4 Perception

One of the most important tasks of autonomous systems of any kind is to acquire knowledge about its environment. This is done by taking measurements using various sensors and then extracting meaningful information from those measurements.

In this chapter we present the most common sensors used in mobile robots and then discuss strategies for extracting information from the sensors. For more detailed information about many of the sensors used on mobile robots, refer to the comprehensive book *Sensors for Mobile Robots* written by H.R. Everett [2].

4.1 Sensors for Mobile Robots

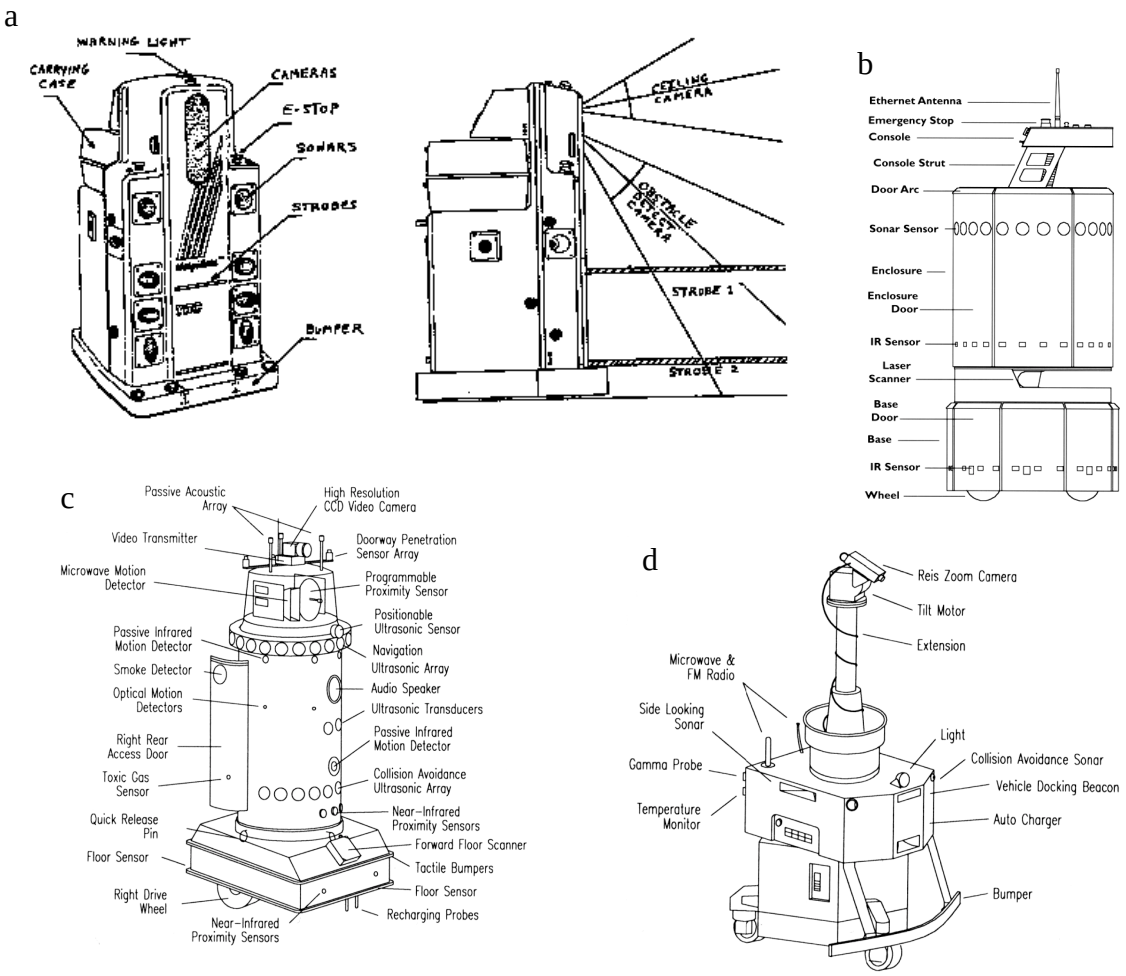


Fig 4.1 *Examples of robots with multi-sensor systems:*
a) HelpMate from Transition Research Corp.
b) B21 from Real World Interface
c) Roboart II, built by H.R. Everett [2]
d) The Savannah River Site nuclear surveillance robot

There is a wide variety of sensors used in mobile robots (Fig. 4.1). Some sensors are used to measure simple values like the internal temperature of a robot's electronics or the rota-

tional speed of the motors. Other, more sophisticated sensors can be used to acquire information about the robot's environment or even to directly measure a robot's global position. In this chapter we focus primarily on sensors used to extract information about the robot's environment. Because a mobile robot moves around, it will frequently encounter unforeseen environmental characteristics, and therefore such sensing is particularly critical. We begin with a functional classification of sensors. Then, after presenting basic tools for describing a sensor's performance, we proceed to describe selected sensors in detail.

4.1.1 Sensor Classification

We classify sensors using two important functional axes: *proprioceptive/exteroceptive* and *passive/active.*

Proprioceptive sensors measure values internal to the system (robot); e.g. motor speed, wheel load, robot arm joint angles, battery voltage.

Exteroceptive sensors acquire information from the robot's environment; e.g. distance measurements, light intensity, sound amplitude. Hence exteroceptive sensor measurements are interpreted by the robot in order to extract meaningful environmental features.

Passive sensors measure ambient environmental energy entering the sensor. Examples of passive sensors include temperature probes, microphones and CCD or CMOS cameras.

Active sensors emit energy into the environment, then measure the environmental reaction. Because active sensors can manage more controlled interactions with the environment, they often achieve superior performance. However, active sensing introduces several risks: the outbound energy may affect the very characteristics that the sensor is attempting to measure. Furthermore, an active sensor may suffer from interference between its signal and those beyond its control. For example, signals emitted by other nearby robots, or similar sensors on the same robot my influence the resulting measurements. Examples of active sensors include wheel quadrature encoders, ultrasonic sensors and laser rangefinders.

Table 4.1 provides a classification of the most useful sensors for mobile robot applications. The sensors types which are ***highlighted*** (bold italic) will be discussed in this chapter.

Table 4.1:

General Classification (typical use)	**Sensor** **Sensor System**	**PC:** Proprioceptive **EC:** Exteroceptive	**P:** Passive **A:** Active
Tactile Sensors (detection of physical contact or closeness; security switches)	Contact switches, bumpers	EC	P
	Optical barriers	EC	A
	Non-contact proximity sensors	EC	A

Table 4.1:

General Classification (typical use)	Sensor Sensor System	PC: Proprioceptive EC: Exteroceptive	P: Passive A: Active
Wheel/motor sensors (wheel/motor speed and position)	Brush Encoders	PC	P
	Potentiometers	PC	P
	Synchros, Resolvers	PC	A
	Optical Encoders	PC	A
	Magnetic Encoders	PC	A
	Inductive Encoders	PC	A
	Capacitive Encoders	PC	A
Heading sensors (orientation of the robot in relation to a fixed reference frame)	***Compass***	EC	P
	Gyroscopes	PC	P
	Inclinometers	EC	P/A
Ground based beacons (localization in a fixed reference frame)	***GPS***	EC	A
	Active optical or RF beacons	EC	A
	Active ultrasonic beacons	EC	A
	Reflective beacons	EC	A
Active ranging (reflectivity, time-of-flight and geometric triangulation)	Reflectivity sensors	EC	A
	Ultrasonic sensor	EC	A
	Laser rangefinder	EC	A
	Optical triangulation (1D)	EC	A
	Structured light (2D)	EC	A
Motion/speed sensors (speed relative to fixed or moving objects)	***Doppler radar***	EC	A
	Doppler sound	EC	A
Vision-based sensors (visual ranging, whole-image analysis, segmentation, object recognition)	***CCD/CMOS camera(s)***	EC	P
	Visual ranging packages		
	Object tracking packages		

The sensor classes in Table (4.1) are arranged in ascending order of complexity and descending order of technological maturity. Tactile sensors and proprioceptive sensors are critical to virtually all mobile robots, and are well understood and easily implemented. Commercial quadrature encoders, for example, may be purchased as part of a gear-motor assembly used in a mobile robot. At the other extreme, visual interpretation by means of one or more CCD/CMOS cameras provides a broad array of potential functionalities, from obstacle avoidance and localization to human face recognition. However, commercially available sensor units that provide visual functionalities are only now beginning to emerge [105, 106].

4.1.2 Characterizing Sensor Performance

The sensors we describe in this chapter vary greatly in their performance characteristics. Some sensors provide extreme accuracy in well-controlled laboratory settings, but are over-

come with error when subjected to real-world environmental variations. Other sensors provide narrow, high precision data in a wide variety settings. In order to quantify such performance characteristics, first we formally define the sensor performance terminology that will be valuable throughout the rest of this chapter.

4.1.2.1 Basic sensor response ratings

A number of sensor characteristics can be rated quantitatively in a laboratory setting. Such performance ratings will necessarily be best-case scenarios when the sensor is placed on a real-world robot, but are nevertheless useful.

Dynamic range is used to measure the spread between the lower and upper limits of inputs values to the sensor while maintaining normal sensor operation. Formally, the dynamic range is the ratio of the maximum input value to the minimum measurable input value. Because this raw ratio can be unwieldy, it is usually measured in *Decibels,* which is computed as ten times the common logarithm of the dynamic range. However, there is potential confusion in the calculation of Decibels, which are meant to measure the ratio between *powers*, such as Watts or Horsepower. Suppose your sensor measures motor current and can register values from a minimum of 1 Milliampere to 20 Amperes. The dynamic range of this current sensor is defined as:

$$10 \cdot \log\left[\frac{20}{0.001}\right] = 43dB \tag{4.1}$$

Now suppose you have a voltage sensor that measures the voltage of your robot's battery, measuring any value from 1 Millivolt to 20 Volts. Voltage is not a unit of power, but the square of voltage is proportional to power. Therefore, we use 20 instead of 10:

$$20 \cdot \log\left[\frac{20}{0.001}\right] = 86dB \tag{4.2}$$

Range is also an important rating in mobile robot applications because often robot sensors operate in environments where they are frequently exposed to input values beyond their working range. In such cases, it is critical to understand how the sensor will respond. For example, an optical rangefinder will have a minimum operating range and can thus provide spurious data when measurements are taken with object closer than that minimum.

Resolution is the minimum difference between two values that can be detected by a sensor. Usually, the lower limit of the dynamic range of a sensor is equal to its resolution. However, in the case of digital sensors, this is not necessarily so. For example, suppose that you have a sensor that measures voltage, performs an analog-to-digital conversion and outputs the converted value as an 8-bit number linearly corresponding to between 0 and 5 Volts. If this sensor is truly linear, then it has $2^8 - 1$ total output values, or a resolution of $\frac{5V}{255} = 20mV$.

Linearity is an important measure governing the behavior of the sensor's output signal as

the input signal varies. A linear response indicates that if two inputs x and y result in the two outputs $f(x)$ and $f(y)$, then for any values a and b, $f(ax + by) = af(x) + bf(y)$. This means that a plot of the sensor's input/output response is simply a straight line.

Bandwidth or ***Frequency*** is used to measure the speed with which a sensor can provide a stream of readings. Formally, the number of measurements per second is defined as the sensor's frequency in *Hertz*. Because of the dynamics of moving through their environment, mobile robots often are limited in maximum speed by the bandwidth of their obstacle detection sensors. Thus increasing the bandwidth of ranging and vision-based sensors has been a high-priority goal in the robotics community.

4.1.2.2 *In Situ* sensor performance

The above sensor characteristics can be reasonably measured in a laboratory environment, with confident extrapolation to performance in real-world deployment. However, a number of important measures cannot be reliably acquired without deep understanding of the complex interaction between all environmental characteristics and the sensors in question. This is most relevant to the most sophisticated sensors, including active ranging sensors and visual interpretation sensors.

Sensitivity itself is a desirable trait. This is a measure of the degree to which an incremental change in the target input signal changes the output signal. Formally, sensitivity is the ratio of output change to input change. Unfortunately, however, the sensitivity of exteroceptive sensors is often confounded by undesirable sensitivity and performance coupling to other environmental parameters.

Cross-sensitivity is the technical term for sensitivity to environmental parameters that are orthogonal to the target parameters for the sensor. For example, a flux-gate compass can demonstrate high sensitivity to magnetic north and is therefore of use for mobile robot navigation. However, the compass will also demonstrate high sensitivity to ferrous building materials, so much so that its cross-sensitivity often makes the sensor useless in some indoor environments. High cross-sensitivity of a sensor is generally undesirable, especially so when it cannot be modeled.

Error of a sensor is defined as the difference between the sensor's output measurements and the true values being measured, within some specific operating context. Given a true value v and a measured value m, we can define ***error*** as: $error = m - v$.

Accuracy is defined as the degree of conformity between the sensor's measurement and the true value, and is often expressed as a proportion of the true value (e.g. 97.5% accuracy):

$$\left(accuracy = 1 - \frac{|m - v|}{v}\right) \tag{4.3}$$

Of course, obtaining the ground truth, v, can be difficult or impossible, and so establishing a confident characterization of sensor accuracy can be problematic. Further, it is important to distinguish between two different sources of error:

Systematic errors are caused by factors or processes that can in theory be modeled. These errors are, therefore, deterministic (i.e. predictable). Poor calibration of a laser rangefinder, unmodeled slope of a hallway floor and a bent stereo camera head due to an earlier collision are all possible causes of systematic sensor errors.

Random errors cannot be predicted using a sophisticated model nor can they be mitigated with more precise sensor machinery. These errors can only be described in probabilistic terms (i.e. stochastically). Hue instability in a color camera, spurious rangefinding errors and black level noise in a camera are all examples of random errors.

Precision is often confused with accuracy, and now we have the tools to clearly distinguish these two terms. Intuitively, high precision relates to reproducibility of the sensor results. For example, one sensor taking multiple readings of the same environmental state has high precision if it produces the same output. In another example, multiple copies of this sensors taking readings of the same environmental state have high precision if their outputs agree. Precision does not, however, have any bearing on the accuracy of the sensor's output with respect to the true value being measured. Suppose that the *random error* of a sensor is characterized by some mean value μ and a standard deviation σ. The formal definition of precision is the ratio of the sensor's output range to the standard deviation:

$$precision = \frac{range}{\sigma} \tag{4.4}$$

Note that only σ and not μ has impact on precision. In contrast mean error μ is directly proportional to overall sensor error and inversely proportional to sensor accuracy.

4.1.2.3 Characterizing error: the challenges in mobile robotics

Mobile robots depend heavily on exteroceptive sensors. Many of these sensors concentrate on a central task for the robot: acquiring information on objects in the robot's immediate vicinity so that it may interpret the state of its surroundings. Of course, these "objects" surrounding the robot are all detected from the viewpoint of its local reference frame. Since the systems we study are mobile, their ever-changing position and their motion has a significant impact on overall sensor behavior. In this section, empowered with the terminology of the last two sections, we describe how dramatically the sensor error of a mobile robot disagrees with the ideal picture drawn in the previous section.

Blurring of systematic and random errors

Active ranging sensors tend to have failure modes that are triggered largely by specific relative positions of the sensor and environment targets. For example, a sonar sensor will product specular reflections, producing grossly inaccurate measurements of range, at specific angles to a smooth sheetrock wall. During motion of the robot, such relative angles occur at stochastic intervals. This is especially true in a mobile robot outfitted with a ring of multiple sonars. The chances of one sonar entering this error mode during robot motion is high. From the perspective of the moving robot, the sonar measurement error is a random error in this case. Yet, if the robot were to stop, becoming motionless, then a very different error

modality is possible. If the robot's static position causes a particular sonar to fail in this manner, the sonar will fail consistently and will tend to return precisely the same (and incorrect!) reading time after time. Once the robot is motionless, the error appears to be systematic and high precision.

The fundamental mechanism at work here is the cross-sensitivity of mobile robot sensors to robot pose and robot-environment dynamics. The models for such cross-sensitivity are not, in an underlying sense, truly random. However, these physical interrelationships are rarely modeled and therefore, from the point of view of an incomplete model, the errors appear random during motion and systematic when the robot is at rest.

Sonar is not the only sensor subject to this blurring of systematic and random error modality. Visual interpretation through the use of a CCD camera is also highly susceptible to robot motion and position because of camera dependency on lighting changes, lighting specularity (e.g. glare) and reflections. The important point is to realize that, while systematic error and random error are well-defined in a controlled setting, the mobile robot can exhibit error characteristics that bridge the gap between deterministic and stochastic error mechanisms.

Multi-modal error distributions

It is common to characterize the behavior of a sensor's random error in terms of a probability distribution over various output values. In general, one knows very little about the causes of random error and therefore several simplifying assumptions are commonly used. For example, we can assume that the error is *zero-mean*, in that it symmetrically generates both positive and negative measurement error. We can go even further and assume that the probability density curve is Gaussian. Although we discuss the mathematics of this in detail in Section 4.2, it is important for now to recognize the fact that one frequently assumes *symmetry* as well as *unimodal distribution*. This means that measuring the correct value is most probable, and any measurement that is further away from the correct value is less likely than any measurement that is closer to the correct value. These are strong assumptions that enable powerful mathematical principles to be applied to mobile robot problems, but it is important to realize how wrong these assumptions usually are.

Consider, for example, the sonar sensor once again. When ranging an object that reflects the sound signal well, the sonar will exhibit high accuracy, and will induce random error based on noise, for example, in the timing circuitry. This portion of its sensor behavior will exhibit error characteristics that are fairly symmetric and unimodal. However, when the sonar sensor is moving through an environment and is sometimes faced with materials that cause coherent reflection rather than returning the sound signal to the sonar sensor, then the sonar will grossly overestimate distance to the object. In such cases, the error will be biased toward positive measurement error and will be far from the correct value. The error is not strictly systematic, and so we are left modeling it as a probability distribution of random error. So the sonar sensor has two separate types of operational modes, one in which the signal does return and some random error is possible, and the second in which the signal returns after a multi-path reflection, and gross overestimation error occurs. The probability distribution could easily be at least bimodal in this case, and since overestimation is more common than underestimation it will also be asymmetric.

As a second example, consider ranging via stereo vision. Once again, we can identify two modes of operation. If the stereo vision system correctly correlates two images, then the resulting random error will be caused by camera noise and will limit the measurement accuracy. But the stereo vision system can also correlate two images *incorrectly*, matching two fence posts for example that are not the same post in the real world. In such a case stereo vision will exhibit gross measurement error, and one can easily imagine such behavior violating both the unimodal and the symmetric assumptions.

The thesis of this section is that sensors in a mobile robot *may* be subject to multiple modes of operation and, when the sensor error is characterized, unimodality and symmetry may be grossly violated. Nonetheless, as you will see, many successful mobile robot systems make use of these simplifing assumptions and the resulting mathematical techniques with great empirical success.

The above sections have presented a terminology with which we can characterize the advantages and disadvantages of various mobile robot sensors. In the following sections, we do the same for a sampling of the most commonly used mobile robot sensors today.

4.1.3 Wheel/motor sensors

Wheel/motor sensors are devices use to measure the internal state and dynamics of a mobile robot. These sensors have vast applications outside of mobile robotics and, as a result, mobile robotics has enjoyed the benefits of high-quality, low-cost wheel and motor sensors that offer excellent resolution. In the next subsection, we sample just one such sensor, the optical incremental encoder.

4.1.3.1 Optical Encoders

Optical incremental encoders have become the most popular device for measuring angular speed and position within a motor drive or at the shaft of a wheel or steering mechanism. In mobile robotics, encoders are used to control the position or speed of wheels and other motor-driven joints. Because these sensors are *proprioceptive*, their estimate of position is best in the reference frame of the robot and, when applied to the problem of robot *localization*, significant corrections are required as discussed in Chapter 5.

An optical encoder is basically a mechanical light chopper that produces a certain number of sine or square wave pulses for each shaft revolution. It consists of an illumination source, a fixed grating that masks the light, a rotor disc with a fine optical grid that rotates with the shaft, and fixed optical detectors. As the rotor moves, the amount of light striking the optical detectors varies based on the alignment of the fixed and moving gratings. In robotics, the resulting sine wave is transformed into a discrete square wave using a threshold to choose between *light* and *dark* states. Resolution is measured in *Cycles Per Revolution* (CPR). The minimum angular resolution can be readily computed from an encoder's CPR rating. A typical encoder in mobile robotics may have 2,000 CPR while the optical encoder industry can readily manufacture encodres with 10,000 CPR. In terms of required bandwidth, it is of course critical that the encoder be sufficiently fast to count at the shaft spin speeds that are expected. Industrial optical encoders present no bandwidth limitation to mobile robot appli-

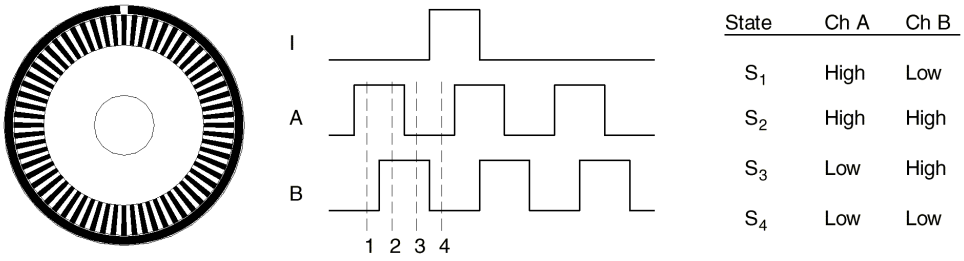


Fig 4.2 *Quadrature optical wheel encoder: The observed phase relationship between channel A and B pulse trains are used to determine the direction of the rotation. A single slot in the outer track generates a reference (index) pulse per revolution.*

cations.

Usually in mobile robotics the *quadrature encoder* is used. In this case, a second illumination and detector pair is placed 90° shifted with respect to the original in terms of the rotor disc. The resulting twin square waves, shown in Fig. 4.2, provide significantly more information. The ordering of which square wave produces a rising edge first identifies the direction of rotation. Furthermore, the four detectably different states improve the resolution by a factor of four with no change to the rotor disc. Thus, a 2,000 CPR encoder in quadrature yields 8,000 counts. Further improvement is possible by retaining the sinusoidal wave measured by the optical detectors and performing sophisticated interpolation. Such methods, although rare in mobile robotics, can yield 1000-fold improvements in resolution.

As with most proprioceptive sensors, encoders are generally in the controlled environment of a mobile robot's internal structure, and so systematic error and cross-sensitivity can be engineered away. The accuracy of optical encoders is often assumed to be 100% and, although this may not entirely correct, any errors at the level of an optical encoder are dwarfed by errors downstream of the motor shaft.

4.1.4 Heading Sensors

Heading sensors can be *proprioceptive* (gyroscope, inclinometer) or *exteroceptive* (compass). They are used to determine the robots orientation and inclination. They allow us, together with appropriate velocity information, to integrate the movement to a position estimate. This procedure, which has its roots in vessel and ship navigation, is called *dead reckoning.*

4.1.4.1 Compasses

The two most common modern sensors for measuring the direction of a magnetic field are the Hall Effect and Flux Gate compasses. Each has advantages and disadvantages, as described below.

The Hall Effect describes the behavior of electric potential in a semiconductor when in the presence of a magnetic field. When a constant current is applied across the length of a semiconductor, there will be a voltage difference in the perpendicular direction, across the semiconductor's width, based on the relative orientation of the semiconductor to magnetic flux

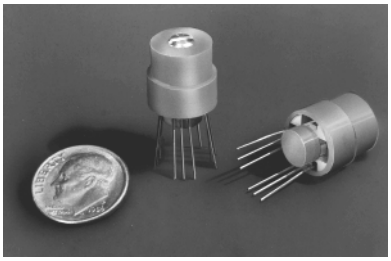


Fig 4.3 *Digital compasses: Sensors such as the Digital/Analog hall effect sensors shown, available from Dinsmore [http://dinsmoregroup.com/dico], enable inexpensive (< $US15) sensing of magnetic fields.*

lines. In addition, the sign of the voltage potential identifies the direction of the magnetic field. Thus, a single semiconductor provides a measurement of flux and direction along one dimension. Hall Effect digital compasses are popular in mobile robotics, and contain two such semiconductors at right angles, providing two axes of magnetic field (thresholded) direction, thereby yielding one of 8 possible compass directions. The instruments are inexpensive but also suffer from a range of disadvantages. Resolution of a digital hall effect compass is poor. Internal sources of error include the nonlinearity of the basic sensor and systematic bias errors at the semiconductor level. The resulting circuitry must perform significant filtering, and this lowers the bandwidth of hall effect compasses to values that are slow in mobile robot terms. For example the hall effect compasses pictured in figure 4.3 needs 2.5 seconds to settle after a 90° spin.

The Flux Gate compass operates on a different principle. Two small coils are wound on ferrite cores and are fixed perpendicular to one-another. When alternating current is activated in both coils, the magnetic field causes shifts in the phase depending upon its relative alignment with each coil. By measuring both phase shifts, the direction of the magnetic field in two dimensions can be computed. The flux-gate compass can accurately measure the strength of a magnetic field and has improved resolution and accuracy; however it is both larger and more expensive than a Hall Effect compass.

Regardless of the type of compass used, a major drawback concerning the use of the Earth's magnetic field for mobile robot applications involves disturbance of that magnetic field by other magnetic objects and man-made structures, as well as the bandwidth limitations of electronic compasses and their susceptibility to vibration. Particularly in indoor environments mobile robotics applications have often avoided the use of compasses, although a compass can conceivably provide useful *local* orientation information indoors, even in the precense of steel structures.

4.1.4.2 Gyroscope

Gyroscopes are heading sensors which preserve their orientation in relation to a fixed reference frame. Thus they provide an absolute measure for the heading of a mobile system. Gyroscopes can be classified in two categories, mechanical gyroscopes and optical gyroscopes.

