EURASIP Journal on Applied Signal Processing 2004:17, 2650–2662

Autonomous Mobile Robot That Can Read

Dominic Létourneau
Research Laboratory on Mobile Robotics and Intelligent Systems (LABORIUS), Department of Electrical Engineering and Computer Engineering, University of Sherbrooke, Sherbrooke, Quebec, Canada J1K 2R1
Email: dominic.letourneau@usherbrooke.ca

François Michaud
Research Laboratory on Mobile Robotics and Intelligent Systems (LABORIUS), Department of Electrical Engineering and Computer Engineering, University of Sherbrooke, Sherbrooke, Quebec, Canada J1K 2R1
Email: francois.michaud@usherbrooke.ca

Jean-Marc Valin
Research Laboratory on Mobile Robotics and Intelligent Systems (LABORIUS), Department of Electrical Engineering and Computer Engineering, University of Sherbrooke, Sherbrooke, Quebec, Canada J1K 2R1
Email: jean-marc.valin@usherbrooke.ca

Received 18 January 2004; Revised 11 May 2004; Recommended for Publication by Luciano da F. Costa

The ability to read would surely contribute to increased autonomy of mobile robots operating in the real world. The process seems fairly simple: the robot must be capable of acquiring an image of a message to read, extract the characters, and recognize them as symbols, characters, and words. Using an optical *Character Recognition* algorithm on a mobile robot however brings additional challenges: the robot has to control its position in the world and its pan-tilt-zoom camera to find textual messages to read, potentially having to compensate for its viewpoint of the message, and use the limited onboard processing capabilities to decode the message. The robot also has to deal with variations in lighting conditions. In this paper, we present our approach demonstrating that it is feasible for an autonomous mobile robot to read messages of specific colors and font in real-world conditions. We outline the constraints under which the approach works and present results obtained using a Pioneer 2 robot equipped with a Pentium 233 MHz and a Sony EVI-D30 pan-tilt-zoom camera.

Keywords and phrases: character recognition, autonomous mobile robot.

1. INTRODUCTION

Giving to mobile robots the ability to read textual messages is highly desirable to increase their autonomous navigating in the real world. Providing a map of the environment surely can help the robot localize itself in the world (e.g., [1]). However, even if we humans may use maps, we also exploit a lot of written signs and characters to help us navigate in our cities, office buildings, and so on. Just think about road signs, street names, room numbers, exit signs, arrows to give directions, and so forth. We use maps to give us a general idea of the directions to take to go somewhere, but we still rely on some forms of symbolic representation to confirm our location in the world. This is especially true in dynamic and large open areas. Car traveling illustrates that well. Instead of only looking at a map and the vehicle's tachometer, we rely on road signs to give us cues and indications on our progress toward our destination. So similarly, the ability to read characters, signs, and messages would undoubtedly be a very useful complement for robots that use maps for navigation [2, 3, 4, 5].

The process of reading messages seems fairly simple: acquire an image of a message to read, extract the characters, and recognize them. The idea of making machines read is not new, and research has been going on for more than four decades [6]. One of the first attempts was in 1958 with Frank Rosenblatt demonstrating his Mark I Perceptron neurocomputer, capable of *Character Recognition* [7]. Since then, many systems are capable of recognizing textual or handwritten characters, even license plate numbers of moving cars using a fixed camera [8]. However, in addition to *Character Recognition*, a mobile robot has to find the textual message to capture as it moves in the world, position itself autonomously in front of the region of interest to get a good image to process, and use its limited onboard processing capabilities to decode the message. No fixed illumination, stationary backgrounds, or correct alignment can be assumed.

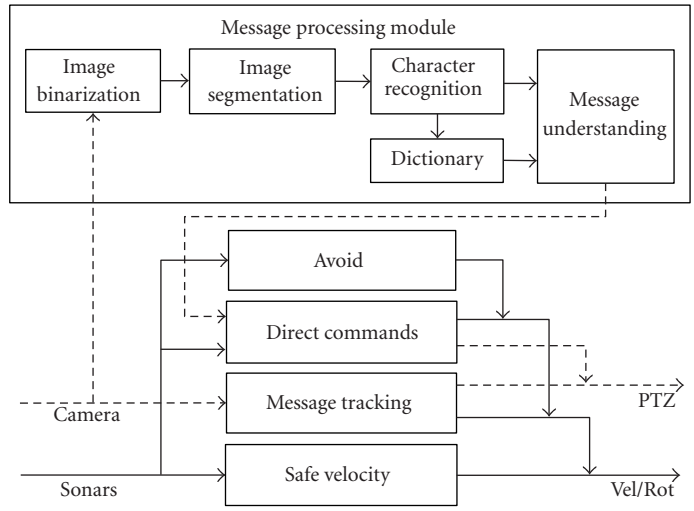


Figure 1: Software architecture of our approach.

So in this project, our goal is to address the different aspects required in making an autonomous robot recognize textual messages placed in real-world environments. Our objective is not to develop new *Character Recognition* algorithms. Instead, we want to integrate the appropriate techniques to demonstrate that such intelligent capability can be implemented on a mobile robotic platform and under which constraints, using current hardware and software technologies. Our approach processes messages by extracting characters one by one, grouping them into strings when necessary. Each character is assumed to be made of one segment (all connected pixels): characters made of multiple segments are not considered. Messages are placed perpendicular to the floor on flat surfaces, at about the same height of the robot. Our approach integrates techniques for (1) perceiving characters using color segmentation, (2) positioning and capturing an image of sufficient resolution using behavior-producing modules and proportional-integral-derivative (PID) controllers for the autonomous control of the pan-tilt-zoom (PTZ) camera, (3) exploiting simple heuristics to select image regions that could contain characters, and (4) recognizing characters using a neural network.

The paper is organized as follows. Section 2 provides details on the software architecture of the approach and how it allows a mobile robot to capture images of messages to read. Section 3 presents how characters and messages are processed, followed in Section 4 by experimental results. Experiments were done using a Pioneer 2 robot equipped with a Pentium 233 MHz and a Sony EVI-D30 PTZ camera. Section 5 presents related work, followed in Section 6 with a conclusion and future work.

2. CAPTURING IMAGES OF MESSAGES TO READ

Our approach consists of making the robot move autonomously in the world, detect a potential message (characters, words, or sentences) based on color, stop, and acquire an image with sufficient resolution for identification, one character at a time starting from left to right and top to bottom. The software architecture of the approach is shown in Figure 1. The control of the robot is done using four behavior-producing modules arbitrated using Subsumption [9]. These behaviors control the velocity and the heading of the robot, and also generate the PTZ commands to the camera. The behaviors implemented are as follows: *Safe-Velocity* to make the robot move forward without colliding with an object (detected using sonars); *Message-Tracking* to track a message composed of black regions over a colored or white background; *Direct-Commands* to change the position of the robot according to specific commands generated by the *Message Processing Module*; and *Avoid*, the behavior with the highest priority, to move the robot away from nearby obstacles based on front sonar readings. The *Message Processing Module*, described in Section 4, is responsible for processing the image taken by the *Message-Tracking* behavior for message recognition.

The *Message-Tracking* behavior is an important element of the approach because it provides the appropriate PTZ commands to get the maximum resolution of the message to identify. Using an algorithm for color segmentation, the *Message-Tracking* behavior allows the robot to move in the environment until it sees with its camera black regions, presumably characters, surrounded by a colored background (either orange, blue, or pink) or white area. To do so, two processes are required: one for color segmentation, allowing to detect the presence of a message in the world, and one for controlling the camera.

