République Algérienne Démocratique et Populaire

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université Hadj-Lakhdar. Batna

Faculté de Technologie

Département d'Electronique

MEMOIRE

Présenté en vue de l'obtention du diplôme de

Magister *en Electronique*

Option: Robotique

Par

Soumia BENSAADI

Ingénieur d'état en Electronique,
Université de Batna

Système de Perception Visuelle pour un Robot Mobile

Date: 30 *avril*, 2015

MELAAB Djamel	*Maître de Conférences –A*	Université de Batna	président
LOUCHENE Ahmed	*professeur*	Université de Batna	rapporteur
TAIBI Mahmoud	*professeur*	Université de Annaba	examinateur
BENOUDJIT Nabil	*professeur*	Université de Batna	examinateur
BENZID Redha	*professeur*	Université de Batna	examinateur

Avril 2015

People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research

Hadj-Lakhdar University, Batna
Facuty of Technology
Department of Electronics

A Dissertation

Presented in partial fulfillment of the requirements for the degree of

'Magister' *in Electronics*
Option: Robotics

By

Soumia BENSAADI
Engineer in Electronics,
University of Batna

A visual Perception System for a Mobile Robot

Date: *April* 30, 2014

ExaminationCommittee

MELAAB Djamel	*'Maître de Conférences –A'*	University of Batna	president
LOUCHENE Ahmed	*Professor*	University of Batna	supervisor
TAIBI Mahmoud	*Professor*	Universityof Annaba	reviewer
BENOUDJIT Nabil	*Professor*	University of Batna	reviewer
BENZID Redha	*Professor*	University of Batna	reviewer

April 2015

Content

Introduction

The development of mobile robots is one of the most challenging research fields within robotics. The hardware of a robot can be divided into its mechanical components and implementation, actuators, sensors, and computing hardware. Building on top of this hardware, it is the software that makes the difference between a simple machine and an intelligent robot. Among the components and abilities to be implemented are the most important ones related to motor control, speech processing and synthesis, visual perception, and some kind of high-level planning or control module. All of these are integrated and connected within an overall architecture, which is often referred to as the cognitive architecture - whose design and implementation are currently among the most discussed topics within the robotics community.

Visual information makes up about seventy five percent of all the sensorial information received by a person during a lifetime. This information is processed not only efficiently but also transparently. Our sight is the most perfect and most delightful of all our senses. It fills the mind with the largest variety of ideas, converses with its objects at the greatest distance, and continues the longest in action without being tired or satiated with its proper enjoyments.

To understand why the performance of generic computer vision algorithms is still far away from that of human visual perception, we should consider the hierarchy of computer vision tasks. They can be roughly classified into three large categories:

– *Low level*, dealing with extraction from a single image of salient simple features, such as edges, corners, homogeneous regions, curve fragments;

– *Intermediate level*, dealing with extraction of semantically relevant characteristics from one or more images, such as grouped features, depth, motion information;

– *High level*, dealing with the interpretation of the extracted information.

A similar hierarchy is difficult to distinguish in human visual perception, which appears as a single integrated unit. In the visual tasks performed by a human observer an

extensive top-down information flow carrying representations derived at higher levels seems to control the processing at lower levels.

A large amount of psychophysical evidence supports this "closed loop" model of human visual perception. Preattentive vision phenomena, in which salient information pops-out from the image, or perceptual constancies, in which changes in the appearance of a familiar object are attributed to external causes, are only some of the examples. Similar behavior is yet to be achieved in generic computer vision techniques. For example, preattentive vision type processing seems to imply that a region of interest is delineated *before* extracting its salient features.

Motivation and Objective

It is a challenge to mimic human visual perception involved sometimes in realizing complex decision making tasks. Computer vision is a vast research area with many applications in artificial intelligence. Furthermore many interesting higher-level skills, tasks, and scenarios cannot be realized without a strong vision system as a basis.

Therefore, in the broadest sense, robustness of a computer vision algorithm is judged against the performance of a human observer performing an equivalent task. In this context, robustness is the ability to extract the visual information of relevance for a specific task, even when this information is carried only by a small subset of the data, and/or is significantly different from an already stored representation.

One of the most traditional disciplines and the objective of this thesis is the *object recognition* with its two subproblems, *object identification* and *object location*.

Problem Statement: Model (Edge)-Based Object Recognition

Given an input image frame taken from the robot environment, it is asked to perceive its location in terms of a reference fixed image based mark. In this context, model-based object recognition addresses two problems:

Identification: which 2D shapes in the image will be identified?

Location: given that the geometric structure of the object in the image is identified what is the location in space (rotation and translation) of the 2-D object imaged?

1

Perception and Navigation systems for mobile robots

1.1. Mobile Robots

After proving to be an efficient tool for improving quality, productivity, and competitiveness of manufacturing organizations, robots now expand to service organizations, offices, and even homes. Global competition and the tendency to reduce production cost and increase efficiency creates new applications for robots that stationary robots can't perform. These new applications require the robots to move and perform certain activities at the same time.

The proposed architecture for machine learning is also based on the perceptual creative controller for an intelligent robot that uses a multi- modal adaptive critic for performing learning in an unsupervised situation but can also be trained for tasks in another mode and then is permitted to operate autonomously. The robust nature is derived from the automatic changing of modes based on internal measurements of error at appropriate locations in the controller. [1]

1.1.1. Types of Mobile Robots

Many different types of mobile robots had been developed depending on the kind of application, velocity, and the type of environment whether it is water, space, terrain with fixed or moving obstacles. Four major categories had been identified:

i. Terrestrial or Ground-contact Robots

There are three main types of ground-contact robots: wheeled robots, tracked vehicles, and limbed vehicles.

***Wheeled robots*:** exploit friction or ground contact to enable the robot to move. Different kinds of wheeled robots exist: the differential drive robot, synchronous drive robot, steered wheels robots and Ackerman steering (car drive) robots, the tricycle, bogey, and bicycle drive robots, and robots with complex or compound or omnidirectional wheels.

Tracked vehicles: are robust to any terrain environment, their construction is similar to the differential drive robot but the two differential wheels are extended into treads which provide a large contact area and enable the robot to navigate through a wide range of terrain.

Limbed vehicles: are suitable in rough terrains such as those found in forests, near natural or man-made disasters, or in planetary exploration, where ground contact support is not available for the entire path of motion. Limbed vehicles are characterized by the design and the number of legs, the minimum number of legs needed for a robot to move is one, to be supported a robot need at least three legs, and four legs are needed for a statically stable robot, six, eight, and twelve legs robots exists.

Figure 1.1. Experimental Unmanned Vehicle in action at Ft. Indiantown Gap. Photo courtesy of the Army Research Labs (Greenhouse & Norris, 2002)

ii. Aquatic Robots

Aquatic vehicles support propulsion by utilizing the surrounding water. There are two common structures:

Torpedo-like structures; where a single propeller provides forward, and reverse thrust while the navigation direction is controlled by the control surfaces, the buoyancy of the vessel controls the depth. The disadvantage of this type is poor manoeuvrability.

Twin-burger and URV vehicles robots which use a collection of thrusters that are distributed over the vessel, more manoeuvrable are attained by controlling sets of the thrusters to change the vehicle orientation and position independently; however, the URV comes with the expense of operational speed. An example of an aquatic robot is shown in *Figure2a.*

iii. Flying Robots

Fixed-wing autonomous vehicles: This utilizes control systems very similar to the ones found in commercial autopilots. Ground station can provide remote commands if needed, and with the help of the Global Positioning System (GPS) the location of the vehicle can be determined.

Automated helicopters: this use onboard computation and sensing and ground control, their control is very difficult compared to the fixed-wing autonomous vehicles.

Buoyant (aerobots, aerovehicles, or blimps) vehicles: These vehicles can float and are characterized by having high energy efficiency ratio, long-range travel and duty cycle, vertical mobility, and they usually has no disastrous results in case of failure.

Unpowered autonomous flying vehicles: These vehicles reach their desired destination by utilizing gravity, GPS, and other sensors. An example of a flying robot is shown in *Figure2b.*

a. Aquatic Vehicle

b. Flying Vehicle

Figure 1.2. Aquatic and Flying vehicle (SSC San Diego, 2002)

iv. Space Robots

These are needed for applications related to space stations like construction, repair, and maintenance. Free-flying systems have been proposed where the spacecraft is equipped with thrusters with one or more manipulators, the thrusters are utilized to modify the robot trajectory. [1]

1.1.2. Integrated Perception and Navigation systems

Using highly developed sensors, mobile robots can accomplish very sophisticated tasks even in human environment. This chapter concentrates on the sensors used in mobile robot systems to avoid obstacles allocating position and gathering information from the environment and reach its goal autonomously. The task of the sensor system is not only for collecting the information but to translate it to meaningful data to the control system. Acquainting communication techniques and interfaces to connect the sensors to the control system is also required to be able to develop a mobile robot platform. For example an autonomous robot could deliver parts between various assembly stations or stores. A robot team with on-board cameras and image processing system could guard a vast industrial or military area. Of course, to accomplish these tasks, a robot has to have a sophisticated sensor system, durable mechanic structure and highly developed computing system.

Figure1 shows the basic structure of a robot. [2]

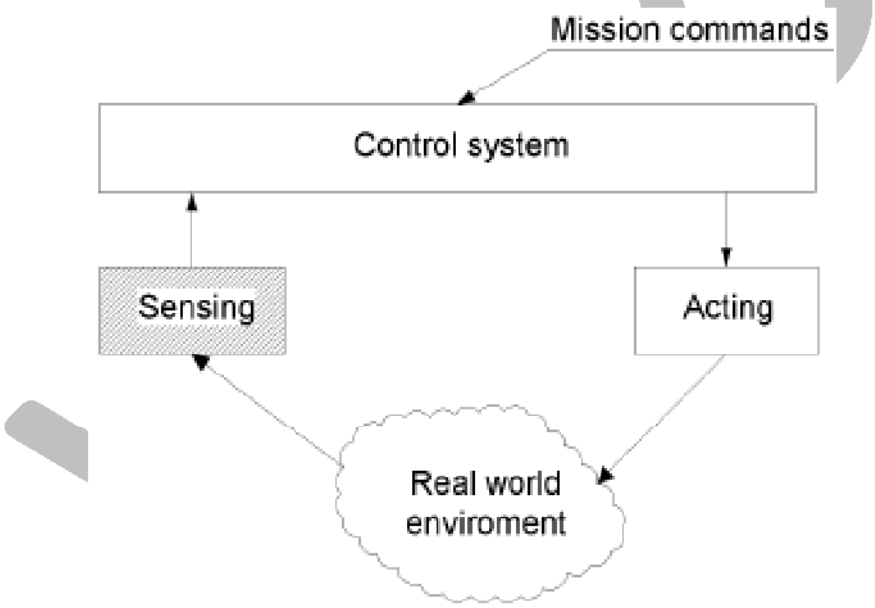


Figure 1.3. Scheme of mobile robot systems

1.2. Perception

One of the most important tasks of an autonomous mobile robot of any kind is to acquire some useful information about its environment. This is done by taking measurements using various sensors and then extracting meaningful information data and translates it to the control system. [3]

Sensors can be classified to proprioceptive and exteroceptive sensors.

***Proprioceptive* sensors:** measure values internally to the system (robot), like motor speed, wheel load, heading of the robot, battery status.

***Exteroceptive* sensors:** acquire information from the robots environment; like distances to objects, intensity of the ambient light, unique features.

Furthermore, Sensors can be classified to ***active*, energy emitting**- emit their proper energy and measure the reaction-*(for example scanners)* and ***passive* energy receiving** sensors- energy coming for the environment-*(for example CCD cameras)*.

The sensors collecting information from the real world environment can be arranged to other groups according to their functions. However, only depth perception is not adequate for mobile robot navigation. In order to safely navigation obstacle detection must be done by the robot perception system. Actually, not only obstacle detection but also position measurement of the robot has a crucial role during navigation. Because of that reason, Position measurement and motion planning sub system has been developed for safe navigation. *Perception system of mobile robots* can be easily divided into two main groups:

Obstacle avoidance: sensing dynamic or static obstacles.

***Localization*:** collecting data to determine the accurate position of the robot. [2]

1.2.1. Obstacle avoidance

Time of flight active ranging. Time of flight ranging make use of the propagation speed of an emitted wave and measure the traveling time. The transmitted wave can be sound, light or electromagnetic wave. In the case of mobile robots, the measuring range is usually between 5 cm and 10 20 m. If sound is transmitted, the propagation speed is342 m/s, so the time of the flight at a distance of 3 m is about 20 ms (to reach the target and get back) which is a measurable value. If the transmitted wave is a laser beam, the propagation speed is 3.108 m/s which is about 106 times faster than it was in the case of sound transmitter. The time of flight is 20 ns! If the target is even closer than 3 m it is a very challenging task to measure so short times with small error. On the other hand, if we need short scanning time and measuring huge amount of points, the propagation speed of the sound does not suit our requirements.

1.2.1.1. Ultrasonic sensors

The ultrasonic sensor's basic principle is to transmit ultrasonic wave packages and to measure the time it takes to get back to the receiver. The ultrasonic wave's typical frequency is between 40 and 180 kHz and it is generated by piezo-transcuders. The main disadvantage of the ultrasonic sensors is that it can only give information whether there is an obstacle in an area, but it cannot give information about the exact position. It is because the sound propagates in a cone with an angle of about 30 degrees It cannot be determined which direction the reflected wave came back within the distribution angle. There are several other drawbacks of ultrasonic sensors. The speed of the sound depends on the temperature, and the strength of the reflected signal depends on the acoustic behavior of the obstacle's material. Ultrasonic sensors have relatively slow cycle time. As it was described, at a distance of 3m, the traveling time is 20m. If more sensors are used, and interference needs to be avoided the scanning time multiplies by the number of sensors.

1.2.1.2. Laser range finder

A laser rangefinder is a rangefinder which uses a laser beam to determine the distance to an object. The most common form of laser rangefinder operates on the time of flight principle by sending a laser pulse in a narrow beam towards the object and measuring the time taken by the pulse to be reflected off the target and returned to the sender. Due to the high speed of light, this technique is not appropriate for high precision sub-millimeter measurements, where triangulation and other techniques are often used.

There are two methods to measure distance with laser light beam. One way, like in the case of ultrasonic sensors, laser pulses are transmitted and the reflection time is measured. The other easier method is to transmit 100% amplitude modulated light beam at a defined frequency and compare the phase shift between the transmitted and the reflected light. This scanner has much higher resolution than the ultrasonic sensor. Pointing the measuring beam to rotating mirror a fast scanning can be accomplished in a vertical dimension, but the whole scanner has to be able to move horizontally if 3D scanning is needed.

The distance between point A and B is given by:

$$D = \frac{ct}{2}$$

where c is the speed of light in the atmosphere and t is the amount of time for the round-trip between A and B.

$$t = \frac{\varphi}{\omega}$$

where φ is the phase delay made by the light traveling and ω is the angular frequency of optical wave.

Then substituting the values in the equation,

$$D = \frac{1}{2}ct = \frac{1}{2}\frac{c\varphi}{\omega} = \frac{c}{4\pi f}(N\pi + \Delta\varphi) = \frac{\lambda}{4}(N + \Delta N)$$

In this equation, λ is the wavelength c/f; $\Delta\varphi$ is the part of the phase delay that does not fulfill π (that is, φ modulo π); N is the integer number of wave half-cycles of the round-trip and ΔN the remaining fractional part. [4]

1.2.1.3. Triangulation-based active ranging

The triangulation-based measurement consists of a light source and a sensor installed at a defined distance to each other. The lights source transmits a known light pattern (known transmission vector) to the environment and the sensor measures the vector of the reflected light so the distance can be calculated.

i. 1D laser triangulation sensors

The distance measurement with 1D laser triangulation sensors uses a coherent laser beam and a line CCD installed at a defined "L" distance from each other. The orientation of the laser beam is known and the line CCD chip can map the vector of the reflected laser beam (Figure 1.5). The distance "D" can be calculated from the measured reflection and the given measuring layout.

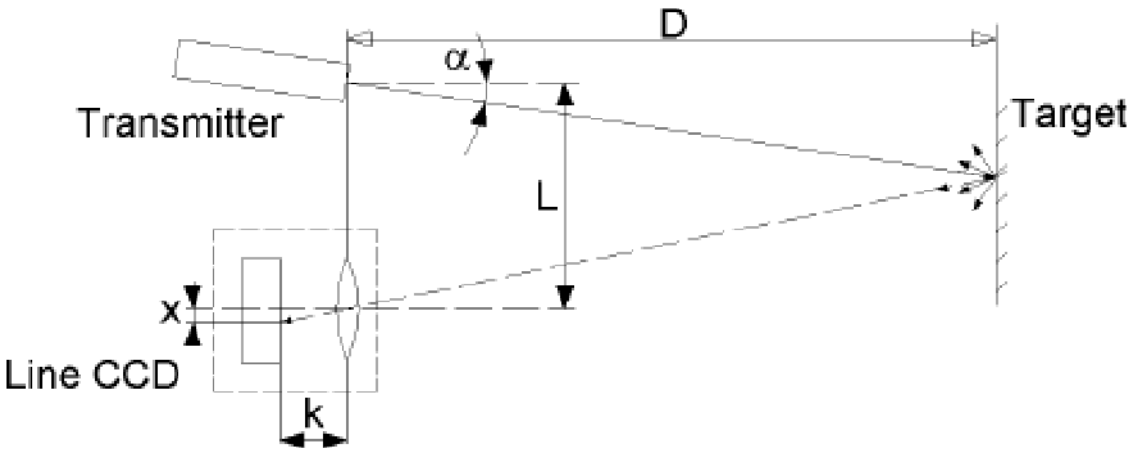


Figure 1.5. Scheme of 1D triangulation sensor

ii. 2. 2D laser triangulation sensor

The base principle of 2D triangulation sensors is the same as described before, but the transmitter is a line laser which excises a plane, and the reflected light is captured by a regular 2D CCD camera (Figure 1.6).

