ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA

CAMPUS DI CESENA
SCUOLA DI INGEGNERIA E ARCHITETTURA

Corso di Laurea in Ingegneria e Scienze Informatiche Magistrale

MOBILE ROBOTS AND VISUAL SENSORS

Corso di: Sistemi Autonomi

Elaborato di:
MICHELE ROBERTI

Docente:
Prof. ANDREA OMICINI

ANNO ACCADEMICO 2017-2018

Contents

Abstract

This survey provides an overview of the use of visual sensors in robots navigation. Vision is becoming more and more common in applications such as localization, automatic map construction, autonomous navigation, path following, inspection, monitoring or risky situation detection. Two major components of this paper are indoor navigation and outdoor navigation. Each component is subdivided in more specific subsections. For indoor navigation we discuss about navigation based on Map-Based approaches, on Map-Building approaches and on Mapless approaches. In the latter one are included navigation based on optical flow, on feature tracking and appearance-based methods. About outdoor navigation we deal with the navigation in structured and unstructured environments.

Chapter 1

Introduction

Mobile robot vision-based navigation is the source of countless research contributions, from the domains of both computer vision and control theory. Progress has gone far in visual navigation techniques for land, aerial and autonomous underwater vehicles. In this survey we present several techniques which can be described according to two orthogonal dimensions: indoor vs outdoor and map-based vs mapless navigation. By definition, navigation can be described *as the process of determining a suitable and safe path between a starting and a goal point for a robot travelling between them* [1]. Traditionally, vision-based navigation solutions have mostly been devised for autonomous ground vehicles (AGV), but, recently, visual navigation is gaining more and more popularity among researchers developing unmanned aerial vehicles (UAV). UAVs offer great perspectives in many applications, such as surveillance, patrolling, search and rescue, outdoor and indoor building inspection, real-time monitoring, high risk aerial missions, mapping, fire detection or cinema recording. Besides, the typically small size of UAVs limits their payload capabilities so that they cannot carry the same sensors ground vehicles can, such as lasers or certain brands of sonars because of their weight and volume. For underwater environments, more traditional solutions (e.g. acoustic-based) are still the preferred ones because of the special characteristics of light propagation undersea. Dalgleish et al. [2] state that sonar based systems are limited in resolution and size. Such limitations are imposed by the frequency band usually employed by sonars and by their need for accommodation space. Vision systems reduce space and cost and increase the resolution of the images, although their

range dramatically decreases in muddy or turbid waters.
In the remainder of this document we provide an overview about the main approaches to localization and mapping employing vision systems.

Chapter 2

Indoor Navigation

Since the very beginning of robotic vehicle navigation it became clear that it is mandatory for a vision system conceived for navigation to incorporate some knowledge about what the computer is supposed to see. Some of the first vision systems for mobile robot navigation heavily relied on the geometry of space and other metrical information for driving the vision processes and performing self-localization. In particular, indoor space is represented by CAD models of varying complexity and later in the following works [3] they were replaced by simpler models, such as occupancy maps, topological maps or even sequences of images.

Depending on the avaliablity and dinamicity of a map for the robot, navigation approaches can be cathegorized by the following three groups:

- **Map-Based Navigation**: Where robots rely on user-created geometric models or topological maps of the environment.

- **Map-Building-Based Navigation**: Where robots rely on the use of sensors to construct their own geometric or topological models of the environment and then use these models for navigation.

- **Mapless Navigation**: Where robots cannot rely on any explicit representation at all about the space in which navigation is to take place, but rather resort to recognizing objects found in the environment or to tracking those objects by generating motions based on visual observations.

2.1 Map-Based Approaches

Map-Based Navigation consists of providing the robot with a model of the environment. Such models may contain different degrees of detail, varying from a complete CAD model of the environment to a simple graph of interconnections or interrelationships between the elements in the environment.

Some of the very first vision systems used to represent the enviromental knowledge by means of a grid where every object in the enviroment was represented by its 2D projection onto the horizontal plane.

Such a representation is usually referred to as *occupancy map* and it was formally introduced in [4]. An improved elaboration consist of incorporating uncertainty in an occupancy map to account for errors in the measurement of the position coordinates associated with the objects in space and in the quantization of those coordinates into "occupied/vacant" cells.

Nowdays the central idea in any map-based navigation is to provide to the robot a sequence of landmarks expected to be found during navigation, the task of the vision system is then to search and identify the landmarks observed in an image. Once they are identified, the robot can use the provided map to estimate the robot's position (self-localization) by matching the observation (image) against the expectation (landmark description in the list of landmarks known by the robot). The computations involved in vision-based localization can be divided into the following four steps:

1. **Acquire sensory information**: in this step features are extracted from images employing state of the art tecniques such as acquiring and digitizing camera images.

