(Figure 1.8) shows how hard it is to model vision accurately. If you make up a version of this disk into a spinner (push a matchstick through the centre) and spin it anticlockwise, you do not see three dark rings, you will see three *coloured* ones. The outside one will appear to be *red*, the middle one a sort of *green* and the inner one deep *blue*. (This can depend greatly on lighting, and contrast between the black and white on the disk. If the colours are not clear, try it in a different place, with different lighting.) You can appear to explain this when you notice that the red colours are associated with the long lines and the blue with short lines. But this is from physics, not psychology. Now spin the disk clockwise. The order of the colours reverses: *red* is associated with the *short* lines (inside) and *blue* with the *long* lines (outside). So the argument from physics is clearly incorrect, since red is now associated with short lines, not long ones, revealing the need for psychological explanation of the eyes' function. This is not colour perception; see Armstrong (1991) for an interesting (and interactive) study of colour theory and perception.

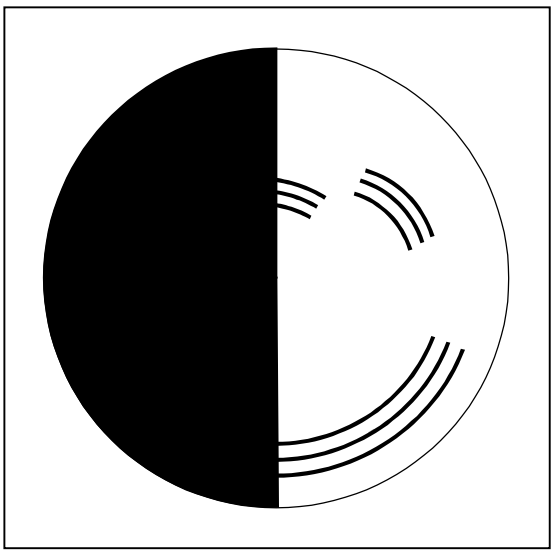

Figure 1.8 Benham's disk

There are many texts on human vision. One popular text on human visual perception is by Schwartz (2004) and there is an online book, *The Joy of Vision* (http://www.yorku.ca/eye/thejoy.htm): useful, despite its title! Marr's seminal text (Marr, 1982) is a computational investigation into human vision and visual perception, investigating it from a computer vision viewpoint. For further details on pattern processing in human vision, see Bruce and Green (1990); for more illusions see Rosenfeld and Kak (1982). Many of the properties of human vision are hard to include in a computer vision system, but let us now look at the basic components that are used to make computers see.

1.4 Computer vision systems

Given the progress in computer technology, computer vision hardware is now relatively inexpensive; a basic computer vision system requires a camera, a camera interface and a computer. These days, some personal computers offer the capability for a basic vision system, by including a camera and its interface within the system. There are specialized systems for vision, offering high performance in more than one aspect. These can be expensive, as any specialist system is.

1.4.1 Cameras

A *camera* is the *basic sensing element*. In simple terms, most cameras rely on the property of light to cause hole/electron pairs (the charge carriers in electronics) in a conducting material. When a potential is applied (to attract the charge carriers), this charge can be sensed as current. By Ohm's law, the voltage across a resistance is proportional to the current through it, so the current can be turned in to a voltage by passing it through a resistor. The number of hole/electron pairs is proportional to the amount of incident light. Accordingly, greater charge (and hence greater voltage and current) is caused by an increase in brightness. In this manner cameras can provide as output, a voltage that is proportional to the brightness of the points imaged by the camera. Cameras are usually arranged to supply video according to a specified standard. Most will aim to satisfy the *CCIR standard* that exists for closed circuit television (CCTV) systems.

There are three main types of camera: *vidicons*, *charge coupled devices* (CCDs) and, more recently, *CMOS* cameras (complementary metal oxide silicon, now the dominant technology for logic circuit implementation). Vidicons are the *older* (analogue) technology which, although cheap (mainly by virtue of longevity in production), are being replaced by the *newer* CCD and CMOS *digital* technologies. The digital technologies now dominate much of the camera market because they are *lightweight* and *cheap* (with other advantages) and are therefore used in the domestic video market.