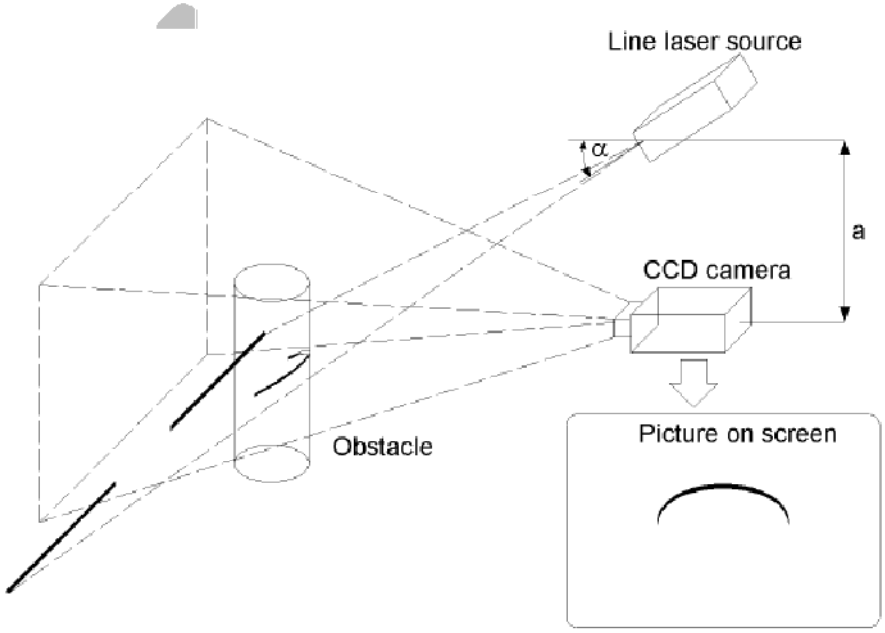


Figure1.6 Scheme of 2D laser range finders

- Optimizing the parameters of the 2D laser range finder

Choosing bigger, "a" makes the sensor less compact but it increases the distance measuring resolution. If only one scanning laser line is used, the obstacles hanging down at the height of the laser source cannot be detected and cause collision of the laser source with the obstacle (*Figure 1.7.A*).

With proper settings of laser source direction, rapid declension or hole can be detected. In the example shown at Figure 1.7.B,f the robot goes on a flat ground on the picture, captured by the CCD camera, there is always a line. If obstacle comes a part of the line goes up and the height of the line showed on the picture is in inverse proportion with the distance of the obstacle. If the line disappears, there has to be a gap or a hole.

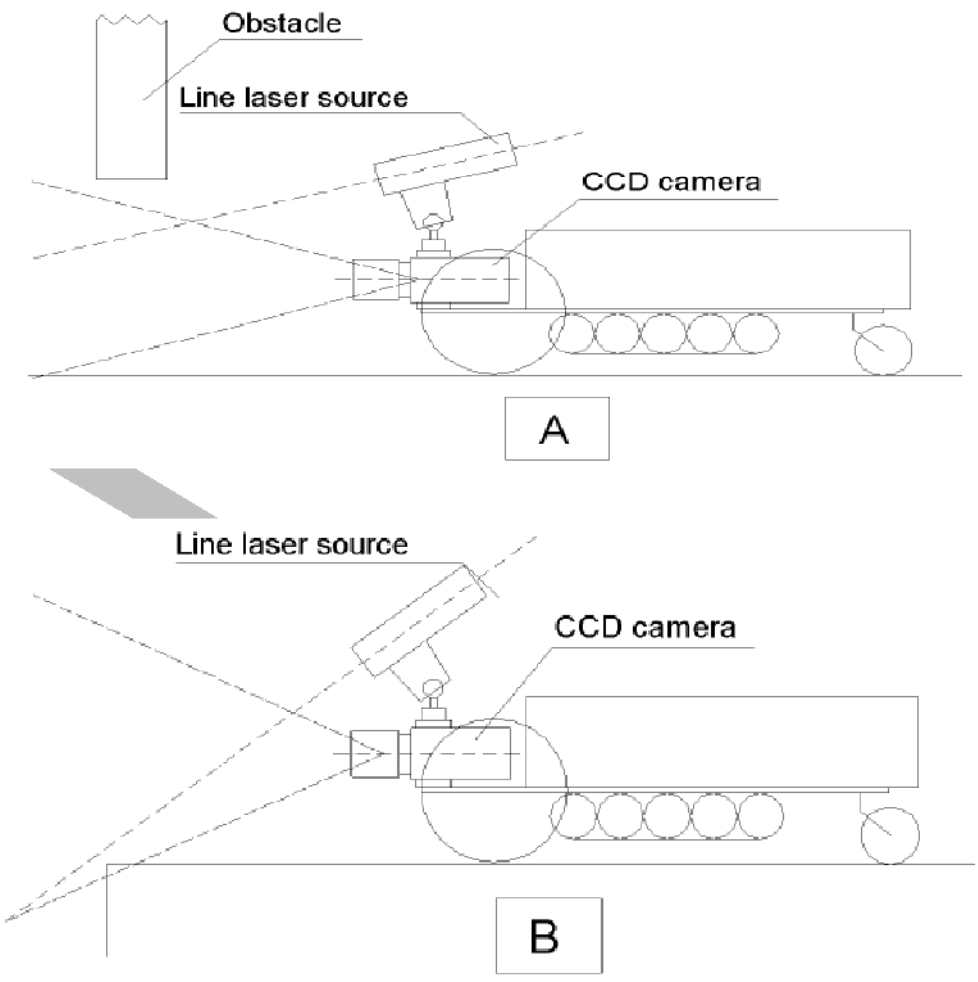


Figure 1.7. Opportunities and limitations of one laser line scanning

- FPGA based 2D laser range finder

The developed sensor is built on a Xilinx Spartan-3 FPGA platform and a Texas Instruments RGB video A/D converter. The used FPGA has 250.000 system gates and 172 I/O pins and it is driven by a 50 MHz quartz oscillator. At first the FPGA generate the proper sample clock frequency for the A/D by dividing the system clock.

The digitalization runs constantly and the 8 bit parallel dataflow is received by the FPGA. The second block separates the line and filed syncs.

The digitalization has two steps. First a complete bitmap picture is red into the memory "A" without the laser lines witched on. It takes 20 ms at a normal speed (25 frame/sec) camera since the composite PAL video signal sends two half pictures successively.

Using just a half picture decreases the resolution, but, what is more important, it decreases the scanning time as well. During the second digitalization the laser line is switched on and the bitmap is written into Memory "B". At the next step the FPGA extracts Memory "A" and Memory "B" and sends the results to a FIFO buffer. Using this method the laser beam can be emphasized from the bitmap. Because of the mechanical oscillations of the robot structure there are usually other differential errors on the picture, but they are not arranged but scattered therefore with a low pass filter the errors can be filtered out.

After the data procession about 720 byte of information come off. Each byte stands for a direction from the left to the right, and the value of the byte means the position (height) of the laser line at that specific direction.

For example if there is a narrow obstacle in front of the robot just in the middle and relatively close and the installation shown in Figure 1.7.B is used, the first ~300 bytes will be low values (depending on there life). The next ~100 bytes will be relatively high values, around 155 and the last ~300bytes will be small again. With the foreshowed sensor system around 1° resolution can be reached at a horizontal 45° angle. The scanning frequency can be about 20 Hz. The main limitation of the system is that it is very sensitive to the light conditions. At normal daylight the system cannot filter out the light of the laser beam from the surrounding sunlight. This drawback can be improved by using higher power lasers and optical filter at the wave length of the used laser.

In the first row of Figure 1.8 the captured image can be seen without the filtering. In the second row Median filter was used to remove impulse noise by replacing center pixels with median values in its neighborhoods.

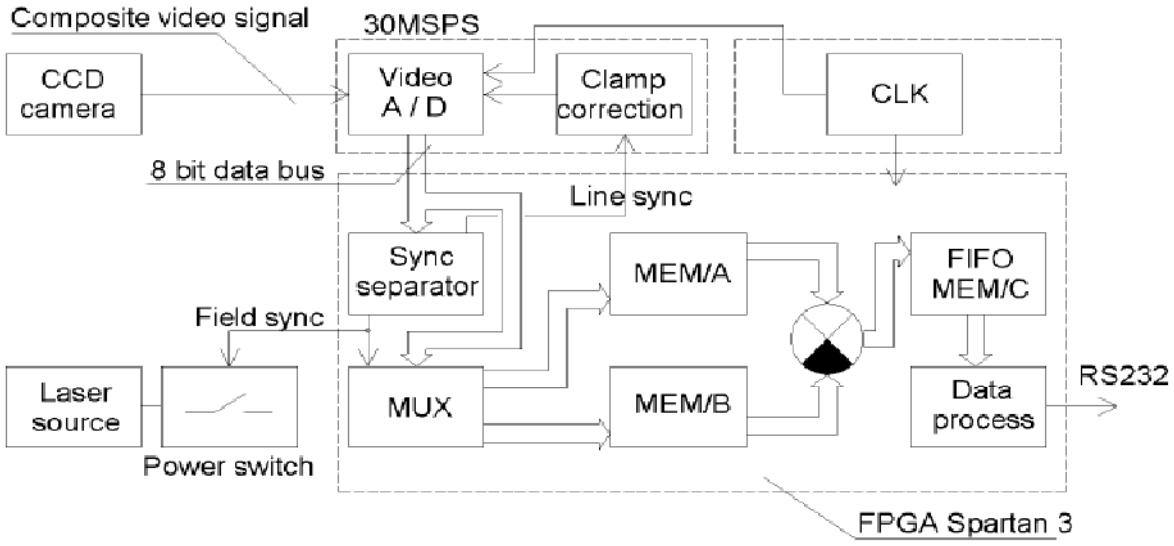


Figure 1.8. Scheme of 2D laser range finder

Vision and stereo image based sensors. This sensing method is the most progressive field of avoiding obstacles and sensing real environment. The basic principle is similar to the function of human vision, so it is a very complex and calculation-intensive task. Because of the limited volume of this article, this field cannot be detailed here. [8]

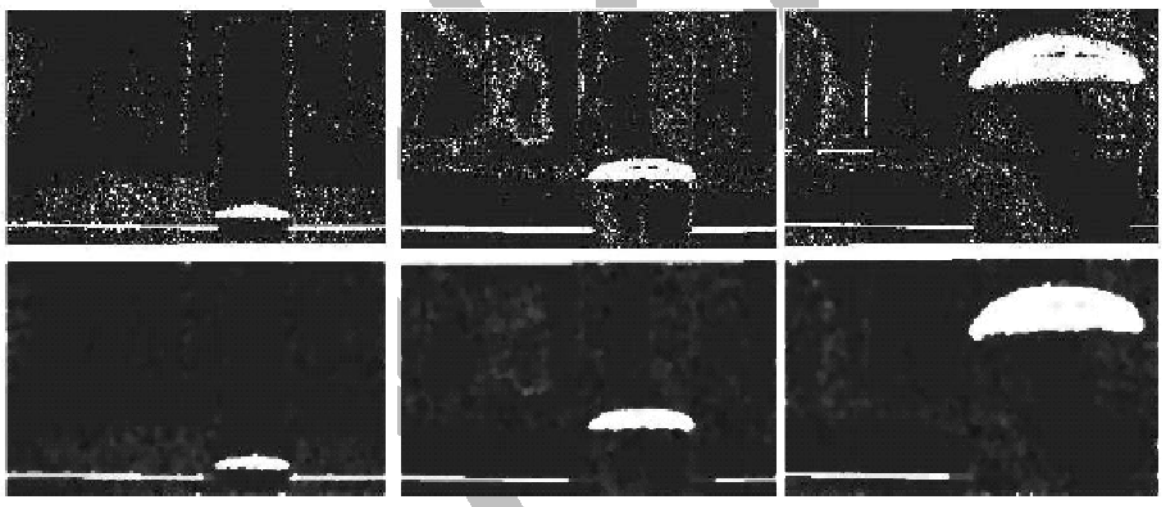


Figure 1.9. Captured image of 2D triangulation sensor

1.2.2. Localization

The sensor used for localization highly depends on the mechanic structure of the robot and the environment where the robot is used. In this section some commonly use mobile robot sensors are demonstrated.

1.2.2.1. Wheel encoders

Figure 1.10 shows a simple mechanical structure. Using wheel encoders, the rotation of the wheel can be measured very accurately. If flat area is assumed and non-kidding wheels are used the position can be calculated by simple geometric equations.

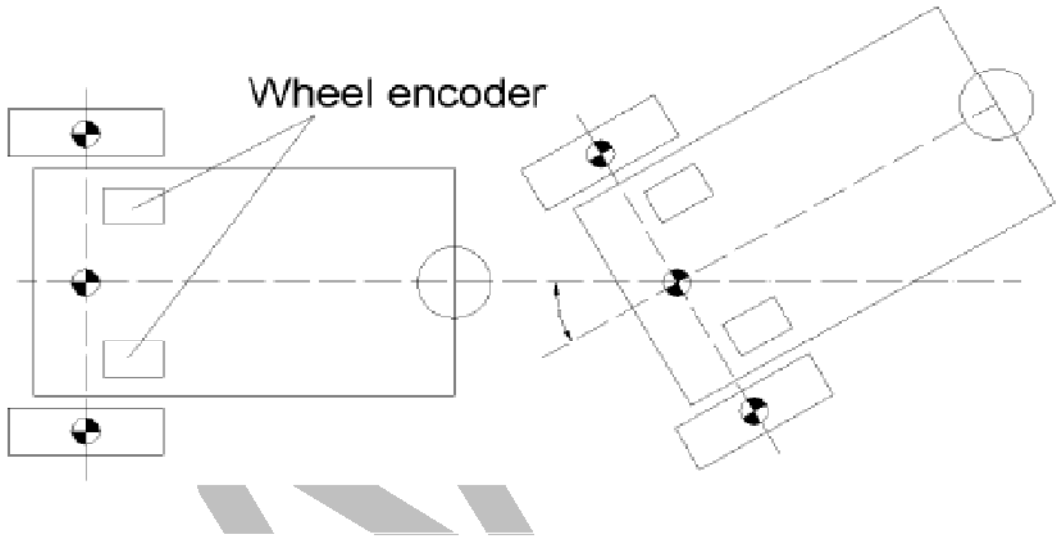


Figure 1.10. A simple mechanical structure example of a mobile robot

1.2.2.2. Gyroscope

There are two commonly used types of gyros in mobile robot systems:

Optical gyroscopes work on a principle that the speed of light remains unchanged in a closed mechanical system. If two rings of optical fiber are used and opposite directional laser beams are sent from the same source to the rings phase difference can be measured between the two laser beams in the case of mechanical rotation.

Solid state gyroscopes all work by detecting Coriolis forces. These are forces which can be observed whenever linear motion occurs in a rotating frame. The simplest form of Coriolis gyro, the simple oscillator, uses a single 'beam' of material (usually quartz or ceramic), which is vibrated to create a standing wave along its length. When subjected to rotation, this standing wave moves around the beam, causing it to vibrate in a new direction. This change in

the vibration position is proportional to the rate of rotation the gyro has been subjected to. This type of gyro is highly susceptible to external shocks and vibration.

1.2.2.3. Mechanical, Hall Effect and Fluxgate Compasses

The *Mechanic compasses* based on a permanent magnet mounted in a low-friction bearing. Four hall-effect semiconductors are arranged around the periphery of the magnet and detect the rotation of it. The rise up time can be 2.5 s so it's very slow. The *Fluxgate compasses* make use of the manner of the magnetic lines getting trough permeable materials. When a highly permeable material is placed into a uniform magnetic field, the magnetic lines of the field are traveling through the lowest resistance path, presented by the permeable material (Figure 1.11). However, if the permeable material is forced into saturation by an additional forcing field, the material does not influence the external field. If the forcing field is switched on an off periodically, the difference can be measured between the two states.

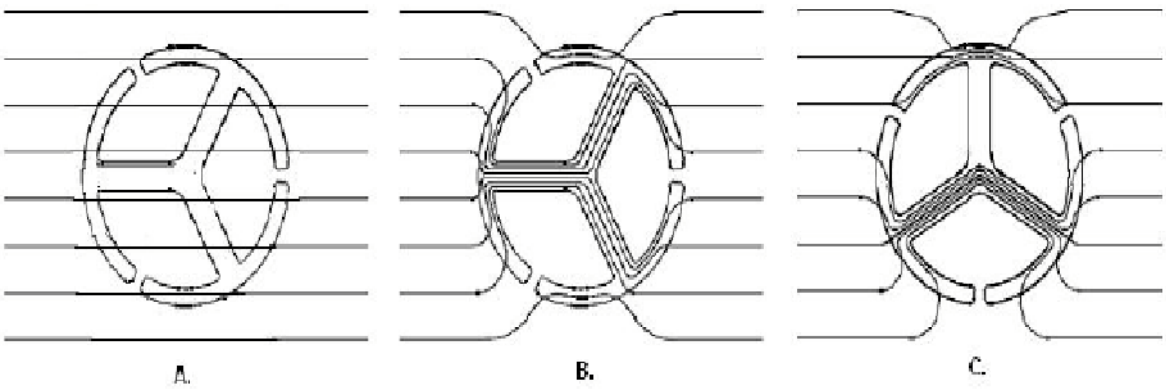


Figure1.11. Possible layout of a fluxgate compass. (A: saturated; B and C: not saturated)

Hall-Effect sensors are based on the effect that DC voltage can be measured across he conductor or semiconductor when external magnetic field being present.

