Integrated Object Recognition with Extended Hamming Distance

Vladimir A. Kulyukin

School of Computer Science
DePaul University
Chicago, IL 60604-2301

Abraham Bookstein

Center for Information and Language Studies
The University of Chicago
Chicago, IL 60637

Abstract

We present a model-based object recognition metric integrated with the control system of a mobile robot. The metric draws on the information-theoretic framework of string processing. Object models and images are viewed as sequences of binary strings whose local pairwise similarities contribute to global matches. We present and analyze the results of evaluating the metric on several trash collection tasks.

1. Introduction

Many problems in robotics depend on reliable object recognition [6], [7], [8]. To be practical, object recognition must use metrics that are robust in the presence of sensor noise and account not only for exact matches but also for approximate ones. But, robustness alone is not sufficient for a metric to be used on a mobile robot. If a metric is to be used on a robot, it must be integrated with the robot's control system. This integration, at the very least, presupposes that the metric helps the robot transform raw images into discrete messages about the world used by the control system to make decisions.

Most current object recognition metrics are either model-based or sample-based. Model-based metrics, such as correlation [9], template matching [9], and color histogram matching [11], [3], rely on model libraries. Sample-based metrics, such as Bayesian Networks (BN) [2] and Artificial Neural Networks (ANN) [12], require large pre-classified samples for training. Neither approach accounts for approximate matches in a principled way. Model-based metrics make recognition decisions by adjusting thresholds. Sample-based metrics, while sensitive to approximate matches, cannot explicitly control the degree of approximation. Nor do they offer insights into why the matching happens the way it does, because the reasons are internalized and hidden away from inspection during training. In addition, most metrics from both camps focus exclusively on robustness, bypassing integration altogether or pushing it into future work. An exception is the research reported in Kahn et. al [1996]. The authors present a vision architecture, which, they argue, integrates symbolic and continuous controls. However, no specifics are given on how vision routines are integrated with the robot's control system. It is also difficult to determine how well those routines generalize, because the evaluation is done on a very small image collection.

Thus, there is a practical and theoretical need for metrics that are integrated with the robot's control system, account for approximate matches in a principled way, do not require large pre-classified sample collections to become operational, and perform as well as or better than their model-based or sample-based counterparts. This paper presents a model-based integrated metric that seems to satisfy these criteria. The metric explicitly specifies the degree of approximation it tolerates in matches. The metric's model library is created from the images of the objects to be recognized. The robustness experiments suggest that the metric performs better than some of its counterparts and as well as others.

The metric is implemented and integrated with the control system of a mobile robot collecting trash. The robot patrols an office area and looks for soda cans, coffee cups, and crumpled pieces of paper. When an object is recognized in the image, the robot must pick it up and place it in an area of the floor designated for that type of object. Thus, it is critical that the robot distinguish each type of object.

The paper is organized as follows. In section 2, we discuss the information-theoretic concepts of string processing on which the metric draws. We also introduce the Extended Hamming distance as a similarity measure between binary strings, and offer an algorithm for computing it. Section 3 explains how the object models are constructed and used in matching. In section 4, we outline how the metric is integrated with the robot's control system. Section 5 describes the experiments that compare the Extended Hamming distance metric with color histogram matching and normalized correlation. Section 6 describes future work. Section 7 offers conclusions.

2 Extended Hamming Distance

The Hamming distance has long been used to measure the similarity between two bit vectors of the same dimension [10]. In particular, it was extensively used in the early applications of information theory that focused on measuring the error introduced by the noise in the channel through which messages are sent from the source to the target. The classic, or crisp, Hamming distance (CHD) does not recognize the notion of approximate similarity, because it measures only the *exact* extent to which the corresponding bits in two bit vectors agree.

As an example, consider the target bit vector $A = 1100100000$ and the two source bit vectors $B = 1100010000$ and $C = 1100000001$. As measured by the CHD, both B and C approximate A equally well since both differ from it by a distance of 2. In the case of B, the distance is obtained by deleting the 6th bit, which incurs a cost of 1, and inserting it in the 5th position, which incurs another cost of 1. In the case of C, the 10th bit is deleted and inserted in the 5th position. Intuitively, however, B is a better approximation, because it misses A by only one bit while C misses it by five. In applications with noisy channels, e.g., robot cameras, finding such better approximations is important.

