Virtual View Networks for Object Reconstruction

João Carreira, Abhishek Kar, Shubham Tulsiani and Jitendra Malik
University of California, Berkeley - Berkeley, CA 94720
{carreira,akar,shubtuls,malik}@eecs.berkeley.edu

Abstract

All that structure from motion algorithms "see" are sets of 2D points. We show that these impoverished views of the world can be faked for the purpose of reconstructing objects in challenging settings, such as from a single image, or from a few ones far apart, by recognizing the object and getting help from a collection of images of other objects from the same class. We synthesize virtual views by computing geodesics on networks connecting objects with similar viewpoints, and introduce techniques to increase the specificity and robustness of factorization-based object reconstruction in this setting. We report accurate object shape reconstruction from a single image on challenging PASCAL VOC data, which suggests that the current domain of applications of rigid structure-from-motion techniques may be significantly extended.

1. Introduction

Modern structure from motion (SfM) and multiview stereo approaches [45, 12, 18] are widely used to recover viewpoint and shape information of objects and scenes in realistic settings, but require multiple images with overlapping fields of view. If only a single image of the target object is available, or if multiple ones are available but from viewpoints far apart, these methods are, respectively, inapplicable or prone to fail.

Here we aim to extend SfM-style techniques to these cases by incorporating recognition. Once an object is recognized into some potentially broad class such as "cars" or "aeroplanes", one can leverage a reusable collection of images of similar objects to aid reconstruction. This is in the spirit of recent papers on face reconstruction using automatically learned morphable models [31, 30] but we target generic categories and use SfM techniques. Our main insight is the following: SfM algorithms inhabit a rudimentary visual world made of 2D points in correspondence and these are all they "see". In this visual world, novel views can be faked more easily than in ours, where light complicates matters. Our idea, illustrated in fig. 1 is to *synthesize*
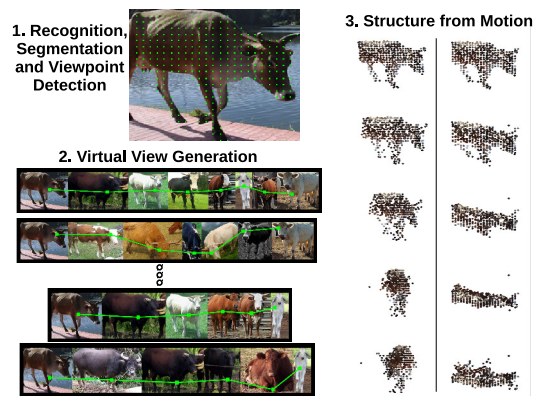

Figure 1. Our goal is to reconstruct an object from a single image using structure from motion techniques on virtual views obtained by aligning points on a regular grid on the test object (shown on top) with points on similar grids defined on objects in a reusable collection. Accurate alignment is achieved by computing geodesics on *virtual view networks*, VVN in short, which induce smooth rotations through the class object collection and simplify matching. Our approach assumes object classification, localization and viewpoint detection as inputs and produces a point cloud (here shown for different camera azimuths on the left and different elevations on the right). Better seen on a computer screen with color and zoom.

virtual (SfM) views of the target object by aligning it with images of different instances from the same class then employing robust rigid SfM techniques to reconstruct its visible surfaces. This idea is compatible with findings that human perception of structure from motion is robust to small shape deformations of the object [27] and prefers to interpret them as manifestations of a rigid object with slightly altered shape instead of a non-rigid object [51].

The main technical challenge we face is the need to align the target object with every different object in a collection, which may be pictured with arbitrary viewpoint displacements, all the way up to 180 degrees from the viewpoint of the target object. There is no dense 2D alignment technique that we know of that is prepared for such large viewpoint variation, so we propose a new one: instead of attempting to match the target object with each object in the collection

individually, we predict the pose of the target object and identify a subset of objects from the collection with similar poses – the intuition is that these will be easier to align with. Afterwards we propagate the correspondences to all other collection objects along geodesics on our new *virtual view networks*.

Standard RANSAC-based SfM approaches are unlikely to hold in our setting because use images from multiple objects that may not have exactly the same shape. Non-rigid SfM [8], developed for reconstruction from video, has not yet been demonstrated on deformations arising from intra-class variation for generic classes. We pursue instead more regularized scaled-orthographic factorization techniques [49], which optimize fewer parameters, and introduce methodology for a) increasing robustness to the multitude of noise sources we have by extrapolating synthetic inliers using domain knowledge and b) making the resulting reconstructions more specific, by emphasizing collection objects more similar to the target object.

We will review related work in the next section. Sec. 3 explains how we build and use virtual view networks to synthesize large sets of new views from one or more images of a target object to feed to SfM. Sec. 4 introduces techniques for robust SfM from noisy virtual views and sec. 5 shows alignment and reconstruction results. Source code to reproduce all results will be made available online[1].

2. Related Work

Several recent papers have exploited class-specific knowledge to improve SfM. The goal in one line of work is to create denser, higher-quality reconstructions [2, 13, 19], not to regularize SfM from few images and typically requires 3D training data. Closer to our work, Bao and Savarese proposed to reason jointly over object detections and point correspondences [3] to better constrain SfM when there are few scene points shared by different images. Our approach differs in that it focuses on reconstructing the shape of individual objects and can reconstruct from a single image of the target object.

Our work is also related to 2D alignment approaches[2], that can be divided into two camps, class-specific sparse ones, that try to localize the keypoints available in a training set [11, 5, 55, 24], and class-agnostic dense ones such as SIFTflow [36, 42] and related techniques [32] that attempt to align pairs or, as in the interesting concurrent work by Zhou et al [56], more general sets of images. Our alignment method sits in a middle ground as it uses class information but aligns a uniform grid of points inside each object that is much denser (several hundreds of points in practice) than

the typical sets of training keypoints comprising 10 to 20 points per class.

