Monocular Obstacle Detection

Ankur Kumar, Ashraf Mansur

***Abstract* — A feature extraction approach to detect obstacles in an indoor environment using monocular sequence of images is presented. The approach aims to distinguish between two kinds of obstacles – walls and objects. This differentiation is essential for a robot to autonomously navigate indoors. The approach is split into two steps – feature extraction and supervised learning using SVM. In addition to features computed using computer vision techniques in a region of interest (ROI), some features are computed based on unsupervised learning methods. A total of 10 features are extracted per ROI and in the second step, a SVM classifies the ROI as Wall, Object, or Unknown. Of the 9 datasets, 2 were used to train the SVM, and this gave a mean classification accuracy of ~76% using the percent misclassification metric. From this classification, a robot can autonomously detect walls and objects using its one camera as a sensor. The results from the presented approach can be used for obstacle avoidance, path planning, and visual SLAM.**

I. Introduction

ROBOTS are quite able to move about autonomously using range and laser sensors, which tend to be expensive, bulky, and energy consuming. Employing one camera as a sensor is rather preferred. However, a new set of challenges is unearthed. Some of these challenges include depth estimation, perception, and route planning. For indoor environments, route planning can get exceedingly difficult if the robot perceives everything as an obstacle based on a computed set of features. For instance, if a robot is in the bedroom and wants to go into the kitchen to perform a path, it should be able to detect a wall and an opening in that wall through which it can pass through safely. Using features such as Harris corners or Shi and Tomasi's *Good Features To Track* [1] can lead to extraction of features on the wall, especially, the edges between the wall and the ceiling, paintings on the wall, windows, wallpaper, etc. If this situation is complicated by the presence of people, chairs and tables, the robot often is unable to distinguish between the features pertaining to the object and to the wall.

Monocular video sequences are cheaper to acquire for a household robot and do require less processing power. Depth perception is always a challenge in a monocular video sequence. Obstacle detection often depends on depth perception because the obstacle appears to have depth

Ankur Kumar and Ashraf Mansur are students of Robot Learning Course at Cornell University, Ithaca, NY 14850. {ak364, aam243} @ cornell.edu.

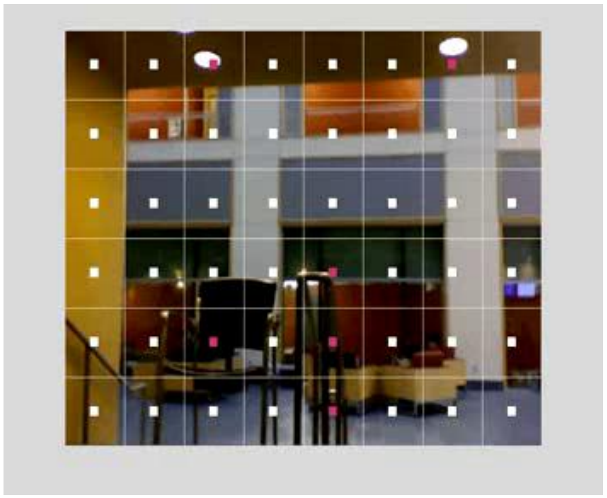

Figure 1: The result of classification on a frame of a dataset. The regions with walls are marked in white, while the regions with an obstacle are marked in red.

characteristics when put against any scene, which lies behind the obstacle. Without depth information, it can get difficult to estimate if the obstacle is indeed an obstacle or part of some cluttered background. Optical flow can record the change in the robot's environment but the generated flow field can have inaccurate information, especially, if the robot is a helicopter or moving irregularly due to control issues. Though optic flow is used as one of the 10 features presented in this paper, other features do make up for its deficiencies. The popular SIFT [2] by Lowe is region based and often yields numerous unwanted features. This can get cumbersome to process.

This paper describes the intuition behind every feature used to differentiate between walls and objects. This is followed by details of the experiments and a discussion of the results. An example snapshot of the result from feature extraction followed by classification is shown in Figure 1.