2.1. Color segmentation on a mobile robot

Color segmentation is a process that can be done in real time with the onboard computer of our robots, justifying why we used this method to perceive messages. First a color space must be selected from the one available by the hardware used for image capture. Bruce et al. [10] present a good summary

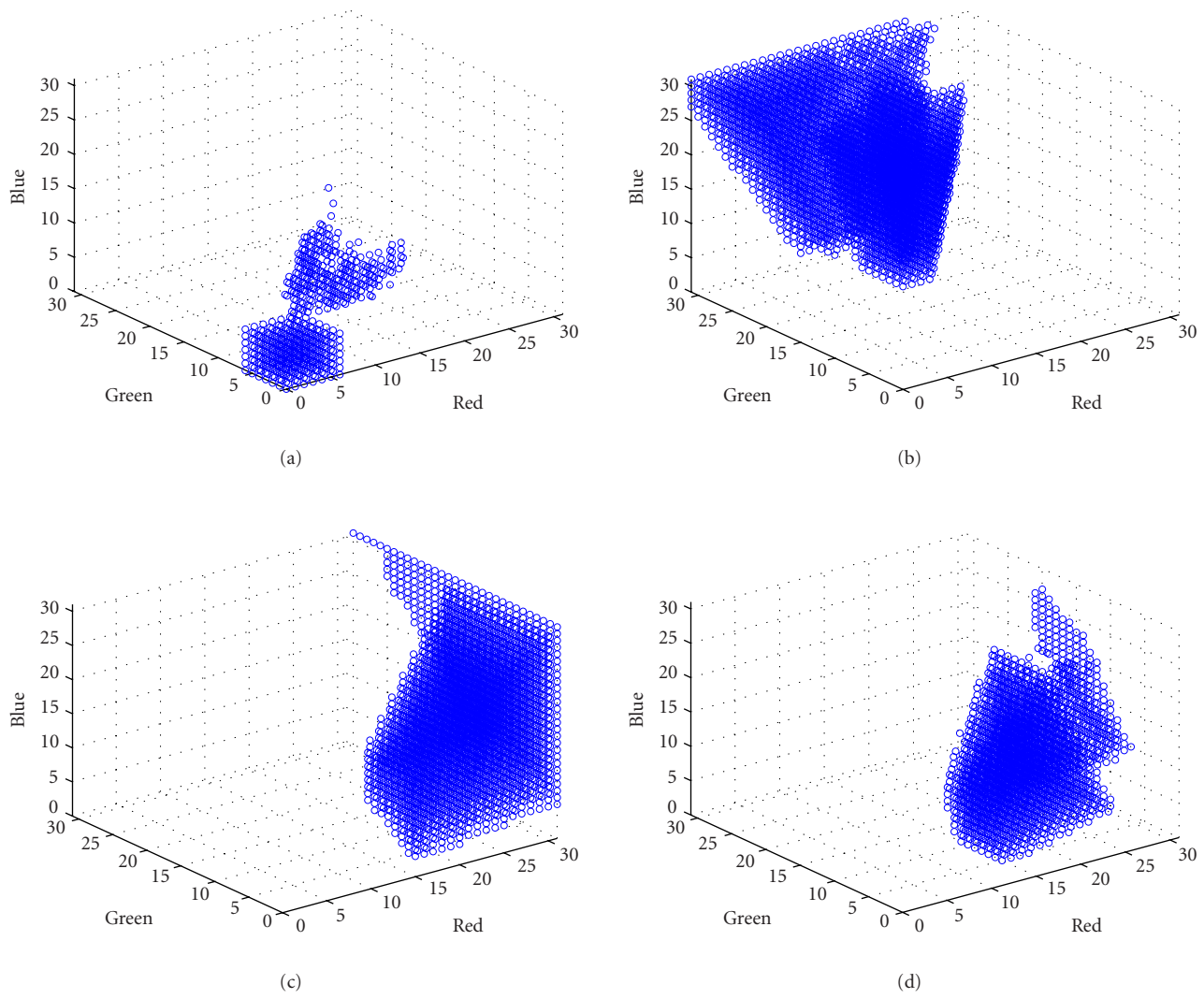
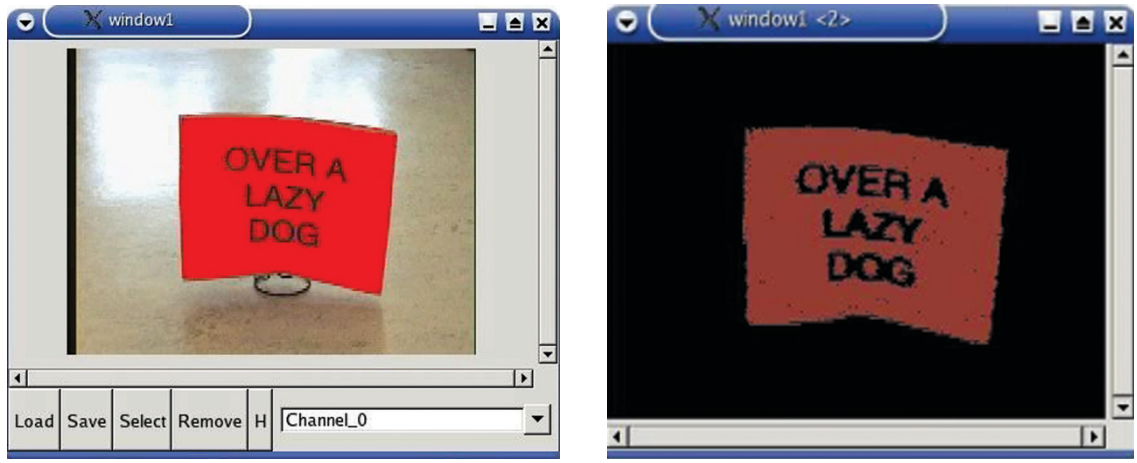


Figure 2: Color membership representation in the RGB color space for (a) black, (b) blue, (c) pink, and (d) orange.

of the different approaches for doing color segmentation on mobile robotic platforms, and describe an algorithm using the YUV color format and rectangular color threshold values stored into three lookup tables (one for Y, U, and V, resp.). The lookup values are indexed by their Y, U, and V components. With Y, U, and V encoded using 8 bits each, the approach uses three lookup tables of 256 entries. Each entry of the tables is an unsigned integer of 32 bits, where each bit position corresponds to a specific color channel. Thresholds verification of all 32 color channels for a specific Y, U, and V values are calculated with three lookups and two logical AND operations. Full segmentation is accomplished using 8 connected neighbors and grouping pixels that correspond to the same color into blobs.

In our system, we use a similar approach, using however the RGB format, that is, 0RRRRRGGGGGBBBBB, 5 bits for each of the R, G, B components. It is therefore possible to generate only one lookup table of 2^{15} entries (or 32 768 entries) 32 bits long, which is a reasonable lookup size. Using one lookup table indexed using RGB components to define colors has several advantages: colors that would require multiple thresholds to define them in the RGB format (multiple cubic-like volumes) are automatically stored in the lookup table; using a single lookup table is faster than using multiple if-then conditions with thresholds; membership to a color channel is stored in a single-bit (0 or 1) position; color channels are not constrained to using rectangular-like thresholds (this method does not perform well for color segmentation under different lighting conditions) since each combination of the R, G, and B values corresponds to only one entry in the table. Figure 2 shows a representation of the black, blue, pink, and orange colors in the RGB color space as it is stored in the lookup table.

