Detecting Specular Surfaces on Natural Images

Andrey DelPozo
Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, IL USA
delpozo2@uiuc.edu

Silvio Savarese
Beckman Institute
University of Illinois at Urbana-Champaign
Urbana, IL USA
silvio@uiuc.edu

Abstract

Recognizing and localizing specular (or mirror-like) surfaces from a single image is a great challenge to computer vision. Unlike other materials, the appearance of a specular surface changes as function of the surrounding environment as well as the position of the observer. Even though the reflection on a specular surface has an intrinsic ambiguity that might be resolved by high level reasoning, we argue that we can take advantage of low level features to recognize specular surfaces. This intuition stems from the observation that the surrounding scene is highly distorted when reflected off regions of high curvature or occluding contours. We call these features static specular flows (SSF). We show how to characterize SSF and use them for identifying specular surfaces. To evaluate our result we collect a dataset of 120 images containing specular surfaces. Our algorithm can achieve good performances on this challenging dataset. Particularly, our results outperform other methods that follow a more naive approach.

1. Introduction

The ability to perceive and interpret the geometric shape and semantic meaning of objects is essential for an intelligent visual system. At that end, it is desirable to design algorithms that allow the identification and classification of different types of materials and textures from an image. Even if many techniques have been proposed for generic texture classification, objects with highly reflective surface properties such as a polished car, a teapot or bike pipe, pose great challenges for recognition. One reason is that their appearance is highly dependent on the object shape, the appearance of the surrounding scene and the position of the observer. This gives rise to a extremely difficult representation problem due to the daunting number of possibilities that one would need to take into account for the subsequent modeling step. For instance, Figure 1 shows the same reflective surface with different background scenes. The appearance of the two is completely different even if the object is made of the same material. Typical features and models proposed in texture classification literature [7, 17, 2] would most certainly not be able to capture such extreme visual variability.

Figure 1. The figure shows the same reflective surface with different background scenes and under different view points. Notice that even if the material is the same, the appearance is completely different in the two cases.

It is clear that this recognition problem may be tackled at several levels. One level would be the one of interpreting a reflective surface by using high level semantic reasoning about the scene [8]. For instance, prior knowledge about the appearance of an urban scene would tell us that the deformed buildings within the mirroring sculpture in Figure 1 cannot be real buildings but only reflections.

We argue, however, that there is still much research to be done at the feature level. We would like to be able to answer questions such as: can we identify low-level signatures that allow us segment and recognize reflective objects? As shown by Osadchy *et al*. [10], specular highlights are certainly one of those low level signatures. Specular highlights, however, are not always detectable and often even completely absent from the object. Hence, they cannot be used as a main or unique cue for recognizing specular surfaces.

In this work, we introduce a novel feature that we argue can be successfully used as low-level cue for such a recognition task. We call this cue *static specular flow* (SSF). A SSF cue may be identified in curved reflective surfaces in prox-

1-4244-1180-7/07/$25.00 ©2007 IEEE

imity of either occluding contours of the object or regions where the difference of the two surface principal curvatures is high. Figure 3-upper panel, shows examples of SSF. In such regions, the reflected scene is highly deformed, and produces consistent patterns that can be used as a signature for specular surfaces. We propose a multi-scale approach for automatically extracting and characterizing such SSF cues and show that SSF allows recognizing the presence of specular object with high classification accuracy.

Our approach requires minimal assumptions about the surrounding scene and illumination conditions, assumes no prior knowledge about the shape of the reflective object and utilizes a single monocular image for the recognition task. To the best of our knowledge, our work is the first attempt aimed at recognizing specular objects from natural images with such a small set of assumptions.

2. Previous Work

Very few works have attempted to classify and recognize reflective materials due to the above-mentioned difficulties. Osadchy *et al*. [10] propose a recognition scheme based on specular highlights as a primary source of information. In their approach, the geometry of the object requires be known a priori. Highlight features are combined with other cues, such as multiple views of the object or/and motion. While this approach has been successful in general, real world cases often include situations where highlights are absent or multiple views and motion cues are not available. Dror *et al*. [3] take advantage of the statistics of natural scenes to estimate surface reflectance. While their approach is useful to characterize different material classes, it requires to build statistics from the whole image, which in turn might not be possible when the material surface is not the dominant element in the image. Additionally, assumptions on the object shape are made which limit the application of these techniques to the general case. Roth and Black[12] propose a technique for differentiating reflections from non-reflective textures under the assumption that they both belong to the same surface element; the hypothesis of moving observer is also required.Work on classifying moving specular highlights is also presented by[16]. It is also worth mentioning the work by McHenry and Ponce [9] which tackles the equally difficult problem of recognizing transparent surfaces. The authors introduce the interesting idea of classifying a transparent material by comparing it with the surrounding background. A similarity measure based on several cues indicates the presence of the transparent element. This approach, however, cannot be applied to reflective materials as it is very unlikely for a scene and its reflection to be available in the same image.