One can view the CHD as an edit distance with the operations of insertion and deletion. Given two bit vectors of the same dimension, the CHD is the minimum number of bit changes required to change one bit vector into the other. The *Extended Hamming distance* (EHD) is also an edit distance, but the set of operations is extended with a shift operation, which has not received much attention in the string literature [5]. By extending the operation set with the shift operation, we account for the notion of approximate matches, i.e, situations when corresponding bits in two bit vectors are considered aligned even when they are not in the exact same positions.

Formally, assume that we wish to measure the distance between a target bit vector T and a source bit vector S. For example, T may be a representation of an object while S may be the output of a vision routine, e.g., edge detection. Ideally, of course, S and T are identical. Now let $B = [b_1, ..., b_D]$ be a bit vector of dimension D and let $N(B)$ be the number of 1-bits in B. For example, if $B = 1001$, $N(B) = 2$. To compute the EHD, we associate a cost with each of the three operations. Given the costs, we compute the EHD by transforming S into T and adding up the costs of the operations used in the transformation. To eliminate the ambiguity associated with multiple transformation sequences, we quantify the EHD as the minimum cost of the sequences of operations that transform S into T.

In the string literature, the computation of edit distances typically requires processing whole strings. However, since the EHD depends only on 1-bits, bit vectors are represented as lists of indices of their 1-bits given in ascending order. If B is a bit vector, let $I(B) =< s_1, s_2, ..., s_{N(B)} >$ be an index list, where s_i is the position of the i-th 1-bit. For example, if $B = 1010$, $I(B) =< 1, 3 >$. The indices start with 1, because 0 is reserved for the dynamic programming algorithm to compute the EHD presented below. Let $c(i, j)$ denote the minimum cost of transforming the first i 1-bits of S into the first j 1-bits of T, where $1 \leq i \leq N(S)$ and $1 \leq j \leq N(T)$. The objective is to compute $c(N(S), N(T))$. The computation is carried out by filling the entries of an $N(S) + 1$ by $N(T) + 1$ table with the following dynamic programming technique:

- Insertion: S is considered to have missed the j-th 1-bit of T. A 1-bit is inserted into S at location s_j at a cost of $c_I > 0$. Since the insertion is applied to an optimal transformation taking $< s_1, ..., s_i >$ to $< t_1, ..., t_j >$, $c(i, j) = c_I + c(i, j - 1)$.

- Deletion: S is considered to have an extra 1-bit in the i-th position. The 1-bit is changed into a 0-bit at a cost of $c_D > 0$ so that $c(i, j) = c_D + c(i - 1, j)$.

- Shift: The i-th 1-bit of S and the j-th 1-bit of T are considered misaligned. The i-th 1-bit of S is shifted P positions to align it with the j-th bit of T so that $c(i, j) = c_{S(P)} + c(i - 1, j - 1)$. The incurred cost, $c_{S(P)}$, is a non-negative function that monotonically increases with $P = |j - i|$. The function $c_{S(D)}$ can be defined as KP, for some non-negative constant K chosen in such a way that for large values of P it is cheaper to delete and insert than to shift. The shift cost function offers an explicit way to control the degree of approximation in matches.

The table is initialized by inserting the following values in its 0-th row and 0-th column: $c(0, j) = jc_I$ and $c(i, 0) = ic_D$. Intuitively, $c(0, j)$ defines the cost of inserting j 1-bits into S, while $c(i, 0)$ defines the cost of deleting i 1-bits from S. Once the table is initialized, the costs are computed according to the above three rules.

The complexity of the computation is significantly reduced, because the size of the table is $N(S)N(T)$, not D^2, where D is the dimension of the bit vectors. We define the function that computes the EHD as $ehd(S, T, c_I, c_D, K)$. The parameter K defines each instance of the metric in terms of the degree of approximation that it tolerates in matches between the source and the target. If K is sufficiently large, the cost of shifting becomes prohibitively expensive, and the shift operation is never chosen, thus making the EHD and the CHD identical. Hence, the EHD generalizes the CHD. The choice of K depends on how much approximation can be safely tolerated. If good matches occur only between strings whose bits are closely aligned, K

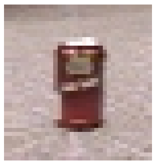

Figure 1: A soda can.

