

UNIT I INTRODUCTION

Image representation and image analysis tasks Basic concepts, Image digitization, Digital image properties, Data structures for Image Analysis - Levels of image data representation, Traditional and Hierarchical image data structures.

COMPUTER VISION:

Computer vision is a field of artificial intelligence (AI) enabling computers to derive information from images, videos, and other inputs. Vision allows human to perceive and understand the world surrounding them, while computer vision aims to duplicate the effect of human vision by electronically perceiving and understanding an image.

Computers use digital images and deep learning models to accurately identify and classify objects and react to them. Computer Vision in AI is dedicated to the development of automated systems that can interpret visual data such as photographs or motion pictures) in same manner as people do. The idea behind Computer Vision is to instruct computers to interpret and comprehend images on a pixel-by-pixel basis, which is the foundation of Computer Vision field. Regarding the technical side, computer will seek to extract visual data, manage it and analyze the outcomes using sophisticated software programs.

How Computer Vision Works?

Computer Vision application use input from sensing devices, AI, machine learning and deep learning to replicate the way the human vision systems works. Computer Vision applications run on algorithms that are trained on massive amounts of visual data /images in the cloud. They recognize patterns in the visual data and use those patterns to determine the content of other images.

How An Image Is Analyzed with Computer Vision?

A sensing device captures an image. The sensing device is often just a camera, but could be a video camera, medical imaging device or any other type of device that captures an image for analysis. The image is then sent to an interpreting device (which uses pattern recognition to break the image down, compare the patterns in the image against its library of known patterns, and determine if any of the content in the image is a match). The pattern could be something general, like the appearance of a certain type of object or it could be based on unique identifiers such as facial features. A user requests specific information about an image, and interpreting device provide the information requested based on its analysis of the image.

Deep Learning and Computer Vision

Modern Computer Vision applications are shifting away from statistical methods for analyzing images and increasingly rely on deep learning. With deep learning, a Computer Vision application runs on a type of algorithm called a neural network, which allows it deliver even more accurate

analyses of images. In addition, deep learning allows a computer vision program to retain the information from each image it analyses, so it gets more and more accurate the more it is used.

Applications Of Computer Vision

- 1) Facial recognition- Identifying individuals through visual analysis.
- 2) Self driving cars- Using CV to navigate and avoid obstacles.
- 3) Robotic automation- Enabling robots to perform tasks and make decisions based on visual inputs.
- 4) Medical anomaly detection- Detecting abnormalities in medical images for improved diagnosis.
- 5) Sports performance analysis- Tracking athlete movements to analyze and enhance performance.
- 6) Manufacturing fault detection- Identifying defects in products during the manufacturing process.
- 7) Agricultural monitoring- Monitoring crop growth, livestock health and weather conditions through visual data.

Why Is Computer Vision Difficult?

- 1) Loss of information in 3D 2D as the geometric properties of the object have been approximated by the image capturing devices.
- 2) Interpretation of image- The practical ability of a machine to understand observations remains very limited.
- 3) Noise- It is inherently present in each measurement in the real world. Thus mathematical tools are used to cope with uncertainty. More complex tools makes image analysis much more complicated.
- 4) Too much data- Image and video sequences are huge. If the processing we devise is not very simple, then it is hard to achieve real-time performance.
- 5) Brightness measured in an image is given by complicated image formation physics.
- 6) Local window vs need for global view- It is often very difficult to interpret an image if it is seen only locally or if only a few local keyholes are available.

Computer sees the image through a keyhole. Seeing world through a keyhole makes it very difficult to understand more global context.

IMAGE REPRESENTATION AND IMAGE ANALYSIS TASKS:

Transition from the input image(s) to the model reduces the information contained in the image to relevant information for the application domain. This process is usually divided into several steps and several levels representing the image are used.

The bottom layer contains raw image data and the higher levels interpret the data. Computer vision designs these intermediate representations and algorithms serving to establish and maintain relations between entities within and between layers.

Image representation can be roughly divided according to data organization into four levels.

The boundaries between individual levels are inexact. The flow of information does not need to be unidirectional; often feedback loops are introduced which allow the modification of algorithms according to intermediate results.

This hierarchy of image representation and related algorithms is frequently categorized in an even simpler way—*low-level* image processing and *high-level* image understanding.

Low-level processing methods usually use very little knowledge about the content of images. Low-level methods may include image compression, pre-processing methods for noise filtering, edge extraction, and image sharpening. Low-level image processing uses data which resemble the input image; If the image is to be processed using a computer it will be digitized first, after which it may be represented by a rectangular matrix with elements corresponding to the brightness at appropriate image locations. More probably, it will be presented in color, usually three channels: red, green and blue.

High-level processing is based on knowledge, goals, and plans of how to achieve those goals, and artificial intelligence methods are widely applicable. High-level computer vision tries to imitate human cognition and the ability to make decisions according to the information contained in the image.

High-level vision begins with some form of formal model of the world, and then the ‘reality’ perceived in the form of digitized images is compared to the model. A match is attempted, and when differences emerge, partial matches (or subgoals) are sought that overcome them; the computer switches to low-level image processing to find information needed to update the model. This process is then repeated iteratively, and ‘understanding’ an image thereby becomes a co-operation between top-down and bottom-up processes. A feedback loop is introduced in which high-level partial results create tasks for low-level image processing, and the iterative image understanding process should eventually converge to the global goal.

Computer vision is expected to solve very complex tasks, the goal being to obtain similar results to those provided by biological systems.

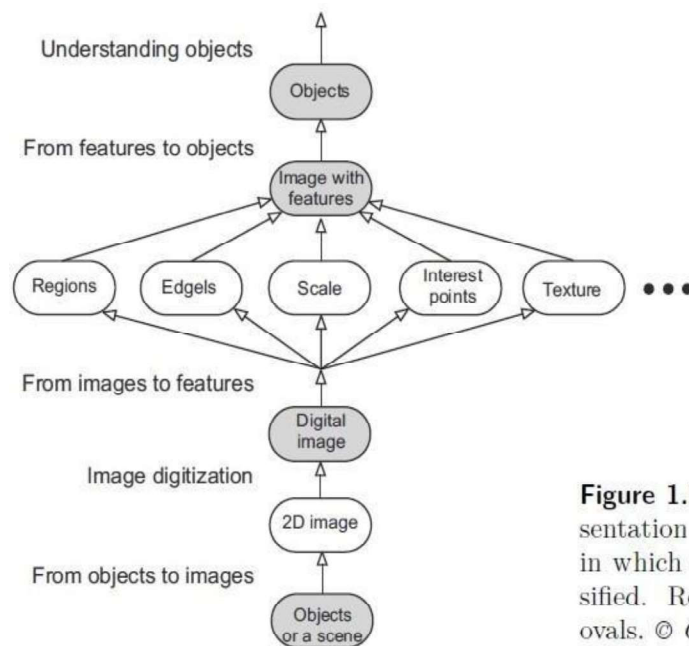


Figure 1.7: Four possible levels of image representation suitable for image analysis problems in which objects have to be detected and classified. Representations are depicted as shaded ovals. © Cengage Learning 2015.