3. Problem Statement

We address the problem of detecting and recognizing specular or mirror-like surfaces on natural scenes from a single image. We wish to do so, by introducing a novel set of features which provide a low-level signature that allows discriminating reflective materials from the non-reflective ones. A specular surface produces a systematic distortion of the surrounding scene being reflected off its surface. Such distortion depends on the object shape, the position of the observer and the appearance of the illuminant as shown by the differential characterization in Savarese *et al*. [13]. Research by [14] and [5] suggest that humans take advantage of such deformations for interpreting the shape of reflective surfaces. By using the results in [13] and as also noticed by Weidenbacher *et al*. [18], it is easy to show that there are regions within reflective objects that produce a highly distorted version of the reflected scene. These regions can be identified in curved reflective surfaces in proximity of either occluding contours of the object or regions where the difference of the two surface principal curvatures is high. We have called these regions *static specular flows* (SSF) as here the reflected scene presents a typical pattern akin to a flow. Thus, we argue that SSF can be used to provide a powerful signature for recognizing reflective materials. To the best of our knowledge, we are the first who use SSF for recognizing specular surfaces.

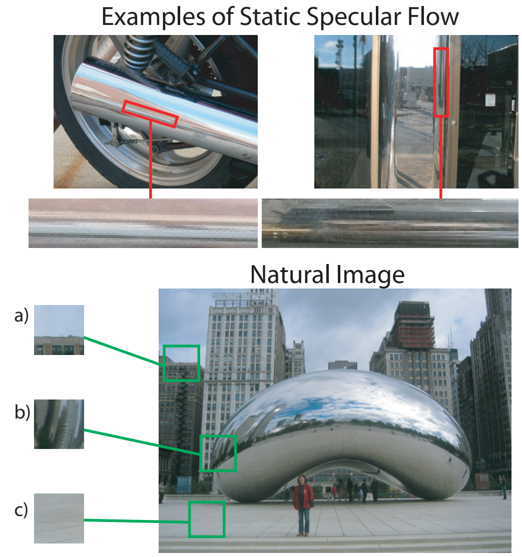

Figure 2. **Upper row:** Examples of static specular flow. **Lower row:**Regions from a natural scene. (a) local appearance of an edge patch. (b) local appearance of a curved specular region. (c) region of approximated uniform appearance.

The main goal of this paper is to provide tools for extracting SSF from images containing reflective surfaces. The first step is to extract regions characterized by anisotropic patterns regardless whether these are SSF or not. Figure 3 shows three regions from a natural scene. Figure 3(a) shows an anisotropic region where the predominant structure is an edge. Figure 3(b) shows a SSF. Figure 3(c) is an isotropic region of approximate uniform intensity. We propose a multi-scale approach for identifying anisotropic regions and discriminating them from isotropic textures. Next, we propose a new descriptor for characterizing SSF and a classifier that is able to discriminate SSF from anisotropic regions that are not SSF. Finally we show that SSF can be used for recognizing specular surfaces with high classification accuracy. We assess the performance of our approach with respect to a database comprising images containing reflective objects of various shapes, under arbitrary configurations and illumination conditions.

Notice that we do not make assumptions on the object shape as long as the object is curved enough to produce SSF. Also, the only hypothesis on the illuminating source (i.e., the surrounding scene) is that it must contain enough structure so that a deformed pattern may be identified. For instance, a scene composed of a homogeneous white background would not be able to produce any meaningful signal on the reflective surface. Finally, our approaches uses just a single image; thus, no additional information such as moving observers or multiple cameras are necessary.

To motivate more the use of SSF as a cue for recognition, let's think of a classifier that is agnostic to those cues. This naive classifier would use features extracted from the specular surface and the scene. Since there is an ambiguity between the specular object and the scene, one would expect that a naive classifier would have close to chance performance. We will validate this intuition experimentally later on this paper.