Approaches building networks of objects have gained popularity in vision in the last few years [38] and have been recently proposed for 3D reconstruction from a single image [47] but using a collection of 3D CAD models, whereas we use annotated images. Other approaches requiring some form of 3D training data have been proposed for generic [4, 28] and class-specific [23, 10] object and scene reconstruction [25, 43, 15, 35, 29].

3. Virtual View Networks

As in popular class-specific sparse alignment setups [11, 5, 55, 24], we assume that a collection of training images $\{\mathbf{I}_1, ..., \mathbf{I}_N\}$ is available for each class, together with a small set of Z consistent keypoints $L_i = \{\mathbf{m}_1^i, ..., \mathbf{m}_Z^i\}$ for each image i, where some of them may be missing due to occlusion. We bootstrap scaled orthographic cameras, represented by rotation matrices $\{R_1, ..., R_N\}$ from the keypoints for all images using the method from Vicente *et al.* [52] [3]. We also assume for simplicity that all objects in a collection are segmented and that at test time the localization problem has been solved and we have a segmentation of the test object, which could be obtained using a state-of-the-art semantic segmentation algorithm [9, 21] or cosegmentation [41] if multiple test images are available[4]. We use a fixed stride for feature extraction, resulting in a regular grid of 2D locations for matching in each image $X_i = \{\mathbf{x}_1^i, ..., \mathbf{x}_M^i\}$ with M associated descriptors $\{\mathbf{d}_1^i, ..., \mathbf{d}_M^i\}$, inside each object segmentation. The particular descriptors used will be described in the experimental section.

SfM algorithms operate on a set of point tracks, which were traditionally obtained by tracking local features in video frames and later also from unstructured image collections [45]. Here we aim to compute one track for each feature in the target object by matching it to corresponding features in "virtual views" borrowed from every object in the training collection, a hard problem because local appearance changes dramatically with viewpoint variation. We convert this hard wide-baseline problem into many easier small-baseline ones by defining a distance between feature points that considers a network over the whole collection of objects. Let network $G = \{\mathcal{V}, \mathcal{E}\}$ have nodes for all points in $\{X_i, ... X_N\}$ and edges derived by matching points in objects having similar pose. We dock the image of the test object to the network by matching it to a few objects

---

[1]Videos with all our reconstructions can be accessed online: http://goo.gl/8Xzy3m.

[2]There are also several papers studying alignment using 3D models [48, 40, 33].

[3]Cameras for non-rigid classes are computed from a representative subset of keypoints in a rigid part such as the torso, in animal classes.

[4]This is a stronger assumption but it allows us to focus entirely on reconstruction without having to dabble with the intricacies of segmentation. In the long run the two problems are likely to be best handled in conjunction.

Virtual View Network

Docking edges

Test object

Network edges

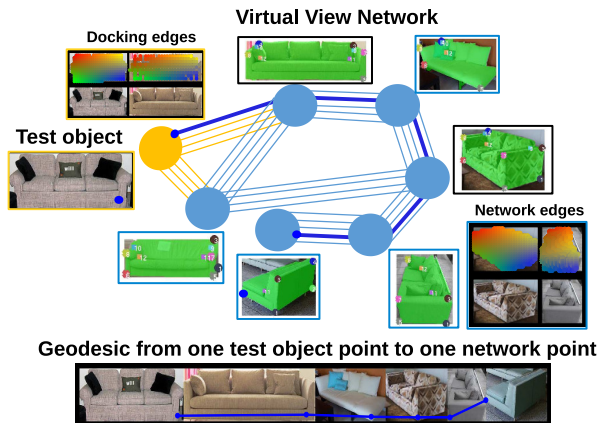**Geodesic from one test object point to one network point**

Figure 2. Instead of matching a test object to each training object directly, which may be difficult due to viewpoint variation, we match through a network connecting training objects with similar viewpoint. A test object is docked to the network by matching it to a few network objects with similar viewpoint (10 in practice), then it is aligned with all other objects based on geodesic distances in the network. Points connected by an edge are shown with similar color.

also chosen based on pose, which is assumed to be computed using a pose detector for the test object, and compute geodesics (shortest paths) between each point in the target object and all points in the collection, which can be done efficiently [17] using Dijkstra's algorithm M times, one for each feature in the target object. This network distance can then be used as a more meaningful alternative to standard euclidean distance based on appearance features, for matching the test object to all training objects (using for instance nearest neighbor matching). The overall idea is illustrated in fig. 2.

3.1. Network Construction

We match separately each object in the collection to a fixed number of nearest neighbors in pose space (30 in practice), measured using the riemannian metric on the manifold of rotation matrices $||log(R_i R_j^T)||_F$, where log refers to the matrix logarithm and $||.||_F$ denotes the Frobenius norm of the matrix. *Drifting* is a major concern in any tracking approach and is especially hard to deal with automatically in our case, over different objects. We counter drifting by regularizing feature matching using symmetric warping priors derived from the manually annotated keypoints. Let α be a weighting parameter. We define the cost of matching points u and v with locations $\mathbf{x}_u^i$ and $\mathbf{x}_v^j$ with descriptors $\mathbf{d}_u^i$ and $\mathbf{d}_v^j$ in two different images i and j as:

$$E(u,v) = ||\mathbf{d}_u^i - \mathbf{d}_v^j|| + \alpha \cdot \big[E_w(\mathbf{x}_u^i, \mathbf{x}_v^j, L_i, L_j) + E_w(\mathbf{x}_v^j, \mathbf{x}_u^i, L_j, L_i)\big]. \quad (1)$$

and model the warping cost using an interpolant $g : \mathbf{R}^2 \to \mathbf{R}^2$, here instantiated as thin plate splines which are popular for modeling various types of shapes in vision and can be fit in closed form [6, 7]. We define the warping cost as:

$$E_w(\mathbf{x}_u^i, \mathbf{x}_v^j, L_i, L_j) = ||\mathbf{x}_v^j - g_{ij}(\mathbf{x}_u^i)|| \quad (2)$$

where g_{ij} is fit to map L_i to L_j. We use two warping costs for symmetry, one in each direction, as we found this to lead to more accurate alignment in practice.