The point is that a lot of a priorknowledge is used by humans to interpret the images; the machine only begins with an array of numbers and so will be attempting to make identifications and draw conclusions from data. Increasingly, data capture equipment is providing very large data sets that do *not* lend themselves to straightforward interpretation by humans.

Low-level computer vision techniques overlap almost completely with digital image processing, which has been practiced for decades. The following sequence of processing steps is commonly seen:

An image is captured by a sensor (such as a camera) and digitized; then the computer suppresses noise (image pre-processing) and may be enhances some object features which are relevant to understanding the image. Edge extraction is an example of processing carried out at this stage.

Image segmentation is the next step, in which the computer tries to separate objects from the image background and from each other. Total and partial segmentation may be distinguished; total segmentation is possible only for very simple tasks, an example being the recognition of dark non-touching objects from a light background.

Object description and classification in a totally segmented image are also understood as part of low-level image processing. Other low-level operations are image compression, and techniques to extract information from (but not *understand*) moving scenes.

Low-level image processing and high-level computer vision differ in the data used.

Low-level data are comprised of original images represented by matrices composed of brightness (or similar) values, while high-level data originate in images as well, but only those data which are relevant to high-level goals are extracted, reducing the data quantity considerably.

High-level data represent knowledge about the image content—for example, object size, shape, and mutual relations between objects in the image. High-level data are usually expressed in symbolic form.

High-level vision tries to extract and order image processing steps using all available knowledge—image understanding is the heart of the method, in which feedback from high-level to low-level is used. Unsurprisingly this task is very complicated and computationally intensive.

Figure 1.10 depicts several 3D vision tasks and algorithmic components expressed on different abstraction levels. In most cases, the bottom-up and top-down approach is adopted to fulfill the task.

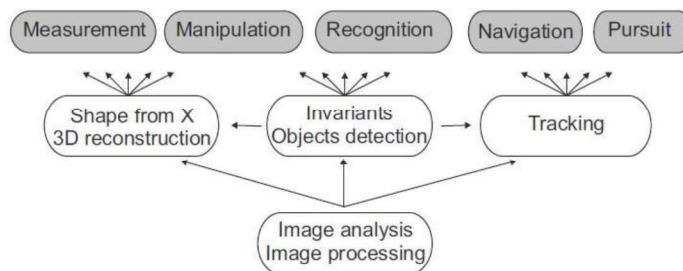


Figure 1.10: Several 3D computer vision tasks from the user’s point of view are on the upper line (filled). Algorithmic components on different hierarchical levels support it in a bottom-up fashion.

IMAGE, ITS REPRESENTATIONS AND PROPERTIES

Image representations:

Mathematical models are often used to describe images.

A monochrome or monochromatic image, object or palette is composed of one color (or values of one color). Images using only shades of grey are called grayscale (typically digital) or black-and-white (typically analog).

A scalar function might be sufficient to describe a monochromatic image, while vector functions may be used to represent color images consisting of three component colors.

Functions are categorized as

○ Continuous

○ Discrete ○

Digital.

A continuous function has continuous domain and range; if the domain set is discrete, then we have a discrete function; if the range set is also discrete, then we have a digital function

can be modeled by a continuous function of two variables $f(x, y)$ where (x, y) are coordinates in a plane, or perhaps three variables $f(x, y, t)$, where t is time.

The continuous image function

The (gray-scale) image function values correspond to brightness at image points. The function value can express other physical quantities as well (temperature, pressure distribution, distance from the observer, etc.). **Brightness** integrates different optical quantities—using brightness as a basic quantity allows us to avoid the complicated process of image formation.

The image on the retina or on a camera sensor is intrinsically two-dimensional (2D). We shall call such an image bearing information about brightness points an **intensity image**. The 2D image on the imaging sensor is commonly the result of projection of a three-dimensional (3D) scene. The simplest mathematical model for this is a pin-hole camera.

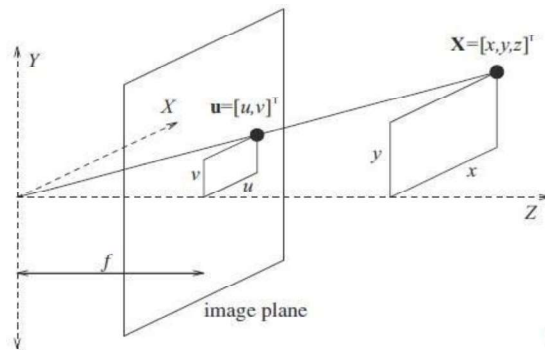


Figure 2.1: Perspective projection geometry.
© Cengage Learning 2015.

The image plane is the plane that contains the object's projected image.

The 2D intensity image is the result of a **perspective projection** of the 3D scene, which is modeled by the image captured by a pin-hole camera illustrated in Figure 2.1. In this figure, the image plane has been reflected with respect to the XY plane in order not to get a mirrored image with negative co-ordinates. The quantities x , y , and z are coordinates of the point $\mathbf{X}$ in a 3D scene, and f is the distance from the pinhole to the image plane. f is commonly called the **focal length** because in lenses it has a similar meaning. The projected point $\mathbf{u}$ has co-ordinates (u, v) in the 2D image plane, which can easily be derived from similar triangles.

$$u = \frac{x f}{z}, \quad v = \frac{y f}{z}. \quad (2.1)$$

A non-linear perspective projection is often approximated by a linear **parallel** (or **orthographic**) **projection**, where $f \rightarrow \infty$. Implicitly, x says that the orthographic projection is a limiting case of the perspective projection for faraway objects.

When 3D objects are mapped into the camera plane by perspective projection, a lot of information disappears because such a transform is not one-to-one.

Recovering information lost by perspective projection is only one, problem of computer vision—another is understanding image brightness.

The only information available in an intensity image is the brightness of the appropriate pixel, which is dependent on a number of independent factors such as object surface reflectance properties (given by the surface material, microstructure, and marking), illumination properties, and object surface orientation with respect to viewer and light source.

It is a non-trivial and again ill-posed problem to separate these components when trying to recover the 3D geometry of an object from the intensity image.

Image processing often deals with **static** images, in which time is constant. A monochromatic static image is represented by a continuous image function $f(x, y)$ whose arguments are coordinates in the plane.

Computerized image processing uses digital image functions which are usually represented by matrices, so co-ordinates are natural numbers. The domain of the image function is a region R in the plane

$$R = (x, y), 1 \leq x \leq x_m, 1 \leq y \leq y_n, \quad (2.2)$$

where x_m, y_n represent the maximal co-ordinates. The function has a limited domain, as the image function value is zero outside the domain R .

The range of image function values is also limited; by convention, in monochromatic images the lowest value corresponds to black and the highest to white. Brightness values bounded by these limits are **gray-levels**. The quality of a digital image grows in proportion to the spatial, spectral, radiometric, and time resolutions. The **spatial resolution** is given by the proximity of image samples in the image plane; **spectral resolution** is given by the bandwidth of the light frequencies captured by the sensor; **radiometric resolution** corresponds to the number of distinguishable gray-levels; and **time resolution** is given by the interval between time samples at which images are captured.