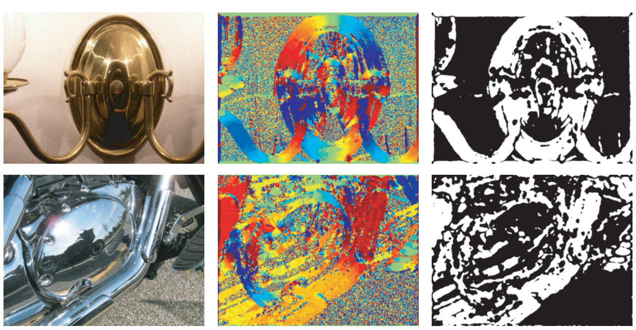

Figure 3. Examples of an orientation map and a thresholded confidence map. From left to right. Original image; aggregated orientation map, the color indicates orientations from $-\pi/2$ in blue to $\pi/2$ in red; the thresholded confidence map. We notice that in regions with 1d structure there is agreement on the orientation of the gradients. Notice that such agreement of orientations is coherent with the size of the 1D structure. In the other hand neither the wall nor the pavement have been smoothed and they present disagreement in the orientations as they should. The figure is best viewed in color and with pdf magnification.

4. Extracting Anisotropic Regions

The goal of Sections 4.1, 4.2 and 4.3 is to extract anisotropic regions. We believe that the SSF is a special case of such type of regions. We would like to segment anisotropic regions where the gradients show agreement on their orientation, for it is likely to find a SSF on such regions. One needs to pay special attention while computing the gradients to be able to capture small and large anisotropic detail simultaneously since we have no assumption on the scale of the specular surface. This might be difficult to do. Suppose that, the difference of offset of Gaussians are used to compute the gradients, then one might use bigger variances to capture large scale details. But the large variance also accounts for some degree of smoothing. The smoothing makes it difficult to discriminate between the highly texture regions and anisotropic regions with small details.

We distinguish two different scales: the scale of the reflected details and the scale of the anisotropic regions. Given that there is no assumption on the scene or on the geometry of the specular object the extraction procedure must be aware of these scales. We will return to the two-scale issue in Section 4.2. The recovery of anisotropic regions consists of the following steps: a) Extract the gradient orientation at several scales; b) Compute anisotropic regions at several scales(confidence map); c) Select scales of the orientation maps; d) Segment the image. See figure 4 for a scheme of the anisotropic region extraction.

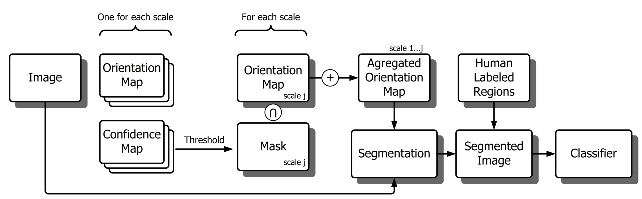

Figure 4. Schematic diagram of the extraction of anisotropic regions. The figure illustrates the proposed procedure to obtain highly informative regions.

4.1. Extraction of the Orientation of the Flows

We use the eigenvectors of the second moment matrix to compute the orientation of an image in a way similar to the work in Blobworld[1]. In [1] the second moment matrix (2MM) is approximated by 1:

$$M_\sigma(x, y) = G_\sigma(x, y) * (\nabla I)(\nabla I)^T \tag{1}$$

where G_σ is a separable binomial approximation of the Gaussian kernel with variance σ^2, I is the image and ∇ is the image gradient. The 2MM is computed using two scales: the natural scale and the artificial scale. The first is used to compute the gradients of the image,(∇I). The second scale (the variance of $G_\sigma(x, y)$) defines the size of the window used to integrate the gradients at a specific pixel. We use both scales during the scale selection step.

At each point of the image the 2x2 matrix $M_\sigma(x, y)$ is computed. Let λ_1 and λ_2 be the 2 eigenvectors of M_σ, where $\lambda_1 > \lambda_2$. Then the orientation of the gradients on the window used to build the 2MM is given by the orientation of λ_1. We compute the orientation at every pixel and at various window sizes. The output of this process is a multiscale orientation map.