Given matching costs between all pairs of points in two neighboring objects, we add a directed edge to the network from each node u to each node v satisfying $\arg\min_v E(u,v)$.

3.2. Docking to the Network

We do not use thin plate splines for computing matching costs when docking test instances to the network because this would require keypoints that are unavailable at test time. Cost functions similar to those used in optical flow [36] would be valid alternatives, but incur some computational cost. We opted instead to simply replace the thin plate splines in eq. 1 by affine interpolants $h : \mathbf{R}^2 \to \mathbf{R}^2$ fit to map between the 4 corners of the bounding boxes of the target object and a docking object, resulting in an anisotropic scaling. This makes sense because it biases corresponding points to be in the same relative location within a bounding box, a good prior since we are docking objects with similar viewpoint. Note that a single spatial term is sufficient in this case, because the mapping is symmetric. Matching proceeds as when constructing the network, but we suppress multiple edges connecting to the same point in a docking object and keep only the one with minimum weight, to enable the speed-up to be presented next.

3.3. Fast Alignment at Test Time

It is usually desirable to push as the burden of computation to an offline stage and to have fast performance at test time. This is also feasible with our method, assuming nearest neighbor matching is used and hence we only need to identify points in training objects having minimum geodesics to points in test objects (e.g. retrieving distances to all points in each training object is unnecessary) by leveraging the recursive properties of shortest paths. We precompute the nearest neighbor matchings using network distances between all pairs of objects in the network and use these to construct a new network where all shortest paths between points in any two objects can be identified by simply selecting the outgoing edges having minimum weight, a property we will call being point-to-point.

At test time there is an additional set of edges between test object points and docking object points and the network ceases being point-to-point. Assuming there is at most a single edge from a test object point to each network node in

docking objects, however, this edge can be pushed forward and summed to all outgoing edges from nodes in docking objects making the network again point-to-point. Geodesics to all points in all objects can then be found by selecting the minimum geodesic from any of its docking points. Using this linear-time operation we manage to align a test object to a collection of roughly 1000 objects having $300,000$ points in around half a second on a modern desktop computer, instead of in more than a minute using Dijkstra's algorithm.

4. Reconstruction

Reconstruction faces three challenges in our setting: integrating sparse, far apart views of the target object, coping with noise in virtual views synthesized from training objects and producing a shape that is specific to the target object while pooling shape evidence from all objects in the training collection. We use all network images to reconstruct each test instance, and deal with noise by assuming that all generated virtual views are of a same rigid object undergoing rigid motion under scaled-orthographic projection, which has the positive effect of allowing us to estimate fewer parameters than in non-rigid reconstruction or formulations assuming perspective projection and to adopt the well-studied Tomasi-Kanade factorization framework [49]. We employ the robust Marques and Costeira algorithm [39] which can handle missing data.

Sparse reconstruction from many views of an object is an almost solved problem [22, 45, 18]. Here we focus on reconstruction from few views, in particular from a single image plus its mirrored version, exploiting bilateral symmetry possessed by most object classes (e.g. cars, aeroplanes). Directly matching original and mirrored views is generally infeasible, e.g. a car seen from the right side shares few points with one seen from the left side. We propose *network-based factorization* to handle these issues. To cope with outliers we introduce a technique called *synthetic inlier extrapolation* and, we also propose two strategies for making the reconstruction more specific to the shape of the target object, *resampling* and *xy-snapping*. We will first describe synthetic inlier extrapolation in the next subsection, after summarizing our overall reconstruction algorithm at test time as follows:

4.1. Synthetic Inlier Extrapolation

Even though factorization has few parameters compared to approaches based on bundle adjustment, they can still be negatively affected by outliers. There is prior work on handling known gaussian noise distributions [1, 26] and outliers [54, 14] within factorization, but these approaches may not be trivial to adapt so they can deal with missing data. Here we propose instead to reduce the influence of outliers by *swamping* the data with synthetic inliers generated using domain knowledge, namely we sample a constant number

---

Algorithm 1 Novel Object Reconstruction using VVN

VIRTUAL VIEW GENERATION
    for each image i of test object **do**
        Predict pose P^i
        Extract dense local features $\mathbf{d}^i$
        Dock with VVN using P^i, $\mathbf{d}^i$
        Compute distances to network points as in sec. 3.3
        Align (match) instances to generate virtual views
    end for

OBJECT RECONSTRUCTION
    Compute test object similarity with VVN objects (sec. 4.3)
    Form observation matrix with virtual views and synthetic inliers, with resampling (sec. 4.2)
    Factorization
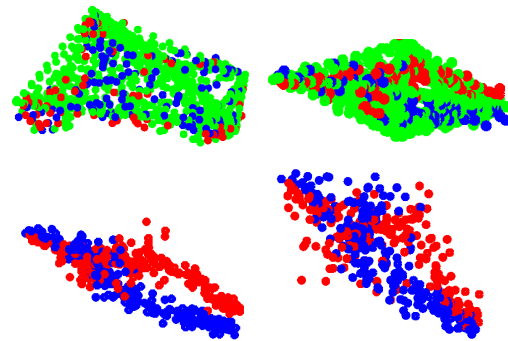   XY-Snapping (sec. 4.3)

Figure 3. The effect of reconstructing points from an image (blue) and its mirrored version (red) with and without extrapolated synthetic inliers (in green). Reconstructed points for a motorbike, including extrapolated inliers, are shown on the top row, seen from side and above. The bottom row shows the computed shapes using extrapolated inliers (left) and not using extrapolated inliers (right) from above. The shape becomes wider and noisier without extrapolated inliers.