Vidicons operate in a manner akin to a television in reverse. The image is formed on a screen, and then sensed by an electron beam that is scanned across the screen. This produces an output which is continuous; the output *voltage* is proportional to the *brightness* of points in the scanned line, and is a continuous signal, a voltage which varies continuously with time. In contrast, CCDs and CMOS cameras use an array of sensors; these are regions where *charge* is collected which is proportional to the *light* incident on that region. This is then available in discrete, or *sampled*, form as opposed to the continuous sensing of a vidicon. This is similar to human vision with its array of cones and rods, but digital cameras use a *rectangular* regularly spaced lattice, whereas human vision uses a *hexagonal* lattice with irregular spacing.

Two main types of semiconductor pixel sensor are illustrated in Figure 1.9. In the *passive sensor*, the charge generated by incident light is presented to a bus through a pass transistor.

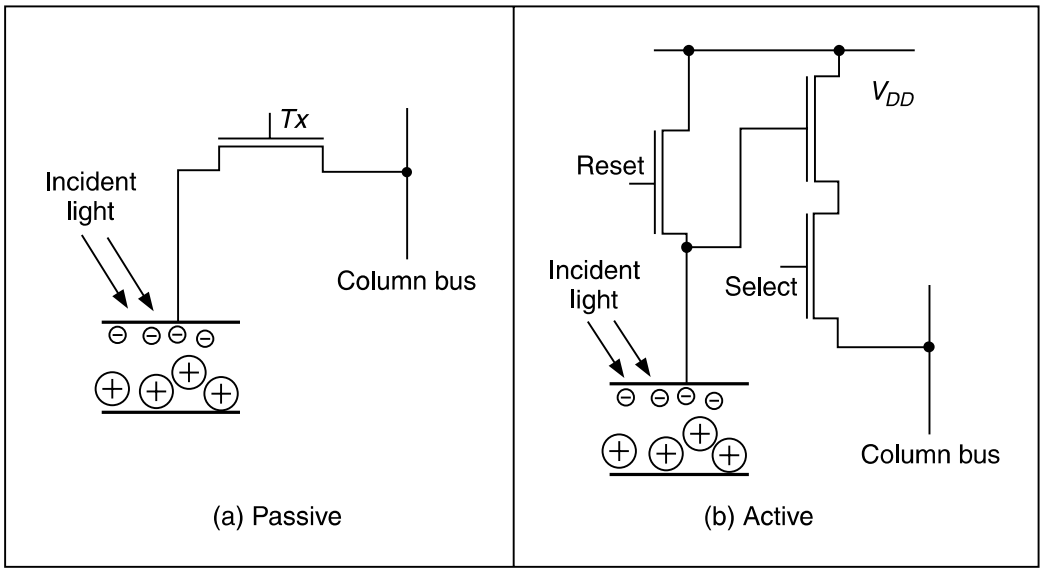

Figure 1.9 Pixel sensors

When the signal Tx is activated, the pass transistor is enabled and the sensor provides a capacitance to the bus, one that is proportional to the incident light. An *active pixel* includes an amplifier circuit that can compensate for limited fill factor of the photodiode. The select signal again controls presentation of the sensor's information to the bus. A further reset signal allows the charge site to be cleared when the image is rescanned.

