Sign recognition

As the ground and aerial robots moved around the maze, they were expected to recognize and respond to various symbols placed at *a priori* unknown locations. These signs included four hazard symbols which the ground robot was to avoid and three bonus symbols which were supposed to function as attractors. Additional start and stop signs marked entrance and egress points for the maze. Figure 1 pictures the nine signs that were to be automatically recognized.

Fig. 1. Seven colored plus two black-and-white signs which were to be recognized by robots as they moved around the Tech Challenge maze. We refer to these nine signs as "stop", "biohazard", "yellow radiation", "water", "gas", "blue radiation", "start", "skull" and "eating utensils".

Digital images of the symbols were supplied to the Tech Challenge teams several weeks before Game Day. All of the signs exhibit contrasting intrinsic colors. Moreover, the symbols embedded within the maze on Game Day were printed on 4'×4' posters. So automatically finding these symbols would naively seem to be a straightforward task. But machine recognition of the signs is significantly complicated by its having to work with video frames shot at random perspectives, under variable illumination conditions, with unknown motion blur and against complex background clutter. Sign recognition consequently turned out to be tricky and difficult.

The basic approach we developed to solve this problem is illustrated by the flow diagram in figure 2. The machine starts by first quantizing the color content of every input video frame. The number of possible RGB combinations within the quantized output is reduced from 16M+ distinct colors down to 33 representatives. The machine next ignores any pixels whose quantized colors do not match a specified symbol's and forms connected components from those that do. Basic shape properties are used to nominate connected pixel regions within a video frame that plausibly correspond to a particular sign. Such regions become inputs to a set of trained classifiers. Regions surviving this final set of filters are marked with bounding box annotations that indentify symbols recognized by the machine.

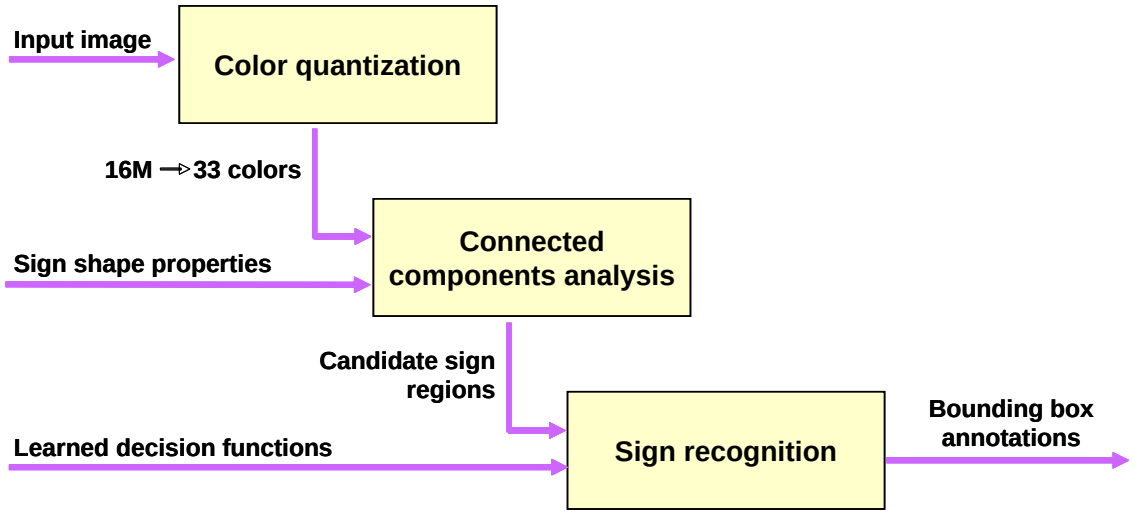

Fig. 2. Basic algorithm flow for detecting and recognizing the Tech Challenge signs.

The algorithm flow in figure 2 was developed on and tested against 4.5K+ video frames shot inside Lincoln Laboratory and the HAFB tennis bubble during summer 2012. Most of our experiments were conducted with images of 2'×2' versions of the TOC12 signs collected by hand-held video cameras. We intentionally placed the signs in laboratory locations with challenging lighting and clutter conditions which we knew would stress automatic recognition. During our algorithm development, we also consciously made trades between accuracy and speed. The final version of our algorithm chain could search a 640×480 video frame for the 9 TOC signs in approximately 6 seconds on a Dell Precision M6400 laptop.