To use this method with the robot, color channels associated with elements of potential messages must be trained. To help build the membership lookup table, we first define



(a) (b)

Figure 3: Graphical user interface for training of color channels.

colors represented in HSV (hue, saturation, value) space. Cubic thresholds in the HSV color format allow a more comprehensive representation of colors to be used for perception of the messages by the robot. At the color training phase, conversions from the HSV representation with standard thresholds to the RGB lookup table are easy to do. Once this initialization process is completed, adjustments to variations of colors (because of lighting conditions for instance) can be made using real images taken from the robot and its camera. In order to facilitate the training of color channels, we designed a graphical user interface (GUI), as shown in Figure 3. The window (a) provides an easy way to select colors directly from the source image for a desired color channel and stores the selected membership pixel values in the color lookup table. The window (b) provides an easy way to visualize the color perception of the robot for all the trained color channels.

2.2. Pan-tilt-zoom control

When a potential message is detected, the *Message-Tracking* behavior makes the robot stop. It then tries to center the agglomeration of black regions in the image (more specifically, the center of area of all the black regions) as it zooms in to get the image with enough resolution.

The algorithm works in three steps. First, since the goal is to position the message (a character or a group of characters) in the center of the image, the x, y coordinates of the center of the black regions is represented in relation to the center of the image. Second, the algorithm must determine the distance in pixels to move the camera to center the black regions in the image. This distance must be carefully interpreted since the real distance varies with current zoom position. Intuitively, smaller pan and tilt commands must be sent when the zoom is high because the image represents a bigger version of the real world. To model this influence, we put an object in front of the robot, with the camera detecting the object in the center of the image using a zoom value of 0. We measured the length in pixels of the object and took such readings at different zoom values (from 0 to maximum range). Considering as a reference the length of the object at zoom 0, the length ratios LRs at different zoom values were evaluated to derive a model for the Sony EVI-D30 camera, as expressed by (1). Then, for a zoom position Z, the x, y values of the center of area of all the black regions are divided by the corresponding LR to get the real distance $\tilde{x}$, $\tilde{y}$ (in pixels) between the center of area of the characters in the image and the center of the image, as expressed by (2).

$$\mathrm{LR} = 0.68 + 0.0041 \cdot Z + 8.94 \times 10^{-6} \cdot Z^2 + 1.36 \times 10^{-8} \cdot Z^3, \tag{1}$$

$$\tilde{x} = \frac{x}{\mathrm{LR}}, \qquad \tilde{y} = \frac{y}{\mathrm{LR}}. \tag{2}$$

Third, PTZ commands must be determined to position the message at the center of the image. For pan and tilt commands (precisely to a 10th of a degree), PID controllers [11] are used. There is no dependance between the pan commands and the tilt commands: both pan and tilt PID controllers are set independently and the inputs of the controllers are the errors $(\tilde{x}, \tilde{y})$ measured in number of pixels from the center of area of the black regions to the center of the image. PIDs parameters were set following Ziegler-Nichols method: first increase the proportional gain from 0 to a critical value, where the output starts to exhibit sustained oscillations; then use Ziegler-Nichols' formulas to derive the integral and derivative parameters.

At a constant zoom, the camera is able to position itself with the message at the center of the image in less than 10 cycles (i.e., 1 second). However, simultaneously, the camera must increase its zoom to get an image with good resolution of the message to interpret. A simple heuristic is used to position the zoom of the camera to maximize the resolution of

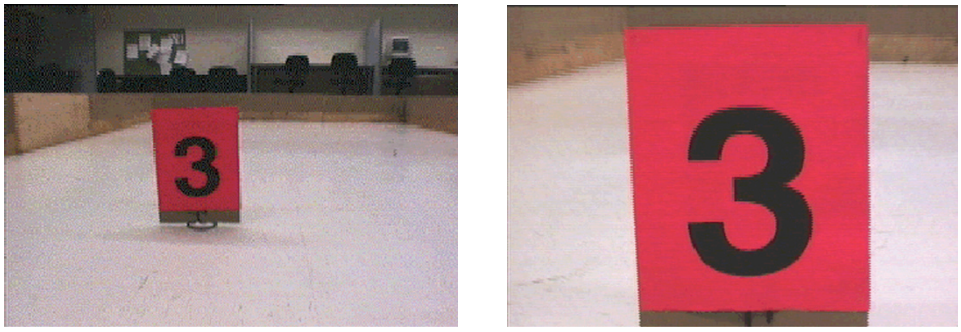


Figure 4: Images with normal and maximum resolution captured by the robot.

(1) IF $|\bar{x}| < 30$ AND $|\bar{y}| < 30$
(2) IF $z > 30$ $Z = Z + 25/\text{LR}$
(3) ELSE IF $z < 10$ $Z = Z - 25/\text{LR}$
(4) ELSE $Z = Z - 25/\text{LR}$

Algorithm 1

the characters in the message. The algorithm allows to keep in the middle of the image the center of gravity of all of the black areas (i.e., the characters), and zoom in until the edges z of the black regions of the image are within 10 to 30 pixels of the borders. The heuristic is given in Algorithm 1.

Rule (1) implies that the black regions are close to being at the center of the image. Rule (2) increases the zoom of the camera when the distance between the black regions and the edge of the colored background is still too big, while rule (3) decreases the zoom if it is too small. Rule (4) decreases the zoom when the black regions are not centered in the image, to make it possible to see more clearly the message and facilitate centering it in the image. The division by the LR factor allows slower zoom variation when the zoom is high, and higher when the zoom is low. Note that one difficulty with the camera is caused by its auto-exposure and advanced backlight compensation systems. By changing the position of the camera, the colors detected may vary slightly. To account for that, the zoom is adjusted until stabilization of the PTZ controls is observed over a period of five processing cycles. Figure 4 shows an image with normal and maximum resolution of the digit 3 perceived by the robot.

Overall, images are processed at about 3 to 4 frames per second. After having extracted the color components of the image, most of the processing time of the *Message-Tracking* behavior is taken sending small incremental zoom commands to the camera in order to insure the stability of the algorithm. Performances can be improved with a different camera with quicker response to the PTZ commands. Once the character is identified, the predetermined or learned meaning associated with the message can be used to affect the robot's behavior. For instance, the message can be processed by a planning algorithm to change the robot's goal. In the simplest scheme, a command is sent to the *Direct-Commands* behavior to make the robot move away from the message not to read it again. If the behavior is not capable of getting stable PTZ controls, or *Character Recognition* reveals to be too poor, the *Message Processing Module*, via the *Message Understanding* module, gives command to the *Direct-Commands* behavior to make the robot move closer to the message, to try recognition again. If nothing has been perceived after 45 seconds, the robot just moves away from the region.