II. RelAted Work

Extracting features from a region of interest in an image is one of the most important problems in computer vision and is the gateway to image matching and object recognition. Through the voluminous work in the area, a handful of papers were selected that might be useful in differentiating between walls and objects in a scene. One of the approaches for retrieving object descriptors is using SIFT, which is primarily used for image matching and object recognition [2]. Another method for extracting feature descriptors is to use shape contexts [3] and steerable filters [4]. Newer methods such as PCA-SIFT, which is the use of Principal Component Analysis (PCA) on the SIFT descriptors to find the [5] also can be used to find descriptors. Moment invariants described in [6] is yet another method for finding feature descriptors. A multi-scale algorithm for the selection of salient regions as feature descriptors is described in [7]. This algorithm treats saliency as local complexity defined by an entropy function. Another well-known method for finding feature descriptors is the computation of the histogram of gradients described in [8]. Other methods for extracting feature descriptors will be explored in Section III (B).

III. Approach

A. Methodology

Features described in Section III (B) are extracted by ROI for two labeled training datasets. This is used to train the SVM. For the test dataset, features are extracted for the ROI and then classified using an SVM. Once classified an accuracy metric based on percent misclassification is computed (Section IV).

B. Features

Ten features are computed for a ROI to distinguish between a wall or ceiling or floor and an object. The features range from being histogram based, spatial based, and gradient based. The input images are converted from RGB to grayscale before features are extracted.

1) Histogram of Gradients (HOG)

This method described by Dalal et al in [8] computes the occurrences of particular gradient orientations in localized portions of the image. Basically, it is a binning algorithm for gradient orientations. The premise of these descriptors is that local object appearance and shape within an image can be described by the distribution of intensity gradients or edge directions. Dividing the image into small connected regions and the gradient or edge directions within this region are compiled into a histogram, which represents the descriptor for the region. HOG is invariant to geometric and photometric transformations which tend to appear in large spatial regions. HOG aids in understanding gradient directions of ROI with walls vs. ROI which have objects or objects and walls. The HOG should behave differently in both cases.

2) Energy e_1

The energy per pixel in an ROI based on its gradients signifies their contribution to the ROI. Preserving energy in an ROI would be to suppress pixels with lower energy and keeping the pixels with highest energy. This energy calculation per pixel is especially useful when comparing a ROI that contains a wall vs. a ROI that contains an object or object and a wall. The energy e_1 is defined in equation 1 [9].

$$e_1(\mathbf{I}) = \left|\frac{\partial}{\partial x}\mathbf{I}\right| + \left|\frac{\partial}{\partial y}\mathbf{I}\right| \quad (1)$$

Since the above energy will be computed per pixel in the ROI, PCA is executed on the energy values to determine the principal components of the energy function. Using this notion, the principal components of walls will differ from an obstacle.

3) Energy of HOG

A seam carving approach is taken where the lower energy pixels are suppressed while the higher energy pixels are retained in the ROI. Computing the energy of HOG, e_{HOG}, will indicate the energy contained in various gradient orientations. The energy contained in various gradient orientations will differ for walls vs. objects. Equation 2 defines e_{HOG} [9].

$$e_{HOG}(\mathbf{I}) = \frac{e_1(\mathbf{I})}{\max(HOG(\mathbf{I}(x,y)))} \quad (2)$$

Once again, PCA can be carried out on e_{HOG} to retrieve the principal components of energy contained in various gradient orientations. These principal components will contain similar information for all ROI containing walls and would be dissimilar for ROIs containing obstacles.

4) Hessian Matrix

The Hessian matrix is based off second partial derivatives and these matrices encode shape information. The determinant of the Hessian matrix can be used to determine blobs with automatic scale selection. Equation 3 shows the Hessian matrix, where $L_{xx}(\mathbf{x})$ is second partial derivative in the x direction.

$$H(\mathbf{x}) = \begin{bmatrix} L_{xx}(\mathbf{x}) & L_{xy}(\mathbf{x}) \\ L_{xy}(\mathbf{x}) & L_{yy}(\mathbf{x}) \end{bmatrix} \quad (3)$$

The eigenvalues of the Hessian matrix can be used to determine tubular, blob, and planar structures in an ROI [10]. Computing the determinant and eigenvalue of the Hessian matrix can be used to characterize the possible shape of the object/obstacle in an ROI. For a wall, the eigenvalues of the Hessian will represent a planar surface.