2. **Detect landmarks**: this means extracting edges, smoothing, filtering, and segmenting regions on the basis of differences in gray levels, color, depth, or motion.

3. **Establish matches between observation and expectation**:in this step, the system tries to identify the observed landmarks by searching in the database for possible matches according to some measurement criteria.

4. **Calculate position**:once a match (or a set of matches) is obtained, the system needs to calculate its position as a function of the observed

landmarks and their positions in the list of landmarks known by the robot.

As one would expect, the third step is the most difficult. Often, it requires a constraint optimisation problem to be solved, possibly taking into account the prior knowledge about the landmarks and any other information which may reduce the uncertainty about the robot position. Concerning this topic, Atiya and Hager [5] developed a real-time vision-based algorithm for self-localization. They proposed that the sensory error should be coupled with a tolerance value, and a set-based algorithm for solving the matching problem and computing the absolute location of a mobile robot for indoor navigation. The basic idea of their approach is to recognize in the camera image those entities that stay invariant with respect to the position and orientation of the robot. For example, consider a triple of point landmarks on a wall in the environment. If all three of these points could be identified in each image of a stereo pair, then the length of each side of the triangle and the angles between the sides would stay invariant as the robot moves to different positions with respect to these three points. So, the length and the angle attributes associated with a triple of landmark points would be sufficient to identify the triple and to set up correspondences between the landmarks points in the environment and the pixels in the camera images. Once such correspondences are established, finding the absolute position of the robot simply becomes an exercise in triangulation. But, of course, in the real world, the coordinates of the landmark points may not be known exactly. Also, the pixel coordinates of the observed image points may be subject to error. Additionally, given a multiplicity of landmark points on a wall (or walls), there may be ambiguity in establishing correspondences between the landmark triples and the observed pixel triples.

2.2 Map-Building Approaches

The vision-based navigation approaches discussed so far have all required the robot to possess a map (or a model) of the environment. But model descriptions are not always easy to generate, especially if a human operator has to manually provide metrical information.
An interest operator was applied to extract distinctive features in the images. These features were then correlated to generate their 3D coordinates.

All this processing was done remotely and took five hours to traverse 20 meters. The "world" was represented by the 3D coordinates of the features plotted in a grid of two square meter cells. The features were tracked at each iteration of the program and marked in the grid and in the image plane. Although this grid indirectly represented the position of obstacles in the world and it was useful for path planning, it did not provide a meaningful model of the environment.

An important topic in map-builing approaches deal with environments where multiple robot systems are presented. For instance Konolige et al.[6] presented a distribuited approach to mapping and exploration enabled by pair-wise relations between robots. Each pair of robots can have four different types of interactions: none, hypothesis generation, hypothesis verification, and coordination. At each point in time, the state of the multirobot system can be summarized by a graph structure where the nodes are individual robots and edges represent the current interaction between two robots.
Here are presented the different types of interactions:

- **No interaction**: The robots are not within communication range.
- **Hypothesis generation**: The robots can communicate but they do not know their relative locations. In this stage, one of the two robots receives sensor data from the other robot and estimates their relative location using its own map. Which of the two robots adopts the estimation role depends on available computational resources.
- **Hypothesis verification**: Robots can communicate and verify a location hypothesis determined in the hypothesis generation phase. In the case of uncertainty, the robots can try to meet. If the robots don't meet at the expected location, the hypothesis is rejected and they continue with the hypothesis generation phase. Otherwise, the robots can establish their relative positions, combine their maps, and coordinate their exploration efforts.
- **Coordinated exploration**: Once the robots have determined their relative locations, they can share their maps and perform coordinated exploration. A nice feature of such as interaction type is transitivity, i.e. if robot i and j can share their maps, and robot j and k can share their maps, then all three robots can build a combined map.

Hence, robots in this interaction mode form exploration clusters in which they can coordinate their actions. Each cluster determines one robot responsible for data combination and robot coordination. All information is frequently spread throughout the cluster, even if direct communication between all robot pairs within the cluster is not required; nor is a central information repository or mission control.

2.2.1 Map Merging as Decision Problem

When two robots i and j exchange map information, they must determine whether they are in each other's partial maps. We have a decision problem that can be phrased by robot i as:

1. *Am I at location x in robot j's map?*
   Once the decision is made to combine maps, it is difficult to undo, and so it is important that the decision (1) is reliable. On the other hand, if decisions are not made when i is indeed within j's map, then the map construction will be inefficient. So we need a decision method that is both highly specific and highly sensitive.