The relative magnitudes of the eigenvalues of the 2MM can be used to characterize the structure of the region. One dominating eigenvalue corresponds to a region with 1d structure such as edges and flows; two comparably small eigenvalues correspond to uniform regions and two large eigenvalues correspond to regions with 2d or corner like structures. Equation 2 defines our measure of confidence that a region has 1d structure, where C is the confidence that the region has a 1d structure. C is computed over the support of G_σ, which in turn is function of the scale. This provides some information on the size of the flow, corner or other structure of the gradients. We use this measure to build our confidence map.

$$C = \frac{\lambda_1 - \lambda_2}{\lambda_1 + \lambda_2} \tag{2}$$

4.2. Scale Selection

To compute the confidence we held fixed the natural scale while we vary the artificial scale. Our confidence measure collapses when the integration window is bigger than the subjacent structure on the image. This step prevents the algorithm from smoothing out regions with high detail but with coherent orientation. The result is a set of confidence maps that capture 1d structure at multiple scales.

The multiscale orientation maps are merged into one aggregated orientation map that represents the orientation of the dominant structure across all scales. We build this aggregated orientation map from our confidence maps. We initialize the aggregated orientation map with the orientation map computed with the smallest integration scale. Then for each scale of the confidence maps, we threshold our confidence map to create a mask for the orientation map at such scale. Let $\Omega^{1\ldots j}$ be the aggregated orientation map from scale 1 to j, O^j the orientation map at scale j and C_τ the confidence map after thresholding. For each successive scale, we do $\Omega^{1\ldots j+1} = (\Omega^{1\ldots j} \wedge \neg C^j_\tau) \vee (O^j \wedge C^j_\tau)$. Fig. 3 is an illustration of the procedure.

Since the confidence map and the orientation map are computed using the same window size, we can view the confidence map as a measure of how coherent a region around a pixel is in terms of having a 1D or flow structure. If in a bigger scale the confidence for that pixel drops below the threshold then we should pick the gradient orientation

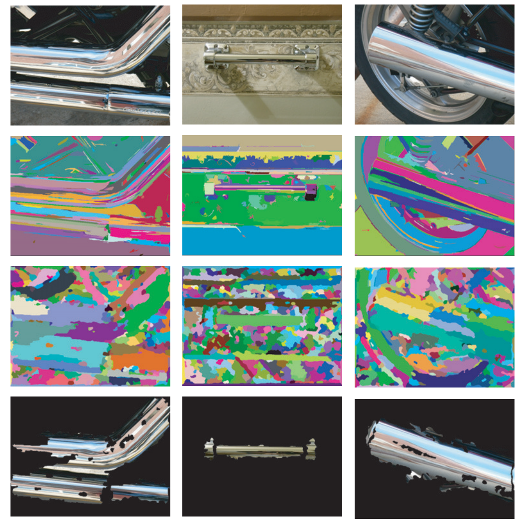

Figure 5. Comparison of a segmentation based on [4] and our algorithm. From top to bottom: first row, original image. Second row, shows the segmentation by [4]. We can see that the reflections on the specular surfaces are segmented into several thin and elongated segments. Third row shows segmentations produced by our dissimilarity function. Note that the segmented regions are more consistent. The fourth row, shows regions extracted from our segmentation algorithm. For clarity, on the fourth row we show the specular labeled regions that the classifier will see. The figure is best viewed in color and with pdf magnification.

of the previous (smaller) scale.

The multi-scale approach adds non-local information about how much structured a region is. This non-local information avoids smoothing textured regions while smoothing out the noise from anisotropic regions.

4.3. Graph Based Segmentation

Our segmentation procedure is based on the graph based segmentation presented in [4], albeit others important methods [15] could be used. The original algorithm [4] performs segmentation by constructing minimum spanning trees where the internal dissimilarity of the trees is less than the inter-tree dissimilarity. The original dissimilarity measure is the Euclidean distance on RGB space. In our case this is not the most appropriate dissimilarity measure since we wish our segmented regions to contain SSF. Experiments show that flow regions would be split in several small segments if the segmentation were solely based on pixel appearance.

To remedy this situation, we propose a new empirical dissimilarity function:

$$Dis_{ij} = (|C_i - C_j|)(\|A_i - A_j\| + \beta|\sin(O_i - O_j)|) \tag{3}$$

The difference on orientation, appearance and confidence is used for this dissimilarity function. Dis_{ij} measures the dissimilarity between pixel i and pixel j. A_j stands for the appearance of the pixel j. We want the dissimilarity cost to be big when the orientation of the two adjacent pixels is orthogonal to each other, hence the term $\sin(O_i - O_j)$. β is normalization constant. The sum of both terms is modulated by the difference on the confidence measure of both pixels. The outcome of the segmentation is a set regions with a wide range of sizes and shapes. This is due to the nature of the segmentation algorithm, which builds regions by merging pixels.

