Detecting and Reconstructing 3D Mirror Symmetric Objects

Sudipta N. Sinha, Krishnan Ramnath, and Richard Szeliski

Microsoft Research, Redmond, WA
{sudipsin,kramnath,szeliski}@microsoft.com

Abstract. We present a system that detects 3D mirror-symmetric objects in images and then reconstructs their visible symmetric parts. Our detection stage is based on matching mirror symmetric feature points and descriptors and then estimating the symmetry direction using RANSAC. We enhance this step by augmenting feature descriptors with their affine-deformed versions and matching these extended sets of descriptors. The reconstruction stage uses a novel edge matching algorithm that matches symmetric pairs of curves that are likely to be counterparts. This allows the algorithm to reconstruct lightly textured objects, which are problematic for traditional feature-based and intensity-based stereo matchers.

Keywords: Symmetry detection, 3D reconstruction, curve matching.

1 Introduction

Mirror (bilaterally) symmetric 3D objects are all around us in the man-made world, and yet most computer vision algorithms do not take advantage of this potential structural regularity. Figure 1 show some of these objects, which often occur in cluttered environments and may also suffer from a dearth of easily matchable keypoint features.

There exists a wide body of literature for detecting skewed planar symmetric shapes [1,2,3,4,5,6]. Techniques have also been developed for reconstructing 3D symmetric scenes [7,8]. However, relatively few algorithms have been developed for reconstructing symmetric 3D objects in cluttered environments. Previously published algorithms either rely on manual intervention [9,10,11], apply photometric stereo to symmetric textureless objects [12], or assume structural regularities, such as piecewise planar surfaces bounded by straight line segments [13].

In this paper, we remove the restrictions found in previous systems and demonstrate the ability to detect and reconstruct a wider range of 3D mirror-symmetric objects in general scenes. We begin by matching pairs of mirrored keypoint descriptors to find the epipolar geometry (vanishing point) corresponding to the direction of symmetry, i.e., the normal of the plane of symmetry (Figure 2a). We enhance the performance of this stage by constructing local affinely deformed versions of each descriptor, which gives us a richer set of potential features to match and better tolerates local foreshortening effects.

A. Fitzgibbon et al. (Eds.): ECCV 2012, Part II, LNCS 7573, pp. 586–600, 2012.

Fig. 1. Some examples of mirror-symmetric 3D objects in cluttered settings

Next, we rectify the input image to obtain a traditional horizontal epipolar geometry (Figure 2b). We then extract edges and curves from the rectified image (Figure 2c) and match these to obtain the disparities (inverse depths) of corresponding points (Figure 2d). Because it uses a different set of features, our curve-based approach to matching is complementary to the feature-based [1,2,3,4,5] and dense pixel-based [7,8] matching approaches used in previous systems and therefore extends the range of applicability of symmetry detection and reconstruction techniques.

The remainder of this paper is structured as follows. We begin with a review of previous work in Section 2. In Section 3, we describe the imaging geometry when viewing a bilaterally symmetric 3D object from both a canonical and general viewpoint. Section 4 describes our RANSAC-based matching algorithm for finding the symmetry direction and our enhanced feature descriptors. Section 5 describes how we rectify each image according to the dominant symmetry direction and determine the likely disparity range. Section 6 describes our curve matching algorithms for finding dense edge correspondences on the symmetric 3D objects. Section 7 summarizes our experiments on a number of challenging scenes and real-world 3D objects. We conclude with a discussion of future work.

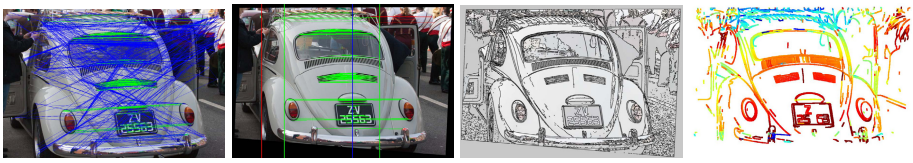
2 Previous Work

The computer vision literature on detecting bilateral, point, and translational symmetries is rich and voluminous. Liu *et al.* [1] provide a comprehensive summary and guide to this literature, while [6] summarizes the results presented at the recent Symmetry Detection from Real World Images CVPR 2011 workshop[1].