The basis of a CCD sensor is illustrated in Figure 1.10. The number of charge sites gives the resolution of the CCD sensor; the contents of the charge sites (or buckets) need to be converted to an output (voltage) signal. In simple terms, the contents of the buckets are emptied into vertical transport registers which are shift registers moving information towards the horizontal transport registers. This is the column bus supplied by the pixel sensors. The horizontal transport registers empty the information row by row (point by point) into a signal conditioning unit, which transforms the sensed charge into a voltage which is proportional to the charge in a bucket, and hence proportional to the brightness of the corresponding point in the scene imaged by the camera. The CMOS cameras are like a form of memory: the charge incident on a particular site in a two-dimensional lattice is proportional to the brightness at a point. The charge is then read like computer memory. (In fact, a computer RAM chip can act as a rudimentary form of camera when the circuit, the one buried in the chip, is exposed to light.)

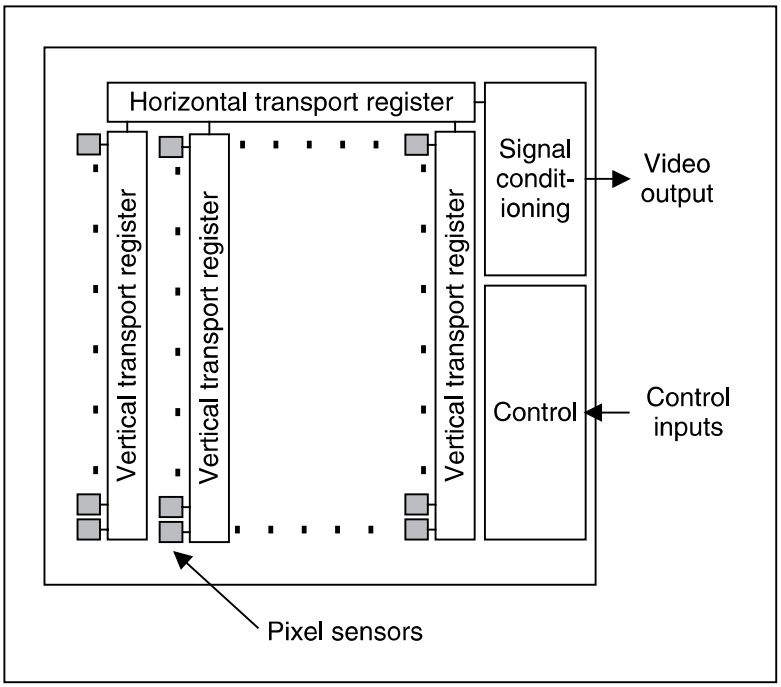

Figure 1.10 CCD sensing element

There are many more varieties of vidicon (Chalnicon, etc.) than there are of CCD technology (charge injection device, etc.), perhaps owing to the greater age of basic vidicon technology. Vidicons are cheap but have a number of intrinsic performance problems. The scanning process essentially relies on moving parts. As such, the camera performance will change with time, as parts *wear*; this is known as *ageing*. Also, it is possible to *burn* an image into the scanned screen by using high incident light levels; vidicons can also suffer *lag*, that is, a delay in response to moving objects in a scene. The digital technologies are dependent on the physical arrangement of charge sites and as such do not suffer from ageing, but can suffer from irregularity in the charge sites' (silicon) material. The underlying technology also makes CCD and CMOS cameras

less sensitive to lag and burn, but the signals associated with the CCD transport registers can give rise to *readout effects*. Charge coupled devices only came to dominate camera technology when technological difficulty associated with *quantum efficiency* (the magnitude of response to incident light) for the shorter, blue, wavelengths was solved. One of the major problems in CCD cameras is *blooming*, where bright (incident) light causes a bright spot to grow and disperse in the image (this used to happen in the analogue technologies too). This happens much less in CMOS cameras because the charge sites can be much better defined and reading their data is equivalent to reading memory sites as opposed to shuffling charge between sites. Also, CMOS cameras have now overcome the problem of *fixed pattern noise* that plagued earlier MOS cameras. The CMOS cameras are actually much more recent than CCDs. This begs a question as to which is better: CMOS or CCD? Given that they will be both be subject to much continued development, CMOS is a cheaper technology and it lends itself directly to intelligent cameras with on-board processing. This is mainly because the feature size of points (pixels) in a CCD sensor is limited to be about 4 μm so that enough light is collected. In contrast, the feature size in CMOS technology is considerably smaller, currently at around 0.1 μm. Accordingly, it is now possible to integrate signal processing within the camera chip and thus it is perhaps possible that CMOS cameras will eventually replace CCD technologies for many applications. However, the more modern CCDs also have on-board circuitry, and their process technology is more mature, so the debate will continue.