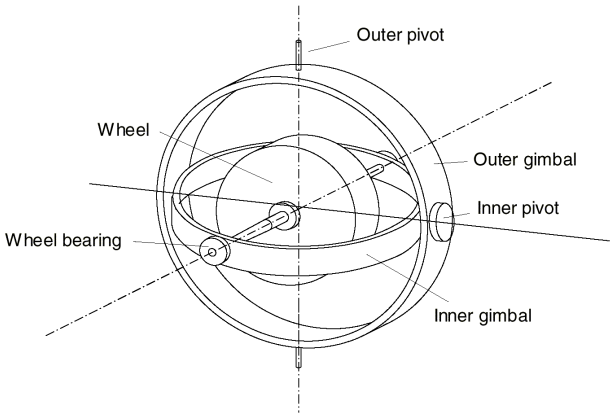


Fig 4.4 Two axis mechanical gyroscope

Mechanical Gyroscopes

The concept of a mechanical gyroscope relies on the inertial properties of a fast spinning rotor. The property of interest is known as the gyroscopic precession. If you try to rotate a fast spinning wheel around its vertical axis, you will feel a harsh reaction in the horizontal axis. This is due to the angular momentum associated with a spinning wheel and will keep the axis of the gyroscope inertially stable. The reactive torque τ and thus the tracking stability with the inertial frame are proportional to the spinning speed ω, the precession speed Ω and the wheel's inertia I.

$$\tau = I\omega\Omega \tag{4.5}$$

By arranging a spinning wheel as seen in Figure 4.4, no torque can be transmitted from the outer pivot to the wheel axis. The spinning axis will therefore be space-stable (i.e. fixed in an inertial reference frame). Nevertheless, the remaining friction in the bearings of the gyro-axis introduce small torques, thus limiting the long term space stability and introducing small errors over time. A high quality mechanical gyroscope can cost up to $100,000 and has an angular drift of about 0.1° in 6 hours.

For navigation, the spinning axis has to be initially selected. If the spinning axis is aligned with the north-south meridian, the earth's rotation has no effect on the gyro's horizontal axis. If it points east-west, the horizontal axis reads the earth rotation.

Rate gyros have the same basic arrangement as shown in Figure 4.4 but with a slight modification. The gimbals are restrained by a torsional spring with additional viscous damping. This enables the sensor to measure angular speeds instead of absolute orientation.

Optical Gyroscopes

Optical gyroscopes are a relatively new innovation. Commercial use began in the early 1980's when they were first installed in aircraft. Optical gyroscopes are angular speed sensors that use two monochromatic light beams, or lasers, emitted from the same source instead of moving, mechanical parts. They work on the principle that the speed of light

remains unchanged and, therefore, geometric change can cause light to take a varying amount of time to reach its destination. One laser beam is sent traveling clockwise through a fiber while the other travels counterclockwise. Because the laser traveling in the direction of rotation has a slightly shorter path, it will have a higher frequency. The difference in frequency Δf of the two beams is a proportional to the angular velocity $\Omega$of the cylinder. New solid-state optical gyroscopes based on the same principle are build using microfabrication technology, thereby providing heading information with resolution and bandwidth far beyond the needs of mobile robotic applications. Bandwidth, for instance, can easily exceed 100KHz while resolution can be smaller than 0.0001°/hr.

4.1.5 Ground-Based Beacons

One elegant approach to solving the localization problem in mobile robotics is to use active or passive beacons. Using the interaction of on-board sensors and the environmental beacons, the robot can identify its position precisely. Although the general intuition is identical to that of early human navigation beacons, such as stars, mountains and lighthouses, modern technology has enabled sensors to localize an outdoor robot with accuracies of better than 5 cm within areas that are kilometers in size.

In the following subsection, we describe one such beacon system, the Global Positioning System (GPS), which is extremely effective for outdoor ground-based and flying robots. Indoor beacon systems have been generally less successful for a number of reasons. The expense of environmental modification in an indoor setting is not amortized over an extremely large useful area, as it is for example in the case of GPS. Furthermore, indoor environments offer significant challenges not seen outdoors, including multipath and environment dynamics. A laser-based indoor beacon system, for example, must disambiguate the one true laser signal from possibly tens of other powerful signals that have reflected off of walls, smooth floors and doors. Confounding this, humans and other obstacles may be constantly changing the environment, for example occluding the one true path from the beacon to the robot. In commercial applications such as manufacturing plants, the environment can be carefully controlled to ensure success. In less structured indoor settings, beacons have nonetheless been used, and the problems are mitigated by careful beacon placement and the useful of passive sensing modalities.

4.1.5.1 The Global Positioning System

The GPS was initially developed for military use but is now freely available for civilian navigation. There are at least 24 operational GPS satellites at all times. The satellites orbit every 12 hours at a height of 20.190km. Four satellites are located in each of six planes inclined 55° with respect to the plane of the earth's equator (figure 4.5).

Each satellite continuously transmits data that indicates its location and the current time. Therefore, GPS receivers are completely passive but exteroceptive sensors. The GPS satellites synchronize their transmissions so that their signals are sent at the same time. When a GPS receiver reads the transmission of two or more satellites, the arrival time differences inform the receiver as to its relative distance to each satellite. By combining information

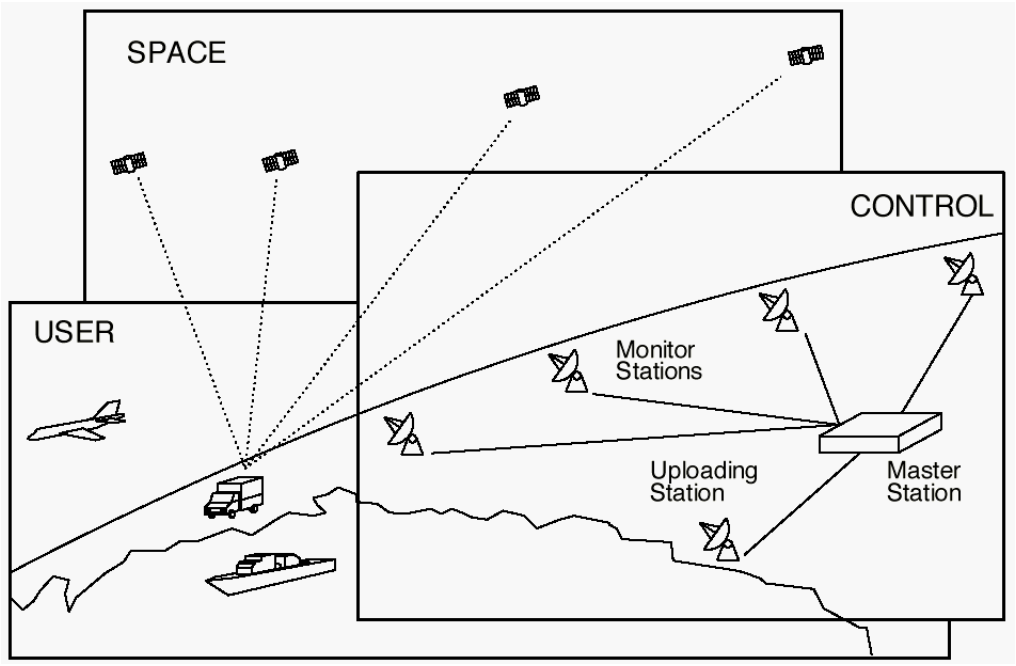


Fig 4.5 Calculation of position and heading based on GPS

regarding the arrival time and instantaneous location of four satellites, the receiver can infer its own position. In theory, such triangulation requires only three data points. However, timing is extremely critical in the GPS application because the time intervals being measured are in the nanoseconds. It is, of course, mandatory that the satellites be well synchronized. To this end, they are updated by ground stations regularly and each satellite carries on-board atomic clocks for timing.

The GPS receiver clock is also important so that the travel time of each satellite's transmission can be accurately measured. But GPS receivers have a simple quartz clock. So, although 3 satellites would ideally provide position in three axes, the GPS receiver requires 4 satellites, using the additional information to solve for 4 variables: three position axes plus a time correction.

The fact that the GPS receiver must read the transmission of 4 satellites simultaneously is a significant limitation. GPS satellite transmissions are extremely low-power, and reading them successfully requires direct line-of-sight communcation with the satellite. Thus, in confined spaces such as city blocks with tall buildings or dense forests, one is unlikely to receive 4 satellites reliably. Of course, most indoor spaces will also fail to provide sufficient visibility of the sky for a GPS receiver to function. For these reasons, GPS has been a popular sensor in mobile robotics, but has been relegated to projects involving mobile robot traversal of wide-open spaces and autonomous flying machines.

A number of factors affect the performance of a localization sensor that makes use of GPS. First, it is important to understand that, because of the specific orbital paths of the GPS satellites, coverage is not geometrically identical in different portions of the Earth and therefore resolution is not uniform. Specifically, at the North and South poles, the satellites are very close to the horizon and, thus, while resolution in the latitude and longitude directions is good, resolution of altitude is relatively poor as compared to more equatorial locations.

The second point is that GPS satellites are merely an information source. They can be employed with various strategies in order to achieve dramatically different levels of localization resolution. The basic strategy for GPS use, called *pseudorange* and described above, generally performs at a resolution of 15m. An extension of this method is *differential GPS*, which makes use of a second receiver that is static and at a known exact position. A number of errors can be corrected using this reference, and so resolution improves to the order of 1m or less. A disadvantage of this technique is that the stationary receiver must be installed, its location must be measured very carefully and of course the moving robot must be within kilometers of this static unit in order to benefit from the *DGPS* technique.

A further improved strategy is to take into account the phase of the carrier signals of each received satellite transmission. There are two carriers, at 19cm and 24cm, therefore significant improvements in precision are possible when the phase difference between multiple satellites is measured successfully. Such receivers can achieve 1cm resolution for point positions and, with the use of multiple receivers as in *DGPS*, sub-1cm resolution.

A final consideration for mobile robot applications is bandwidth. GPS will generally offer no better than 200 - 300ms latency, and so one can expect no better than 5Hz GPS updates. On a fast-moving mobile robot or flying robot, this can mean that local motion integration will be required for proper control due to GPS latency limitations.

4.1.6 Active Ranging

Active range sensors continue to be the most popular sensors in mobile robotics. Many ranging sensors have a low price point, and most importantly all ranging sensors provide easily interpreted outputs: direct measurements of distance from the robot to objects in its vicinity. For obstacle detection and avoidance, most mobile robots rely heavily on active ranging sensors. But the local freespace information provided by range sensors can also be accumulated into representations beyond the robot's current local reference frame. Thus active range sensors are also commonly found as part of the localization and environmental modeling processes of mobile robots. It is only with the slow advent of successful visual interpretation competency that we can expect the class of active ranging sensors to gradually lose their primacy as the sensor class of choice among mobile roboticists.

Below, we present two *time-of-flight* active range sensors: the ultrasonic sensor and the laser rangefinder. Then, we present two geometric active range sensors: the optical triangulation sensor and the structured light sensor.

4.1.6.1 Time-of-Flight active ranging

Time-of-flight ranging makes use of the propagation speed of sound or an electromagnetic wave. In general, the travel distance of a sound of electromagnetic wave is given by:

$$d = c \cdot t \tag{4.6}$$

where

d = distance traveled (usually round-trip)

c = speed of wave propagation

t = time of flight.

It is important to point out that the propagation speed v of sound is approximately 0.3 m/ms whereas the speed of electromagnetic signals are 0.3 m/ns, which is one million times faster. The time of flight for a typical distance, say 3 meters, is 10 ms for an ultrasonic system but only 10 ns for a laser rangefinder. It is thus evident that measuring the time of flight t with electromagnetic signals is more technologically challenging. This explains why laser range sensors have only recently become affordable and robust for use on mobile robots.

The quality of time-of-flight range sensors depends mainly on:

- Uncertainties in determining the exact time of arrival of the reflected signal
- Inaccuracies in the time of flight measurement (particularly with laser range sensors)
- The dispersal cone of the transmitted beam (mainly with ultrasonic range sensors)
- Interaction with the target (e.g. surface absorption, specular reflections)
- Variation of propagation speed
- The speed of the mobile robot and target (in the case of a dynamic target)

As discussed below, each type of time-of-flight sensor is sensitive to a particular subset of the above list of factors.

The Ultrasonic Sensor (time-of-flight, sound)

The basic principle of an ultrasonic sensor is to transmit a packet of (ultrasonic) pressure waves and to measure the time it takes for this wave to reflect and return to the receiver. The distance d of the object causing the reflection can be calculated based on the propagation speed of sound c and the time of flight t.

$$d = \frac{c \cdot t}{2} \tag{4.7}$$

The speed of sound c in air is given by

$$c = \sqrt{\gamma RT} \tag{4.8}$$

where

γ: ratio of specific heats

R: gas constant

T: temperature in degree Kelvin

In air at standard pressure and 20° Celsius the speed of sound is approximately c = 343 m/s.

Figure 4.6 shows the different signal output and input of an ultrasonic sensor. First, a series of sound pulses are emitted, comprising the *wave packet*. An integrator also begins to linearly climb in value, measuring the time from the transmission of these sound waves to de-

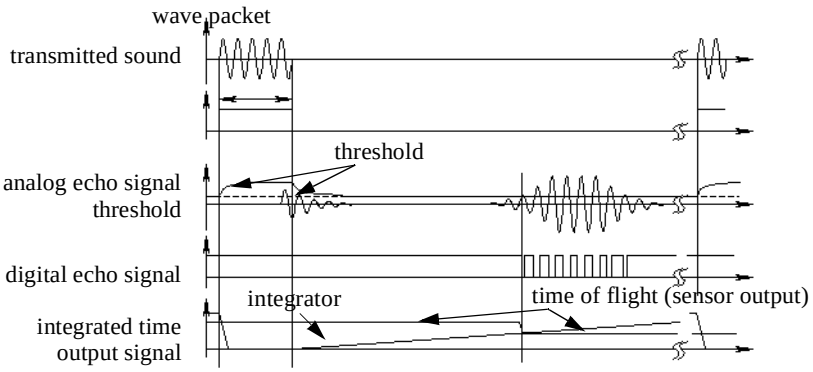


Fig 4.6 Signals of an ultrasonic sensor

tection of an echo. A threshold value is set for triggering an incoming sound wave as a valid echo. This threshold is often decreasing in time, because the amplitude of the expected echo decreases over time based on dispersal as it travels longer. But during transmission of the initial sound pulses and just afterwards, the threshold is set very high to suppress triggering the echo detector with the outgoing sound pulses. A transducer will continue to ring for up to several milliseconds after the initial transmission, and this governs the *blanking time* of the sensor. Note that if, during the blanking time, the transmitted sound were to reflect off of an extremely close object and return to the ultrasonic sensor, it may fail to be detected.

However, once the blanking interval has passed, the system will detect any above-threshold reflected sound, triggering a digital signal and producing the distance measurement using the integrator value.

The ultrasonic wave typically has a frequency between 40 and 180 kHz and is usually generated by a piezo or electrostatic transducer. Often the same unit is used to measure the reflected signal, although the required blanking interval can be reduced through the use of separate output and input devices. Frequency can be used to select a useful range when choosing the appropriate ultrasonic sensor for a mobile robot. Lower frequencies correspond to a longer range, but with the disadvantage of longer post-transmission ringing and, therefore, the need for longer blanking intervals. Most ultrasonic sensors used by mobile robots have an effective range of roughly 12cm to 5 metres. The published accuracy of commercial ultrasonic sensors varies between 98% and 99.1%. In mobile robot applications, specific implementations generally achieve a resolution of approximately 2cm.

In most cases one may want a narrow opening angle for the sound beam in order to also obtain precise directional information about objects that are encountered. This is a major limitation since sound propagates in a cone-like manner (fig. 4.7) with opening angles around 20°- 40°. Consequently, when using ultrasonic ranging one does not acquire depth data points but, rather, entire regions of constant depth. This means that the sensor tells us only that there is an object at a certain distance in within the area of the measurement cone. The sensor readings must be plotted as segments of an arc (sphere for 3D) and not as point measurements (fig. 4.8). However, recent research developments show significant improvement of the measurement quality in using sophisticated echo processing [87].

Ultrasonic sensors suffer from several additional drawbacks, namely in the areas of error,

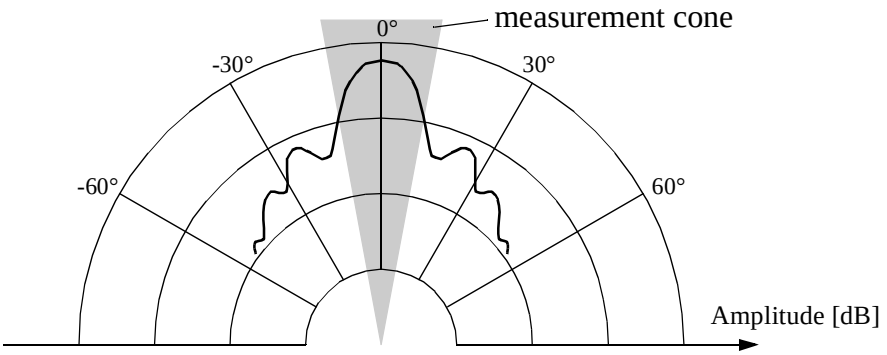


Fig 4.7 *Typical intensity distribution of a ultrasonic sensor*

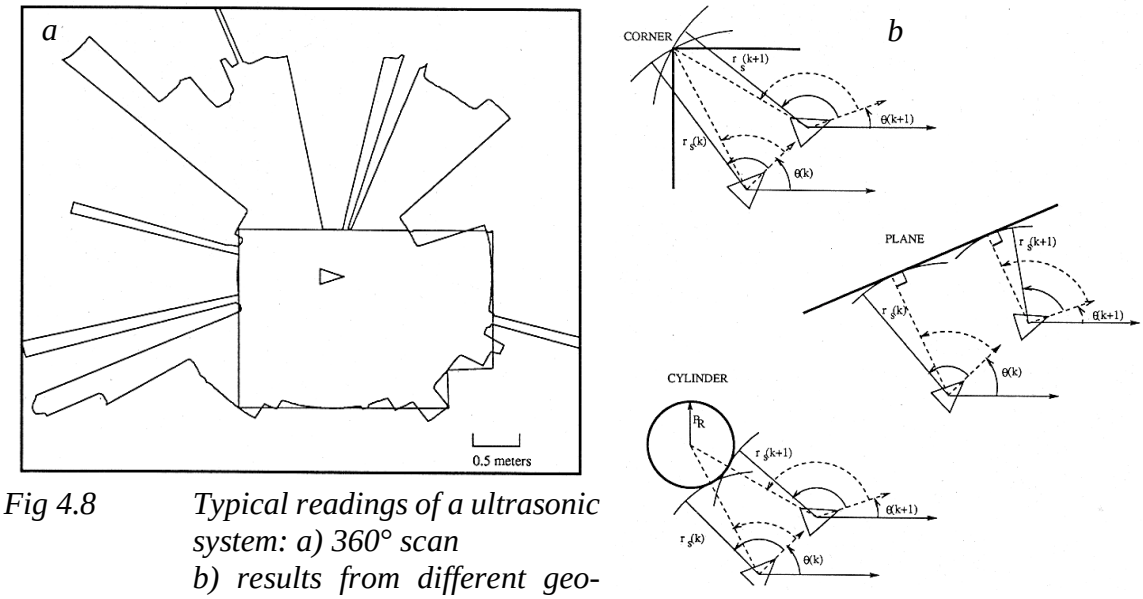


Fig 4.8 *Typical readings of a ultrasonic system: a) 360° scan b) results from different geometric primitives [9].*

bandwidth and cross-sensitivity. The published accuracy values for ultrasonics are nominal values based on successful, perpendicular reflections of the sound wave off an acoustically reflective material. This does not capture the effective error modality seen on a mobile robot moving through its environment. As the ultrasonic transducer's angle to the object being ranged varies away from perpendicular, the chances become good that the sound waves will coherently reflect away from the sensor, just as light at a shallow angle reflects off of a mirror. Therefore, the true error behavior of ultrasonic sensors is compound, with a well-understood error distribution near the true value in the case of a successful retro-reflection, and a more poorly-understood set of range values that are grossly larger than the true value in the case of coherent reflection. Of course the acoustic properties of the material being ranged have direct impact on the sensor's performance. Again, the impact is discrete, with one material possibly failing to produce a reflection that is sufficiently strong to be sensed by the unit. For example, foam, fur and cloth can, in various circumstances, acoustically absorb the sound waves.

A final limitation for ultrasonic ranging relates to bandwidth. Particularly in moderately open spaces, a single ultrasonic sensor has a relatively slow cycle time. For example, measuring the distance to an object that is 3 meters away will take such a sensor 20ms, limiting

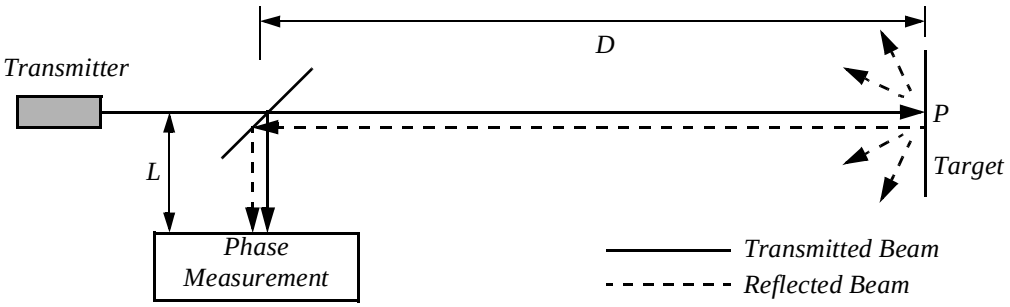


Fig 4.9 Schematic of laser rangefinding by phase-shift measurement.

its operating speed to 50 Hz. But if the robot has a ring of 20 ultrasonic sensors, each firing sequentially and measuring to minimize interference between the sensors, then the ring's cycle time becomes 0.4s and the overall update frequency of any one sensor is just 2.5 Hz. For a robot conducting moderate speed motion while avoiding obstacles using ultrasonics, this update rate can have a measurable impact on the maximum speed possible while still sensing and avoiding obstacles safely.

Laser Rangefinder (time of flight, electromagnetic)

The laser rangefinder is a time-of-flight sensor that achieves significant improvements over the ultrasonic range sensor due to the use of laser light instead of sound. This type of sensor consists of a transmitter which illuminates a target with a collimated beam (e.g. laser), and a receiver capable of detecting the component of light which is essentially coaxial with the transmitted beam. Often referred to as optical radar or *lidar* (*light detection and ranging*), these devices produce a range estimate based on the time needed for the light to reach the target and return. A mechanical mechanism with a mirror sweeps the light beam to cover the required scene in a plane or even in 3 dimensions, using a rotating, nodding mirror.

One way to measure the time of flight for the light beam is to use a pulsed laser and then measured the elapsed time directly, just as in the ultrasonic solution described earlier. Electronics capable of resolving picoseconds are required in such devices and they are therefore very expensive. A second method is to measure the beat frequency between a frequency modulated continuous wave (F.M.C.W.) and its received reflection. Another, even easier method is to measure the phase shift of the reflected light. We describe this third approach in detail.

Phase-Shift Measurement

Near infrared light (from an LED or a laser) is collimated and transmitted from the transmitter T in figure 4.9 and hits a point P in the environment. For surfaces having a roughness greater than the wavelength of the incident light, diffuse reflection will occur, meaning that the light is reflected almost isotropically. The wavelength of the infrared light emitted is 824 nm and so most surfaces with the exception of only highly polished reflecting objects, will be diffuse reflectors. The component of the infrared light which falls within the receiving aperture of the sensor will return almost parallel to the transmitted beam, for distant objects.

The sensor transmits 100% amplitude modulated light at a known frequency and measures

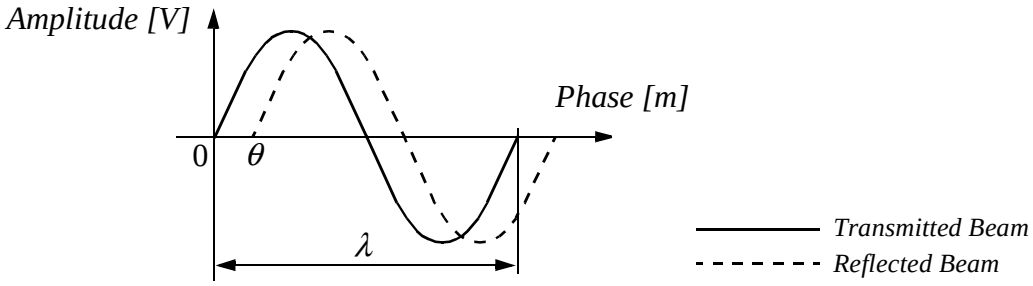


Fig 4.10 *Range estimation by measuring the phase shift between transmitted and received signals.*

the phase shift between the transmitted and reflected signals. Figure 4.10 shows how this technique can be used to measure range. The wavelength of the modulating signal obeys the equation $c = f\lambda$ where c is the speed of light and f the modulating frequency. For f = 5 Mhz (as in the AT&T sensor), λ = 60 meters. The total distance D' covered by the emitted light is

$$D' = L + 2D = L + \frac{\theta}{2\pi}\lambda \tag{4.9}$$

where D and L are the distances defined in figure 4.9. The required distance D, between the beam splitter and the target, is therefore given by

$$D = \frac{\lambda}{4\pi}\theta \tag{4.10}$$

where θ is the electronically measured phase difference between the transmitted and reflected light beams, and λ the known modulating wavelength. It can be seen that the transmission of a single frequency modulated wave can theoretically result in ambiguous range estimates since for example if λ = 60 meters, a target at a range of 5 meters would give an indistinguishable phase measurement from a target at 65 meters, since each phase angle would be 360° apart. We therefore define an "*ambiguity interval*" of λ, but in practice we note that the range of the sensor is much lower than λ due to the attenuation of the signal in air.

It can be shown that the confidence in the range (phase estimate) is inversely proportional to the square of the received signal amplitude, directly affecting the sensor's accuracy. Hence dark, distant objects will not produce as good range estimates as close, bright objects.