5) Moments

Moment invariants described in [6] indicate a certain particular weighted average of the image pixel’s intensities. Moments can describe the centroid and the orientation of a ROI. Higher order moments, M_{ij}, describe various shape measures. Computation of *n*th order moments are given by equation 4. Moments are translation, scale and rotational invariant and thus, provide a valuable descriptor. Moments pertaining to walls can be learned by the SVM classifier, which it can then employ to classify ROI as a wall. Moments up to 3rd order were used as features.

$$M_{ij} = \sum_x \sum_y x^i y^j I(x,y) \quad (4)$$

6) Texture

Using the algorithm presented in [11], moment based texture for an ROI is implemented as a feature. Texture can be used to identify surfaces in an image and computing moments in local regions of the image can be used as texture features. Image moments within a window around each pixel are calculated based on moment masks as shown in Figure 2.

$$m_{00} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \quad m_{10} = \begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \quad m_{01} = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}$$

$$m_{20} = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix} \quad m_{11} = \begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix} \quad m_{02} = \begin{bmatrix} 1 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 1 \end{bmatrix}$$

Figure 2: Moment masks, up to 2nd order, for computing image moments around each pixel

The texture features from these moments are determined by applying a nonlinear transformation followed by an averaging operation. The nonlinear operation used is the *tanh* function. The size of the window determines if global features are detected. Once the texture feature is calculated, its 'strength' is determined by taking the average of the absolute values of the texture feature. The texture for walls will be quite different from the texture of an object or an object and a wall within a ROI. Walls usually have texture mostly due to paint and their planar properties. Object and walls residing within the same ROI will have a more complex texture. Thus, this difference in complexity makes texture a feature to examine.

7) Optical Flow Divergence

Optical flow divergence for detection of obstacles is discussed by Nelson and Aloimonos in [12]. Optic flow for two consecutive frames of the ROI is computed based on hierarchical Lucas Kanade optical flow method using pyramids. The flow divergence of the optical flow in the ROI is computed as shown in equation 5.

$$div(\mathbf{F}) = \nabla \bullet \mathbf{F} = \frac{\partial U}{\partial x} + \frac{\partial V}{\partial y} \quad (5)$$

In equation 5, **F** denotes the computed optical flow. *U* and *V* denote the velocities in the two dimensions, x and y, respectively. When an obstacle is in relative motion towards the camera, the camera produces an expanding image. As a result, the intensity values have diverged from its previous point outwardly and this causes the *div*(**F**) to be positive. Subsequently, an ROI can be classified as whether it contains an obstacle or not based on the positivity of *div*(**F**). A wall would have low levels of *div*(**F**), while an obstacle or an obstacle and a wall would have higher levels of *div*(**F**)

8) Entropy of Histogram

The histogram of intensities for a ROI is calculated. The entropy for a ROI with objects should be much higher than the entropy for a ROI with just a wall. The number of salient features usually increases when relatively complex structures such as objects appear in an ROI containing a wall, which have less salient features. Equation 6 shows the calculation for entropy of the histogram and *p* contains histogram counts.

$$Entropy = -sum(p * \log_2 p) \quad (6)$$

9) *PCA of Histogram*

Principal Component Analysis (PCA) of the histogram of intensities for a ROI will characterize orders of variability in the ROI. For instance, the variability of a wall will be much lower and hence, a principal component characterizing this variability will be computed. This principal component can be learned and when a similar match to this principal component characterizing the variability of the wall is found in the test data, the ROI in the test data can be marked as a wall. In a way, PCA reveals the internal structure of the ROI which explains the variability of the histogram of the ROI.

10) Blind Source Separation and ICA

Blind Source Separation (BSS) is used to seek independent unknown sources which are linearly combined in an unknown fashion into a mixture. Using higher order statistics, Independent Component Analysis (ICA) is able to solve this BSS under certain assumptions such as non-Gaussianity of the sources. When an object is in the vicinity of a wall, the histograms of the wall and the object get linearly combined in an unknown fashion into a histogram mixture. This histogram mixture can be treated as a blind source separation problem and ICA can be utilized to separate the two sources of the histogram mixture. This approach has been inspired by Ankur Kumar's work on timbre extraction using blind source separation [13].