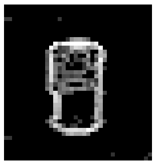

Figure 2: Edges of the soda can.

```
0000000000000000000000000000000
0000000000000000111111000000000
0000000000000111111111110000000
0000000000011111111111110000000
0000000000110111111110011000000
0000000000111111010010111000000
0000000000111011011110111000000
0000000000111110100111110000000
0000000000111110000111111000000
0000000000111111111100110000000
0000000000110111111000110000000
0000000000111111111111110000000
0000000000111111000111110000000
0000000000110111111111110000000
0000000000011111100000111000000
0000000000000110000000001100000
0000000000000110000000001100000
0000000000000110000000001100000
0000000000000110000000001100000
0000000000000110000000001100000
0000000000001110001000111000000
0000000000111000001000111000000
0000000000011110110001110000000
0000000000000111111111110000000
0000000000000001111111000000000
```

Figure 3: Bit array of the soda can.

should be small. In general, the larger the value of K, the greater the degree of approximation allowed.

3 Object Representation and Matching

Objects are represented by model libraries created from images taken from different distances and with different camera tilts. The camera used is the Pioneer Pan-Tilt-Zoom (PTZ) Robotic Camera mounted on a Pioneer 2DX mobile robot manufactured by ActivMedia, Inc. [www.activmedia.com]. Figure 6 depicts the robot used in the experiments. The camera has a horizontal angle of view of 48.8 degrees and a vertical angle of view of 37.6 degrees. The images are saved as 120 by 160 color bitmaps. The red, blue, and green intensities range from 0 to 255. The origin of the image coordinate system is in the bottom left corner of the image. The positive X-axis measures the width. The positive Y-axis measures the height. The distance is measured in meters from the robot's gripper to the object.

The models were taken from the following distances: 0.5, 1.0, 1.5, and 2 meters. The camera tilt for the distances 1.0, 1.5, and 2 was 0 degrees. The camera tilts for 0.5 were 0 and -10 degrees. The negative tilt, which indicates that the camera is tilted 10 degrees downward, was chosen for the distance 0.5, because as the robot approaches an object, the

```
0000111111010000
0001101111011100
1011110111101100
0110111111100110
0111111010110111
```

Figure 4: Bit model.

```
0000001111100000
0001111111111000
0011111111111100
0110111111100110
0111111010010111
```

Figure 5: Bit image.

pickup skill tilts the camera in order to ensure that the object is still present or to correctly identify the object. Each object has two types of models: Hamming distance models (HDMs) and color histogram models (CHMs) [11], [3].

3.1 Hamming distance models

To create an HDM, an object's image is taken from a given distance and with a given camera tilt. The image is convolved with the gradient edge detection mask [9] and turned into a bit array by thresholding pixel intensities to 0 or 1. The object's model is the smallest region of the image containing the object. Thus, HDMs are 2D bit arrays. Figure 1 contains part of an image containing a soda can. Figure 2 contains the image from Figure 1 after the application of the gradient edge detection mask. After the edges are detected, the image is turned into a 2D bit array by thresholding pixel intensities to 0 or 1. Figure 3 contains the bit array obtained from the image from Figure 2. An HDM model is obtained from the 2D bit array by taking the smallest region containing the object. In this case, the region is a rectangle containing all of the bits that consitute the soda can.

Each object has five HDMs: two models for the distance 0.5, one with tilt 0 and one with tilt -10, and three models with tilt 0 for the remaining distances. Each model has two row constraints specifying the region of an image where the model is applied. For example, the row constraints of the two meter pepsican model are 40 and 45. The row constraints are based on the camera's calibration.

To recognize objects in images using HDM models, we convert images into 2D bit arrays. Formally, let $M_H(O, D, T, L, U)$ denote an HDM for object O at distance D and with tilt T and two row constraints L and U for the lower and upper rows, respectively. For example, $M_H(pepsican, 2, 0, 40, 45)$ is the model of a pepsican two meters from the robot and with tilt 0 and row constraints 40 and 45. Let H and W be the model's height and width.

Figure 6: Pionner 2DX robot.

Let I be an image. I is first convolved with the gradient edge detection mask. To recognize O in I, the model $M_H(O,D,T,L,U)$ is matched with the region of I whose bottom row is L and whose top row is $U+H$. For example, if the height of $M_H(pepsican, 2, 0, 40, 45)$ is 13 rows, the matching happens between rows 40 and 58 (45 + 13). The matching is done by moving the model from left to right C columns at a time and from top to bottom R rows at a time. For this reason, R and C are referred to as *model shift parameters*.

