Robust Object Recognition Using a Cascade of Geometric Consistency Filters

Yuetian Xu and Richard Madison
Draper Laboratory
555 Technology Square
Cambridge, MA 02139
(yxu,rmadison)@draper.com

***Abstract*—Bag-of-words is a popular and successful approach to performing object recognition. Its performance is limited by not considering relative geometry information. This limitation is particularly stark when there is significant image noise. We propose a "bag-of-phrases" model which extends bag-of-words by enforcing geometric consistency through application of a "geometric grammar" in a filter cascade. Experimental results on a computer generated dataset show increased robustness to clutter and noise as demonstrated by more than two orders of magnitude reduction in false positives compared with bag-of-words.**

I. Introduction

Fast, reliable object recognition has many applications, including robotic navigation and user interaction. A popular approach for image-based object recognition is called "bag-of-words" [4], [5], [14], [17]. The method is named for a text retrieval method that infers the topic of a document from the histogram of certain key words associated with the topic. By analogy, one may detect an object in an image from the presence of "visual words" associated with the object. The visual words, typically called "features", represent image patches with interesting content. An object recognition system builds an index listing which features should appear in which objects. It detect objects in novel imagery by extracting the features from the image and checking which objects contain that set of features. This method allows for high retrieval speed and in many cases achieves fairly accurate results.

The accuracy of the "bag-of-words" approach depends on features in an input image being matched correctly to those in the index. This assumption breaks down in environments with significant noise or clutter. Feature matchers decide that a novel feature matches a referenced feature if the value of some similarity metric between them exceeds a threshold. With noise and clutter, it becomes increasingly difficult to find a threshold that cleanly separates matches from non-matches (see Fig. 1). As a result, a feature matcher invariably accepts some incorrect matches and rejects some correct matches, reducing the accuracy of the "bag-of-words".

One way to help disambiguate correct and incorrect matches is to observe that correctly matched sets of features should obey certain geometric constraints, while incorrectly matched points do not. In its simplest form, features representing a single object should appear *close* to each other. Features appearing by themselves probably represent incorrect feature matches. We can further separate correct and incorrect matches using more complicated geometric constraints.

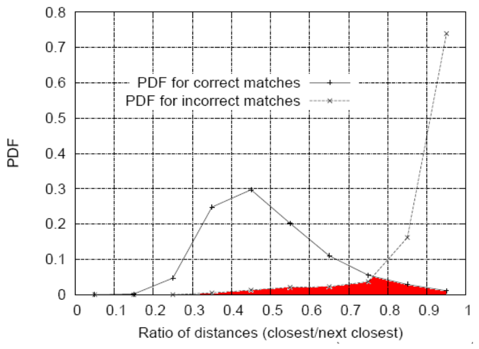


Fig. 1. Histograms of (SIFT [11]) feature matcher) comparison metric for correctly and incorrectly matched, noisy features show that correct matches tend to be similar in value, but no threshold on the metric cleanly separates the two groups.

In this paper, we apply a set of geometric constraints to reinforce "bag-of-words" object recognition in noisy and cluttered imagery. We begin with an overview of relevant, prior work. Then, we borrow from natural language processing the idea of consistency within phrases and enforce semi-local geometric consistency between visual words. We show how this "bag-of-phrases" extension to bag-of-words can be computed efficiently using a cascade of filters. Finally, we present experimental results demonstrating more than two orders of magnitude reduction in false positives from utilizing this approach.

II. Previous Work

There is significant, early precedent (more than a decade ago) for using geometric constraints to improve reliability over matching simple, individual features. References [7], [8] modeled objects as polyhedra, matched individual faces (features), and then applied several geometric constraints to the distance and angles between faces. Reference [13] demonstrated better separation between correct and incorrect matches by calculating the angles between nearby interest points and eliminating sets of matches that did not conform to these angles. Reference [19] used distance between features as a constraint for matching with the closer neighbors of the feature consider having greater weight. These early uses of geometric constraints provided improvements over local image feature matching, but the features utilized were not very robust. As a result, much of this work was limited to fairly simple models and/or restricted domains *e.g.*, recognition of industrial tools in binary images.

In recent years, robust and descriptive *local* feature detectors such as SIFT [11] have improved accuracy in many computer vision applications including object recognition. A "bag-of-words" approach using these local features provides great speed and fairly good accuracy. However, the ultimate accuracy of "bag-of-words" is limited by its use of only local information. Context and other forms of non-local information play an important role in human object recognition. Reference [16] notes humans can successfully segment a scene and identify objects in low resolution imagery even if any given object is too ambiguous to identify by themselves.

Several modern approaches take into account more than just local information.

One approach uses larger features that cover a larger area but have reduced weight in areas further from the center. This incorporates more image context (reducing probability of spurious match) but also allows for greater blurring/deformation further out. Reference [2] is a good example of this approach. This takes into account some information about geometry of neighboring regions and partially imitates foveation of the eye. However, this approach is sensitive to the scale of the image and is thrown off by occlusions.