In figure 4.11 the schematic of a typical 360° laser range sensor and two examples are shown. Figure 4.12 shows a typical range image of a 360° scan taken with an laser range sensor.

As expected, the angular resolution of laser rangefinders far exceeds that of ultrasonic sensors. The Sick laser scanner shown in Figure 4.11 achieves an angular resolution of 0.5°. Depth resolution is approximately 5cm, over a range from 5cm up to 20m or more, depending upon the brightness of the object being ranged. This device performs 25 180° scans per second but has no mirror nodding capability for the vertical dimension.

As with ultrasonic ranging sensors, an important error mode involves coherent reflection of the energy. With light, this will only occur when striking a highly polishes surface. Practi-

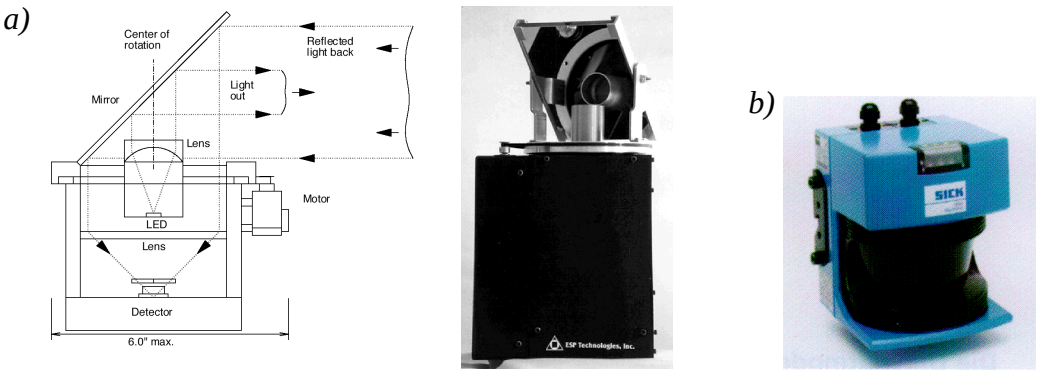


Fig 4.11 *a) Schematic drawing and photo of a laser range sensor with rotating mirror (EPS Technologies Inc.)*
b) Industrial 180° laser range sensor from Sick Inc., Germany

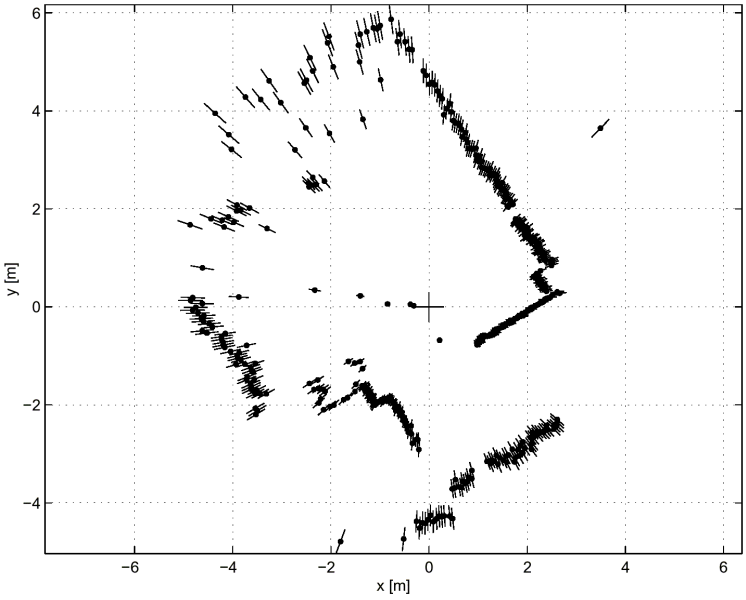


Fig 4.12 *Typical range image of a 2D laser range sensor with a rotating mirror. The length of the lines through the measurement points indicate the uncertainties.*

cally, a mobile robot may encounter such surfaces in the form of a polished desktop, file cabinet or of course a mirror. Unlike ultrasonic sensors, laser rangefinders cannot detect the presence of optically transparent materials such as glass, and this can be a significant obstacle in environments, for example museums, where glass is commonly used.

4.1.6.2 Triangulation-based Active Ranging

Triangulation-based ranging sensors use geometrical properties manifest in their measuring strategy to establish distance readings to objects. The simplest class of triangulation-based rangers are *active* because they project a known light pattern (e.g. a point, a line or a texture) onto the environment. The reflection of the known pattern is captured by a receiver and, together with known geometric values, the system can use simple triangulation to establish range measurements. If the receiver measures the position of the reflection along a single axis, we call the sensor an optical triangulation sensor in 1D. If the receiver measures the position of the reflection along two orthogonal axes, we call the sensor a structured light sensor. These two sensor types are described in the two subsections below.

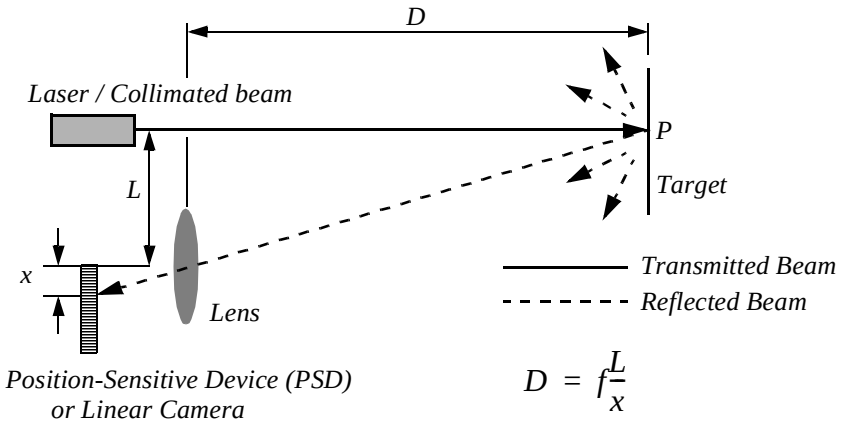


Fig 4.13 *Principle of 1D laser triangulation*

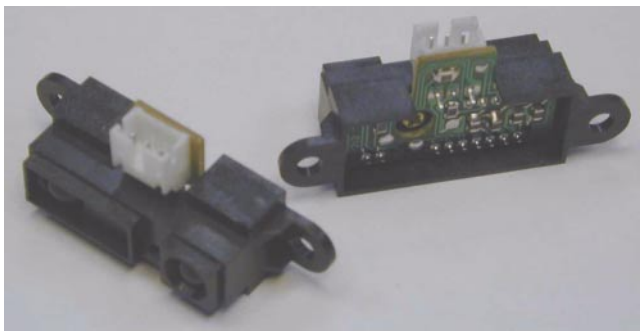


Fig 4.14 *A commercially available, low-cost optical triangulation sensor: the Sharp GP-series infrared rangefinders provide either analog or digital distance measures and cost only about $15.*

Optical Triangulation (1D sensor)

The principle of optical triangulation in 1D is straightforward, as depicted in figure 4.13. A collimated beam (e.g. focused infrared L.E.D., laser beam) is transmitted toward the target. The reflected light is collected by a lens and projected onto a position sensitive device (PSD) or linear camera. Given the geometry of figure 4.13 the distance *D* is given by

$$D = f\frac{L}{x} \tag{4.11}$$

The distance is proportional to 1/*x*, therefore the sensor resolution is best for close objects and becomes poor at a distance. Sensors based on this principle are used in range sensing up to one or two meters, but also in high precision industrial measurements with resolutions far below one µm.

Optical triangulation devices can provide relatively high accuracy with very good resolution (for close objects). However, the operating range of such a device is normally fairly limited by geometry. For example, the optical triangulation sensor pictured in Figure 4.14 operates over a distance range of between 8cm and 80cm. It is inexpensive as compared to ultrasonic

and laser rangefinder sensors. Although more limited in range than sonar, the optical triangulation sensor has high bandwidth and does not suffer from cross-sensitivities that are more common in the sound domain.

Structured Light (2D sensor)

If one replaced the linear camera or PSD of an optical triangulation sensor with a two-dimensional receiver such as a CCD or CMOS camera, then one can recover distance to a large set of points instead of to only one point. The emitter must project a known pattern, or *structured light*, onto the environment. Many systems exist which either project light textures (fig. 4.15b) or emit collimated light (possibly laser) by means of a rotating mirror. Yet another popular alternative is to project a laser stripe (fig. 4.15a) by turning a laser beam into a plane using a prism. Regardless of how it is created, the projected light has a known structure, and therefore the image taken by the CCD or CMOS receiver can be filtered to identify the pattern's reflection.

Note that the problem of recovering depth is in this case far simpler than the problem of passive image analysis. In passive image analysis, as we discuss later, existing features in the environment must be used to perform *correlation*, while the present method projects a known pattern upon the environment and thereby avoids the standard correlation problem

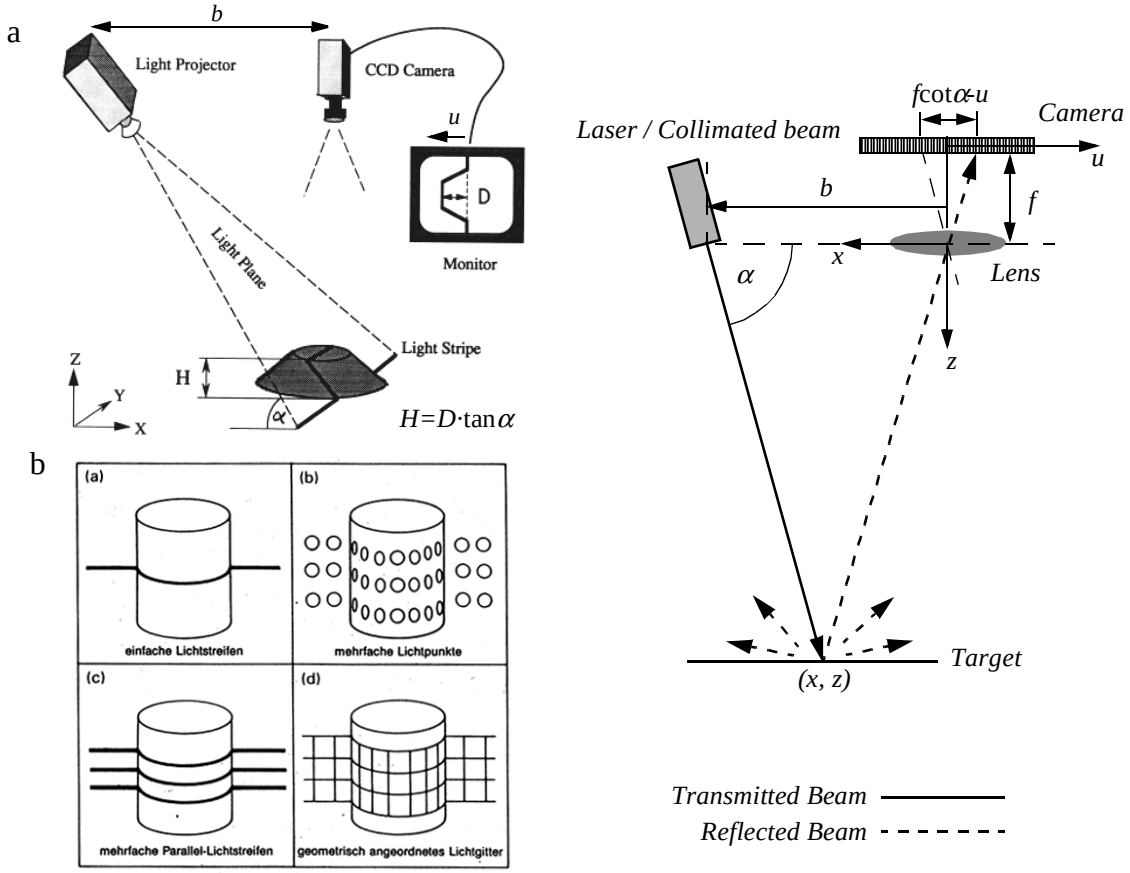


Fig 4.15 *a) Principle of active two dimensional triangulation*
b) Other possible light structures
c) One-dimensional schematic of the principle

altogether. Furthermore, the structured light sensor is an active device; so, it will continue to work in dark environments as well as environments in which the objects are featureless (e.g. uniformly colored and edgeless). In contrast, stereo vision would fail in such texture-free circumstances.

Figure 4.15c shows a one-dimensional active triangulation geometry. We can examine the trade-off in the design of triangulation systems by examining the geometry in figure 4.15c. The measured values in the system are α and u, the distance of the illuminated point from the origin in the imaging sensor. (Note the imaging sensor here can be a camera or an array of photo diodes of a position sensitive device (e.g. a 2D PSD).

From figure 4.15c, simple geometry shows that:

$$x = \frac{b \cdot u}{f\cot\alpha - u}; \quad z = \frac{b \cdot f}{f\cot\alpha - u} \tag{4.12}$$

where f is the distance of the lens to the imaging plane. In the limit, the ratio of image resolution to range resolution is defined as the triangulation gain G_p and from equation 4.12 is given by:

$$\frac{\partial u}{\partial z} = G_p = \frac{b \cdot f}{z^2} \tag{4.13}$$

This shows that the ranging accuracy, for a given image resolution, is proportional to source/detector separation b and focal length f, and decreases with the square of the range z. In a scanning ranging system, there is an additional effect on the ranging accuracy, caused by the measurement of the projection angle α. From equation 4.12 we see that:

$$\frac{\partial \alpha}{\partial z} = G_\alpha = \frac{b\sin\alpha^2}{z^2} \tag{4.14}$$

We can summarize the effects of the parameters on the sensor accuracy as follows:

- *Baseline length b*: the smaller b is the more compact the sensor can be. The larger b is the better the range resolution will be. Note also that although these sensors do not suffer from the correspondence problem, the disparity problem still occurs. As the baseline length b is increased, one introduces the chance that, for close objects, the illuminated point(s) may not be in the receiver's field of view.
- *Detector length and focal length f*: A larger detector length can provide either a larger field of view or an improved range resolution or partial benefits for both. Increasing the detector length however means a larger sensor head and worse electrical characteristics (increase in random error and reduction of bandwidth). Also, a short focal length gives a large field of view at the expense of accuracy and vice versa.

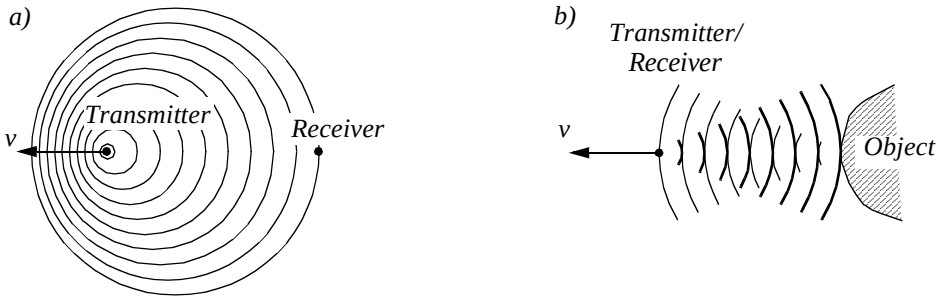


Fig 4.16 *Doppler effect between two moving objects (a) or a moving and a stationary object(b)*

At one time, laser stripe-based structured light sensors were common on several mobile robot bases as an inexpensive alternative to laser rangefinding devices. However, with the increasing quality of laser rangefinding sensors in the 1990's the structured light system has become relegated largely to vision research rather than applied mobile robotics.

4.1.7 Motion/Speed sensors

Some sensors directly measure the relative motion between the robot and its environment. Since such motion sensors detect relative motion, so long as an object is moving relative to the robot's reference frame, it will be detected and its speed can be estimated. There are a number of sensors that inherently measure some aspect of motion or change. For example, a Pyroelectric sensor detects change in heat. When a human walks across the sensor's field of view, his motion triggers a change in heat in the sensor's reference frame. In the next subsection, we describe an important type of motion detector based on the Doppler effect. These sensors represent a well-known technology with decades of general applications behind them. For fast-moving mobile robots such as autonomous highway vehicles and unmanned flying vehicles, Doppler-based motion detectors are the obstacle detection sensor of choice.

4.1.7.1 Doppler Effect Based Sensing (radar or sound)

Anyone who has noticed the change in siren pitch that occurs when an approaching fire truck passes by is familiar with the Doppler effect.

A transmitter emits an electromagnetic or sound wave with a frequency f_t. It is either received by a receiver (fig. 4.16a) or reflected from an object (fig. 4.16b). The measured frequency f_r at the receiver is a function of the relative speed v between transmitter and receiver according to

$$f_r = f_t \frac{1}{1 + v/c} \tag{4.15}$$

if the transmitter is moving and

$$f_r = f_t(1 + v/c) \tag{4.16}$$

if the receiver is moving.

In the case of a reflected wave (fig. 4.16b) there is a factor of two introduced, since any change x in relative separation affects the round-trip path length by $2x$. Furthermore, in such situations it is generally more convenient to consider the change in frequency Δf, known as the *Doppler shift*, as opposed to the *Doppler frequency* notation above.

$$\Delta f = f_t - f_r = \frac{2 f_t v \cos\theta}{c} \tag{4.17}$$

$$v = \frac{\Delta f \cdot c}{2 f_t \cos\theta} \tag{4.18}$$

where:

Δf = Doppler frequency shift

θ = relative angle between direction of motion and beam axis

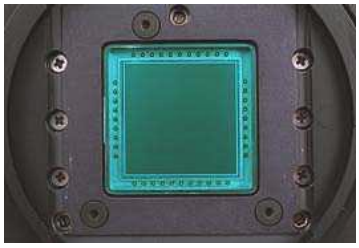
The Doppler effect applies to sound and electromagnetic waves. It has a wide spectrum of applications:

- *Sound waves*: e.g. industrial process control, security, fish finding, measure of ground speed
- *Electromagnetic waves*: e.g. vibration measurement, radar systems, object tracking

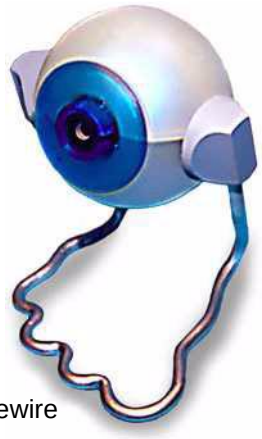
A current application area is both autonomous and manned highway vehicles. Both microwave and laser radar systems have been designed for this environment. Both systems have equivalent range, but laser can suffer when visual signals are deteriorated by environmental conditions such as rain, fog, etc. Commercial microwave radar systems are already available for installation on highway trucks. These systems are called VORAD (vehicle onboard radar) and have a total range of approximately 150m. With an accuracy of approximately 97%, these systems report range rate from 0 to 160 km/hr with a resolution of 1 km/hr. The beam is approximately 4° wide and 5° in elevation. One of the key limitations of radar technology is its bandwidth. Existing systems can provide information on multiple targets at approximately 2 Hz.

4.1.8 Vision-based sensors

Vision is our most powerful sense. It provides us with an enormous amount of information about the environment and enables rich, intelligent interaction in dynamic environments. It is therefore not surprising that a great deal of effort has been devoted to providing machines with sensors that mimic the capabilities of the human vision system. The first step in this process is the creation of sensing devices that capture the same raw information- light- that the human vision system uses. The next subsection describes the two current technologies for creating vision sensors: CCD and CMOS. These sensors have specific limitations in performance when compared to the human eye, and it is important for the reader to understand these limitations. Afterwards, the second and third subsections describe vision-based sensors that are commercially available, like the sensors discussed previously in this chapter,



2048 x 2048 CCD array



Orangemicro iBOT Firewire

Sony DFW-X700

Cannon IXUS 300

Fig 4.17 *Commercially available CCD chips and CCD cameras. Because this technology is relatively mature, cameras are available in widely varying forms and costs [http://www.howstuffworks.com/digital-camera2.htm].*

along with their disadvantages and most popular applications.

4.1.8.1 CCD and CMOS sensors

The Charged Coupled Device (CCD) is the most popular basic ingredient of robotic vision systems today. The CCD chip (see Fig. 4.17) is an array of light-sensitive picture elements, or pixels, usually with between 20,000 and 2 million pixels total. Each pixel can be thought of as a light-sensitive, discharging capacitor that is 5 to 25 microns in size. First, the capacitors of all pixels are charged fully, then the integration period begins. As photons of light strike each pixel, they liberate electrons, which are captured by electric fields and retained at the pixel. Over time, each pixel accumulates a varying level of charge based on the total number of photons that have struck it. After the integration period is complete, the relative charges of all pixels need to be frozen and read. In a CCD, the reading process is performed at one corner of the CCD chip. The bottom row of pixel charges are transported to this corner and read, then the rows above shift down and the process repeats. This means that each charge must be transported across the chip, and it is critical that the value be preserved. This requires specialized control circuitry and custom fabrication techniques to ensure the stability of transported charges.

The photodiodes used in CCD chips (and CMOS as well) are not equally sensitive to all frequencies of light. They are sensitive to light between 400nm and 1000nm wavelength. It is important to remember that photodiodes are less sensitive to the ultraviolet part of the spectrum (e.g. blue) and are overly sensitive to the infrared portion (e.g. heat).

You can see that the basic light-measuring process is colorless: it is just measuring the total number of photons that strike each pixel in the integration period. There are two common approaches for creating *color* images. If the pixels on the CCD chip are grouped into 2x2 sets of 4, then red, green and blue dyes can be applied to a color filter so that each individual pixel receives only light of one color. Normally, two pixels measure green while one pixel each measures red and blue light intensity. Of course, this 1-chip color CCD has a geometric

resolution disadvantage. The number of pixels in the system has been effectively cut by a factor of 4, and therefore the image resolution output by the CCD camera will be sacrificed.

The 3-chip color camera avoids these problems by splitting the incoming light into three complete (lower intensity) copies. Three separate CCD chips receive the light, with one red, green or blue filter over each entire chip. Thus, in parallel, each chip measures light intensity for one color, and the camera must combine the CCD chips' outputs to create a joint color image. Resolution is preserved in this solution, although the 3-chip color cameras are, as one would expect, significantly more expensive and therefore more rarely used in mobile robotics.

Both 3-chip and single chip color CCD cameras suffer from the fact that photodiodes are much more sensitive to the near-infrared end of the spectrum. This means that the overall system detects blue light much more poorly than red and green. To compensate, the gain must be increased on the blue channel, and this introduces greater absolute noise on blue than on red and green. It is not uncommon to assume at least 1 - 2 bits of additional noise on the blue channel. Although there is no satisfactory solution to this problem today, over time the processes for blue detection have been improved and we expect this positive trend to continue.

The CCD camera has several camera parameters that affect its behavior. In some cameras, these parameter values are fixed. In others, the values are constantly changing based on built-in feedback loops. In higher-end cameras, the user can modify the values of these parameters via software. The *iris position* and *shutter speed* regulate the amount of light being measured by the camera. The iris is simply a mechanical aperture that constricts incoming light, just as in standard 35mm cameras. Shutter speed regulates the integration period of the chip. In higher-end cameras, the effective shutter speed can be as brief at 1/30,000s and as long as 2s. *Camera gain* controls the overall amplification of the analog signal, prior to A/D conversion. However, it is very important to understand that, even though the image may appear brighter after setting high gain, the shutter speed and iris may not have changed at all. Thus gain merely amplifies the signal, and amplifies along with the signal all of the associated noise and error. Although useful in applications where imaging is done for human consumption (e.g. photography, television), gain is of little value to a mobile roboticist.

In color cameras, an additional control exists for *white balance*. Depending on the source of illumination in a scene (e.g. fluorescent lamps, incandescent lamps, sunlight, underwater filtered light, etc.) the relative measurements of red, green and blue light that define pure white light will change dramatically. The human eyes compensate for all such effects in ways that are not fully understood, however, the camera can demonstrate glaring inconsistencies in which the same table looks blue in one image, taken during the night, and yellow in another image, taken during the day. White balance controls enable the user to change the relative gain for red, green and blue in order to maintain more consistent color definitions in varying contexts.

The key disadvantages of CCD cameras are primarily in the areas of inconstancy and dynamic range. As mentioned above, a number of parameters can change the brightness and colors with which a camera creates its image. Manipulating these parameters in a way to

provide consistency over time and over environments, for example ensuring that a green shirt always looks green, and something dark grey is always dark grey, remains an open problem in the vision community. For more details in the fields of color constancy and luminosity constancy, consult [Roland, see text notes for the right reference for here! (Kobus Barnard is the ref)].

The second class of disadvantages relates to the behavior of a CCD chip in environments with extreme illumination. In cases of very low illumination, each pixel will receive only a small number of photons. The longest possible integration period (i.e. shutter speed) and camera optics (i.e. pixel size, chip size, lens focal length and diameter) will determine the minimum level of light for which the signal is stronger than random error noise. In cases of very high illumination, a pixel fills its well with free electrons and, as the well reaches its limit, the probability of trapping additional electrons falls and therefore the linearity between incoming light and electrons in the well degrades. This is termed *saturation* and can indicate the existence of a further problem related to cross-sensitivity. When a well has reached its limit, then additional light within the remainder of the integration period may cause further charge to leak into neighboring pixels, causing them to report incorrect values or even reach secondary saturation. This effect, called *blooming*, means that individual pixel values are not truly independent.

The camera parameters may be adjusted for an environment with a particular light level, but the problem remains that the dynamic range of a camera is limited by the well capacity of the individual pixels. For example, a high quality CCD may have pixels that can hold 40,000 electrons. The noise level for reading the well may be 11 electrons, and therefore the dynamic range will be 40,000:11, or 3,600:1, which is 35dB.