Among all of the symmetries studied in this field, the most closely related to this paper are *2D bilateral (mirror) skew symmetries*, which are observed when a bilaterally symmetric planar object (e.g., butterfly wings) are observed from an arbitrary perspective camera. Most approaches to detecting such symmetries start by matching affine covariant regions or interest points and then voting for the axis of skew symmetry [2,3,4,5,14].

In this paper, we are interested in detecting and modeling full 3D objects with interesting depth and structure rather than just planar configurations [15].

[1] `http://vision.cse.psu.edu/research/symmComp/workshop/index.shtml`



(a) (b) (c) (d)

Fig. 2. Processing pipeline: (a) epipolar geometry from matched points; (b) rectified image with depth range and inlier matches (see text for a longer description); (c) extracted curves; (d) recovered depths for matched curves.

The observation that an object and its image seen in a mirror can be treated as a stereo pair goes back at least two decades. The system described by Mitsumoto *et al.* [9] uses manual correspondences to determine the 3D points from which a wireframe model can be constructed. François *et al.* [10] observe that a mirror-symmetric scene can be similarly treated as a stereo pair; they use the commercial PhotoModeler system to establish manual correspondences in order to create texture-mapped 3D models of objects. Shimshoni *et al.* [12] apply photometric stereo to images of symmetric textureless Lambertian objects.

Jiang *et al.* [11] use orthogonal vanishing points to estimate the camera pose and intrinsics; they then use automated feature detection and matching to estimate a sparse 3D reconstruction, which is then turned into a continuous texture-mapped model with the help of user sketching. Xue *et al.* [13] describe a related system that uses automated straight line segment detection and mapping to produce piecewise-planar models of furniture. The detection of 3D symmetry in range scan data is also an active topic of study in computer graphics [16].

Fully automated methods for dense reconstruction of scenes with extended symmetries have recently been developed. Wu *et al.* [17,7] describe a system for detecting repetitions in architectural scenes and use dense stereo matching to build 3D texture-mapped façade models. Koser *et al.* [8] find the symmetry plane in a bilaterally symmetric architectural interior, façade, or detail using automated feature matching; they then use dense stereo to obtain a depth map.

In this paper, we use similar automated matching techniques to estimate the plane of symmetry, but then use curve matching to reconstruct a sparse 3D object model for objects that lack dense texture. We also demonstrate our results on objects seen in cluttered environments, where determining the epipolar geometry and the extent of the symmetric objects are challenging problems.

3 Problem Formulation

A bilaterally symmetric 3D object has a *plane of symmetry*, where corresponding points are opposite to each other with respect to this plane. Let us call the normal associated with this plane as the *direction of symmetry*.

The simplest case to analyze is when the direction of symmetry is aligned with the x-axis, as in Figure 3a. Under this condition, all corresponding points,

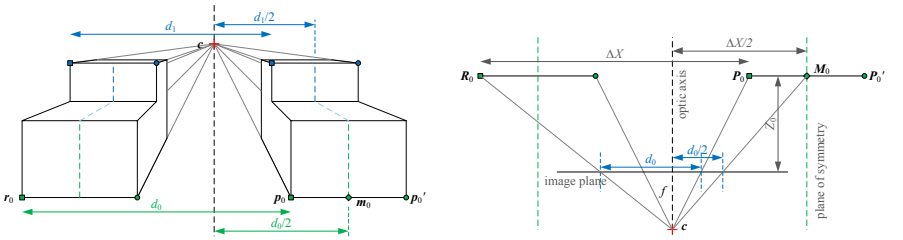


Fig. 3. Epipolar geometry: (a) In a rectified configuration, the disparity d_0 between a point $\mathbf{p}_0$ matched in an image with its reflected version $\mathbf{r}_0$ is equal to twice the horizontal distance $d_0/2$ between the match midpoint $\mathbf{m}_0$ and the optic center $\mathbf{c}$. Note that in general, objects may have more than one local axis of symmetry per scanline. (b) A top-down view shows the relationship between the 3D horizontal displacement ΔX between corresponding points, between the optic center and the midline $\Delta X/2$, and the pixel horizontal disparity $d_0 = f\Delta X/Z_0$.

e.g., $\mathbf{p}_0$ and $\mathbf{p}'_0$, lie on corresponding scanlines. Furthermore, points that lie at the same depth from the camera have the same *horizontal disparity* $d_0 = \|\mathbf{r}_0 - \mathbf{p}_0\|$, where $\mathbf{r}_0$ is the point in the horizontally reflected image that would end up being matched with $\mathbf{p}_0$ by a traditional stereo matcher.