Figure 4.3 shows the original image, the segmentation based on appearance, the segmentation based on the Eq. 3 and the segmented regions. Notice that if appearance only is used each part of the flow are segmented independently.

In order to reduce the computational cost and the complexity of the modeling step, we keep the anisotropic segmented regions only. We do this by intersecting each segmented region with the "thresholded" confidence map and keeping the intersection only if there is more than 50% of overlap.

4.4. Descriptor

In this section we build our feature vector to describe each anisotropic region. The main issues that we need to cope with are: i) capturing the flow-like nature of SSF; ii) making the descriptor independent of the size of the region. The latter is both an artifact of the segmentation and a consequence of our lack of assumptions about the scene.

We can use classic texture recognition techniques to solve both problems. Texture recognition techniques [17, 7] based on models build out of histograms of textons have been shown to be a powerful discriminative tool. The main idea is to learn a 'visual' vocabulary (or code book) of the texture categories and model each category as the joint distribution of these words. Likewise, we learn a 'visual' vocabulary of the anisotropic regions and represent each of them as a histogram of words of such vocabulary.

In the Texture recognition literature a word of the vocabulary is called 'texton'. A texton is defined as the center of a cluster on the space of filter responses. We extend the filter bank used by Leung *et al.* [7]. Leung's original filter bank consists of 48 filters, partitioned as follows: 2^{nd} derivative of Gaussians at 6 orientations, 3 scales and 2 phases; 8 Laplacian of Gaussian; and 4 Gaussians. The scale of the filters ranges between $\sigma = 1$ to $\sigma = 10$. Here we increase the number of orientation to 18 and add a set of Gabor filters. The Gabor filters are elongated to capture the periodic pattern of SSF. The Gabor filter bank consisted of: 18 orientations, 3 scales and 3 wavelengths. The filters responses form the 282 dimensional feature vector.

We build the descriptor of a region as follows. We begin by convolving each image with our filter bank. For each pixel, the outputs of the filter are stacked and clustered using the standard K-means algorithm. In practice, we subsample each region for efficiency reasons. The centers of the clusters are the textons. Second, we build a histogram over the textons by taking the filter responses for each pixel and computing the closest texton on filter response space. Finally we normalize the histograms to sum up to one.

Our resulting features have high dimensionality, thus making the clustering problematic, if the data points are not sufficient. Thus, it is desirable to reduce the dimensionality of the feature vector. One way to do so is to use a rotational invariant filter bank. To achieve a rotational invariant filter bank, we select the oriented filter responses that have the same orientation as the local orientation with a tolerance of 30 degrees. The local orientation is computed from patches extracted from the region. The patches are extracted by following the dominant orientation of the region and sampling a patches at each step for a fixed number of steps. This selection process achieves rotational invariance and dimensionality reduction.

5. Classifying SSF

So far we have characterized anisotropic regions whether these are SSF or not. In this section we describe how to build models of SSF, and use them for classifying SSF regions versus non-SSF regions.

5.1. Model of the Anisotropic Regions

As described in Sec. 4.4, we represent each anisotropic region as a histogram over the learned textons. Textons are learnt from the training set images. We model the class of SSF anisotropic regions as the set of all the texton histograms that belong to regions that contain SSF. Likewise, anisotropic regions that do not contain SSF are modeled by the set of texton histograms from region that do not present SSF.

5.2. Classifier

We select boosted decision trees as our classifier. Decision tree (DT) [11] are classifiers that partition the feature space into cuboids. Although decision trees show issues of overfitting on noisy data, it is possible to combine several of these threes, with a low model complexity, into a committee in order to improve classification performance. Boosting [6] is a powerful technique for combining such classifiers. In boosting the classifiers are trained in sequence using a weighted form of the data. The weights force each new classifier to focus on 'difficult' examples. Given the set of histograms from SSF and non-SSF anisotropic regions, the boosted DT learn a linear combination of shallow trees. We run 100 iterations of boosting and set to two the maximum depth of each DT.

Configuration	100	200	300	400
OR	80.47%	82.03%	79.77%	81.41%
O	79.00%	77.68%	79.65%	78.62%
ORS	75.08%	77.83%	76.18%	77.34%
RS	72.10%	77.83%	70.88%	71.86%
Naive	56.90%	56.46%	56.11%	53.30%

Table 1. Accuracy comparison of different feature vectors.