Finally, there are specialist cameras, which include *high-resolution* devices, which can give pictures with a great number of points, *low-light level* cameras, which can operate in very dark conditions (this is where vidicon technology is still found), and *infrared* cameras, which sense heat to provide thermal images. Increasingly, *hyperspectral* cameras are available, which have more sensing bands. For more detail concerning modern camera practicalities and imaging systems, see Nakamura (2005). For more detail on sensor development, particularly CMOS, Fossum (1997) is well worth a look.

1.4.2 Computer interfaces

This technology is in a rapid state of change, owing to the emergence of digital cameras. Essentially, the image sensor converts light into a signal which is expressed either as a continuous signal or in sampled (digital) form. Some (older) systems expressed the camera signal as an analogue continuous signal, according to a standard, often the CCIR standard, and this was converted at the computer (and still is in some cases). Modern digital systems convert the sensor information into digital information with on-chip circuitry and then provide the digital information according to a specified standard. The older systems, such as surveillance systems, supplied (or supply) video, whereas the newer systems are digital. Video implies delivering the moving image as a sequence of *frames* and these can be in analogue (continuous) or discrete (sampled) form, of which one format is digital video (DV).

An interface that converts an *analogue* signal into a set of digital numbers is called a *framegrabber*, since it grabs frames of data from a *video sequence*, and is illustrated in Figure 1.11. Note that cameras that provide digital information do not need this particular interface (it is inside the camera). However, an analogue camera signal is *continuous* and is transformed into *digital* (discrete) format using an analogue-to-digital (A/D) converter. *Flash converters* are usually used owing to the high speed required for conversion, say 11 MHz, which cannot be met by any other conversion technology. Usually, 8 bit A/D converters are used; at 6 dB/bit, this gives 48 dB, which just satisfies the CCIR stated *bandwidth* of approximately 45 dB. The output of the A/D converter is often fed to *look-up tables* (LUTs), which implement

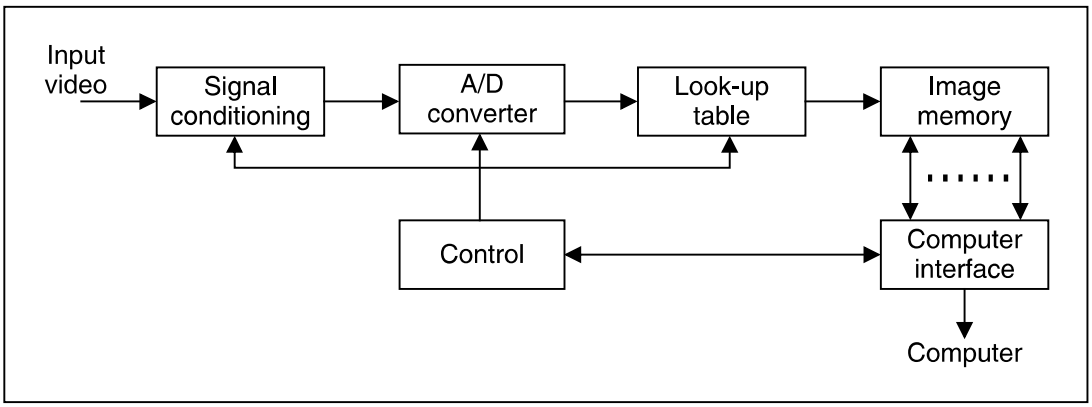

Figure 1.11 A computer interface: a framegrabber

designated conversion of the input data, but in hardware rather than in software, and this is very fast. The outputs of the A/D converter are then stored. Note that there are aspects of the sampling process that are of considerable interest in computer vision; these are covered in Chapter 2.