Another approach is to utilize spatial histograms, with various extensions [6] to probabilistic Latent Semantic Analysis (pLSA) [9]. PLSA models the co-occurrence probability of words and documents as a mixture of conditionally independent distributions. ABSolute position pLSA (ABS-pLSA) adds a location term to the standard pLSA model. Translation and Scale Invariant pLSA (TSI-pLSA) expands on this by adding in position relative to centroid of a sub-window to produce translation and scale invariance. This approach is computationally efficient and considers the rough geometrical arrangement of features. However, it is not rotationally invariant and can be sensitive to histogram boundaries.

Yet another approach is to accept a feature match between two images only if there are several other matches nearby. For instance, if a feature in one image seems to match a candidate in a second image, but few of the feature's k nearest neighbors in the first image match to the candidate's k nearest neighbors in the second image, then the match probably is spurious. This heuristic is appropriate for detecting clusters of features representing a single object. The heuristic was successfully applied in Video Google [14]. However, it does not operate as well in noisy or cluttered environments where the nearest k features include very few correct matches. If k is made large to include more correct matches, accuracy becomes sensitive to the value of k and computation becomes expensive.

We propose an alternative, called bag-of-phrases. It revisits the original approach of geometric constraints by applying it to modern robust features such as SIFT. This produces better invariance and discriminative properties compared with other modern approaches to capturing non-local information. It is able to recognize objects, and perform general image correspondence, robustly even in noisy and cluttered environments.

III. Geometric Grammar

A. Natural Language Processing (NLP) Parallel

Our goal in applying geometric constraints is to improve the *semi-local* geometric coherency of the image. Natural language processing systems performing automatic text generation or translation constrain bigrams and trigrams of words [3], [10] to generate not just words but viable phrases. The constraints are *semi-local* in that they apply to a few words, rather than the letters within one word (local) or the full text (global). By analogy, we apply geometric constraints between small sets of "visual words" to eliminate words that do not form viable "visual phrases".

B. Differences from NLP

Text and imagery differ in the notion of semi-locality. Text is inherently one dimensional, with each word having at most two neighbors. Some text processing systems constrain consecutive quadruples and quintuples of words. Imagery is two dimensional, with each visual word having an arbitrary number of neighbors. The number of possible combinations of N nearby features grows intractably quickly, leading us to consider only pairs and triplets of features in our semi-local constraints. Our experimentation (Section V-B) showed that constraining feature pairs provided significant improvement over using single features by eliminating all false positives. Use of triplets to provide further filtering is therefore unnecessary.

Natural language processing applications and our proposed model also differ in the use of the pairs of words/features. The grammaticality of a given bigram cannot be explicitly modeled and must be implicitly captured in the frequency distribution. By contrast, we can calculate metrics of geometric consistency between two pairs of features. These measures are independent and can be considered to be equivalent to grammar rules. In aggregate, they form a visual geometric grammar.

C. Specialized Grammar for SIFT

For each feature in an image, a given feature matcher generates a hopefully-invariant feature descriptor (signature distilled from the pixels of an image patch) and some presumably-transient geometric information (such as the 2D location of the patch). The "bag-of-words" approach is concerned solely with the descriptor and discards the geometric information. However, the geometric information can be used to construct useful semi-local constraints, such as the distance between neighboring features. Such a "geometric measure", defined in one image, becomes a grammar rule to constrain potential feature matches in another image. Multiple measures form a geometric grammar.

We will focus on application to SIFT features. SIFT is chosen due to its demonstrated performance in comparison with other local feature descriptors [12]. In the case of SIFT, the geometric information extracted for each feature includes its absolute location and orientation in the image as well as its scale.

D. What Makes a Good Grammar?

A good grammar maintains the *invariance* properties of a given feature type while improving *discrimination* as much as possible.

Individual grammar rules can support this by constraining only those elements of transient geometric information that have the same invariance properties as the feature descriptor. For example, the SIFT feature descriptor is invariant to scale. If an object scales, the feature descriptors of its features can still match unscaled versions, but the distances between features all scale together. A grammar rule may constrain the ratio of distances between features, which is scale invariant. It should not constrain absolute distance between two features, which changes with scale.

In aggregate, the rules comprising a grammar should also leverage every last bit of information encoded in the geometric relationship between features. In other words, the geometric measures constrained by the set of grammar rules should form a spanning set of sorts over the geometric information created by the feature matcher. From a computational point of view, the set should preferably be a minimal spanning set.

E. A "Bag-of-Phrases" Grammar

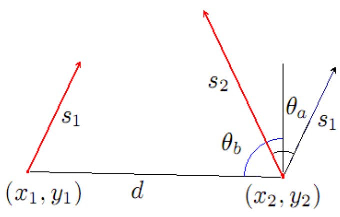


Fig. 2. SIFT features s have geometric information to include a scale $|s|$, orientation θ, and position (x, y). For two SIFT features, these quantities can be represented by two vectors (s_1 and s_2). We can develop geometric rules to constrain the relative angles and scales of s_1, s_2, and vector d that connects their base.