3. MESSAGE PROCESSING MODULE

Once an image with maximum resolution is obtained by the *Message-Tracking* behavior, the *Message Processing Module* can now begin the *Character Recognition* procedure, finding lines, words, and characters in the message and identifying them. This process is done in four steps: *Image Binarization*, *Image Segmentation*, *Character Recognition*, and *Message Understanding* (to affect or be influenced by the decision process of the robot). Concerning image processing, simple techniques were used in order to minimize computations, the objective pursued in this work being the demonstration of the feasibility of a mobile robot to read messages, and not the evaluation or the development of the best image processing techniques for doing so.

3.1. Image binarization

Image binarization consists of converting the image into black and white values (0,1) based on its grey-scale representation. Binarization must be done carefully using proper thresholding to avoid removing too much information from the textual message. Figure 5 shows the effect of different thresholds for the binarization of the same image.

Using hard-coded thresholds gives unsatisfactory results since it can not take into consideration variations in the lighting conditions. So the following algorithm is used to adapt the threshold automatically.

(1) The intensity of each pixel of the image is calculated using the average intensity in RGB. Intensity is then transformed in the $[0, 1]$ grey-scale range, 0 representing completely black and 1 representing completely white.
(2) Randomly selected pixel intensities in the image (empirically set to 1% of the image pixels) are used to compute the desired threshold. Minimum and maximum

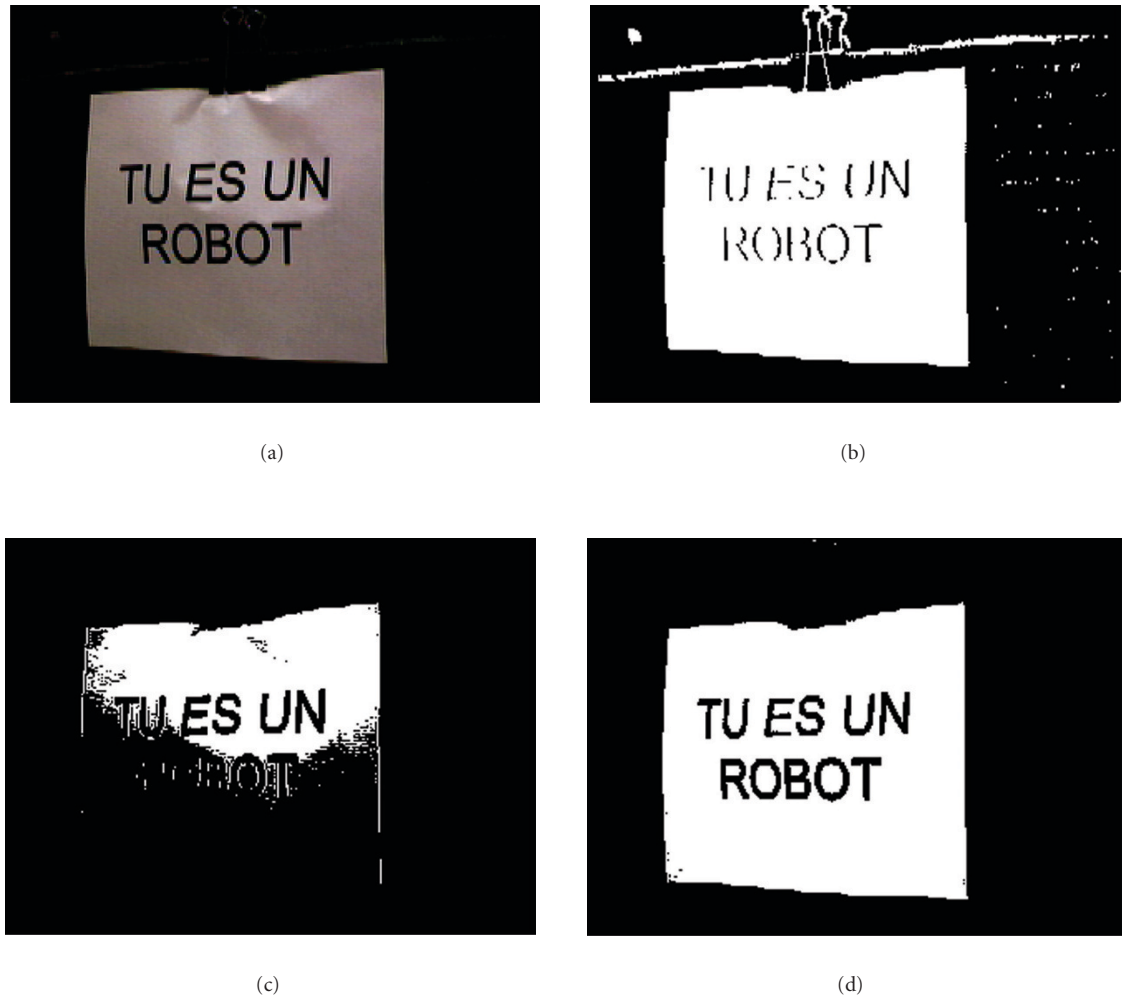


Figure 5: Effects of thresholds on binarization: (a) original image, (b) large threshold, (c) small threshold, and (d) proper threshold.

image intensities are found using these pixels. We experimentally found that the threshold should be set at 2/3 of the maximum pixel intensity minus the minimum pixel intensity found in the randomly selected pixels. Using only 1% of the pixels for computing the threshold offers good performances without requiring too much calculations.

(3) Binarization is performed on the whole image converting pixels into binary values. Pixels with intensity higher than or equal to the threshold are set to 1 (white) while the others are set to 0 (black).

3.2. Image segmentation

Once the image is binarized, black areas are extracted using standard segmentation methods [10, 12]. The process works by looking, pixel by pixel (from top to bottom and left to right), if the pixel and some of its eight neighbors are black. Areas of black pixels connected with each other are then delimited by rectangular bounding boxes. Each box is characterized by the positions of all pixels forming the region, the center of gravity of the region (x_c, y_c), the area of the region, and the upper-left and lower-right coordinates of the bounding box. Figure 6 shows the results of this process. In order to prevent a character from being separated in many segments (caused by noise or bad color separation during the binarization process), the segmentation algorithm allows connected pixels to be separated by at most three pixels. This value can be set in the segmentation algorithm and must be small enough to avoid connecting valid characters together.

Once the black areas are identified, they are grouped into lines by using the position of the vertical center of gravity (y_c) and the height of the bounding boxes, which are in fact the characters of the message. To be a part of a line, a character must respect the following criteria.

(i) In our experiments, minimum height is set to 40 pixels (which was set to allow characters to be recognized easily by humans and machines). No maximum height is specified.

Figure 6: Results of the segmentation of black areas.

(ii) The vertical center of gravity (y_c) must be inside the vertical line boundaries. Line boundaries are found using the following algorithm. The first line, L_1, is created using the upper-left character $c1$. Vertical boundaries for line L_1 are set to $y_{c1} \pm (h_{c1}/2 + K)$, with h_{c1} the height of the character $c1$ and K being a constant empirically set to $0.5 \cdot h_{c1}$ (creating a range equal to twice its height). For each character, the vertical center of gravity y_{ci} is compared to the line boundaries of line L_j: if so, then the character i belongs to the line j; otherwise, a new line is created with vertical boundaries set to $y_{ci} \pm (h_{ci}/2 + K)$ and $K = 0.5 \cdot h_{ci}$. A high value of K allows to consider characters seen in a diagonal as being part of the same line. Adjacent lines in the image having a very small number of pixels constitute a line break. Noise can deceive this simple algorithm, but adjusting the noise tolerance usually overcomes this problem.