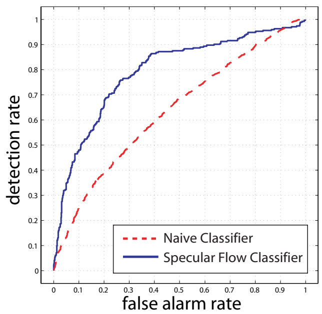

Figure 6. Comparison of a naive classifier and the classifier proposed. The red ROC curve refers to a naive classifier working on appearance segmented regions and ignoring orientation information. The blue curve refers to the performance of a classifier exploiting the specular flow cues. Notice that the naive classifier performs close to chance, while ours is much better.

5.3. Training set labeling

We need to build label for our anisotropic regions since our learning is supervised. To set the ground truth we asked a human subject to label as SSF all the regions that where found by the automatic segmentation described in subsection 4.3. These labels are used in the learning and testing stages of this work. We also asked the subject to outline, as best as possible, regions that he perceived as a specular surface. We allowed the subject to outline multiple regions per image and there where no restriction on the shape of the region.

5.4. Testing

In testing we follow a similar procedure as in learning. For each image we build a compound orientation map from a set of multiscale orientation and confidence maps. We automatically segment the image based on its orientation and appearance. We convolve each image with our filter bank and use the learnt vocabulary of textons to build a histogram of textons for each anisotropic region. Finally we proceed to classify each anisotropic region using the learnt decision trees. We evaluate the classification result using the labeling described above.

6. Experiments

6.1. Classification of SSF

In this section we assess the classification accuracy of SSF and the effects of different configurations of the descriptor over performances. Our dataset consists on 120 images of objects with specular surfaces. The images have been taken indoors, outdoors at close and far distances. The size of the images is 640 by 480 pixels. The database is divided in 73 training images and 47 testing images each of them containing several anisotropic regions to be classified as SSF versus non-SSF. The total number of anisotropic regions is approximately 5500; our classifier is learned and tested by using all the anisotropic regions extracted from the learning and testing images respectively. The first row of Table 1 shows percentage performances of the classifier. We use here the descriptor presented in Sec. 4.4 ”as is” with respect to different sizes of the vocabulary of textons.

Additional questions we would like to answer are: would performance be affected by the number of filter responses used to build the descriptor? How is this classifier working with respect to a more naive classifier which does not use SSF cues? Since the segmented regions can have arbitrary size, it might be possible for the representation of big region to contain specular and non specular code words; given that the last statement is possible, would the size of the region play a role on the discriminative power of the learned vocabulary? In the following we address these questions.

At that end, we try different configurations of our feature vector. The base feature vector that we modify is the one described on section 4.4. We call this feature vector RO for being oriented and with reduced dimensionality. O is a feature vector which has been oriented but all the filter responses have been retained. We build the O by performing a circular shift of the filter responses before clustering.They are aligned according to the local orientation of the patch from where they were sampled. ORS is similar to OR. The main difference is that the models are learned from smaller regions. We do this by splitting the segmented regions into subregions. The size of the split subregions is proportional to the size of the original region. If a region is too small it is left un-split. The naive classifier differs in two points to our classifier. First, the models are built from regions that have being segmented using appearance information only. Second, it ignores any information regarding the anisotropy of the region or any local orientation.

Table 1 summarizes our findings. First, notice that a naive classifier can only perform slightly better than chance. Also, notice there is no sharing of code words on big regions since both OR and O have better performance than ORS and RS across vocabulary sizes. As expected, the orientation of the feature vector and the dimensionality reduction improves the performance of the classifier. When we reduce the dimensionality of the O feature vector (O→OR) the performance increases. The performance decreases when we do not use the information of the local orientation (ORS→RS).

Figure 6 shows the ROCs curves of the naive classifier

	100	200	300	400
Error	4.48%	9.42%	11.23%	11.38%

Table 2. Examples of classified images from the control dataset

	100	200	300	400
Accuracy	74.39%	88.89%	85.37%	85.19%

Table 3. Accuracy results of the simple classifier of images containing specular surfaces.

and our classifier. The red curve shows the performance of the naive classifier. Notice that this classifier has close to chance performance. The blue curve plots the performance of a classifier using OR as feature vector and a vocabulary of 400 code words.