IMAGE DIGITIZATION

An image to be processed by computer must be represented using an appropriate discrete data structure, for example, a matrix. An image captured by a sensor is expressed as a continuous function $f(x, y)$ of two coordinates in the plane. Image digitization means that the function $f(x, y)$ is **sampled** into a matrix with M rows and N columns. Image **quantization** assigns to each continuous sample an integer value—the continuous range of the image function $f(x, y)$ is split into K intervals. The finer the sampling (i.e., the larger M and N) and quantization (the larger K), the better the approximation of the continuous image function $f(x, y)$ achieved.

Sampling

A continuous image is digitized at **sampling points**. These sampling points are ordered in the plane, and their geometric relation is called the **grid**. The digital image is then a data structure, usually a matrix. Grids used in practice are usually square (Figure 2.2a) or hexagonal (Figure 2.2b). It is important to distinguish the grid from the raster; the **raster** is the grid on which a neighborhood relation between points is defined.

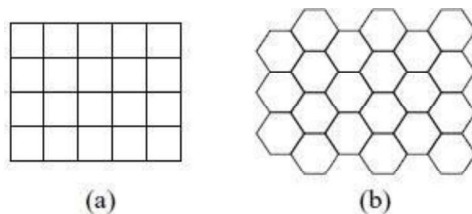


Figure 2.2: (a) Square grid. (b) Hexagonal grid.
© Cengage Learning 2015.

One infinitely small sampling point in the grid corresponds to one picture element also called a **pixel** or **image element** in the digital image; in a three-dimensional image, an image element is called a **voxel** (volume element). The set of pixels together covers the entire image;

Quantization

A value of the sampled image $f_s(j \Delta x, k \Delta y)$ is expressed as a digital value in image processing. The transition between continuous values of the image function (brightness) and its digitalequivalent is called **quantization**. The number of quantization levels should be high enough to permit human perception of shading details in the image. Most digital image processing devices use quantization into k equal intervals. If b bits are used to express the values of the pixel brightness then the number of brightness levels is $k = 2^b$.

The main problem in images quantized with insufficient brightness levels is the occurrence of false contours which effect arises when the number of brightness levels is lower than that which humans can easily distinguish.

DIGITAL IMAGE PROPERTIES

A digital image has several properties, both metric and topological.

Metric and topological properties of digital images

A digital image consists of picture elements with finite size—these pixels carry information about the brightness of a particular location in the image. Usually (and we assume this hereafter) pixels are arranged into a rectangular sampling grid. Such a digital image is represented by a two-dimensional matrix whose elements are natural numbers corresponding to the quantization levels in the brightness scale.

Some intuitively clear properties of continuous images have no straightforward analogy in the domain of digital images. **Distance** is an important example. Any function D holding the following three conditions is a ‘distance’ (or a *metric*)

$$\begin{aligned} D(\mathbf{p}, \mathbf{q}) &\geq 0 ; D(\mathbf{p}, \mathbf{q}) = 0 \text{ if and only if } \mathbf{p} = \mathbf{q} , && \textit{identity}, \\ D(\mathbf{p}, \mathbf{q}) &= D(\mathbf{q}, \mathbf{p}) , && \textit{symmetry}, \\ D(\mathbf{p}, \mathbf{r}) &\leq D(\mathbf{p}, \mathbf{q}) + D(\mathbf{q}, \mathbf{r}) , && \textit{triangular inequality}. \end{aligned}$$

The distance between points with co-ordinates (i, j) and (h, k) may be defined in several different ways.

The **Euclidean distance** D_E known from classical geometry and everyday experience is defined by

$$D_E((i, j), (h, k)) = \sqrt{(i - h)^2 + (j - k)^2}. \quad (2.3)$$

The advantage of Euclidean distance is that it is intuitively obvious. The disadvantages are costly calculation due to the square root, and its non-integral value.

The distance between two points can also be expressed as the minimum number of elementary steps in the digital grid which are needed to move from the starting point to the end point. If only horizontal and vertical moves are allowed, the **‘city block’ distance** D_4 is obtained (also called the L_1 metric or Manhattan distance, because of the analogy with the distance between two locations in a city with a rectangular grid of streets):
 $D_4(i, j), (h, k) = |i - h| + |j - k|$. (2.4)

If moves in diagonal directions are allowed in the digitization grid, we obtain the distance D_8 , or **‘chessboard’ distance**. D_8 is equal to the minimal number of king- moves on the chessboard from one part to another:

$$D_8((i, j), (h, k)) = \max \{ |i - h|, |j - k| \} \quad (2.5)$$

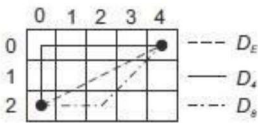


Figure 2.4: Distance metrics D_E , D_4 , and D_8 .

Pixel **adjacency** is another important concept in digital images. Two pixels (p, q) are called **4-neighbors** if they have distance $D_4(p, q) = 1$. Analogously, **8-neighbors** have $D_8(p, q) = 1$

It will become necessary to consider important sets consisting of several adjacent pixels—**regions** (in set theory, a region is a connected set). More descriptively, we can define a **path** from pixel P to pixel Q as a sequence of points $A_1, A_2, \dots, A_n$, where $A_1 = P$, $A_n = Q$, and A_{i+1} is a neighbor of A_i , $i = 1, \dots, n-1$; then a region is a set of pixels in which there is a path between any pair of its pixels, all of whose pixels also belong to the set. If there is a path between two pixels in the set of pixels in the image, these pixels are called **contiguous**

Assume that R_i are disjoint regions in the image, R be the union of all regions R_i ; R^C be the set complement of R with respect to the image. The subset of R^C which is contiguous with the image bounds is called the **background**, and the remainder of the complement R^C is called **holes**. A region is called **simple contiguous** if it has no holes. Equivalently, the complement of a simply contiguous region is contiguous. A region with holes is called **multiple contiguous**.

Note that the concept of region uses only the property ‘to be contiguous’. The brightness of a pixel is a very simple property which can be used to find objects in some images. All such points which are also contiguous constitute one object. A hole consists of points which do not belong to the object and are surrounded by the object, and all other points constitute the background.

An **edge** is a local property of a pixel and its immediate neighborhood—it is a vector given by a magnitude and direction which tells us how fast the image intensity varies in a small neighborhood of a pixel. Images with many brightness levels are used for edge computation, and the gradient of the image function is used to compute edges. The edge direction is perpendicular to the gradient direction which points in the direction of the fastest image function growth.

The **crack edge** creates a structure between pixels in a similar manner to that of cellular complexes. The **border** (boundary) of a region is another important concept in image analysis. The border of a region R is the set of pixels within the region that have one or more neighbors outside R . The definition corresponds to an intuitive understanding of the border as a set of points at the bound of the region. This definition is sometimes referred to as the **inner border** to distinguish it from the **outer border**, that is, the border of the background (i.e., its complement) of the region.

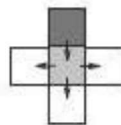


Figure 2.12: Crack edges.
© Cengage Learning 2015.