It is easy to show that the horizontal disparity $d_0 = f\Delta X/Z_0$ is equal to twice the horizontal distance from the optic center to the midpoint $\mathbf{m}_0$ (see Figure 3b). The disparity is linearly related to inverse depth by the scale factor $f\Delta X$, where f is the focal length and ΔX is twice the horizontal distance between the plane of symmetry and the optic axis.[2]

In the case of a camera in a general configuration, the direction of symmetry, i.e., the lines joining all corresponding points, point at a finite epipole $\mathbf{e}$ (Figure 4). By assuming that the camera has orthonormal axes, square pixels and principal point at the center of the image, and by assuming that the focal length f is known[3], we can compute a *rectifying rotation* around the optic center that transforms an arbitrary image into a horizontally rectified one, i.e., where $\mathbf{e}' = (1, 0, 0)$ and all corresponding points lie on corresponding scanlines. Let us denote the rotation between the world coordinate system and the rectified coordinate system using a 3×3 rotation matrix

$$\mathbf{R} = \left[\mathbf{r}_0 \middle| \mathbf{r}_1 \middle| \mathbf{r}_2\right]. \tag{1}$$

$\mathbf{R}$ maps the ideal point $\mathbf{i} = (1, 0, 0)$ to an epipole direction $\mathbf{r}_0 = \mathbf{R}\mathbf{i}$, which then gets projected to the epipole $\mathbf{e}_0 = (e_x, e_y, f)$. We can therefore determine the

[2] This is the same formula as in traditional rectified stereo matching, where $d = fB/Z$ and B is the horizontal baseline between cameras [18].

[3] Modern cameras report their focal lengths in their EXIF tags and mobile vision applications can also assume a calibrated sensor. If such information is missing, additional cues in the image, e.g., orthogonal vanishing points, can be used instead.

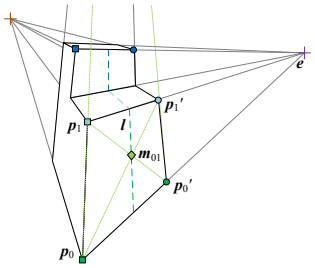


Fig. 4. In a general image configuration (arbitrary camera rotation), lines joining mirror-symmetric points $(\mathbf{p}_i, \mathbf{p}'_i)$ intersect at the vanishing point $\mathbf{e}$ corresponding to the direction of symmetry. Pairs of correspondences delineating a planar surface with bilateral planar symmetry can be used to compute a midline $\mathbf{l}$, which can be made vertical during the rectification stage.

first column of $\mathbf{R}$ by simply normalizing the calibrated epipole coordinates, i.e., $\mathbf{r}_0 = \mathcal{N}(\mathbf{e}_0)$, where $\mathcal{N}(\mathbf{v}) = \mathbf{v}/\|\mathbf{v}\|$ turns a vector into a unit vector.

How can we determine the other entries in $\mathbf{R}$?

One possibility is to minimize the amount of roll around the x axis during the rectification process. This can be achieved by setting

$$\mathbf{r}_1 = \mathcal{N}(\mathbf{k} \times \mathbf{r}_0) \tag{2}$$

$$\mathbf{r}_2 = \mathbf{r}_0 \times \mathbf{r}_1, \tag{3}$$

where $\mathbf{k} = (0, 0, 1)$.

Another possibility is to pick a plane in the image that passes through two matching pairs, say $(\mathbf{p}_0, \mathbf{p}'_0)$ and $(\mathbf{p}_1, \mathbf{p}'_1)$ in Figure 4, and to make this plane orthogonal to the camera in the rectified image. To do this, we first compute the midline $\mathbf{l}$ as the line joining midpoint $\mathbf{m}_{01} = \mathbf{p}_0\mathbf{p}'_1 \times \mathbf{p}_1\mathbf{p}'_0$ and "vanishing point"[4] $\mathbf{v}_{01} = \mathbf{p}_0\mathbf{p}_1 \times \mathbf{p}'_0\mathbf{p}'_1$. We then set

$$\mathbf{r}_1 = \mathcal{N}(\mathbf{l} \times \mathbf{r}_0) \tag{4}$$

$$\mathbf{r}_2 = \mathbf{r}_0 \times \mathbf{r}_1. \tag{5}$$

This choice makes the green midline $\mathbf{l}$ in Figure 4 map to the vertical green midline in Figure 3a.

