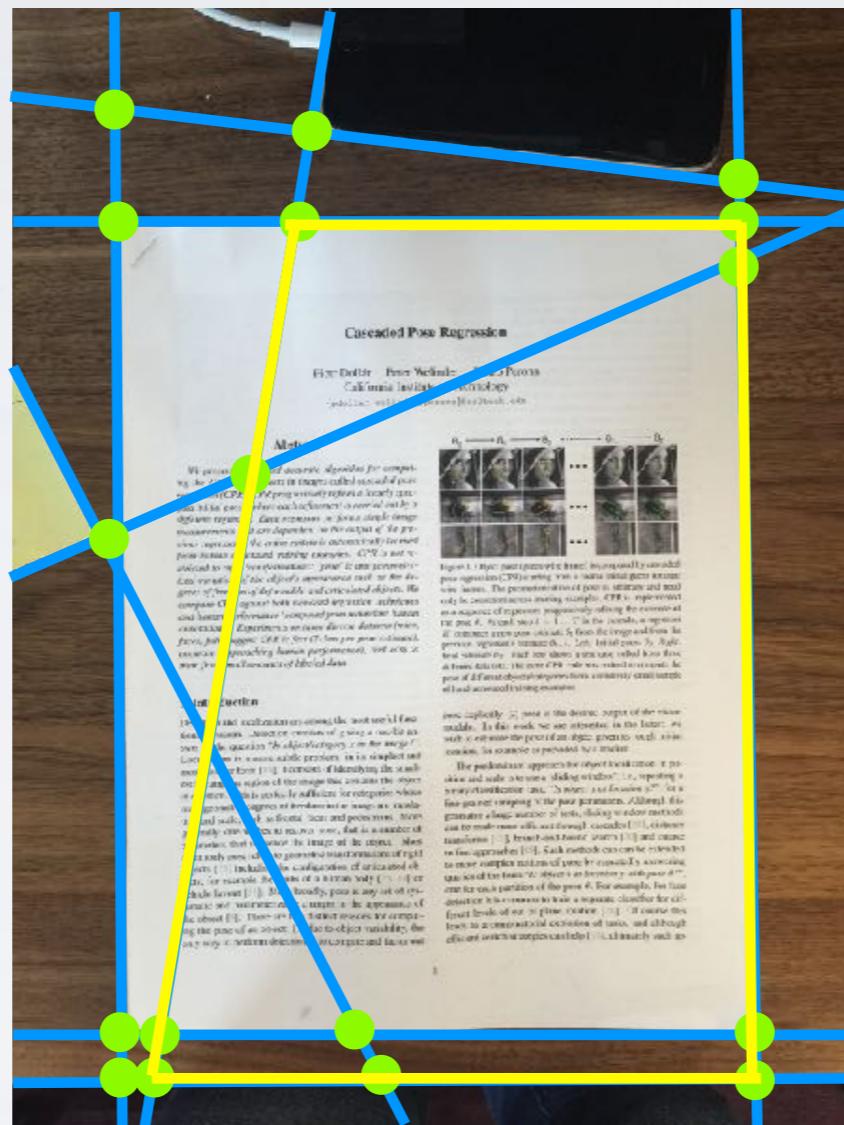
Find Lines



Find Corners



Find Quadrilateral



STEP BY STEP