In digital camera systems this processing is usually performed on the camera chip, and the camera eventually supplies digital information, often in coded form. IEEE 1394 (or *firewire*) is a way of connecting devices external to a computer and is often used for digital video cameras as it supports high-speed digital communication and can provide power; this is similar to universal serial bus (USB), which can be used for still cameras. Firewire needs a connection system and software to operate it, and these can be easily acquired. One important aspect of Firewire is its support of isochronous transfer operation which guarantees timely delivery of data, which is of importance in video-based systems.

There are many different ways to design framegrabber units, especially for specialist systems. Note that the control circuitry has to determine exactly when image data is to be sampled. This is controlled by synchronization pulses that are supplied within the video signal: the sync signals, which control the way video information is constructed. Television pictures are constructed from a set of *lines*, those lines scanned by a camera. To reduce requirements on transmission (and for viewing), the 625 lines in the *PAL* system (*NTSC* is of lower resolution) are transmitted in two *fields*, each of 312.5 lines, as illustrated in Figure 1.12. (Currently, in high-definition television,

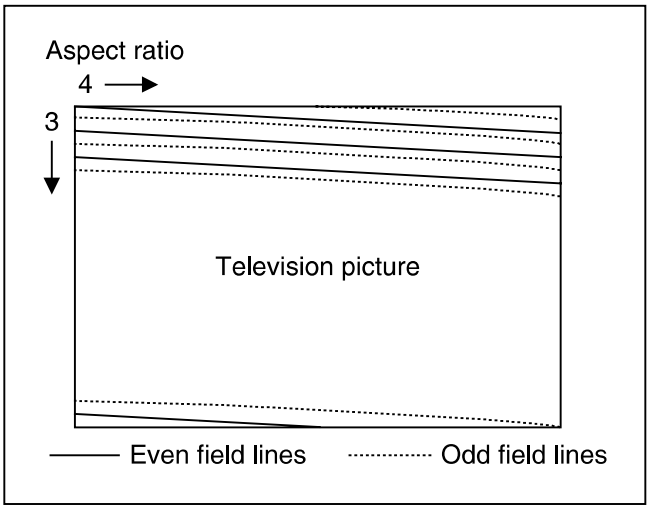

Figure 1.12 Interlacing in television pictures

there is some debate between the corporations who do not want interlacing, and those who do, e.g. the television broadcasters.) If you look at a television, but not directly, the flicker due to *interlacing* can be perceived. When you look at the television directly, persistence in the human eye ensures that you do not see the *flicker*. These fields are called the *odd* and *even* fields. There is also an *aspect ratio* in picture transmission: pictures are arranged to be 1.33 times longer than they are high. These factors are chosen to make television images attractive to *human* vision, and can complicate the design of a framegrabber unit. Some conversion systems accept PAL or NTSC video and convert it to the firewire system.

Nowadays, *digital video cameras* can provide digital output, in *progressive scan* (without interlacing), delivering sequences of images that are readily processed. Or there are *webcams*, or just *digital camera systems* that deliver images straight to the computer. Life just gets easier!

This completes the material we need to cover for basic computer vision systems. For more detail concerning the practicalities of computer vision systems see, for example, Davies (2005) (especially for product inspection) or Umbaugh (2005) (and both offer much more than this).