(10 in practice) of equally spaced points along 2D lines connecting all pairs of ground truth keypoints in the training images. Such points define correct correspondences between different images (as much as they can, ignoring object shape variation) under scaled orthographic projection.

4.2. Network-Centered Factorization

Classic rigid factorization builds an observation matrix having two rows for each of N frames in an input video

sequence and one column for each of K tracked points:

$$W = \begin{bmatrix} x_1^1 & \cdots & x_K^1 \\ y_1^1 & \cdots & y_K^1 \\ \vdots & & \\ x_1^N & \cdots & x_K^N \\ y_1^N & \cdots & y_K^N \end{bmatrix}, \qquad (3)$$

then compute a $3 \times K$ shape S as well as rotation matrices, translation vectors, and scale parameter for each image from this matrix. In our case, each column will contain the coordinates of one point in the target object and the coordinates of those points in network objects that are aligned to it. Our observation matrix has a more specific structure as well, shown in fig. 4, motivated by our reliance on the virtual view network as an alignment hub which multiple target images can be docked to, hence the name network-centered factorization. We create one set of distinct points for each image of the target object, because we do not know a priori if points are shared by multiple views, and fill in tracks only between points in target images and images in the network (e.g. we do not match the target images directly), then set the rest of the matrix as missing data for the factorization algorithm to fill in. The extrapolated synthetic inliers are also added as separate points which are available for the training images but not for the test images, where they are also set as missing data. We use these points just as an additional source of regularization and ignore their reconstruction afterwards and this may be better understood by consulting fig. 3.

4.3. Building up Target Specificity

We use two strategies for increasing the specificity of the reconstruction towards the target object: resampling and xy-snapping.

Resampling. Factorization algorithms compute low-rank matrix approximations and these can be weighted so that some of the observations are given more importance. Sophisticated algorithms [46] for this task have been developed but not yet demonstrated on structure from motion. Here we propose instead to boost the importance of the target images and a few nearest neighbors from the training set (based on appearance), by simply resampling their rows in the observation matrix. This is equivalent to finding the low-rank factorization that minimizes a weighted euclidean loss with the rows corresponding to the important instances having higher weights.

xy-Snapping. The points from the target object are the only ones we can trust blindly and which should be considered correct. We enforce this as post-processing by snapping the points of the target object back to their original coordinates in a reference image after reconstruction. If we are recon-

	Points Image 1	Points Image 2	Synthetic Inliers
Image 1 (Target)	Available	Missing	Missing
Image 2 (Mirrored Target)	Missing	Available	Missing
Network Image 1	Partially Available	Partially Available	Partially Available
...	Partially Available	Partially Available	Partially Available
Network Image N	Partially Available	Partially Available	Partially Available

Figure 4. The blockwise pattern of missing data in the observation matrix (inside the bold lines) for our network-centered factorization approach, here instantiated in the case where two images of the object are docked to the network – if more images are available they can be used in the same way. The SfM algorithm fills in the missing data so we retrieve the reconstructed points for all images from the first block-row shown in this table and ignore the reconstructed points of synthetic inliers, which are only used as regularization.

structing also using a mirrored image, we can compute correspondences across the symmetry axis trivially by tracking where points move to during mirroring, and then just translating them in the image plane by the same offset as the points in the original image.

5. Experiments

Our focus is on alignment and reconstruction so we will assume that target objects have been localized and segmented as discussed in the introduction. We will evaluate 2D alignment and reconstruction separately, in each of the following subsections. We will study the impact of the accuracy of pose prediction on 2D alignment and will assume viewpoint has been correctly detected in the reconstruction section. All experiments used PASCAL VOC [16], where there are segmentations and around 10 keypoints available for all objects in each class [20]. The same setup and the same $9,087$ fully visible objects were used as in [52], but we split them into 80% training data and 20% test data and built virtual view networks on the training images and their mirrored versions, and evaluated alignment performance on test data without using keypoints. We discarded classes "dining table", "bottle" and "potted plant" in the reconstruction section because their keypoints are marked in a view-dependent way (e.g. bottles have keypoints marked on their silhouettes, so the induced cameras are always frontal and direct depth recovery requires additional cues).

5.1. 2D Alignment

We resized the image of each object to be 150 pixels tall and computed a regular grid of features taken by concatenating the fourth and fifth convolutional layers of the AlexNet convolutional network [34], resulting in 640 dimensional feature vectors at each grid location. We obtained a stride of 8 pixels by offsetting the image appropriately and passing it through AlexNet multiple times, then carefully assembling back the multiple resulting grids (similar to [44]). We also evaluated SIFT features, computed with a stride of 2 pixels and all feature extraction was performed with the background pixels set to black [5]. Each object's figure-ground segmentation was also used for ignoring grid points in the background during matching.

Our full proposed approach, VVN, aligns a test object to each training object using nearest neighbor matching on a distance function defined by geodesics on a network connecting all grid points on all training examples. While there are many class-specific techniques for localizing a set of keypoints available in training data, we are not aware of techniques of that kind that are able to align arbitrary grids of points. We opted then to compare with techniques that can align grids even though they do not use class-specific knowledge: nearest neighbor matching using the euclidean distance and SIFTflow [36], using either SIFT or the same deep features we employ. We evaluate alignment by matching each test image to all training images and checking how the ground truth test keypoints match to the training image keypoints. We average the following per-pair matching error:

$$L(C) = \frac{1}{B} \sum_u ||\mathbf{c}_u - \mathbf{m}_u||, \qquad (4)$$

where u iterates over the grid points closest to each ground truth keypoint u on the test image, C is the set of corresponding points $c_1, ..., c_B$ on a training image according to the matching, and $\mathbf{m}_u$ is the position of ground truth keypoint u on the training image. We average the errors over all images in all 20 PASCAL classes.