To make the model matching more efficient for the distances of 1, 1.5 and 2 meters, the floor line is computed to connect the lowest edge pixel in each column of the image [7]. The model matching happens only around the floor line: 10 rows up and 10 rows down. It is assumed that since all objects are on the floor, they are either part of the floor line, i.e., the lowest edge pixels in several consecutive columns belong to an object, or are next to the floor line. When that assumption does not hold, e.g., an object is more than 10 rows behind a power cord on the floor, the robot does not recognize the object.

The similarity coefficient between the model and an image region covered by the model is the sum of the EHDs between the corresponding bit strings of the model and the image region normalized by the model's size. Formally, let S be the image region covered by $M_H(O,D,T,L,U)$. The similarity between M_H and S, $sim(M_H,S)$, is given by $sim(M_H,S) = 1/HW\sum_{R=0}^{H-1} ehd(M_R, S_R, c_I, c_D, K)$, where M_R and S_R are the two corresponding bit strings, i.e., rows, from M_H and S, respectively. Consider Figure 4 and Figure 5. The former contains the top of an HDM model for a pepsican. The latter contains a region of a 2D array obtained from an image with another pepsican. Once the model top overlaps the region, the similarity between the model top and the region is computed by summing the EHDs between the corresponding horizontal bit strings. In other words, the EHD between the first horizontal pair is added to the EHD between the second horizontal pair, etc. The sum thus obtained is normalized by the product of the model's height and width. Similarity results are 4-tuples $< x, y, m, s >$, where x and y are the coordinates of the bottom left corner of the image region that matches the model m with the similarity score s. Similarity results are sorted in nondecreasing order by similarity scores and the top N matches are taken.

3.2 Color histogram models

The CHMs were created for the same distances and tilts as the HDMs. Thus, each object has five CHMs. An object's image is taken from a given distance and with a given camera tilt. The smallest region of the image containing the object is cropped. Three color histograms are created from the cropped region. The first histogram records the relative frequencies of red intensities. The other two do the same for blue and green. Each histogram has sixteen intensity intervals, i.e., the first is from 0 to 15, the second is from 16 to 31, etc. Thus, each CHM consists of three color histograms. The same row and floor line constraints as for the HDMs hold.

Formally, let $M_C(O,D,T,L,U)$ be the $H \times W$ CHM of object O at distance D and with tilt T and two row constraints L and U. Let I be an image. To recognize O in I with $M_C(O,D,T,L,U)$, the $H \times W$ mask is matched with the appropriate region of the image R. The similarity between M_C and R is the weighted sum of the similarities between their corresponding color histograms. Let $h(H_1,H_2)$ be the similarity between two color histograms H_1 and H_2 computed as the sum of the absolute differences of their relative frequencies so that $0 \leq h(H_1,H_2) \leq Q$, where Q is a suitably chosen constant. Let $RH(M_C)$ denote the model's red histogram, $BH(M_C)$ denote its blue histogram, and $GH(M_C)$ denote its green histogram. Let $RH(R)$ be the red intensity histogram of R, and let $BH(R)$ and $GH(R)$ be the blue and green intensity histograms, respectively. Then $sim(M_C,R) = 1/Q[A \times h(RH(M_C),RH(R)) + B \times h(BH(M_C),BH(R)) + C \times h(GH(M_C),GH(R))]$, where A, B, and C are the weights assessing the relative importance of each color that sum up to 1. In the experiments, $A = .34$, $B = C = .33$. In other words, all colors are equally important. Thus, $0 \leq sim(M_C,R) \leq 1$. This matching metric is different from the standard intersection metric used in color histogram matching [11], because it provides a way to control the relative importance of different colors.

4 Integrating Object Recognition with Robot Control

The object recognition routines are part of the robot's control system. The control system is the bottom tier in the three-tier robot architecture [6], [1], with the top two tiers being the planner and the reactive executor. In addition to the vision routines, the control system includes a set of low level behaviors, e.g., obstacle avoidance, that run on the P2OS operating system for Pioneer robots [www.activmedia.com].

The reactive executor is the Reactive Action Package (RAP) system [6]. RAPs are sets of methods for achieving goals under different circumstances. For example, there are two methods in a RAP for navigating towards the door. One method assumes that the robot knows where the door is. The other method makes no such assumption and instructs the robot to find the door first.