We developed grammar rules around a set of four geometric measures that fully leverage the geometric information available from a pair of SIFT features (illustrated in Fig. 2) while preserving the invariance properties of the SIFT feature descriptor. We use the scale $|s|$ and orientation θ of each feature and the vector d between the positions of the two features.

The absolute location of positions of the two features are irrelevant (features may shift around the image), so only the relative positions are used, reformulated as a vector d connecting the two.

1) Ratio of feature scales: $\frac{|s_1|}{|s_2|}$.
2) Ratio of average feature scale to baseline: $\frac{|s_1|+|s_2|}{2|d|}$.
3) Signed difference in feature orientation: $\theta_a = \theta_1 - \theta_2$.
4) Signed difference in average feature orientation and baseline : $\theta_b = \frac{\theta_1+\theta_2}{2} - \theta_d$.

These four measures are all invariant to scale and in-plane rotation, thus maintaining the invariance properties provided by the SIFT feature descriptor. Feature scales $|s_1|$ and $|s_2|$ and the baseline length $|d|$ are invariant to in-plane rotation. Further, they all scale together as the object scales, so their ratios are scale invariant. The feature scale and the baseline length scale together and maintain their ratio even if they may be in different units. The orientations of the two features and the baseline are scale invariant and, because the features and baseline rotate together, differences between the orientations are rotation invariant. In the second and fourth measures, we average feature scale and orientation to avoid favoring one feature, but the same logic demonstrates that invariance is preserved.

The four measures also maximally use the available geometric information. We know this because, once we fix enough geometric information to define an object pose, we can reconstruct the remaining geometric information from the measures. An object can translate (2D), rotate, and scale arbitrarily, so we must fix four parameters to define an object's pose. Let us fix the two feature positions. Then we can calculate the length $|d|$ of the baseline between them, use the second measure to calculate the average feature scale, and combine with the first measurement to determine the scales of the two features. Similarly, we can calculate the orientation of the baseline, use the fourth measure to calculate average feature orientation, and combine with the third measure to determine orientation of each feature.

IV. Algorithm

We implemented a bag-of-phrases algorithm to detect objects in noisy imagery. The algorithm extracts SIFT features from both *template* images containing just the objects to be detected and *scene* images containing noise, clutter, and occassionally the objects to be detected. It matches template and scene features using SIFT descriptors, as a bag-of-words algorithm would. Then it applies a cascade of filters, representing a "geometric grammar" developed in Section III-E, to eliminate false matches. Each stage of filtering uses a voting mechanism to eliminate many of the remaining false matches. In this section, we explain the options for voting mechanisms, the idea of a filter cascade, and the specific voting and cascade parameters used in our implementation.

A. Voting

Each filtering stage inputs a set of proposed matches between features in a template and a scene. In theory, features *close* to each other in an image are likely to be part of the same object. Further, if a set of scene features is *close* together, then the correctly matched template features should exhibit some sort of *consistency*, suggesting that they also represent a single object. If the scene feature is falsely matched, then few of its neighbors will have *consistent* matches. These observations can be used to detect and eliminate some false matches. Each filter stage evaluates each scene feature by counting how many of the scene feature's *close* neighbors are matched with *consistent* template features. If the count represents a sufficiently low percentage of the scene feature's close neighbors, then the scene point is incorrectly matched, so its match is invalidated. We now explore two different voting mechanisms (nearest-neighbor and distance-based), which apply different notions of *close* and *consistent*, and why we select distance-based voting for our algorithm.

1) Why Not Nearest Neighbor Voting?: The nearest neighbor (NN) voting mechanism is illustrated in Fig. 3. Under NN voting, a feature's k nearest neighbors are considered *close*. Two matches are considered *consistent* if their features are *close* in both the template and the scene. This reflects the fact that features on an object should be close together in any view of the object. To vote on a proposed match between a template feature F_T and scene feature F_S, we count how many of the k features nearest F_S in the scene are matched to the k features nearest F_T in the template. If these neighborhoods do not include at least m matches, then the (F_T, F_S) match is invalidated. Video Google [14] used this method as a successful, light-weight heuristic to augment "bag-of-words". They used a value of $k = 15$ and $m = 1$.

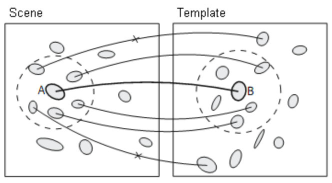


Fig. 3. Illustration of NN spatial consistency voting. In this case, a match (A,B) is checked for consistency. The nearest $k = 5$ neighbors of each are inspected. Each match connecting members of the two neighbor sets casts a positive vote. In this case, three other matches cast votes in support of (A,B). If there are fewer than m votes, (A,B) is rejected. This diagram is an excerpt from Video Google [14].