1.4.3 Processing an image

Most image processing and computer vision techniques are implemented in computer *software*. Often, only the simplest techniques migrate to hardware, although coding techniques to maximize efficiency in image transmission are of sufficient commercial interest that they have warranted extensive, and very sophisticated, hardware development. The systems include the Joint Photographic Expert Group (JPEG) and the Moving Picture Expert Group (MPEG) image coding formats. C, C++ and Java™ are by now the most popular languages for vision system implementation, because of strengths in integrating high- and low-level functions, and the availability of good compilers. As systems become more complex, C++ and Java become more attractive when encapsulation and polymorphism may be exploited. Many people use Java as a development language partly because of platform independence, but also because of ease in implementation (although some claim that speed and efficiency are not as good as in C/C++). There is considerable implementation advantage associated with use of the Java Advanced Imaging API (application programming interface). There is an online demonstration site, for educational purposes only, associated with this book, to be found of the book's website at http://www.ecs.soton.ac.uk/~msn/book/new_demo/. This is based around Java, so that the site can be used over the web (as long as Java is installed and up to date). Some textbooks offer image processing systems implemented in these languages. Many commercial packages are available, although these are often limited to basic techniques, and do not include the more sophisticated shape extraction techniques. The Visiquest (was Khoros) image processing system has attracted much interest; this is a schematic (data-flow) image processing system where a user links together chosen modules. This allows for better visualization of information flow during processing. However, the underlying mathematics is not made clear to the user, as it can be when a mathematical system is used. There is a textbook, and a very readable one at that, by Efford (2000), which is based entirely on Java and includes, via a CD, the classes necessary for image processing software development. Other popular texts include those that present working algorithms, such as Seul et al. (2001) and Parker (1996).

In terms of *software packages*, one of the most popular is OpenCV, whose philosophy is to 'aid commercial uses of computer vision in human–computer interface, robotics, monitoring, biometrics and security by providing a free and open infrastructure where the distributed efforts of the vision community can be consolidated and performance optimized'. This contains a wealth

of technique and (optimized) implementation; there is even a Wikipedia entry and a discussion website supporting it. Then there are the VXL libraries (the Vision-*something*-Libraries, groan). This is 'a collection of C++ libraries designed for computer vision research and implementation'. Finally, there is Adobe's Generic Image Library (GIL), which aims to ease difficulties with writing imaging-related code that is both generic and efficient. Note that these are open source, but there are licences and conditions on use and exploitation.

A set of web links is shown in Table 1.2 for established freeware and commercial software image processing systems. Perhaps the best selection can be found at the general site, from the computer vision homepage software site at Carnegie Mellon (repeated later in Table 1.5).

Table 1.2 Software websites

Packages (freeware or student version indicated by *)		
General Site	Carnegie Mellon	http://www.cs.cmu.edu/afs/cs/project/cil/ftp/html/v-source.html (large popular index including links to research code, image processing toolkits, and display tools)
Visiquest (Khoros)	Accusoft	http://www.accusoft.com/
	Hannover University	http://www.tnt.uni-hannover.de/soft/imgproc/khoros/
AdOculos* (+ Textbook)	The Imaging Source	http://www.theimagingsource.com/
CVIPtools*	Southern Illinois University	http://www.ee.siue.edu/CVIPtools/
LaboImage*	Geneva University	http://cuiwww.unige.ch/~vision/LaboImage/labo.html
TN-Image*	Thomas J. Nelson	http://brneurosci.org/tnimage.html (scientific image analysis)
OpenCV	Intel	http://www.intel.com/technology/computing/opencv/index.htm and http://sourceforge.net/projects/opencvlibrary/
VXL	Many international contributors	http://vxl.sourceforge.net/
GIL	Adobe	http://opensource.adobe.com/gil/

1.5 Mathematical systems

Several *mathematical systems* have been developed. These offer what is virtually a word-processing system for mathematicians. Many are screen based, using a Windows system. The advantage of these systems is that you can transpose mathematics pretty well directly from textbooks, and see how it works. Code functionality is not obscured by the use of data structures, although this can make the code appear cumbersome. A major advantage is that the system provides low-level functionality and data visualization schemes, allowing the user to concentrate on techniques alone. Accordingly, these systems afford an excellent route to understand, and appreciate, mathematical systems before the development of application code, and to check that the final code works correctly.