With the characters localized and grouped into lines, they can be grouped into words by using a similar algorithm: going from left to right, characters are grouped into a word if the horizontal distance between two characters is under a specified tolerance (set to the average character's width multiplied by a constant set empirically to 0.5). Spaces are inserted between the words found.

3.3. Character recognition

The algorithm we used in this first implementation of our system is based on standard backpropagation neural networks, trained with the required sets of characters under different lighting conditions. Backpropagation neural networks can be easily used for basic *Character Recognition*, with good performance even for noisy inputs [13]. A feedforward network with one hidden layer is used, trained with the delta-bar-delta [14] learning law, which adapts the learning rate of the back-propagation learning law. The activation function used is the hyperbolic tangent, with activation values of +1 (for a black pixel) and −1 (for a white pixel). The output layer of the neural network is made of one neuron per character in the set. A character is considered recognized when the output neuron associated with this character has the maximum activation value greater to 0.8. Data sets for training and testing the neural networks were constructed by letting the robot move around in an enclosed area with the same character placed in different locations, and by memorizing the images captured. The software architecture described in Section 2 was used for doing this. Note that no correction to compensate for any rotation (skew) of the character is made by the algorithm. Images in the training set must then contain images taken at different angles of view of the camera in relation to the perceived character. Images were also taken of messages (characters, words) manually placed at different angles of vision in front of the robot to ensure an appropriate representation of these cases in the training sets. Training of the neural networks is done off-line over 5000 epochs (an epoch corresponds to a single pass through the sequence of all input vectors).

3.4. Message understanding

Once one or multiple characters have been processed, different analysis can be done. For instance, for word analysis, performance can be easily improved by the addition of a dictionary. In the case of using a neural network for *Character Recognition*, having the activation values of the output neurons transposed to the $[0, 1]$ interval, it can be shown that they are a good approximation of $P(x_k = w_k)$, the probability of occurrence of a character x at position k in the word w of length N. This is caused by the mean square minimization criterion used during the training of the neural network [15]. For a given word $\mathbf{w}$ in the dictionary, the probability that the observation $\mathbf{x}$ corresponds to the word $\mathbf{w}$ is given by the product of the individual probabilities of each character in the word, as expressed by

$$P(\mathbf{x}|\mathbf{w}) = \prod_{k=1}^{N} P(x_k = w_k). \tag{3}$$

The word in the dictionary with the maximum probability is then selected simply by taking the best match W using the maximum likelihood criterion given by

$$W = \underset{\mathbf{w}}{\operatorname{argmax}}\, P(\mathbf{x}|\mathbf{w}). \tag{4}$$

4. RESULTS

The robots used in the experiments are Pioneer 2 robots (DX and AT models) with 16 sonars, a PTZ camera, and a Pentium 233 MHz PC-104 onboard computer with 64 Mb of RAM. The camera is a Sony EVI-D30 with 12X optical zoom, high-speed auto-focus lens and a wide-angle lens, pan range of ±90° (at a maximum speed of 80°/s), and a tilt range of ±30° (at a maximum speed of 50°/s). The camera also uses auto-exposure and advanced backlight compensation systems to ensure that the subject remains bright even in harsh backlight conditions. This means that brightness of the image is automatically adjusted when zooming on an object. The frame grabber is a PXC200 color frame grabber from imagenation, which provides in our design 320 × 240 images at a maximum rate of 30 frames per

(a) (b)

Figure 7: (a) Pioneer 2 AT robot in front of a character and (b) Pioneer 2 DX in front of a message.

second. However, commands and data exchanged between the onboard computer and the robot controller are set at 10 Hz. All processing for controlling the robot and recognizing characters is done on the onboard computer. RobotFlow (http://robotflow.sourceforge.net) is the programming environment used. Figure 7 represents the setup.

The experiments were done in two phases: Phase 1 consisted in making the robot read one character per sheet of paper, and Phase 2 extended this capability to the interpretation of words and sentences. For Phase 1, the alphabet was restricted to numbers from 0 to 9, the first letters of the names of our robots (H, C, J, V, L, A), the four cardinal points (N, E, S, W), front, right, bottom, and left arrows, and a charging station sign, for a total of 25 characters. Fonts used were Arial and Times. In Phase 1, tests were made with different neural network topologies in order to find adequate configurations for *Character Recognition* only. For Phase 2, the character set was 26 capital letters (A to Z, Arial font) and 10 digits (0 to 9) in order to generate words and sentences. All symbols and messages were printed in black on a legal size (8.5 inches $\times$ 11 inches) sheet of paper (colored or white, specified as a parameter in the algorithm). Phase 2 focused more on the recognition of sets of words, from the first line to the last, word by word, sending characters one by one to the neural network for recognition and then applying the dictionary.

4.1. Phase 1

In this phase, the inputs of the neural networks are taken from a scaled image, 13 $\times$ 9 pixels, of the bounding box of the character to process. This resolution was set empirically: we estimated visually that this was a sufficiently good resolution to identify a character in an image. Fifteen images for each of the characters were constructed while letting the robot moves autonomously, while thirty five were gathered using manually placed characters in front of the robot not in motion. Then, of the 50 images for each character, 35 images were randomly picked for the training set, and the 15 images left were used for the testing set.

Tests were done using different neural network configurations such as having one neural network for each character, one neural network for all of the characters (i.e., with 25 output neurons), and three neural networks for all of the characters, with different number of hidden neurons and using a majority vote (2 out of 3) to determine that the character is correctly recognized or not. The best performance was obtained with one neural network for all of the characters, using 11 hidden neurons. With this configuration, all characters in the training set were recognized, with 1.8% of incorrect recognition for the testing set [16].

We also characterized the performance of the proposed approach in positioning the robot in front of a character and in recognizing characters in different lighting conditions. Three sets of tests were conducted. First, we placed a character at various distances in front of the robot, and recorded the time required to capture the image with maximum resolution of the character using the heuristics described in Section 2.2. It took between 8.4 seconds (at two feet) to 27.6 seconds (at ten feet) to capture the image used for *Character Recognition*. When the character is farther away from the robot, more positioning commands for the camera are required, which necessarily takes more time. When the robot is moving, the robot stops around 4 to 5 feet of the character, taking around 15 seconds to capture an image. For distances of more than 10 feet, *Character Recognition* was not possible. The height of the bounding box before scaling is approximately 130 pixels. The approach can be made faster by taking the image with only the minimal height for adequate recognition performance. This is close to 54 pixels. The capture time then varied from 5.5 seconds at 2 feet to 16.2 seconds at 10 feet.

Another set of tests consisted of placing the robot in an enclosed area where many characters with different background colors (orange, blue, and pink) were placed at specific positions. Two lighting conditions were used in these tests: standard (fluorescent illumination) and low (spotlights embedded in the ceiling). For each color and illumination condition, 25 images of each of the 25 characters were taken. Table 1 presents the recognition rates according to the background color of the characters and the illumination conditions. Letting the robot move freely for around half an hour in the pen, for each of the background color, the robot tried

Table 1: Recognition performances in different lighting conditions.