CMOS technology

The Complementary Metal Oxide Semiconductor (CMOS) chip is a significant departure from the CCD. It too has an array of pixels, but located alongside each pixel are several transistors specific to that pixel. Just as in CCD chips, all of the pixels accumulate charge during the integration period. During the data collection step, the CMOS takes a new approach: the pixel-specific circuitry next to every pixel measures and amplifies the pixel's signal, all in parallel for every pixel in the array. Using more traditional traces from general semiconductor chips, the resulting pixel values are all carried to their destinations.

CMOS has a number of advantages over CCD technologies. First and foremost, there is no need for the specialized clock drivers and circuitry required in the CCD to transfer each pixel's clock down all of the array columns and across all of its rows. This also means that specialized semiconductor manufacturing processes are not required to create CMOS chips. Therefore, the same production lines that create microchips can create inexpensive CMOS chips as well (see Fig. 4.18). The CMOS chip is so much simpler that it consumes significantly less power; incredibly, it operates with a power consumption that is 1/100 the power consumption of a CCD chip. In a mobile robot, power is a scarce resource and therefore this is an important advantage.

On the other hand, the CMOS chip also faces several disadvantages. Most importantly, the

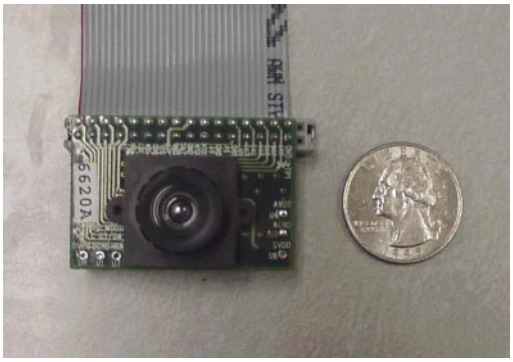


Fig 4.18 A commercially available, low-cost CMOS camera with lens attached.

circuitry next to each pixel consumes valuable real estate on the face of the light-detecting array. Many photons hit the transistors rather than the photodiode, making the CMOS chip significantly less sensitive than an equivalent CCD chip. Second, the CMOS technology is younger and, as a result, the best resolution that one can purchase in CMOS format continues to be far inferior to the best CCD chips available. Time will doubtless bring the high end CMOS imagers closer to CCD imaging performance.

Given this summary of the mechanism behind CCD and CMOS chips, one can appreciate the sensitivity of any vision-based robot sensor to its environment. As compared to the human eye, these chips all have far poorer adaptation, cross-sensitivity and dynamic range. As a result, vision sensors today continue to be fragile. Only over time, as the underlying performance of imaging chips improves, will significantly more robust vision-based sensors for mobile robots be available.

4.1.8.2 Visual ranging sensors

Range sensing is extremely important in mobile robotics as it is a basic input for successful obstacle avoidance. As we have seen earlier in this chapter, a number of sensors are popular in robotics explicitly for their ability to recover depth estimates: ultrasonic, laser rangefinder, optical rangefinder, etc. It is natural to attempt to implement ranging functionality using vision chips as well.

However, a fundamental problem with visual images makes rangefinding relatively difficult. Any vision chip collapses the three-dimensional world into a two-dimensional image plane, thereby losing depth information. If one can make strong assumptions regarding the size of objects in the world, or their particular color and reflectance, then one can directly interpret the appearance of the two-dimensional image to recover depth. But such assumptions are rarely possible in real-world mobile robot applications. Without such assumptions, a single picture does not provide enough information to recover spatial information.

The general solution is to recover depth by looking at *several* images of the scene to gain more information, hopefully enough to at least partially recover depth. The images used must be different, so that taken together they provide additional information. They could differ in viewpoint, yielding *stereo* or *motion* algorithms. An alternative is to create different images, not by changing the viewpoint, but by changing the camera geometry, such as the focus position or lens iris. This is the fundamental idea behind depth from focus and

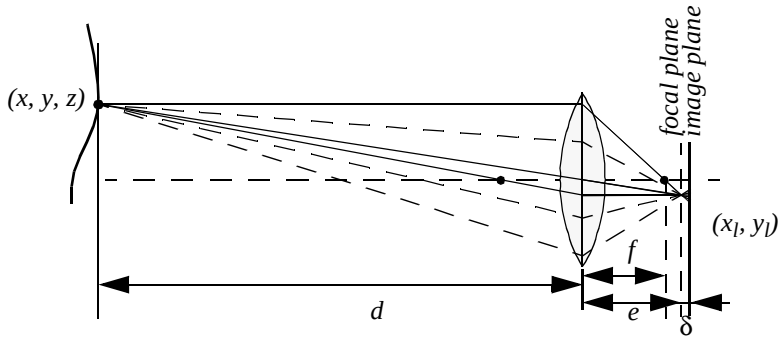


Fig 4.19 *Depiction of the camera optics and its impact on the image. In order to get a sharp image, the image plane must coincide with the focal plane. Otherwise the image of the point (x,y,z) will be blurred in the image as can be seen in the drawing above.*

depth from defocus techniques.

In the next section, we outline the general approach to the depth from focus techniques because it presents a straightforward and efficient way to create a vision-based range sensor. Subsequently, we present details for the correspondence-based techniques of depth from stereo and motion.

Depth from focus

The depth from focus class of techniques relies on the fact that image properties not only change as a function of the scene, but also as a function of the camera parameters. The relationship between camera parameters and image properties is depicted in figure 4.19.

The basic formula governing image formation relates the distance of the object from the lens, *d* in the above figure, to the distance *e* from the lens to the focal point, based on the focal length *f* of the lens:

$$\frac{1}{f} = \frac{1}{d} + \frac{1}{e} \tag{4.19}$$

If the image plane is located at distance *e* from the lens, then for the specific object voxel depicted, all light will be focused at a single point on the image plane and the object voxel will be *focused*. However, when the image plane is not at *e*, as is depicted in Figure (4.19), then the light from the object voxel will be cast on the image plane as a *blur circle*. To a first approximation, the light is homogeneously distributed throughout this blur circle, and the radius *R* of the circle can be characterized according to the equation:

$$R = \frac{L\delta}{2e} \tag{4.20}$$

L is the diameter of the lens or aperture and δ is the displacement of the image plan from the focal point.

Given these formulae, several basic optical effects are clear. For example, if the aperture or

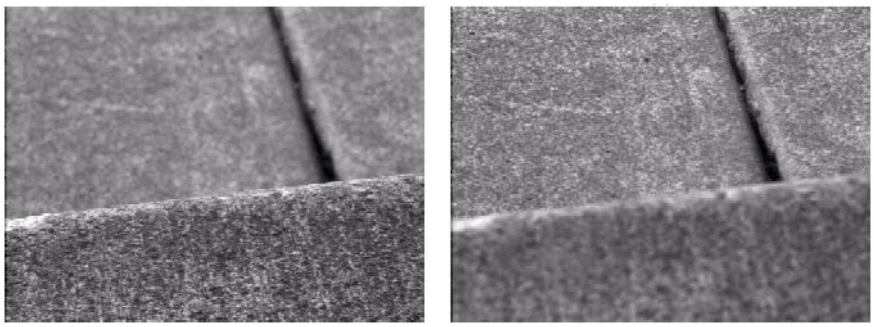


Fig 4.20 *Two images of the same scene taken with a camera at two different focusing positions. Note the significant change in texture sharpness between the near surface and far surface. The scene is an outdoor concrete step.*

lens is reduced to a point, as in a pin-hole camera, then the radius of the blur circle approaches zero. This is consistent with the fact that decreasing the iris aperture opening causes the *depth of field* to increase until all objects are in focus. Of course, the disadvantage of doing so is that we are allowing less light to form the image on the image plane and so this is practical only in bright circumstances.

The second property that can be deduced from these optics equations relates to the sensitivity of blurring as a function of the distance from the lens to the object. Suppose the image plane is at a fixed distance 1.2 from a lens with diameter $L = 0.2$ and focal length $f = 0.5$. We can see from Equation (4.20) that the size of the blur circle R changes proportionally with the image plane displacement δ. If the object is at distance $d = 1$, then from Equation (4.19) we can compute $e=1$ and therefore $\delta = 0.2$. Increase the object distance to $d = 2$ and as a result $\delta = 0.533$. Using Equation (4.20) in each case we can compute $R = 0.02$ $R = 0.08$ respectively. This demonstrates high sensitivity for defocusing when the object is close to the lens.

In contrast suppose the object is at $d = 10$. In this case we compute $e = 0.526$. But if the object is again moved one unit, to $d = 11$, then we compute $e = 0.524$. Then resulting blur circles are $R = 0.117$ and $R = 0.129$, far less than the quadrupling in R when the obstacle is 1/10 the distance from the lens. This analysis demonstrates the fundamental limitation of depth from focus techniques: they lose sensitivity as objects move further away (given a fixed focal length). Interestingly, this limitation will turn out to apply to virtually all visual ranging techniques, including depth from stereo and depth from motion.

Nevertheless, camera optics can be customized for the depth range of the intended application. For example, a "zoom" lens with a very large focal length f will enable range resolution at significant distances, of course at the expense of field of view. Similarly, a large lens diameter, coupled with a very fast shutter speed, will lead to larger, more detectable blur circles.

Given the physical effects summarized by the above equations, one can imagine a visual ranging sensor that makes use of multiple images in which camera optics are varied (e.g. image plane displacement δ) and the same scene is captured (see Fig. 4.20). In fact this approach is not a new invention. The human visual system uses an abundance of cues and

Fig 4.21 *The Cheshm robot uses three monochrome cameras as its only ranging sensor for obstacle avoidance in the context of humans, static obstacles such as bushes and convex obstacles such as ledges and steps. Roland, the picture Im sending you is Cheshm.jpg*

techniques, and one system demonstrated in humans is depth from focus. Humans vary the focal length of their lens continuously at a rate of about 2 Hz. Such approaches, in which the lens optics are actively searched in order to maximize focus, are technically called *depth from focus.* In contrast, *depth from defocus* means that depth is recovered using a series of images that have been taken with different camera geometries.

Depth from focus methods are one of the simplest visual ranging techniques. To determine the range to an object, the sensor simply moves the image plane (via focusing) until maximizing the sharpness of the object. When the sharpness is maximized, the corresponding position of the image plane directly reports range. Some autofocus cameras and virtually all autofocus video cameras use this technique. Of course, a method is required for measuring the sharpness of an image or an object within the image. The most common techniques are approximate measurements of the sub-image *gradient*:

$$sharpness_1 = \sum_{x,\ y} |I(x,\ y) - I(x-1,\ y)| \tag{4.21}$$

$$sharpness_2 = \sum_{x,\ y} (I(x,\ y) - I(x-2,\ y-2))^2 \tag{4.22}$$

A significant advantage of the horizontal sum of differences technique (Equation (4.21)) is that the calculation can be implemented in analog circuitry using just a rectifier, a low-pass filter and a high-pass filter. This is a common approach in commercial cameras and video recorders. Such systems will be sensitive to contrast along one particular axis, although in practical terms this is rarely an issue.

However depth from focus is an active search method and will be slow because it takes time to change the focusing parameters of the camera, using for example a servo-controlled focusing ring. For this reason this method has not been applied to mobile robots.

A variation of the depth from focus technique has been applied to a mobile robot, demonstrating obstacle avoidance in a variety of environments as well as avoidance of concave obstacles such as steps and ledges [95]. This robot uses three monochrome cameras placed as close together as possible with different, fixed lens focus positions (Fig. 4.21).

Several times each second, all three frame-synchronized cameras simultaneously capture three images of the same scene. The images are each divided into five columns and three rows, or 15 subregions. The approximate sharpness of each region is computed using a variation of Equation (4.22), leading to a total of 45 sharpness values. Note that Equation 22 calculates sharpness along diagonals but skips one row. This is due to a subtle but important issue. Many cameras produce images in *interlaced* mode. This means that the odd rows are captured first, then afterwards the even rows are captured. When such a camera is used in dynamic environments, for example on a moving robot, then adjacent rows show the dynamic scene at two different time points, differing by up to 1/30 seconds. The result is an artificial blurring due to motion and not optical defocus. By comparing only even-number rows we avoid this interlacing side effect.

Recall that the three images are each taken with a camera using a different focus position. Based on the focusing position, we call each image *close*, *medium* or *far*. A 5x3 coarse depth map of the scene is constructed quickly by simply comparing the sharpness values of each three corresponding regions. Thus, the depth map assigns only two bits of depth information to each region using the values *close*, *medium* and *far*. The critical step is to adjust the focus positions of all three cameras so that flat ground in front of the obstacle results in *medium* readings in one row of the depth map. Then, unexpected readings of either *close* or *far* will indicate convex and concave obstacles respectively, enabling basic obstacle avoidance in the vicinity of objects on the ground as well as drop-offs into the ground.

Although sufficient for obstacle avoidance, the above depth from focus algorithm presents unsatisfyingly coarse range information. The alternative is *depth from defocus*, the most desirable of the focus-based vision techniques.

Depth from defocus methods take as input two or more images of the same scene, taken with different, known camera geometry. Given the images and the camera geometry settings, the goal is to recover the depth information of the three-dimensional scene represented by the images. We begin by deriving the relationship between the actual scene properties (irradiance and depth), camera geometry settings and the image g that is formed at the image plane.

The *focused image* $f(x,y)$ of a scene is defined as follows. Consider a pinhole aperture (L=0) in lieu of the lens. For every point p at position (x,y) on the image plane, draw a line through the pinhole aperture to the corresponding, visible point P in the actual scene. We define $f(x,y)$ as the irradiance (or light intensity) at p due to the light from P. Intuitively, $f(x,y)$ represents the intensity image of the scene perfectly in focus.

The *point spread function* $h(x_g, y_g, x_f, y_f, R_{x,y})$ is defined as the amount of irradiance from point P in the scene (corresponding to (x_f, y_f) in the focused image f that contributes to point (x_g, y_g) in the observed, defocused image g. Note that the point spread function depends not only upon the source, (x_f, y_f), and the target, (x_g, y_g), but also on R, the blur circle radius. R, in turn, depends upon the distance from point P to the lens, as can be seen by studying Equations (4.19) and (4.20).

Given the assumption that the blur circle is homogeneous in intensity, we can define h as follows:

$$h(x_g,\ y_g,\ x_f,\ y_f,\ R_{x,\ y}) = \begin{bmatrix} \frac{1}{\pi R^2} & if\ ((x_g - x_f)^2 + (y_g - y_f)^2) \leq R^2 \\ 0 & if\ ((x_g - x_f)^2 + (y_g - y_f)^2) > R^2 \end{bmatrix} \tag{4.23}$$

Intuitively, point P contributes to the image pixel $(x_g,\ y_g)$ only when the blur circle of point P contains the point $(x_g,\ y_g)$. Now we can write the general formula that computes the value of each pixel in the image, $(x,\ y)$, as a function of the point spread function and the focused image:

$$g(x_g,\ y_g) = \sum_{x,\ y} h(x_g,\ y_g,\ x,\ y,\ R_{x,\ y}) f(x,\ y) \tag{4.24}$$

This equation relates the depth of scene points via R to the observed image g. Solving for R would provide us with the depth map. However, this function has another unknown, and that is f, the focused image. Therefore, one image alone is insufficient to solve the depth recovery problem, assuming we do *not* know how the fully focused image would look.

Given two images of the same scene, taken with varying camera geometry, in theory it will be possible to solve for g as well as R because f stays constant. There are a number of algorithms for implementing such a solution accurately and quickly. The classical approach is known as *inverse filtering* because it attempts to directly solve for R, then extract depth information from this solution. One special case of the inverse filtering solution has been demonstrated with a real sensor. Suppose that the incoming light is split and sent to two cameras, one with a large aperture and the other with a pinhole aperture [94]. The pinhole aperture results in a fully focused image, directly providing the value of f. With this approach, there remains a single equation with a single unknown, and so the solution is straightforward. Pentland has demonstrated such a sensor, with several meters of range and better than 97% accuracy. Note, however, that the pinhole aperture necessitates a large amount of incoming light, and that furthermore the actual image intensities must be normalized so that the pinhole and large-diameter images have equivalent total radiosity. More recent depth from defocus methods use statistical techniques and characterization of the problem as a set of linear equations [93]. These matrix-based methods have recently achieved significant improvements in accuracy over all previous work.

In summary, the basic advantage of the depth from defocus method is its extremely fast speed. The equations above do not require search algorithms to find the solution, as would the correlation problem faced by depth from stereo methods. Perhaps more importantly, the depth from defocus methods also need not capture the scene at different perspectives, and are therefore unaffected by occlusions and the disappearance of objects in a second view.

As with all visual methods for ranging, accuracy decreases with distance. Indeed, the accuracy can be extreme; these methods have been used in microscopy to demonstrate ranging at the micrometer level.

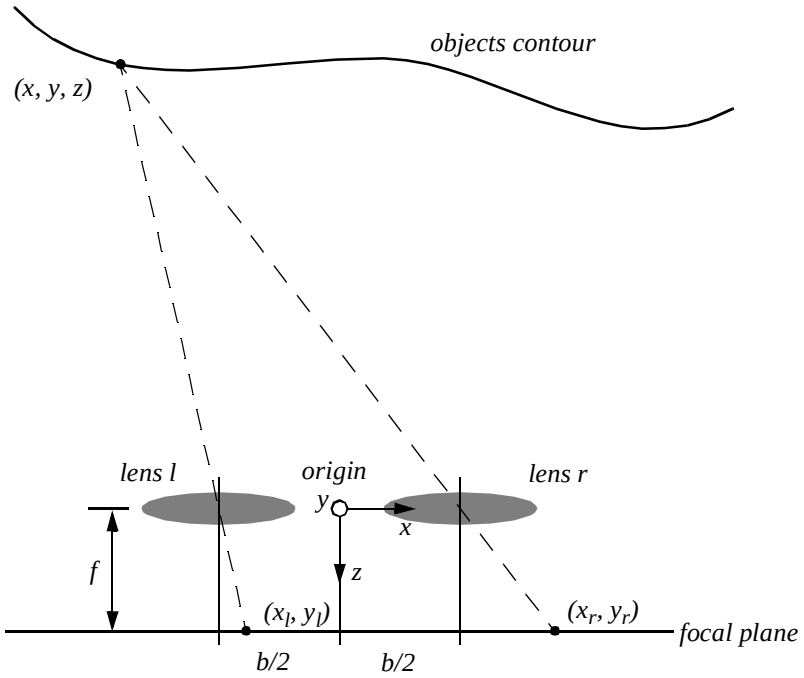


Fig 4.22 Idealized camera geometry for stereo vision

Stereo vision

Stereo vision is one of several techniques in which we recover depth information from two images that depict the scene from different perspectives. The theory of depth from stereo has been well understood for years, while the engineering challenge of creating a practical stereo sensor has been formidable [15, 16, 22]. Recent times have seen the first successes on this front, and so after presenting a basic formalism of stereo ranging, we describe the state of the art algorithmic approach and one of the recent, commercially available stereo sensors.

The geometry of stereo

First, we consider a simplified case in which two cameras are placed with their optical axes parallel, at a separation (called the *baseline*) of *b*, shown in Figure 4.22.

In this figure, a point on the object is described as being at coordinate (*x*,*y*,*z*) with respect to a central origin located between the two camera lenses. The position of this point's light rays on each camera's image is depicted in camera-specific local coordinates. Thus, the origin for the coordinate frame referenced by points of the form (x_l,y_l) is located at the center of lens *l*.

From the figure 4.22, it can be seen that

$$\frac{x_l}{f} = \frac{x + b/2}{z} \text{ and } \frac{x_r}{f} = \frac{x - b/2}{z} \tag{4.25}$$

and (out of the plane of the page)

$$\frac{y_l}{f} = \frac{y_r}{f} = \frac{y}{z} \tag{4.26}$$

where f is the distance of both lenses to the image plane. Note from equation 4.25 that:

$$\frac{x_l - x_r}{f} = \frac{b}{z} \tag{4.27}$$

where the *difference* in the image coordinates, x_l - x_r is called the *disparity*. This is an important term in stereo vision, because it is only by measuring disparity that we can recover depth information. Using the disparity and solving all three above equations provides formulae for the three dimensions of the scene point being imaged:

$$x = b\frac{(x_l + x_r)/2}{x_l - x_r} \; ; \quad y = b\frac{(y_l + y_r)/2}{x_l - x_r} \; ; \quad z = b\frac{f}{x_l - x_r} \tag{4.28}$$

Observations from these equations are as follows:

1. Distance is inversely proportional to disparity. The distance to near objects can therefore be measured more accurately than that to distant objects, just as with depth from focus techniques. In general this is alright for mobile robotics, because for navigation and obstacle avoidance closer objects are of higher importance.
2. Disparity is proportional to b. For a given disparity error, the accuracy of the depth estimate increases with increasing baseline b.
3. As b is increased, because the physical separation between the cameras is increased, some objects may appear in one camera but not in the other. Such objects by definition will not have a disparity and therefore will not be ranged successfully.
4. A point in the scene visible to both cameras produces a pair of image points (one via each lens) known as a *conjugate pair*. Given one member of the conjugate pair, we know that the other member of the pair lies somewhere along a line known as an *epipolar line*. In the case depicted by Fig. (4.22), because the cameras are perfectly aligned with one-another, the epipolar lines are horizontal lines (i.e. along the x direction).

However the assumption of perfectly aligned cameras is normally violated in practice. In order to optimize the range of distances that can be recovered, it is often useful to turn the cameras inward towards one-another for example. Figure 4.22 shows the *orientation* vectors that are necessary to solve this more general problem. We will express the position of a scene point P in terms of the reference frame of each camera separately. The reference frames of the cameras need not be aligned, and can indeed be at any arbitrary orientation relative to one-another.

For example the position of point P will be described in terms of the left camera frame as: $r'_l = (x'_l, y'_l, z'_l)$. Note that these are the coordinates of point P, not the position of its counterpart in the left camera image. P can also be described in terms of the right camera

frame as: $r'_r = (x'_r, y'_r, z'_r)$. If we have a rotation matrix R and translation matrix r_0 relating the relative positions of cameras l and r, then we can define r'_r in terms of r'_l:

$$r'_r = R \cdot r'_l + r_0 \tag{4.29}$$

where R is a 3 x 3 rotation matrix and r_0 = offset translation matrix between the two cameras. Expanding equation 4.29 yields:

$$\begin{bmatrix} x'_r \\ y'_r \\ z'_r \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{21} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} x'_l \\ y'_l \\ z'_l \end{bmatrix} + \begin{bmatrix} r_{01} \\ r_{02} \\ r_{03} \end{bmatrix} \tag{4.30}$$

The above equations have two uses:

1. We could find r'_r if we knew R, r'_l and r_0. Of course, if we knew r'_l then we would have complete information regarding the position of P relative to the left camera, and so the depth recovery problem would be solved. Note that, for perfectly aligned cameras as in Figure (4.22), $R=I$ (the identify matrix).
2. We could calibrate the system and find $r_{11}, r_{12} \ldots$ given a set of conjugate pairs $\{(x'_l, y'_l, z'_l), (x'_r, y'_r, z'_r)\}$.

In order to carry out the calibration step of step 2 above, we must find values for 12 unknowns. Therefore calibration will require 12 equations. This means that calibration requires, for a given scene, 4 conjugate points. The standard approach to calibration involves creation and use of a calibration tool, often a white cube or panel with black marks that can easily be located with simple vision algorithms. The known object is placed at several orientations to the stereo camera system and a number of conjugate points are quickly identified. Research continues on robust methods for *adaptively* adjusting calibration parameters on-the-fly.

Assuming that the calibration step is complete, we can now formalize the range recovery problem. To begin with, we do not have the position of P available, and therefore (x'_l, y'_l, z'_l) and (x'_r, y'_r, z'_r) are unknowns. Instead, by virtue of the two cameras we have pixels on the image planes of each camera, (x_l, y_l, z_l) and (x_r, y_r, z_r). Given the focal length f of the cameras we can relate the position of P to the left camera image as follows:

$$\frac{x_l}{f} = \frac{x'_l}{z'_l} \text{ and } \frac{y_l}{f} = \frac{y'_l}{z'_l} \tag{4.31}$$

Let us concentrate first on recovery of the values z'_l and z'_r. From equations 4.30 and 4.31

we can compute these values from any two of the following equations:

$$\left(r_{11}\frac{x_l}{f} + r_{12}\frac{y_l}{f} + r_{13}\right)z'_l + r_{01} = \frac{x_r}{f}z'_r \tag{4.32}$$

$$\left(r_{21}\frac{x_l}{f} + r_{22}\frac{y_l}{f} + r_{23}\right)z'_l + r_{02} = \frac{y_r}{f}z'_r \tag{4.33}$$

$$\left(r_{31}\frac{x_l}{f} + r_{32}\frac{y_l}{f} + r_{33}\right)z'_l + r_{03} = z'_r \tag{4.34}$$

The same process can be used to identify values for x' and y', yielding complete information about the position of point P. However, using the above equations requires us to have identified conjugate pairs in the left and right camera images: image points that originate at the same object point P in the scene. This fundamental challenge, identifying the conjugate pairs and thereby recovering disparity, is the *correspondence problem*. Intuitively, the problem is, given two images of the same scene from different perspectives, how can we identify the same object points in both images? For every such identified object point, we will be able to recover its 3D position in the scene.

The *correspondence problem*, or the problem of matching the same object in two different inputs, has been one of the most challenging problems in the computer vision field and the artificial intelligence field. The basic approach in nearly all proposed solutions involves converting each image in order to create more stable and more information-rich data. With this more reliable data in hand, stereo algorithms *search* for the best conjugate pairs representing as many of the images' pixels as possible.

The search process is well understood, but the quality of the resulting depth maps depends heavily upon the way in which images are treated to reduce noise and improve stability. This has been the chief technology driver in stereo vision algorithms, and one particular method has become widely used in commercially available systems.

Zero crossings of Laplacian of Gaussian

The zero crossings of Laplacian of Gaussian (ZLoG) is a strategy for identifying features in the left and right camera images that are stable and will match well, yielding high-quality stereo depth recovery. This approach has seen tremendous success in the field of stereo vision, having been implemented commercially in both software and hardware with good results. It has led to several commercial stereo vision systems and yet it is extremely simple. Here we summarize the approach and explain some of its advantages.