Different Features and Segmentation. We first compared SIFT and deep feature matching using nearest neighbor with the euclidean distance, and evaluated how important it is to have segmentation for this task. Deep features do better in general and the matching errors are slightly worse without segmentation, especially when matching with SIFT features - the deep features seem better prepared in the presence of background clutter which is promising but we will assume segmentation for the rest of the paper. For a complementary, more focused study on these matters see [37].
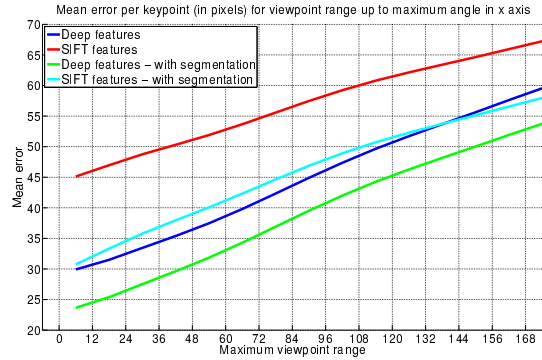

Figure 5. Mean error in eq. 4 when matching points in two objects from the same class using nearest neighbor, as a function of the viewpoint difference between the objects. Deep features allow for more accurate alignment, and this is more evident when segmentation is not available.

Results are shown in fig. 5.

Euclidean and Network Distance. We compare our proposed nearest neighbor matching using the network distance to two baselines, nearest neighbor with the euclidean distance and SIFTflow. All methods used images with the background masked and the correspondences constrained to be inside the segmentation. In this experiment we assume knowledge of ground truth cameras for selecting the elements in the network to dock the test object with, which we will also assume in the reconstruction section. The results are shown in fig. 6 and demonstrate that given an accurate pose estimate for the test object, the network distance leads to more accurate alignment up to the maximum 180 degrees viewpoint difference. SIFTflow leads to large gains over nearest neighbor using SIFT features and euclidean distance but is less robust to viewpoint variation.

Pose Prediction and Alignment. Our final and main experiment in this subsection evaluates VVN alignment with automatic pose prediction. We used predictors for the 12 rigid categories in PASCAL VOC from Tulsiani and Malik [50], which are finetuned for pose using annotations from PASCAL3D+ [53], including the additional training examples from Imagenet. Our results were obtained with an early AlexNet-based version of the models. Using more advanced convolutional network architectures [50] should boost performance further. The alignment results are shown, for the 12 rigid categories, in fig. 7 and demonstrate that the improvements over nearest neighbor with the euclidean distance still hold with automatic pose prediction. We also measured accuracy when using the best among the 2 and 4 top-scoring predicted poses and found this to bring large improvement, which suggests pose reranking as an interesting direction for future work. We show sample alignments for our method and siftflow on the same grid of deep features in fig. 8.
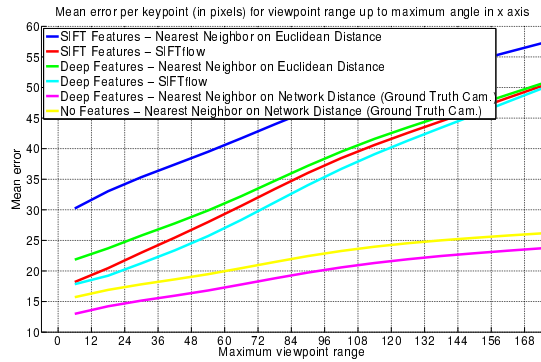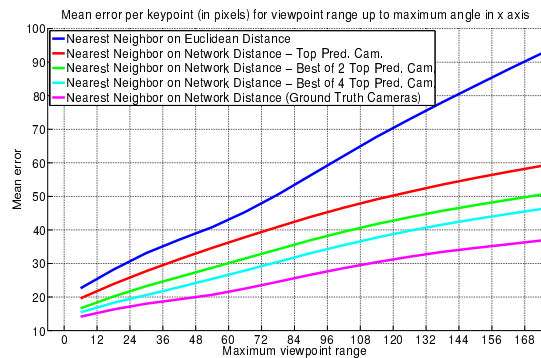
Figure 6. Mean error when matching points in two segmented objects from the same class using nearest neighbor with euclidean distance and SIFTflow, compared to nearest neighbor and our proposed network distance, as a function of the viewpoint difference between the objects. SIFTflow improves considerably over nearest neighbor matching using euclidean distance but is not robust to large viewpoint variation. Results are good even using a network built without features, using just the spatial terms.

Figure 7. Mean error when matching points in two segmented objects from the same class using nearest neighbor with euclidean distance and our proposed network distance, as a function of the viewpoint difference between the objects. The network distance leads to more accurate alignment across all viewpoint differences. The same deep features are used in all cases here and we consider only rigid categories, for which there is pose prediction training data from Imagenet. See text for details.