Background color	Recognized (%)	Unrecognized (%)	Incorrect (%)
Orange (std.)	89.9	5.6	4.5
Blue (std.)	88.3	5.4	6.3
Pink (std.)	89.5	8.0	2.5
Orange (low)	93.2	4.7	2.1
Blue (low)	94.7	3.1	2.2
Pink (low)	91.5	5.3	3.2

to identify as many characters as possible. Recognition rates were evaluated manually from HTML reports containing all of the images captured by the robot during a test, along with the identification of the recognized characters. A character is not recognized when all of the outputs of the neural system have an activation value less than 0.8. Overall, results show that the average recognition performance is 91.2%, with 5.4% of unrecognized character and 3.6% of false recognition, under high and low illumination conditions. This is very good considering that the robot can encounter a character from any angle and at various distances. Recognition performances vary slightly with the background color. Incorrect recognition and character unrecognized were mostly due to the robot not being well positioned in front of the characters: the angle of view was too big and caused too much distortion. Since the black blob of the characters does not completely absorb white light (the printed part of the character creates a shining surface), reflections may segment the character into two or more components. In that case, the positioning algorithm uses the biggest black blob that only represents part of the character, which is either unrecognized or incorrectly recognized as another character. That is also why performances in low illumination conditions are better than in standard illumination, since reflections are minimized.

Table 2 presents the recognition performance for each character with the three background colors, under both standard and low illumination conditions. Characters with small recognition performance (such as 0, 9, W, and L) are usually not recognized without being confused with other characters. This is caused by limitations in the color segmentation. Confusion however occurs between characters such as 3 and 8.

We also tested discrete cosinus transform for encoding the input images before sending them to a neural network and see if performance could be improved. Even though the best neural network topology required only 7 hidden neurons, the performance of the network in various illumination conditions was worse than that with direct scaling of the character in a 13×9 window [16].

Finally, we used the approach with our entry to the AAAI 2000 Mobile Robot Challenge [17], making a robot attend the National Conference on Artificial Intelligence (AI). There were windows in various places in the convention center,

Table 2: Recognition performance for each character with the three background colors, in standard and low illumination conditions, in Phase 1.

Character	Standard	Low
0	74.7	93.3
1	85.3	90.7
2	94.7	96.0
3	73.3	89.3
4	88.7	89.3
5	96.0	98.7
6	98.6	93.3
7	96.0	86.3
8	86.7	96.0
9	60.0	94.7
A	86.7	94.5
C	100	100
E	89.3	96.0
H	87.5	77.0
J	98.7	94.7
L	88.0	90.7
N	74.3	82.4
S	95.9	100
V	90.7	93.2
W	84.7	88.0
Arrow up	98.7	98.7
Arrow down	100	100
Arrow left	89.3	90.7
Arrow right	93.3	94.6
Charge	98.7	100

and some areas had very low lighting (and so we sometimes had to slightly change the vertical angle of the characters). Our entry was able to identify characters correctly in such real-life settings, with identification performance of around 83%, with no character incorrectly identified.

4.2. Phase 2

In this phase, the inputs of the neural networks are taken from a scaled image of the bounding box of the character to process, this time 13×13 pixels large. We used four messages to derive our training and testing sets. The messages are shown in Figure 8 and have all of the characters and numbers of the set. Thirty images of these four messages were taken by the robot, allowing to generate a data set of 1290 characters. The experiments are done in the normal fluorescent lighting conditions of our laboratory.

We again conducted several tests with different number of hidden units and by adding three additional inputs to the network (the horizontal center of gravity (x_c), vertical center of gravity (y_c), and the height/width ratio). The best results were obtained with the use of the three additional inputs and seven hidden units. The network has an overall success rate of 93.1%, with 4.0% being of unrecognized character and 2.9% of false recognition. The characters extracted by the

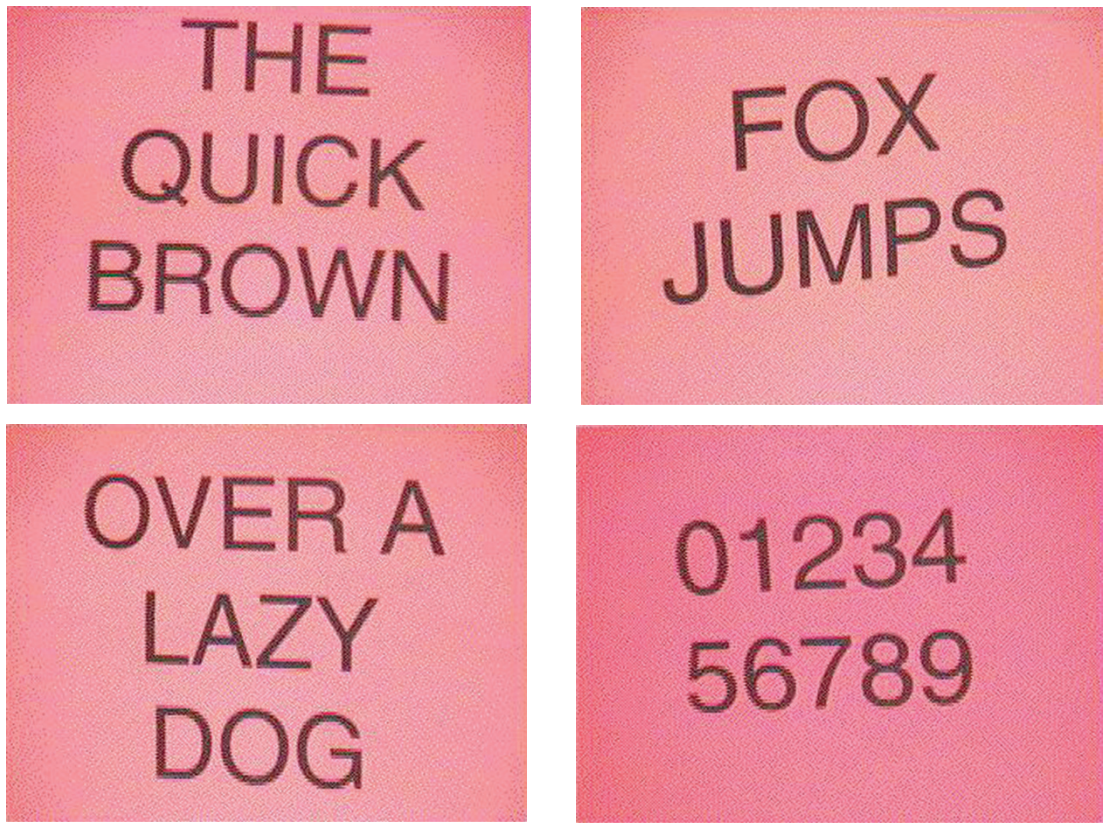


Figure 8: Messages used for training and testing the neural networks in Phase 2.

Image Segmentation module are about 40 pixels high. Table 3 presents the recognition performance for each of the characters. Note that using Arial font does not make the recognition task easy for the neural network: all characters have a spherical shape, and the O is identical to the 0. In the *False* column, the characters falsely recognized are presented between parenthesis. Recognition rates are again affected by the viewpoint of the robot: when the robot is not directly in front of the message, characters are somewhat distorted. We observed that characters are well recognized in the range $\pm 45°$.

To validate the approach for word recognition, we used messages like the ones shown in Figures 5 and 8 and the ones in Figure 9 as testing cases. These last messages were chosen in order to see how the robot would perform with letters that were difficult to recognize (more specifically J, P, S, U, and X). The robot took from 30 to 38 images of these messages, from different angles and ranges.