We present algorithmic details for the primary blocks of figure 2 within the subsections below.

Color quantization

The colorings of the nine TOC12 signs arguably represent their most striking characteristics. Searching for particular color combinations within input imagery is consequently an obvious first step for automatic symbol recognition. However, color analysis of passive targets is complicated by unknown illumination sources as well as sensor response functions that vary from camera to camera. Though the intrinsic RGB values for each of the signs in figure 1 are *a priori* known, their mapping to measured RGB pixel outputs are not. Indeed, such color analysis perils are widely-appreciated in the computer vision community. Researchers therefore often try to avoid RGB pitfalls by working with grey-scale versions of input images.

Nevertheless, the gross colorings of the TOC12 symbols are so salient that they cannot be ignored. So we first quantize three-dimensional color space in order to reduce image output sensitivity to illumination and sensor response variables. Recall that 24-bit images can have 256^3=16,777,216 color combinations per pixel. Rather than describe three-dimensional color space in terms of Cartesian red, blue and green coordinates, it is often more useful to work instead with hue, saturation and value coordinates. Hue is an angular coordinate which ranges from 0° to 360° and intuitively correlates with colors of the rainbow. Saturation is a coordinate ranging from 0 to 100 which measures admixture of white. The value coordinate also ranges from 0 to 100 and indicates color brightness or darkness. A two-dimensional slice through HSV space with hue=0 is displayed in figure 3.

The figure also displays cuts within the saturation-value plane that define quantized color sectors. The upper right region corresponding to S > 50 and V > 50 is simply labeled "red". The lower right region with S > 50 and 5 < V < 50 is labeled "dark red". Similarly, the "light red" and "grey-red" regions have more white admixture than their "red" and "dark red" counterparts. The "grey" and black" regions in figure 3 have such small saturations and values that they cannot be meaningfully distinguished from similar regions in other 2D slices through HSV space corresponding to different hues.

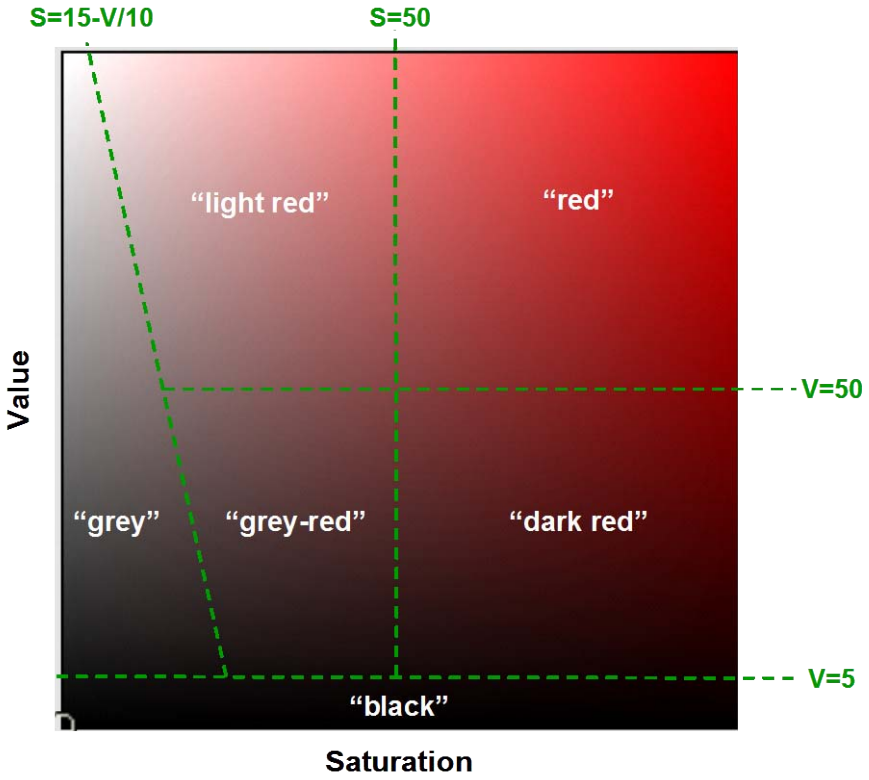

Fig. 3. A two-dimensional slice through HSV color space corresponding to zero hue. Saturation and value coordinates range from 0 to 100 along the horizontal and vertical axes. For each of 7 quantized hues, four subregions (e.g. "red", "light red", "dark red" and "grey-red") are defined within the SV plane. "Grey" and "black" regions also correspond to color space points with relatively low saturations and low values.