   This question is similar to a decision problem for localization within a given map:

2. *Am I at location x in the localization map?*
   In Markov localization [7], standard Bayesian techniques are used to find the most likely place for the robot, given that the robot must be somewhere in the map. But Markov localization does not address the decision problem (2), which can be important for planning.

We formulate the decision problem (1) in terms of likelihood over positions of the robot, both in the partial map and in any other possible map that the robot could be in.

Formally:

$$L(x) > \alpha \text{ , and}$$
$$MIN\left[\frac{L(x)}{L(x')}\right] > \beta$$

where x is the map position of the decision, $L(x)$ is the likelihood of being at x, and the domain is the set of possible maps that the robot could

be in. There are two thresholds for the decision: the absolute likelihood has to be large (α is close to 1.0), and the likelihood relative to all other possible places should be high (β larger than 1.0). To understand this decision equation, consider a complex of several buildings, in which the offices are given unique tags. A robot seeing the tag EK288 would have a high likelihood of being at the office EK288, and a very low likelihood of being anywhere else, because the tags are unique; the minimum ratio is very high. Conversely, a robot in front of an office, but not seeing the tag, would have a high likelihood of being in front of any office, and while the absolute likelihood $L(x)$ would be high, the minimum ratio would be close to 1. This example illustrates some key aspects of the decision problem. The problem is easy if we have features with good discrimination power, such as the office tag; if we can add such tags to the environment, we have solved the problem. But, in general we won't have the benefit of unique single features; rather, we will have to judge how likely it is that a set of features the robot encounters uniquely determines its position, i.e., estimate the value of $L(x')$ over all environments, for some current set of features.

The proposal model of map merging is as follow.
An environment consists of a set of range scans registered correctly into a map. Four large maps of office buildings are used. The maps have a total corridor length of approximately 600 meters. A robot constructs a partial map as it explores the environment. The local environment around the robot is called a patch. Figure 2.1 shows a typical office environment, and a set of sample patches. To make the map merging problem more challenging, are used small patches, consisting of 15 scans taken about 0.5 meters apart. These patches are just on the edge of having enough information to enable reliable merging. Longer patches, of course, make the decision problem easier: in the limit, one could use the whole map. There is also the possibility of using sets of patches for a more reliable decision. Features are configurations of scan points. Three basic features are considered: doors, junctions, and corners. These features are marked in the patches at the right in Figure 2.1. The doors here are closed, and are somewhat difficult to distinguish from the hallway wall. A patch contains from 2 to 8 features in the data set used. The features were extracted by hand. Likelihoods are calculated by placing the patch onto an environment at a pose x. The likelihood $L(x)$ is a function of matching the patch range points against the map, and matching patch features against map features. Because it is geometrically based, the

likelihood function automatically takes into account the spatial relationship among features. Patches are classified (by hand) into one of a number of classes. Each class is a population of patches with certain characteristics in common, e.g., corridors with no junctions. The classes considered are:

- 1. Corridor scans (no junctions)
- 2. Corridor scans with junctions
- 3. Single corner scans
- 4. Multiple corner / junction scans
- 5. Corridor scans + door features
- 6. Junction scans + door and junction features
- 7. Single corner scans + door and corner features
- 8. Multiple corner scans + door and corner features

The likelihood function $L(x)$ measures the probability of seeing a patch (and its associated features) at the position x. The likelihood is a combination of the scan match likelihood and the feature match likelihood:

$$L(x) = L_S(x) \prod_i L_i(x)$$

where i is an index over all the features of the patch. The likelihood $L(x)$ is given by the sensor response function $p(r|x, E)$, the probability that we would see the map patch r from the robot pose x, given the environment. The sensor response can be approximated by a correlation operator. A regular grid is imposed on the map area, and for each cell i we calculate the probability $p(r_i)$ of the map patch impinging on the cell and $p(E_i)$ of the map impinging on the cell. The correlation operator is:

$$\sum_i p(r_i)p(m_i)$$

Map *smearing* is employed as an efficient means of approximating uncertainty in the scan readings, and errors in the map. Feature likelihoods are computed using a simple model that assumes a linear degradation in the match with distance and angle to the nearest similar feature. A small false negative response takes care of the situation in which the robot does not see the feature.