The nearest neighbor approach runs into problems, however, in noisy and cluttered imagery. There, feature detectors generate dense clouds of mostly spurious features, filling in between correct matches so that a correct match's k nearest neighbors contain few if any correct, consistent matches. As an example, nearest neighbor voting fails when applied to data generated as described in Section V-A. This problem can fixed by increasing the number of neighbors around the feature. Unfortunately, the cost of sorting makes it impractical to use for large values of k.

2) Distance-based voting: The distance-based voting mechanism is illustrated in Fig. 4. We define a neighborhood of some radius in the template image and scale that radius by the maximum credible magnification of the scene relative to the template to produce a maximum radius R for the same neighborhood in the scene. Scene features within radius R of each other are considered *close* together. In instance recognition, the template represent a single object, so we choose the neighborhood radius to include the entire template. The resulting R represents the maximum credible size of the object in the scene. For image-to-image matching, the template neighborhood can be restricted to a smaller radius (perhaps based on the average scale or spatial extent of the k nearest neighbors) with R again corresponding to the maximum reasonable size of that neighborhood in the scene image.

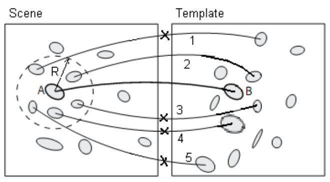


Fig. 4. Illustration of distance-based spatial consistency voting for proposed match (A,B). Each match (C,D) for features C within radius R of A in the scene is considered. If the pair of features A and C in the scene and the pair of features B and D in the template are consistent according to the "geometric grammar", then feature C casts one vote in support of feature A. The number of votes is normalized by number of neighbor features C that could have voted to generate an approval rating. If the approval is lower than a threshold, then match (A,B) is rejected

Using large R to define *close* ensures that, even under noise and magnification, a feature's correctly matched neighbors will be *close* and thus count in favor of the feature. Unfortunately, many spurious matches to noise and nearby clutter will also be *close* and contribute to the voting. To reduce the number of these incorrect votes, we expand the definition of *consistent* matches in two ways. First, we require that the pairs of matches satisfy one or more "geometric grammar" rules. Thus, for instance in Fig. 4, numbered arcs show 5 proposed matches that may support match (A,B), but only one (arc 2) connects features with the same relative scale and orientation as features A and B. This more restrictive definition of *consistency* eliminates many spurious matches. Second, we specify that, if several neighbors of a scene feature match to the same template feature, they count as only one *consistent* match. Thus, a high vote count requires the presence of neighbors representing several unique template features, not just several copies of a single feature generated by noise or clutter. A template may still have several features with identical signatures, for instance several identical windows, because those copies occupy different physical locations in the template and are thus unique features and each can have a separate vote.

B. Grammar Cascade

A previous section identified four "geometric grammar" rules that we can apply to filter out false matches. As with feature matching, each rule consists of a calculation that produces high values for correct matches and low values for false matches, followed by a threshold that never cleanly separates correct and incorrect matches. Instead of relying on such separation, we choose low thresholds that reject egregiously incorrect matches but accept nearly all correct matches along with many false matches. The idea is that most bad matches are likely to egregiously violate at least one of our four grammar rules, so there is no need to carefully tune the threshold on any one rule.

Viola and Jones [18] demonstrated remarkable success in face detection, by organizing such a group of weak classifiers into a cascade of filters. Each filter in the cascade filters out as many false matches as possible while preserving all of the true matches, so that in the end only the true matches remain. Organizing the filters as a cascade rather than a battery also reduces computation as each new filter applies to a smaller set of potential matches. Expensive filters can be saved for later in the cascade, once many false matches have been eliminated.

Operating filters as a cascade also allows us to apply the same "geometric grammar," rule multiple times to successively eliminate more false matches. As described earlier, each filter evaluates each proposed feature match, retaining it if enough neighbors have consistent matches. Correctly matched features should receive one vote from each nearby correct match plus some small number of votes from nearby incorrect matches that by chance conform to the grammar rule. Incorrect matches should receive a small number of chance votes from incorrectly matched neighbors. Given a large number of spurious matches, some incorrectly matched features will receive enough chance votes to masquerade as correct matches. However, if we apply the same filter a second time to the surviving points, the remaining, smaller pool of incorrect matches provides fewer chance votes, so that points that previously suvived based on chance votes no longer have enough support and are rejected. This process can be repeated to eliminate more incorrect matches, limited only when sets of spurious features are mutually supporting by being geometrically consistent with each other. But unless they are consistent according to all of the grammar rules, running a full set of filters will eliminate some, so that a second round of filtering will find them without the mutual support and reject them as well. If a set of spurious feature matches satisfies all of the geometric grammar rules and so remains mutually supporting, then by our definition it is an instance of the object we wish to detect, even if it is spurious.

C. Final Algorithm

Here we list the parameters of our algorithm.

We used distance-based voting with $R = 100$ pixels. Most of the objects in our test scenes had an approximate radius of 120 pixels or less. Choosing R smaller than this does not significantly reduce the number of correct neighbors available for matching but does reduce the number of spurious neighbors.