We define 7 quantized hues: red, orange, yellow, green, cyan, blue and purple. We also distinguish white, light grey, grey, dark grey and black. Thus the total number of our quantized colors equals 7×4+5=33. The mapping between 16M+ RGB values to the color labels is stored in precalculated lookup tables. So color quantization for any input image requires essentially no real-time computation and proceeds very quickly. Figure 4 presents one example of a video frame before and after quantization.

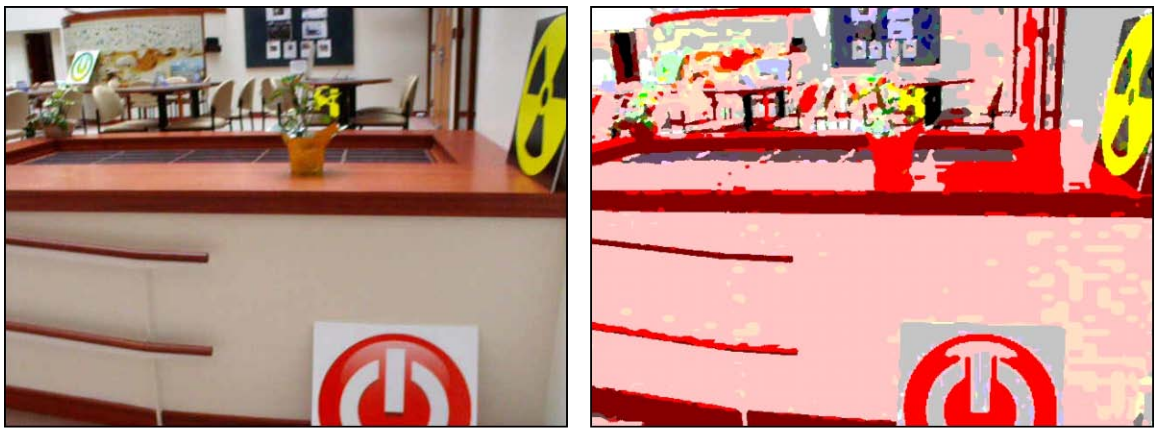

Fig. 4. General color quantization. (a) One representative video frame shot outside Lincoln Lab's library containing four TOC12 signs. (b) Color-quantized result after replacing every pixel's RGB values with one of 33 discrete triples.

It is advantageous to allow the precise borders for the 33 quantized sectors in 3D color space to depend upon the nine symbols. For instance when the machine searches for the "water" sign, it utilizes a lookup table where the definitions of "blue", "dark blue" and "grey-blue" are relaxed to allow for illumination and detector response variations. Similarly, the machine quantizes input pixels via a different lookup table with liberalized definitions for "red" and "light red" when it looks for the "stop" sign.

Only pixels whose quantized labels match the dominant color content for a specified symbol are passed from the first block in our algorithm flow to the second (see figure 5). Such pixels along with the sign's basic shape properties become inputs to connected component analysis which we consider next.