Figure 8 shows the ROC curves of a classifier using the oriented and reduced feature vector (OR) across several vocabulary sizes. A final control experiment is carried on an independent set of 57 images. In this set there are no specular objects. The images tend to have a significant number of edges. The goal of this control experiment is to evaluate how well the algorithm performs in the presence of anisotropic regions (edges) that are non-SSF. Table 2 summarizes the results of this experiment. Figure 9 shows some examples of classified images from the control dataset.

6.2. Recognition of Specular Objects

In order to assess the value of SSF as cue for recognizing specular objects we design a simple classifier to recognize the presence of a specular surface in a image.

We define a positive and negative test sets. The positive test set consists of 47 images where a specular object is present. The negative test set consists of 50 images without any specular object. For each image we define a score. The score of a positive image is given by $A_{SSF}/(A_{SSF}+A_s)$; where, A_{SSF} is the total area occupied by SSF regions within the specular object and A_s the area of occupied by the spe cular object in the image; this ratio gives the percentage of SSF detected within the specular object; The score of a negative image is given by $A_{SSF}/(A_{SSF}+A_I)$; where, A_{SSF} is the total area occupied by SSF regions within the negative image and A_I the area occupied by the image; this ratio gives the percentage of SSF detected in the negative image. Given the score S_i of a positive or negative image, we classify it as specular if $S_i > \theta$, and as non-specular, otherwise; θ is some learnt threshold. We learn θ by using a validation set composed of 30 images taken aside from the testing set.

The results of the experiment are summarized in table 3 and in figure 7. Notice that SSF appear to be a sound cue for recognizing specular objects.

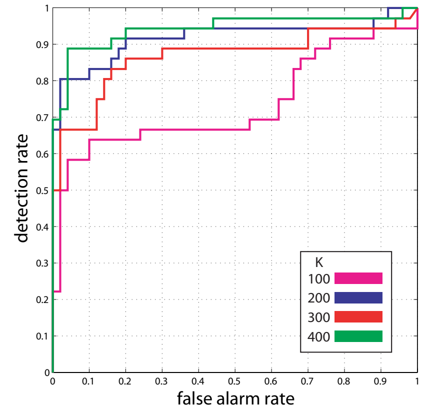

Figure 7. ROC curve illustrating the performance of the classifier of specular objects.

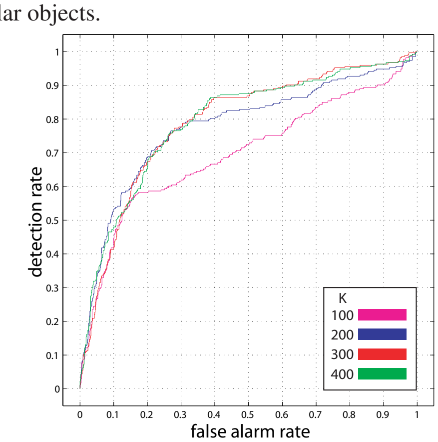

Figure 8. Effect of the codebook size on our classifier. ROC curve illustrating the classification performance using the OR feature vector across a range of codebooks sizes.

Figure 9. Examples of classified images from the control dataset. Notice that although there are several edge structures they are classified as non-specular.

7. Conclusions and future work

We have introduced a novel low-level feature for recognizing reflective objects from natural images. We have called this feature static specular flow (SSF) and shown that it can be used as a robust signature for identifying the presence of specular objects. We have proposed a multi-scale characterization of SSF and have shown that we can clas-

Figure 10. Recognition results. The classified regions have been outlined on the images. In blue color we outline the regions that the classifier and the human subjects have labeled as specular. On cyan we show the regions that the classifier has labeled as specular but the human subjects have labeled as non-specular. In red color we show the regions that the classifier and the human have labeled as non-specular. In magenta color we show the regions where the classifier has labeled a region as non-specular while the subjects have labeled it as a specular region. The columns have been organized in the following way: The first and third column present the regions that have been classified by our algorithm as specular and non-specular, respectively. The second column shows the misclassifications according to the human labels of a specular region. The fourth column shows the misclassifications according to the human labels of a non-specular region. Note that in the figures in the first and third row we classify correctly the specular surface in the presence of a textured background. Also note in the figure in the second and fifth row that we correctly detect regions on the occluding contours of the object. One could use those regions to outline the object. The figure is best viewed in color and with pdf magnification.