We used the four geometric measures described in Section III-E. For each, we calculated the geometric measure g for a pair of features in the scene and the matched pair of features in the template. We defined a "distortion factor" $\max(\frac{g(template)}{g(scene)}, \frac{g(scene)}{g(template)})$, which is 1.0 when g is the same for the feature pairs in scene and template and grows as the pairs differ. Experimentally, for a pair of genuine feature matches, this distortion factor can be as large as 1.2.

We used a cascade with three stages of filtering by voting.

The first stage applies the first two grammar rules, requiring feature pairs to have consistent ratios of feature scales to each other and to their baseline. These two tests were combined to save computation, because they involved the same memory accesses and intermediate calculations. This filter stage multiplied the distortion factors for the two measures and thresholded the product at 1.4, above the largest expected value for genuine matches. To survive the filter, a feature required an approval rating of 20%, meaning it is geometrically consistent with at least 20% of the other features within distance R. The threshold is set low so that even when most features are spurious, correct matches have enough votes to survive the filter.

The second stage applies the remaining two grammar rules, requiring feature pairs to have consistent difference in feature orientations to each other and to their baseline. Again, the two tests were combined, and the product of their scores was thresholded at 1.4. Features required an approval rating of 40% to survive to the next filter. This threshold is higher than in the first filter stage because the first stage eliminated many spurious matches. The set of correct matches represents a larger fraction of the reduced set of proposed matches.

The third stage repeats the first stage but with approval rating cutoff at 40% instead of 20%. Repeating the initial filter eliminates matches that survived previously based on support from spurious features that have since been eliminated. The increased threshold is viable now because many of the of remaining matches are correct matches. Experimentally, no additional filtering stages were necessary, as the vast majority of incorrect matches had been eliminated.

After all this filtering, a RANSAC variant called MLESAC [15] was used to recover the transform (rotation, translation, and scale) relating the features in the template with the matched features in the scene. The transformed template features could then be used to draw a fairly accurate bounding box around the region detected. The MLESAC procedure also threw out some of the occasional, false matches that had survived filtering.

V. Experiments

We performed three sets of experiments to evaluate the ability of the bag-of-phrases approach to detect objects in noisy imagery. We tested how well the algorithm can reject false feature matches, distinguish between similar geometric configurations of features, and use geometric configurations to localize detected objects within images.

A. Experimental Setup

We tested bag-of-phrases object detection on synthetic, noisy video from a camera flying above vehicles in a cluttered, urban scene. Synthetic video has the advantages of (1) having associated ground truth, particularly regarding the orientation of the objects of interest, and (2) being more readily available than real video from airborne platforms flying low across crowded urban areas.

To synthesize the video, we developed an application based on the OpenSceneGraph library (an Open Source object rendering/simulation framework built on top of OpenGL) [1]. The application reads CAD models and renders images containing them from camera viewpoints along a user-specified trajectory. The cluttered, urban backdrop for the imagery was a high resolution, textured CAD model of the Technology Square area in Cambridge. The application rendered scenery using this model and several copies of CAD models of interesting vehicles (Humvee, Mig, Abrams, and Apache) downloaded from the Internet. These vehicles served as the objects of interest to be detected. After rendering the scene, the application added salt-and-pepper noise (corrupting up to 15% of the pixels) to some of the imagery to simulate one type of imaging noise that may be found in a real camera.

The rendering application was also used to generate templates imagery of each of the the models. The templates were generated by taking snapshots of the CAD vehicles at 10° intervals of out-of-plane rotation against an uniform background. Illumination conditions for template generation were significantly different compared with that of the synthesized video.

B. *Results - False Match Rejection*

The bag-of-phrases approach showed good ability to reject false feature matches that would contribute to false detections of objects. An example result is depicted in Fig. 6. Each sub-image shows a template of a Mig (right side) and a cluttered scene containing that Mig. SIFT matches between scene and template features are indicated by red and cyan lines, for correct and incorrect matches, respectively. The left column shows the output of SIFT feature matching using different matching thresholds. This would be the input to a naive "bag-of-words" approach. The false matches completely obscure any signal that the true matches can provide. Each row shows the gradual elimination of false matches over the three cascaded filtering stages. The final image shows the features remaining after the algorithm's final transform recovery. The "geometric grammar" filtering has eliminated all false matches, even when initial SIFT matching used a ridiculously permissive threshold to avoid missing any noise-distorted matches on the actual object.

Table I summarize numerically the number of SIFT feature matches before and after filtering. Without filtering, there are always at least 47 false positives regardless of threshold with no ability to eliminate them. A "bag-of-words" approach would be hard pressed to find the signal in that noise. After filtering, however, all false positives had been eliminated, and except with threshold 0.95, all all true positives had been retained. Setting SIFT threshold to 0.95 produced two additional true matches, one of which survived filtering. Overall, the goal to eliminate as many false positives as possible while preserving true positives was fulfilled.