The compasses always detect other magnetic field noises from the robot itself or from the environment therefore the measured values could be unreliable.

1.2.2.4. Global Positioning System (GPS)

If the mobile robot is used in an outdoor environment, for example in agriculture, GPS can be used to determine its position. The GPS was originally developed for military purposes and

great amount of satellites were put into orbit around the earth. The GPS receiver has to get at least three satellites signal to be able to determine its position. The satellites synchronize their transmission and send their location at the same time and the receiver measures the time it took to arrive and also the time differences of the received signals. From this data it can calculate its own position. The accuracy is 1m in the best case. GPS navigation cannot be used in indoor environment or in built-in area where the overlook is not covered (Figure 1.12). An average GPS receiver supplies information in every second so it has to be complemented by a proprioceptive sensor to ensure the continuous navigation.

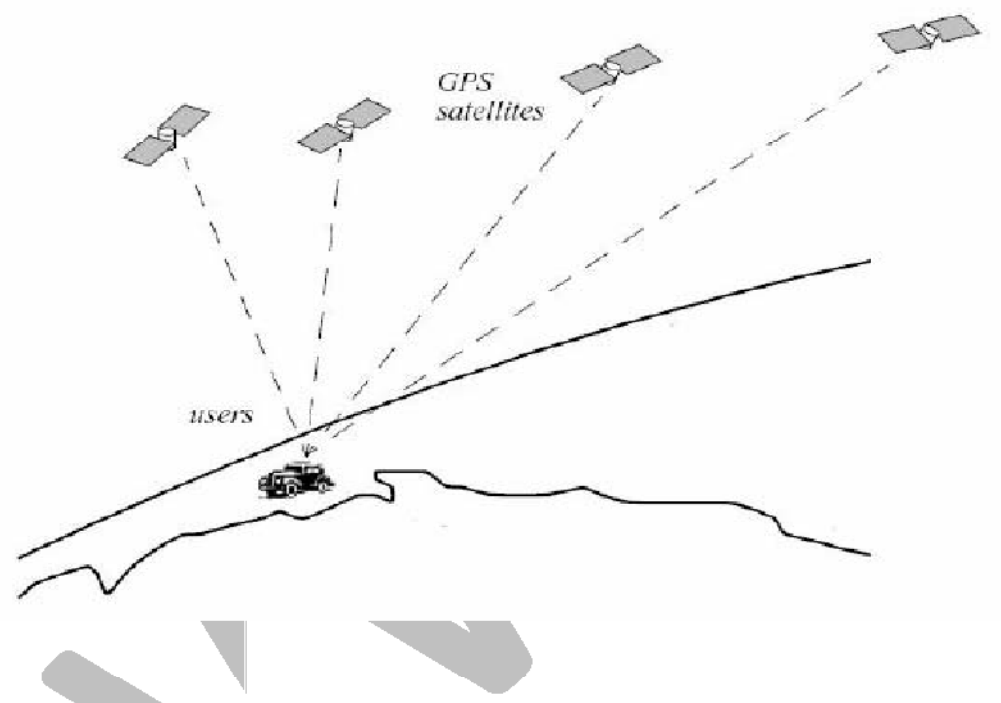


Figure 1.12. Global Positioning System

1.2.2.5. Camera posing system

If the mobile robot system is used in an indoor environment like in a manufacturing plat, an installed image processing system can recognize a specific mark on the board of the robot so it can determine the accurate position of the robot as well This system could be a complementary sensor to correct the calculated error of the wheel encoders or the gyroscope. [2]

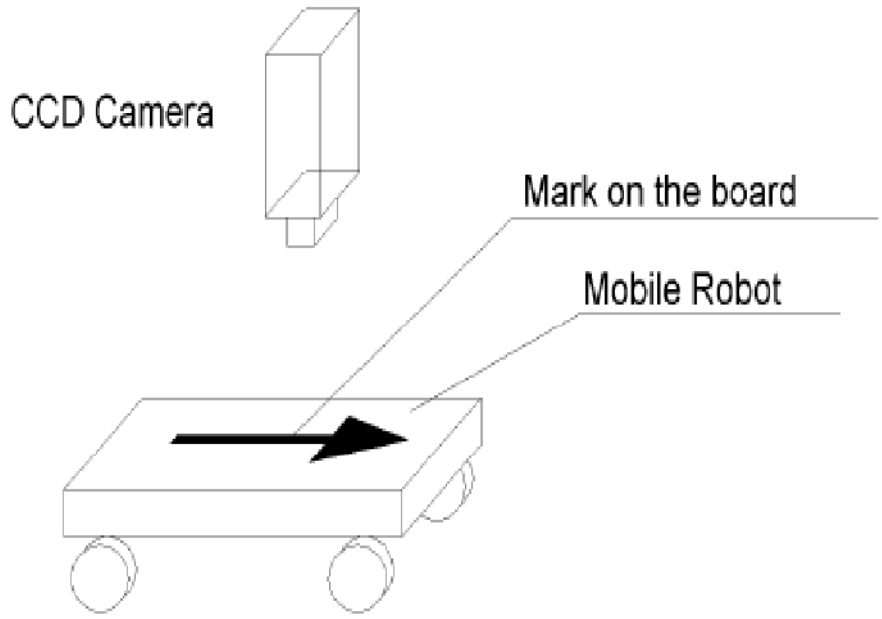


Figure 1.13. Camera posing system

1.3. Mobile Robot Navigation

For any mobile robotic system, the ability to navigate in its environment is one of the most important capabilities. Mobile Robots, which are equipped with computer vision, may be able to navigate around an unknown environment acquiring visual information of their surroundings with the aim of estimating the position of obstacles which stay in front of it. In other words, navigation can be roughly described as the process of determining a suitable and safe path between a starting and a goal point for a robot travelling between them.
Several capabilities are needed for autonomous navigation:

- _The ability to execute elementary goal achieving actions such as going to a given location or following a leader;

- _The ability to react to unexpected events in real time such as avoiding a suddenly appearing obstacle;

- _The ability to formulate a map of the environment;

- _The ability to learn which might include noting the location of an obstacle and of a three-dimensional nature of the terrain and adapt the drive torque to the inclination of hills.

The following, are common systems and methods for mobile robot navigation.

1. **Odometry and other dead-reckoning methods:** These methods use encoders to measure wheel rotation and/or steering orientation.

2. **Sensor based navigation:** Sensor based navigation systems that rely on sonar or laser scanners that provide one dimensional distance profiles have been used for collision and obstacle avoidance. A general adaptable control structure is also required. The mobile robot must make decisions on its navigation tactics; decide which information to use to modify its position, which path to follow around obstacles, when stopping is the safest alternative, and which direction to proceed when no path is given. In addition, sensors information can be used for constructing maps of the environment for short term reactive planning and long term environmental learning.

3. **Inertial navigation:** This method uses gyroscopes and sometimes accelerometers to measure the rate of rotation and acceleration.

4. **Active beacon navigation systems:** This method computes the absolute position of the robot from measuring the direction of incidence of three or more actively transmitted beacons.
   The transmitters, usually using light or radio frequencies must be located at known sites in the environment.

5. **Landmark navigation:** In this method distinctive artificial landmarks are placed at known locations in the environment to be detected even under adverse environmental conditions.

6. **Map-based positioning:** In this method information acquired from the robot's onboard sensors is compared to a map or world model of the environment. The vehicle's absolute location can be estimated if features from the sensor-based map and the world model map match.

7. **Biological navigation:** biologically-inspired approaches were utilized in the development of intelligent adaptive systems; biomimetic systems provide a real world test of biological navigation behaviors besides making new navigation mechanisms available for indoor robots.

8. **Global positioning system (GPS):** This system provides specially coded satellite signals that can be processed in a GPS receiver, enabling it to compute position, velocity, and time.

9. **Vision based navigation:** Computer vision and image sequence techniques were proposed for obstacle detection and avoidance for autonomous land vehicles that can navigate in an outdoor road environment. The object shape boundary is first extracted from the image, after the translation from the vehicle location in the current cycle to that in the next cycle, the position of the object shape in the image of the next cycle is predicted, then it is matched with the extracted shape of the object in the image of the next cycle to decide whether the object is an obstacle.[1]

Regardless of the type of vehicle, systems that use vision for navigation can be roughly divided in those that need previous knowledge of the whole environment *(Map-based Systems)* and those that perceive the environment as they navigate through it *(Mapless Navigation).*[5]

1.3.1. Map-based Systems

The systems that need a map can be in turn subdivided in *metric map using systems*, *metric map-building systems* and *topological map-based systems*. Another kind of *map-building navigation* systems can be found, as for example *visual sonar-based systems* or *local map-based systems*.

i. Metric Map-using and -building Navigation Systems

This group includes systems that need a complete map of the environment before the navigation starts. There are systems that are unable to map the environment and need to be equipped with it *(map-using systems)*. Other systems explore the environment and automatically build a map of it *(map-building systems)*. The navigation phase starts only if the map of the environment is available for the robot or after the map has been build. The map information can be directly used for navigation, or it can be post-processed to improve the

map accuracy, and thus, achieve a more precise localization. This is the navigation technique that requires more computational resources, time and storage capability.

Since outdoor environments can be large in size and extremely irregular, visual navigation techniques based on maps are in most occasions applied to indoor environments.

Map building and *self-localization* in the navigation environment are two functionalities that nonreactive systems tend to incorporate. In *map-building* standard approaches, it is assumed that the localization in the environment can be computed by some other techniques, while in pure *localization* approaches, the map of the environment is presumably available. Robots using this navigation approach need to track their own position and orientation in the environment in a continuous way.

Metric maps include information such as distances or map cell sizes with respect to a predefined coordinate system, and, in general, are also more sensible to sensor errors. Accurate metric maps are essential for good localization, and precise localization becomes necessary for building an accurate map.

If the exploration and mapping of an unknown environment is done automatically and on-line, the robot must accomplish three tasks: safe exploration/navigation, mapping and localization, preferably in a simultaneous way. *Simultaneous Localization and Mapping* (SLAM) and *Concurrent Mapping and Localization* (CML) techniques search for strategies to explore, map and self-localize simultaneously in unknown environments.

ii. Topological Map-based Navigation Systems

A topological map is a graph-based representation of the environment. Each node corresponds to a characteristic feature or zone of the environment, and can be associated with an action, such as turning, crossing a door, stopping, or going straight ahead. Usually, there are no absolute distances, nor references to any coordinate frame to measure space. These kinds of maps are suitable for long distance qualitative navigation, and specially for path planning. In general, they do not explicitly represent free space so that obstacles must be detected and avoided on line by other means. Topological maps are simple and compact, take up less computer memory, and consequently speed up computational navigation processes.

iii. Local Map-building Navigation Systems and Obstacle Avoidance

The strategies seen so far base their strength in a global description of the environment. This model can be obtained automatically by the robot, or in a previous human guided stage, but it has to be acquired before the robot begins the navigation. Since the early nineties, some

authors have developed solutions where visual navigation processes are supported by the on-line construction of a local occupancy grid. In vision-based navigation, the local grid represents the portion of the environment that surrounds the robot and the grid size is determined by the camera field of view. This local information can be used for a subsequent complete map construction or simply updated frame by frame and used as a support for on-line safe navigation. Since robot decisions depend, to a large extent, on what the robot perceives at every moment in the field of view, these navigation techniques arise a debate about what can be considered *deliberative* and what can be considered *reactive* vision-based navigation techniques.

iv. Visual Sonar

In recent years, *visual sonar* has become an original idea to provide range data and depth measurements for navigation and obstacle avoidance using vision in an analogous way to ultrasound sensors. Therefore, the originality of the concept is not in the navigation process itself, but in the way the data is obtained.

1.3.2. Mapless Navigation

The system is able to produce enough information about the unknown and just perceived environment to navigate through it safely.

Mostly, Mapless Navigation systems include reactive techniques that use visual clues derived from the segmentation of an image, optical flow, or the tracking of features among frames. No global representation of the environment exists; the environment is perceived as the system navigates, recognizes objects or tracks landmarks.

i. Optical Flow-based Navigation Systems

Optical flow can be roughly defined as the apparent motion of features in a sequence of images.

During navigation, the robot movement is perceived as a relative motion of the field of view, and, in consequence, it gives the impression that static objects and features move respect to the robot. To extract optical flow from a video stream, the direction and magnitude of translational or rotational scene feature movement must be computed at every pair of consecutive camera frames. Optical flow between two consecutive frames is usually represented by a vector for every pixel, where its norm depends on the motion speed and its direction represents the movement of the corresponding pixel in consecutive images. In some cases, the execution time and the computation resources required can be optimized by first extracting the image prominent features, such as for corners or edges, and then computing the

optical flow only for these features. Image optical flow has been used by some researchers to implement reactive mobile robot navigation strategies, either for indoor or for outdoor environments. Object boundaries appear as regions with significant optical flow, and thus as regions to be avoided. Specularities or irregularities on the floor and textured floors also appear as regions with optical flow and therefore can be wrongly considered as obstacles causing errors during navigation.

ii. Appearance-based Navigation

Appearance-based strategies consist of two procedures. First, in a pre-training phase, images or prominent features of the environment are recorded and stored as model templates. The models are labeled with certain localization information and/or with an associated control steering command. Second, in the navigation stage, the robot has to recognize the environment and self-localize in it by matching the current on-line image with the stored templates. The main problems of appearance-based strategies are: finding an appropriate algorithm to create the environment representation, and defining the on-line matching criteria. Deviations between the route followed in the guided pre-training phase and the route navigated autonomously yield different sets of images for each case, and thus differences in the perception of the environment.

Main researchers have focused their contributions on improving the way how images are recorded in the training phase, as well as on the subsequent image matching processes. There are two main approaches for environment recognition without using a map:

Model-based Approaches, They utilize pre-defined object models to recognize features in complicated environments and self-localize in it.

View-based Approach, No features are extracted from the pre-recorded images. The self-localization is performed using image matching algorithms.

iii. Image Qualitative Characteristics Extraction for Visual Navigation

Reactive visual techniques for robot navigation and obstacle avoidance are often devised around the extraction of image qualitative characteristics and their interpretation. There are two main types of reactive visual obstacle avoidance systems: *model-based* obstacle avoidance systems, which need predefined models of known objects, and *sensor-based* obstacle avoidance systems, which process every on-line sensor information to determine what could be an obstacle or what could be free space. These strategies can be included in what is known as qualitative navigation. Reactive navigation systems based on qualitative information avoid as much as possible using, computing or generating accurate numerical data such as distances, position coordinates, velocity, projections from image plane onto real

world plane, or contact time to obstacles. In general, a coordinated behavior-based architecture is needed to manage all qualitative image information and the subsequent reactions.

Of particular relevance to this sort of navigation systems, due to their critical dependence on unprocessed sensorial data, is the change of the imaging conditions: illumination intensity, position of light sources, glossiness of the scene materials, etc. As a consequence, and mostly for outdoor applications, depending on time, weather conditions, season, etc. the performance of certain visual navigation systems can be seriously limited.

iv. Navigation Techniques Based on Feature Tracking

Techniques for tracking moving elements (corners, lines, object outlines or specific regions) in a video sequence have become robust enough so as to be useful for navigation. Many times, the systems divide a tracking task into two sub-problems: first, *motion detection*, which, given a feature to be tracked, identifies a region in the next frame where it is likely to find such a feature, and second, *feature matching*, by which the feature tracked is identified within the identified region.

In general, feature tracking-based navigation approaches do not comprise an obstacle avoidance module, but this task has to be implemented by other means. Although video tracking and mobile robot navigation belong to separate research communities, some authors claim to bridge them to motivate the development of new navigation strategies. Some authors center their research in detecting and tracking the ground space across consecutive images, and steering the robot towards free space. [5]

2

Hough Transform based Recognition

2.1. Introduction, Computer Vision

Scientists and science fiction writers have been fascinated by the possibility of building intelligent machines, and the capability of understanding the visual world is a prerequisite that some require of such a machine. The important precise questions that we have to ask: What is the goal of computer vision, and which problems are we attempting to tackle? And how do we plane to solve them?

The goal of computer vision is to make useful decisions about real physical objects and scenes based on sensed images. In order to make decisions about real objects, it is almost always necessary to construct some description or model of them from the image. Because of this, many experts will say that the *goal of computer vision is the construction of scene descriptions from images.*

The target problem is computing properties of the 3-D world from one or more digital images. The properties that interest us are mainly *geometric* (for instance, shape and position of solid objects) and *dynamic* (for instance, object velocities). Most of the presented solutions assume that a considerable amount of image processing has already taken place; that is, new images have been computed from the original ones, or some image parts have been identified to make explicit the information necessary to the target computation.

2.2. Computer Vision tools and applications

Computer vision involves *computers interpreting images*. Therefore, the tools needed by computer vision system include hardware for acquiring and storing digital images in a

computer, processing the images, and communicating results to users or other automated systems. The part that interests us is the *algorithms* of computer vision; it contains very little material about hardware, but suffusing enough to realize where digital images come from. This does not mean that algorithms and software are the only important aspect of a computer vision system. On the contrary, in some applications, one can choose the hardware and can engineer the scene to facilitate the task of vision system; for instance, by controlling the illumination, using high-resolution cameras, or constraining the pose and the location of the objects. In many situations, however, one has little or no control over the scene. For instance, in the case of outdoor surveillance or autonomous navigation in unknown environments, appropriate algorithms are the key to success.