5.2. Reconstruction

We reconstructed PASCAL VOC objects in the test set of each class, producing fuller 3D reconstructions from a single view by taking advantage of bilateral symmetry as discussed in sec. 4. We used the same parameters for all classes, except xy-snapping which helped noticeably in most cases but degraded subtle aeroplane wings and bicycle handles so we disabled it on these two classes - the only class-specific option we introduced. We resampled the target image and its mirrored version 0.05 times the number of training examples, and their 4 nearest neighbors from the training set 0.02 times the total number of training examples for the class (see sec. 4.3). Nearest neighbors were com-
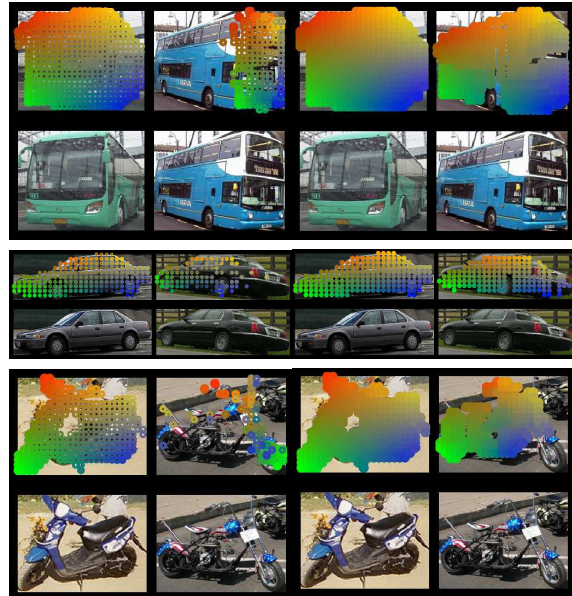

Figure 8. Example alignments using our proposed network-based approach, VVN, with automatic pose prediction (first two columns) and SIFTflow (last two columns), on the same grid of deep features and assuming correct figure-ground segmentation. Corresponding points are colored the same. VVN exploits class-specific knowledge and pose prediction to obtain resillience to viewpoint variation. See the text for additional details and the supplementary material for images showing other alignments.

puted from those training examples in the 30 to 60 degree range of viewpoint differences to the pose of the test example, selected based on euclidean distance between descriptors obtained using second-order pooling [9] on the AlexNet layer 5 features. The idea was to discard the spatial information in the layer 5 grid to better cope with viewpoint variation.

Reconstructions for all considered classes are shown in fig. 9, assuming ground truth object segmentation and viewpoints from [52]. Inlier extrapolation helped visibly in many cases, especially for the tv/monitors class which completely failed without it, becoming curved shapes not unlike Dali clocks. Highly accurate shapes are obtained for most classes, the clearest exception being horses, seemingly due to noise in the cameras used. See the caption for additional comments. There is no existing dataset for evaluating this task, also because there are few, if any, methods developed for it, so we simply put all our reconstructions on youtube for anyone to evaluate http://goo.gl/8Xzy3m. The method with closest capabilities is perhaps *carvi* [52], which produces full meshes but uses only silhouette information in a visual hull framework, e.g. image information is ignored and correspondences are not used, so it cannot deal with concavities. We will include side-by-side reconstructions in supplementary material.
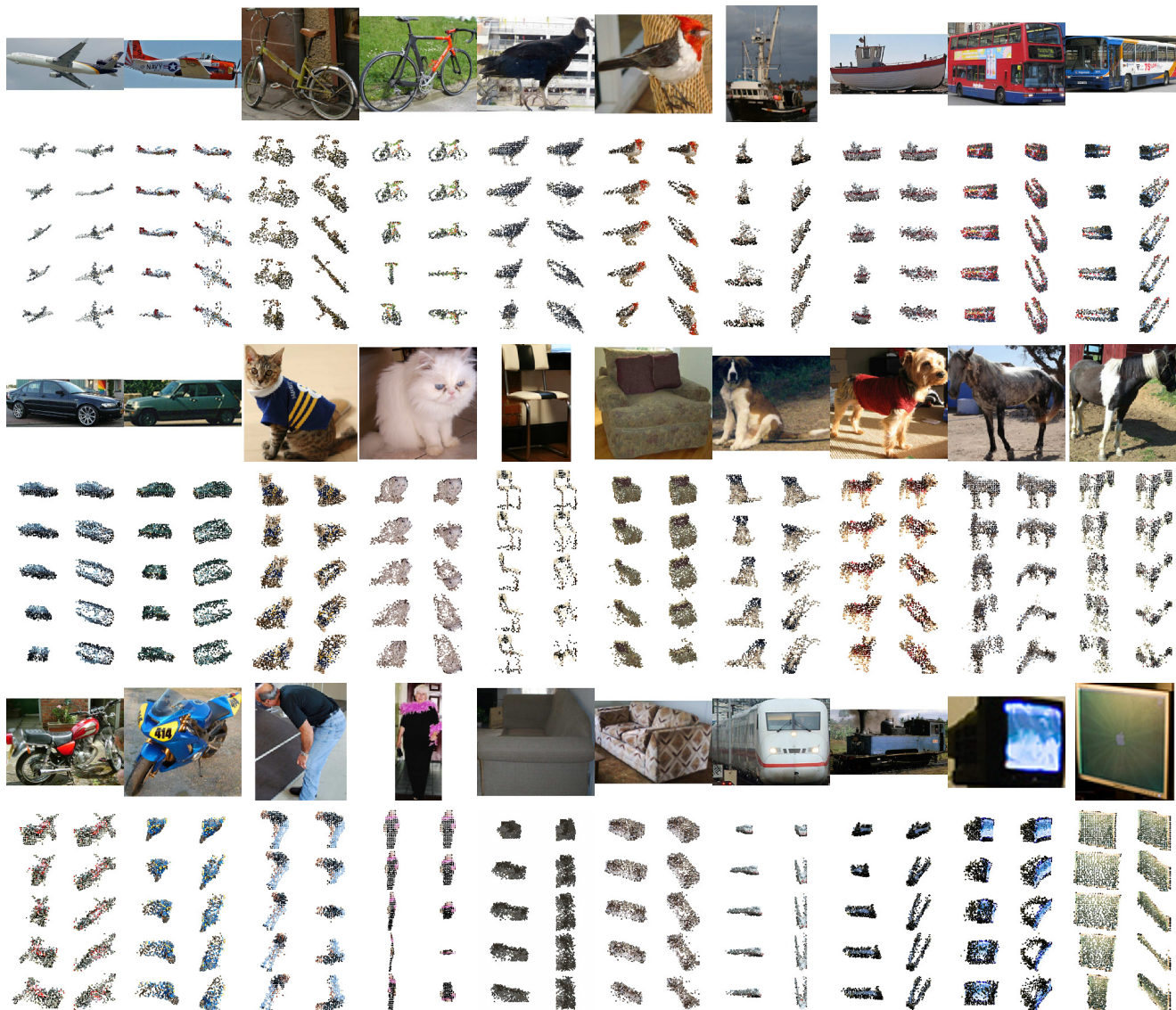
Figure 9. Example reconstructions produced by VVN on 15 PASCAL VOC categories. The first column below each image shows shapes from increasing camera azimuths, the second from different elevations as in fig. 1. We do not show two classes due to lack of space: cows, which we shown in fig. 1 and sheep, which are reconstructed with quality similar to cats and dogs. All test set reconstructions can be better visualized online at http://goo.gl/8Xzy3m and a selection is available in the supplementary material. Reconstruction is quite successful for most classes, including most animal categories except horses, that are mostly flat and consistently get a long, tilted neck. Boat is not very good either, perhaps because due to extreme intra-class variation and aeroplane and birds wings are challenging to capture accurately when seen from the side. The person class is not aided much by bilateral symmetry, except for side views, and their reconstructions feel less 3D than for most other classes.