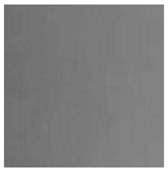

Figure 3: Wall in a ROI

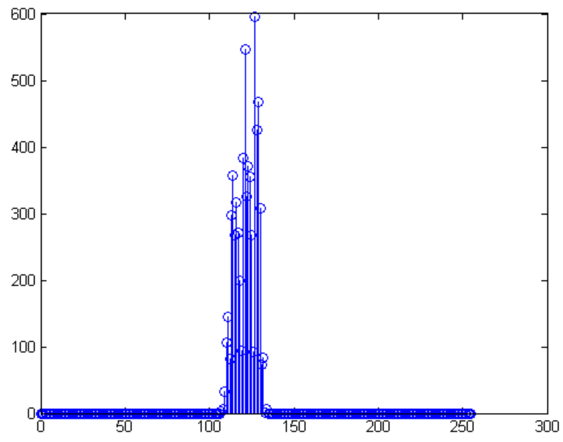

Figure 4: Histogram of Wall from Figure 3

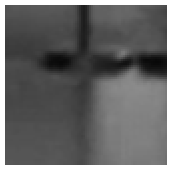

Figure 5: Wall and Object in a ROI

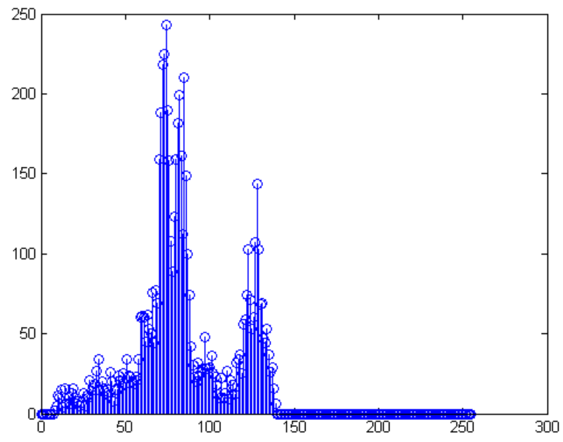

Figure 6: Histogram of Wall and Object in Figure 5

Figure 3 shows a wall and its corresponding histogram in figure 4. Figure 5 shows a wall and an object in a ROI and its corresponding histogram in Figure 6. A histogram mixture containing the histogram of Figure 4 and an unknown histogram of the object were mixed together to form the histogram mixture shown in Figure 6. This histogram mixture can be analyzed using blind source separation techniques under the assumption of non-Gaussianity of the sources of the histogram mixture. The separation of the sources is achieved by estimating the "de-mixing" matrix acting on the histogram mixture to separate the sources. The separation is done in the Fourier Domain because histogram is based on frequency counts of intensity values.

A Short-Time Fourier Transform (STFT), shown in equation 7, is used to compute the Fourier Transform of the histogram mixture based on a moving window scheme. Since features are extracted on grayscale images, the length of the STFT is limited to 256. The STFT acting on the histogram mixture will result in the output **F**.

$$STFT(\tau,\omega) = \int_t x(t)g(t-\tau)e^{-j\omega t}dt \quad (7)$$

The histogram mixture or mixture signal needs to be "whitened." Whitening strives to attain decorrelation between several dimensions of a dataset by applying a linear transformation with the use of second order statistics i.e. variance. The basic idea of PCA is to provide the set of components that gives the maximum variance of the dataset. Using PCA to "whiten" the histogram mixture, the variance has been maximized and the redundancy reduced. The matrix with retained eigenvectors, $\mathbf{E_V}$, and the diagonal matrix corresponding to the retained eigenvalues, $\mathbf{E_D}$, can be used to find the whitening matrix, **V**, as shown in equation 8.

$$\mathbf{V} = \mathbf{E}_{\mathbf{D}}^{-\frac{1}{2}} * \mathbf{E}_{\mathbf{V}}^{T} \quad (8)$$

The dimensionally reduced whitened version of **F** called $\mathbf{F_W}$ is computed as shown in equation 9. The rows of $\mathbf{F_W}$ are decorrelated.

$$\mathbf{F_W} = \mathbf{V} * \mathbf{F} \quad (9)$$

Until now, only second order statistics have been exploited. With the notion of ICA, higher order statistics are used to perform source separation on the whitened matrix $\mathbf{F_W}$. The ICA operation results in a transformation matrix **H**, which aids in determining the de-mixing matrix, $\mathbf{D_M}$. During the ICA operation, non-Gaussianity is measured by the approximation of negentropy. Fast ICA, a fixed point algorithm, is employed with a non-quadratic function, chosen from a hyperbolic family of functions, to maximize negentropy. Fast ICA is used on the whitened matrix, $\mathbf{F_W}$, and the resulting **H** is used to find the de-mixing matrix, $\mathbf{D_M}$, as shown in equation 10. Equation 11 can be used to compute the mixing matrix, **M**, can also be determined and its columns represent the spectral information of the individual source.