Image processing is a vast research area. For our purposes, it differs from computer vision in that it concerns *image properties* and *image to image transformations*; whereas the main target of computer vision is the 3-D world. As most computer vision algorithms require some preliminary image processing, the overlap between the two disciplines is significant. Examples of image processing include *enhancement* (computing an image of better quality than the original one), *compression* (devising compact representations for digital images; typically for transmission purposes), *restoration* (eliminating the effect of known degradations), and *feature extraction* (locating special image elements like contours, or textures areas).

Pattern recognition has produced techniques for *recognizing and classifying objects using digital images*. Many methods developed in the past worked well with 2-Dobjects or 3-Dobjects presented in constrained poses, but were unsuitable for the general 3-Dworld.

Photogrammetry is concerned with obtaining reliable and accurate measurements from noncontact imaging. The main differences are that photogrammetry pursues higher levels of accuracies than computer vision, and not all of computer vision is related to measuring.

– Research and application areas

Research areas refer to topics addressed by significant number of computer vision publications, such as Image feature detection, Contour representation, Feature based segmentation, Range image analysis, Shape modeling and representation, Shape reconstruction from single image cues (shape from X), Stereo vision, Motion analysis, Color

vision, Active and purposive vision, Invariants, Uncelebrated and self-calibrating systems, Object detection, 3-D object recognition and location, High-performance and real time architectures.

While application areas refer to domain in which computer vision methods are used, possibly in conjunction with other technologies, to solve real world problems. We can find: industrial inspection and quality control, surveillance and security, face recognition, autonomous vehicles[6]

2.3. Recognition

Recognition implies that object descriptions, or models, are already available; we cannot recognize what we do not know yet. *Model-based recognition* is the c*omparison* of *image data* with a *database of models*. When a model is found to correspond to a subset of the data (for example, particular configuration of contours), we say that a *match has been found*, or that *the model matches the data*, and the *matches model* is the *identity* of the object imaged (data and model represent the same object in the scene). For our purpose, object recognition entails two basic operations:

Identification: Determine the nature of objects imaged. For instance we may want to know whether there is a circle, for example, among the many objects in an image, or whether the only object we are looking at is indeed a circle. TheHough transform that will be presented later is a traditional method used for the recognition of 2D shapes.

Location: determines the position in the space of the object in view. [7]

2.4. Image Segmentation

Segmentation is the process that subdivides an image into a number ofuniformly homogeneous regions. Each homogeneous region is a constituentpart or object in the entire scene. In other words, segmentation of an image isdefined by a set of regions that are connected and nonoverlapping, so that eachpixel in a segment in the image acquires a unique region label that indicatesthe region it belongs to. Segmentation is one of the most important elementsin automated image analysis, mainly because at this step the objects or otherentities of interest are extracted from an image for subsequent processing, such as description and recognition. For example, in case of an aerial imagecontaining the ocean and land, the

problem is to segment the image initiallyinto two parts-land segment and water body or ocean segment. Thereafterthe objects on the land part of the scene need to be appropriately segmentedand subsequently classified. [8]

So, for many other image processing tasks, the basis is to *extract relevant areas of the image* as a unit. These units can either be regions containing potential objects,geometric primitives such as lines or circles, or point features. In this context,one distinguishes the segmentation of global views, i.e. containing an objectas a whole, and the segmentation of local features, i.e. features an object viewis composed of.

Humans are able to solve this so-called segmentation problem almost perfectly and effortlessly, influenced by a lot of background knowledge and experience. Computer vision, on the other hand, is still far from solving the segmentation problem as well as humans. In practice, only partial solutions are available, which are adapted to a specific setup and require certain assumptions in order to hold true. [9]

According to the application context and limits, we can apply one of the following approaches:

The segmentation by thresholding, like binarization, this approach allows to obtain a black (background) and white (foreground) image by using a threshold, previously defined, or separately defined for every pixel (Algorithms by Niblack and Sauvola),

The background subtraction, the basic principle is to store a view I_0 of the empty scene, which is subtractedfrom the images captured at all following time steps $t > 0$. For a gray scale image, the mapping gives:

$$\grave{I}(u,v) = \begin{cases} 255 & if\ |I_t(u,v) - I_0(u,v| > predefined\ threshold \\ 0 & otherwise \end{cases}$$

Region growing, in this algorithm, connected areas with similar gray value or color can be determined. On the basis of one seed point which can be determined manually or automatically, the surrounding area is searched. This is done by checking each of the four (or eight, for more accuracy) direct neighbors to see if they possess a similar (or smoothly different but close) value. If this is the case then the neighbor is added to the data structure for the points to be checked. This list is then processed until it is finally empty. In order to prevent pixels from being visited repeatedly, visited pixels must be marked.

Edge detectors, Edges, lines, and points carry a lot of information about the various regions in the image. These features are usually termed as localfeatures, since they are extracted from

the local property alone. Though theedges and lines are both detected from the abrupt change in the gray level,yet there is an important difference between the two. An edge essentiallydemarcates between two distinctly different regions, which means that anedge is the border between two different regions. A line, on the other hand,may be embedded inside a single uniformly homogeneous region. For example,a thin line may run between two plots of agricultural land, bearing the samevegetation.

The *edge detection* operation is essentially an operation to detect significantlocal changes in the intensity level in an image. The change in intensity levelis measured by the gradient of the image. [10]

The Hough transform can be used to identify the parameter(s) of a curve which best fits a set of given edge points. This edge description is commonly obtained from a feature detecting operator such as the Roberts Cross, Sobel or Canny Edge Detector and may be noisy; it may contain multiple edge fragments corresponding to a single whole feature. Furthermore, as the output of an edge detector defines only *where* features are in an image, the work of the Hough transform is to determine both *what* the features are and *how many* of them exist in the image. We need to explain first the concept of Edge detection through the canny type Before introducing the Hough transform in detail.

2.5. Canny Edge Detector

Canny edge detector ensures good noise immunity and at the same time detects true edge points with minimum error[16]. Canny has optimized the edge detection process by:

a) Maximizing the signal-to-noise ratio of the gradient,
b) An edge localization factor, which ensures that the detected edge is localized *as* accurately *as*possible,
c) Minimizing multiple responses to a single edge.

The signal-to-noise ratio of the gradient is maximized when true edges are detected and false edges are avoided. Thus by discarding the false responses when there are multiple number of responses to a single edge, the noise-corrupted edges may be removed. In this method the image is first convolved with Gaussian smoothing filter with standard deviation σ. This operation is followed by gradient computation on the resultant smoothed image.

2.5.1. Non-Maxima Suppression

The Canny's edge detector produces thick edges wider than a pixel. The operation of non maxima suppression thins down the broad ridges of gradient magnitude. There are several

techniques for such a thinning operation. In one technique, the edge magnitudes of two neighboring edge pixels, perpendicular to the edge direction are considered and the one with lesser edge magnitude is discarded.

2.5.2. Double Thresholding

The gradient image obtained after non-maxima suppression may still contain many false edge points. To remove false edge points, an appropriate threshold is selected such that all the edge points having magnitude greater than the threshold may be preserved as true edge points, while others are removed as false edge points. If the threshold is small, then a number of false edge points may be detected as true edge points, otherwise some true edge points may be missed.

To avoid this problem, two thresholds T_1 and T_2may be chosen to create two different edge images $E_1$and E_2, where $T_2 = 1.5T_1$. E_1will contain some false edge points, whereas E_2will contain very few false edge points and miss a few true edge points. Threshold selection algorithm starts with the edge points in E_2, linking the adjacent edge points in E_2forming a contour, and the process continues till no more adjacent edge points are available. At the boundary of the contour the algorithm searches for the next edge points from the edge image E_1in its 8-neighborhood. The gaps between two edge contours may be filled by taking edge points from E_1till the gap has been completely filled up. This process yields complete contour constituted by the true edges of the image.

2.5.3. Edge Threshold Selection

The detection of edges is based on comparing the edge gradient with a threshold. This threshold value can be chosen low enough only when there is no noise in the image, so that all true edges can be detected without miss. In noisy images, however, the threshold selection becomes a problem of maximum likelihood ratio optimization. [11]

2.6. The Hough Transform

The Hough Transform is a method for detecting straight lines, ellipses or curves defining the contours of objects in gray tone (color) images. The method is given the family of curves being sought and produces the set of curves from the family that appear on the image.

2.6.1. The Hough Transform for lines

The two basic steps of HT for lines are:

- *Transform line detection into a line intersection problem:* the line $y = mx + n$ identified by the unique pair, (m, n) is represented by a point in the m, n plane *(the parameter space)*. Conversely, any point $p = [x, y]^T$ in the image corresponds to a line $n = x(-m) + y$ in parameter space, which as m and n vary, represents all possible image lines through$p$; where the line defined by N collinear image points, $p_1 \dots p_N$, is identified in the parameter space by the intersection of the lines associated with $p_1 \dots p_N$, as illustrated in Figures; 2.1, 2.2and 2.3, For $N = 2$.
- *Transform line intersection in a simple peak detection problem:* or search for a maximum. Depending on the resolution (accuracy) we need, we divide the m, n plane into a finite grid of cells, and associate a counter,$c(m, n)$ initially set to zero, to each cell. Assume that the image contains a line$(\acute{m}, \acute{n})$, formed by points $p_1 \dots p_N$, for each image point p_i, increment all counters on the corresponding line in parameter space. All parameter space lines $l_1 \dots l_N$ associated to $p_1 \dots p_N$, go through $(\acute{m}, \acute{n})$, so that $c(\acute{m}, \acute{n}) = N$. Any other counter on $l_1 \dots l_N$is 1. Therefore the image line is identified by the peak of $c(m, n)$ in the parameter space.

To avoid missing important parameter ranges because m and n can take on values in $[-\infty, +\infty]$, which implies that we cannot sample the whole parameter space, in addition the equation $y = mx + b$ for straight lines does not work for vertical lines, so we use the polar representation: $d = x\cos\theta + y\sin\theta$ where d is the distance from the line to the origin and θ is the line orientation. The intervals of variation of d and θ are finite, and any line can be represented. This operation keeps parameter space finite, but at the price of reducing resolution, as there is a limit to the size of the discrete parameter space that we can search at acceptable time. [7]

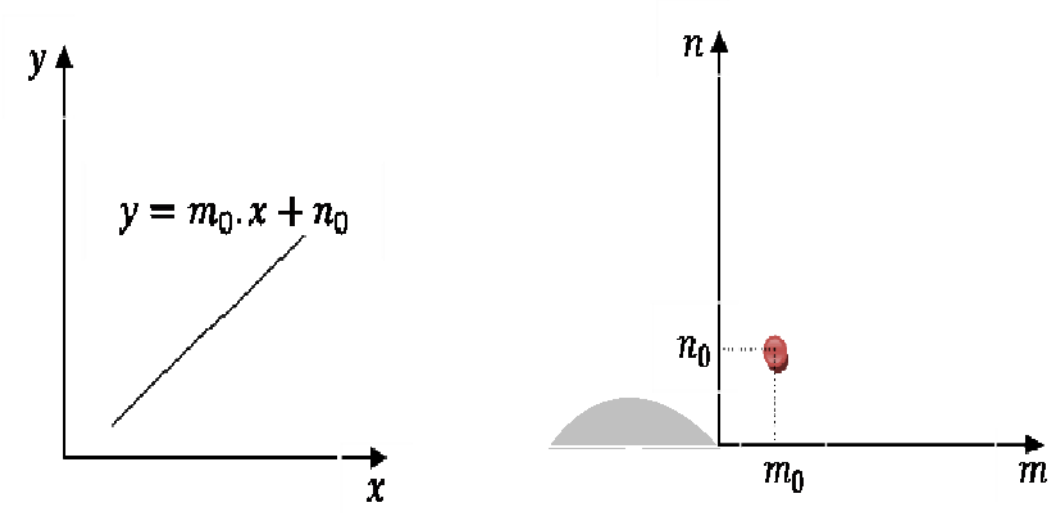


Image space *Hough parameter space*

Figure 2.1: A line in the image corresponds to a point in the Hough space

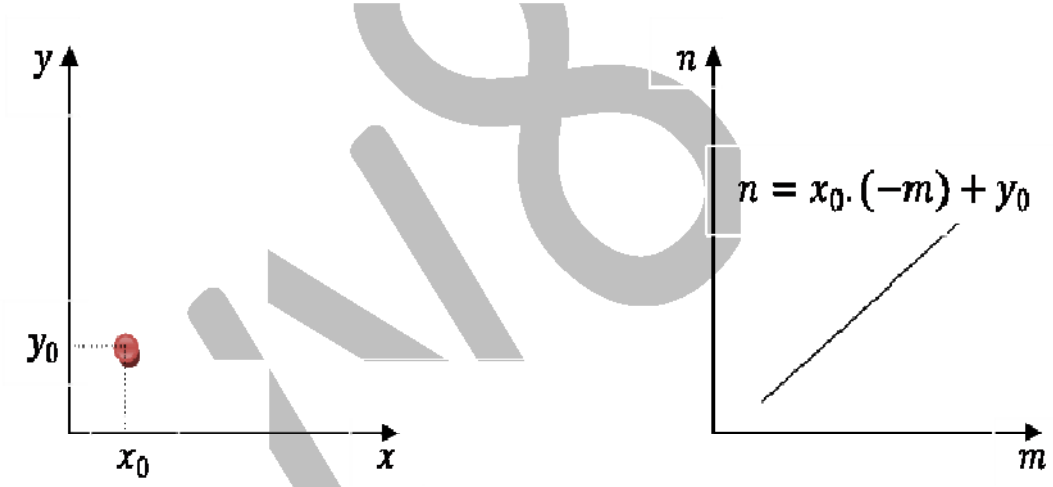


Image space *Hough parameter space*

Figure 2.2: A point in the image corresponds to a line in the Hough space

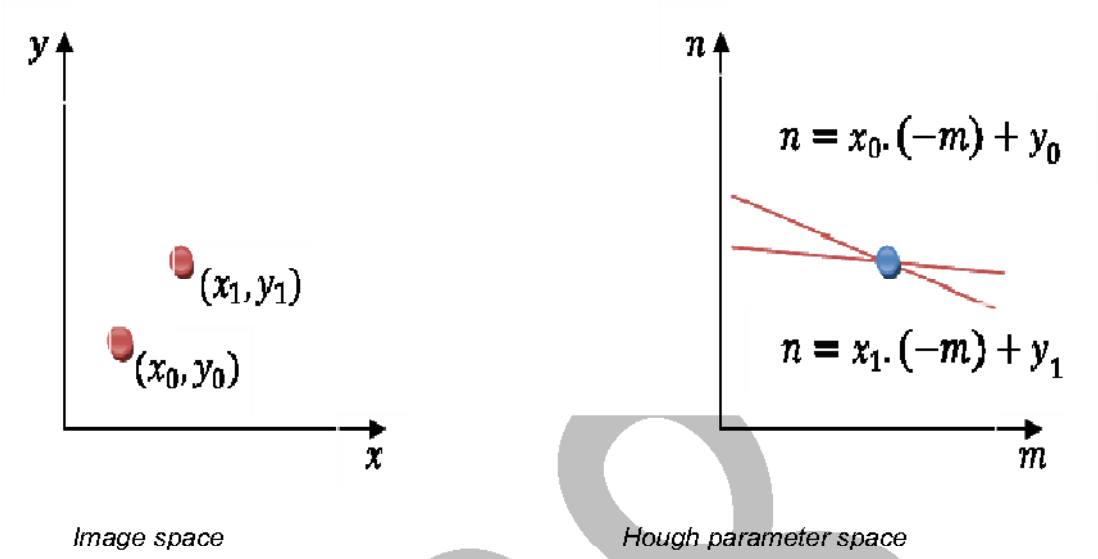


Figure 2.3: Transform line detection into a line intersection problem

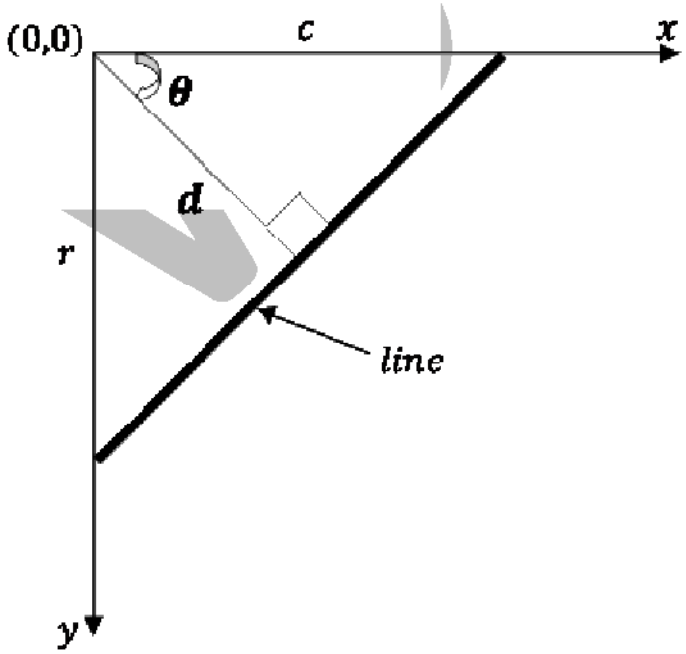


Figure 2.4: polar representation of lines

Note

i) Notice that any image point is now represented by sinusoid, not a line, in parameter space.

ii) Edge images can contain multiple lines, to find them all, we must look for all *local maxima*of $c(m,n)$.

iii) Edge images usually contain points not belonging to any line, for instance curved contours, or just noise introduced by the edge detector. These points result in spread of low, random counter values throughout the parameter space. Consequently, a multitude of local, noisy peaks appears, and we must divide it from those identifying lines. The simplest way to achieve this is to threshold $c(m,n)$.

iv) Because of pixelization and the limited accuracy of edge detection-presence of noisy points-, not all the lines of parameter space corresponding to the points of an image line intersect at the same point. Consequently, the image points contribute to *several counters* within a small neighborhood of the correct one, so the peak is spread over that neighborhood. Depending on the resolution of parameter space and the accuracy required, one can estimate the true parameters just as the local maximum $(\acute{m},\acute{n})$, or as weighted average of the values (m,n) in a neighborhood of $(\acute{m},\acute{n})$ with the condition:

$c(m,n) > \tau c(\acute{m},\acute{n})$, where τ is a fixed fraction (e.g., 0.9) and the weights are proportional to the counters values. [7]

2.6.2. The Hough Transform for circles

$$u = u_m + r\cos\theta$$
$$v = v_m + r\sin\theta \qquad (2.1)$$

Based on this equation a suitable formulation for HT can be derived, $\theta\epsilon[0,\pi]$:

$$u_m = u - r\cos\theta$$
$$v_m = v \pm r\sin\theta \qquad (2.2)$$

Where r is the radius, u_m is the row coordinate of the center and v_m is the column coordinate of the center.