Table II shows the cost of this improved performance - increased processing time. At lower SIFT thresholds, where "bag-of-words" might operate in low-noise imagery, the additional cost of filtering is negligible. As the SIFT threshold increases to detect true positives in increasingly noisy imagery, filtering time increases rapidly because of the proliferation of spurious SIFT features. Big variations in time for MLESAC is due to occasional numerical instability.

TABLE I
SIFT PERFORMANCE COMPARISON BEFORE AND AFTER FILTERING (TP AND FP DENOTE TRUE POSITIVES AND FALSE POSITIVES)

	Naive			Filtered		
SIFT Thresh	Matches	TP	FP	Matches	TP	FP
0.8	53	6	47	6	6	0
0.85	130	6	124	6	6	0
0.9	270	6	264	6	6	0
0.95	566	8	558	7	7	0

Experimentation on four videos with a matching threshold of 0.8 generated showed a false positive rejection rate of over 99% while false negative rate tended was generally less than 10%. This approach had problems with detecting objects less than approximately 40 pixels across. At that point, reliability of feature detection degrades significantly due to pixelization effects and image noise.

C. *Results - Distinguishing Among Viewpoints*

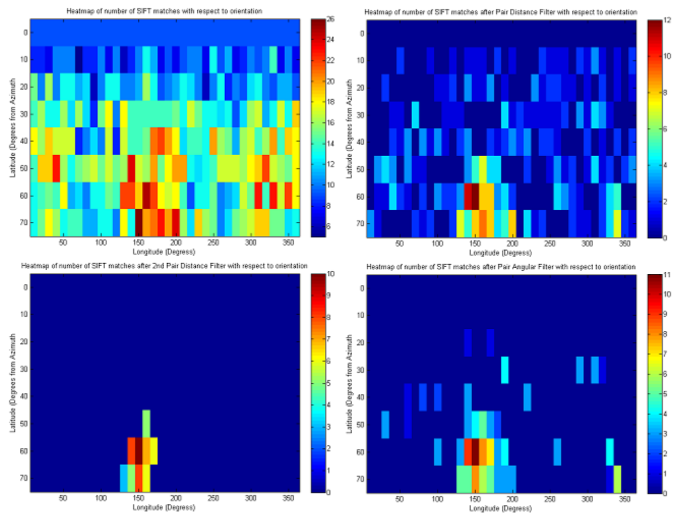
To evaluate how well "bag-of-phrases" helps distinguish between similar objects, we tested its ability to distinguish the same object at slightly different orientations. Rotating an object out-of-plane slowly changes the set of SIFT features that appears on the object and more quickly changes the geometry between the SIFT features. Theoretically, the permissive thresholds on grammar rules should overcome some geometrical distortion as the object rotates out-of-plane, but it should reject more matches as rotation increases so that the template representing the most similar rotation will produce the largest set of feature matches.

We tested the algorithm on a noisy image of a Humvee and templates showing the Humvee at a range of orientations. Fig. 5 shows a heatmap representation of the number of SIFT feature matches between the image and the various templates across stages of filtering. Initially, many templates show large numbers of matching SIFT features. There is no clear indication which template is actually represented in the image. As the filtering stages are applied, spurious matches disappear, leaving a significant number of matches only near the correct orientation. Only a small set of neighboring orientation templates detect the object. This leads to three observations. First, the correct orientation has the largest share of the SIFT features, so the correct match is detected most strongly. This contrasts with the situation before filtering, where the template with the most supporting SIFT features did not represent the correct object orientation. Second, because templates from several orientations detected the object, the geometric constraints are robust enough to detect the object rotating out-of-plane using a smaller set of templates. This would save time if the goal is to detect an object at any out-of-plane rotation. Third, the distribution of detections at different

TABLE II
FILTERING SPEED BREAKDOWN (IN SECONDS)

SIFT Threshold	Matching	Pair Dist	Pair Ang	Pair Dist2	MLESAC	Total
0.6	0.0341	0.0024	0.0021	0.0014	0	0.04
0.7	0.0367	0.0029	0.0027	0.0020	0	0.0442
0.8	0.0345	0.0059	0.0028	0.0019	0.0092	0.0542
0.85	0.0360	0.0232	0.0026	0.0020	0.3404	0.4043
0.9	0.0352	0.0928	0.0025	0.0019	0.0093	0.1418
0.95	0.0356	0.4020	0.0033	0.0028	0.3453	0.7889

Fig. 5. Heat map representation of the number of SIFT matches among the various orientation (left and right edges wrap) of a Humvee model after successive stages of filtering starting clockwise at the top left with no filtering. Note the highest ranking viewpoints turn into a more coherent cluster after more filtering.



angles can be used to interpolate the actual orientation of the object, to support landmark navigation, 3D localization, mapping, motion estimation, and a variety of other applications. Preliminary experimentation have shown that sub-degree accuracy can be achieved under certain circumstances with templates at 10° intervals.