The RAP system consists of the RAP programming language and an interpreter for executing RAPs written in that language. The interpreter maps external goals to methods that achieve them in a context-sensitive manner. For example, when choosing between the two methods in the door navigation RAP, the interpreter queries the RAP memory to find out if the robot knows where the door is. The result of the query determines which RAP gets to execute. The chosen methods become tasks that are refined into commands to enable sensing and action.

These commands enable and disable low-level behaviors called skills. When necessary, skills invoke P2OS primitives to interface with the robot hardware, e.g., read sonars, set rotational and translational velocities, open and close the gripper, etc. Only the skills make assertions in the robot's memory, which ensures that the memory is synchronized with the world as much as possible.

The finite set of robot skills completely describes the robot's physical abilities. Each skill can be viewed as a process parameterized for input and output. When a skill is enabled its process starts running; when the skill is disabled, the process terminates. For example, `ehd-detect-obj-skill` is enabled when a RAP method enables the EHD visual routine to detect an object. The input parameters allow the enabler of a skill, i.e., a RAP method, to influence the skill's behavior. For example, `ehd-detect-obj-skill` is parameterized for object type, cost of insertion, deletion, and shifting, model shift parameters, and matching threshold.

Conceptually, the output of a skill is a set of messages that the skill can send to the robot's memory. We define the output of a skill as a set of predicate templates that are instantiated with specific values extracted from the sensory data. These predicate templates allow the robot to transform raw sensory data from the world into discrete statements about it. The `ehd-detect-obj-skill` skill has the following predicate templates associated with it: `(ehd-detected-obj <obj-type>)`, `(ehd-dist-to <dist>)`, `(ehd-obj-bottom <x> <y>)`, and `(ehd-sim-score <score> <model-id>)`. For example, when activated to look for a pepsican, the skill may put the following assertions in the memory: `(ehd-detected-obj :pepsican)`, `(ehd-dist-to 1.5)`, `(ehd-obj-bottom 10 43)`, and `(ehd-sim-score .73 :m10)`. These assertions mean that the routine detected a pepsican 1.5 meters away from the robot, and the bottom left coordinates of the image region that had the best matching score of .73 are $X = 10$, $Y = 43$. The second argument of the ehd-sim-score assertion is the ID of a model.

This object recognition scheme is indexical-functional [4], because only the types of objects are recognized, not their identities. One should note that the described skill chooses the best matching model. There is another skill which, when enabled, makes assertions about the top n matching models, where n is a small integer, e.g., 2 or 3. Once these assertions are in the robot's memory, navigation RAPs and skills use them to home on the detected object. It is up to the higher level modules, i.e., RAPs and plans, to decide how to use the assertions made by the visual routines. For example, if the color histogram skill asserts that there is a pepsican .5 meter away from the robot and the EHD skill asserts that there is a coffee cup 2 meters away from the robot, the higher level module instructs the robot to focus on the can, because it is closer. Thus, the robot's memory is a message board through which different modules communicate and keep abreast of each other's activities.

An important objective of an integrated visual skill is to help the robot stay in sync with the world. This objective is met in our implementation by having visual skills run in three steps. First, the skill erases the old assertions it made in the robot's memory during its previous run. Second, the skill does its computation and generates a set of assertions by filling its parameterized predicates with the results of its computation. Finally, the new assertions are put in the robot's memory. One potential disadvantage of this strategy is that the skill does not have access to the history of its own assertions. The history can be helpful, for example, when looking for objects detected before. In our future work, we plan to investigate this issue by adding time stamps to assertions in the robot's memory.

5 Evaluation

Trash collection is a well-known task in AI robotics [6], [8]. In the experiments, the standard trash collection task was extended insomuch as the robot had to recognize types of trash to place them in designated areas.