- The Hough Circle Transform works in a *roughly* analogous way to the Hough Line Transform explained in the previous tutorial.

- In the line detection case, a line was defined by two parameters (d, θ). In the circle case, we need three parameters to define a circle:$C(u_m, v_m, r)$

  Where(u_m, v_m)define the center position and $r$is the radius, which allows us to completely define a circle.

2.6.3. The Generalized Hough Transform

The generalized Hough transform is used when the shape of the feature that we wish to isolate does not have a simple analytic equation describing its boundary. In this case, instead of using a parametric equation of the curve, we use a look-up table to define the relationship between the boundary positions and orientations and the Hough parameters. (The look-up table values must be computed during a preliminary phase using a prototype shape).

For example, suppose that we know the shape and orientation of the desired feature (Figure 2.5) We can specify an arbitrary reference point (x_{ref}, y_{ref})within the feature, with respect to which the shape (the distance $r$and angle of normal lines drawn from the boundary to this reference point ω) of the feature is defined. Our look-up table (*R-table*) will consist of these distance and direction pairs, indexed by the orientation$\omega$of the boundary.

The Hough transform space is now defined in terms of the possible positions of the shape in the image (the possible ranges of $(\boldsymbol{x_{ref}}, \boldsymbol{y_{ref}})$). In other words, the transformation is defined by:

$$\begin{aligned} x_{ref} &= x + r\cos\beta \\ y_{ref} &= y + r\sin\theta \end{aligned} \tag{2.3}$$

The r andβvalues are derived from the R-table for particular known orientationω. If the orientation of the desired feature is unknown, this procedure is complicated by the fact that we must extend the accumulator by incorporating an extra parameter to account for changes in orientation. [12].

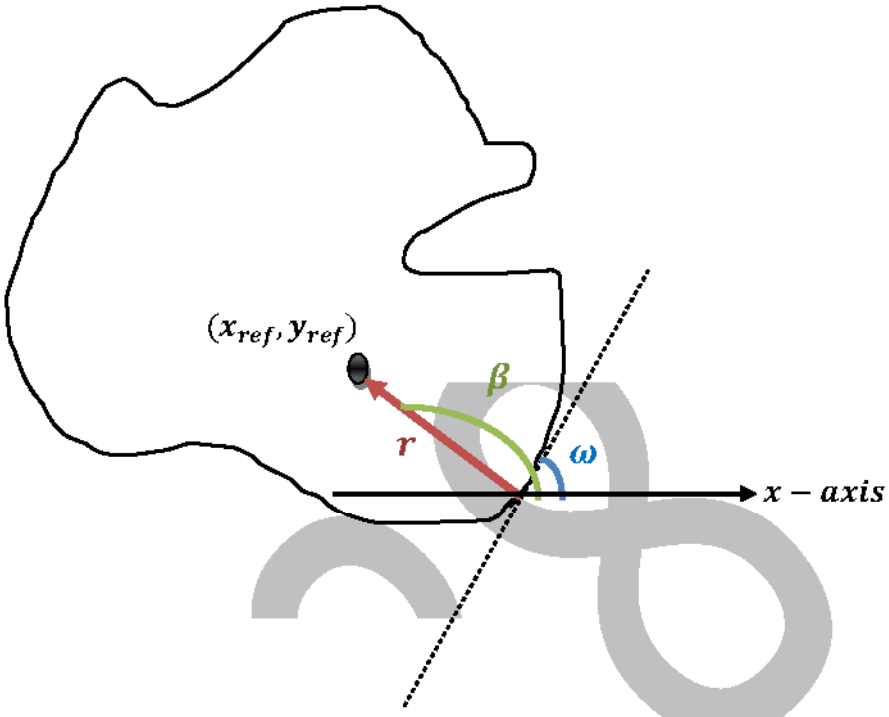


Figure 1.5. Description of R-table components

2.6.4. The Hough Transform Algorithms [10]

Lines, (inputs: a binarized input image I, of height h and width w,
a: number of discretizations of the angle $\theta \in [0, \pi]$,)
(output: H: the two-dimensional Hough space)

$$n = 1 + 2\left(\text{ent}(\sqrt{h^2 + w^2}) + 1\right)$$

```
for t=0 to a-1 do
     for r=0 to n-1 do
     H(r,θ) = 0
     end for
end for

for all pixels (u,v) in I do
     ifI(u,v)=q then
```

```
            for t=0 to a-1 do
            θ = tπ/a
            r = u cos θ + v sin θ
            r̀ = round(r + (n-1)/2)
            H(t, r̀) = H(t, r̀) + 1
            end for
      end if
end for
```

Circles, (inputs: a binarized input image $\boldsymbol{I}$, of height $\boldsymbol{h}$ and width $\boldsymbol{w}$,
$\boldsymbol{a}$: number of discretizations of the angle $\theta \in [0, \pi]$,
r_{min}, r_{max})

(output: $\boldsymbol{H}$: the two-dimensional Hough space)

```
n=w+2r_max
m=h+2r_max
for r=r_min to r_max do
      for v=0 to m-1 do
            for u=0 to n-1 do
            H(r, v, u) = 0
            end for
      end for
end for

for all pixels (u,v) in I do
      if I(u,v)=q then
            for t=0 to a-1 do
            θ = tπ/a
                  for r=r_min to r_max do
                  u_m = round(u − r cos θ)
                  v_m,1 = round(v − r sin θ)
                  v_m,2 = round(v + r sin θ)
                  H(r − r_min, v_m,1 + r_max, u_m + r_max) =
                  1 + H(r − r_min, v_m,1 + r_max, u_m + r_max)
```

$$H(r - r_{min}, v_{m,2} + r_{max}, u_m + r_{max}) =$$
$$1 + H(r - r_{min}, v_{m,2} + r_{max}, u_m + r_{max})$$

```
                end for
            end for
        end if
end for
```

3

The perception system modeling considerations.

–Camera Calibration and the Artificial Neural Networks based nonlinear approximation

3.1. Camera calibration

3.1.1. Objectives and techniques overview

Camera calibration is a necessary step in 3D computer vision in order to extract metric information from 2D images. It has been studied extensively in computer vision and photogrammetry, and even recently new techniques have been proposed.

According to the dimension of the calibration objects, we can classify those techniques roughly into three categories.

i. **3D reference object based calibration**

Camera calibration is performed by observing a calibration object whose geometry in 3-D space is known with very good precision. Calibration can be done very efficiently.

The calibration object usually consists of two or three planes orthogonal to each other. Sometimes, a plane undergoing a precisely known translation is also used, which equivalently provides 3D reference points. This approach requires an expensive calibration apparatus andan elaborate setup.

ii. **2D plane based calibration**

Techniques in this category requireobserving a planar pattern shown at a few different orientations.

The knowledge of the plane motion is not necessary,because almost anyone can make such a calibrationpattern by him/her-self, the setup is easier for camera calibration.

iii. **1D line based calibration**

Calibration objects used in this category arecomposed of a set of collinear points, a cameracan be calibrated by observing a moving line around a fixed point, suchas a string of balls hanging from the ceiling.

iv. **Self-calibration**

Techniques in this category do not use any calibrationobject, and can be considered as 0D approach because only imagepoint correspondences are required. Just by moving a camera in astatic scene, the rigidity of the scene provides in general two constraints on the cameras' internal parameters from one cameradisplacement by using image information alone. Therefore, if imagesare taken by the same camera with fixed internal parameters, correspondencesbetween three images are sufficient to recover both theinternal and external parameters which allow us to reconstruct 3-Dstructure up to a similarity. Although no calibration objectsare necessary, a large number of parameters need to be estimated, resultingin a much harder mathematical problem.Other techniques exist: vanishing points for orthogonal directions,and calibration from pure rotation.

Before going further, I'd like to point out that no single calibration techniqueis the best for all. It really depends on the situation. [10]

3.1.2. Camera model and camera calibration

The camera model consists of the intrinsic camera parameters $f_x, f_y, c_x, c_y$and$d_1 \dots d_4$, and the extrinsic camera parametersR, t.

3.1.2.1. Coordinate Systems

First, the three commonly used coordinate systems are defined; an illustration is given in Fig.1.

Image coordinate system: The image coordinate system is a two-dimensionalcoordinate system. Its origin lies in the top left-hand corner of theimage, the u-axis points to the right, the v-axis downward. The units are inpixels.

Camera coordinate system: The camera coordinate system is a three-dimensionalcoordinate system. Its origin lies in the projection center Z, the x-and y-axes run parallel to the u- and v-axes of the image coordinate system.The z-axis points forward i.e. toward the scene. The units are in millimeters.

World coordinate system: The world coordinate system is a three-dimensionalcoordinate system. It is the basis coordinate system, and can lie anywherein the area arbitrarily. The units are in millimeters.[1]

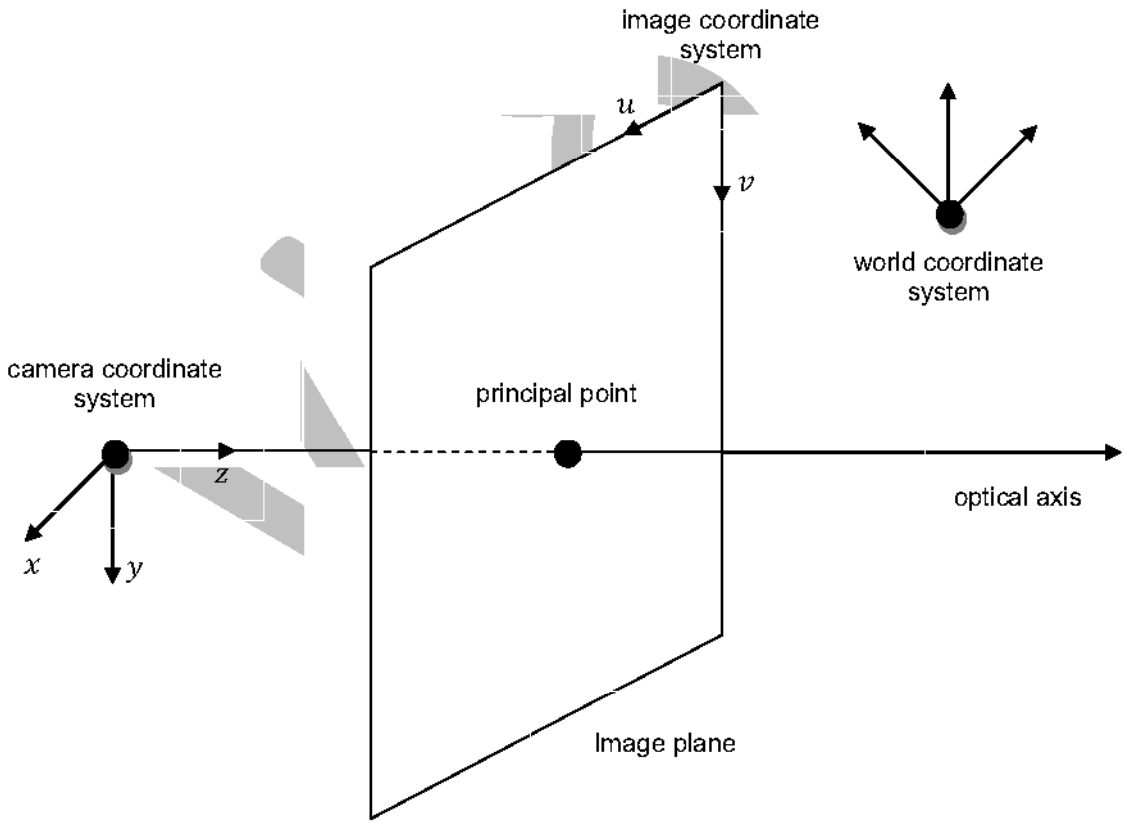


Figure3.1.Illustration of the coordinate systems of the camera model.
The optical axis and the image plane are perpendicular.
The x- and the u-axes run parallel, as do the y- and v-axes.

3.1.2.2. Intrinsic Camera Parameters of the Linear Mapping. [12]

The linear mapping function of the camera model is described by the intrinsiccamera parameters f_x, f_y, c_x, c_y. The parameters f_x and f_y denote the cameraconstants in u- and v-direction, usually referred to as the focal length, theunits are in pixels. They contain the conversion factor from [mm] to [pixels],independently for each direction, and can therefore also model non-square pixels.The principal point (c_x, c_y) is the intersection of the optical axis with the image plane, specified in image coordinates. Using a purelylinear projection defined by the intrinsic parameters f_x, f_y, c_x, c_y, the mapping from cameracoordinates $x_c, y_c, z_c$to image coordinates u, vreads

$$\begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} c_x \\ c_y \end{pmatrix} + \frac{1}{z_c}\begin{pmatrix} f_x c_x \\ f_x c_y \end{pmatrix} \tag{3.1}$$

This mapping can also be formulated as a matrix multiplication with the calibration matrix

$$K = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix}. \tag{3.2}$$

using homogeneous coordinates:

$$\begin{pmatrix} u.z_c \\ v.z_c \\ z_c \end{pmatrix} = K\begin{pmatrix} x_c \\ y_c \\ z_c \end{pmatrix} \tag{3.3}$$

The inverse of this mapping is ambiguous; the possible points $(x_c, y_c, z_c,)$ that are mapped to the pixel (u, v) lie on a straight line through the projection center. It can be formulated through the inverse calibration matrix

$$K^{-1} = \begin{pmatrix} \frac{1}{f_x} & 0 & -\frac{c_x}{f_x} \\ 0 & \frac{1}{f_y} & -\frac{c_y}{f_y} \\ 0 & 0 & 1 \end{pmatrix} \tag{3.4}$$

and the equation

$$\begin{pmatrix} x_c \\ y_c \\ z_c \end{pmatrix} = K^{-1} \begin{pmatrix} u.z_c \\ v.z_c \\ z_c \end{pmatrix} \quad (3.5)$$

Here the depth $z_c$is the unknown variable, for each z_c, the coordinatesx_c,$y_c$of the point $(x_c, y_c, z_c,)$ that maps to the pixel (u, v) are calculated. In line with the notation from Eq. (1), the mapping defined by Eq. (5) can analogously be formulated as

$$\begin{pmatrix} x_c \\ y_c \\ z_c \end{pmatrix} = z_c \begin{pmatrix} \frac{u-c_x}{f_x} \\ \frac{v-c_y}{f_y} \\ 1 \end{pmatrix} \quad (3.6)$$

3.1.2.3. Extrinsic Camera Parameters

Arbitrary twists and shifts between the camera coordinate system and theworld coordinate system are modeled by the extrinsic camera parameters. They define a coordinate transformation from the world coordinate system tothe camera coordinate system, consisting of a rotation Rand a translation$t$:

$$x_c = R.x_w + t \quad (3.7)$$

where$x_w = (x, y, z)$ define the world coordinates and $x_c = (x_c, y_c, z_c,)$the camera coordinates of the same 3D point. The complete mapping from the world coordinate system to the image coordinate system can finally be described in closed-form by the projection matrix

$$P = K(R\backslash t)$$

using homogeneous coordinates:

$$\begin{pmatrix} u.z_c \\ v.z_c \\ z_c \end{pmatrix} = P \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \quad (3.8)$$

The inversion of the mapping from Eq. (7) reads

$$x_w = R^T.x_c - R^T.t \quad (3.9)$$

Thus the inverse of the complete mapping from Eq. (8), which is ambiguous like the inverse mapping from Eq. (5), can be formulated as:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = P^{-1} \begin{pmatrix} u.\,z_c \\ v.\,z_c \\ z_c \\ 1 \end{pmatrix} \tag{3.10}$$

with the inverse projection matrix

$$P^{-1} = R^T(K^{-1} \backslash -t)$$

3.1.2.4. Distortion Parameters

The intrinsic camera parameters $d_1 \ldots d_4$model the effects that are caused by lens distortions, which lead to non-linearities during the camera mapping. The most important kind of distortion is radial lens distortion, which arises particularly strongly from lenses with a small focal length≤ 4mm.

