Textual Message Read by a Mobile Robot

Dominic Létourneau[1], François Michaud[1], Jean-Marc Valin[1], Catherine Proulx[1]

[1] *Université de Sherbrooke, Sherbrooke (Québec Canada), laborius@gel.usherb.ca*

Abstract

Giving the ability to read characters and symbols is highly desirable for increased autonomy of mobile robots operating in the real world. The idea is fairly simple: give a robot the ability to acquire an image of a message to read, extract the symbols and recognize them. Image character recognition research has been going on for decades now, with good results. But compared to conventional character recognition systems, the challenge with a mobile robot is to find a textual message to capture in the world and to get a good view of the message, knowing that the viewpoint of the robot depends on its position in relation to the message, which cannot be pre-specified. In this paper we present our approach making it possible for an autonomous mobile robot to read messages. 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 pan-tilt-zoom camera.

1 Introduction

It has been demonstrated that mobile robots can do simultaneous localization and mapping (SLAM) to localize themselves in an indoor environment using a probabilistic representation of the world [9]. This is an ability that us humans do not exploit, because even if we may use maps, we also exploit a lot of written signs and symbols to help us navigate in our cities, office buildings and so on. Just think about road signs, door numbers, exit signs, arrows to give directions, etc. We use maps to give us a general idea of the directions to take to go somewhere, but we still rely on signs to confirm our location in the world. This is especially true in dynamic and large open areas, or in environments that cannot be mapped a priori. The ability to read symbols, signs and messages would undoubtedly be a nice complement to robots that use maps for navigation.

With all the research that has been going on in the area of optical character recognition [8], it is surely possible to make a mobile robot read symbols and signs. But contrarily to conventional character recognition system, a mobile robot has to find a textual message to capture, try to compensate for its viewpoint of the message, and use limited processing capabilities to decode the message. The robot also has to deal with non-uniform illumination conditions in the world. In this work, our goal is not to develop new character recognition techniques, but to address the different aspects of the character recognition process to be incorporated into the higher level intelligence modules of a mobile robotic platform.

We have recently demonstrated that a mobile robot is capable of reading symbols and deriving useful information that can affect its decision-making process [7]. Our approach integrates techniques for : 1) perceiving symbols using color segmentation, 2) positioning and capturing an image with sufficient resolution using behavior-producing modules and a PID controller for a Pan-Tilt-Zoom (PTZ) camera, 3) exploiting simple heuristics to select image regions that could contain symbols, and 4) recognizing symbols using a neural network. The objective of this work is to extend this capability so that the mobile robot can read words instead of only symbols. Dulimarta and Jain [4] also address the problem of making a robot recognize messages. However, their approach is based on pre-determined templates to be recognized, with a precise camera position and no zoom, while our approach aims at reading any messages written using the appropriate font and a pan-tilt-zoom camera.

The paper is organized as follows. Section 2 describes the experimental setup, i.e., the mobile robot, its camera and how it is controlled to capture images of messages to read. Section 3 presents how messages are processed by our system, followed in Section 4 by some results.

2 Experimental Setup

The robot used in the experiments is a Pioneer 2 DX robot with 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 len, pan range of $\pm 90°$ (at a maximum speed of $80°$/sec),

and a tilt range of $\pm 30°$ (at a maximum speed of $50°$/sec). 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 when zooming on an object from the same position, brightness of the image is automatically adjusted. 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 second. However, commands and data exchanged between the onboard computer and the robot controller are set at 10 Hz. Note that all processing for controlling the robot and recognizing symbols is done on the onboard computer, so the decision processes of the robot must be optimized as much as possible. RobotFlow[1] is the programming environment used.

In our previous work [7], our approach was designed to recognize one symbol at a time (i.e., by having only one symbol on a sheet of paper), with each symbol made of one segment. Also, each symbol was placed perpendicular to the floor on flat surfaces, at the same height of the robot. The symbol recognition technique was done in three steps: 1) Symbol perception: using an algorithm for color segmentation in RGB format, symbol perception was done by looking for a black blob completely surrounded by an colored (orange, pink or blue) background; 2) Positioning and image capture: behavior-producing modules were used for positioning the robot and controlling the PTZ camera, and for capturing an image of the symbol to recognize with sufficient resolution; 3) Symbol identification: a backpropagation neural network with 15 hidden neurons was trained and used to recognize 25 symbols using Times font. The approach had a recognition performance of 91%, with 5.4% unrecognized symbol of and 3.6% of false recognition, under high and low illumination conditions. The height resolution of the symbols was approximately 130 pixels.