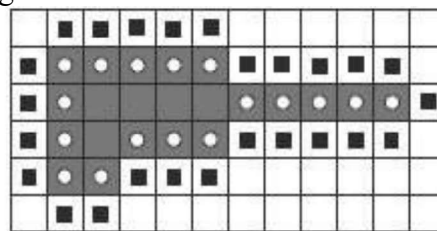


Figure 2.13: Region inner borders shown as white circles and outer borders shown as black squares (using 4-neighborhoods). © Cengage Learning 2015.

A region is described as **convex** if any two points within it are connected by a straight line segment, and the whole line lies within the region—see Figure 2.14. The property of convexity decomposes all regions into two equivalence classes: convex and non-convex.

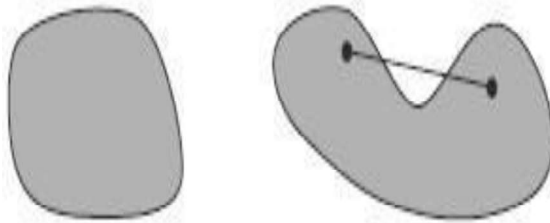


Figure 2.14: A convex region (left) and non-convex region (right). © Cengage Learning 2015.

A **convex hull** of a region is the smallest convex region containing the input (possibly non-convex) region. Consider an object whose shape resembles the letter ‘R’ (see Figure 2.15). Imagine a thin rubber band pulled around the object; the shape of the rubber band provides the convex hull of the object.

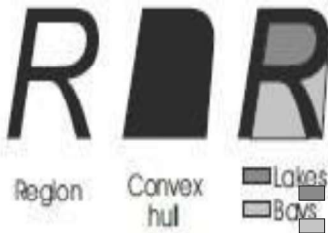


Figure 2.15: Description using topological components: An ‘R’ object, its convex hull, and the associated lakes and bays. © Cengage Learning 2015.

Topological properties are not based on the distance concept. An object with non-regular shape can be represented by a collection of its topological components. The set inside the convex hull which does not belong to an object is called the **deficit of convexity**. This can be split into two subsets: **lakes** (dark gray) are fully surrounded by the object; and **bays** (light gray) are contiguous with the border of the convex hull of the object.

Histograms

The **brightness histogram** $h_f(x)$ of an image provides the frequency of the brightness value x in the image—the histogram of an image with L gray-levels is represented by a one-dimensional array with L elements.

The histogram provides a natural bridge between images and a probabilistic description. The histogram is often displayed as a bar graph. The histogram is usually the only global information about the image which is available. It is used when finding optimal illumination conditions for capturing an image, gray-scale transformations, and image segmentation to objects and background.

The histogram of a digital image typically has many local minima and maxima, which may complicate its further processing. This problem can be avoided by local smoothing of the histogram; this may be done, for example, using local averaging of neighboring histogram elements as the base, so that a new histogram $h'_f(x)$ is calculated according to

$$h'_f(z) = \frac{1}{2K+1} \sum_{j=-K}^K h_f(z+j), \quad (2.6)$$

where K is a constant representing the size of the neighborhood used for smoothing. This algorithm would need some boundary adjustment, and carries no guarantee of removing all local minima.

Entropy

If a probability density p is known then image information content can be estimated regardless of its interpretation using **entropy** H .

Entropy is formally defined assuming a discrete random variable X with possible outcomes (called also states) $x_1, \dots, x_n$. Let $p(x_k)$ be the probability of the outcome x_k , $k = 1, \dots, n$. Then the entropy is defined as

$$H(X) \equiv \sum_{k=1}^n p(x_k) \log_2 \left(\frac{1}{p(x_k)} \right) = - \sum_{k=1}^n p(x_k) \log_2 p(x_k). \quad (2.7)$$

The entropy of the random variable X is the sum, over all possible outcomes k of X , of the product of the probability of outcome x_k with the logarithm of the inverse of the probability of x_k

Visual perception of the image

Anyone who creates or uses algorithms or devices for digital image processing should take into account the principles of human image perception. If an image is to be analyzed by a human the information should be expressed using variables which are easy to perceive; these are psycho-physical parameters such as contrast, border, shape, texture, color, etc. Humans will find objects in images only if they may be distinguished effortlessly from the background.

Contrast

Contrast is the local change in brightness and is defined as the ratio between average brightness of an object and the background. Apparent brightness depends very much on the brightness of the local surroundings; this effect is called conditional contrast.



Figure 2.17: Conditional contrast effect. Circles inside squares have the same brightness and are perceived as having different brightness values.

Acuity

Acuity is the ability to detect details in an image. The human eye is less sensitive to slow and fast changes in brightness in the image plane but is more sensitive to intermediate changes. Acuity also decreases with increasing distance from the optical axis.

Resolution in an image is firmly bounded by the resolution ability of the human eye; there is no sense in representing visual information with higher resolution than that of the viewer. Resolution in optics is defined as the inverse value of a maximum viewing angle between the viewer and two proximate points which humans cannot distinguish, and so fuse together.

Human vision has the best resolution for objects which are at a distance of about 250 mm from an eye under illumination of about 500 lux; this illumination is provided by a 60 W bulb from a distance of 400 mm. Under these conditions the distance between two distinguishable points is approximately 0.16 mm.

Image quality

An image might be degraded during capture, transmission, or processing, and measures of image quality can be used to assess the degree of degradation. The quality required naturally depends on the purpose for which an image is used.

Methods for assessing **image quality** can be divided into two categories: subjective and objective. Subjective methods are often used in television technology, where the ultimate criterion is the perception of a selected group of professional and lay viewers. They appraise an image according to a list of criteria and give appropriate marks.

Objective quantitative methods measuring image quality are more interesting for our purposes. Ideally such a method also provides a good subjective test, and is easy to apply; we might then use it as a criterion in parameter optimization. The quality of an image $f(x, y)$ is usually estimated by comparison with a known reference image $g(x, y)$, and a synthesized image is often used for this purpose. One class of methods uses simple measures such as the mean quadratic difference but this does not distinguish a few big differences from many small differences. Instead of the mean quadratic difference, the mean absolute difference or simply maximal absolute difference may be used. Correlation between images f and g is another alternative.

Another class measures the resolution of small or proximate objects in the image. An image consisting of parallel black and white stripes is used for this purpose; then the number of black and white pairs per millimeter gives the resolution.

Noise in images

Real images are often degraded by some random errors—this degradation is usually called **noise**. Noise can occur during image capture, transmission, or processing, and may be dependent on, or independent of, the image content.

1) white noise

white noise is **a random signal having equal intensity at different frequencies, giving it a constant power spectral density.**

2) Gaussian noise

Gaussian noise, named after Carl Friedrich Gauss, is a kind of signal noise that has a probability density function equal to that of the normal distribution (which is also known as the Gaussian distribution)

3) Additive noise

Additive noise is the undesired abrupt signal that gets added into some genuine signal.

4) Multiplicative noise

Multiplicative noise is the undesired abrupt signal that gets multiplied into some genuine signal.

5) Quantization noise

Quantization noise results when a continuous random variable is converted to a discrete one or when a discrete random variable is converted to one with fewer levels. In images, quantization noise often occurs in the acquisition process.