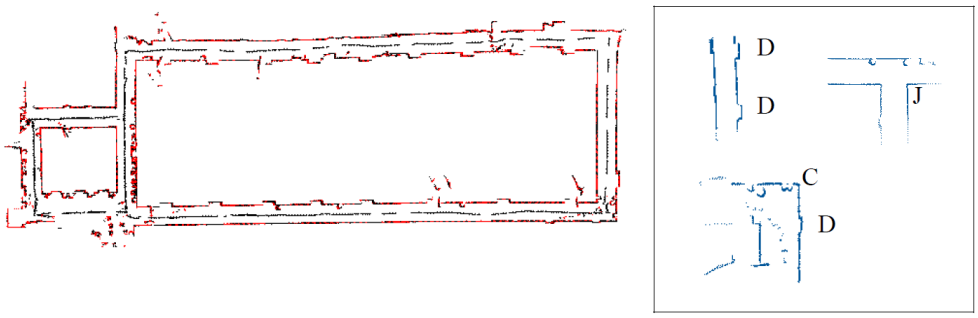
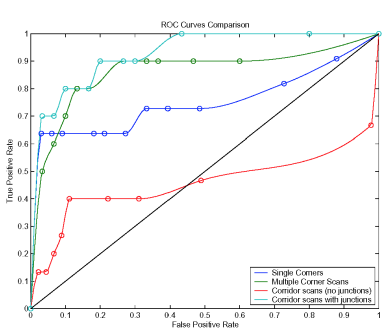


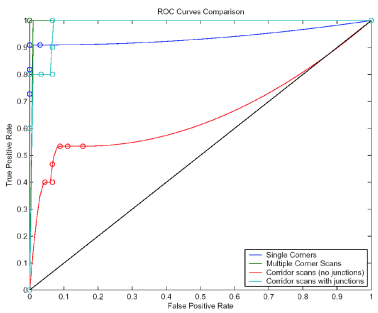
Figure 2.1: *An office environment (60m long) created from laser range scans, and three patches of 15 range scans extracted from the map: a corridor, a junction, and a corner [8]*

2.2.2 Raw Scan Matching vs Feature Matching

Figure 2.2a shows the ROC curves for raw scan matching, over four different classes of scans. An ideal discriminator would have a curve with maximum area, i.e., it would lie along the left and top sides of the graph. Corridors, as expected, have the weakest discrimination, and the most chance of false positives. Even in different buildings, corridors tend to be about the same width, and there are relatively few features along their length to help distinguish them, using raw range matching. Given the relatively coarse grid resolution (200 mm), there were even some cases in which the highest response was from the wrong location. The best discrimination, in terms of both sensitivity and selectivity, comes from corner scans that also contain nearby junctions. But even here, using just 4 sample environments, it is easy to make a wrong decision: there is a significant false positive rate, even for low recognition rates.
When features are added to the matching process, the ROC curves change dramatically (Figure 2.2b). With the exception of the corridor and single corner patches, the response is perfect for certain thresholds, with all patches correctly placed, and no false positives. Although it is not shown in the graph, the robustness of the threshold parameter also improves. For values over 2, there are essentially no false positives; while for values under 5, almost all true positives are present. It is still difficult to match corridors, even with door features identified. In fact, there are several cases in



(a) ROC curve for range scan matching, over all four classes of range scans.



(b) ROC curve for combined range scan and feature matching,over four classes of range scans.

Figure 2.2: ROC curve for range scan matching and feature matching [8]

which the wrong match again has a higher response than the correct one. The placement of doorways along a corridor is thus not a very strong discriminator in these indoor environments, apparently because they occur in repeating patterns. On the other hand, door features around junctions and hall corners are much less ambiguous, and lead to good recognition rates.

2.3 Mapless Navigation

This section discusses a representative selection of mostly reactive visual-based navigation techniques in which navigation is performed without any prior description of the environment. Mapless navigation is scarcely new compared with the previously defined solutions, but new projects using vision systems have been developed in several directions in the last few years. No maps are created, the robots can navigate by observing and extracting relevant information about the landmarks in the environment. These elements can be objects such as desks, boxes, and doorways. The mapless visual navigation techniques discussed here are classified in accordance with the main vision technique or types of clues used during the navigation which are optical flow, appearance based, and object recognition navigation techniques based on feature tracking.

2.3.1 Optical Flow for Robot Navigation

Optical flow is defined as the motion of all the surface elements from the visual world. When a person moves through the world, the objects in the visual environment flow around this person. The human visual system can detect that person direction of travel from the movement of these objects. Optical flow can be defined as the apparent motion of features in a sequence of images. It is believed that when the insect is in relative motion with respect to the environment. Accuracy and the range of operation can be altered by changing the relative speed. For instance, features such as "time-to-contact" (depending on speed) are more relevant than distance when it is necessary to avoid an obstacle. Bernardino and Santos-Victor [9] developed an optical flow-based navigation system imitating the visual behavior of bees, called *Robee*, and was equipped with a stereo vision system, mimicking the centring reflex behaviour used by a bee to navigate safely. The robot localizes itself using the difference between the velocity of the image seen with the left eye and the velocity seen in the right eye. If the difference is close to the zero, the robot keeps moving forward. However, if the velocities are different, the robot moves toward the side whose image changes with a lower velocity. The robot always moves in the direction of the optical flow minor amplitude. The main problem of this technique is that the walls need to be textured enough to present an optimum optical flow computation.