The core of ZLoG is the Laplacian transformation of an image. Intuitively, this is nothing more than the second derivative. Formally, the Laplacian $L(x,y)$ of an image with intensities $I(x,y)$ is defined as:

$$L(x,\ y)\ =\ \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2} \tag{4.35}$$

So the Laplacian represents the second derivative of the image, and is computed along both axes. Such a transformation, called a *convolution*, must be computed over the discrete space of image pixel values, and therefore an approximation of Equation (4.35) is required for application:

$$L\ =\ P \otimes I \tag{4.36}$$

We depict a discrete operator P, called a *kernel*, that approximates the second derivative operation along both axes as a 3 x 3 table:

$$\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \tag{4.37}$$

Application of the kernel P to convolve an image is straightforward. The kernel defines the contribution of each pixel in the image to the corresponding pixel in the target as well as its neighbors. For example, if a pixel (5,5) in the image I has value $I(5,5)=10$, then application of the kernel depicted by Equation (4.37) causes pixel $I(5,5)$ to make the following contributions to the target image L:

$L(5,5)$ += -40;

$L(4,5)$ += 10;

$L(6,5)$ += 10;

$L(5,4)$ += 10;

$L(5,6)$ += 10;

Now consider the graphical example of a step function, representing a pixel row in which the intensities are dark, then suddenly there is a jump to very bright intensities. The second derivative will have a sharp positive peak followed by a sharp negative peak, as depicted in Figure (4.23). The Laplacian is used because of this extreme sensitivity to changes in the image. But the second derivative is in fact over-sensitive. We would like the Laplacian to trigger large peaks due to real changes in the scene's intensities, but we would like to keep signal noise from triggering false peaks.

For the purpose of removing noise due to sensor error, the ZLoG algorithm applies Gaussian smoothing first, then executes the Laplacian convolution. Such smoothing can be effected via convolution with a 3 x 3 table that approximates Gaussian smoothing:

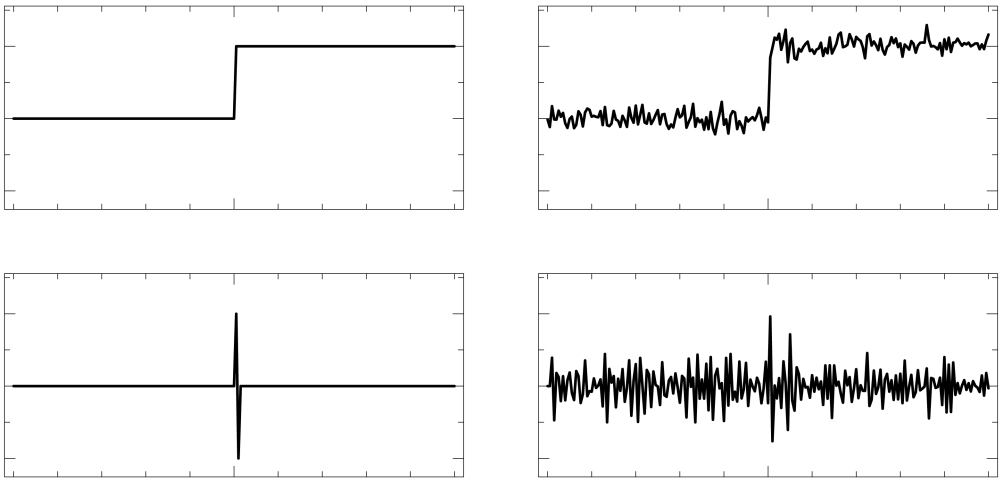


Fig 4.23 Step function example of second derivative shape and the impact of noise.

$$\begin{bmatrix} \frac{1}{16} & \frac{2}{16} & \frac{1}{16} \\ \frac{2}{16} & \frac{4}{16} & \frac{2}{16} \\ \frac{1}{16} & \frac{2}{16} & \frac{1}{16} \end{bmatrix} \tag{4.38}$$

Gaussian smoothing does not really remove error, it merely distributes image variations over larger areas. This should seem familiar. In fact, Gaussian smoothing is almost identical to the blurring caused by defocused optics. It is, nonetheless, very effective at removing high frequency noise, just as blurring removes fine-grained detail. Note that, like defocusing, this kernel does not change the total illumination but merely redistributes it (by virtue of the divisor 16).

The result of Laplacian of Gaussian (LoG) image filtering is a target array with sharp positive and negative spikes identifying boundaries of change in the original image. For example, a sharp edge in the image will result in both a positive spike and a negative spike, located on either side of the edge.

To solve the correspondence problem, we would like to identify specific *features* in LoG that are amenable to matching between the left camera and right camera filtered images. A very effective feature has been to identify each *zero crossing* of the LoG as such a feature. Many zero crossings do lie at edges in images, but their occurrence is somewhat broader than that. An interesting characteristic of zero crossings is that they are very sharply defined, covering just one "pixel" width in the filtered image. The accuracy can even be further enhanced by using interpolation to establish the position of the zero crossing with subpixel accuracy. All told, the accuracy of the zero crossing features in ZLoG have made them the preferred features in state-of-the-art stereo depth recovery algorithms.

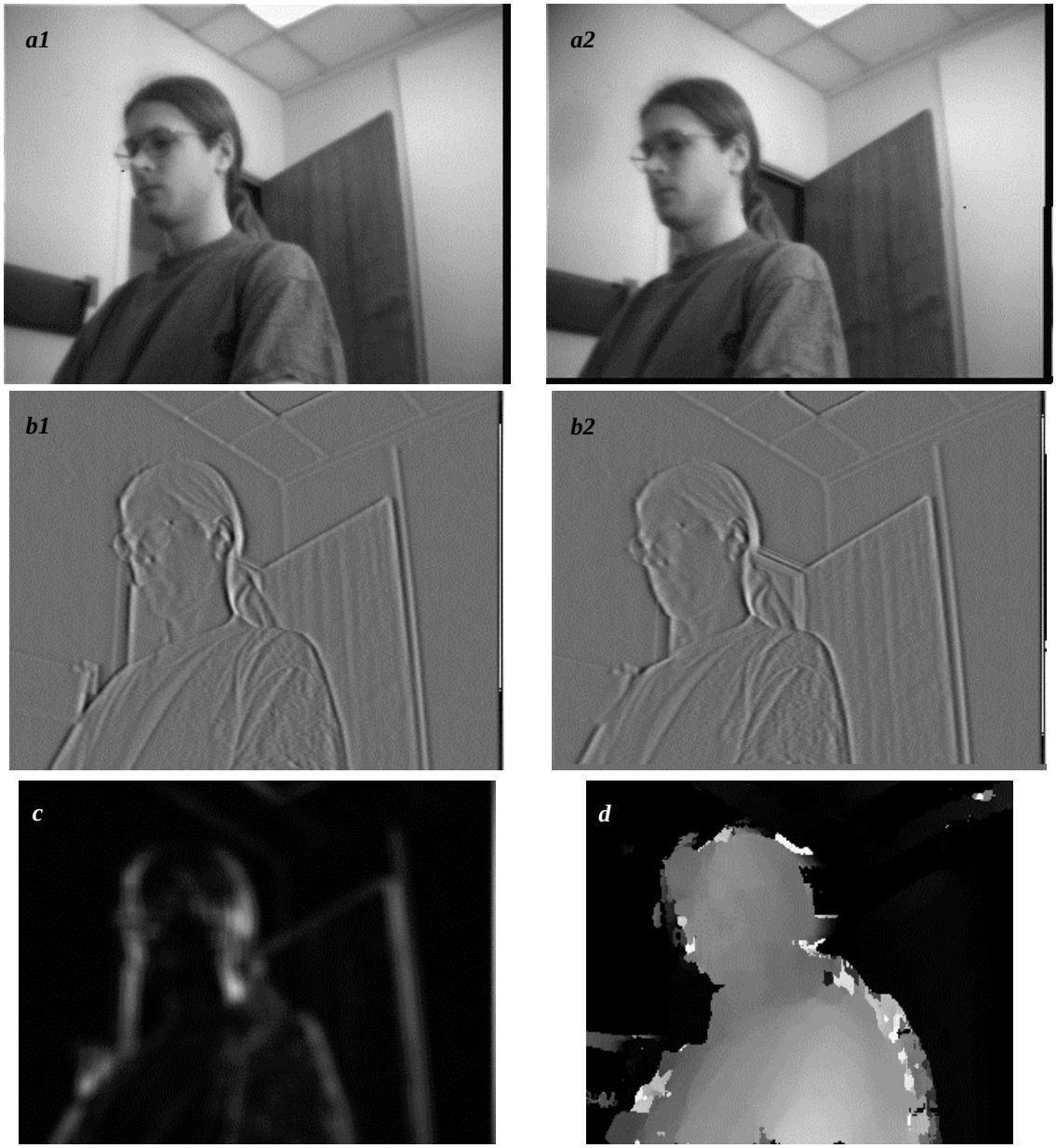


Fig 4.24 *Extracting depth information from a stereo image*
a1 *and* ***a2****: left and right image*
b1 *and* ***b2****: vertical edge filtered left and right image*
filter = [1 2 4 -2 -10 -2 4 2 1]
c*: confidence image:*
bright = high confidence (good texture)
dark = low confidence (no texture)
d*: depth image (disparity):*
bright = close, dark = far

Stereo Vision Example

Figure 4.24 shows the various steps required to extract depth information from a stereo image.

Commercial stereo vision sensors

Several commercial stereo vision depth recovery sensors have been available for researchers

Fig 4.25 The SVM module mounted on EPFL's Shrimp robot.

over the past 10 years. A popular unit in mobile robotics today is the Digital Stereo Head (or SVM) from Videre Design shown in Fig. 4.25.

The SVM uses the Laplacian of Gaussian operator, following it by tessellating the resulting array into subregions within which the sum of absolute values are computed. The correspondence problem is solved at the level of these sub-regions, a process called *area correlation* and after correspondence is solved the results are interpolated to 1/4 pixel precision. An important feature of SVM is that it produces, not just a depth map, but distinct measures of match quality for each pixel. This is valuable because such additional information can be used over time to eliminate spurious, incorrect stereo matches that have poor match quality.

The performance of SVM provides a good representative of the state of the art in stereo ranging today. The SVM consists of sensor hardware, including two CMOS cameras and DSP hardware. In addition, the SVM includes stereo vision software that makes use of a standard computer (e.g. a Pentium processor). On a 320x240 pixel image pair, the SVM assigns one of 32 discrete levels of disparity (i.e. depth) to every pixel at a rate of 12 frames per second (based on the speed of a 233 Mhz Pentium II). This compares favorably to both laser rangefinding and ultrasonics, particularly when one appreciates that ranging information with stereo is being computed for not just one target point, but all target points in the image.

It is important to note that the SVM uses CMOS chips rather than CCD chips, demonstrating that resolution sufficient for stereo vision algorithms is readily available using the less expensive, power-efficient CMOS technology.

The resolution of a vision-based ranging system will depend upon the range to the object, as we have stated before. It is instructive to observe the published resolution values for the SVM sensor. Although highly dependent on the camera optics, using a standard 6mm focal length lens pair, the SVM claims a resolution of 10mm at 3 meters range, and a resolution of 60mm at 10 meters range. These values are based on ideal circumstances, but neverthe-

less exemplify the rapid loss in resolution that will accompany vision-based ranging.

4.1.8.3 Motion and Optical Flow

A great deal of information can be recovered by recording time varying images from a fixed (or moving) camera. Firstly we distinguish between the motion field and optical flow:

- Motion field: this assigns a velocity vector to every point in an image. If a point in the environment moves with velocity v_o, then this induces a velocity v_i in the image plane. It is possible to determine mathematically the relationship between v_i and v_o.
- Optical flow: it can also be true that brightness patterns in the image move as the object that causes them moves (light source). Optical flow is the apparent motion of these brightness patterns.

In our analysis here we assume that the optical flow pattern will correspond to the motion field, although this is not always true in practice. This is illustrated in figure 4.26a where a sphere exhibits spatial variation of brightness, or shading, in the image of the sphere since its surface is curved. If the surface moves however, this shading pattern will not move - hence the optical flow is zero everywhere even though the motion field is not zero. In figure 4.26a, the opposite occurs. Here we have a fixed sphere with a moving light source. The shading in the image will change as the source moves. In this case the optical flow is nonzero but the motion field is zero. If the only information accessible to us is the optical flow and we depend on this, we will get false results.

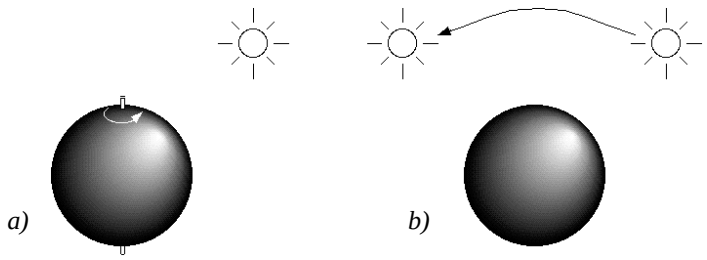


Fig 4.26 *Motion of the sphere or the light source here demonstrates that optical flow is not always the same as the motion field.*

Optical Flow

Let $E(x, y, t)$ be the image irradiance at time t at the image point (x, y). If $u(x, y)$ and $v(x, y)$ are the x and y components of the optical flow vector at that point, we need to search a new image for a point where the irradiance will be the same at time $t+\delta t$, i.e.: at point $(x+\delta x, y+\delta y)$, where $\delta x = u\delta t$ and $\delta x = u\delta t$. i.e:

$$E(x + u\delta t, y + v\delta t,\ t + \delta t) = E(x,\ y,\ t) \tag{4.39}$$

for a small time interval, δt. From this single constraint u and v cannot be determined uniquely. We therefore use the fact that the optical flow field should be continuous almost everywhere. Hence if brightness varies smoothly with x, y and t we can expand the left hand side

of equation 4.39 as a Taylor series to obtain:

$$E(x,\ y,\ t) + \delta x\frac{\partial E}{\partial x} + \delta y\frac{\partial E}{\partial y} + \delta t\frac{\partial E}{\partial t} + e\ =\ E(x,\ y,\ t) \tag{4.40}$$

where e contains second and higher order terms in δx etc. In the limit as δt tends to zero we obtain:

$$\frac{\partial E}{\partial x}\frac{dx}{dt} + \frac{\partial E}{\partial y}\frac{dy}{dt} + \frac{\partial E}{\partial t}\ =\ 0 \tag{4.41}$$

from which we can abbreviate:

$$u\ =\ \frac{dx}{dt}\ ;\quad v\ =\ \frac{dy}{dt} \tag{4.42}$$

and

$$E_x\ =\ \frac{\partial E}{\partial x}\ ;\quad E_y\ =\ \frac{\partial E}{\partial y};\quad E_t\ =\ \frac{\partial E}{\partial t}\ =\ 0 \tag{4.43}$$

so that we obtain:

$$E_x u + E_y v + E_t\ =\ 0 \tag{4.44}$$

The derivatives E_x, E_y and E_t are estimated from the image. Equation 4.44 is known as *the optical flow constraint equation.*

It can be seen that equation 4.44 represents a straight line in velocity (u, v) space. Hence a local measurement of the three derivatives E_x … etc. can only identify this line and not unique values for u and v. We therefore introduce an additional constraint.

Smoothness of the Optical Flow

We now make the assumption that optical flow patterns are smooth. We can do this mathematically by finding a measure of departure from smoothness:

$$e_s\ =\ \int\int\ (u^2 + v^2)dxdy \tag{4.45}$$

which is the integral of the square of the magnitude of the gradient of the optical flow. We also determine the error in the optical flow constraint equation (which in practice will not quite be zero).

$$e_c\ =\ \int\int\ (E_x u + E_y v + E_t)^2 dxdy \tag{4.46}$$

Both of these equations should be as small as possible so we want to minimize $e_s + \lambda e_c$, where λ is a parameter that weights the error in the image motion equation relative to the departure from smoothness. A large parameter should be used if the brightness measurements are accurate and small if they are noisy.

This problem them amounts to the calculus of variations, and the Euler equations yield:

$$\nabla^2 u = \lambda(E_x u + E_y v + E_t)E_x \tag{4.47}$$

$$\nabla^2 v = \lambda(E_x u + E_y v + E_t)E_y \tag{4.48}$$

where:

$$\nabla^2 = \frac{\partial^2}{\delta x^2} + \frac{\partial^2}{\delta y^2} \tag{4.49}$$

which is the Laplacian operator.

Equation 4.47 and 4.48 form a pair of elliptical second order partial differential equations which can be solved iteratively.

Discontinuities in Optical Flow

Where silhouettes (one object occluding another) occur, discontinuities in the optical flow will occur. We should try to find these points to exclude them before the method above joins them with a smooth solution.

This is done by incorporating segmentation into the iterative solutions to equations 4.47 and 4.48 above. After each iteration we look for regions where the flow changes rapidly. At these points, we inhibit the next iteration of the above equations from smoothly connecting the solution across these points.

4.1.8.4 Color tracking sensors

Although depth from stereo will doubtless prove to be a popular application of vision-based methods to mobile robotics, it mimics the functionality of existing sensors, including ultrasonic, laser and optical rangefinders. An important aspect of vision-based sensing is that the vision chip can provide sensing modalities and cues that no other mobile robot sensor provides. One such novel sensing modality is detecting and tracking color in the environment.

Color represents an environmental characteristic that is orthogonal to range, and it represents both a natural cue and an artificial cue that can provide new information to a mobile robot. For example, the annual robot soccer events make extensive use of color both for environmental marking and for robot localization (see Fig. 4.27).

Color sensing has two important advantages. First, detection of color is a straightforward function of a single image, therefore no correspondence problem need be solved in such al-

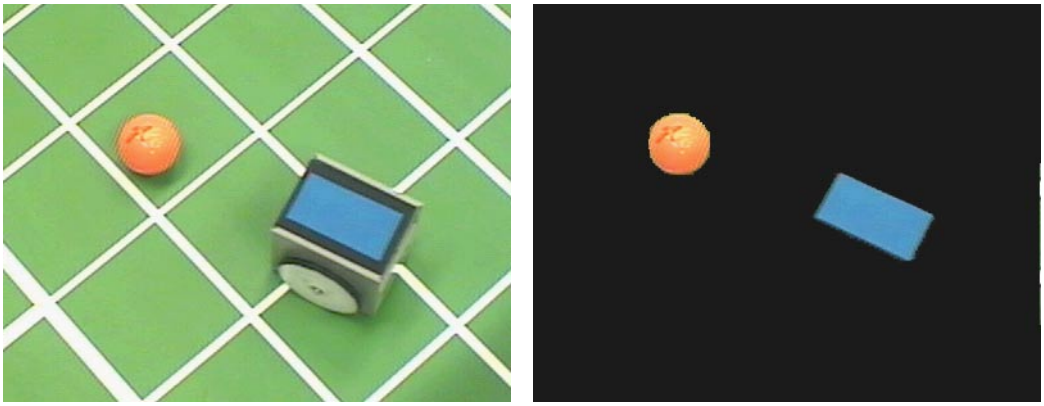


Fig 4.27 *Color markers on the top of EPFL's STeam Engine soccer robots enable a color-tracking sensor to locate the robots and the ball in the soccer field.*

gorithms. Second, because color sensing provides a new, independent environmental cue, if it is combined (i.e. *sensor fusion*) with existing cues, such as data from stereo vision or laser rangefinding we can expect significant information gains.

Efficient color tracking sensors are now available commercially. Below, we briefly describe two commercial, hardware-based color-tracking sensors as well as a publicly available software-based solution.

Cognachrome color tracking system

The Cognachrome Vision System is a color-tracking hardware-based sensor capable of extremely fast color tracking on a dedicated processor [109]. The system will detect color blobs based on 3 user-defined colors at a rate of 60 Hz. The Cognachrome system can detect and report on a maximum of 25 objects per frame, providing centroid, bounding box, area, aspect ratio and principal axis orientation information for each object independently.

This sensor uses a technique called *constant thresholding* to identify each color. In *RGB* space, the user defines for each of *R*, *G* and *B* a minimum and maximum value. The three-dimensional box defined by these six constraints forms a color bounding box, and any pixel with *RGB* values that are all within this bounding box is identified as a target. Target pixels are merged into larger objects that are then reported to the user.

The Cognachrome sensor achieves a position resolution of one pixel for the centroid of each object in a field that is 200 x 250 pixels in size. The key advantage of this sensor, just as with laser rangefinding and ultrasonics, is that there is no load on the mobile robot's main processor due to the sensing modality. All processing is performed on sensor-specific hardware (i.e. a Motorola 68332 processor and a mated framegrabber). The Cognachrome system costs several thousand dollars, but is being superceded by higher-performance hardware vision processors at Newton Labs, Inc.

CMUcam robotic vision sensor

Recent advances in chip manufacturing, both in terms of CMOS imaging sensors and high-speed, readily available microprocessors at the 50+ MHz range, have made it possible to

Fig 4.28 *The CMUcam sensor consists of 3 chips: a CMOS imaging chip, a SX28 microprocessor and a Maxim RS232 level shifter. Roland, cmucam.jpg!*

Fig 4.29 *Color-based object extraction as applied to a human hand. Roland, cmucamhand1.jpg and cmucamhand2.jpg*

manufacture low-overhead intelligent vision sensors with functionality similar to Cognachrome for a fraction of the cost. The CMUcam sensor is a recent system that mates a low-cost microprocessor with a consumer CMOS imaging chip to yield an intelligent, self-contained vision sensor for $100, as shown in Figure 4.29.

This sensor is designed to provide high-level information extracted from the camera image to an external processor that may, for example, control a mobile robot. An external processor configures the sensor's streaming data mode, for instance specifying tracking mode for a bounded RGB or YUV value set. Then, the vision sensor processes the data in real time and outputs high-level information to the external consumer. At less than 150 milliamps of current draw, this sensor provides image color statistics and color tracking services at approximately 20 frames per second at a resolution of 80 x 143 [Roland, reference here new reference in my text notes, Anthony Rowe et al.].

Figure 4.29 demonstrates the color-based object tracking service as provided by CMUcam once the sensor is trained on a human hand. The approximate shape of the object is extracted as well as its bounding box and approxiamate Center of Mass.

CMVision color tracking software library

Because of the rapid speedup of processor in recent times, there has been a trend towards executing basic vision processing on a main processor within the mobile robot. Intel Corporation's Computer Vision library is an optimized library for just such processing [106]. In this spirit, the CMVision color tracking software represents a state-of-the-art software solution for color tracking in dynamic environments [97]. CMVision can track up to 32 colors at 30 Hz on a standard 200 MHz pentium computer.

The basic algorithm this sensor uses is constant thresholding, as with Cognachrome, with the chief difference that the YUV color space is used instead of the RGB color space when defining a 6-constraint bounding box for each color. While R, G and B values encode the intensity of each color, YUV separates the color (or *chrominance*) measure from the brightness (or *luminosity*) measure. Y represents the image's luminosity while U and V together capture its chrominance. Thus, a bounding box expressed in YUV space can achieve greater stability with respect to changes in illumination than is possible in RGB space.

The CMVision color sensor achieves a resolution of 160 x 120 and returns, for each object detected, a bounding box and a centroid. The software for CMVision is available freely with a Gnu Public License at [108].

Key performance bottlenecks for both the CMVision software, the CMUcam hardware system and the Cognachrome hardware system continue to be the quality of imaging chips and available computational speed. As significant advances are made on these frontiers one can expect packaged vision systems to witness tremendous performance improvements.

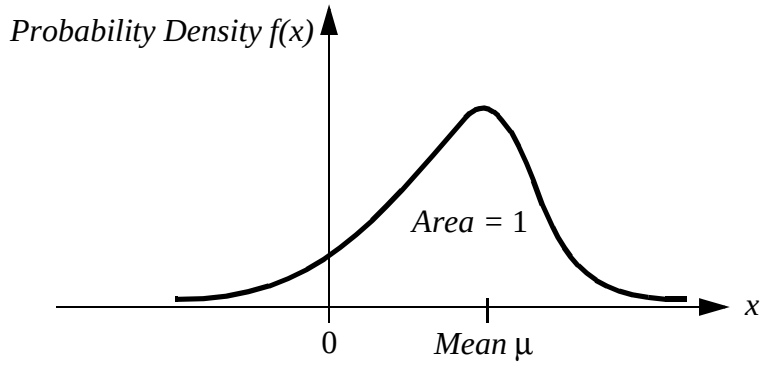


Fig 4.30 *A sample probability density function, showing a single probability peak (i.e. unimodal) with asymptotic drops in both directions.*

4.2 Representing Uncertainty

In Section (4.1.2) we presented a terminology for describing the performance characteristics of a sensor. As mentioned there, sensors are imperfect devices with errors of both the systematic and random nature. Random errors, in particular, cannot be corrected, and so they represent atomic levels of sensor uncertainty.

But when you build a mobile robot, you combine information from many sensors, even using the same sensors repeatedly, over time, to possibly build a model of the environment. How can we scale up, from characterizing the uncertainty of a single sensor to the uncertainty of the resulting robot system?

We begin by presenting a statistical representation for the random error associated with an individual sensor [12]. With a quantitative tool in hand, the standard Gaussian uncertainty model can be presented and evaluated. Finally, we present a framework for computing the uncertainty of conclusions drawn from a set of quantifiably uncertain measurements, known as the *error propagation law*.

4.2.1 Statistical representation

We have already defined *error* as the difference between a sensor measurement and the true value. From a statistical point of view, we wish to characterize the error of a sensor, not for one specific measurement but for any measurement. Let us formulate the problem of sensing as an estimation problem. The sensor has taken a set of n measurements with values ρ_i. The goal is to characterize the estimate of the true value $E[X]$ given these measurements:

$$E[X] = g(\rho_1, \rho_2, \ldots, \rho_n) \tag{4.50}$$

From this perspective, the true value is represented by a random (and therefore unknown) variable X. We use a *probability density function* to characterize the statistical properties of the value of X.