We have experimented with both of these choices, and found that while the second strategy (which can be implemented using RANSAC and then counting inliers to this plane) does indeed expose the bilateral symmetry on the chosen plane, it also creates larger pixel distortions during the rectification stage. We therefore use the first strategy, which minimizes the amount of distortion while moving the horizontal epipole to infinity.

[4] $\mathbf{v}_{01}$ is not a true vanishing point, since in general $\mathbf{p}_0\mathbf{p}_1$ and $\mathbf{p}'_0\mathbf{p}'_1$ need not be parallel, just bilaterally symmetric.

4 Feature-Based Symmetry Detection

Given this geometric analysis of bilateral 3D symmetry, how can we go about finding corresponding matching points and determining the rectifying transformation? Unlike traditional structure from motion, which requires 5 (calibrated) or 7 or 8 (uncalibrated) correspondences to determine the epipolar geometry [19], we only need to find two sets of corresponding pairs, which makes a RANSAC search for the epipole $\mathbf{e}$ much more efficient.[5]

(a) (b) (c) (d)

Fig. 5. Feature matching results: (a) Hessian-Laplace; (b) FAST-ER; (c) FAST-ER with affine distortions; (d) detected features for FAST-ER+Affine. The blue lines show all feature matches and the green lines show the inliers after finding the best epipole.

The traditional approach to finding such correspondences is to detect scale and affine-covariant features [2,3,4,5,6,7,8] and to then perform nearest neighbor-based matching [20] on the extracted features. In order to compensate for local perspective distortions, affine adaptation of features is used in Hessian-Affine and Harris-Affine detectors [21]. However, performing affine adaptation on the interest points hurts the repeatability, as it increases the likelihood that different features map to the same descriptor.

In order to maintain the repeatability while also allowing for affine distortions, we first run a feature detector with high repeatability. We then extract a bag of features for each interest point location, using an approach similar to the one described in [22]. We sample a set of affine transformations that allow locally distorted patches to match better. However, rather than matching every affine transformed patch in the query image to every patch in the other image, we perform nearest neighbor matching while maintaining the grouping relationship of the features, as described next.

For each query image, given a set of detected keypoints, we compute DAISY descriptors [23] for each of the affinely deformed patches. We then query for nearest neighbor matches using a k-d tree to speed up the search. This gives us a sorted list of neighbors based on descriptor distance. This list can then be scored using the ratio test [20] to find the inlier matches.

This traditional method of scoring, however, does not account for the fact that we have multiple descriptors (affine-transformed patches) per interest point

[5] Note that unlike papers that detect planar skew symmetric configurations [1,2,3,4,5,6], we are looking for a rectifying 3D transformation, since in general we do not expect to find a single bilateral symmetry axis.

location. Performing a k-NN query without considering this grouping can result in repetitive matches, i.e., multiple matching descriptors from the same bag. This prevents us from evaluating potential matches from other bags that did not make it to the k top neighbors list. We adapt our approach by enumerating all neighbors per query descriptor and consider only the best matched pair *per bag* for entry into the ratio test. Hence, each of the k nearest neighbors that are returned are guaranteed to be from unique bags, which results in superior matching performance. A few examples matches using three different interest point variants are shown in Figure 5.

Once we have determined a set of raw correspondences, we use the Random Sample Consensus (RANSAC) algorithm [24] to find the best epipole. For every pair of random correspondences chosen, say $(\mathbf{p}_0, \mathbf{p}'_0)$ and $(\mathbf{p}_1, \mathbf{p}'_1)$, we compute the epipole $\mathbf{e}$ using a cross product of the line equations and then measure the perpendicular distances of each of the endpoints to the line joining its corresponding point to the epipole. We then convert this distance in pixels to a log likelihood assuming a contaminated Gaussian model, where the Gaussian has a standard deviation of $\sigma = 2$ pixels and the outlier probability is $\epsilon = 0.01$. The sum of log likelihoods is then used to score all random samples, and the best sample is used to determine the epipole.

5 Rectification and Disparity Range Estimation

Once we have determined the epipole, we rectify the image (using the first, minimal distortion, heuristic) and re-center the resulting image at the original image center, remembering the true optic center value to later compute disparities.