2.3.2 Appearance-Based Methods

Appearance-based methods fundamentally rely on the idea of memorizing the working environment. The main idea is to store images or templates of the environment and associate these images with commands that will steer the robot to its final destination. Appearance-Based Methods are composed by two steps. The first one is the training phase in which images or prominent features of the environment are stored as model templates. The models are labeled with certain localization information, which are then associated with appropriate steering command. Secondly, in the navigation stage, the robot has to identify the environment and localize itself by matching the current image with the stored templates. The main problems with the method are to find an appropriate algorithm for the representation of the environment and to define the on-line matching criteria. An important concept in visual-based mobile robot navigation is the idea of visual homing, inspired by insect behaviour. Insects are able to return to important places in their environment by storing an image of the surroundings while at the goal, and later computing a home direction from a match between this snapshot image and the currently perceived image. For instance, an agent employing a visual homing algorithm captures an image at the home location. It then attempts to return to this location from a nearby position. It compares the current image with the snapshot and infers the direction to the location of the goal from the disparity between these images. It is considered that these aspects of insect behaviour can be a basis for the development of robust navigation algorithms for mobile robots. Visual homing is an appearance-based navigation strategy whose homing algorithms are based on image-based holistic methods using disparities between whole images to compute homing vectors. Image warping is a popular method which is considered to be one of the most reliable visual homing methods for indoor use in this category. It involves calculating the set of all changes in position and orientation between the current and snapshot images. Warping methods distort the current image as if the agent would move according to certain movement parameters. The space of possible movement parameters is then scanned for the parameter combination leading to the warped image that is as similar as possible to the stored image. To do so, warped images are compared to stored image using a pixel-by-pixel correlation measure. The current home vectors are determined based on the strongest similarity

between those images.

2.3.3 Navigation Techniques Based on Feature Tracking

Tracking the motion of moving elements, including lines, corners, or specific regions in a video sequence has become a popular and robust way of performing navigation. Techniques inspired from tracking are called feature-based approaches which, in essence, determine the trajectory and motion of the robot by tracking and finding relative changes in the position of extracted features. Feature-based methods fundamentally segment snapshots and current images into landmarks and background. *Image segmentation* is the process of partitioning a digital image into multiple segments (sets of pixels). The goal of segmentation is to simplify and/or change the representation of an image into something that is more meaningful and easier to analyze. Snapshots are instantaneous copies of instances linked to past moments, while current images are present copies of istances. To operate successfully, feature-based navigation algorithms must extract the same features from snapshots and current images. Each feature extracted from the snapshot image must then be paired with a feature from the current image.

The Scale Invariant Feature Transform (SIFT) [10] method, whose steps are briefly synthetised in Figure 2.3, is a milestone among the techniques aimed at detecting features or relevant points withing images, and nowadays it has become a method commonly used in landmark detection applications. The SIFT extracts features that are invariant to image scaling, rotation, and illumination. During the robot navigation process, invariant features which have been detected are then observed from different points of view, angles, and distances and under different illumination conditions and thus become highly appropriate landmarks to be tracked for navigation. However, SIFT isn't invariant with respect to perspective, so features have to be examined from the same point of view for being reconized.
For any object within an image, interesting points on the object can be extracted to provide a "feature description" of the object. This description, extracted from a training image, can then be used to identify the object when attempting to locate the object in a test image containing many other objects. To perform reliable recognition, it is important that the features extracted from the training image be detectable even under changes in im-