D. *Results - Object Localization in Image*

The “bag-of-words” approach can detect whether an object is in a (low noise) image, but lacking geometry, it can only localize the object centroid. This is inaccurate when only part of the object is visible, or multiple copies of the object appear in the same image. By contrast, “bag-of-phrases” has an explicit geometrical model and can identify the bounding box of an individual object based on the locations of a few features matches. Fig. 7 show capability to localize multiple objects in the same frame as well as a partially occluded object. Fig. 8 shows some more examples, along with the best matched templates. In-plane rotation between the template and the scene is irrelevant, as the SIFT matcher and geometric rules are invariant to in-plane rotation. Occasional failures, such as confusing a grayed-out canopy with a Humvee, are visually understandable and point to the utility of higher levels of processing and/or human oversight.

Fig. 7. Multiple instances of objects can be accurately localized individually. Occluded parts of objects can still be inferred.

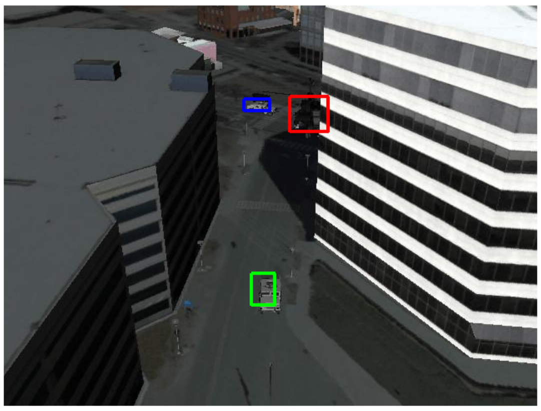
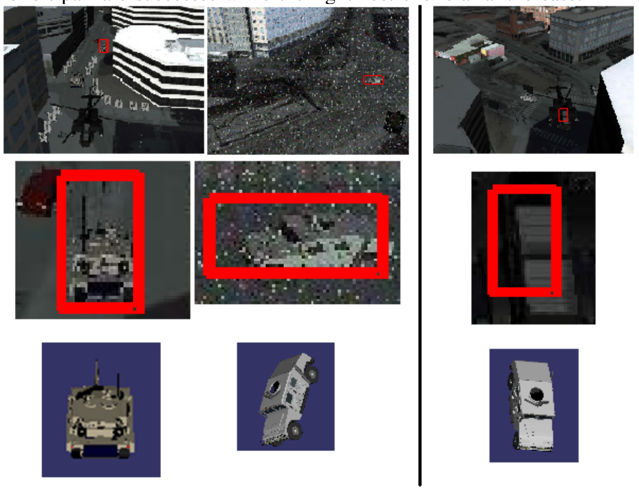


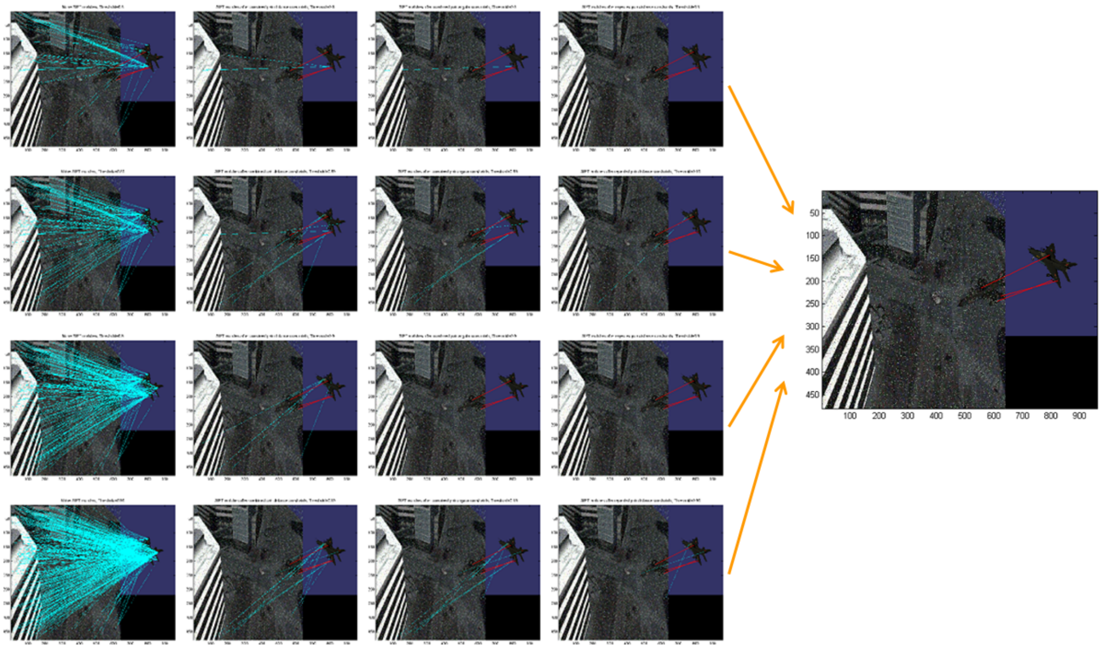
Fig. 8. Accurate localization can be achieved even in heavy noise. Top row shows the input test images. Middle row shows an enlarged version of each detection area with bounding box. Bottom row shows each matching template. The left pair are successes while the rightmost one is a failure case.