In figure 4.30, the density function identifies, for each possible value x of X a probability density $f(x)$ along the y-axis. The area under the curve is 1, indicating the complete chance

of X having *some* value:

$$\int_{-\infty}^{\infty} f(x)dx = 1 \tag{4.51}$$

The probability of the value of X falling between two limits a and b is computed as the bounded integral:

$$P[a < X \leq b] = \int_{a}^{b} f(x)dx \tag{4.52}$$

The probability density function is a useful way to characterize the possible values of X because it not only captures the range of X but also the comparative probability of different values for X. Using $f(x)$ we can quantitatively define the mean, variance and standard deviation as follows.

The *mean value* μ is equivalent to the expected value $E[X]$ if we were to measure X an infinite number of times and average all of the resulting values. We can easily define $E[X]$:

$$\mu = E[X] = \int_{-\infty}^{\infty} xf(x)dx \tag{4.53}$$

Note in the above equation that calculation of $E[X]$ is identical to the weighted average of all possible values of x. In contrast, the *mean square value* is simply the weighted average of the squares of all values of x:

$$E[X^2] = \int_{-\infty}^{\infty} x^2 f(x)dx \tag{4.54}$$

Characterization of the "width" of the possible values of X is a key statistical measure, and this requires first defining the *variance* σ^2:

$$Var(X) = \sigma^2 = \int_{-\infty}^{\infty} (x-\mu)^2 f(x)dx \tag{4.55}$$

Finally, the *standard deviation* σ is simply the square root of variance. σ and σ^2 will play important roles in our characterization of the error of a single sensor as well as the error of a model generated by combining multiple sensor readings.

Independence of random variables

With the tools presented above, we will often evaluate systems with multiple random variables. For instance, a mobile robot's laser rangefinder may be used to measure the position

of a feature on the robot's right and, later, another feature on the robot's left. The position of each feature in the real world may be treated as a random variable, X_1 and X_2.

Two random variables X_1 and X_2 are *independent* if the particular value of one has no bearing on the particular value of the other. In this case we can draw several important conclusions about the statistical behavior of X_1 and X_2. First, the expected value (or mean value) of the product of random variables is equal to the product of their mean values:

$$E[X_1X_2] = E[X_1]E[X_2] \tag{4.56}$$

Second, the variance of their sums is equal to the sum of their variances:

$$Var(X_1 + X_2) = Var(X_1) + Var(X_2) \tag{4.57}$$

In mobile robotics, we will often assume the independence of random variables even when this assumption is not strictly true. The simplification that results makes a number of the existing mobile robot mapping and navigation algorithms tenable, as described in Chapter 5. A further simplification, described in the next sub-section, revolves around one specific probability density function used more often than any other when modeling error: the Gaussian distribution.

4.2.2 Gaussian distribution

The Gaussian distribution, also called the *normal distribution* is used across engineering disciplines when a well-behaved error model is required for a random variable for which no error model of greater felicity has been discovered. The Gaussian has many characteristics that make it mathematically advantageous to other *ad hoc* probability density functions. It is symmetric around the mean μ. There is no particular bias for being larger than or smaller than μ, and this makes sense when there is no information to the contrary. The Gaussian distribution is also unimodal, with a single peak that reaches a maximum at μ (necessary for any symmetric, unimodal distribution). This distribution also has tails (the value of $f(x)$ as x approaches $-\infty$ and ∞) that only approach 0 asymptotically. This means that all amounts of error are possible, although very large errors may be highly improbable. In this sense, the Gaussian is conservative. Finally, as seen in the formula for the Gaussian probability density function, the distribution depends only on two parameters:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{4.58}$$

The Gaussian's basic shape is determined by the structure of this formula, and so the only two parameters required to fully specify a particular Gaussian are its mean μ and its standard deviation σ. Figure 4.31 shows the Gaussian function with $\mu = 0$ and $\sigma = 1$.

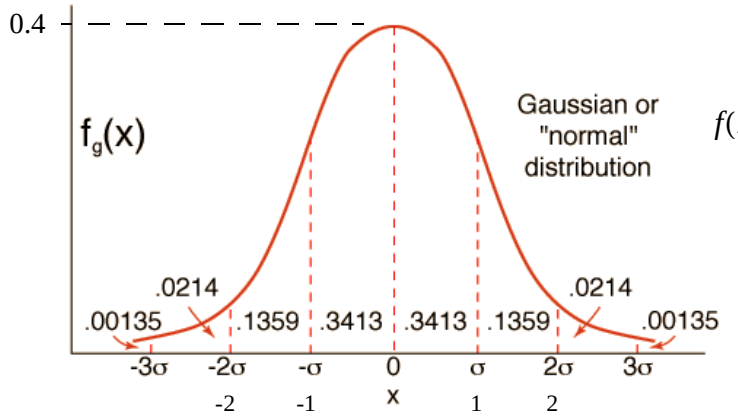


Fig 4.31 *The Gaussian function with* $\mu = 0$ and $\sigma = 1$. We shall refer to this as the Reference Gaussian. The value 2σ is often refereed for the signal quality. 95.44% of the values are falling within $\pm 2\sigma$.

Suppose that a random variable X is modeled as a Gaussian. How does one identify the chance that the value of X is within one standard deviation of μ? In practice, this requires integration of $f(x)$, the Gaussian function to compute the area under a portion of the curve:

$$Area = \int_{-\sigma}^{\sigma} f(x)dx \tag{4.59}$$

Unfortunately, there is no closed-form solution for the integral in Equation (4.59), and so the common technique is to use a Gaussian *cumulative probability table*. Using such a table, one can compute the probability for various value ranges of X:

$$P[\mu - \sigma < X \leq \mu + \sigma] = 0.68$$

$$P[\mu - 2\sigma < X \leq \mu + 2\sigma] = 0.95$$

$$P[\mu - 3\sigma < X \leq \mu + 3\sigma] = 0.997$$

For example, 95% of the values for X fall within two standard deviations of its mean. This applies to *any* Gaussian distribution. As is clear from the above progression, under the Gaussian assumption once bounds are relaxed to 3σ, the overwhelming proportion of values (and, therefore, probability) is subsumed.

4.2.3 Error propagation: combining uncertain measurements

The probability mechanisms above may be used to describe the errors associated with a single sensor's attempts to measure a real-world value. But in mobile robotics, one often uses a series of measurements, all of them uncertain, to extract a single environmental measure. For example, a series of uncertain measurements of single points can be fused to extract the position of a line (e.g. a hallway wall) in the environment (fig. 4.36).

Consider the system in figure 4.32, where X_i are n input signals with a known probability distribution and Y_j are m outputs. The question of interest is: what can we say about the prob-

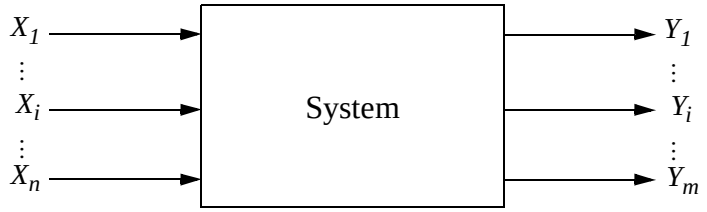


Fig 4.32 *Error propagation in a multiple-input multi-output system with n inputs and m outputs.*

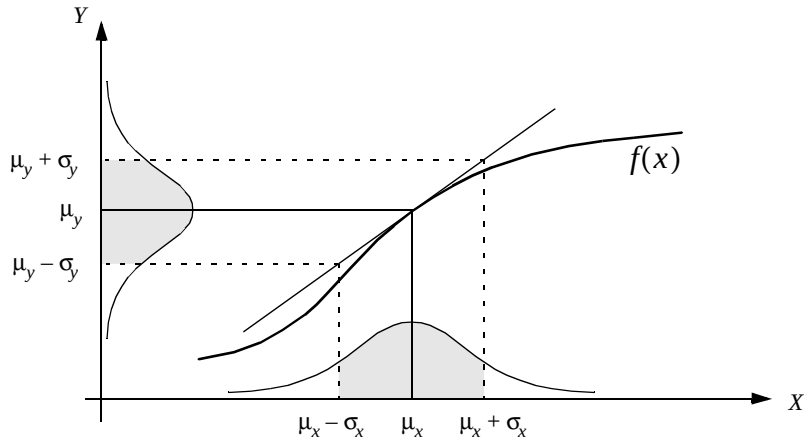


Fig 4.33 *One-dimensional case of a nonlinear error propagation problem*

ability distribution of the output signals Y_j if they depend with known functions f_j upon the input signals? Figure 4.33 depicts the one-dimensional version of this error propagation problem as an example.

The general solution can be generated using the first order Taylor expansion of f_j. The output covariance matrix C_Y is given by the error propagation law:

$$C_Y = F_X C_X F_X^T \tag{4.60}$$

where

C_X: covariance matrix representing the input uncertainties

C_Y: covariance matrix representing the propagated uncertainties for the outputs.

F_X is the *Jacobian* matrix defined as:

$$F_X = \nabla f = \left[\nabla_X \cdot \; f(X)^T\right]^T = \begin{bmatrix} f_1 \\ \vdots \\ f_m \end{bmatrix} \left[\frac{\partial}{\partial X_1} \cdots \frac{\partial}{\partial X_n}\right] = \begin{bmatrix} \frac{\partial f_1}{\partial X_1} & \cdots & \frac{\partial f_1}{\partial X_n} \\ \vdots & \cdots & \vdots \\ \frac{\partial f_m}{\partial X_1} & \cdots & \frac{\partial f_m}{\partial X_n} \end{bmatrix} \tag{4.61}$$

This is also the transpose of the gradient of $f(X)$.

We will not present a detailed derivation here (see Appendix A) but will use Equation 4.60 in order to solve an example problem in Section 4.3.1.1.

4.3 Feature Extraction

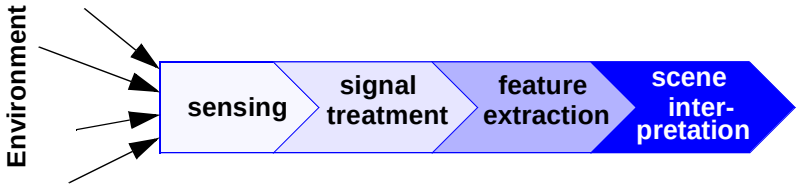


Fig 4.34 The Perceptual Pipeline: From sensor readings to knowledge models

An autonomous mobile robot must be able to determine its relationship to the environment by making measurements with its sensors and then using those measured signals. A wide variety of sensing technologies are available, as shown in the previous section. But every sensor we have presented is imperfect: measurements always have error and, therefore, uncertainty associated with them. Therefore, sensor inputs must be used in a way that enables the robot to interact with its environment successfully in spite of measurement uncertainty.

There are two strategies for using uncertain sensor input to guide the robot's behavior. One strategy is to use each sensor measurement as a raw and individual value. Such raw sensor values could for example be tied directly to robot behavior, whereby the robot's actions are a function of its sensor inputs. Alternatively, the raw sensors values could be used to update an intermediate model, with the robot's actions being triggered as a function of this model rather than the individual sensor measurements.

The second strategy is to extract information from one or more sensor readings first, generating a higher-level *percept* that can then be used to inform the robot's model and perhaps the robot's actions directly. We call this process *feature extraction*, and it is this next, optional step in the perceptual interpretation pipeline (Fig. 4.34) that we will now discuss.

In practical terms, mobile robots do not necessarily use *feature extraction* and *scene interpretation* for every activity. Instead, robots will interpret sensors to varying degrees depending on each specific functionality. For example, in order to guarantee emergency stops in the face of immediate obstacles, the robot may make direct use of raw forward-facing range readings to stop its drive motors. For local obstacle avoidance, raw ranging sensor strikes may be combined in an occupancy grid model, enabling smooth avoidance of obstacles meters away. For map-building and precise navigation, the range sensor values and even vision sensor measurements may pass through the complete perceptual pipeline, being subjected to feature extraction followed by scene interpretation to minimize the impact of individual sensor uncertainty on the robustness of the robot's map-making and navigation skills. The pattern that thus emerges is that, as one moves into more sophisticated, long-term perceptual tasks, the feature extraction and scene interpretation aspects of the perceptual pipeline become essential.

Feature: Definition

Features are recognizable structures of elements in the environment. They usually can be extracted from measurements and mathematically described. Good features are always per-

ceivable and easily detectable from the environment. We distinguish between *low-level features* (*geometric primitives*) like lines, circles or polygons and *high-level features* (*objects*) such as edges, doors, tables or a trash can. At one extreme, raw sensor data provides a large volume of data, but with low distinctiveness of each individual quantum of data. Making use of raw data has the potential advantage that every bit of information is fully used, and thus there is a high conservation of information. Low level features are abstractions of raw data, and as such provide a lower volume of data while increasing the distinctiveness of each feature. The hope, when one incorporates low level features, is that the features are filtering out poor or useless data, but of course it is also likely that some valid information will be lost as a result of the feature extraction process. High level features provide maximum abstraction from the raw data, thereby reducing the volume of data as much as possible while providing highly distinctive resulting features. Once again, the abstraction process has the risk of filtering away important information, potentially lowering data utilization.

Although features must have some spatial locality, their geometric extent can range widely. For example, a corner feature inhabits a specific coordinate location in the geometric world. In contract, a visual "fingerprint" identifying a specific room in an office building applies to the entire room, but has a location that is spatially limited to the one, particular room.

In mobile robotics, features play an especially important role in the creation of environmental models. They enable more compact and robust descriptions of the environment, helping a mobile robot during both map-building and localization. When designing a mobile robot, a critical decision revolves around choosing the appropriate features for the robot to use. A number of factors are essential to this decision:

Target environment. For geometric features to be useful, the target geometries must be readily detected in the actual environment. For example, line features are extremely useful in office building environments due to the abundance of straight walls segments while the same feature is virtually useless when navigating Mars.

Available sensors. Obviously the specific sensors and sensor uncertainty of the robot impacts the appropriateness of various features. Armed with a laser rangefinder, a robot is well qualified to use geometrically detailed features such as corner features due to the high quality angular and depth resolution of the laser scanner. In contrast, a sonar-equipped robot may not have the appropriate tools for corner feature extraction.

Computational power. Vision-based feature extraction can effect a significant computational cost, particularly in robots where the vision sensor processing is performed by one of the robot’s main processors.

Environment representation. Feature extraction is an important step toward scene interpretation, and by this token the features extracted must provide information that is consonant with the representation used for the environment model. For example, non-geometric vision-based features are of little value in purely geometric environment models but can be of great value in topological models of the environment. Figure 4.35 shows the application of two different representations to the task of modeling an office building hallway. Each approach has advantages and disadvantages, but extraction of line and corner features has much more relevance to the representation on the left. Refer to Chapter 5, Section 5.5 for a

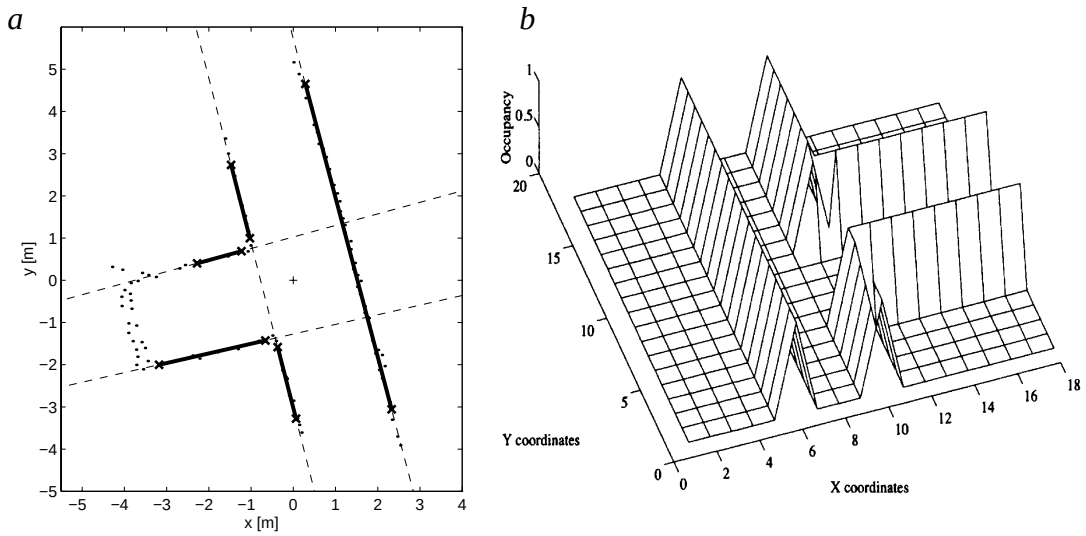


Fig 4.35 *Environment representation and modeling: a) feature based (continuos metric); b) occupancy grid (discrete metric);*

close look at map representations and their relative tradeoffs.

In the following two sections, we present specific feature extraction techniques based on the two most popular sensing modalitites of mobile robotics: range sensing and visual appearance-based sensing.

4.3.1 Feature extraction based on range data (laser, ultrasonic, vision-based ranging)

Most of today's features extracted from ranging sensors are geometric primitives such as line segments or circles. The main reason for this is that for most other geometric primitives the parametric description of the features becomes too complex and no closed form solution exists. Here we will describe line extraction in detail, demonstrating how the uncertainty models presented above can be applied to the problem of combining multiple sensor measurements. Afterwards, we briefly present another very successful feature for indoor mobile robots, the corner feature, and demonstrate how these features can be combined in a single representation.

4.3.1.1 Line Extraction

Geometric feature extraction is usually the process of comparing and matching measured sensor data against a predefined description, or template, of the expect feature. Usually, the system is overdetermined in that the number of sensor measurements exceeds the number of feature parameters to be estimated. Since the sensor measurements all have some error, there is no perfectly consistent solution and, instead, the problem is one of optimization. One can, for example, extract the feature that minimizes the discrepancy with all sensor measurements used (e.g. least squares estimation).

In this section we present an optimization-based solution to the problem of extracting a line feature from a set of uncertain sensor measurements. For greater detail than is presented be-

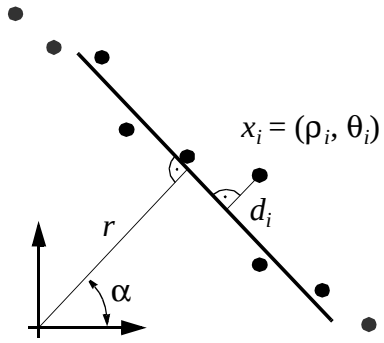


Fig 4.36 *Estimating a line in the least squares sense. The model parameters r (length of the perpendicular) and α (its angle to the abscissa) uniquely describe a line.*

low, refer to [19], pp. 15 and 221.

Probabilistic line extraction from uncertain range sensor data

Our goal is to extract a line feature based on a set of sensor measurements as shown in Figure 4.36. There is uncertainty associated with each of the noisy range sensor measurements, and so there is no single line that passes through the set. Instead, we wish to select the best possible match, given some optimization criterion.

More formally, suppose *n* ranging measurement points in polar coordinates $x_i = (\rho_i, \theta_i)$ are produced by the robot's sensors. We know that there is uncertainty associated with each measurement, and so we can model each measurement using two random variables $X_i = (P_i, Q_i)$. In this analysis we assume that uncertainty with respect to the actual value of *P* and *Q* are independent. Based on Equation (4.56) we can state this formally:

$$E[P_i \cdot P_j] = E[P_i]E[P_j] \qquad \forall i, j = 1, \ldots, n \tag{4.62}$$

$$E[Q_i \cdot Q_j] = E[Q_i]E[Q_j] \qquad \forall i, j = 1, \ldots, n \tag{4.63}$$

$$E[P_i \cdot Q_j] = E[P_i]E[Q_j] \qquad \forall i, j = 1, \ldots, n \tag{4.64}$$

Furthermore, we will assume that each random variable is subject to a Gaussian probability density curve, with a mean at the true value and with some specified variance:

$$P_i \sim N(\rho_i, \sigma_{\rho_i}^2) \tag{4.65}$$

$$Q_i \sim N(\theta_i, \sigma_{\theta_i}^2) \tag{4.66}$$

Given some measurement point (ρ, θ), we can calculate the corresponding Euclidean coordinates as $x = \rho\cos\theta$ and $y = \rho\sin\theta$. If there were no error, we would want to find a line for which all measurements lie on that line:

$$\rho\cos\theta\cos\alpha + \rho\sin\theta\sin\alpha - r = \rho\cos(\theta - \alpha) - r = 0 \tag{4.67}$$

Of course there is measurement error, and so this quantity will not be zero. When it is non-zero, this is a measure of the error between the measurement point (ρ, θ) and the line, specifically in terms of the minimum orthogonal distance between the point and the line. It is always important to understand how the error that shall be minimized is being measured. For example a number of line extraction techniques do not minimize this orthogonal point-line distance, but instead the distance parallel to the y-axis between the point and the line. A good illustration of the variety of optimization criteria is available in [18] where several algorithms for fitting circles and ellipses are presented which minimize algebraic and geometric distances.

For each specific (ρ_i, θ_i), we can write the orthogonal distance d_i between (ρ_i, θ_i) and the line as:

$$\rho_i\cos(\theta_i - \alpha) - r = d_i. \tag{4.68}$$

If we consider each measurement to be equally uncertain, we can sum the square of all errors together, for all measurement points, to quantify an overall fit between the line and all of the measurements:

$$S = \sum_i d_i^2 = \sum_i (\rho_i\cos(\theta_i - \alpha) - r)^2 \tag{4.69}$$

Our goal is to minimize S when selecting the line parameters (α, r). We can do so by solving the nonlinear equation system

$$\frac{\partial S}{\partial \alpha} = 0 \qquad \frac{\partial S}{\partial r} = 0. \tag{4.70}$$

The above formalism is considered an *unweighted least squares* solution because no distinction is made from among the measurements. In reality, each sensor measurement may have its own, unique uncertainty based on the geometry of the robot and environment when the measurement was recorded. For example, we know with regards to vision-based stereo ranging that uncertainty and, therefore, variance increases as a square of the distance between the robot and the object. To make use of the variance σ_i^2 that models the uncertainty regarding distance ρ_i of a particular sensor measurement, we compute an individual weight w_i for each measurement using the formula:

$$w_i = 1/\sigma_i^2 1. \tag{4.71}$$

Then, equation (4.69) becomes

$$S = \sum w_i d_i^2 = \sum w_i (\rho_i \cos(\theta_i - \alpha) - r)^2 . \tag{4.72}$$

It can be shown that the solution of (4.70) in the *weighted* least square sense[2] is:

$$\alpha = \frac{1}{2}\text{atan}\left(\frac{\sum w_i \rho_i^2 \sin 2\theta_i - \frac{2}{\sum w_i}\sum\sum w_i w_j \rho_i \rho_j \cos\theta_i \sin\theta_j}{\sum w_i \rho_i^2 \cos 2\theta_i - \frac{1}{\sum w_i}\sum\sum w_i w_j \rho_i \rho_j \cos(\theta_i + \theta_j)}\right) \tag{4.73}$$

$$r = \frac{\sum w_i \rho_i \cos(\theta_i - \alpha)}{\sum w_i} \tag{4.74}$$

In practice equation (4.73) uses the four-quadrant arc tangent (atan2)[3].

Let us demonstrate Equations (4.73) and (4.74) with a concrete example. The 10 measurements (ρ_i, θ_i) in table 4.1 have been taken with a laser range sensor installed on a mobile robot. We assume that the uncertainties of all measurements are equal, uncorrelated and that

1. The issue of determining an adequate weight when σ_i is given (and perhaps some additional information) is complex in general and beyond the scope of this text. See [20] for a careful treatment.
2. We follow here the notation of [19] and distinguish a weighted least squares problem if C_X is diagonal (input errors are mutually independent) and a generalized least squares problem if C_X is non-diagonal.
3. Atan2 computes $\text{an}(x/y)^{-1}$ but uses the signs of both x and y to determine the quadrant in with the resulting angles lies. For example $\tan 2(-2, -2) = -135°$, whereas $\tan 2(2, 2) = -45°$, a distinction which would be lost with a single-argument arc tangent function.

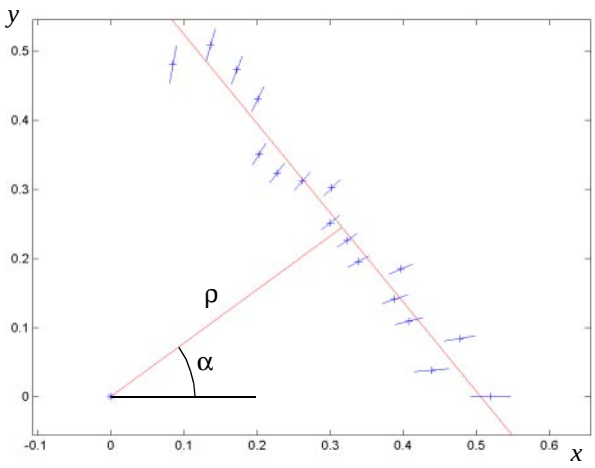


Fig 4.37 *Extracted line from laser range measurements (+). The small lines at each measurement point represent the measurement uncertainty σ that is proportional to square of the measurement distance.*

the robot was static during the measurement process.

Table 4.1: Measured values

pointing angle of sensor θ_i [deg]	range ρ_i [m]
0	0.5197
5	0.4404
10	0.4850
15	0.4222
20	0.4132
25	0.4371
30	0.3912
35	0.3949
40	0.3919
45	0.4276
50	0.4075
55	0.3956
60	0.4053
65	0.4752
70	0.5032
75	0.5273
80	0.4879

Direct application of the above solution equations yields the line defined by $\alpha = 37.36$ and $r = 0.4$. This line represents the best fit in a least square sense and is shown visually in Fig. 4.37.