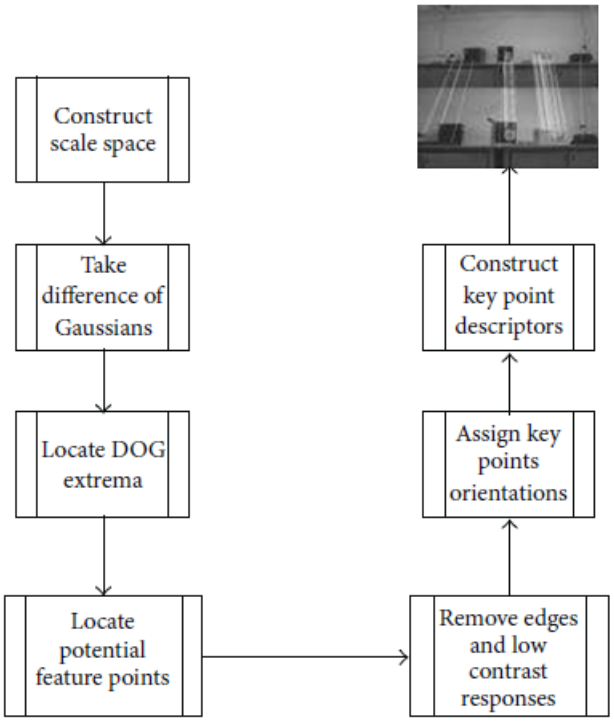


Figure 2.3: *Overview of the SIFT algorithm*

age scale, noise and illumination. Such points usually lie on high-contrast regions of the image, such as object edges. An important characteristic of these features is that the relative positions between them in the original scene shouldn't change from one image to another. For example, if only the four corners of a door were used as features, they would work regardless of the door's position; but if points in the frame were also used, the recognition would fail if the door is opened or closed. Similarly, features located in articulated or flexible objects would typically not work if any change in their internal geometry happens between two images in the set being processed. However, in practice SIFT detects and uses a much larger number of features from the images, which reduces the contribution of the errors caused by these local variations in the average error of all feature matching errors.

Pons et al. [11] employed the SIFT algorithm for feature-based homing, and are utilized to recover the misalignment of orientation between the current and goal positions. The basis of their approach is to construct a representation of the goal location which is stable to changes in the scene using local invariant features.

Chapter 3

Outdoor Navigation

As with indoor navigation, outdoor navigation usually involves obstacle-avoidance, landmark detection, map-building/updating, and position estimation. However, ato the best of our knowledge concerning outdoor navigation, a complete map of the environment is hardly ever known a priori and the system has to cope with the objects as they appear in the scene, without prior information about their expected position. Nevertheless, outdoor navigation can still be divided into two classes according to the level of structure of the environment: outdoor navigation in structured environments and in unstructured environments, like for instance road vs countryside.

3.1 Outdoor Navigation in Structured Environments

Outdoor navigation in structured environments requires some sort of road-following. Road-following means an ability to recognize the lines that separate the lanes or the road from the berm, the texture of the road surface, and the adjoining surfaces, etc. In systems that carry out road following, the models of the environment are usually simple, containing only information such as vanishing points, road and lane widths, etc. Road-following for outdoor robots can be like hallway-following for indoor robots, except for the problems caused by shadows, changing illumination conditions, changing colors, etc.
One of the most prominent pieces of work in outdoor navigation is the

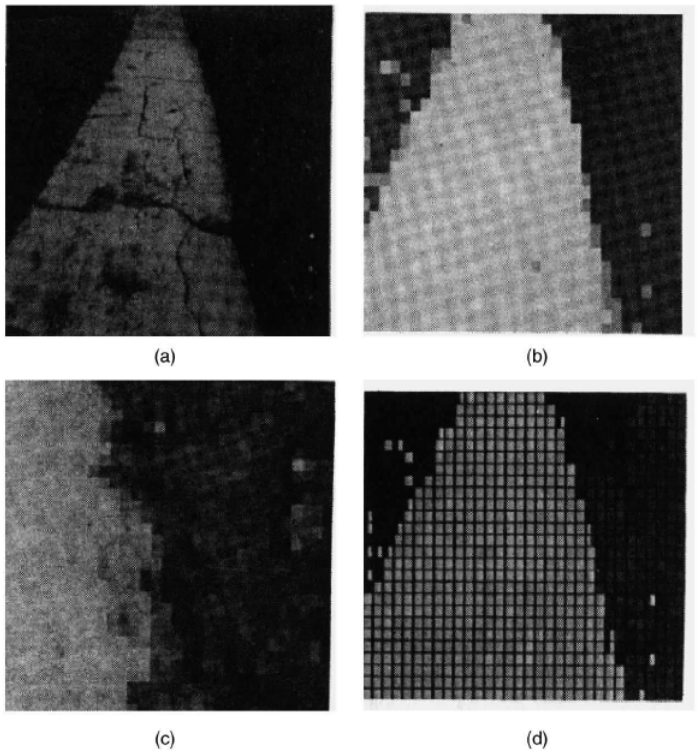


Figure 3.1: *Pixel classification: "road" and "nonroad" classes. (a) Original image. (b) Color classification. (c) Texture classification. (d) Final classification.[12]*