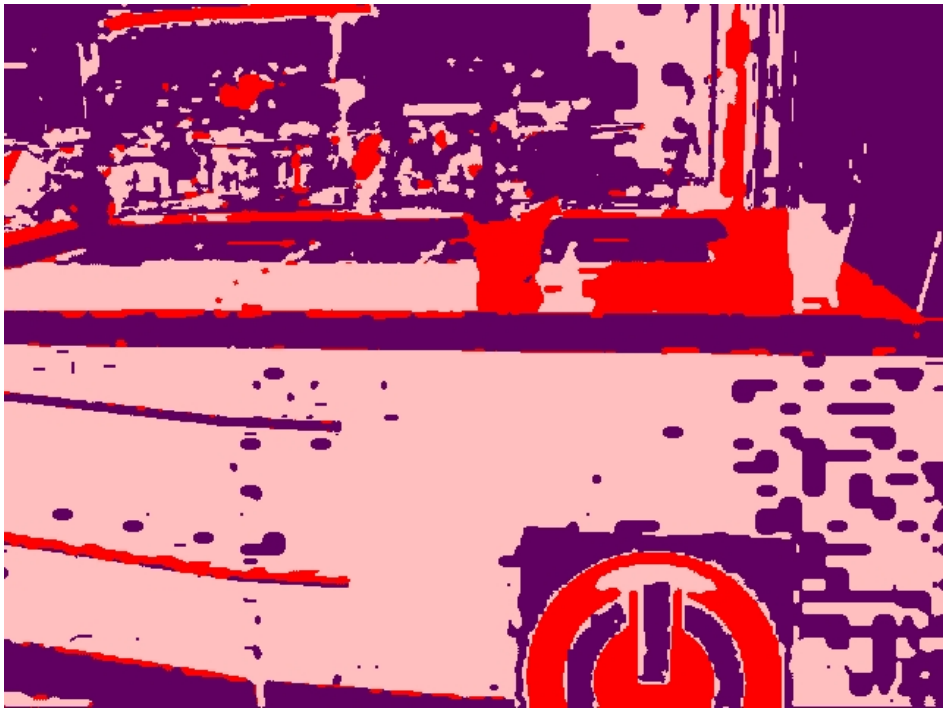

Fig. 5. Color quantization results for the "stop" sign. All pixels in figure 4b whose quantized colors do not correspond to red or light-red are replaced by a purple coloring. The machine ignores all purple pixels when it continues to search for a "stop" symbol in figure 4a.

Connected component analysis

The nine TOC12 symbols exhibit fairly simple morphological characteristics. For example, all of their intrinsic aspect ratios are 1:1. Projecting signs from world-space into video cameras angled at arbitrary perspectives introduces 2D aspect ratio variations. But for roll, pitch and yaw angles that yield sign views which the machine has any realistic chance of recognizing, projected sign aspect ratios range around a small neighborhood of 1:1.

Other symbol shape properties such as number of interior holes represent topological invariants that theoretically remain constant independent of viewing geometry. For instance, the red "stop" sign in figure 1 exhibits two islands of white pixels surrounded by a sea of red pixels. When projected onto a finite-sized image plane, the number of interior white holes can actually increase (see figure 4a). Nevertheless, our machine can reasonably search for connected pixel regions containing 2 or 3 interior holes when it looks for a "stop" symbol in an image. In contrast, connected components containing many holes are unlikely to correspond to a "stop" sign.

The connected regions within a binary version of figure 5 are color-coded in figure 6. For each connected component, we follow [1] and compute the region's aspect ratio, compactness, number of holes and number of horizontal crossings. The region's values for these variables are compared with empirically determined reasonable

ranges as a function of symbol type. Only components whose measurements are consistent with these ranges are retained for subsequent processing.

Fig. 6. Connected components within a binary version of figure 5 are color coded here. Note that the TOC12 "stop" sign corresponds to a single connected component in this view.

All of the TOC12 symbols also exhibit strong internal edge content. As the background behind any symbol is *a priori* unknown, we cannot assume that detecting the square edges of the sign itself is straightforward. But under reasonable illumination conditions, our machine can be expected to locate edges inside a sign. Moreover as figure 1 illustrates, such internal edges all have a side which is white, black or purple. So we search for strong edges within video frames that are adjacent to one of these three internal colors as a function of sign type. White edge results are illustrated in figure 7. As anticipated, the machine found strong internal edges within the foreground "stop" sign.

After iterating over all connected components in figure 6 and rejecting those whose shape and edge contents are grossly inconsistent with the "stop" sign, only 3 candidate regions remain which might plausibly contain the symbol of interest. Orange, red and green bounding boxes mark those regions in figure 8. At this stage in the algorithm flow, much more discriminating tests need to be performed on the surviving candidates in order for the machine to determine if any of them actually contain a "stop" sign. So we describe the classifier used to recognize TOC12 symbols in the following subsection.

Fig. 7. Purple coloring indicates strong edge contents for connected components in figure 5b that border upon regions with white or light grey quantized colors. Note that such edges occur inside the TOC12 “stop” sign.

Fig. 8. Orange, red and green bounding boxes mark candidate regions possibly containing the TOC12 “stop” sign which share the symbol’s basic color, edge and shape properties.

Sign classification

Automatic recognition of objects in imagery represents an active area of computer vision research. Over the past two decades, classifiers have been developed to look for a wide variety of targets such as faces, people and cars. Image classifiers are generally trained to map features extracted from pixel patches into an abstract vector space that is segmented into decision regions. A new test patch is subsequently declared to contain an object of interest if its features map to a vector space region that has previously been assigned the object’s label.