In order to improve the effectiveness of the edge matching algorithm which is described in the next section, we use the inlier matches from the epipole estimate to compute a set of plausible disparity (inverse depth) values. Recall that in this paper, our aim is to reconstruct compact bilaterally symmetric objects rather than extended scenes. For such objects, we expect some of the feature matches to lie on or near the frontmost visible surface of the object. We also expect the object to be bounded in depth, e.g., to be no more than four times as deep as the distance between the frontmost part and the camera.

In order to compute the disparity range, we compute the disparities (horizontal distances between match midlines and the optic center) for all of the matches that are inliers to the epipolar geometry. We then discard matches whose disparity is on the wrong side of the epipole (and hence have negative depth). From the remaining matches, we find the 80% percentile largest disparity (likely front of the object), and set the disparity range to 25%–130% of this value. This corresponds to the assumptions that the object can be no more that four times as deep as the distance to camera and that it cannot have small protrusions more than one third this distance.

Figure 2b shows an example of the inlier and depth range computation and Figure 8b (and the supplementary materials [30]) show some additional examples. The vertical blue line shows the disparity corresponding to the 80% disparity for all matched epipolar correspondences. The vertical red line shows the

(a) (b) (c) (d) (e) (f)

Fig. 6. Curve matching results: (a) cost image generated from all pairs of matching edges with the red line showing the winner on each scanline; (b) the corresponding edge matches, color-coded by depth. (c) the same cost image with the results of vertical dynamic programming with (d) the corresponding edge matches; (e) the cost image associated with running an intra-scanline dynamic program starting inside each inter-edge interval, along with the dynamic programming solution (red) and the midlines (disparities) for all the matches; (f) the corresponding edge matches. Additional examples are included in the supplementary materials [30].

(horizontal) location of the optic center after rectification. The two vertical green lines show the allowable disparity range. Matches whose midpoints fall outside the range defined by the vertical green lines are considered not part of the object. The horizontal match lines for such correspondences are drawn in red, while the "object inlier" correspondences are drawn in green.

6 Curve Matching

Once we have rectified our image and computed a plausible disparity range, we could use various techniques to recover a dense 3D model of the object. The most common approach [7,8] is to use dense pixel-based stereo matching. A lot of our objects, however, have large textureless areas or photometric inconsistencies such as reflections. For this reason, we explore the use of edge and curve-based matching, as a set of techniques complementary to pixel-based stereo.

Over the years, a number of edge-based stereo algorithms have been developed [25]. For example, Ohta and Kanade [26] develop a dynamic programming algorithm that encourages smoothness in disparities both within scanlines and along curves. Collins [27] introduces the notion of a plane sweep and matches binary edges in scenes such as surveillance video. None of these techniques, however, admit a simple mechanism or extension for enforcing that matches are symmetric, which is a fundamental constraint we can exploit in our domain to make our matches more reliable.

Note, however, that occluding contours for rounded symmetric objects need not be symmetric counterparts even though locally they may have similar visual appearance under a mirror transformation. In fact for many symmetric 3D objects, the symmetric counterpart could either be self occluded or may not produce a salient image curve. This inability to accurately reconstruct the occluding contours is a limitation of our method as well as prior methods [7,8].

Recall from Section 3 that the disparity (inverse depth) between a point in our original image and its corresponding point in the mirror-reflected image is twice the horizontal displacement between the midpoint of the equivalent within-image correspondence and the optic center (Figure 3). A simple way to determine

correspondences is to have potential matches on each scanline "vote" for their midpoint and to then pick the midpoint with the highest score as indicating the desired disparity.

Finding potential matches: We first split all detected image curves to generate y-monotone segments to ensure that each segment intersects a scanline at most once. We then match all pairs of curves after pruning pairs whose scanline intervals do not overlap or overlap very little. Since the image is rectified, pixels where matching curves meet each scanline must correspond. We perform two simple tests to check the plausibility of curve matches under bilateral symmetry. We check whether under symmetry, the polarity of the intensities around the matching edge (or ridge) pixels agree and also whether the local gradient orientation of the 2D curves at these pixels are similar. A curve pair is assigned a matching score equal to the number of pixels where both conditions are satisfied. Using these scores, we then compute $K = 10$ nearest neighbors for every curve in the image and retain the reciprocal matches, i.e., any match between curves c_0 and c_1 where c_0 is c_1's neighbor and vice versa. The set of filtered curve correspondences specifies a set of potential pixel correspondences, each of which can vote for their midpoints on the respective scanlines.