So far,u, v have denoted the image coordinates for the purely linear mapping.Now, u, v denote the undistorted image coordinates, i.e. those coordinates that are calculated by the pure linear mapping from 3D to 2D. Additionally, the distorted image coordinates u_d, v_d are now introduced. These are the coordinates of that point that is eventually mapped onto the image sensor. The task of modeling the lens distortions is the mathematical description of the relationship between the undistorted image coordinates u, v and the distorted image coordinatesu_d, v_d. For this, usually $u_d, v_d$are expressed as a function ofu, v. As the basis for the following calculations serve the projection of u, vonto the plane $z = 1$in the camera coordinate system. The result of thisprojection is calculated by

$$\begin{pmatrix} x_n \\ y_n \end{pmatrix} = \begin{pmatrix} \frac{u - c_x}{f_x} \\ \frac{v - c_y}{f_y} \end{pmatrix} \tag{3.11}$$

From the coordinates x_n, y_n the distorted coordinates x_d, y_d are then calculated in the plane $z = 1$ in accordance with the distortion model. For the distorted image coordinates it applies

$$\begin{pmatrix} u_d \\ v_d \end{pmatrix} = \begin{pmatrix} f_x x_d + c_x \\ f_y y_d + c_y \end{pmatrix} \tag{3.12}$$

On the basis of the radius $\sqrt{(x_n{}^2 + y_n{}^2)}$ the corrective terms for the description of the lens distortion are defined. If two parameters d_1, d_2 are used for radial lens distortion, then the relationship between x_d, y_d and x_n, x_ncanbe expressed as

$$\begin{pmatrix} x_d \\ y_d \end{pmatrix} = (1 + d_1 r^2 + d_2 r^4) \begin{pmatrix} x_n \\ y_n \end{pmatrix} \tag{3.13}$$

Independently from radial lens distortion, tangential lens distortion can be described with the two parameters d_3, d_4 by the relationship

$$\begin{pmatrix} x_d \\ y_d \end{pmatrix} = \begin{pmatrix} x_n \\ y_n \end{pmatrix} + \begin{pmatrix} d_3(2.x_n y_n) + d_4(r^2 + 2.x_n{}^2) \\ d_3\left(r^2 + 2.y_n{}^2 +\right) + d_4(2.x_n y_n) \end{pmatrix} \tag{3.14}$$

The relationship between x_d, y_d and x_n, x_n finally reads

$$\begin{pmatrix} x_d \\ y_d \end{pmatrix} = (1 + d_1 r^2 + d_2 r^4) \begin{pmatrix} x_n \\ y_n \end{pmatrix} + \begin{pmatrix} d_3(2.x_n y_n) + d_4(r^2 + 2.x_n{}^2) \\ d_3\left(r^2 + 2.y_n{}^2 +\right) + d_4(2.x_n y_n) \end{pmatrix} \tag{3.15}$$

3.2. Artificial Neural Network based nonlinear approximation [13]

3.2.1. The Multilayer Perceptron

The most common neural network found in applications today is the feedforward multilayer perceptron (MLP), also known simply as the multilayer feedforward neural network. It is a collection of artificial neuronsin which the output of one neuron is passed to the input of another. Theneurons (or nodes) are typically arranged in collections called layers, suchthat the output of all the nodes in a given layer are passed to the inputs ofthe nodes in the next layer.

An input layer is the input vector x, while the output layer is the connectionbetween the neural network and the rest of the world. A hiddenlayer is a collection of nodes which lie between the input and output layersas shown in fig.3.2.

Figure 3.2 shows an MLP with:

- 'n' input neurons,
- 'q' hidden neurons: activations functions $\{\psi_j\}$, weights $\{w_{ij}\}$, biases $\{b_j\}$,
- *One* output neuron: linear activation function, weights $\{c_j\}$, bias d,

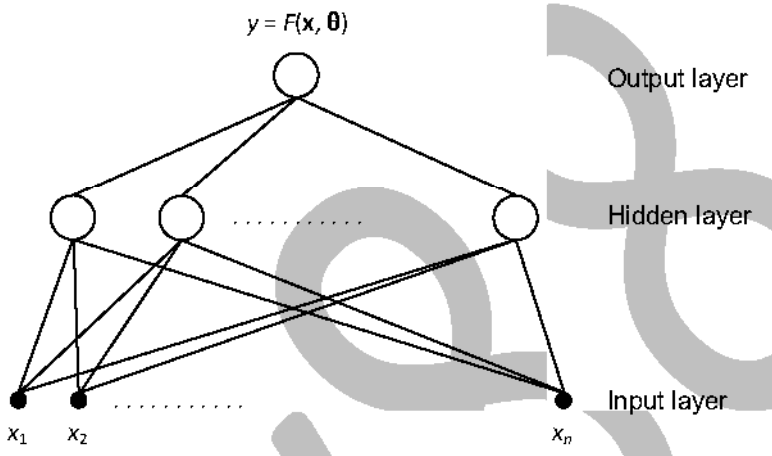


Figure 3.2.Multilayer perceptron

In a fully connected MLP, each neuron output is fed to each neuron inputwithin the next 1ayer. If we let $\boldsymbol{\theta}$ be a vector of all the adjustable parametersin the network (weights and biases, and sometimes the parameters of theactivation functions), then we denote the input-ouptut mapping of a MLP(with one hidden layer) by:

$$F(\mathbf{x}, \boldsymbol{\theta}) = \sum_{j=1}^{q} c_j \psi_j \left(\sum_{i=1}^{n} \omega_{ij} x_i + b_j \right) + d \qquad (3.16)$$

Where

$$\boldsymbol{\theta} = [d, c_1, \ldots, c_q, b_1, \ldots, b_q, w_{11}, \ldots, w_{nq}]^T$$

Within a multilayer perceptron, if there are many layers and many nodesin each layer, there will be a large number of adjustable parameters (e.g.,weights and biases). MLP's with several hundred or thousands of adjustableweights are common in complex real-world applications.

3.2.2. Radial Basis Neural Network

A radial basis neural network (RBNN) is typically comprised of a layerof radial basis activation functions with an associated Euclidean input mapping(but there are many ways to define this class of neural networks). Theoutput is then taken as a linear activation function with an inner productor weighted average input mapping. A RBNN with two inputs and 4 nodesis shown in Fig.3.3.

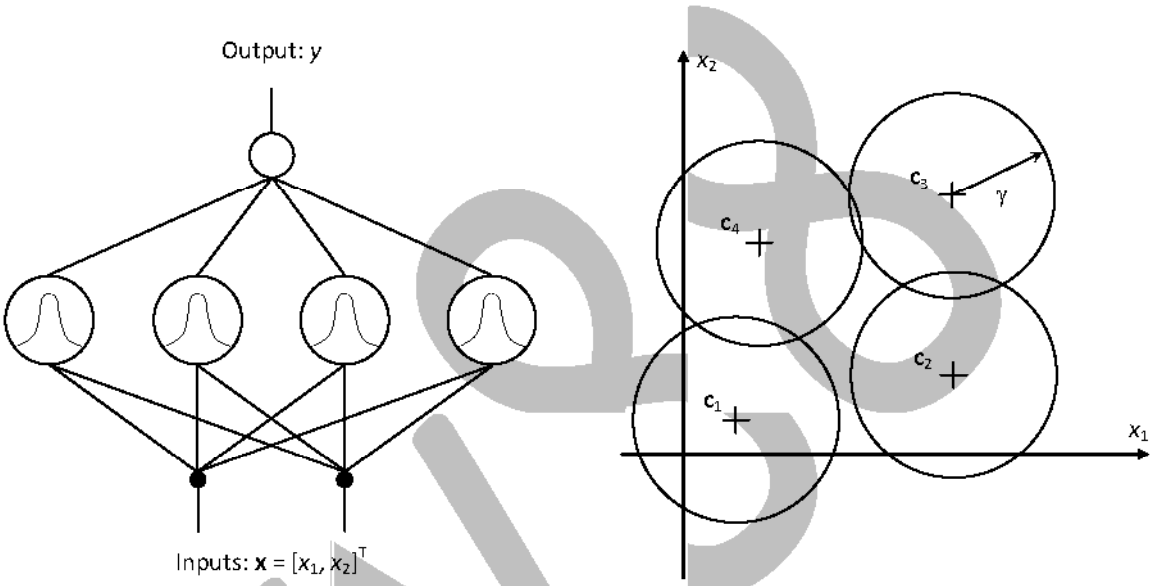


Figure 3.3.RBNN with 4 nodes

The input-output relationship in a RBNN with $x = [x_l, \ldots, x_n]^T$ as an input is given by:

$$F(\mathbf{x}, \boldsymbol{\theta}) = \sum_{j=1}^{q} \omega_j \exp\left(-\left|\mathbf{x} - \mathbf{c}_j\right|^2 / \gamma_j^2\right) \qquad (3.17)$$

$$= \boldsymbol{\theta}^T \zeta(\mathbf{x})$$

Where

$\boldsymbol{\theta} = [\omega_1, \ldots, \omega_p]^T$: the weights vector,

$\mathbf{c}_j = [c_{j1}, \ldots, c_{jn}]^T$: j^{th} hidden layer gaussian function center

Typically, the values of the vectors $\{\mathbf{c}_j\}$andthe scalar γ are held fixed, while the values of $\boldsymbol{\theta}$ are adjusted so that themapping produced by the RBNN matches some desired mapping. [19].

3.2.3. Gradient Optimization

Consider the situation in which it is desired to cause an approximator $F(\mathbf{x}, \boldsymbol{\theta})$ to match a given function. For the ANN, it is a *Learning problem* (how to adjust in time the parameters $\boldsymbol{\theta}$ so that the matching error converges to an acceptable minimal limit)

i. Single training data pair

It is desired to cause an approximator $F(\mathbf{x}, \boldsymbol{\theta})$ to match the function at only a single point $\mathbf{x}^1$ where $y^1 = f(\mathbf{x}^1)$. How to adjust $\boldsymbol{\theta}$ so that the difference,

$$e = y^1 - F(\mathbf{x}^1, \boldsymbol{\theta}) \tag{3.18}$$

is reduced?

In terms of an optimization problem, we want to minimize the cost function:

$$J(\boldsymbol{\theta}) = \frac{1}{2} e^{\mathrm{T}} e \tag{3.19}$$

Taking infinitesimal steps along the negative gradient of $J(\boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$ will ensure that $J(\boldsymbol{\theta})$ is nonincreasing. That is, choose

$$\dot{\boldsymbol{\theta}} = -\eta \left. \frac{\partial J(z)}{\partial z} \right|_{z=\boldsymbol{\theta}}^{\mathrm{T}} \tag{3.20}$$

Where, ηpositive constant, $\boldsymbol{\theta} = [\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_p]$, and $\frac{\partial J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \left[\frac{\partial J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_1}, \ldots, \frac{\partial J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}_p} \right]$

The proof,

$$\frac{dJ(\boldsymbol{\theta}(t))}{dt} = \left. \frac{\partial J(z)}{\partial z} \right|_{z=\boldsymbol{\theta}} \cdot \frac{d\boldsymbol{\theta}}{dt} = \left. \frac{\partial J(z)}{\partial z} \right|_{z=\boldsymbol{\theta}} \cdot \dot{\boldsymbol{\theta}}$$

$$= -\eta \left| \frac{\partial J}{\partial \boldsymbol{\theta}} \right|^2 < 0 \tag{3.21}$$

How to adjust $\boldsymbol{\theta}$ *?*

$$\dot{\boldsymbol{\theta}} = -\eta \left. \frac{\partial J(z)}{\partial z} \right|_{z=\boldsymbol{\theta}}^{\mathrm{T}} = -\frac{\eta}{2} \cdot \frac{\partial e^{\mathrm{T}} e}{\partial \boldsymbol{\theta}}$$

$$= -\frac{\eta}{2}\cdot\frac{\partial}{\partial\boldsymbol{\theta}}\left(y^1 - F(\mathbf{x}^1,\boldsymbol{\theta})\right)^{\mathrm{T}}\left(y^1 - F(\mathbf{x}^1,\boldsymbol{\theta})\right)$$

$$= -\frac{\eta}{2}\cdot\frac{\partial}{\partial\boldsymbol{\theta}}\left(y^{1\mathrm{T}}y^1 - 2F(\mathbf{x}^1,\boldsymbol{\theta})^{\mathrm{T}}y^1 + F(\mathbf{x}^1,\boldsymbol{\theta})^{\mathrm{T}}F(\mathbf{x}^1,\boldsymbol{\theta})\right)$$

$$= -\eta\left(-\frac{\partial F(\mathbf{x}^1,\boldsymbol{\theta})^{\mathrm{T}}}{\partial\boldsymbol{\theta}}y^1 + \frac{\partial F(\mathbf{x}^1,\boldsymbol{\theta})^{\mathrm{T}}}{\partial\boldsymbol{\theta}}F(\mathbf{x}^1,\boldsymbol{\theta})\right)$$

$$= \eta\left.\frac{\partial F(\mathbf{x}^1,z)}{\partial z}\right|_{z=\boldsymbol{\theta}}^{\mathrm{T}}\cdot\left(y^1 - F(\mathbf{x}^1,\boldsymbol{\theta})\right)$$

then,

$$\dot{\boldsymbol{\theta}} = \eta\zeta\left(\mathbf{x}^1,\boldsymbol{\theta}\right)e \tag{3.22}$$

where

$$\zeta\left(\mathbf{x}^1,\boldsymbol{\theta}\right) = \left.\frac{\partial F(\mathbf{x}^1,\boldsymbol{\theta})}{\partial z}\right|_{z=\boldsymbol{\theta}}^{\mathrm{T}} \tag{3.23}$$

ii. Multiple training data pairs

Now consider the problem when M input-output pairs, or patterns, $(\mathbf{x}^i, y^i)$ where $y^i = f(\mathbf{x}^i)|_{i=1,\ldots,M}$ are to be matched. In this case, we let

$$e^i = y^i - F(\mathbf{x}^i, \boldsymbol{\theta}) \tag{3.24}$$

and let the cost function

$$J(\boldsymbol{\theta}) = \sum_{i=1}^{M} e^{i\mathrm{T}}e^i \tag{3.25}$$

Using an approach similar to the single input-output pair case, we can show that the gradient update law is defined by

$$\dot{\boldsymbol{\theta}} = \eta\sum_{i=1}^{M}\zeta^i e^i \tag{3.26}$$

where

$$\zeta^i = \left.\frac{\partial F(\mathbf{x}^i,z)}{\partial z}\right|_{z=\boldsymbol{\theta}}^{\mathrm{T}} \tag{3.27}$$

This update law will adjust the approximator parameters such that $J(\boldsymbol{\theta})$ does not increase over time:

The cost function for multiple data pairs in (4.30) is typically moreuseful in off-line training than the one for a single data pair since it maybe used to cause an approximator $F(\mathbf{x},\boldsymbol{\theta})$ to approximate the continuousfunction $f(x)$ over $D\subset\Re^n$. In other words, it may be possible tochoose some $\boldsymbol{\theta}$ such that

$$\sup_{\mathbf{x}\in D}\left|f(\mathbf{x})-F(\mathbf{x},\boldsymbol{\theta})\right|<\varepsilon \tag{3.28}$$

Discretizedversion (programmable formula)

$$\boldsymbol{\theta}(k+1)=\boldsymbol{\theta}(k)+\eta\sum_{i=1}^{M}\zeta^i(k)e^i(k) \tag{3.40}$$

This is often referred to as a gradient update law or batchback propagation in the neural network community.[13]

The problem, and basic solution design

4.1. Problem definition

The objective of this work is to provide an Artificial Intelligence solution that enablesa mobile robot to perceive its environment for a given global system to be able to perform practical tasks autonomously in anindoor environment for example. Here an embedded camera system primarily employed for other tasks is exploited in the perception problem.

Given an input image-frame taken from the robot environment, it is asked to perceive its location, which allows to identify (sign and value) its status of approaching or leaving this reference position.

4.2. Basic solution Design

Our contribution is based upon the following considerations:

- The reference point is chosen to be a typical printed mark on the facing plane,
- The robot is supposed to act (move) perpendicularly (special case) to this reference plane, or with a non-zero angle with the perpendicular axis,
- The system (Camera - Digital Computer) is employed to:
  - Acquire an instantaneous reference mark image,*a circle*.
  - Identify the robot position/velocity with respect to the reference coordinates, by means of image processing.

4.2.1. Off-line step

The objective of this step is to find a localization model; we applied for this purpose the *camera calibration algorithm* with a curve *fitting process*. With the camera calibration, intrinsic and extrinsic parameters are computed, while the relationship between the image and the robot position is defined in the fitting step, this one is tried by *exponential interpolation* and an RBF based nonlinear approximator. The Hough transform is considered in the modeling process.

4.2.1. On-line activity

Here, the position and velocity are estimated in real time. The mobile robot, mechanical plant, is supposed to be relatively slow. The frame acquisition rate is considered to be $0.5Hz$ (one image every 2seconds). Every sample time, the radius and center coordinates are computed by the HT, the fitting phase estimates the actual position with respect to the reference mark.

4.3. Identifying positions

The experience shows how to use the Hough Transform as a method for detecting objects defining their contours in gray tone (color) images. From a given family of curves being sought, this method produces the set of curves from the family that appear on the image.Furthermore, this experience shows how to calibrate the camera and how to use it to measure the distance from the recognized object (circle).

The experience is repeated for many different distances in order to determine how a given feature of the seen object changes according to the distance variation. Therefore, any distance can be predicted from the measured feature of a given object.

In curve fitting we have raw data and a function with unknown coefficients. We want to find the best values for the coefficients such that the function matches the raw data as well as possible,

4.3.1. Preliminary tests

Detecting lines. The HT of a pentagon

The Hough peaks are the locations of peaks in the Hough transform matrix, H.