Table 4 shows the recognition performance of the different words recognized by the robot. The average recognition rate is 84.1%. Difficult words to read are SERVICE, PROJECT, and JUMPS because of erroneous recognition or unrecognized characters. With PROJECT however, the most frequent problem observed was caused by wrong word separation. Using a dictionary of 30 thousands words, performance reaches 97.1% without visible time delay for the additional process.

5. RELATED WORK

To our knowledge, making autonomous mobile robots capable of reading characters in messages placed anywhere in the world is something that has not been frequently addressed. Adorni et al. [18] use characters (surrounded by a shape) with a map to confirm localization. But their approach uses shapes to detect a character, black and white images, and no zoom. Dulimarta and Jain [2] present an approach for making a robot recognize door numbers on plates. The robot is programmed to move in the middle of a corridor, with a black-and-white camera with no zoom, facing the side to gather images of door-number plates. Contours are used to detect plates. An algorithm is used to avoid multiple detection of the same plate as the robot moves. Digits on the plate are localized using knowledge about their positions on the plates. Recognition is done using template-matching from a set of stored binary images of door-number plates. Liu et al. [3] propose a feature-based approach (using aspect ratios, alignment, contrast, spatial frequency) to extract potential Japanese characters on signboards. The robot is programmed to look for signboards at junctions of the corridor. The black-and-white camera is fixed with no zoom. Rectification of the perspective projection of the image is required before doing *Character Recognition* (the technique used is not described). In our case, our approach allows the robot to find messages anywhere in the world based on knowledge of color composition of the messages. The pan, the tilt, and the zoom of

Table 3: Recognition performance of the *Character Recognition* module in Phase 2.

Character	Recognized	Unrecognized	False
1	96.7	3.3	0
2	93.3	0	6.7 (1)
3	100	0	0
4	86.7	6.7	6.6 (F, T)
5	83.3	16.7	0
6	60	30	10 (3, C)
7	100	0	0
8	56.7	6.6	36.7 (P)
9	86.7	3.3	10 (3, P)
A	98.3	0	1.7 (F)
B	96.7	3.3	0
C	100	0	0
D	100	0	0
E	98.3	0	1.7
F	100	0	0
G	96.6	3.4	0
H	100	0	0
I	96.7	0	3.3 (V)
J	78.9	18.4	2.6 (G)
K	100	0	0
L	100	0	0
M	94.7	5.2	0
N	100	0	0
O	98.7	1.3	0
P	89.4	7.9	2.6 (R)
Q	100	0	0
R	100	0	0
S	81.6	18.4	0
T	100	0	0
U	89.7	5.9	4.4 (O)
V	96.6	3.4	0
W	100	0	0
X	86.8	5.3	7.8 (F, I, N)
Y	93.1	0	6.9 (6)
Z	100	0	0

the camera are used to localize messages independently of the motion of the robot (i.e., limited by the motion and the zoom capabilities of the camera).

The most similar approach to ours is from Tomono and Yuta [5], who present a model-based object recognition of room-number plates. Using a color camera with no zoom, the system first has to recognize doors (based on shape and color), then plates on the wall near the doors. An algorithm for rectifying the perspective projection is used before *Character Recognition*. The character set is A to Z, a to z, 0 to 9, with a resolution of 50×50 pixels. No indications are provided on the *Character Recognition* algorithm. The performance on *Character Recognition* is 80%. Combined with the performance of recognizing doors and locating plates, the overall performance of the system drops to 41.3%. With our approach, color-based identification of messages and the use of the zoom do not require the added complexity of recognizing doors or the use of particular knowledge about the world to locate messages.

6. CONCLUSION AND FUTURE WORK

This work demonstrates that it is feasible for mobile robots to read messages autonomously, using characters printed on colored sheets and a neural network trained to identify characters in different lighting conditions. Making mobile robots read textual messages in uncontrolled conditions, without a priori information about the world, is surely a challenging task. Our long-term objective is to improve the techniques proposed in the work by addressing the following.

(i) *Robot control*. There are several potential improvements to improve the PTZ capabilities (speed, precision, zoom) of the camera. These would allow the robot to read messages from top to bottom and from left to right when they do not fit in the field of view of the camera. In the future, we would like to avoid having to stop the robot from moving while reading. This could be done by successively sending images of a message at different zoom values (affected by the motion of the robot as perceived using the wheel encoders) until a successful recognition is observed.

(ii) *Image processing*. Our goal is to allow the robot to read messages on various types of background without the need to specify the background and foreground colors. This would likely require an increase in resolution of the grabbed image, as well as some image filtering to deal with character distortion and illumination variations. Also, developing a multi-image character tracking behavior that scans the environment would be an interesting project. This behavior could likely work like the human eye, reading sentences in an ordered manner and remember which words were part of the sentence, instead of multiple words and sentences at the same time. Other thresholding algorithms for image binarization (such as [19] that allows to select the optimal threshold for binarization maximizing the variance, or others given in [20]), for character segmentation (see [21]), word extraction, and optical *Character Recognition* could be used. It would be interesting to see which ones would provide the appropriate processing for a given robot and sets of messages to read.

(iii) *Message understanding*. Including natural language processing techniques to extract and understand textual messages meaning would provide useful information about the messages perceived. A language model based on N-grams could help improving sentence recognition rate.

Integration of all the techniques mentioned above are the key ingredients to having mobile robots of different shapes and sizes successfully accomplish useful tasks and interact in human settings. The approach and methodology described

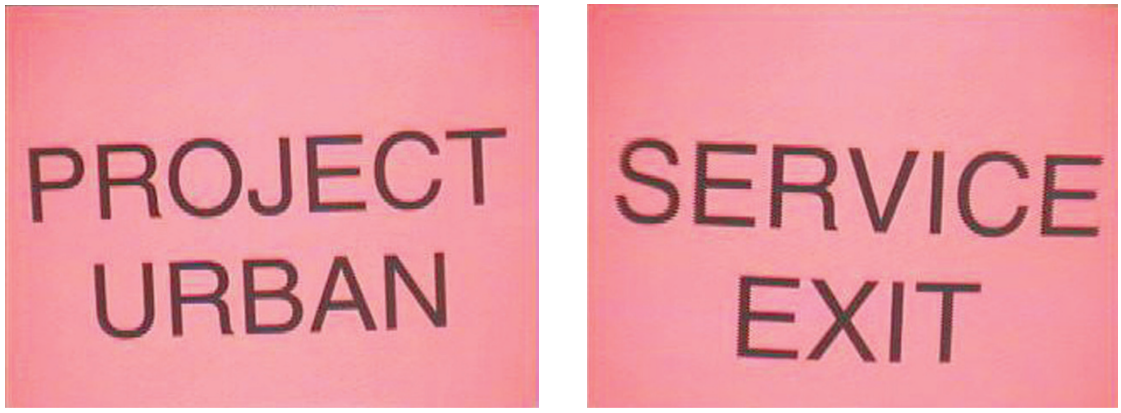


Figure 9: Validation messages.

Table 4: Recognition performance of the *Character Recognition* module in Phase 2.