Navlab project and its various incarnations. Navlab, initially developed by Thorpe et al. [12], in its first implementation it used color vision for road following and 3D vision for obstacle detection and avoidance. The R, G, and B images were stored in a pyramid of decreasing resolutions. The images start with 480x512 pixels and end with 30x32 pixels (Fig. 3.1). The higher resolution images are used for texture classification, while the lower resolution images are used for color classification. The algorithm for road following consists of three stages: pixel classification, best-fit road position vote, and color update. Pixel classification in Navlab is performed by combining two separate approaches: color classification and texture classification. During color classification, each pixel is assigned a class depending on the color distributions. The road pixels are represented by four sepa-

rate Gaussian clusters to account for changes in illumination, road surface properties, etc. Similarly, the nonroad pixels are also represented by four separate Gaussian clusters. Each Gaussian distribution is represented by a three-dimensional mean vector (mean R, G, and B), by a three-by-three covariance matrix, and by the fraction of pixels expected a priori to be represented by that distribution. Navlab bases texture classification on the observation that road regions tend to appear much smoother in an image compared to non-road regions. Smoothness is measured via a normalized gradient index that uses a high-resolution gradient (at 240x256 pixels) divided by a weighted sum of a low-resolution gradient (at 60x64 pixels) and the mean pixel value per region. The smoothnes indices are thresholded to produce a counter of "microedges" per 8x8 block of pixels. The result is a 30x32 texture image with values between 0 and 255 representing the number of microedge pixels in the corresponding 8x8 block of the full-resolution image. As mentioned before, the final classification of a pixel is carried out by combining the results obtained from color and texture. Once the pixels are classified, a Hough-like transform [13] is applied to the road pixels to obtain the two parameters that specify the road vanishing point and orientation: intercept, P, and orientation, θ . As in a Hough transform, each pixel votes for the points in the (P, θ) space to which it could belong. The point in the (P, θ) space with the most votes is the one that represents the parameters of the road. Finally, once the position of the road and its edges leading to a vanishing point are determined using the above voting procedure, the pixels are reclassified taking into account the determined road edges.

3.2 Unstructured Outdoor Navigation

An outdoor environment with no regular properties that could be perceived and tracked for navigation may be referred to as unstructured environment. In such cases, the vision system can make use of at most a generic characterization of the possible obstacles in the environment.
Unstructured environments arise in cross-country navigation as, for example, in planetary (lunar/martian-like) terrain navigation. Sometimes in these sorts of applications, the robot is supposed to just wander around, exploring the vicinity of the robot without a clearcut goal. However, in other applications, the task to be performed requires that the robot follow

a specific path to a goal position. In those cases, a map of the traversed areas has to be created and a localization algorithm has to be implemented in order to carry out goal-driven navigation. When maps need to be built by a robot, they may be created in either a vehicle centered coordinate frame or with respect to some external reference, such as an external camera attached to a companion device or a virtual viewpoint with respect to which range data is projected to form a Cartesian Elevation Map or a global-positioning reference such as the sun. A highly noted accomplishment in vision-based navigation in unstructured environments is the Mars Pathfinder project. The mission was performed by two devices: the lander and the rover. The lander is a 1.5 m wide and 1.0 m high tetrahedron with a mass of 264 kg. It is composed of a VME based, RAD 6000 computer with 128Mb of memory, a multispectral stereo camera pair with a pan/tilt base mounted on a mast 1.5 m above the ground. The cameras have a baseline of 15 cm and resolution of 256x256 pixels with an angular resolution of 0.001 radians per pixel. The rover is 65 cm long, 48 cm wide, and 30 cm high. It is equipped with an Intel 8085 processor and 500Kb of memory. The navigation by the rover, carried out in cooperation with the lander, consists of four different functions:

- Goal designation
- Path selection
- Rover localization
- Hazard detection

In goal designation, human operators at the mission control used a Web-based tool to specify waypoints in 3D views of the landing site that were generated from images obtained using the stereo cameras on the lander. This required that a map of the landing site, centered at the lander, is maintained by the mission control and the images recorded by the lander is used for tracking and locating the rover. Waypoint coordinates were extracted from the 3D views by pointing to pixels in a stereographic display using a special device called the spaceball. Once the 3D coordinates were extracted, the rover could be commanded to move towards those coordinates using a simple path selection algorithm:

```
if there is no hazard then
    move forward and turn toward the goal
else
    if there is a hazard on the left then
        turn in place to the right until no hazard is detected
    else
        if there is a hazard on the right then
            turn in place to the left until no hazard is detected
        end if
    end if
end if
```