The number of groups of peaks refers to the number of lines in the image; we have five groups so we have five lines.

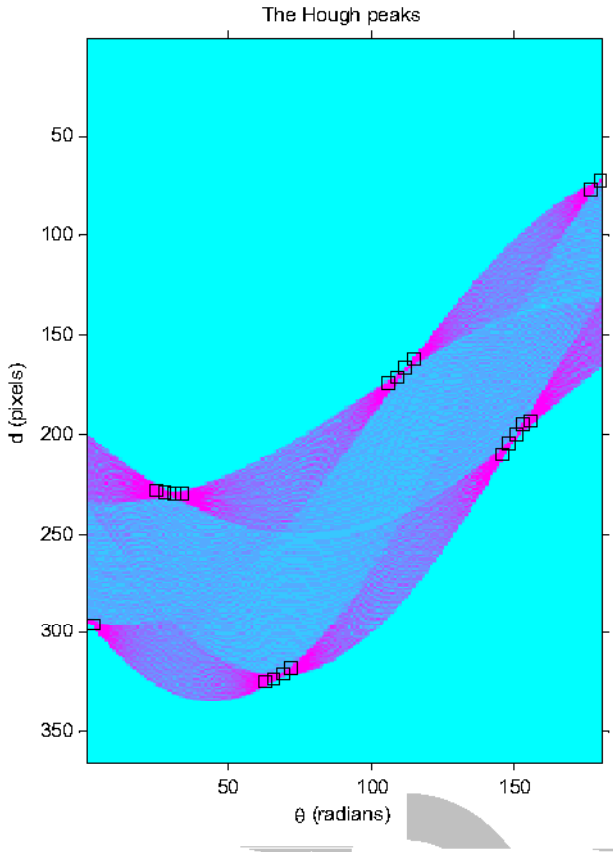


Figure 4.1. Hough peaks for a pentagon

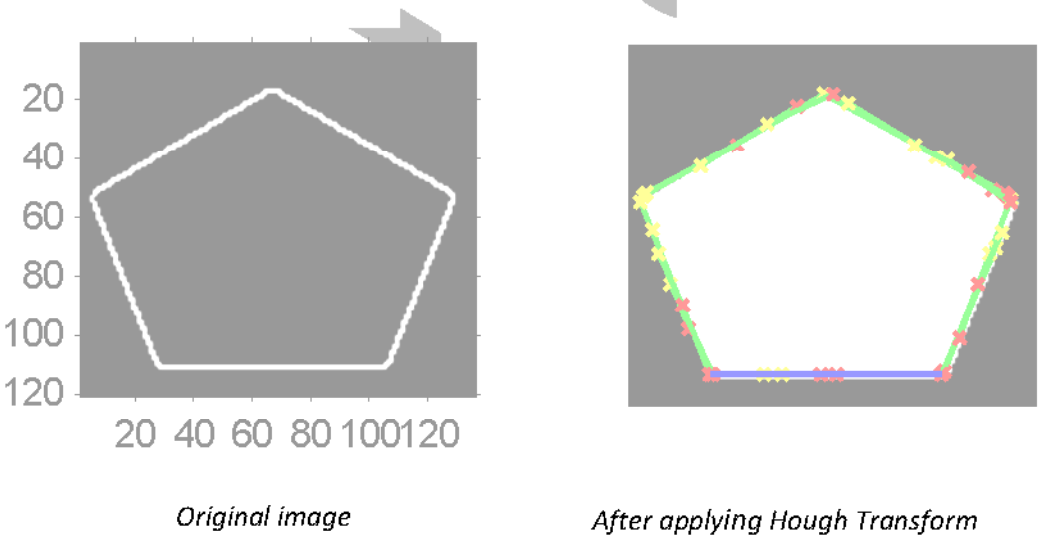


Figure 4.2. *Lines detected by the Hough Transform*

Using the Hough transform algorithm for lines, we can extract line segments in the image. When we find the peaks in the Hough transform matrix H; we can determine theta and rho vectors thatcontains the row and column coordinates of the Hough transform binsto be used in searching line segments.The Hough transform algorithm for linesreturns lines,a structure

array whose length equals the number of merged line segmentsfound. It extracts all lines of the polygon in the image, and the number of lines is equal to the number of peaks.

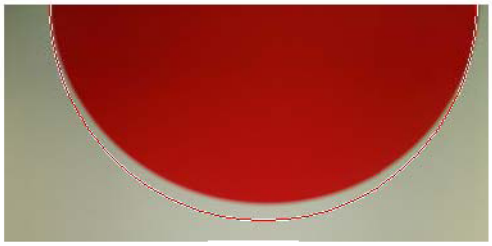
Detecting circles

A circle image is taken for a given distance; this experience is repeated for 9distance cases.Detection of contours by The Hough Transform algorithms is shown in fig4.3 The experimental (distance *vs* radius) is shown in table 4.1

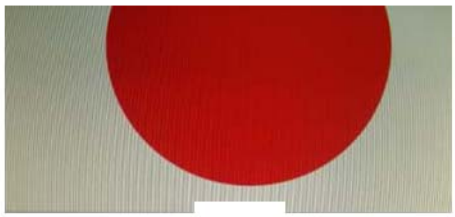
Table 4 1*. Center coordinates and radius of circles detected by the Hough Transform versus experimentally measured distance*

Distance: d *(cm)*	**Radius** *(pixels)*	**Center: (u,v)** *(pixels)*
20	449.00	(546.19,7.01)
30	303.57	(592.85,89.04)
40	221.92	(657.58,176.24)
50	180.40	(611.18,208.75)
60	148.92	(607.11,235.48)
70	127.75	(593.78,277.24)
80	111.85	(580.45,283.19)
90	99.82	(553.33,311.949)
100	89.94	(556.14,306.63)

d=20cm

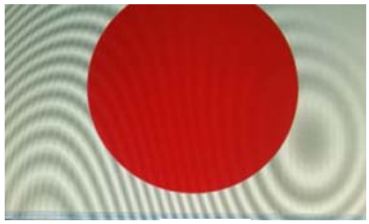


a1 a2

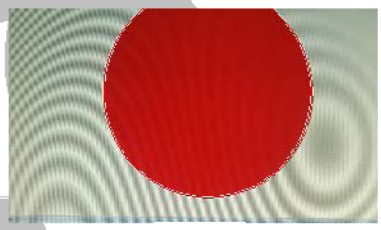


d=30cm

b1 b2

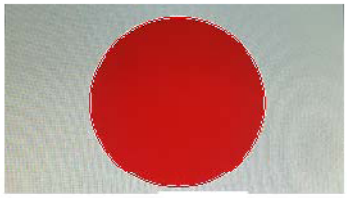


d=40cm



c1 c2

d=50cm



d1 d2

d=60cm

e1 e2

d=70cm

f1 f2

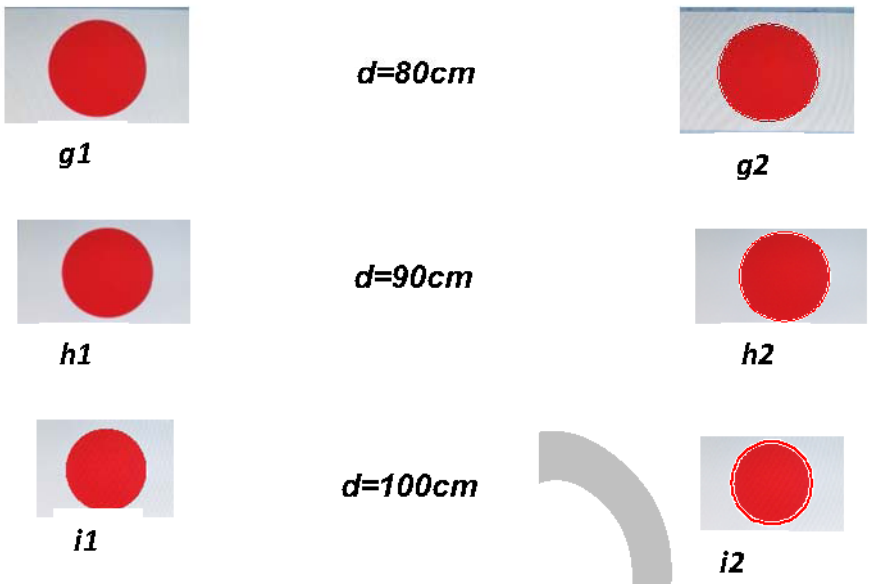


***Figure 4.3**. original images on the left and their corresponding edge detection on the right*

Two main features can be deduced from the circle contour obtained, the radius and the center coordinates. The radius of the circle decreaseswhen the distanceis increasing.

4.3.2. Camera calibration

Camera calibration is the process of estimating the parameters of the lens and the image sensor. These parameters are needed to measure objects captured by the camera; we first need to take multiple images of a calibration pattern from different angles.

A typical calibration pattern is an asymmetric checkerboard, where one side contains an even number of squares, both black and white, and the other contains an odd number of squares. The pattern must be affixed to a flat surface, and it should be at approximately the same distance from the camera as the objects we want to measure.

Once we calibrate the camera and estimate its parameters (intrinsic and extrinsic parameters), we use them to compute the transformation matrix from world to image and then apply the inverse transformation from image to world to find the experimental distance.

Finally, we obtain the mathematical model that allows finding distance (a real world measure) from a reference image (image measure). This model is a fitting curve $R = f(D)$.

Experimentally, the system (robot-camera-computer) moves towards the circle-printed mark and computes the distanceto this reference by using the fitting process found with the camera calibration procedure.

The camera calibration experiments

As known, the camera calibration is a procedure that highly depends on the type of the used camera. In our case, we have used a canon camera with features PowerShot A[illegible]32 MA PIXELS. The calibration of this latter is carried out on a checkerboard and an image with a shape that is considered as shown by the figure 4.4.

Seven photos of the pattern (checkerboard) are taken for each distance, and we use two separate images: one containing the pattern, and the other containing the reference mark. Furthermore, the objects and the pattern must be in the same plane and the images must be captured from exactly the same view point. A test image sample is given in fig. 4.4.

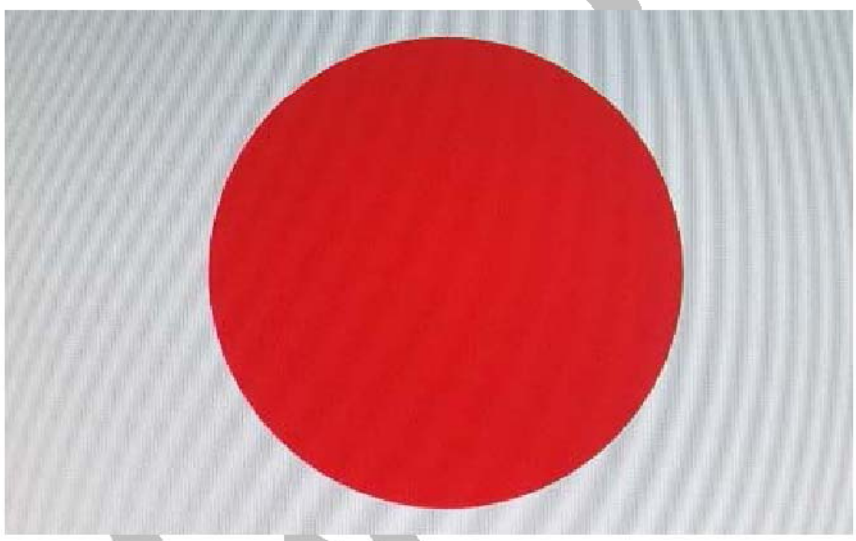


Figure 4.4-a. image captured at [illegible]cm

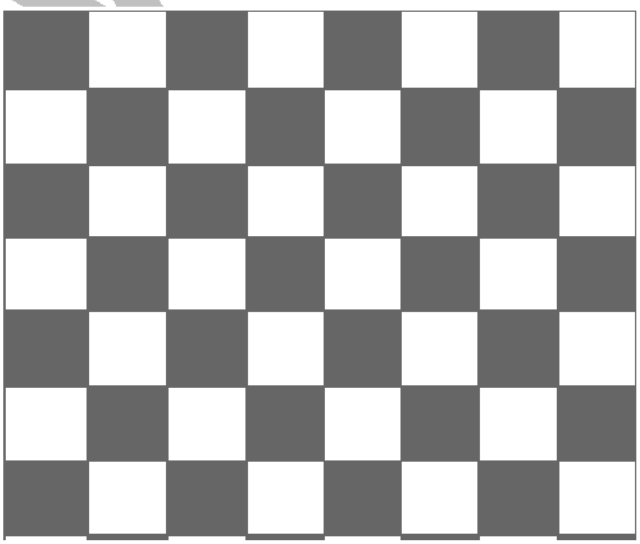


Figure 4.4-b. checkerboard captured at [illegible]cm

Figure 4.4-c. checkerboard samples for the camera calibration step

The Intrinsic Parameters Matrix is:

$$K = \begin{bmatrix} 2.4 & 0 & 1.3 \\ 0 & 2.9 & 0.7 \\ 0 & 0 & 1 \end{bmatrix}$$

The extrinsic parameters:

Rotation matrix:

$$R = \begin{bmatrix} 0.0 & -0.9 & -0.0 \\ 0.9 & 0.0 & -0.2 \\ 0.2 & 0.0 & 0.9 \end{bmatrix}$$

Translation vector:

$$t = \begin{bmatrix} -0 & .5 \\ 6 & .0 \\ 8 & .2 \end{bmatrix}$$

Using these three parameters we can find the co ordinates of any point in the real world.

4.3.3.The fitting step results

The fitting step results are shown in fig.4.6 for exponential interpolation model type; and fig.4.7 for a (1input,4hidden neurons, 1output) RBF model type found using 21 samples.The table 4.2 summarizes the computed ra dius of the circle taken by the camera at a real distance $\mathbf{d_r}$, the distance $\mathbf{d_{ex}}$ deduced from the camera calibration and $\mathbf{d_c}$ computed by curve fitting method.The three curves (fitted, real and experimental) are acceptably close each to the other.

The considered RBF neural network(1input, 9hidden neurons, 1output)with $\gamma = 80$, $\eta = 0.01$, and the centers evenly spaced in the practical distance domain, is trained for 8iterations to reach the following parameter setting:$\theta = [10.168 \ -10.831 \ 42.953 \ 22.382 \ 30.999 \ 44.325 \ 81.901 \ 117.474 \ 457.272]$

This fitter will be then inserted inside the whole system as shown in fig.4.5

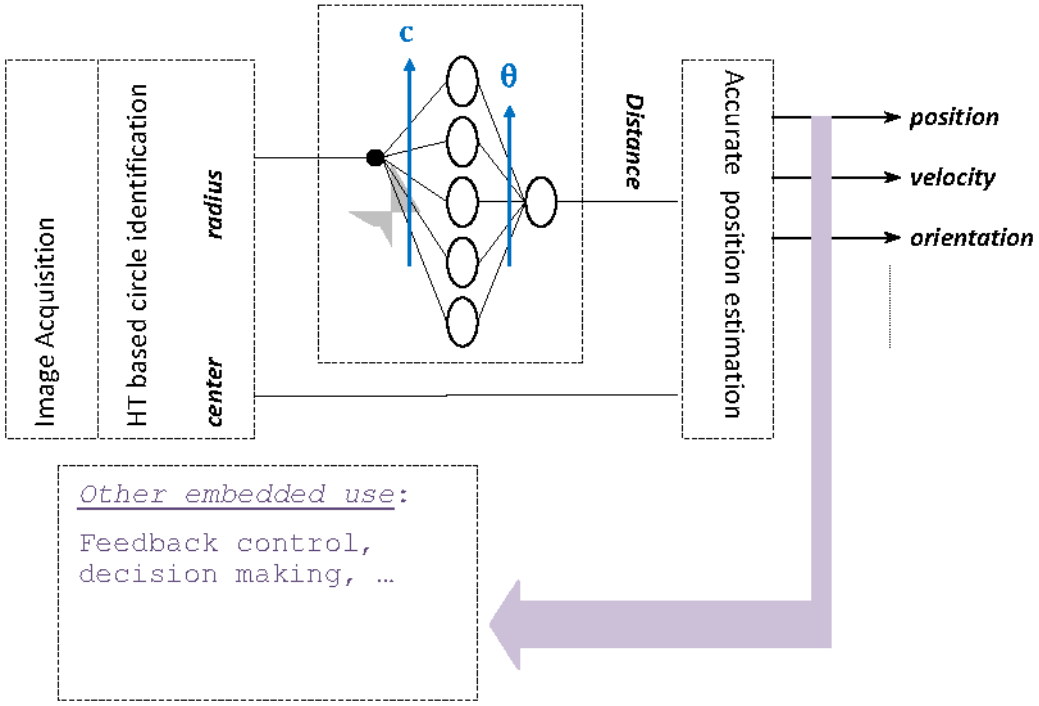
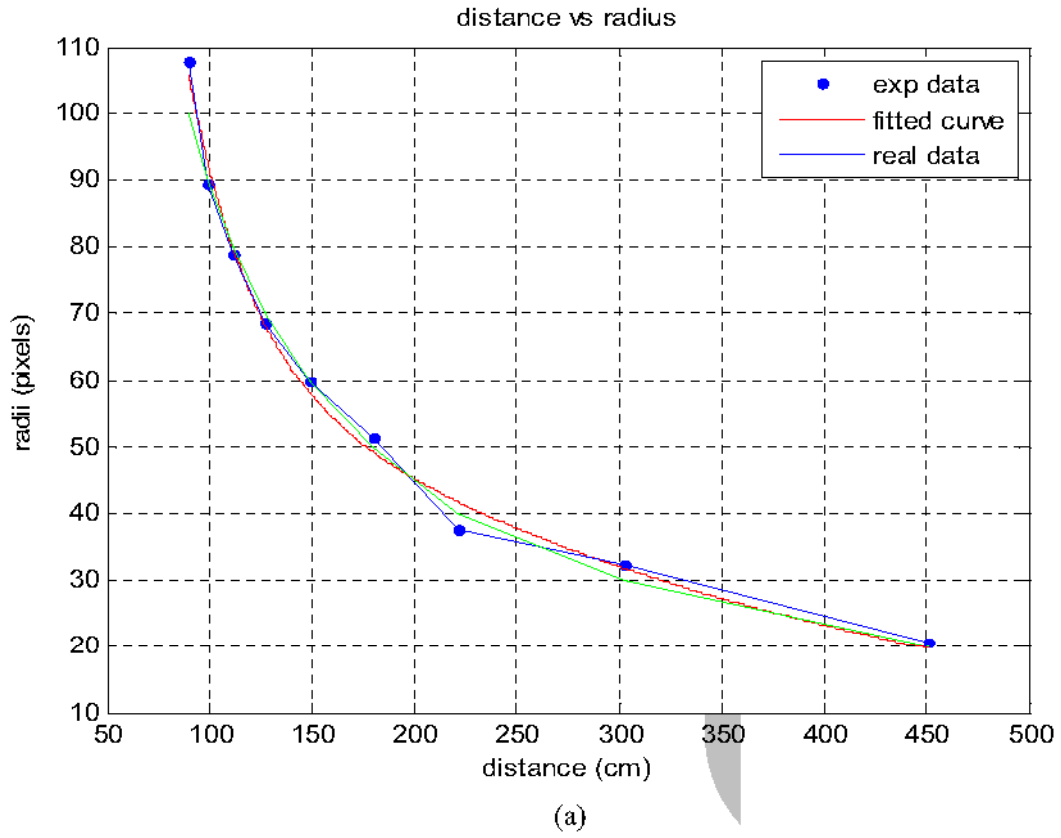