6. Conclusions

We have introduced a framework for shape reconstruction from a single image of a target object, using structure from motion on virtual views computed from a collection of training images of other objects from the same class. At the core of our approach is a method for 2D alignment that builds a network over the image collection in order to achieve robustness across wide viewpoint variation. We have also developed techniques to increase the robustness and specificity of factorization-based rigid structure from motion using virtual views and obtained stable and accurate reconstructions of challenging objects with diverse shapes. The main challenge now is to relax the need for keypoints or to confine it to a few seed classes, which may be feasible using more advanced matching techniques [56] and features. The ability to reconstruct from one image opens new avenues for both structure from motion and recognition.

References

[1] P. M. Aguiar and J. M. Moura. Rank 1 weighted factorization for 3d structure recovery: algorithms and performance analysis. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(9):1134–1149, 2003. 4

[2] S. Y. Bao, M. Chandraker, Y. Lin, and S. Savarese. Dense object reconstruction with semantic priors. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 1264–1271. IEEE, 2013. 2

[3] S. Y. Bao and S. Savarese. Semantic structure from motion. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 2025–2032. IEEE, 2011. 2

[4] J. T. Barron and J. Malik. Shape, albedo, and illumination from a single image of an unknown object. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 334–341. IEEE, 2012. 2

[5] P. N. Belhumeur, D. W. Jacobs, D. Kriegman, and N. Kumar. Localizing parts of faces using a consensus of exemplars. In *CVPR*, pages 545–552. IEEE, 2011. 2

[6] S. Belongie, J. Malik, and J. Puzicha. Shape matching and object recognition using shape contexts. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(4):509–522, 2002. 3

[7] F. L. Bookstein. *Morphometric tools for landmark data: geometry and biology*. Cambridge University Press, 1997. 3

[8] C. Bregler, A. Hertzmann, and H. Biermann. Recovering non-rigid 3d shape from image streams. In *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, volume 2, pages 690–696. IEEE, 2000. 2

[9] J. Carreira, R. Caseiro, J. Batista, and C. Sminchisescu. Semantic segmentation with second-order pooling. In *Computer Vision–ECCV 2012*, pages 430–443. Springer, 2012. 2, 7

[10] T. J. Cashman and A. W. Fitzgibbon. What shape are dolphins? building 3d morphable models from 2d images. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(1):232–244, 2013. 2

[11] T. F. Cootes, C. J. Taylor, D. H. Cooper, and J. Graham. Active shape models-their training and application. *CVIU*, 61(1):38–59, 1995. 2

[12] D. Crandall, A. Owens, N. Snavely, and D. Huttenlocher. Discrete-continuous optimization for large-scale structure from motion. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 3001–3008. IEEE, 2011. 1

[13] A. Dame, V. A. Prisacariu, C. Y. Ren, and I. Reid. Dense reconstruction using 3d object shape priors. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 1288–1295. IEEE, 2013. 2

[14] F. De La Torre and M. J. Black. A framework for robust subspace learning. *International Journal of Computer Vision*, 54(1-3):117–142, 2003. 4

[15] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. *arXiv preprint arXiv:1406.2283*, 2014. 2

[16] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010. 5

[17] M. L. Fredman and R. E. Tarjan. Fibonacci heaps and their uses in improved network optimization algorithms. *Journal of the ACM (JACM)*, 34(3):596–615, 1987. 3

[18] Y. Furukawa and J. Ponce. Accurate, dense, and robust multiview stereopsis. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(8):1362–1376, 2010. 1, 4

[19] C. Hane, N. Savinov, and M. Pollefeys. Class specific 3d object shape priors using surface normals. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 652–659. IEEE, 2014. 2

[20] B. Hariharan, P. Arbeláez, L. Bourdev, S. Maji, and J. Malik. Semantic contours from inverse detectors. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 991–998. IEEE, 2011. 5

[21] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Simultaneous detection and segmentation. In *Computer Vision–ECCV 2014*, pages 297–312. Springer, 2014. 2