$$\mathbf{D_M} = \mathbf{H}^T * \mathbf{V} \quad (10)$$
$$\mathbf{M} = \mathbf{V_D} * \mathbf{H} \quad (11)$$

Finally, the independent component vectors, as a matrix $\mathbf{S_I}$, can be derived by multiplying the original STFT, **F**, with the de-mixing matrix, $\mathbf{D_M}$. This is given in equation 12. The row vectors of $\mathbf{S_I}$ are called the ICA filters and these carry the varying gain of the spectral components of the individual sources. For the entire process of blind source separation, it is assumed that the ROI can contain a maximum of two sources - the wall and the object or obstacle.

$$\mathbf{S_I} = \mathbf{D_M} * \mathbf{F} \quad (12)$$

The whitening matrix, **V**, the de-mixing matrix, $\mathbf{D_M}$, and the sources, $\mathbf{S_I}$, form a feature vector for this feature extraction stage. Each column of $\mathbf{S_I}$ represents an object and a wall as sources of the histogram mixture.

C. *Classification Algorithm*

Support Vector Machine is chosen as the classifier to discriminate between the two classes, wall and object, based on the 10 features mentioned in Section III (B). In particular we train a binary SVM model for each class type using flow divergence only, BSS only, and all features. A sigmoid *tanh* function is used as the kernel for the SVM. SVM$^{\text{light}}$ was used to classify and run the experiment [14].

IV. Experiments

The experiments conducted correspond to the trained models: flow divergence (FD) only, BSS only, and all features.
Specifically, each experiment consists of 7 test datasets.

To build our dataset, we created obstacle courses in Phillips Hall corridors, and recorded the course with a laptop camera placed on wobbly lab chair. Due to the unstable movements of the chair, we were able to simulate some shaky/noisy images which are expected from the rotorcraft camera. In particular, we recorded 9 videos at 25 frames per second, each having length less than one minute.

From the 9 videos, we labeled 2 for the training set. To label the training set, each frame was evenly partitioned based on aspect ratio into regions and manually annotated as wall, floor, ceiling, or object. These regions were of size 80x80. In particular, any portion of an object took preference over walls during labeling. For the initial baseline estimate, since our classification algorithm only discriminates between walls (includes floor and ceiling) and objects, our performance baseline using random selection would be 50% accuracy/error.

The system then classified all 7 test datasets using the 3 training model categories for each class type – walls and objects. Using the predictions of the SVM classifications, we marked each region of the dataset frames based on max vote prediction between wall and object. The predictions were then verified by inspection of the annotated test data every 50th frame of the dataset. The performance results computed based on the following misclassification metric are captured in Table 1:

$$\frac{misclassified_{walls} + misclassified_{objects}}{totalclassifications} * 100$$

Dataset	Base%	FD%	BSS%	All%
Obs5	50	15	42	12
Obs9	50	35	18	22
Obs3	50	40	55	20
Obs6	50	20	25	35
Obs7	50	20	25	35
Obs8	50	25	30	20
Obs10	50	45	55	25

Table 1 Captures precent misclassification for baseline (Base%), flow divergence (FD%), BSS%, and all features (All%). Note that the best misclassification percentage is 0

The misclassification percentages are represented as estimates to mitigate human error in labeling (i.e. those introduced when distinguishing between mounted objects on walls from walls, etc).

V. Analysis and Discussion

From the results above we see that using all 10 features helped improve overall system performance for 4 out of the 7 datasets.

For the flow divergence experiment, we observed via inspection of the predicted label outputs, that this feature doesn't detect objects well and possibly not at all. The feature's performance is reasonable since it only chooses the wall label which is the most frequent in our dataset.

For the BSS experiment, we observed some correct object and wall classifications; however the feature appears to favor the object label over wall for non-object regions. Although the histogram mixture can be separated into wall and object, the de-mixing matrix will differ from scene to scene. This is because the histogram mixture differs from scene to scene based on the type of objects present in the scene. This will cause variability in the de-mixing matrix, which is learned by the SVM during the training phase.

For the final experiment, we observed relatively good performance for the Obs3 dataset (Phillips Lab contained clusters of objects in tight quarters), Obs5 & Obs8 (Phillips corridor with dim lighting and few objects), Obs9 (Duffield). Below are some snapshots from the datasets that our system performs well on.