Machine learning researchers have expended significant effort over the years manually crafting clever sets of features which they believe capture salient information needed to identify objects. But recently, accumulating work within the artificial intelligence community has demonstrated that machines can automatically nominate large sets of features relevant for recognition tasks [2]. Such unsupervised feature learning has been applied to a wide range of problems including human face identification, voice recognition and real-time speech translation [3]. Moreover, unsupervised learning algorithms have been demonstrated to recognize text characters within uncooperatively-collected images as well as other approaches involving carefully hand-selected features [4]. So we adopt this general strategy for recognizing TOC12 signs within video frames.

The basic components of our sign classifier subsystem are presented in figure 9 which is adapted from [2]. We first synthesize a large corpus of training examples for each symbol to be recognized. The training sets must incorporate expected variations for a sign which can reasonably be expected to appear within test video frames. We consequently start with the TOC12 symbols in figure 1, rotate each in three-dimensions by random roll, pitch and yaw angles, and then project into an image plane. The coloring of the background plane, lighting color plus direction, and gaussian noise content are all randomized. Finally, some variable amount of smearing is intentionally introduced in order to simulate motion blur. Representative examples from 15,000 synthesized "stop" sign images 32×32 pixels in size are shown in figure 10.

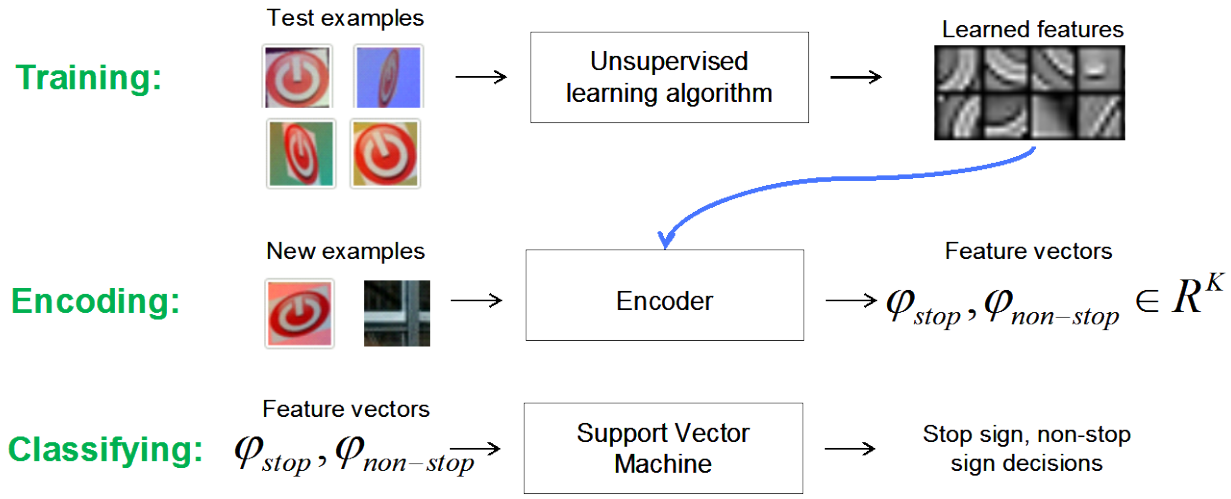

Fig. 9. Basic procedure for training a classifier to recognize different instances of the TOC12 "stop" sign. Thousands of synthesized variations of the symbol are first used to generate a feature dictionary via unsupervised learning. The dictionary subsequently becomes an input to an encoder which converts pixel patches into high-dimensional feature vectors. Such feature vectors along with "stop sign" or "non-stop sign" labels are used to train a linear Support Vector Machine. Once trained, the SVM can classify new pixel patches whose labels are *a priori* unknown.

A variant of K-means clustering is performed on 8×8 pixel patches sampled from the synthesized training images [4]. We choose K=1024 and refine the clusters over 15 iterations. The unsupervised learning algorithm generates a dictionary of 1024 features (see figure 11). These dictionary elements capture essential properties of the "stop" sign symbol such as its edges, curves and holes that transcend the rotation, coloring and blurring variations within the training set.

As figure 9 indicates, the dictionary becomes input to an encoder which converts 32×32 image regions into feature vectors. The encoder loops over all 8×8 pixel patches within an input image, projects each onto the dictionary's K elements, and performs a soft, non-linear reduction onto K-dimensional histograms [4,5]. The histogram descriptors are averaged together within 3×3 regions of the 32×32 image. The encoder's output feature vectors are consequently 9K dimensional.