The idea now is to start from this approach and adapt it to allow the interpretation of messages (words and sentences). The extended approach uses a simple line and word extraction procedure based on character's center of gravity position, width and height. To do so, we still assume that the messages will be composed of characters using a prespecified font, printed in black on a 8.5×11 sheet of paper (colored or white, specified as a parameter in the algorithm). We expanded the character set to 26 capital letters (A to Z) and 10 digits (0 to 9). The experiments are done in the normal fluorescent lighting conditions of our laboratory.

The approach consists of making the robot move autonomously in the world, detect a potential message based on color, 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. It consists of four behavior-producing modules arbitrated using subsumption[1]. These behaviors control the velocity and the heading of the robot, and also generate the pan-tilt-zoom (PTZ) commands to the camera. These behaviors implemented are: *Safe-Velocity* to make the robot move forward without colliding with an object (using sonars); *Character-Tracking* to track a message composed of black regions over a colored or white background; *Direct-Commands* changes the position of the robot according to specific commands generated by the *Message Processing Module*; and *Avoid*, the behavior with the highest priority, moves the robot away from nearby obstacles based on front sonar readings.

The *Character-Tracking* behavior is an important element of the approach because it provides the appropriate pan-tilt-zoom commands to get the maximum resolution of the message to identify. Using the RGB15 color format, our algorithm stores in a 2^{15} lookup table the membership of a pixel to one of 32 color channels. Membership values for a color channel are stored for a given index (RGB15 value) using a single bit position in an unsigned integer of 32 bits. The robot is programmed to move in the environement until it sees with its camera black regions, presumably characters, surrounded by a colored or white area. The *Character-Tracking* behavior makes the robot stop and 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. PID controllers are used for the pan and the tilt commands. More details on our color segmentation algorithm are provided in [7]. A simple heuristic is used to position the zoom of the camera to maximize the resolution of 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 of the black region of the image is within 15 pixels of the borders. If the resolution is not good enough, the *Message Processing Module* gives command to the *Direct-Commands* to make the robot move closer to the message, and the process is repeated. It takes approximately from 5 to 16 seconds over a range of 2 to 10 feet for the robot to position itself and take an image with sufficient resolution for message recognition.

[1] http://robotflow.sourceforge.net

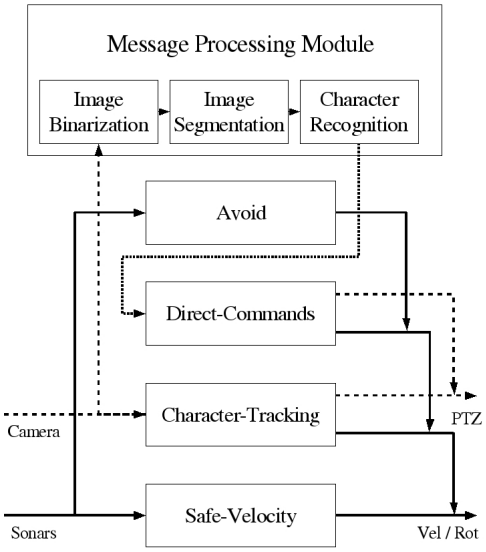

Figure 1: *Architecture diagram of the approach.*

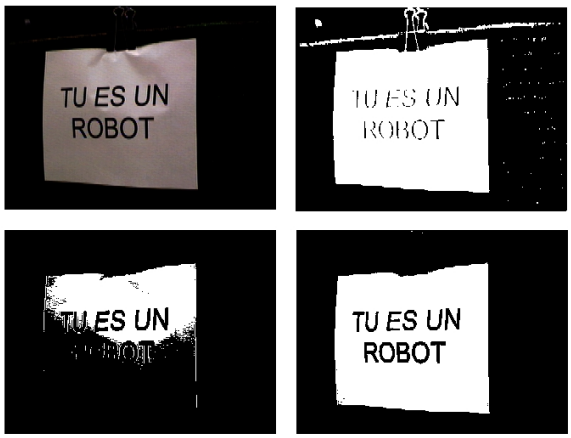

Figure 2: *Effects of thresholds on binarization: (top left) original image; (top right) large threshold; (bottom left) small threshold; (bottom right) proper threshold.*

3 Message Processing Module

With the image obtained by the *Character-Tracking* behavior, the *Message Processing Module* must find the lines, the words and the symbols in the message. This process is done in three steps: Image Binarization, Image Segmentation and Image Character Recognition.