Propagation of uncertainty during line extraction

Returning to the subject of Section (4.2.3), we would like understand how the uncertainties of specific range sensor measurements propagate to govern the uncertainty of the extracted line. In other words, how does uncertainty in ρ_i and θ_i propagate in Equations (4.73) and (4.74) to affect the uncertainty of α and r?

This requires direct application of Equation 4.60 with A and R representing the random out-

puts variables of $\alpha$and r respectively. The goal is to derive the 2×2 output covariance matrix

$$C_{AR} = \begin{bmatrix} \sigma_A^2 & \sigma_{AR} \\ \sigma_{AR} & \sigma_R^2 \end{bmatrix}, \tag{4.75}$$

given the $2n \times 2n$ input covariance matrix

$$C_X = \begin{bmatrix} C_P & \mathbf{0} \\ \mathbf{0} & C_Q \end{bmatrix} = \begin{bmatrix} diag(\sigma_{\rho_i}^2) & \mathbf{0} \\ \mathbf{0} & diag(\sigma_{\theta_i}^2) \end{bmatrix} \tag{4.76}$$

and the system relationships (4.73) and (4.74). Then by calculating the Jacobian:

$$F_{PQ} = \begin{bmatrix} \frac{\partial \alpha}{\partial P_1} & \frac{\partial \alpha}{\partial P_2} & \cdots & \frac{\partial \alpha}{\partial P_n} & \frac{\partial \alpha}{\partial Q_1} & \frac{\partial \alpha}{\partial Q_2} & \cdots & \frac{\partial \alpha}{\partial Q_n} \\ \frac{\partial r}{\partial P_1} & \frac{\partial r}{\partial P_2} & \cdots & \frac{\partial r}{\partial P_n} & \frac{\partial r}{\partial Q_1} & \frac{\partial r}{\partial Q_2} & \cdots & \frac{\partial r}{\partial Q_n} \end{bmatrix} \tag{4.77}$$

we can instantiate the uncertainty propagation equation (4.63) to yield C_{AR}:

$$C_{AR} = F_{PQ} C_X F_{PQ}^T \tag{4.78}$$

Thus we have calculated the probability C_{AR} of the extracted line (α, r) based on the probabilities of the measurement points. For more details about this method refer to Appendix A

4.3.1.2 Segmentation for Line Extraction

The previous section has described how to extract a line feature given a set of range measurements. Unfortunately, the feature extraction process is significantly more complex than this. A mobile robot does indeed acquire a set of range measurements, but in general the range measurements are not all part of one line. Rather, only some of the range measurements should play a role in line extraction and, further, there may be more than one line feature represented in the measurement set. This more realistic scenario is shown in Figure 4.38.

The process of dividing up a set of measurements into subsets that can be interpreted one-by-one is termed *segmentation* and is an important aspect of both range-based and vision-based perception. A diverse set of techniques exist for segmentation of sensor input in general. This general problem is beyond the scope of this text and, for details concerning seg-

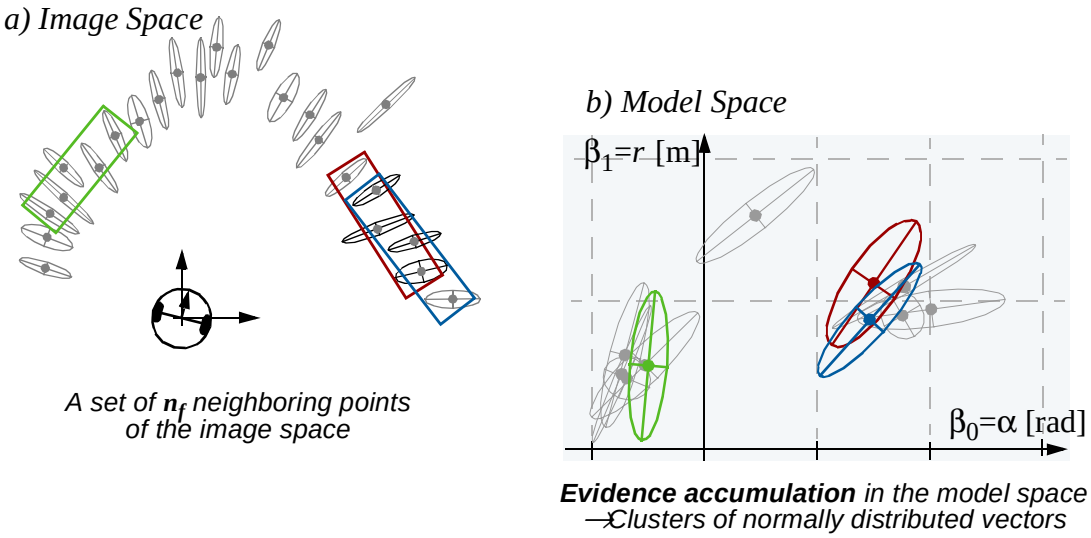


Fig 4.38 Clustering: Finding neighboring segments of a common line

mentation algorithms, refer to [Roland, reference here Koperski et al. and Shi et al. and I've put the references for in my text file for you].

For example, one segmentation technique is the merging, or *bottom-up* technique in which smaller features are identified and then merged together based on decision criteria to extract the goal features. Suppose that the problem of Fig. (4.38) is solved through merging. First, one may generate a large number of line segments based on adjacent groups of range measurements. The second step would be to identify line segments that have a high probability of belonging to the same extracted light feature. The simplest measure of the closeness of two line segments[4] $x_1 = [\alpha_1, r_1]$ and $x_2 = [\alpha_2, r_2]$ in the model space is given by Euclidean distance:

$$(x_1 - x_2)^T (x_1 - x_2) = (\alpha_1 - \alpha_2)^2 + (r_1 - r_2)^2 \tag{4.79}$$

The selection of all line segments x_j that contribute to the same line can now be done in a threshold-based manner according to:

$$(x_j - \bar{x})^T (x_j - \bar{x}) \leq d_m \tag{4.80}$$

where d_m is a threshold value and $\bar{x}$ is the representation of the reference line (from a model, average of a group of lines, etc.).

But the approach of Equation (4.80) does not take into account the fact that, for each measurement and therefore for each line segment we have a measure of uncertainty. One can improve upon this equation by selecting line segments that are weighted by their covariance matrix C_j:

4. Note: The lines are represented in polar coordinates.

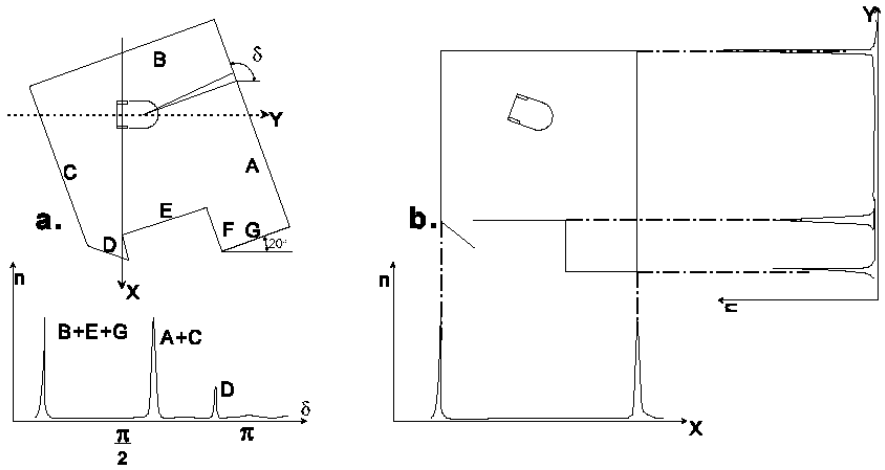


Fig 4.39 *Clustering: Finding neighboring segments of a common line (Weiß et al 1994 [110])*

$$(x_j - \bar{x})^T (C_j + \bar{C})^{-1} (x_j - \bar{x}) \leq d_m \tag{4.81}$$

The distance measure of Equation (4.81) discriminates the distance of uncertain points in model space considerably more effectively by taking uncertainty into account explicitly.

4.3.1.3 Range histogram features

A Histogram is a simple way to combine characteristics elements of an image. An angle histogram as presented in figure 4.39 plots the statistics of lines extracted by two adjacent range measurements. First, a 360 degree scan of the room is taken with the range scanner, and the resulting “hits” are recorded in a map. Then the algorithm measures the relative angle between any two adjacent hits (see Figure 4.39). After compensating for noise in the readings (caused by the inaccuracies in position between adjacent hits), the angle histogram shown in Figure 4.39a bottom can be built. The uniform direction of the main walls are clearly visible as peaks in the angle histogram. Detection of peaks yields only two main peaks: one for each pair of parallel walls. This algorithm is very robust with regard to openings in the walls, such as doors and windows, or even cabinets lining the walls.

4.3.1.4 Extracting other geometric features

Line features are of particular value for mobile robots operating in man-made environments, where for example building walls and hallway walls are usually straight. In general a mobile robot makes use of multiple features simultaneously, comprising a *feature set* that is most appropriate for its operating environment. For indoor mobile robots, the line feature is certainly a member of the optimal feature set.

In addition, other geometric kernels consistently appear throughout the indoor man-made environment: *Corners* features, defined as a point feature with an orientation; *step discontinuities*, defined as a step change perpendicular to the direction of hallway travel, are char-

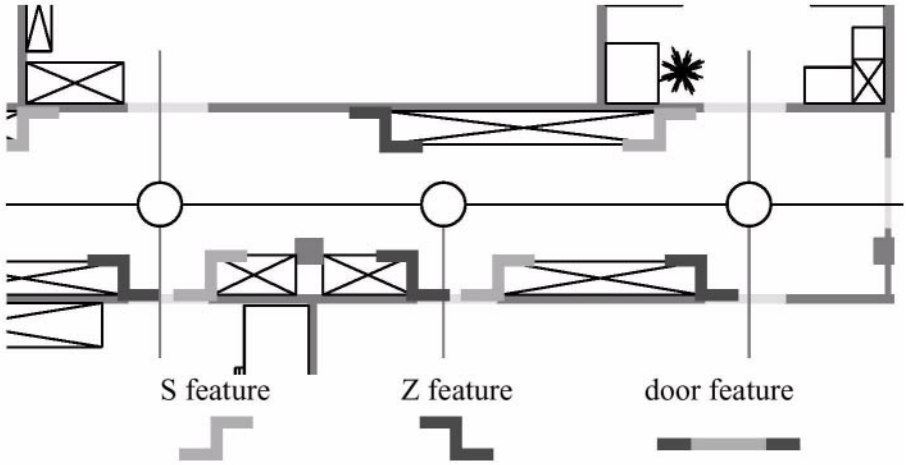


Fig 4.40 *Multiple geometric features in a single hallway, including doorways and discontinuities in the width of the hallway.*

acterized by their form (convex or concave) and step size; *doorways*, defined as openings of the appropriate dimensions in walls, are characterized by their width.

Thus, the standard segmentation problem is not so simple as deciding on a mapping from sensor readings to line segments, but rather it is a process in which features of different types are extracted based on the available sensor measurements. Figure 4.40 shows a model of an indoor hallway environment along with both indentation features (i.e. step discontinuities) and doorways.

Note that different feature types can provide quantitatively different information for mobile robot localization. The line feature, for example, provides two degrees of information, angle and distance. But the step feature provides two dimensional relative position information as well as angle.

The set of useful geometric features is essentially unbounded, and as sensor performance improves we can only expect greater success at the feature extraction level. For example, an interesting improvement upon the line feature described above relates to the advent of successful vision-based ranging systems. Because stereo vision provides a full three dimensional set of range measurements, one can extract plane features in addition to line features from the resulting data set. Plane features are valuable in man-made environments due to the flat walls, floors and ceilings of our indoor environments. Thus they are promising as another highly informative feature for mobile robots to use for mapping and localization.

4.3.2 Visual appearance-based feature extraction

Visual interpretation is, as we have mentioned before, an extremely challenging problem in the large. Significant research effort has been dedicated to inventing algorithms for understanding a scene based on 2D images over the past several decades, and the research efforts have slowly produced fruitful results. Covering the field of computer vision and image processing is of course beyond the scope of this work. To explore this discipline, refer to [22, 26, 107]

In Section 4.1.8 we have already seen vision-based ranging and color tracking sensors that

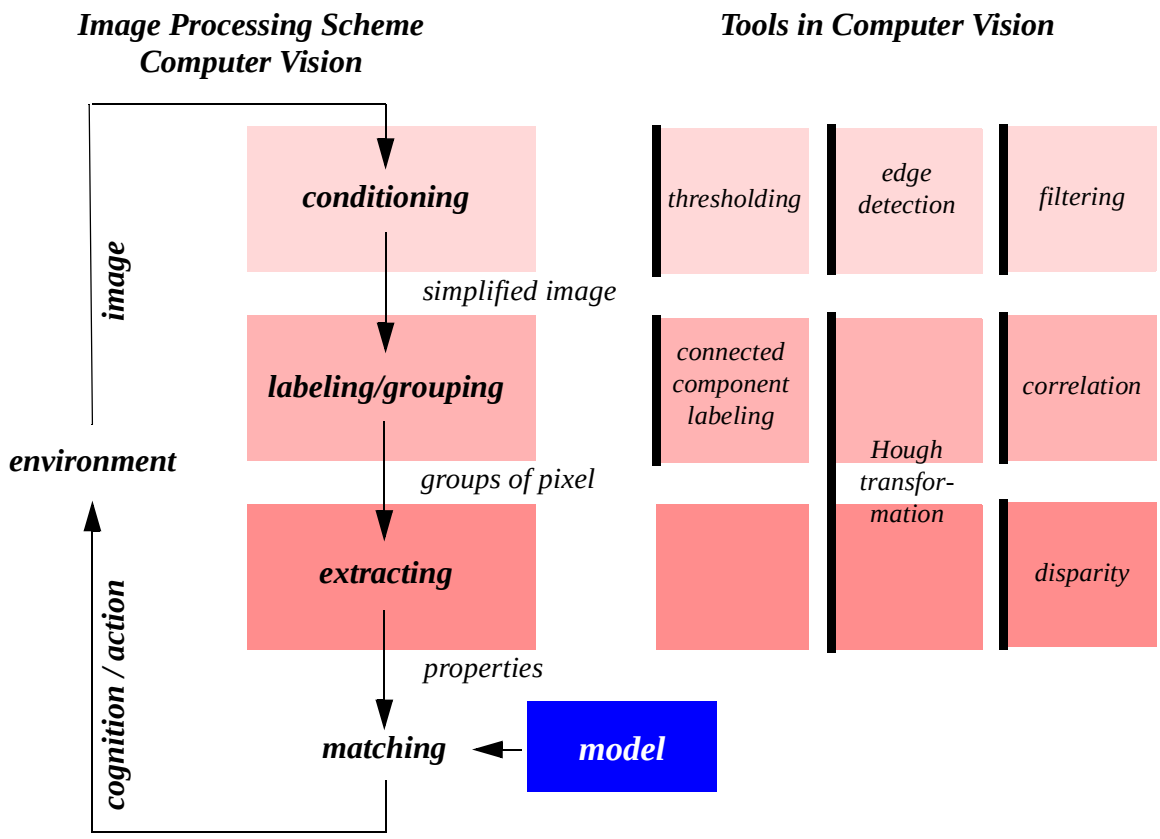


Fig 4.41 Scheme and tools in computer vision. See also [26]

are commercially available for mobile robots. These specific vision applications have witnessed commercial solutions primarily because the challenges are in both cases relatively well-focused and the resulting, problem-specific algorithms are straightforward. But images contain much more than implicit depth information and color blobs. We would like to solve the more general problem of extracting a large number of feature types from images.

This section presents some appearance-based feature extraction techniques that are relevant to mobile robotics along these lines. Two key requirements must be met for a vision-based feature extraction technique to have mobile robotic relevance. First, the method must operate in real time. Mobile robots move through their environment, and so the processing simply cannot be an off-line operation. Second, the method must be robust to the real-world conditions outside of a laboratory. This means that carefully controlled illumination assumptions and carefully painted objects are unacceptable requirements.

Throughout the following descriptions, keep in mind that vision-based interpretation is primarily about the challenge of *reducing information*. A sonar unit produces perhaps 50 bits of information per second. By contrast, a CCD camera can output 240 *million* bits per second! The sonar produces a tiny amount of information from which we hope to draw broader conclusions. But the CCD chip produces too much information, and this overabundance of information mixes together relevant and irrelevant information haphazardly. For example, we may intend to measure the color of a landmark. The CCD camera does not simply report its color, but also measures the general illumination of the environment, the direction of illumination, the defocusing caused by optics, the side effects imposed by nearby objects with

different colors, etc. Therefore the problem of visual feature extraction is largely one of removing the majority of irrelevant information in an image so that the remaining information unambiguously describes specific features in the environment.

We divide vision-based feature extraction methods into two classes based on their spatial extent. *Spatially localized features* are those features found in sub-regions of one or more images, corresponding to specific locations in the physical world. *Whole image features* are those features that are functions of the entire image or set of images, corresponding to a large visually connected area in the physical world.

Before continuing it is important to note that all vision-based sensors supply images with such a significant amount of noise that a first step usually consists of "cleaning" the image before launching any feature extraction algorithm. Therefore, we first describe the process of initial image filtering, or pre-processing.

Image Pre-Processing

Many image processing algorithms make use of the second derivative of the image intensity. Indeed, the Laplacian of Gaussian method we studied in Section 4.1.8.2 for stereo ranging is such an example. Because of the susceptibility of such high-order derivative algorithms to changes in illumination in the basic signal, it is important to smooth the signal so that changes in intensity are due to real changes in the luminosity of objects in the scene rather than random variations due to imaging noise. A standard approach is convolution with a Gaussian distribution function, as we described earlier in Section 4.1.8.2:

$$\hat{I} = G \otimes I \tag{4.82}$$

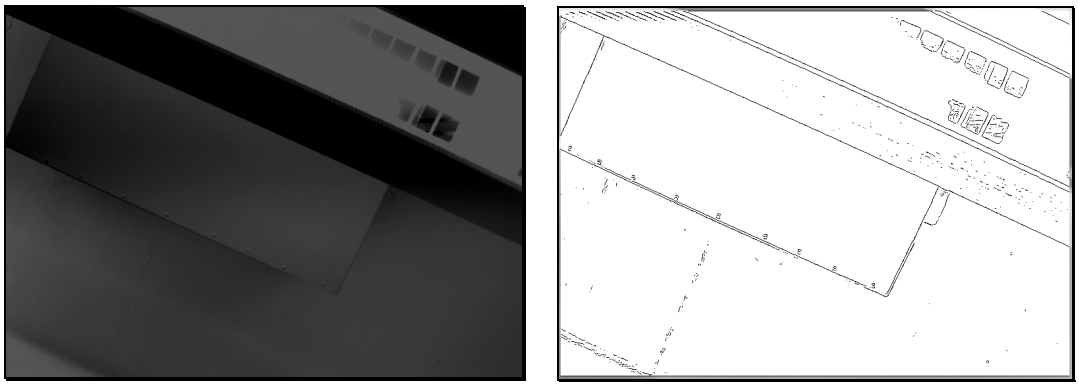
Of course, when approximated by a discrete kernel, such as a 3 x 3 table, the result is essentially local, weighted averaging:

$$G = \frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \tag{4.83}$$

Such a low pass filter effectively removes high-frequency noise, and this in turn causes the first derivative and especially the second derivative of intensity to be far more stable. Because of the important of gradients and derivatives to image processing, such Gaussian smoothing pre-processing is a popular first step of virtually all computer vision algorithms.

4.3.2.1 Spatially localized features

In the computer vision community many algorithms assume that the object of interest occupies only a sub-region of the image, and therefore the features being sought are localized spatially within images of the scene. Local image processing techniques find features that are local to a subset of pixels, and such local features map to specific locations in the phys-



a b

Fig 4.42 (a) Photo of a ceiling lamp. (b) Edges computed from (a)

ical world. This makes them particularly applicable to geometric models of the robot's environment.

The single most popular local feature extractor used by the mobile robotics community is the edge detector, and so we begin with a discussion of this classical topic in computer vision. However, mobile robots face the specific mobility challenges of obstacle avoidance and localization. In view of obstacle avoidance, we present vision-based extraction of the floor plane, enabling a robot to detect all areas that can be safely traversed. Finally, in view of the need for localization we discuss the role of vision-based feature extraction in the detection of robot navigation landmarks.

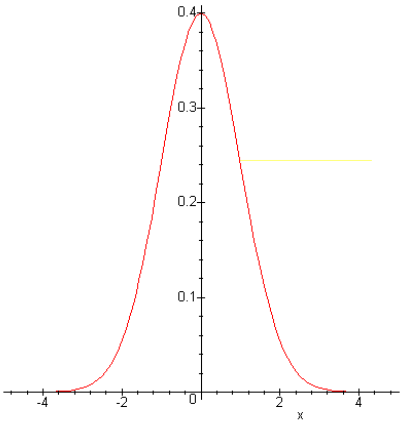
Edge Detection

Figure 4.42 shows an image of a scene containing a part of a ceiling lamp as well as the edges extracted from this image. Edges define regions in the image plane where a *significant* change in the image brightness takes place. As shown in this example, edge detection significantly reduces the amount of information in an image, and is therefore a useful potential feature during image interpretation. The hypothesis is that edge contours in an image correspond to important scene contours. As the Figure 4.42(b) shows, this is not entirely true. There is a difference between the output of an edge detector and an ideal line drawing. Typically, there are missing contours, as well as noise contours that do not correspond to anything of significance in the scene.

The basic challenge of edge detection is visualized in Figure 4.23. Figure 4.23(top left) shows the 1-D section of an ideal edge. But the signal produced by a camera will look more like figure 4.23(top right). The location of the edge is still at the same x value, but a significant level of high-frequency noise affects the signal quality.

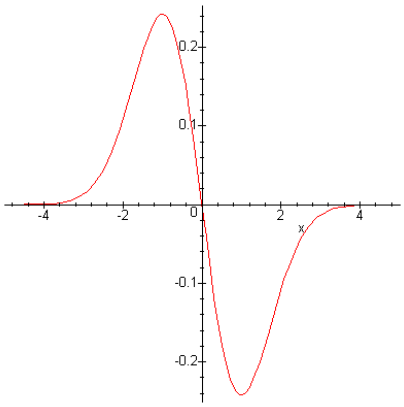
A naive edge detector would simply differentiate, since an edge by definition is located where there are large transitions in intensity. As shown in figure 4.23(bottom right), differentiation of the noisy camera signal results in subsidiary peaks that can make edge detection very challenging. A far more stable derivative signal can be generated simply by pre-processing the camera signal using the Gaussian smoothing function described above. Below, we present several popular edge detection algorithms, all of which operate on this same basic principle, that the derivative(s) of intensity, following some form of smoothing, comprise

a



$$G_\sigma(x) = \frac{1}{\sqrt{2\pi\sigma}} e^{-\frac{x^2}{2\sigma^2}}$$

b



$$G_\sigma'(x) = \frac{-x}{\sqrt{2\pi\sigma^3}} e^{-\frac{x^2}{2\sigma^2}}$$

Fig 4.43 (a) A Gaussian function. (b) The first derivative of a Gaussian function.

the basic signal from which to extract edge features.

Optimal Edge Detection: Canny

The current reference edge detector throughout the vision community was invented by John Canny in 1983 [16]. This edge detector was born out of a formal approach in which Canny treated edge detection as a signal processing problem in which there are three explicit goals:

1. Maximize the signal-to-noise ratio
2. Achieve the highest precision possible on the location of edges
3. Minimize the number of edge responses associated with each edge

The Canny edge extractor smooths the image I via Gaussian convolution and then looks for maxima in the (rectified) derivative. In practice the smoothing and differentiation are combined into one operation because:

$$(G \otimes I)' = G' \otimes I \tag{4.84}$$

Thus, smoothing the image by convolving with a Gaussian G_σ and then differentiating is equivalent to convolving the image with G'_σ the first derivative of a Gaussian (Figure 4.43(b)).

We wish to detect edges in any direction. Since G' is directional, this requires application of two perpendicular filters, just as we did for the Laplacian in Equation (4.35). We define the two filters as $f_V(x,y) = G'_\sigma(x)G_\sigma(y)$ and $f_H(x,y) = G'_\sigma(y)G_\sigma(x)$. The result is a basic algorithm for detecting edges at arbitrary orientations:

The algorithm for detecting edge pixels at an arbitrary orientation is:

1. Convolve the image $I(x, y)$ with $f_V(x, y)$ and $f_H(x, y)$ to obtain the gradient compo-

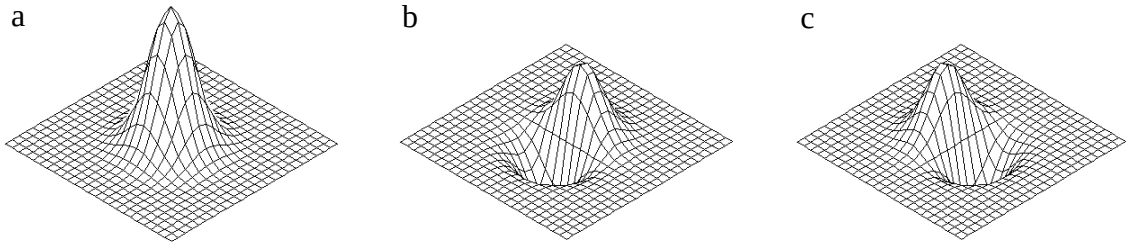


Fig 4.44 (a) Two dimensional Gaussian function; (b) Vertical filter; (c) Horizontal filter.