Word	Recognized (%)	Problems	Dictionary (%)
THE	100	—	100
QUICK	93.3	Either U or C not recognized	100
BROWN	96.7	B not recognized	100
FOX	86.8	X recognized as I or N, or not recognized	97.4
JUMPS	57.9	Either J or P not recognized	89.5
OVER	90	E recognized as F, or R recognized as 4	96.7
A	100	—	100
LAZY	86.7	Y recognized as 6	100
DOG	93.3	G not recognized	100
PROJECT	60	T recognized as 4; R recognized as P; wrong word separation PR OJECT or PROJEC T	83.3
URBAN	70	B recognized as R, or not recognized; N recognized as H	96.7
SERVICE	38.7	V recognized as 6, 7, or Y, or not recognized; C not recognized	100
EXIT	100	—	100
TU	100	—	100
ES	86.7	S not recognized	90
UN	96.7	U not recognized	96.7
ROBOT	73.3	B recognized as R or 8, or not recognized	100

in this paper provide a good starting point to do so. With the processing power of mobile robots increasing rapidly, this goal is surely attainable. We plan to work on these improvements in continuation of our work on the AAAI Mobile Robot Challenge by combining message recognition with a SLAM approach, improving the intelligence manifested by autonomous mobile robots.

ACKNOWLEDGMENTS

François Michaud holds the Canada Research Chair (CRC) in Mobile Robotics and Autonomous Intelligent Systems. This is supported by CRC, the Natural Sciences and Engineering Research Council of Canada (NSERC), the Canadian Foundation for Innovation (CFI), and the Fonds pour la Formation de Chercheurs et l'Aide à la Recherche (FCAR), Québec. Special thanks to Catherine Proulx and Yannick Brosseau for their help in this work.

REFERENCES

[1] S. Thrun, W. Burgard, and D. Fox, "A real-time algorithm for mobile robot mapping with applications to multi-robot and 3D mapping," in *Proc. IEEE International Conference on Robotics and Automation (ICRA '00)*, vol. 1, pp. 321–328, San Francisco, Calif, USA, April 2000.

[2] H. S. Dulimarta and A. K. Jain, "Mobile robot localization in indoor environment," *Pattern Recognition*, vol. 30, no. 1, pp. 99–111, 1997.

[3] Y. Liu, T. Tanaka, T. Yamamura, and N. Ohnishi, "Character-based mobile robot navigation," in *Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '99)*, vol. 2, pp. 610–616, Kyongju, South Korea, October 1999.

[4] M. Mata, J. M. Armingol, A. de la Escalera, and M. A. Salichs, "Mobile robot navigation based on visual landmarks recognition," in *Proc. 4th IFAC Symposium on Intelligent Autonomous Vehicles (IAV '01)*, Sapporo, Japan, September 2001.

[5] M. Tomono and S. Yuta, "Mobile robot navigation in indoor environments using object and character recognition," in *Proc. IEEE International Conference on Robotics and*

Automation (ICRA '00), vol. 1, pp. 313–320, San Francisco, Calif, USA, April 2000.
[6] C. C. Tappert, C. Y. Suen, and T. Wakahara, "The state of the art in online handwriting recognition," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 12, no. 8, pp. 787–808, 1990.
[7] R. Hecht-Nielsen, *Neurocomputing*, Addison-Wesley, Boston, Mass, USA, 1989.
[8] S.-L. Chang, L.-S. Chen, Y.-C. Chung, and S.-W. Chen, "Automatic license plate recognition," *IEEE Transactions on Intelligent Transportation Systems*, vol. 5, no. 1, pp. 42–53, 2004.
[9] R. A. Brooks, "A robust layered control system for a mobile robot," *IEEE Journal of Robotics and Automation*, vol. 2, no. 1, pp. 14–23, 1986.
[10] J. Bruce, T. Balch, and M. Veloso, "Fast color image segmentation using commodity hardware," in *Workshop on Interactive Robotics and Entertainment (WIRE '00)*, pp. 11–16, Pittsburgh, Pa, USA, April–May 2000.
[11] K. Ogata, *Modern Control Engineering*, Prentice Hall, Upper Saddle River, NJ, USA, 1990.
[12] F. Michaud and D. Létourneau, "Mobile robot that can read symbols," in *Proc. IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA '01)*, pp. 338–343, Banff, Canada, July–August 2001.
[13] H. Demuth and M. Beale, *Matlab Neural Network Toolbox*, The MathWorks, Natick, Mass, USA, 1994.
[14] R. A. Jacobs, "Increased rates of convergence through learning rate adaptation," *Neural Networks*, vol. 1, pp. 295–307, 1988.
[15] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, Wiley-Interscience, New York, NY, USA, 2001.
[16] D. Létourneau, "Interprétation visuelle de symboles par un robot mobile," M.S. thesis, Department of Electrical Engineering and Computer Engineering, Université de Sherbrooke, Québec, Canada, 2001.
[17] F. Michaud, J. Audet, D. Létourneau, L. Lussier, C. Théberge-Turmel, and S. Caron, "Experiences with an autonomous robot attending the AAAI," *IEEE Intelligent Systems*, vol. 16, no. 5, pp. 23–29, 2001.
[18] G. Adorni, G. Destri, M. Mordonini, and F. Zanichelli, "Robot self-localization by means of vision," in *Proc. 1st Euromicro Workshop on Advanced Mobile Robots (EUROBOT '96)*, pp. 160–165, Kaiserslautern, Germany, October 1996.
[19] N. Otsu, "A threshold selection method for grey-level histograms," *IEEE Trans. Systems, Man, and Cybernetics*, vol. 9, no. 1, pp. 62–66, 1979.
[20] M. Sezgin and B. Sankur, "Survey over image thresholding techniques and quantitative performance evaluation," *Journal of Electronic Imaging*, vol. 13, no. 1, pp. 146–165, 2004.
[21] H. Bunke and P. Wang, *Handbook of Character Recognition and Document Image Analysis*, World Scientific, Singapore, 1997.

Dominic Létourneau has a Bachelor's degree in computer engineering and a Master's degree in electrical engineering from the Université de Sherbrooke. Since 2001, he is a research engineer at the LABORIUS, a research laboratory on mobile robotics and intelligent systems. His research interests cover combination of systems and intelligent capabilities to increase the usability of autonomous mobile robots in the real world. His expertise lies in artificial vision, mobile robotics, robot programming, and integrated design. He is a Member of OIQ (Ordre des ingénieurs du Québec).

François Michaud is the Canada Research Chairholder in autonomous mobile robots and intelligent systems, and an Associate Professor at the Department of Electrical Engineering and Computer Engineering, the Université de Sherbrooke. He is the Principal Investigator of LABORIUS, a research laboratory on mobile robotics and intelligent systems working on applying AI methodologies in the design of intelligent autonomous systems that can assist humans in everyday lives. His research interests are in architectural methodologies for intelligent decision making, autonomous mobile robotics, social robotics, robot learning, and intelligent systems. He received his Bachelor's degree, Master's degree, and Ph.D. degree in electrical engineering from the Université de Sherbrooke. He is a Member of IEEE, AAAI, and OIQ.

Jean-Marc Valin has a Bachelor's and a Master's degree in electrical engineering from the Université de Sherbrooke. Since 2002, he is pursuing a Ph.D. at the LABORIUS, a research laboratory on mobile robotics and intelligent systems. His research focuses on bringing hearing capabilities to a mobile robotics platform, including sound source localization and separation. His other research interests cover speech coding as well as speech recognition. He is a Member of the IEEE Signal Processing Society and OIQ.