6) Impulse noise

Impulse noise is a category of noise that includes unwanted, almost instantaneous sharp sounds—typically caused by electromagnetic interference, scratches on disks, gunfire, explosions, and synchronization issues in digital audio.

7) Salt and pepper noise

salt--and--pepper noise describes a situation where random pixels get replaced by extremely dark or bright values.

COLOR IMAGES

Color may be considered a psycho-physical phenomenon.

Physics of color

The electromagnetic spectrum is illustrated in Figure 2.23.

Only a narrow section of the electromagnetic spectrum is visible to a human, with wavelength from approximately 380 nm to 740 nm. Visible colors with the wavelengths shown in Figure 2.24 are called **spectral colors**. Colors can be represented as combinations of the **primary colors**, e.g., red,

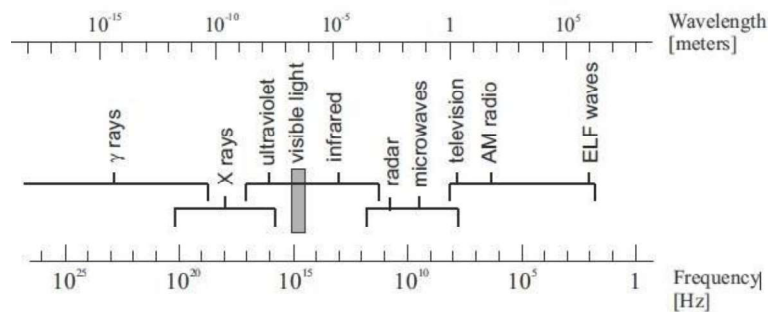


Figure 2.23: Division of the electromagnetic spectrum (ELF is Extremely Low Frequencies).

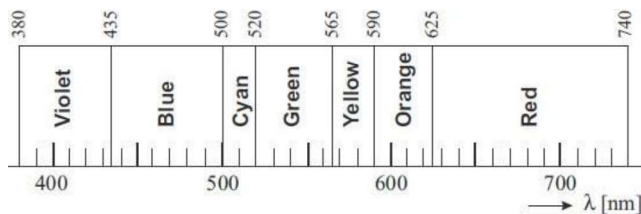


Figure 2.24: Wavelength λ of the spectrum visible to humans.

green, and blue, which for the purposes of standardization have been defined as 700 nm, 546.1 nm, and 435.8 nm, respectively.

The intensity of irradiation for different wavelengths λ usually changes. This variation is expressed by a **power spectrum** (called also power spectrum distribution) $S(\lambda)$. There are two predominant physical mechanisms describing what happens when a surface is irradiated. First, the **surface reflection** rebounds incoming energy in a similar way to a mirror.. Second, the energy diffuses into the material and reflects randomly from the internal pigment in the matter. This mechanism is called **body reflection**.

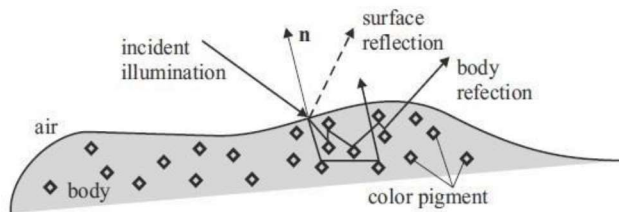


Figure 2.25: Observed color of objects is caused by certain wavelength absorptions by pigment particles in dielectrics.

Each spectral band is digitized independently and is represented by an individual digital image function as if it were a monochromatic image. In this way, **multispectral images** are created. Multispectral images are commonly used in remote sensing from satellites, airborne sensors and in industry.

Color perceived by humans

Color sensitive receptors on the human retina are the **cones**. The other light sensitive receptors on the retina are the **rods**. Cones are categorized into three types based on the sensed wavelength range: S (short) $\approx$ with maximum sensitivity $\approx$ at 430 nm, M (medium) at 560 nm, and L (long) at 610 nm. Cones S, M, L are occasionally called cones B, G and R, respectively.

The reaction of a photoreceptor or output from a sensor in a camera can be modeled mathematically. Let i be the specific type of sensor, $i = 1, 2, 3$, (the retinal cone type S, M, L in the human case). Let $R_i(\lambda)$ be the spectral sensitivity of the sensor, $I(\lambda)$ be the spectral density of the illumination, and $S(\lambda)$ describe how the surface patch reflects each wavelength of the illuminating light. The spectral response q_i of the i -th sensor, can be modeled by integration over a certain range of wavelengths

A phenomenon called **color metamer** is relevant. A metamer, in general, means two things that are

$$q_i = \int_{\lambda_1}^{\lambda_2} I(\lambda) R_i(\lambda) S(\lambda) d\lambda. \quad (2.16)$$

physically different but perceived as the same. Red and green adding to produce yellow is a color metamer, because yellow could have also been produced by a spectral color. The human visual system is fooled into perceiving that red and green is the same as yellow.

The **color model** was introduced as a mathematical abstraction allowing us to express colors as tuples of numbers, typically as three or four values of color component called **XYZ color space**.

The standard is given by the three imaginary lights $X=700.0$ nm, $Y=546.1$ nm, $Z=435.8$ nm and by the color matching functions $X(\lambda)$, $Y(\lambda)$ and $Z(\lambda)$ corresponding to the perceptual ability of an average human viewing a screen through an aperture providing a 2° field of view.

The XYZ color standard fulfills three requirements:

- Unlike the color matching experiment yielding negative lobes of color matching functions, the matching functions of XYZ space are required to be non-negative.
- The value of $Y(\lambda)$ should coincide with the brightness (luminance).
- Normalization is performed to assure that the power corresponding to the three color matching functions is equal (i.e., the area under all three curves is equal). The resulting color matching functions are shown in Figure 2.28. The actual color is a mixture (more precisely a convex combination) of $c_X X + c_Y Y + c_Z Z$, (2.17)

where $0 \leq c_X, c_Y, c_Z \leq 1$ are weights (intensities) in the mixture. The subspace of colors perceivable by humans is called the color gamut and is demonstrated in Figure 2.29.

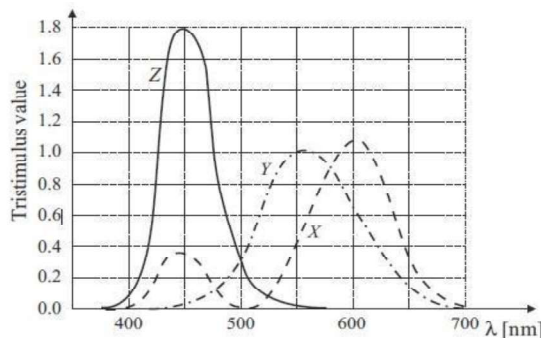


Figure 2.28: Color matching functions for the CIE standard from 1931. $X(\lambda)$, $Y(\lambda)$, $Z(\lambda)$ are color matching functions. Based on [Wandell, 1995].