3.1 Image Binarization

Once the proper color image is taken from the zoomed-in characters, the *Message Processing Module* can now begin the character recognition procedure. The first step 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 2 shows the effect of different thresholds for the binarization of the same image.

In the first version of our binarization algorithm, thresholds were hard-coded and results were unsatisfactory since the algorithm did not take into consideration variations in the lighting conditions. We changed our algorithm to adapt the threshold automatically using the following procedure:

1. Intensity of each pixel of the image is calculated using the average intensity of the three color channels (red, green, blue). 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. 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 to compute the threshold offers good performances without requiring too much calculations.

3. Binarization is then performed on the whole image converting pixels into binary values. Pixels with intensity higher or equal than 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 [2, 7]. 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 pixel connected with each other are then delimited by a 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 co-

ordinates of the bounding box. Figure 3 shows the results of this process.

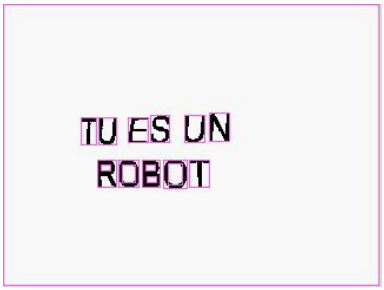

Figure 3: *Results of the segmentation of black areas.*

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 :

- In our experiments, minimum height is set to 40 pixels. No maximum height is specified.
- The vertical center of gravity (y_{c1}) must be inside the vertical line boundaries. Line boundaries are found using the following algorithm: y_{c1} and the height h_{c1} of the first upper left character sets the center of the first line; if y_{ci} of the other characters are within $y_{c1} \pm (h_{c1}/2 + K)$, with K being a constant empirically set to $0.5 \cdot h_{c1}$ (creating a range equal to twice its height), then the character i belongs to the first line; otherwise a new line is created. 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 can usually overcome this problem.

With the characters localized and grouped into lines, they can be grouped into words by using a similar algorithm but at the vertical instead of at the horizontal: 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

From the first line to line to the last, word by word, the part of the image for each character is sent to a neural network for recognition. A feedforward network with one hidden layer is used, trained with the delta-bar-delta [5] 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 black pixel) and -1 (for white pixel).

Using Arial font, we used four messages to derive our training and testing sets. The messages are shown in Figure 4 and has all of the characters and numbers. Thirty images of these four messages were taken by the robot, allowing to generate a data set of 1290 symbols.

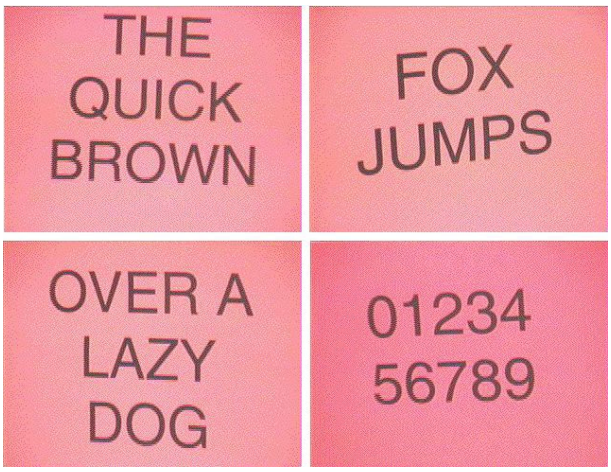

Figure 4: *Messages used for training and testing the neural network.*

The inputs of the network is a scaled image, 13 pixels $\times$ 13 pixels, of the bounding box of the character to process. We did 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). A character is considered recognized when the output neuron associated with this character has the maximum activation value greater to 0.8. Training is done over 5000 epochs.