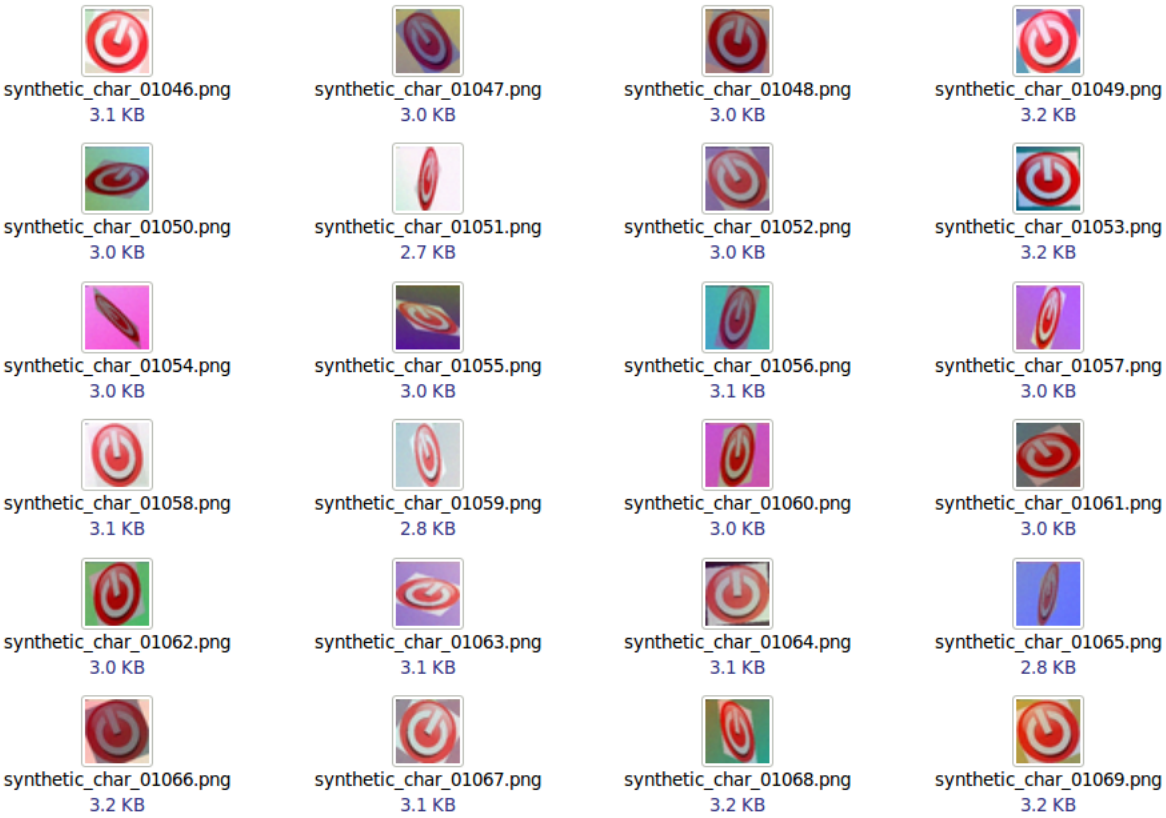

Fig. 10. Synthetically generated views of the TOC12 “stop” sign.