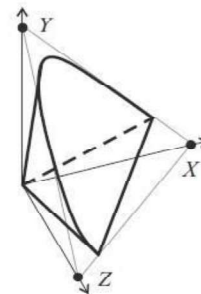


Figure 2.29: Color gamut - a subspace of the X, Y, Z color space showing all colors perceivable by humans. © Cen-

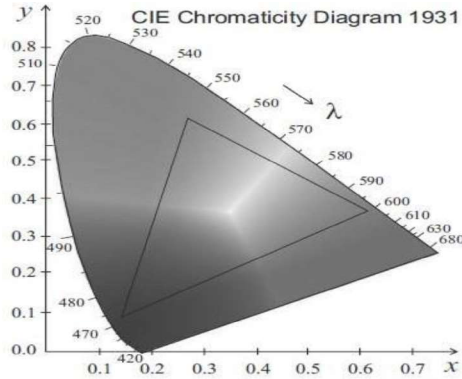


Figure 2.30: CIE chromaticity diagram is a projection of XYZ color space into a plane. The triangle depicts a subset of colors spanned by red, green, and blue. These are TV colors, i.e., all possible colors, which can be seen on a CRT display. © Cengage Learning 2015. A color version of this figure may be seen in the color inset—Plate 1.

3D figures are difficult to represent, and so a planar view of a 3D color space is used. The projection plane is given by the plane passing through extremal points on all three axes, i.e., points X , Y , Z . The new 2D coordinates x , y are obtained as

$$x = \frac{X}{X+Y+Z}, \quad y = \frac{Y}{X+Y+Z}, \quad z = 1 - x - y.$$

The result of this plane projection is the CIE chromaticity diagram, see Figure 2.30. The horseshoe like subspace contains colors which people are able to see. All mono-chromatic spectra visible to humans map into the curved part of the horseshoe—their wavelengths are shown in Figure 2.30.

Color spaces

The value of a particular color is expressed as a vector of three elements—intensities of three primary colors and a transformation to a different color space is expressed by a 3×3 matrix. Assume that values for each primary are quantized to $m = 2^n$ values; let the highest – intensity value be $k = m - 1$; then $(0, 0, 0)$ is black, (k, k, k) is (television) white, $(k, 0, 0)$ is ‘pure’ red, and so on. The value $k = 255 = 2^8 - 1$ is common, i.e., 8 bits per color channel. There are $256^3 = 2^{24} = 16, 777, 216$ possible colors in such a discretized space.

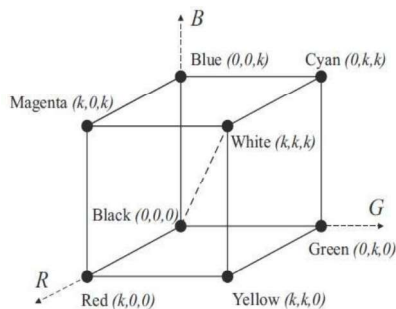


Figure 2.32: RGB color space with primary colors red, green, blue and secondary colors yellow, cyan, magenta.

The RGB model may be thought of as a 3D co-coordinatization of color space. One of the transformations between RGB and XYZ color spaces is

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 3.24 & -1.54 & -0.50 \\ -0.98 & 1.88 & 0.04 \\ 0.06 & -0.20 & 1.06 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix},$$

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.41 & 0.36 & 0.18 \\ 0.21 & 0.72 & 0.07 \\ 0.02 & 0.12 & 0.95 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}. \quad (2.18)$$

The US and Japanese color television formerly used **YIQ** color space. The Y component describes intensity and I, Q represent color. YIQ is another example of additive color mixing. This system stores a luminance value with two chrominance values, corresponding approximately to the amounts of blue and red in the color.

The **CMY**—for Cyan, Magenta, Yellow—color model uses subtractive color mixing which is used in printing processes. It describes what kind of inks need to be applied so the light reflected from the white substrate (paper, painter's canvas) and passing through the inks produces a given color. CMYK stores ink values for black in addition. Black can be generated from C, M and Y components but as it is abundant in printed documents, it is of advantage to have a special black ink.

HSV – Hue, Saturation, and Value (also known as HSB, hue, saturation, brightness) is often used by painters because it is closer to their thinking and technique. Artists commonly use three to four dozen colors (characterized by the hue; technically, the dominant wavelength). If another color is to be obtained then it is mixed from the given ones.

Palette images:

Palette images (also called **indexed images**) provide a simple way to reduce the amount of data needed to represent an image. The pixel values constitute a link to a **lookup table** (also called a color table, color map, **palette**). The table contains as many entries as the range of possible values in the pixel, which is typically 8 bits 256 values. Each entry of the table $\equiv$ maps the pixel value to the color, so there are three values, one for each of three color components. In the typical case of the RGB color model, values for red, green and blue are provided. It is easy to see that this approach would reduce data consumption to one-third if each of the RGB channels had originally been using 8 bits (plus size of the look up table).

Many widely used image formats for raster images such as TIFF, PNG and GIF can store palette images.

If the number of colors in the input image is less than or equal to the number of entries in the lookup table then all colors can be selected and no loss of information occurs. Such images may be cartoon movies, or program outputs. In the more common case, the number of colors in the image exceeds the number of entries in the lookup table, a subset of colors has to be chosen, and a loss of information occurs.

This color selection may be done many ways. The simplest is to quantize color space regularly into cubes of the same size. In the 8 bit example, there would be $8 \times 8 \times 8 = 512$ such cubes. **Vector quantization** which is widely used in analyzing large multi-dimensional datasets. It is also possible to view the occupation by the pixels of RGB space. The term **pseudo-color** is usually used when an original image is gray-level and is displayed in color; this is often done to exploit the color discriminatory power of human vision.

DATA STRUCTURES FOR IMAGE ANALYSIS

Data and an algorithm are the two essentials of any program. Data organization often considerably affects the simplicity of the selection and the implementation of an algorithm, and the choice of data structures is therefore a fundamental question when writing a program.

LEVELS OF IMAGE DATA REPRESENTATION

The aim of computer visual perception is to find a relation between an input image and models of the real world. Several levels of visual information representation are defined on the way between the input image and the model; computer vision then comprises a design of the:

- **Intermediate representations** (data structures).
- **Algorithms** used for the creation of representations and introduction of relations between them.

The representations can be stratified in four levels. The information flow between the levels may be bi-directional, and for some specific uses, some representations can be omitted.

- 1) The lowest representational level—**iconic images**—consists of images containing original data: integer matrices with data about pixel brightness. I
- 2) The second level is **segmented images**. Parts of the image are joined into groups that probably belong to the same objects.
- 3) The third level is **geometric representations** holding knowledge about 2D and 3D shapes.
- 4) The fourth representational level is **relational models**. They give us the ability to treat data more efficiently and at a higher level of abstraction.

TRADITIONAL IMAGE DATA STRUCTURES

Traditional image data structures such as matrices, chains, graphs, lists of object properties, and relational databases are important not only for the direct representation of image information, but also as a basis for more complex hierarchical methods of image representation.

Matrices

A **matrix** is the most common data structure for low-level representation of an image. Elements of the matrix are integer numbers corresponding to brightness of the corresponding pixel of the sampling grid. Image data of this kind are usually the direct output of the image-capturing device.