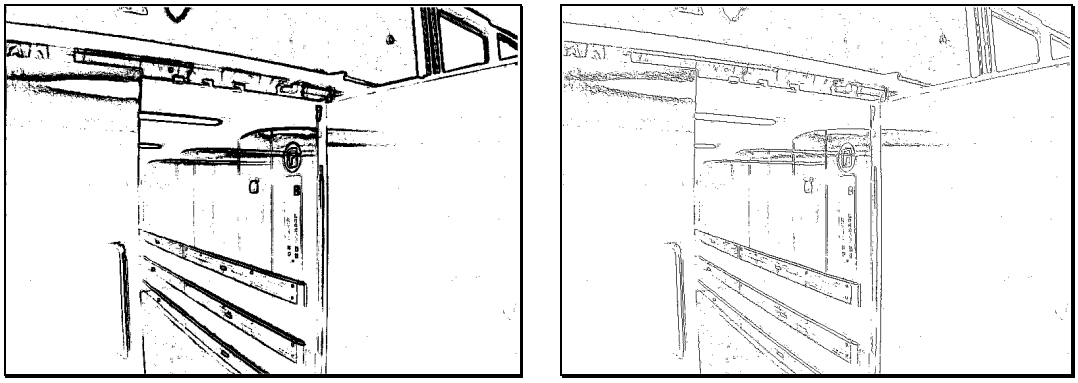


Fig 4.45 (a) Edge image of the Figure 1.2(b); (b) Non-maxima suppression of (a)

nents $R_V(x, y)$ and $R_H(x, y)$, respectively.

2. Define the square of the gradient magnitude $R(x, y) = R^2_V(x, y) + R^2_H(x, y)$.
3. Mark those peaks in $R(x, y)$ that are above some predefined threshold T.

Once edge pixels are extracted, the next step is to construct complete edges. A popular next step in this process is *non-maxima suppression*. Using edge direction information, the process involves revisiting the gradient value and determining whether or not it is at a local maximum. If not, then the value is set to zero. This causes only the maxima to be preserved, and thus reduce the thickness of all edges to a single pixel (Figure 4.45).

Finally, we are ready to go from edge pixels to complete edges. First, find adjacent (or connected) sets of edges and group them into ordered lists. Second, use thresholding to eliminate the weakest edges.

Gradient Edge Detectors

On a mobile robot, computation time must be minimized to retain the real-time behavior of the robot. Therefore simpler, discrete kernel operators are commonly used to approximate the behavior of the Canny edge detector. One such early operators was developed by ***Roberts*** in 1965 [22]. He used two 2 x 2 masks to calculate the gradient across the edge in two diagonal directions. Let r_1 be the value calculated from the first mask and r_2 from the second mask. Roberts obtained the gradient magnitude $|G|$ with the equation

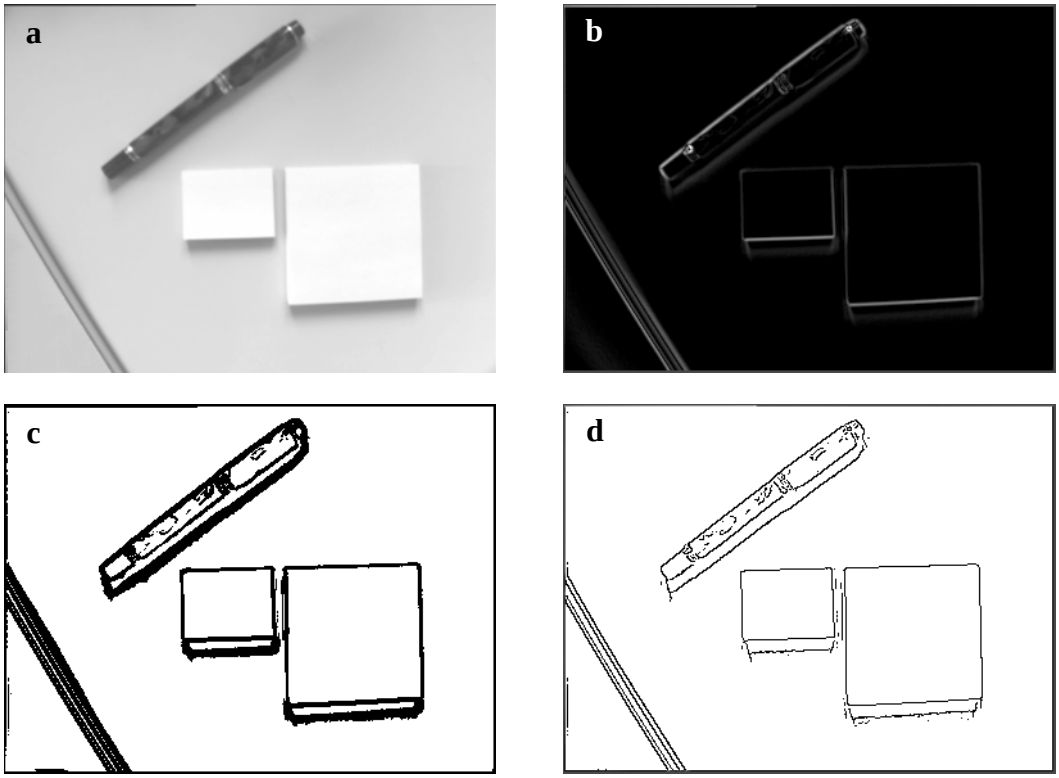


Fig 4.46 Example of vision based feature extraction with the different processing steps:
a: raw image data
b: filtered image using a Sobel filter
c: thresholding, selection of edge pixels
d: non-maxima suppression

$$|G| \cong \sqrt{r_1^2 + r_2^2} \;;\; r_1 = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \;;\; r_2 = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \tag{4.85}$$

Prewitt (1970) [22] used two 3 x 3 masks oriented in the row and column directions. Let p_1 be the value calculated from the first mask and p_2 the value calculated from the second mask. Prewitt obtained the gradient magnitude $|G|$ and the gradient direction θ taken in a clockwise angle with respect to the column axis shown in the following equation.

$$|G| \cong \sqrt{p_1^2 + p_2^2} \;;$$
$$\theta \cong \operatorname{atan}\left(\frac{p_1}{p_2}\right) \;;\quad p_1 = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \;;\; p_2 = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \tag{4.86}$$

In the same year ***Sobel*** [22] used, like Prewitt, two 3 x 3 masks oriented in the row and column direction. Let s_1 be the value calculated from the first mask and s_2 the value calculated from the second mask, he obtained the same results as Prewitt for the gradient magnitude $|G|$ and the gradient direction θ taken in a clockwise angle with respect to the column axis. Figure 4.46 shows application of the Sobel filter to a visual scene.

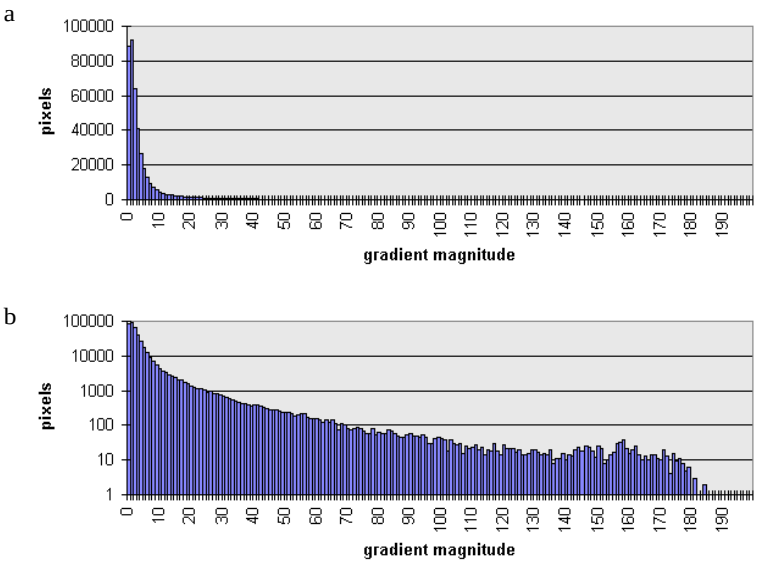


Fig 4.47 (a) Number of pixels with a specific gradient magnitude in the image of Figure 1.2(b). (b) Same as (a), but with logarithmic scale

$$|G| \cong \sqrt{s_1^2 + s_2^2} \quad ;$$

$$\theta \cong \operatorname{atan}\left(\frac{s_1}{s_2}\right) ; \qquad s_1 = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \quad ; \quad s_2 = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \tag{4.87}$$

Dynamic Thresholding

Many image processing algorithms have generally been tested in laboratory conditions or using static image databases. Mobile robots, however, operate in dynamic real-world settings where there is no guarantee regarding optimal or even stable illumination. A vision system for mobile robots has to adapt to the changing illumination. Therefore a constant threshold level for edge detection is not suitable. The same scene with different illumination will results in edge images with considerable differences. To dynamically adapt the edge detector to the ambient light, a more adaptive threshold is required, and one approach involves calculating that threshold based on a statistical analysis of the image about to be processed.

To do this, a histogram of the gradient magnitudes of the processed image is calculated (figure 4.47). With this simple histogram it is easy to consider only the *n* pixels with the highest gradient magnitude for further calculation steps. The pixels are counted backward starting at the highest magnitude. The gradient magnitude of the point where *n* is reached will be used as the temporary threshold value.

The motivation for this technique is that the *n* pixels with the highest gradient are expected to be the most relevant ones for the processed image. Furthermore, for each image, the same number of relevant edge pixels is considered, independent of illumination. It is important to pay attention to the fact that the number of pixels in the edge image delivered by the edge detector is not *n*. Because most detector uses non-maxima suppression, the number of edge pixels will be further reduced.

Straight edge extraction: Hough transforms

In mobile robotics the straight edge is often extracted as a specific feature. Straight vertical edges, for example, can be used as clues to the location of doorways and hallway intersections. The Hough Transform is a simple tool for extracting edges of a particular shape[15, 26]. Here we explain its application to the problem of extracting straight edges.

Suppose a pixel (x_p, y_p) in the Image *I* is part of an edge. Any straight line edge including point (x_p, y_p) must satisfy the equation: $y_p = m_1 x_p + b_1$. This equation can only be satisfied with a constrained set of possible values for m_1 and b_1. In other words, this equation is satisfied only by lines through *I* that pass through (x_p, y_p).

Now consider a second pixel, (x_q, y_q) in *I*. Any line passing through this second pixel must satisfy the equation: $y_q = m_2 x_q + b_2$. What if $m_1 = m_2$ and $b_1 = b_2$? Then the line defined by both equations is one and the same: it is the line that passes through both (x_p, y_p) and (x_q, y_q).

More generally, for all pixels that are part of a single straight line through *I*, they must all lie on a line defined by the *same* values for *m* and *b*. The general definition of this line is, of course, $y=mx + b$. The Hough Transform uses this basic property, creating a mechanism so that each edge pixel can "vote" for various values of the (*m*,*b*) parameters. The lines with the most votes at the end are straight edge features:

1. Create a two-dimensional array *A* with axes that tessellate the values of *m* and *b*
2. Initialize the array to zero: $A[m, b] = 0$ for all values of *m*,*b*
3. For each edge pixel (x_p, y_p) in *I*, loop over all values of *m* and *b*:

   if $y_p = mx_p + b$ then $A[m, b] \mathrel{+}= 1$
4. Search the cells in *A* to identify those with the largest value. Each such cell's indices (*m*,*b*) corresponds to an extracted straight line edge in *I*.

Floor Plane Extraction

Obstacle avoidance is one of the basic tasks required of most mobile robots. Range-based sensors provide effective means for identifying most types of obstacles facing a mobile robot. In fact, because they directly measure range to objects in the world, range-based sensors such as ultrasonic and laser rangefinders are inherently well-suited for the task of obstacle detection. However, each ranging sensor has limitations. Ultrasonics have poor angular resolution and suffer from coherent reflection at shallow angles. Most laser rangefinders are 2D, only detect obstacles penetrating a specific sensed plane. Stereo vision and depth from focus require the obstacles and floor plane to have texture in order to enable correspondence and blurring respectively.

In addition to each individual shortcoming, range-based obstacle detection systems will have difficulty detecting small or flat objects that are on the ground. For example, a vacuum

cleaner may need to avoid large, flat objects, such as paper or money left on the floor. In addition, different types of floor surfaces cannot easily be discriminated by ranging. For example, a sidewalk-following robot will have difficulty discriminating grass from pavement using range sensing alone.

Floor plane extraction is a vision-based approach for identifying the traversable portions of the ground. Because it makes use of edges and color in a variety of implementations, such obstacle detection systems can easily detect obstacles in cases that are difficult for traditional ranging devices.

As is the case with all vision-based algorithms, floor plane extraction succeeds only in environments that satisfy several important assumptions:

1. Obstacles differ in appearance from the ground.
2. The ground is flat and its angle to the camera is known.
3. There are no overhanging obstacles.

The first assumption is a requirement in order to discriminate the ground from obstacles using its appearance. The second and third assumptions allow floor plane extraction algorithms to estimate the robot's distance to obstacles detected.

Floor plane extraction in artificial environments

In a controlled environment, the floor, walls and obstacles can be designed so that the walls and obstacle appear significantly differently than the floor in a camera image. Shakey, the first autonomous robot developed from 1966 through 1972 at SRI, used vision-based floor plane extraction in a manufactured environment for obstacle detection [104]. Shakey's artificial environment used textureless, homogeneously white floor tiles. Furthermore, the base of each wall was painted with a high-contrast strip of black paint and the edges of all simple polygonal obstacles were also painted black.

In Shakey's environment, edges corresponded to non-floor objects, and so the floor plane extraction algorithm simply consisted of the application of an edge detector to the monochrome camera image. The lowest edges detected in an image corresponded to the closest obstacles, and the direction of straight line edges extracted from the image provided clues regarding not only the position but also the orientation of walls and polygonal obstacles.

Although this very simple appearance-based obstacle detection system was successful, it should be noted that special care had to be taken at the time in order to create indirect lighting in the laboratory such that shadows were not cast, as the system would falsely interpret the edges of shadows as obstacles.

Adaptive floor plane extraction

Floor plane extraction has succeeded, not only in artificial environments, but in real-world mobile robot demonstrations in which a robot avoids both static obstacles such as walls and dynamic obstacles such as passers-by based on segmentation of the floor plane at a rate of several Hz. Such floor plane extraction algorithms tend to use edge detection and color detection jointly while making certain assumptions regarding the floor, for example the floor's maximum texture or approximate color range [98]

Fig 4.48 *Examples of adaptive floor plane extraction. The trapezoidal polygon identifies the floor sampling region.*

Each system based on fixed assumptions regarding the floor's appearance is limited to only those environments satisfying its constraints. A more recent approach is that of adaptive floor plane extraction, whereby the parameters defining the expected appearance of the floor are allowed to vary over time. In the simplest instance, one can assume that the pixels at the bottom of the image (i.e. closest to the robot) are part of the floor and contain no obstacles. Then, statistics computed on these "floor sample" pixels can be used to classify the remaining image pixels.

The key challenge in adaptive systems is the choice of what statistics to compute using the "floor sample" pixels. The most popular solution is to construct one or more *histograms* based on the floor sample pixel values. In *Edge Detection* above, we found histograms to be useful in determining the best cut point in edge detection thresholding algorithms. Histograms are also useful as discrete representations of distributions. Unlike the Gaussian representation, a histogram can capture multi-modal distributions. Histograms can also be updated very quickly and use very little processor memory. An intensity histogram of the "floor sample" subregion I_f of image I is constructed as follows:

1. As pre-processing, smooth I_f using a Gaussian smoothing operator
2. Initialize a histogram array H with n intensity values: $H[i] = 0$ for $i = 1, \ldots, n$
3. For every pixel (x,y) in I_f increment the histogram: $H[I_f(x, y)] \mathrel{+}= 1$

The histogram array H serves as a characterization of the appearance of the floor plane. Often, several 1D histograms are constructed, corresponding to intensity, hue and saturation for example. Classification of each pixel in I as floor plane or obstacle is performed by looking at the appropriate histogram counts for the qualities of the target pixel. For example, if the target pixel has a hue that never occurred in the "floor sample," then the corresponding hue histogram will have a count of zero. When a pixel references a histogram value below a pre-defined threshold, that pixel is classified as an obstacle.

Figure 4.48 shows an appearance-based floor plane extraction algorithm operating on both indoor and outdoor images [99]. Note that, unlike the static floor extraction algorithm, the

adaptive algorithm is able to successfully classify a human shadow due to the adaptive histogram representation. An interesting extension of the work has been to not use the static floor sample assumption, but rather to record visual history and to use, as the floor sample, only the portion of prior visual images that has successfully rolled under the robot during mobile robot motion.

Appearance-based extraction of the floor plane have been demonstrated on both indoor and outdoor robots for real-time obstacle avoidance with a bandwidth of up to 10 Hz. Applications include robotics lawn mowing, social indoor robots and automated electric wheelchairs.

4.3.2.2 Whole-Image Features

A single visual image provides so much information regarding a robot's immediate surroundings that an alternative to searching the image for spatially localized features is to make use of the information captured by the entire image to extract a whole-image feature. Whole-image features are not designed to identify specific spatial structures such as obstacles or the position of specific landmarks. Rather, they serve as compact representations of the entire local region. From the perspective of robot localization, the goal is to extract one or more features from the image that are correlated well with the robot's position. In other words, small changes in robot position should cause only small changes to whole-image features, while large changes in robot position should cause correspondingly large changes to whole-image features.

We present two techniques for whole-image feature extraction below. The first technique is another popular application of the image histogramming approach. The resulting image histogram comprises a set of whole-image features derived directly from the pixel information of an image. The second technique, tiered extraction, covers approaches in which a whole-image feature is built by first extracting spatially localized features, then composing these features together to form a single meta-feature.

Direct Extraction: Image Histograms

Recall that we wish to design whole-image features that are insensitive to small amount of robot motion while registering significant changes for large-scale robot motion. A logical first step in designing a vision-based sensor for this purpose is to maximize the field of view of the camera. As the field of view increases, small-scale structure in the robot's environment occupies a smaller proportion of the image, thereby mitigating the impact of individual scene objects on image characteristics. The catadioptric camera system, now very popular in mobile robotics, offers an extremely wide field of view [100]. This imaging system consists of a high-quality CCD camera mounted, together with customized optics, towards a parabolic mirror. The image provides a 360° view of the robot's environment, as shown in Figure 4.49.

The catadioptric image is a 360° image warped onto a two-dimensional image surface. Because of this, it offers another critical advantage in terms of sensitivity to small-scale robot

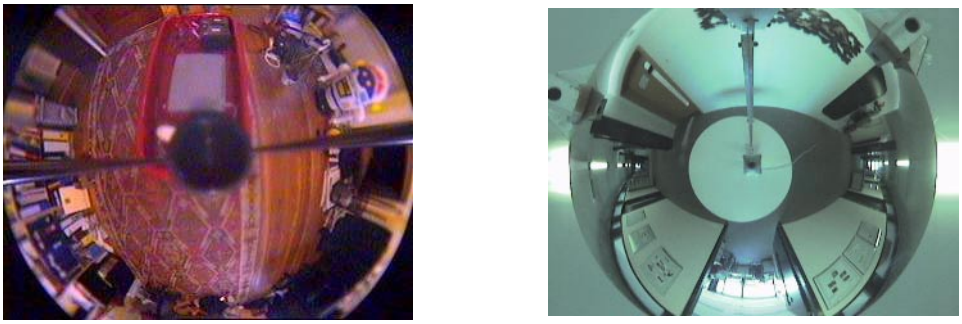


Fig 4.49 Two typical image acquired by the OmniCam catadioptric camera system.

motion. If the camera is mounted vertically on the robot so that the image represents the environment surrounding the robot (i.e. its horizon), then rotation of the camera and robot simply results in image rotation. In short, the catadioptric camera can be rotationally invariant to field of view.

Of course, mobile robot rotation will still change the image; that is, pixel positions will change, although the new image will simply be a rotation of the original image. But we intend to extract image features via histogramming. Because histogramming is a function of the set of pixel values and not the position of each pixel, the process is pixel position-invariant. When combined with the Catadioptric camera's field of view invariance, we can create a system that is invariant to robot rotation and insensitive to small-scale robot translation.

A color camera's output image generally contains useful information along multiple *bands*: r, g and b values as well as hue, saturation and luminance values. The simplest histogram-based extraction strategy is to build separate 1D histograms characterizing each band. Given a color camera image, G, the first step is to create mappings from G to each of the n available bands. We use G_i to refer to an array storing the values in band i for all pixels in G.

Each band-specific histogram H_i is calculated as before:

1. As pre-processing, smooth G_i using a Gaussian smoothing operator
2. Initialize H_i with n levels: $H[j] = 0$ for $= 1, \ldots, n$
3. For every pixel (x,y) in G_i increment the histogram: $_i[G_i[x, y]]+= 1$

Given the image shown in Figure 4.49, the image histogram technique extracts six histograms (for each of r, g, b, hue, saturation and luminance) as shown in Figure 4.50In order to make use of such histograms as whole-image features, we need ways to compare to histograms to quantify the likelihood that the histograms map to nearby robot positions. The problem of defining useful histogram distance metrics is itself an important subfield within the image retrieval field. For an overview refer to [101]. One of the most successful distance metrics encountered in mobile robot localization is the *Jeffrey Divergence*. Given two histograms H and K, with h_i and k_i denoting the histogram entries, the Jeffrey divergence $d(H,K)$ is defined as:

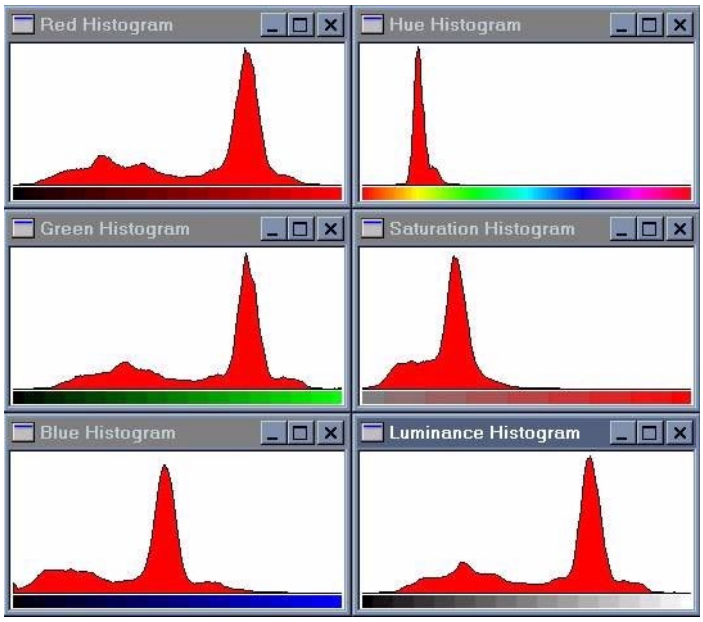


Fig 4.50 *Six 1D histograms of the image above. A 5 x 5 smoothing filter was convolved with each band before histogramming.*

$$d(H,\ K)\ =\ \sum_i \left(h_i \log \frac{2h_i}{h_i + k_i} + k_i \log \frac{2k_i}{h_i + k_i} \right) \tag{4.88}$$

Using measures such as the Jeffrey divergence, mobile robots have used whole-image histogram features to identify their position in real time against a database of previously recorded images of locations in their environment. Using this whole-image extraction approach, a robot can readily recover the particular hallway or particular room in which it is located [102].

Tiered Extraction: Image Fingerprint Extraction

An alternative to extracting a whole-image feature directly from pixel values is to use a tiered approach: first identify spatially localized features in the image, then translate from this set of local features to a single meta-feature for the whole image. We describe one particular implementation of this approach, in which the resulting whole-image feature is called the image *fingerprint* [103]. As with other whole-image extraction techniques, because low sensitivity to small robot motions is desired, the system makes use of a 360° panoramic image, here constructed as a mosaic of images captured with a standard CMOS-chip camera.

The first extraction tier searches the panoramic image for spatially localized features: vertical edges and 16 discrete hues of color. The vertical edge detector is a straightforward gradient approach implementing a horizontal difference operator. Vertical edges are "voted upon" by each edge pixel just as in a vertical edge Hough transform. As described in Section (4.3.2.1), an adaptive threshold is used to reduce the number of edges. Suppose the Hough table's tallies for each candidate vertical line have a mean μ and a standard deviation σ. The

Fig 4.51 *Two panoramic images and their associated fingerprint sequences [103].*

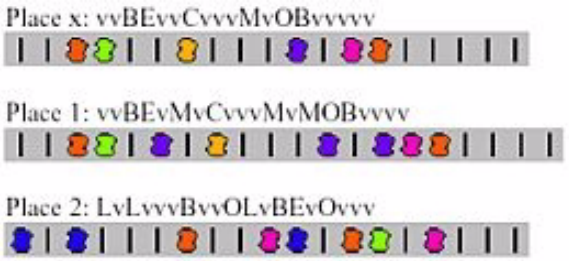


Fig 4.52 *Three actual string sequences. The top two are strings extracted by the robot at the same position [103].*

chosen threshold is simply $\mu + \sigma$.

Vertical color bands are identified in largely the same way, identifying statistics over the occurrence of each color, then filtering out all candidate color patches except those with tallies greater than $\mu + \sigma$. Figure 4.51 shows two sample panoramic images and their associated *fingerprints*. Note that each fingerprint is converted to an ASCII string representation.

Just as with histogram distance metrics in the case of image histogramming, we need a quantifiable measure of the distance between two fingerprints strings. String matching algorithms are yet another large field of study, with particularly interesting applications today in the areas of genetics [96]. Note that we may have strings that differ not just in a single element value, but even in their overall length. For example, Figure 4.52 depicts three actual sequences generated using the above algorithm. The top string should match *Place 1*, but note that there are deletions and insertions between the two strings.

The technique used in the *fingerprinting* approach for string differencing is known as a *Minimum Energy Algorithm.* Taken from the stereo vision community, this optimization-based algorithm will find the minimum energy required to "transform" one sequence into another sequence. The result is a distance metric that is relatively insensitive to the addition or subtraction of individual local features while still able to robustly identify the correct matching string in a variety of circumstances.

It should be clear from the previous two subsections that whole-image feature extraction is straightforward with vision-based perception and can be applicable for mobile robot local-

ization. But it is spatially localized features that continue to play a dominant role because of their immediate application to the more urgent need for real-time obstacle avoidance.