Edge Detection



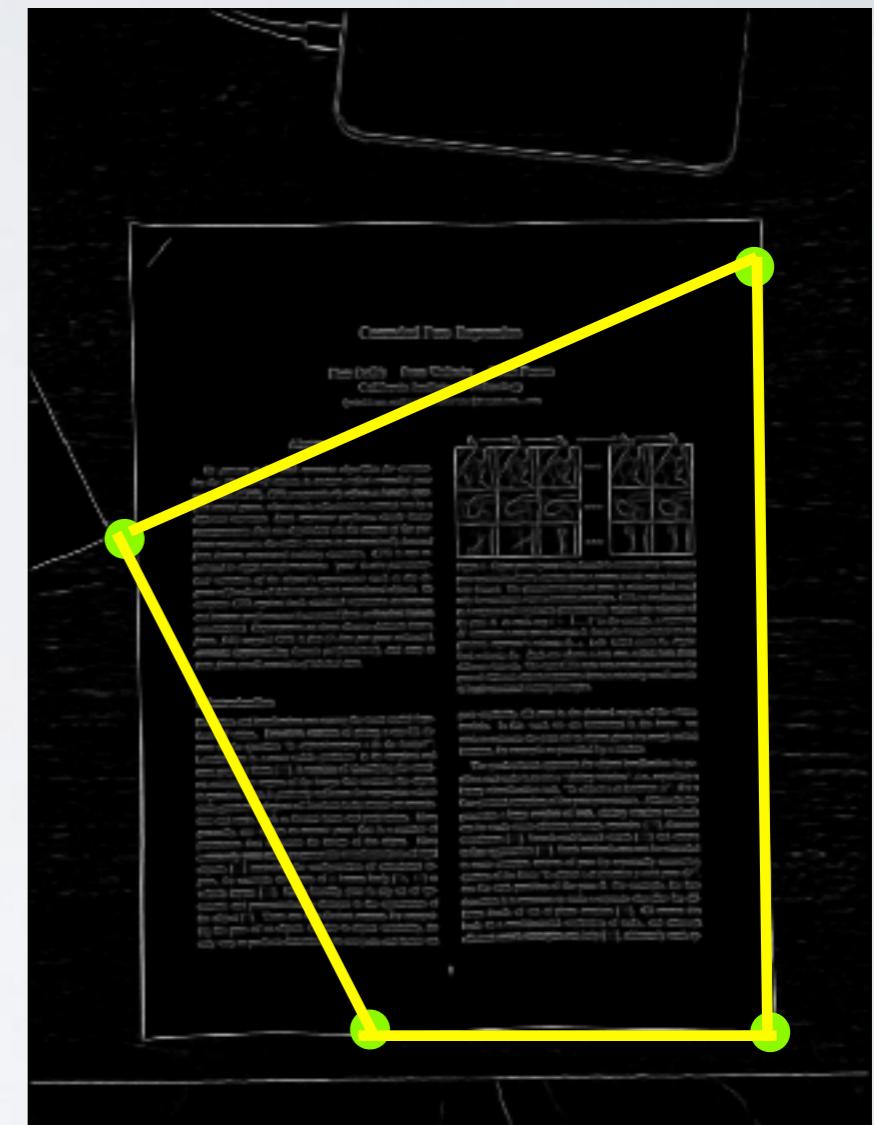
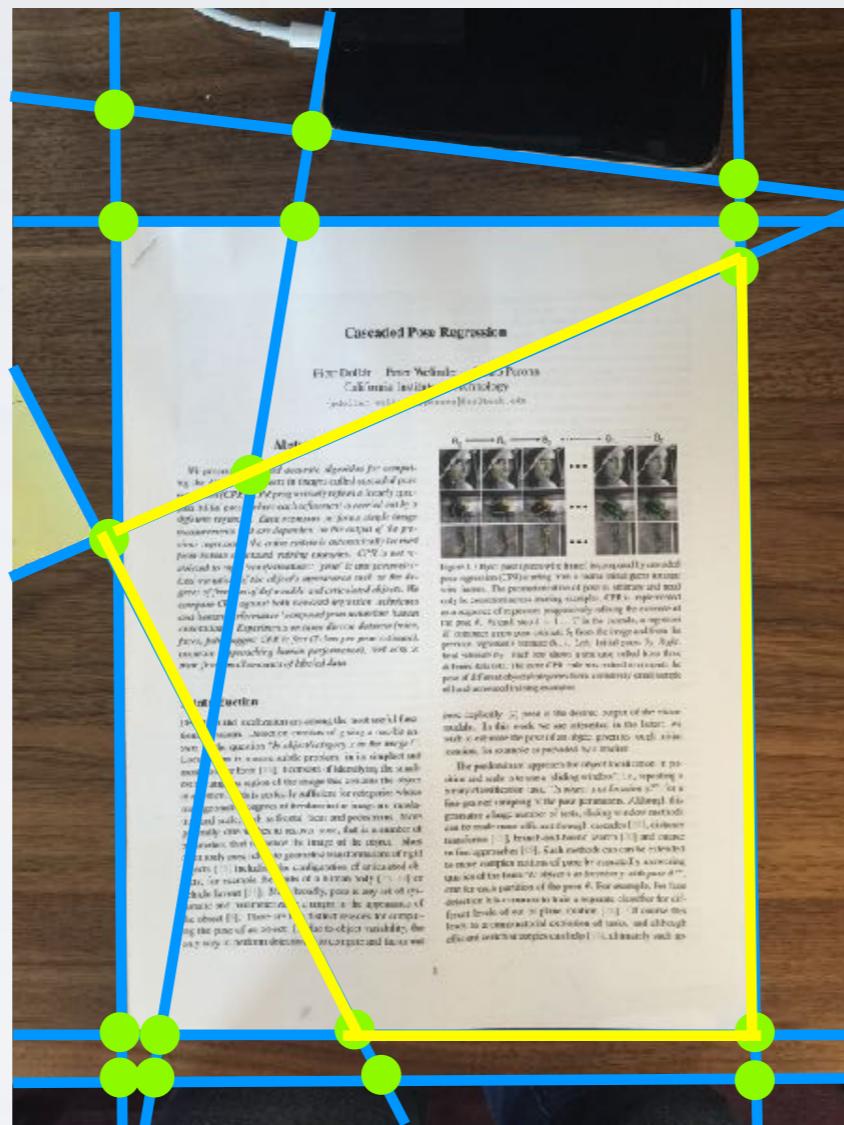
Find Lines