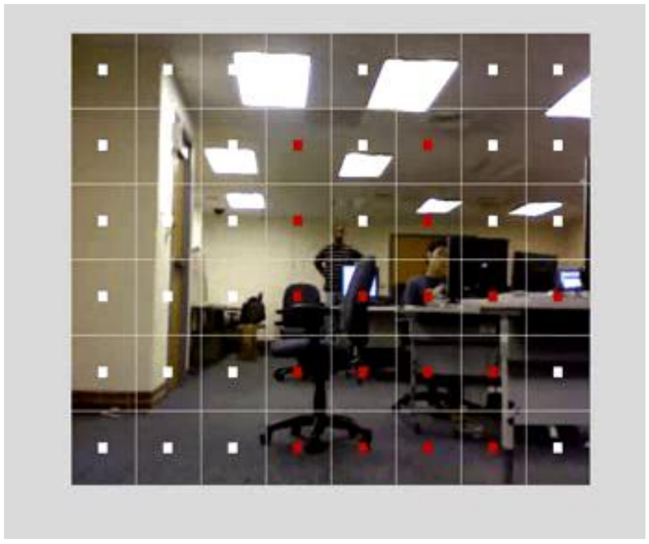
Figure 7 : Captures classification of cluster of objects in Phillips Lab (Obs3).

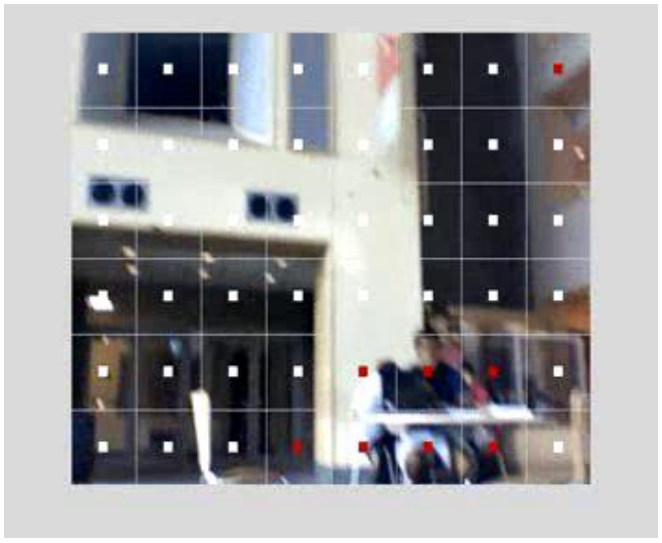
Figure 8 : Duffield Atrium (Obs9).

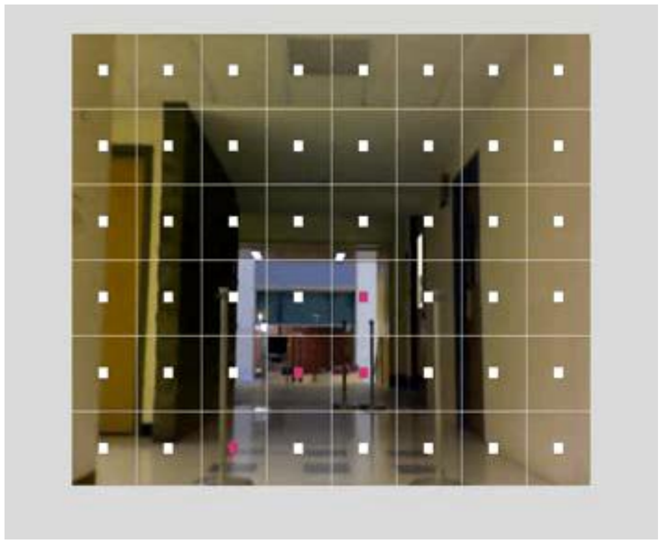
Figure 9 : Phillips Corridor with dim lighting (Obs5).

The performance results for Obs6 and Obs7 are poor due to limited lighting and mislabeling of glass showcases containing obstacles. However from a robot navigation standpoint, the labeled glass regions would be avoided eventually since it contains an obstacle. The figure below captures the label prediction of our system for Obs6 (similarly for Obs7).