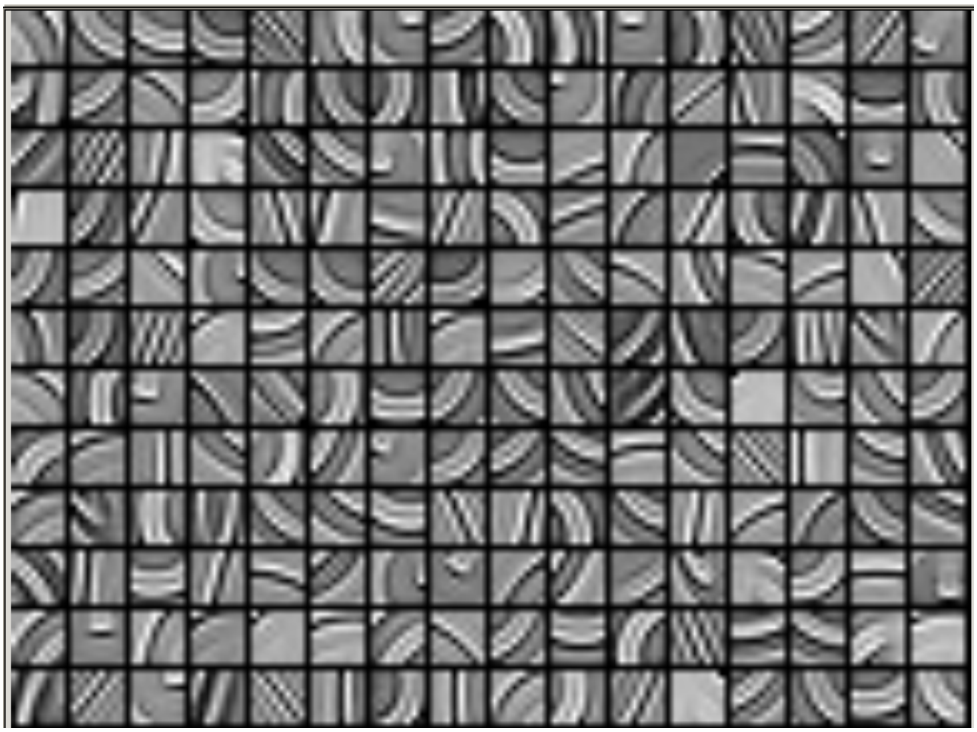

Fig. 11. 8×8 pixel patch elements from the “stop” sign dictionary generated via unsupervised feature learning.

The last step in sign classification step involves a Support Vector Machine. For each TOC12 sign, feature vectors associated with 15K synthesized examples and 96K non-sign examples harvested from random internet pictures are used to train a linear SVM. We employ the dlib library to compute probabilistic decision functions for all nine symbols in figure 1 [6]. New image regions may subsequently be classified as a TOC12 sign or not based upon probabilities output by the nine decision functions (see figure 12). We intentionally set probability thresholds for sign recognition high in order to minimize false alarms.

Fig. 12. The trained SVM correctly classifies one of the three candidate regions in figure 8 as corresponding to the TOC12 "stop" sign and rejects the other two.

Sign recognition results

We tested the end-to-end algorithm flow in figure 2 on 4.5K+ video frames collected during summer 2012. Many of the resulting images contained no sign targets. But others had one or more visible 2'×2' symbols at camera ranges varying from approximately 1 to 25 meters. In the run-up to Game Day in Nov 2012, we unfortunately did not have time to quantify the precision and recall of our sign recognition system. So we settled instead for qualitatively assessing its performance.

Figure 13 displays 25 representative examples of successful sign recognition results. The bounding boxes appearing in this figure enclose genuine TOC12 symbols. Moreover, the boxes' colorings indicate correct sign identities: cyan, blue, yellow, white, purple, brick red and cream correspond to the "stop", "biohazard", "yellow radiation", "water", "gas", "blue radiation", "start", "skull" and "eating utensils" signs. The backgrounds in some of the pictures appearing in figure 13 are highly cluttered. Moreover, 500+ of the 4.5K+ views were collected from the second floor of LL's library atrium looking downwards towards ground level. These pictures were intended to mimic shots gathered by the airship as it looked down upon the maze. We believe the results presented in the figure demonstrate a nontrivial automated recognition capability.