Image information in the matrix is accessible through the co-ordinates of a pixel that correspond with row and column indices. The matrix is a full representation of the image, independent of the contents of image data—it implicitly contains **spatial relations** among semantically important parts of the image. One very natural spatial relation is the **neighborhood relation**. Some special images that are represented by matrices are:

- A **binary image** (an image with two brightness levels only) is represented by a matrix containing only zeros and ones.
- Several matrices can contain information about one **multispectral image**. Each single matrix contains one image corresponding to one spectral band.
- Matrices of different resolution are used to obtain **hierarchical image data structures**. Such hierarchical representations can be very convenient for parallel computers with the ‘processor array’ architecture.

Most programming languages use a standard array data structure to represent a matrix. There is much image data in the matrix. Algorithms can be speed up if global information is derived from the original image matrix first—global information is more concise and occupies less memory.

The **integral image** is matrix representation that holds global image information. It is constructed so that its values $ii(i, j)$ in the location (i, j) represent the sums of all the original image pixel-values left of and above (i, j) :

$$ii(i, j) = \sum_{k \leq i, l \leq j} f(k, l), \quad (4.1)$$

where f is the original image. The integral image can be efficiently computed in a single image pass using recurrences. The main use of integral image data structures is in rapid calculation of simple rectangle image features at multiple scales. This kind of features is used for rapid object identification and for object tracking.

Figure 4.1 illustrates that any rectangular sum can be computed using four array references, and so a feature reflecting a difference between two rectangles requires eight references.

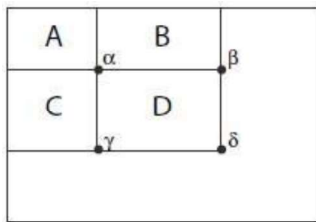


Figure:4.1

The sum of pixels within rectangle D can be obtained using four array references.

$Dsum = ii(\delta) + ii(\alpha) - (ii(\beta) + ii(\gamma))$, where $ii(\alpha)$ is the value of the integral image at point α (and similarly for β, γ, δ).

Chains

Chains are used for the description of object borders in computer vision. One element of the chain is a basic symbol. Chains are appropriate for data that can be arranged as a sequence of symbols, and the neighboring symbols in a chain usually correspond to the neighborhood of primitives in the image. The primitive is the basic descriptive element that is used in syntactic pattern recognition.

This rule of proximity (neighborhood) of symbols and primitives has exceptions—for example, the first and the last symbol of the chain describing a closed border are not neighbors, but the corresponding primitives in the image are.

Chain codes are often used for the description of object borders, or other one-pixel-wide lines in images. The border is defined by the co-ordinates of its reference pixel and the sequence of symbols corresponding to the line of the unit length in several pre-defined orientations.

Run length coding has been used for some time to represent strings of symbols in an image matrix. Run length coding records only areas that belong to objects in the image; the area is then represented as a list of lists. A representative one describes each row of the image by a sublist, the first element of which is the row number. Subsequent terms are coordinate pairs; the first element of a pair is the beginning of a run and the second is the end (the beginning and the end are described by column coordinates).

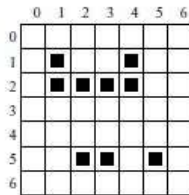


Figure 4.4: Run length coding; the code is ((1114)(214)(52355)).

Topological data structures

Topological data structures describe the image as a set of elements and their relations; these relations are often represented using graphs. A **graph** $G = (V, E)$ is an algebraic structure which consists of a set of nodes

$V = \{v_1, v_2, \dots, v_n\}$ and a set of arcs $E = \{e_1, e_2, \dots, e_m\}$. Each arc e_k is incident to an unordered (or ordered) pair of nodes $\{v_i, v_j\}$ which are not necessarily distinct. The *degree* of a node is equal to the number of incident arcs of the node.

A **weighted graph** is a graph in which values are assigned to arcs, to nodes, or to both—these values may, for example, represent weights, or costs.

The **region adjacency graph** is typical of this class of data structures, in which nodes correspond to regions and neighboring regions are connected by an arc. The segmented image consists of regions with similar properties (brightness, texture, color, . . .) that correspond to some entities in the scene, and the neighborhood relation is fulfilled when the regions have some common border.

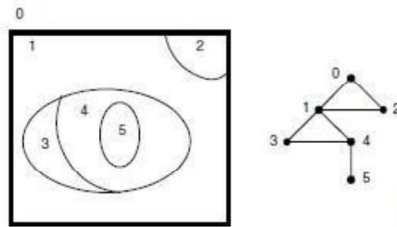


Figure 4.5: An example region adjacency graph.
© Cengage Learning 2015.

The region adjacency graph is usually created from the **region map**, which is a matrix of the same dimensions as the original image matrix whose elements are identification labels of the regions.

Relational structures

Relational databases can also be used for representation of information from an image; all the information is then concentrated in relations between semantically important parts of the image—objects—that are the result of segmentation. Relations are recorded in the form of tables.

HIERARCHICAL DATA STRUCTURES

Computer vision is by its nature very computationally expensive, if for no other reason than the large amount of data to be processed. We are going to introduce two typical hierarchical structures, pyramids and quadtrees.

Pyramids

Pyramids are among the simplest hierarchical data structures. We distinguish between **M-pyramids** (matrix-pyramids) and **T-pyramids** (tree-pyramids).

A **Matrix-pyramid** (M-pyramid) is a sequence $\{M_L, M_{L-1}, \dots, M_0\}$ of images, where M_L has the same dimensions and elements as the original image, and M_{i-1} is derived from the M_i by reducing the resolution by one-half. When creating pyramids, it is customary to work with square matrices having dimensions equal to powers of 2—then M_0 corresponds to one pixel only.

M-pyramids are used when it is necessary to work with an image at different resolutions simultaneously. Often it is advantageous to use several resolutions simultaneously rather than choose just one image from the M-pyramid. For such algorithms we prefer to use **tree-pyramids**, a tree structure.

Let $2L$ be the size of an original image (the highest resolution). A tree-pyramid (T-pyramid) is defined by:

1. A set of nodes $P = \{p = (k, i, j) \text{ such that level } k \in [0, L]; i, j \in [0, 2^k - 1]\}$
2. A mapping F between subsequent nodes P_{k-1}, P_k of the pyramid

$$F(k, i, j) = (k-1, \text{floor}(i/2), \text{floor}(j/2)).$$

3. A function V that maps a node of the pyramid P to Z , where Z is the subset of the whole numbers corresponding to the number of brightness levels, for example, $Z = \{0, 1, 2, \dots, 255\}$.

Nodes of a T-pyramid correspond for a given k with image points of an M-pyramid; elements of the set of nodes $P = \{(k, i, j)\}$ correspond with individual matrices in the Mpyramid— k is called the level of the pyramid. An image $P = \{(k, i, j)\}$ for a specific

k constitutes an image at the k^{th} level of the pyramid. F is the so-called parent mapping, which is defined for all nodes P_k of the T-pyramid except its root $(0, 0, 0)$. Every node of the T-pyramid has four child nodes except leaf nodes, which are nodes of level L that correspond to the individual pixels in the image.