The best results were obtained with the use of the three additional inputs, with 7 hidden units. The network has a overall success rate of 93.1%, with 4.0% of unrecognized character and 2.9% of false recognition. The characters extracted by the *Image Segmentation* module are about 40 pixels high. Table 1 presents the recognition performance for each of the symbols. 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 parenthe-

sis. Recognition rates are 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 $\pm 45°$.

Table 1: *Recognition performance of the Character Recognition module*

Symbol	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

4 Results

To validate the approach, the experiments consist in making the robot read different messages like the ones shown in Figures 2 and 4, and the ones in Figure 5. 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). Figure 6 shows a picture of the experimental setup. The robot took from 30 to 38 images of these messages, from different angles and ranges.

Table 2 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

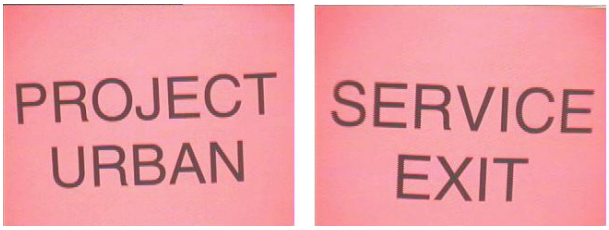

Figure 5: *Validation messages.*

Figure 6: *Experimental setup.*

PROJECT however, the most frequent problem observed was caused by wrong word separation.

Performance can be easily improved by the addition of a dictionary and by changing the way we interpret the outputs of the neural network. Once the activation values transposed to the $[0, 1]$ interval, it can be shown that the neural network outputs are a good approximation of the probabilities $P_{w,k}(w[k])$, the probability of the symbol at position k in the word w. This is caused by the mean square minimization criterion used during the training of the neural network [3]. For a given word w in the dictionary, the probability that $\mathbf{X}$, the observation of characters recognized, for the word w is given by the product of the individual probabilities of each symbol in the word:

$$P(\mathbf{X}|w) = \prod_{k=1}^{N} P_{w,k}(w[k]). \qquad (1)$$

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

Table 2: *Recognition performance of the Message Processing module*

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

$$W = \underset{w}{\text{argmax}} \quad P(\mathbf{X}|w) \tag{2}$$

For the messages used in our experiments, the overall performance of our method using a dictionary is 97.1%.

5 Conclusion and Future Work

This work demonstrates that it is possible for mobile robots to read messages autonomously, using characters printed on colored sheet and a neural network trained to identify characters in different conditions to take into consideration the various viewpoints possible. This is surely a nice starting point to increase the reading capabilities of mobile robots. Potential improvements would be to increase the resolution and zoom capabilities of the camera, to make the robot read messages from top to bottom and from left to right, to look for messages on various types of backgrounds and invariant to colors and lighting. We plan to work on these improvements in continuation of our work on the challenge of making an autonomous mobile robot attend a conference [6].

Acknowledgments

François Michaud holds the Canada Research Chair (CRC) in Mobile Robotics and Autonomous Intelligent Systems. This research is supported financially by the CRC Program, 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) of Québec. Special thanks to Yannick Brosseau for his help in this work.

References

[1] R. A. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, RA-2(1):14–23, 1986.

[2] J. Bruce, T. Balch, and M. Veloso. Fast color image segmentation using commodity hardware. In *Workshop on Interactive Robotics and Entertainment*, 2000.

[3] R. O. Duda, P. E. Hard, and D. G. Stork. *Pattern Classification*. Wiley-Interscience, Second Edition, 2001.

[4] H. S. Dulimarta and A. K. Jain. Mobile robot localization in indoor environment. *Pattern Recognition*, 30(1):99–111, 1997.

[5] R. A. Jacobs. Increased rates of convergence through learning rate adaptation. *Neural Networks*, 1:295–307, 1988.

[6] F. Michaud, J. Audet, D. Létourneau, L. Lussier, C. Théberge-Turmel, and Serge Caron. Experiences with an autonomous robot attending the AAAI conference. *IEEE Intelligent Systems*, 16(5):23–29, 2001.

[7] F. Michaud and D. Letourneau. Mobile robot that can read symbols. In *Proc. IEEE Int. Symp. on Computational Intelligence in Robotics and Automation*, 2001.

[8] C.Y. Suen, C.C. Tappert, and T. Wakahara. The state of the art in on-line handwriting recognition. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 12(8):341–348, 1990.

[9] 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 Int. Conf. on Robotics and Automation*, 2000.