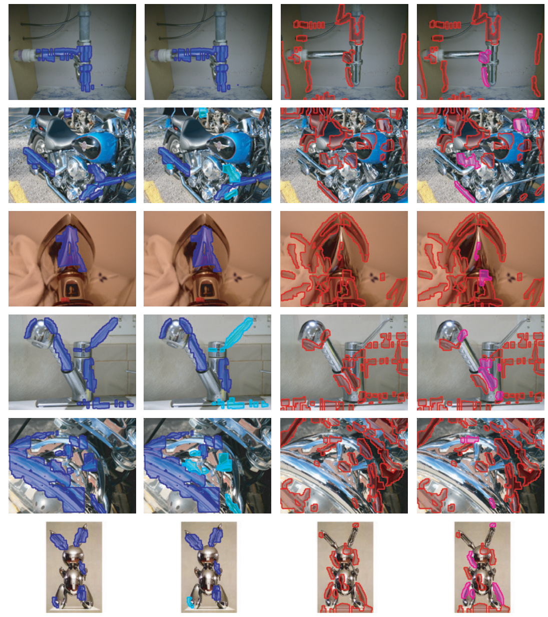

Figure 11. Recognition results continued. See figure 7 for description of the labels

sify SSF regions versus other anisotropic regions with high accuracy. We have demonstrated that SSF can be used to accurately discriminate specular objects from non-specular ones. We have also shown that a naive approach (which does not take advantage of SSF) would yield a much lower classification rate.

We have just tap into a set of possible new features that could be used to characterize specular objects. The issue of localizing a specular surface given the detected SSF is subject of our future investigation. Additional work is also needed in order to integrate SSF with other cues as well as interpret a reflective surface at different level of abstraction.

References

[1] C. Carson, S. Belongie, H. Greenspan, and J. Malik. Blobworld: Image segmentation using expectation-maximization and its application to image querying. *PAMI*, 24(8):1026–1038, 2002.

[2] O. G. Cula and K. J. Dana. Compact representation of bidirectional texture functions. In *CVPR*, 2001.

[3] R. Dror, E. Adelson, and A. Willsky. Recognition of surface reflectance properties from a single image under unknown real-world illumination, 2001.

[4] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient graph-based image segmentation. *IJCV*, 59(2):167–181, 2004.

[5] R. W. Fleming, A. Torralba, and E. H. Adelson. Specular reflections and the perception of shape. *J. Vis.*, 4(9):798–820, 9 2004.

[6] Y. Freund and R. E. Schapire. Experiments with a new boosting algorithm. In *ICML*, pages 148–156, 1996.

[7] T. Leung and J. Malik. Representing and recognizing the visual appearance of materials using three-dimensional textons. *IJCV*, 43(1):29–44, 2001.

[8] F.-F. Li and P. Perona. A bayesian hierarchical model for learning natural scene categories. In *CVPR*, pages 524–531, 2005.

[9] K. McHenry, J. Ponce, and D. Forsyth. Finding glass. In *CVPR*, pages 973–979, 2005.

[10] M. Osadchy, D. Jacobs, and R. Ramamoorthi. Using specularities for recognition, 2003.

[11] J. R. Quinlan. Bagging, boosting, and c4.5. In *AAAI/IAAI, Vol. 1*, pages 725–730, 1996.

[12] S. Roth and M. J. Black. Specular flow and the recovery of surface structure. In *CVPR*, pages 1869–1876, Washington, DC, USA, 2006.

[13] S. Savarese, M. Chen, and P. Perona. Local shape from mirror reflections. *Int. J. Comput. Vision*, 64(1):31–67, 2005.

[14] S. Savarese, L. Fei-Fei, and P. Perona. What do reflections tell us about the shape of a mirror? In *APGV*, pages 115–118, New York, NY, USA, 2004.

[15] J. Shi and J. Malik. Normalized cuts and image segmentation. *PAMI*, 22(8):888–905, 2000.

[16] R. Swaminathan, S. B. Kang, A. Criminisi, and R. Szeliski. On the motion and appearance of specularities in image sequences. In *ECCV*, pages 167–172, May 2002.

[17] M. Varma and A. Zisserman. Classifying images of materials: Achieving viewpoint and illumination independence. In *ECCV*, volume 3, pages 255–271, May 2002.

[18] U. Weidenbacher, P. Bayerl, R. Fleming, and H. Neumann. Extracting and depicting the 3d shape of specular surfaces. In *APGV*, pages 83–86, New York, NY, USA, 2005.