A typical experimental setup for object recognition tests is to collect images first and then compare different metrics on the static collection. Our experimental setup was different, because our objective was to test not only the robustness of metrics but also our integration mechanism. Towards that end, the evaluation was carried out as the robot was engaged in trash collection. The robot patrolled an office

dist and tilt	Pepsi	RootBeer	Coffee	Paper	Pepsi	RootBeer	Coffee	Paper
$0.5(-10)$	.83 : .74	.79 : .67	.81 : .78	0 : .78	.85 : .76	.8 : .71	.82 : .8	0 : .8
$0.5(0)$	.38 : .78	.46 : .71	.8 : .73	0 : .79	.38 : .80	.46 : .73	.83 : .75	0 : .84
$1.0(0)$	.41 : .80	.12 : .83	.15 : .81	0 : .87	.77 : .81	.65 : .85	.69 : .85	0 : .95
$1.5(0)$	.82 : .84	.86 : .85	.42 : .97	.12 : .95	.96 : .85	.91 : .86	.75 : 1	.74 : 1
$2.0(0)$	.5 : .90	.64 : .87	.25 : .91	.22 : .95	.5 : .91	.64 : .9	.38 : 1	.71 : 1

Figure 7: Crisp Hamming Distance. N=1 and N=5

dist and tilt	Pepsi	RootBeer	Coffee	Paper	Pepsi	RootBeer	Coffee	Paper
$0.5(-10)$	1 : .77	1 : .78	.91 : .77	.18 : .8	1 : .78	1 : .75	1 : .77	.21 : .8
$0.5(0)$	.48 : .78	.75 : .78	.91 : .8	.18 : .8	.58 : .78	.81 : .76	1 : .77	.23 : .81
$1.0(0)$	.51 : .79	.61 : .8	.34 : .81	.15 : .84	.77 : .8	.77 : .81	.69 : 78	.21 : .84
$1.5(0)$	.82 : .81	.85 : .82	.57 : .83	.32 : .85	.96 : .81	.96 : .81	.75 : 79	.83 : .84
$2.0(0)$	.71 : .84	.75 : .85	.61 : .84	.67 : .96	.81 : .85	.82 : .83	.87 : .84	1 : .97

Figure 8: Extended Hamming Distance. N=1 and N=5.

dist and tilt	Pepsi	RootBeer	Coffee	Paper	Pepsi	RootBeer	Coffee	Paper
$0.5(-10)$	.33 : .77	.33 : .76	0 : .81	0 : .82	.37 : .78	.33 : .8	0 : .81	0 : .82
$0.5(0)$	.17 : .78	.17 : .77	0 : .82	0 : .83	.25 : .79	.33 : .81	0 : .82	.33 : .82
$1.0(0)$	.41 : .78	.12 : .78	.1 : .83	0 : .84	.71 : .8	.47 : .82	.31 : .83	0 : .83
$1.5(0)$	.61 : .77	.66 : .77	.25 : .84	.43 : .85	.71 : .81	.71 : .83	.58 : .84	.22 : .84
$2.0(0)$	.71 : .79	.67 : .8	.25 : .85	.21 : .86	.75 : .82	.71 : .84	.25 : .85	.22 : .85

Figure 9: Normalized Correlation. N=1 and N=5.

dist and tilt	Pepsi	RootBeer	Coffee	Paper	Pepsi	RootBeer	Coffee	Paper
$0.5(-10)$	1 : .51	.91 : .54	1 : .54	.91 : .55	1 : .51	.91 : .54	1 : .54	.91 : .55
$0.5(0)$	.86 : .52	.8 : .56	1 : .54	.91 : .57	.86 : .52	.8 : .56	1 : .54	.91 : .57
$1.0(0)$	.4 : .57	0 : .57	0 : .57	0 : .59	.4 : .57	0 : .57	0 : .57	0 : .59
$1.5(0)$	0 : .59	0 : .58	0 : .58	.43 : .59	0 : .59	0 : .58	0 : .58	.43 : .59
$2.0(0)$	0 : .61	0 : .71	0 : .71	0 : .71	0 : .71	0 : .75	0 : .81	.33 : .82

Figure 10: Color Histogram Matching. N=1 and N=5.

area of 10 meters by 15 meters. The robot's server was connected to an off-board Windows NT workstation via a radio modem. A human judge was positioned at the workstation. When a vision routine recognized an object in an image, the area where the object was recognized was whitened and displayed at the judge's workstation along with the information on the recognized object type and the model ID. The robot would pause, pending the judge's evaluation. If the whitened area covered a region with no object or the wrong type of object, the match was incorrect. If the whitened area covered less than half of a correct object, the match was incorrect. When the whitened area covered at least half of a correct object, the judge would measure the distance between the robot's gripper and the object. If the measured distance was within 20 centimeters of the distance predicted by the model that found the best match, the match was evaluated as correct. Thus, correct matches were those where the best matching model covered at least half of a correct object type and the distance was within 20 centimeters of the distance predicted by the model.