The mission-controlled supplied localization of the rover was transmitted to the lander once a day. During the rest of the day, the rover would be on its own, using deadreckoning to figure out its position. Because the deadreckoning-based positional errors tend to accumulate, it was determined by simulation of typical navigation scenarios that the rover would be allowed to travel at most 10 m/day. Once the rover was commanded to move, a hazard detection algorithm was started. The rover moved at speeds of 15 cm/s and stopped for hazard detection every 6.5 cm (one wheel radius). The hazard detection algorithm used two cameras and five laser diode-based light stripe emitters. The stripes extended in front of the vehicle for about 30 cm and to the left or right for 13 cm. The hazard detection algorithm searched for four points along each stripe and assumed the presence of a hazard if: *i)* any of the nearest two points for any stripe is not detected or *ii)* the elevation difference between any two 8-connected neighbors in the 4x5 measurement array exceeded a given threshold or *iii)* the difference between the highest and the lowest elevation of the whole area exceeded another threshold.

Chapter 4

Conclusions

In the last decades, vision has become one of the most cheap, challenging and promising via for robots to perceive the environment. The number of prominent navigation approaches based on vision sensors have increased exponentially. Visual navigation techniques have been applied on almost all environments and in all kind of robots.
If the goal is to send a mobile robot from one coordinate location to another coordinate location, there is sufficient accumulated expertise in the research community today to design a mobile robot that could do that in a typical building (tasks such as opening doors, calling elevators, climbing steps, etc.). But, if the goal is to carry out function-driven navigation we are still far. Useful navigation where a robot must be aware of the meaning of the objects encountered in the environment is beset with a harder-to-solve version of the central problem of general computer vision—automatic scene interpretation [14]. Scene interpretation for mobile robots is a harder problem than for stationary arm robots because, in the mobile context, there is much less control over illumination and background scene clutter. It's probably safe to predict that the near future will witness progress in task and environment specific mobile robots. We should see vision-equipped versions of robots that have already been deployed on experimental basis in patient-care, building security, hazardous-site inspection, etc. With vision, such robots would conceivably engage in smarter navigation and smarter interaction with people and objects in the environment.

Bibliography

[1] Howie M Choset et al. *Principles of robot motion: theory, algorithms, and implementation*. MIT press, 2005.

[2] Tetlow S.W. Allwood R.L. Dalgleish F.R. *Vision-based navigation of unmanned underwater vehicles: a survey*.

[3] Raja Chatila and Jean-Paul Laumond. "Position referencing and consistent world modeling for mobile robots". In: *Robotics and Automation. Proceedings. 1985 IEEE International Conference on*. Vol. 2. IEEE. 1985, pp. 138–145.

[4] H.P. Moravec and A. Elfes. *High Resolution Maps from Wide Angle Sonar*.

[5] S. Atiya and G.D. Hager. *Real-Time Vision-Based Robot Localization*.

[6] Kurt Konolige et al. "Map merging for distributed robot navigation." In: *IROS*. 2003, pp. 212–217.

[7] Dieter Fox et al. "Particle filters for mobile robot localization". In: *Sequential Monte Carlo methods in practice*. Springer, 2001, pp. 401–428.

[8] Kurt Konolige et al. "Map merging for distributed robot navigation." In: *IROS*. 2003, pp. 212–217.

[9] Alexandre Bernardino and José Santos-Victor. "Visual behaviours for binocular tracking". In: *Robotics and Autonomous Systems* 25.3-4 (1998), pp. 137–146.

[10] Zheng Yi, Cao Zhiguo, and Xiao Yang. "Multi-spectral remote image registration based on SIFT". In: *Electronics Letters* 44.2 (2008), pp. 107–108.

[11] J Saez Pons et al. "Vision-based robot homing in dynamic environments". In: *13th IASTED International Conference on Robotics and Applications*. 2007, pp. 293–298.

[12] Charles Thorpe et al. "Vision and navigation for the Carnegie-Mellon Navlab". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 10.3 (1988), pp. 362–373.

[13] John Illingworth and Josef Kittler. "A survey of the Hough transform". In: *Computer vision, graphics, and image processing* 44.1 (1988), pp. 87–116.

[14] Bernd Neumann and Ralf Möller. "On scene interpretation with description logics". In: *Cognitive Vision Systems*. Springer, 2006, pp. 247–275.

[15] Andrew P Duchon, Leslie Pack Kaelbling, and William H Warren. "Ecological robotics". In: *Adaptive Behavior* 6.3-4 (1998), pp. 473–507.

[16] Kristoffer Sjö et al. "Object search and localization for an indoor mobile robot". In: *Journal of Computing and Information Technology* 17.1 (2009), pp. 67–80.