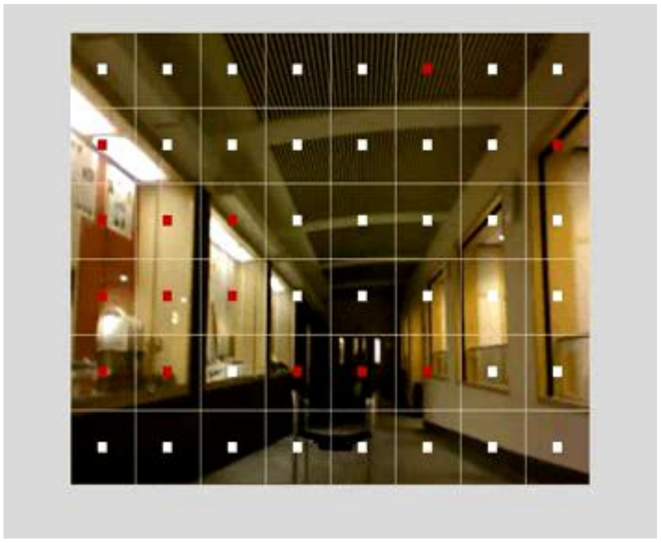
Figure 10 : Phillips Corridor ramp containing glass showcases (obs6).

Our system does not consider spatial coherence between different ROI. Two ROI next to each other are treated independently and the SVM classifies the ROI as an obstacle or wall. Similarly, two ROI identical in the coordinate space but differing temporally (two frames) will also be treated independently and classified independently. This is a limitation of our system and can be greatly improved if other learning techniques such as Markov Random Fields be employed.

VI. Conclusion and Future Work

Based on our approach and the experiments conducted, we see that the selected features perform relatively well in distinguishing between walls and objects but it may not be feasible to compute all these features for real-time processing. Applying a feature selection algorithm or maximum relevance minimum redundancy technique would help reduce our feature set to the most salient given particular accuracy and time constraints. In addition, a better supervised learning with Markov Random Fields may improve the accuracy of the system.

Additionally, to improve labeling of the datasets, we could use multiple annotators that comply to an inter annotator agreement (scored appropriately via Cohen’s Kappa [15] or Fleiss’ Kappa [16]) to generate consistent labels for data. Once the system is real-time, obstacle avoidance methods involving path planning can be tested.

References

[1] J. Shi and C. Tomasi, "Good Features to Track," *1994 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'94)*, 1994, pp. 593 - 600.

[2] D. G. Lowe. "Object recognition from local scale-invariant features". *Proceedings of the International Conference on Computer Vision*. **2**. 1999. pp. 1150–1157.

[3] S. Belongie, J. Malik, and J. Puzicha.” Shape matching and object recognition using shape contexts.” IEEE PAMI 24(4):509.522, 2002.

[4] W. Freeman and E. Adelson. “The design and use of steerable filters” IEEE PAMI, 13(9):891.906, 1991.

[5] Y. Ke and R. Sukthankar. PCA-SIFT: A more distinctive representation for local image descriptors. CVPR, pages 511.517, 2004.

[6] L. Van Gool, T. Moons, and D. Ungureanu. Affine / photometric invariants for planar intensity patterns. *4th ECCV*, pages 642.651, 1996.

[7] T. Kadir and M. Brady. Scale, Saliency and Image Description. IJCV 45 (2):83-105, 2001

[8] N. Dalal, and B. Triggs. Histograms of oriented gradients for human detection. CVPR: 2, 886–893, 2005.

[9] S. Avidan and A. Shamir. “Seam Carving for Content-Aware Image Resizing.” ACM Trans Graph 26, 3, 10, 2007

[10] Y. Sato et al. Tissue classification based on 3D local intensity structures for volume rendering. IEEE Trans. Visualization and Comp. Graphics, 6(2):160-180, 2000.

[11] M. Tuceryan, “Moment based texture segmentation”, Pattern Recognition Letters, volume 15, 1994, pp. 659–668.

[12] R. C. Nelson and J. Aloimonos. “Obstacle avoidance using flow field divergence.” IEEE PAMI 11(10): 1102-1106, 1989

[13] Kumar, Ankur. “Design of a Timbre Extraction Process using Blind Source Separation.” Cornell University, 2008.

[14] T. Joachims, Making large-Scale SVM Learning Practical. Advances in Kernel Methods - Support Vector Learning, B. Schölkopf and C. Burges and A. Smola (ed.), MIT-Press, 1999.

[15] Cohen, Jacob. (1960) A coefficient of agreement for nominal scales, *Educational and Psychological Measurement* Vol.20, No.1, pp. 37–46.

[16] Fleiss, J. L. (1971) "Measuring nominal scale agreement among many raters." *Psychological Bulletin*, Vol. 76, No. 5 pp. 378–382