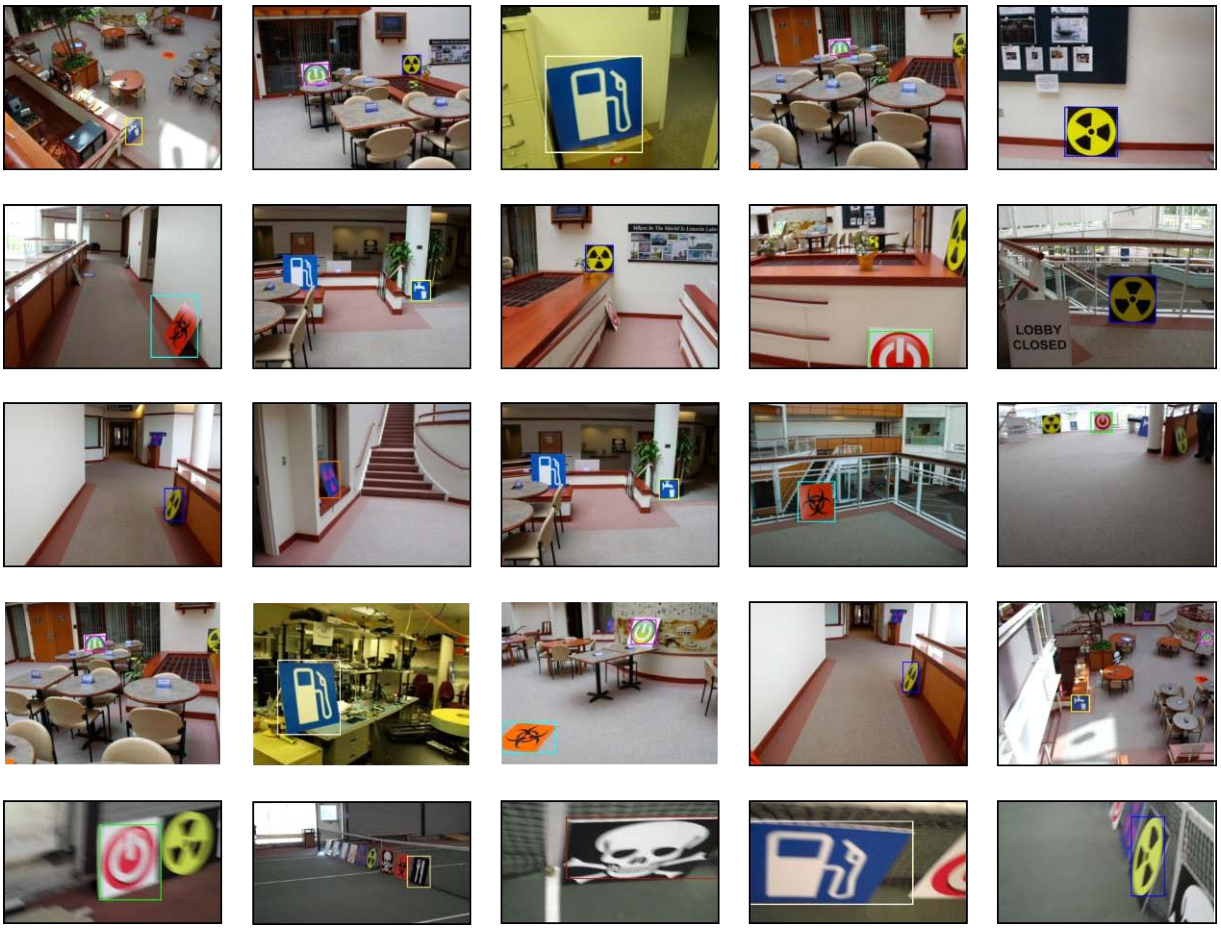

Fig. 13. Representative examples of successful TOC12 sign recognition from the algorithm flow of figure 2. Video frames were collected around S-lab as well as within the tennis bubble on HAFB.

Of course, several missed detections are evident in figure 13. Since we intentionally set recognition thresholds in order to minimize false alarms, such errors are neither unexpected nor overly worrisome. On the other hand, figure 14 exhibits representative examples of sign recognition failures which include several incorrect classifications and a few false alarms.

Fig. 14. Representative examples of sign recognition failures include incorrect classifications and false alarms.

During this year's Tech Challenge, we did not attempt to quantify the ratio of recognition system successes to failures. But for the 4.5K+ video frames collected inside Lincoln Lab and the HAFB tennis bubble of the 2'×2'

TOC12 signs, we estimate that 80% of the frames were correctly classified while 20% contained recognition errors.

References

1. L. Neumann and J. Matas, *Real-time scene text localization and recognition*, CVPR 2012.

2. A. Coates, *Demystifying unsupervised feature learning*, Stanford University Ph.D. thesis, 2012.

3. New York Times, *Great Strides for Deep Learning*, 26 Nov 2012.

4. A. Coates, B. Carpenter, C. Case, S. Satheesh, B. Suresh, T. Wang, D. Wu, and A. Ng, *Text detection and character recognition in scene images with unsupervised feature learning*, ICDAR 2011.

5. A. Coates, H. Lee and A. Ng, *An analysis of single-layer netowrks in unsupervised feature learning*, AISTATS 14, 2011.

6. D. King, *dlib C++ library*, http://dlib.net/ml.html.