The evaluation was done on 182 images of various objects taken from different distances, under different lighting conditions, and with different occlusions. The following metrics were evaluated: the EHD, the CHD, the color histogram metric, and normalized correlation [9]. The normalized correlation similarity between a bit model and the region of a bit image covered by the model is the sum of the products of the corresponding bits normalized by the size of the model.

We used the following values to compute the EHD: $ehd(S, T, c_I = 1, c_D = 1, K = .5)$. The small value of K was chosen because many images taken by the robot contain multiple objects. Thus, the degree of approximation was minimized to avoid false positives as much as possible. This is because if an image contains a single object, a correct model is likely to give a large matching score on the subimage containining the object even if the corresponding bits are misaligned. On the other hand, in images with multiple objects, the value of K should be small, because large values may lead to numerous false positives. The model shift parameters C and R were set to 1. The threshold for all metrics was .6. This threshold was chosen after preliminary experiments, because it gave the best performance for all the compared metrics.

The experiment was to answer the following two questions about the robustness of each metric. First, how likely is it that an object is correctly recognized in an image that contains it? The answer to this question estimates how well a metric detects objects when they are present in images. Second, how likely is it that an object is not recognized in an image that does not contain it? The answer to this question estimates a metric's likelihood of avoiding false positives.

Formally, an image is said to contain an object O at the distance D and with the tilt T if the image is taken when the camera's tilt is T and O is in the image at the distance D from the robot. Let $1 \leq N \leq 5$. Let $REC(O, D, T, N)$ be the event when the metric correctly recognizes O at D and with T within the first N matches. Let $NREC(O, D, T, N)$ be the event when the metric does not recognize O at D and with T within the first N matches. This event occurs when no matches of the object are made. Let $IC(O, D, T)$ be the event when the image contains O at D and with T. Let $NIC(O, D, T)$ be the event when the image does not contain O at D and with T.

The answer to the first question above is approximated with $P_1 = P(REC(O, D, T, N)|IC(O, D, T))$. The answer to the second question above is approximated with $P_2 = P(NREC(O, D, T, N)|NIC(O, D, T)$. An ideal metric has $P_1 = 1$ and $P_2 = 0$. Different metrics can be compared in terms of how close they come to the ideal. We estimated these probabilities by computing the ratio of the numbers of images satisfying the required properties. P_1 was estimated as the ratio L_1/L_2, where L_1 is the number of images that contain O at D and with T and where the metric correctly recognizes O within the first N matches at D and with T, and L_2 is the number of images that contain O at D and with T. P_2 was estimated in a similar fashion.

Figures 7 to 10 contain experimental data. Each cell contains P_1 and P_2 values, in that order, separated by a colon. In each table, columns 1 to 4 present data for $N = 1$, and columns 5 to 8 present data for $N = 5$. Overall, P_1 values for the EHD are best overall, while those for the normalized correlation are worst. The CHD does well on pepsicans and root beer cans, but performs worse on coffee cups and crumpled pieces of paper. The EHD has the same tendency but shows improvement over the CHD, because, unlike the CHD or normalized correlation, the EHD can handle misaligned objects and models. The EHD performs better on images with multiple objects where the object contours are not well defined due to occlusions. The EHD and CHD do equally well on images with single objects where object contours are well delineated. The color histogram metric does well on objects that are close to the robot. However, its performance degrades as the objects get further away from the robot. The EHD tends to be more robust than the color histogram metric in the presence of occlusions and changes in lighting. The color histogram metric did not recognize a number of objects in images where the lighting conditions were different from the lighting conditions of the image from which the models were taken.

The experiments suggest that the EHD performs at least as well as or better than several model-based and sample-based metrics discussed in the literature. For example, Young et al. [1994] present an approach to object recognition based on a multi-layer Hopfield neural network structured as a cascade of several single layer Hopfield networks with links between adjacent layers. The network is evaluated on a set of 51 images of door keys. The success rate on images with single objects is 82 percent and on images with occluded objects is 31 percent. Although Young et al. [1994] do not offer any distance information, both numbers are, on average, below the recognition rates achieved by the EHD.