Find Corners



Find Quadrilateral



STEP BY STEP

Edge Detection



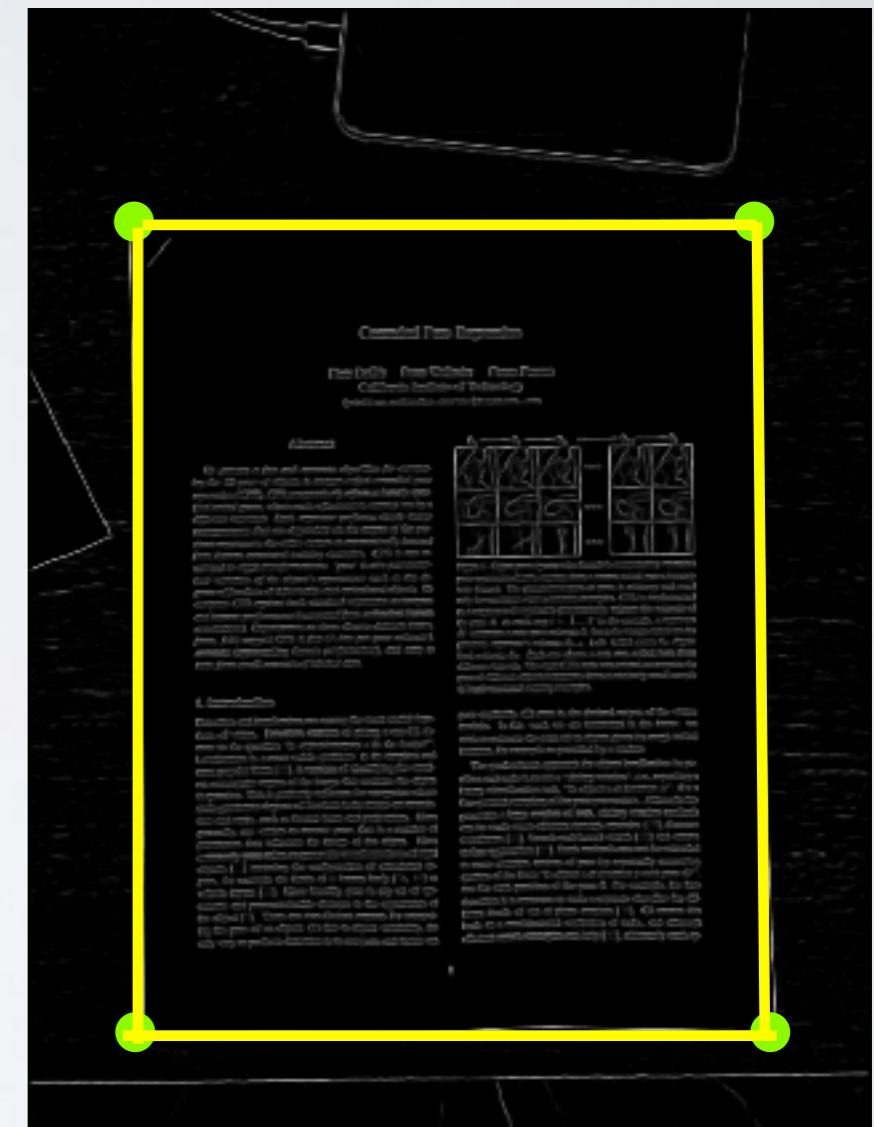
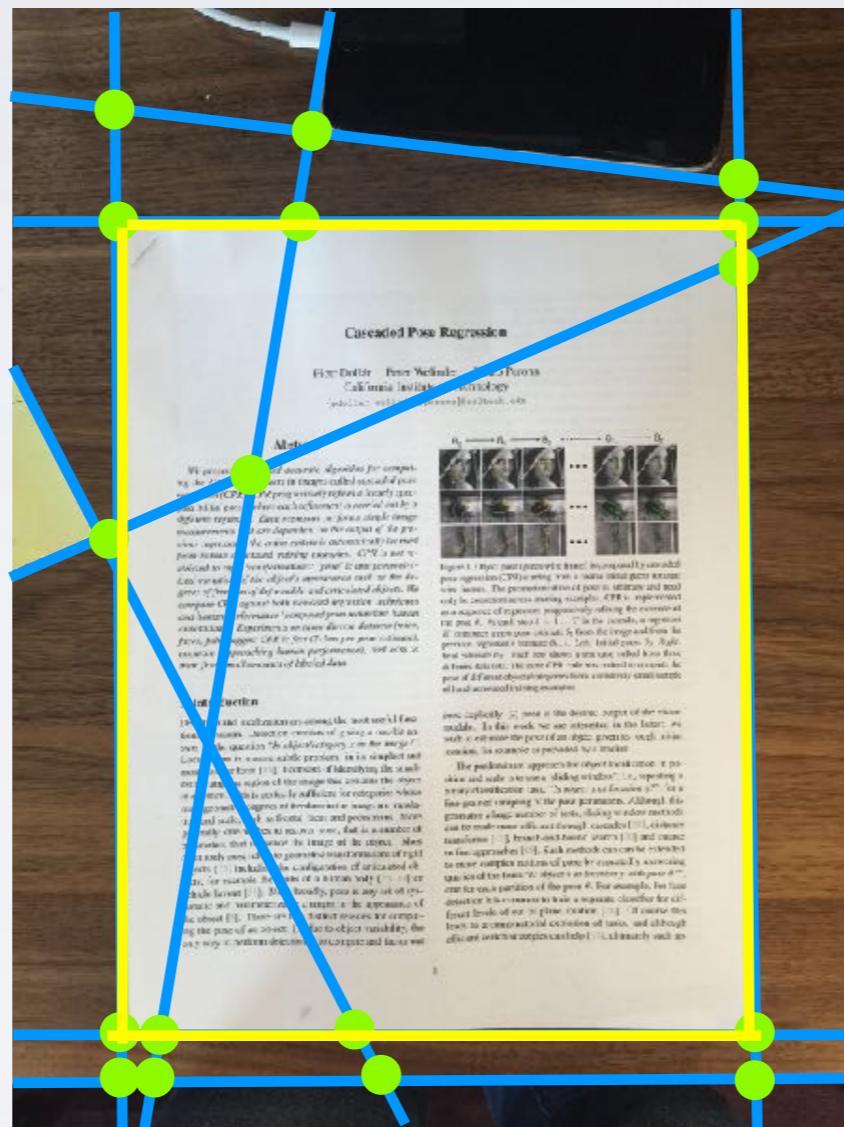
Find Lines



Find Corners



Find Quadrilateral



REAL-TIME DEMO

Edge Detection



Find Lines



Find Corners



Find Quadrilateral