VI. CONCLUSION

This paper has presented a “bag-of-phrases” approach to object recognition that refines “bag-of-words” results by enforcing geometric consistency. The approach draws from natural language processing applications where consideration of phrases results in greater coherence of text output without the need to explicitly model the underlying linguistics. A set of pairwise “geometric grammar” rules can be used to enforce

Fig. 6. SIFT Matches with successive constraints (no filtering and after three stages of pair-wise geometric filtering) applied at SIFT thresholds of 0.8, 0.85, 0.9, and 0.95 from top to bottom. Cyan lines denote false positives, red lines denote ground truth true positives. After MLESAC was performed, they all yielded the same result at the end, regardless of threshold.



geometric consistency across a "visual phrase". These rules are applied in a computationally efficient manner via a filter cascade. This approach is robust to significant illumination variations, noise, and clutter. Experimental results show the ability to filter out over 99% of the false positives, greatly enhancing accuracy even in such adverse environments. While some limitations such as reduced detection for small objects (≤ 40 pixels) exist, this approach shows significant promise in producing robust object recognition for scene understanding and navigation.

ACKNOWLEDGMENT

The authors would like to thank Draper Laboratory, Inc. for funding this research.

REFERENCES

[1] "OpenSceneGraph," www.openscenegraph.org/, 2009.

[2] A. Berg, T. Berg, and J. Malik, "Shape matching and object recognition using low distortion correspondences," in *Computer Vision and Pattern Recognition*, vol. 1, 2005, pp. 26–33 vol. 1.

[3] M. J. Collins, "A new statistical parser based on bigram lexical dependencies," in *Proceedings of the 34th conference on Association for Computational Linguistics*. Morristown, NJ, USA: Association for Computational Linguistics, 1996, pp. 184–191. [Online]. Available: http://portal.acm.org/citation.cfm?id=981888

[4] G. Csurka, J. Willamowski, C. R. Dance, and F. Perronnin, "Incorporating geometry information with weak classifiers for improved generic visual categorization," in *ICIAP*, 2005, pp. 612–620.

[5] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray, "Visual categorization with bags of keypoints," in *In Workshop on Statistical Learning in Computer Vision, ECCV*, 2004, pp. 1–22.

[6] R. Fergus, L. Fei-Fei, P. Perona, and A. Zisserman, "Learning object categories from Google's image search," in *International Conference on Computer Vision*, 2005.

[7] W. E. L. Grimson and T. Lozano-Perez, "Model-based recognition and localization from sparse range or tactile data," *Readings in computer vision: issues, problems, principles, and paradigms*, pp. 382–414, 1987.

[8] W. Grimson, T. Lozano-Perez, and D. Huttenlocher, *Object Recognition by Computer: The Role of Geometric Constraints*. MIT Press, 1990.

[9] T. Hofmann, "Probabilistic latent semantic analysis," in *In Proc. of Uncertainty in Artificial Intelligence, UAI99*, 1999, pp. 289–296.

[10] M. Khalilov and J. A. R. Fonollosa, "N-gram-based statistical machine translation versus syntax augmented machine translation: Comparison and system combination." in *EACL*. The Association for Computer Linguistics, 2009, pp. 424–432.

[11] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[12] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," *IEEE Transactions on Pattern Analysis & Machine Intelligence*, no. 27, p. 2005.

[13] C. Schmid and R. Mohr, "Local greyvalue invariants for image retrieval," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, pp. 530–535, 1997.

[14] J. Sivic and A. Zisserman, "Video Google: Efficient visual search of videos," in *Toward Category-Level Object Recognition*, ser. LNCS, J. Ponce, M. Hebert, C. Schmid, and A. Zisserman, Eds. Springer, 2006, vol. 4170, pp. 127–144. [Online]. Available: http://www.robots.ox.ac.uk/ vgg

[15] P. H. S. Torr and A. Zisserman, "MLESAC: A new robust estimator with application to estimating image geometry," *Computer Vision and Image Understanding*, vol. 78, p. 2000, 2000.

[16] A. Torralba, R. Fergus, and W. T. Freeman, "80 million tiny images: a large database for non-parametric object and scene recognition," *IEEE PAMI*, vol. 30, no. 11, pp. 1958–1970, November 2008.

[17] M. Vidal-Naquet and S. Ullman, "Object recognition with informative features and linear classification," in *ICCV '03: Proceedings of the Ninth IEEE International Conference on Computer Vision*. Washington, DC, USA: IEEE Computer Society, 2003, p. 281.

[18] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Computer Vision and Pattern Recognition*, 2001, pp. 511–518.

[19] Z. Zhang, R. Deriche, O. Faugeras, and Q.-T. Luong, "A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry," in *Sensor Signal and Information Processing and a member of the Australian Research Council's Information Technology and Electrical Engineering Large Grants Panel.*, 1995, pp. 87–119.