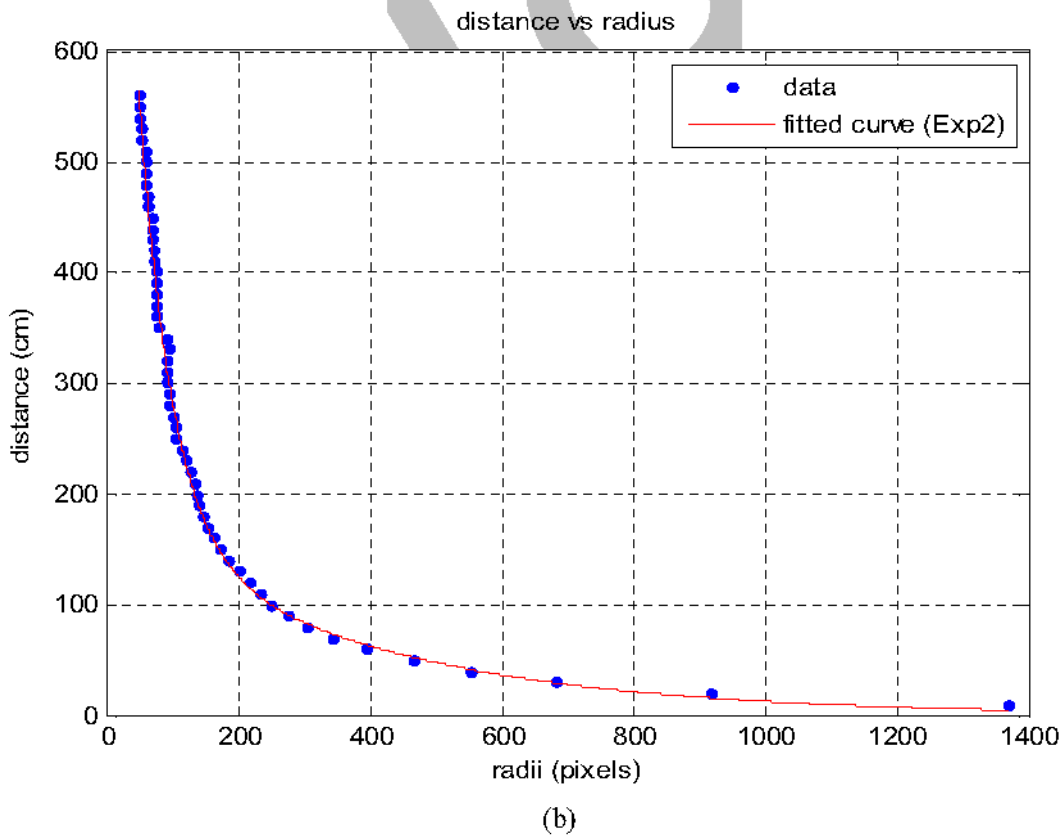
Figure 4.5. *The RBF neural network inside the perception system*

Table 4.2. computed radius of the circle vs distance

Real Distance d_r (cm)	**Computed Radius** (pixels)	**Distance found using the camera, d_{exp}**(cm)	**Distance computed by curve fitting method, d_c**(cm)
20	90	02	26
30	67	32	30
40	52	35	47
50	40	54	45
60	42	68	59
70	37	66	66
80	28	89	85
90	92	92	96
100	90	07	94



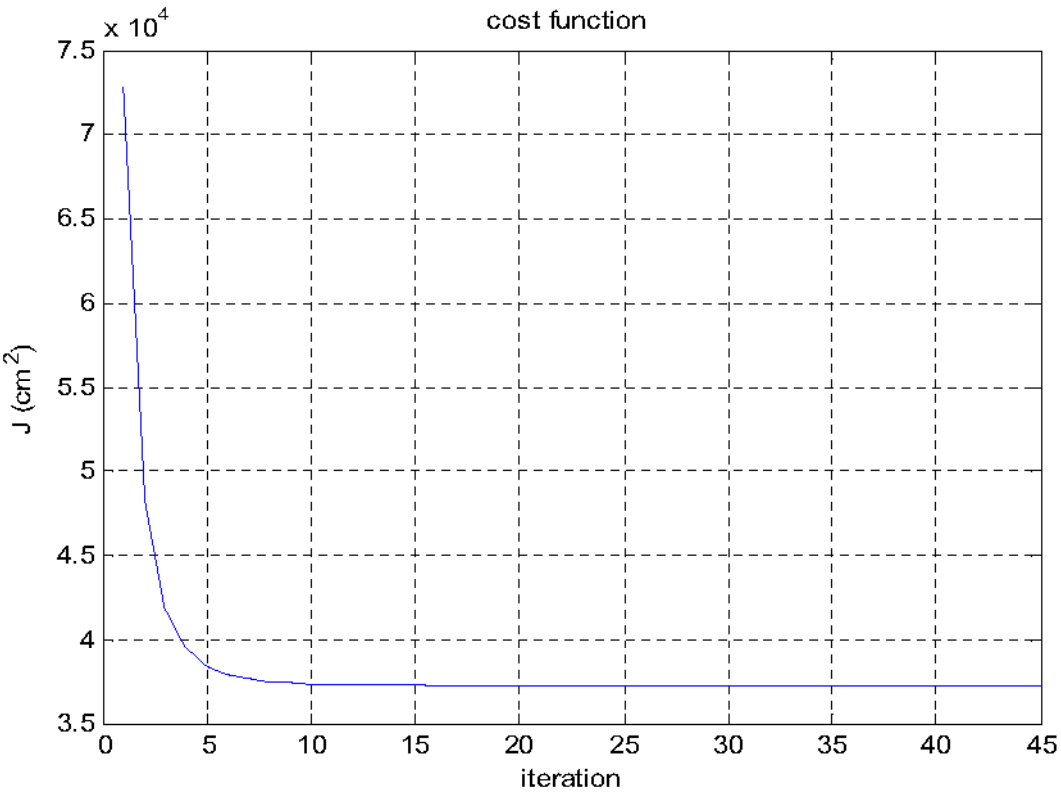
(a)



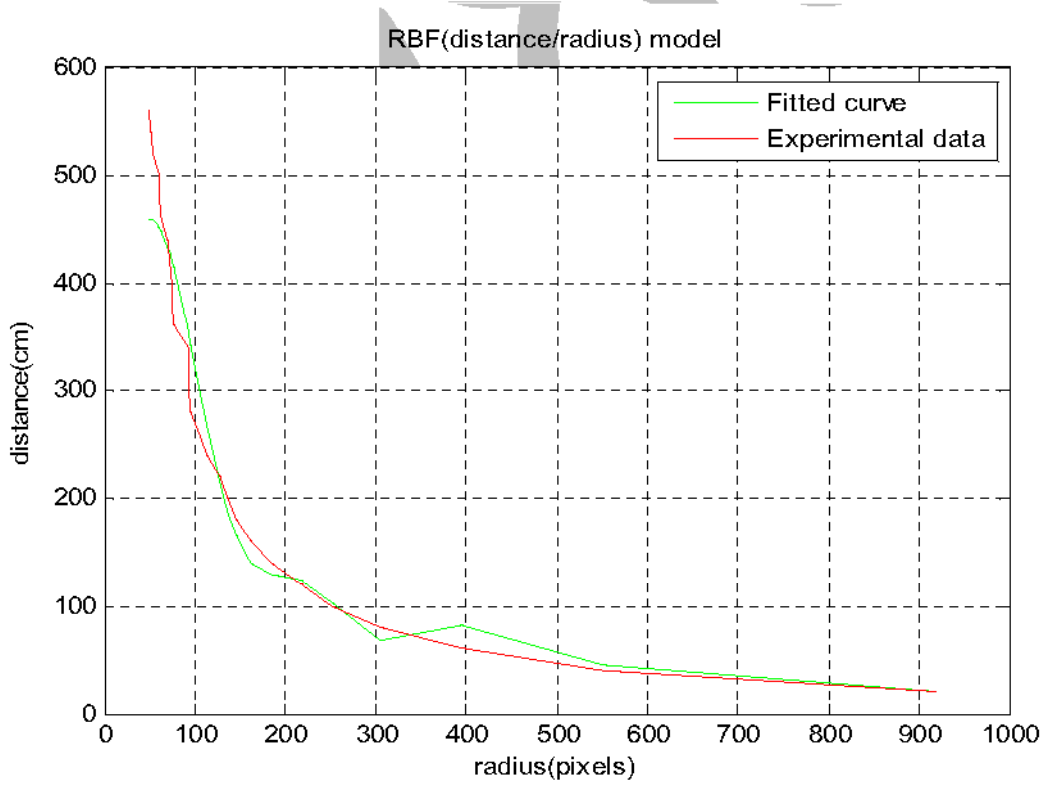
(b)

Figure 4.6. *experimental and exponential interpolation fitted relationship (distance-radius)*

(a) For 10 samples (b) For 56 samples



(a)



(b)

Figure 4.7. *image based measured distance model*
(a) Costfunction (b) RBF model

With this model implemented, the robot has in real time its position while the reference image is acquired; the distance is measured with sufficient accuracy.

The velocity of an object is defined as the rate of change of position with time.

Given the position of an object, we can estimate its velocity as:

$$v(t) \approx \frac{x(t+\Delta t)-x(t)}{\Delta t} \quad (4$$

Where Δt is the time increment. This expression is called the right derivative approximation for the velocity. When Δt is sufficiently small, we should obtain an accurate value for the velocity.

We assume that the robot is moving with constant velocity.

1) If $\Delta d = constant \neq 0$

   - $\Delta d = V_{rob} \cdot T \Rightarrow$static obstacle, approaching or going away depends on the sign of the robot velocity.
   - $\Delta d \neq V_{rob} \cdot T \Rightarrow$dynamic obstacle.

     -If $\Delta d > V_{rob} \cdot T \Rightarrow$ an approaching case.

     -If $\Delta d < V_{rob} \cdot T \Rightarrow$ approaching case.

2) If $\Delta d = 0 \Rightarrow$ the obstacle and the robot are moving in the same sense with the same velocity

The sign of velocity and its rate allow to identify if the robot is approaching or going far from the reference mark, at a given rate. Additional geometric computation is simple to insert when dealing with a non-zero angle direction.

Conclusion

Because of the ability of the robots to deal with a human-like thinking, high-level skills, tasks, and scenarios, involving the intelligent computer vision tools in the decision making schemeare promising ways to solve complex industrial problems.. A great part of research in thisfield is interested in software development and improvement. The autonomy of a mobile robot is highly related to how is its working environment perceived. That is why in the mobile robots research field, the perception are of great interest.

In this project, we designed a visual perception based method of localization for a mobile robot. In the off-line step, we find a model that describes the relationship between the position information and an image type information. It is a hybrid process that uses the camera calibration and the backpropagation algorithm based RBFs learning. This model will allow the robot to identify, in real time, its instantaneous position and velocity with respect to a reference point, by exploiting the Hough Transform. We can include many other measures upon this basic computing like the *approaching-to-obstacle* process quantification (sign and rate).

The Hough Transform is time consuming; the given problem is solvable for low velocities given the sufficient operating frequency of the existing hardware (computer, camera). It will be interesting in a future work to improve the Hough computing (time execution) or to try an other segmentation method with additional design considerations, or to try this software using a more sophisticated hardware (professional camera, high speed computer).

Experimental results show the applicability of the designed system under some practical constraints, an other future work that deals with an intelligent mark extraction from a noisy image (when the robot is far) may reduce this limitations.The designed software allows a real time accurate position/velocity measuring in terms of dimensions of the shape of the selected image.

Abstract

In this work, we designed a visual perception based method of localization for a mobile robot. In the off-line step, we find a *model* that describes the relationship between the *position* information and an *image* type information. It is a hybrid process that uses the *camera calibration* and the *backpropagation* algorithm *based RBFs learning*. This model will allow the robot to identify, in real time, its instantaneous position and velocity with respect to a reference point, by exploiting the *Hough Transform*. The method may be applied with acceptable efficiency for low speed robots.

Résumé

Dans ce travail, nous avons conçu une méthode de localisation par perception visuelle pour robot mobile. Hors ligne, il s'agit de trouver un *modèle* décrivant l'information *position* en fonction d'une autre information du type *image*. C'est un processus parallèle qui fait appel aux techniques de *calibrage de camera* et *apprentissage de RBFs* par une version de l'algorithme de *rétropropagation*. Le modèle validé permettra au robot mobile d'identifier sa position, parapport à un point de référence, et vitesse instantanée en temps réel par détermination et exploitation de la *Transformée de Hough*. La méthode est applicable pour des robots à faible vitesse.

Bibliography

1. Souma Alhaj Ali; Masoud Ghaffari, Xiaoqun Liao and Ernest Hall, "Mobile Robotics, Moving Intelligence", in Mobile Robots, Moving Intelligence, ISBN: 3-86611-284-X, Edited by Jonas Buchli, pp. 576, ARS/plV, Germany, December 2006
2. PETER KUCSERA, Sensors For Mobile Robot Systems; AARMS; August, 2006.
3. Ali KILIÇ, THESIS: Navigation of a Mobile Robot Using Stereo Vision; Mechanical Engineering, University of Gaziantep, January 2010
4. http://en.wikipedia.org/wiki/Laser_rangefinder
5. Francisco Bonin-Font, Alberto Ortiz and Gabriel Oliver, Visual Navigation for Mobile Robots, Department of Mathematics and Computer Science, University of the Balearic Islands, Palma de Mallorca, Spain, 2005
6. Linda Shapiro and George Stockman; Computer Vision; prentice hall, 2001
7. Emanuele Trucco and Alessandro verri, *Introductory Technics for 3D Computer Vision*, prentice hall, 1998
8. Tinku Acharya and Ajoy K Ray; IMAGE PROCESSING, Principles and Applications; WILEY INTERSCIENCE; 2005
9. http://www.mathworks.com/access/helpdesk/help/toolbox/images/
10. Pedram Azad, Tilo Gockel and Rüdiger Dillmann, *Computer Vision, Principles and Practice*, elector, April 2008
11. http://www.Chegg.com/homework-help/physics- for scientists-and-engineers-8th-edition-solution_9780495827818
12. Pedram Azad, Dissertation: *Visual Perception for Manipulation and Imitation in Humanoid Robots*, Fakultät für Informatik; Universität Karlsruhe (TH), Dezember 2008.
13. Jeffrey T. Spooner, Manfredi Maggiore, Raúl Ordóñez and Kevin M. Passino, *Stable Adaptive Control and Estimation for Nonlinear Systems. –Neural and Fuzzy Approximator Techniques*, New York: John Wiley and sons, Inc., 2002
14. Emerging Topics in *Computer Vision*, Gerard Medioni and Sing Bing Kang, prentice hall, 2004
15. Alain Pruski, *Robotique Générale*, ellipses, 1988
16. Teukolsky, and W.T. Vetterling, *Numerical Recipes in C*, 2nd ed., 994 pp., Cambridge University Press, New York, 1992
17. Introduction to Autonomous Mobile Robots, http://www.mobilerobots.org
18. Cyril DROCOURT, Thèse: Localisation Et Modélisation De L'environnement D'un Robot Mobile Par Coopération De Deux Capteurs Omnidirectionnels, Université de Technologie de Compiègne, Centre de Robotique, d'Electrotechnique et d'Automatique, février 2002
19. Appin Knowledge solutions, *ROBOTICS*, INFINITY SCIENCE Press, 2007