Boykov and Huttenlocher [1999] present a Bayesian approach to object recognition that explicitly accounts for dependencies between features of the object. The approach is evaluated with Monte Carlo techniques to estimate Receiver Operating Characteristic (ROC) curves that plot the probability of detection along the y-axis and the probability of false alarms along the x-axis. The recognition rates of the EHD are no worse than the rates reported by Boykov and Huttenlocher [1999]. One advantage of the EHD is that it does not require the expensive computation of the a priori probabilities. It is difficult to make further content-based comparisons, because Boykov and Huttenlocher use synthetically generated images of very simple objects in their experiments.

Chang and Krumm [1999] extend the normal color histogram [11] by adding geometric information to it and obtaining the color co-occurrence histogram. The color co-occurrence histogram keeps track of the number of pairs of colored pixels that occur at certain separation distances. The recognition rates of the EHD are slightly better than the recognition rates of the color co-occurrence histograms.

One advantage of the EHDs is that their model libraries are significantly smaller than the model libraries needed for the color co-occurrence histograms. As with Boykov and Huttenlocher's approach, futher content-based comparisions are hard to make, because Chang and Krumm [1999] use cartoon images for evaluation.

The P_2 values in the tables indicate that none of the metrics were good at rejecting false positives. This result suggests that individual cues from separate metrics are fallible and ambiguous. The P_2 values are the lowest for the color histogram metric. However, all of them are above 50 percent, which means that any metric has a better than random probability to return false positives.

6 Future Work

Our future work will focus on two issues. The first issue is the rejection of false positives. While the EHD metric shows good recognition rates, its rejection of false positives needs improvement. In a preliminary attempt to improve the rejection of false positives, we combined the EHD with the color histogram metric. The EHD ran first to return the top 5 matches, i.e., image regions, each of which was subsequently checked with the color histogram metric. While the P_1 values were not significantly different from the P_1 values for the EHD, the P_2 values were, on the average, 30 percent lower. This result suggests that combining metrics is a promising approach for rejecting false positives.

The second issue concerns different computations of the EHD itself. Currently, the EHD is computed by matching horizontal bit strings of models and image regions. But for some objects we may achieve better results by treating their EHD models as sequences of vertical bit strings. We believe that this technique may work well for taller objects, such as doors and trash cans. Another aspect of the EHD computation worth investigating concerns the model shift parameters. Since EHD is well suited for approximate matches, increasing the model shift parameters may well reduce the computational cost of object recognition without decreasing the acceptable recognition and rejection rates.

7 Conclusion

We presented a model-based object recognition metric integrated with the control system of a mobile robot. The metric draws on the information-theoretic framework of string processing. We presented and analyzed the results of evaluating the metric on several trash collection tasks. The metric was found to perform at least as well as or better than several model-based and sample-based object recognition metrics described in the literature.

References

[1] Bonasso, R.P., Firby, R.J., Gat, E., Kortenkamp, D., and Slack, M. "A Proven Three-tiered Architecture for Programming Autonomous Robots," *Journal of Experimental and Theoretical Artificial Intelligence*, Vol. 9, pp. 171-215, 1997.

[2] Boykov, Y. and Huttenlocher, D. "A New Bayesian Framework for Object Recognition," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, IEEE Computer Society, 1999.

[3] Chang, P. and Krumm, J. "Object Recognition with Color Cooccurrence Histograms," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, IEEE Computer Society, 1999.

[4] Chapman, D. *Vision, Instruction, and Action*, MIT Press, New York, 1991.

[5] Crochemore, M. and Wojciech, R. *Text Algorithms*, Oxford University Press, New York, 1994.

[6] Firby, R. J., Prokopowicz, P., and Swain, M. "Collecting Trash: A Test of Purposive Vision," *Proceedings of the Workshop on Vision for Robotics*, International Joint Conference on Artificial Intelligence, 1995.

[7] Kahn, R., Swain, M., Prokopowicz, P., and Firby, R. J. "Gesture Recognition Using the Perseus Architecture," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, IEEE Computer Society, 1996.

[8] Murphy, R. *AI Robotics*, MIT Press, New York, 2000.

[9] Parker, J. R. *Practical Computer Vision Using C*, John Wiley and Sons, New York, 1993.

[10] Roman, S. *Coding and Information Theory*, Springer-Verlag, New York, 1992.

[11] Swain, M. J. and Ballard, D. H. "Color Indexing," *International Journal of Computer Vision*, Vol. 7, pp. 11-32, 1991.

[12] Young, S. S., Scott, P. D., and Nasrabadi, N. "Multilayer Hopfield Neural Network," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recogntion*, IEEE Computer Society, 1994.