[22] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003. 4

[23] T. Hassner and R. Basri. Single view depth estimation from examples. *arXiv preprint arXiv:1304.3915*, 2013. 2

[24] M. Hejrati and D. Ramanan. Analysis by synthesis: 3d object recognition by object reconstruction. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 2449–2456. IEEE, 2014. 2

[25] D. Hoiem, A. A. Efros, and M. Hebert. Recovering surface layout from an image. *International Journal of Computer Vision*, 75(1):151–172, 2007. 2

[26] M. Irani and P. Anandan. Factorization with uncertainty. In *Computer Vision-ECCV 2000*, pages 539–553. Springer, 2000. 4

[27] G. Jansson and G. Johansson. Visual perception of bending motion. *Perception*, 2(3):321–326, 1973. 1

[28] K. Karsch, Z. Liao, J. Rock, J. T. Barron, and D. Hoiem. Boundary cues for 3d object shape recovery. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 2163–2170. IEEE, 2013. 2

[29] K. Karsch, C. Liu, and S. B. Kang. Depthtransfer: Depth extraction from video using non-parametric sampling. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 2014. 2

[30] I. Kemelmacher-Shlizerman. Internet based morphable model. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 3256–3263. IEEE, 2013. 1

[31] I. Kemelmacher-Shlizerman and S. M. Seitz. Face reconstruction in the wild. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 1746–1753. IEEE, 2011. 1

[32] J. Kim, C. Liu, F. Sha, and K. Grauman. Deformable spatial pyramid matching for fast dense correspondences. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 2307–2314. IEEE, 2013. 2

[33] J. Krause, M. Stark, J. Deng, and L. Fei-Fei. 3d object representations for fine-grained categorization. In *4th International IEEE Workshop on 3D Representation and Recognition (3dRR-13)*, Sydney, Australia, 2013. 2

[34] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. 6

[35] L. Ladicky, J. Shi, and M. Pollefeys. Pulling things out of perspective. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 89–96. IEEE, 2014. 2

[36] C. Liu, J. Yuen, and A. Torralba. Sift flow: Dense correspondence across scenes and its applications. *TPAMI*, 33(5):978–994, 2011. 2, 3, 6

[37] J. L. Long, N. Zhang, and T. Darrell. Do convnets learn correspondence? In *Advances in Neural Information Processing Systems*, pages 1601–1609, 2014. 6

[38] T. Malisiewicz and A. Efros. Beyond categories: The visual memex model for reasoning about object relationships. In *Advances in neural information processing systems*, pages 1222–1230, 2009. 2

[39] M. Marques and J. Costeira. Estimating 3d shape from degenerate sequences with missing data. *Computer Vision and Image Understanding*, 113(2):261–272, 2009. 4

[40] B. Pepik, M. Stark, P. Gehler, and B. Schiele. Teaching 3d geometry to deformable part models. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 3362–3369. IEEE, 2012. 2

[41] C. Rother, T. Minka, A. Blake, and V. Kolmogorov. Cosegmentation of image pairs by histogram matching-incorporating a global constraint into mrfs. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 1, pages 993–1000. IEEE, 2006. 2

[42] M. Rubinstein, A. Joulin, J. Kopf, and C. Liu. Unsupervised joint object discovery and segmentation in internet images. In *CVPR*, pages 1939–1946, 2013. 2

[43] A. Saxena, M. Sun, and A. Y. Ng. Make3d: Learning 3d scene structure from a single still image. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(5):824–840, 2009. 2

[44] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229*, 2013. 6

[45] N. Snavely, S. M. Seitz, and R. Szeliski. Photo tourism: exploring photo collections in 3d. *ACM transactions on graphics (TOG)*, 25(3):835–846, 2006. 1, 2, 4

[46] N. Srebro, T. Jaakkola, et al. Weighted low-rank approximations. In *ICML*, volume 3, pages 720–727, 2003. 5

[47] H. Su, Q. Huang, N. Mitra, Y. Li, and L. Guibas. Estimating image depth using shape collection. *Transaction of Graphics*, (Special Issue of SIGGRAPH 2014), 2014. 2

[48] H. Su, M. Sun, L. Fei-Fei, and S. Savarese. Learning a dense multi-view representation for detection, viewpoint classification and synthesis of object categories. In *CVPR*, pages 213–220, 2009. 2

[49] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: a factorization method. *International Journal of Computer Vision*, 9(2):137–154, 1992. 2, 4

[50] S. Tulsiani and J. Malik. Viewpoint and keypoints. In *Computer Vision and Pattern Recognition, 2015. CVPR 2015. IEEE Conference on*, 2015. 6

[51] S. Ullman. Maximizing rigidity: the incremental recovery of 3-d structure from rigid and rubbery motion. 1983. 1

[52] S. Vicente, J. Carreira, L. Agapito, and J. Batista. Reconstructing pascal voc. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 41–48. IEEE, 2014. 2, 5, 7

[53] Y. Xiang, R. Mottaghi, and S. Savarese. Beyond pascal: A benchmark for 3d object detection in the wild. *IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2014. 6

[54] L. Xu and A. L. Yuille. Robust principal component analysis by self-organizing rules based on statistical physics approach. *Neural Networks, IEEE Transactions on*, 6(1):131–143, 1995. 4

[55] Y. Yang and D. Ramanan. Articulated human detection with flexible mixtures-of-parts. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(12):2878–2890, 2013. 2

[56] T. Zhou, Y. J. Lee, S. X. Yu, and A. A. Efros. Flowweb: Joint image set alignment by weaving consistent, pixel-wise correspondences. In *Computer Vision and Pattern Recognition, 2015. CVPR 2015. IEEE Conference on*, 2015. 2, 8