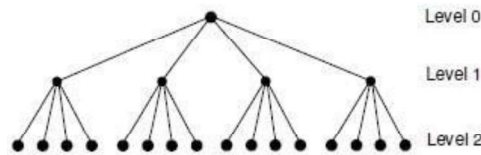


Figure 4.8: T-pyramid.
© Cengage Learning 2015.

Values of individual nodes of the T-pyramid are defined by the function V . Values of leaf nodes are the same as values of the image function (brightness) in the original image at the finest resolution; the image size is $2L$. Values of nodes

in other levels of the tree are either an arithmetic mean of four child nodes or they are defined by coarser sampling, meaning that the value of one child (e.g., top left) is used. Figure 4.8 shows the structure of a simple T-pyramid.

The number of image pixels used by an M-pyramid for storing all matrices is given by

$$N^2 \left(1 + \frac{1}{4} + \frac{1}{16} + \dots \right) \approx 1.33 N^2, \quad (4.4)$$

where N is the dimension of the original matrix (the image of finest resolution)—usually a power of two, $2L$.

The T-pyramid is represented in memory similarly. Arcs of the tree need not be recorded because addresses of the both child and parent nodes are easy to compute due to the regularity of the structure. An algorithm for the effective creation and storing of a T-pyramid is given in [Pavlidis, 1982].

Quadrees

Quadrees are modifications of T-pyramids. Every node of the tree except the leaves has four children (NW, north-western; NE, north-eastern; SW, south-western; SE, south-eastern). Similarly to T-pyramids, the image is divided into four quadrants at each hierarchical level; however, it is not necessary to keep nodes at all levels. If a parent node has four children of the same value (e.g., brightness), it is not necessary to record

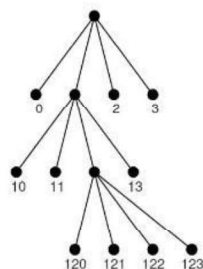
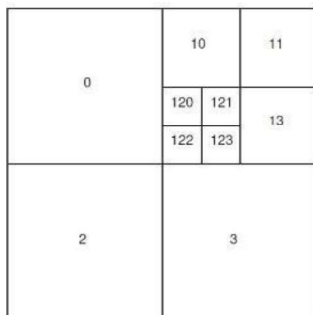


Figure 4.9: Quadtree..

them. This representation is less expensive for an image with large homogeneous regions; Figure 4.9 is an example of a simple quadtree.

An advantage of image representation by means of quadtrees is the existence of simple algorithms for addition of images, computing object areas, and statistical moments. The main disadvantage of quadtrees and pyramid hierarchical representations is their dependence on the position, orientation, and relative size of objects.

These disadvantages can be overcome using a normalized shape of quadtree in which we do not create the quadtree for the whole image, but for its individual objects.

T-pyramids are very similar to quadtrees, but differ in two basic respects. A T- pyramid is a balanced structure, meaning that the corresponding tree divides the image regardless of the contents, which is why it is regular and symmetric. A quadtree is not balanced. The other difference is in the interpretation of values of the individual nodes. Quadtrees have seen widespread application, particularly in the area of Geographic Information Systems (GIS) where, along with their three-dimensional generalization *octrees*, they have proved very useful in hierarchical representation of layered data.

Other pyramidal structures

The pyramidal structure is widely used, and has seen several extensions and modifications. Recalling that a (simple) M-pyramid was defined as a sequence of images $\{M_L, M_{L-1}, \dots, M_0\}$ in which M_i is a 2×2 reduction of M_{i+1} , we can define the notion of a **reduction window**; for every cell c of M_i , the reduction window is its set of children in M_{i+1} , $w(c)$. Here, a *cell* is any single element of the image M_i at the corresponding level of pyramidal resolution. If the images are constructed such that all interior cells have the same number of neighbors (e.g., a square grid, as is customary), and they all have the same number of children, the pyramid is called **regular**.

A taxonomy of regular pyramids may be constructed by considering the reduction window together with the **reduction factor** λ , which defines the rate at which the image area decreases between levels;

$$\lambda \leq \frac{|M_{i+1}|}{|M_i|}, i = 0, 1, \dots, L - 1.$$

In the simple case, in which reduction windows do not overlap and are 2×2 , we have $\lambda = 4$; if we choose to let the reduction windows overlap, the factor will reduce. The notation used to describe this characterization of regular pyramids is *(reduction window)/(reduction factor)*.

The reduction window of a given cell at level i may be propagated down to higher resolution than level $i + 1$. For a cell c_i at level i , we can write $w^0(c_i) = w(c_i)$, and then recursively define

$$w^{k+1}(c_i) = \bigcup_{q \in w(c_i)} w^k(q), \quad (4.5)$$

$w^k(c_i)$ is the **equivalent window** that covers all cells at level $i+k+1$ that link to the cell c_i . Note that the shape of this window is going to depend on the type of pyramid—for example, an $n \times n/2$ pyramid will generate octagonal equivalent windows, while for an

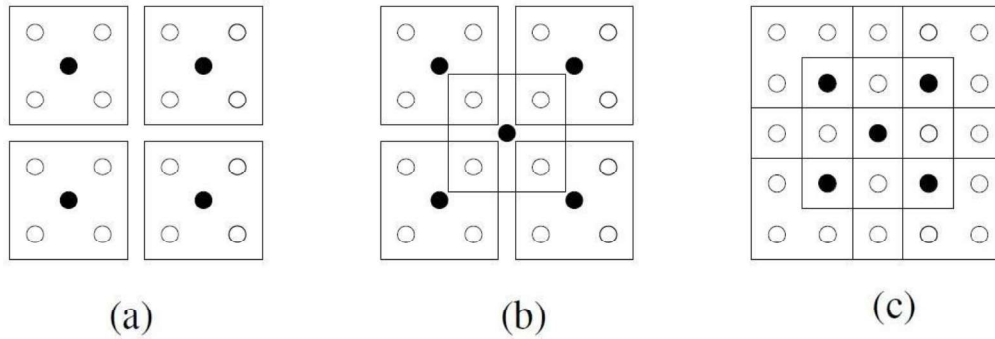


Figure 4.11: Several regular pyramid definitions.

(a) $2 \times 2/4$. (b) $2 \times 2/2$. (c) $3 \times 3/2$. (Solid dots are at the higher level, i.e., the lower resolution level.)

$n \times n/4$ pyramid they will be square. Use of non-square windows prevents domination of square features, as is the case for simple $2 \times 2/4$ pyramids.

The $2 \times 2/4$ pyramid is widely used and is what is usually called an ‘image pyramid’; the $2 \times 2/2$ structure is often referred to as an ‘overlap pyramid’. $5 \times 5/2$ pyramids have been used in compact image coding, where the image pyramid is augmented by a **Laplacian** pyramid of differences. Here, the Laplacian at a given level is computed as the per-pixel difference between the image at that level, and the image derived by ‘expanding’ the image at the next lower resolution. The Laplacian may be expected to have zero (or close) values in areas of low contrast, and therefore be amenable to compression.

Irregular pyramids are derived from contractions of graphical representations of images (for example, region adjacency graphs. Depending on how these selections are made, important structures in the parent graph may be retained while reducing its overall complexity.