Figure 6a shows the resulting cost image for each scanline, where darker colors indicate more votes and the red pixel in each row indicates the winning disparity. This process is analogous to the techniques that vote for axes of symmetry in determining bilateral planar skew symmetry [1,2,3,4,5].

If we use this disparity to select matches on each scanline, we can reconstruct a sparse "ribbon-like" 3D model of the object, where the depths at each scanline are restricted to a small range (Figure 6b). We can improve the quality of the results by running a dynamic programming algorithm (DP) vertically across the scanlines, which has the effect of smoothing out the computed midline and hence removing smaller isolated errors (Figure 6c–d). While this approach works for some objects, it often fails to match all of the corresponding curves.

An alternative to the simple voting scheme is to run a two-stage dynamic programming (DP) algorithm, which we describe next. First an intra-scanline DP step is performed to estimate smooth disparity variation within each scanline while enforcing both ordering as well as uniqueness constraint amongst its pixels. The second stage performs DP across scanlines to encourage the continuity of curve-to-curve correspondences across scanlines.

Intra-scanline Dynamic Programming: For a horizontal scanline, we wish to compute an optimal set of correspondences that satisfies the ordering and uniqueness constraints under mirror symmetry. We formulate this as a best path problem on a directed acyclic weighted graph, which can be efficiently computed using dynamic programming. Unlike prior DP-based stereo matching methods that match scanlines, we need to both match the left and right parts of the scanline and simultaneously compute the best point to partition the scanline. We consider all pixel correspondences for the scanline that agrees with the estimated disparity range and find all horizontal intervals generated by these pixels on the scanline. By picking an arbitrary partition point P within one of the intervals,

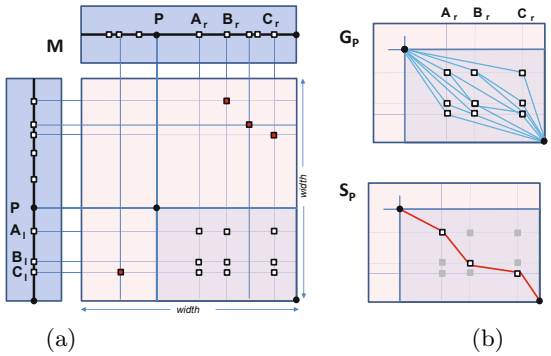


Fig. 7. Intra-scanline DP for a particular scanline: (a) For a partition point P, all potential correspondences are visualized on a 2D grid M. In this figure, (A_l,A_r), (B_l,B_r) and (C_l,C_r) are true correspondences. All the white nodes in M are admissible given the point P whereas the red nodes are not admissible because both pixels lie on the same side of P on the scanline. (b) A directed acyclic weighted graph G_P is constructed (see text for details) and an optimal path is computed from the vertex corresponding to P to the bottom right point of the grid M in order to choose the most likely set of correspondences under the uniqueness and ordering constraint with respect to P.

we construct a directed acyclic graph corresponding to this interval, which we denote as G_P. The corresponding best path in G_P is denoted as S_P.

The graph $G_P(V, E)$ is constructed as follows. Each of its nodes in V correspond to those potential pixel correspondences (x_l,x_r) on the scanline where $x_l < P.x$ and $x_r > P.x$. Here $P.x$ refers to the x-coordinate of P and the subscripts l and r indicate which of the two corresponding pixels is on the left (and right) on the scanline. The graph G_P has an embedding M on a 2D square grid whose dimensions equal the length of the scanline, as shown in Figure 7. The correspondences that do not fall within the bottom right quadrant of M as shown in Figure 7(a) do not have to be considered. Directed edges are created from a node u to every other node $v \in V$, as long as the following conditions hold true. First, $u.x_r < v.x_r$ and $v.x_l < u.x_l$, i.e., u must be encountered before v when traversing outwards on the scanline starting from P. Second, no vertex w should exist between u and v that satisfies the condition $u.x_r < w.x_r < v.x_r$ and $v.x_l < w.x_l < u.x_l$. This condition amounts to saying that u and v will be connected if their 2D bounding box in M has no third vertex w lying strictly inside it. Intuitively, as we move outwards from P, we consider the next pair of pixels encountered or skip one or both of them. Each edge $(u \rightarrow v)$ has a weight

$$w_e(u, v) = \max(0, (\Delta(x)_{\max} - |(v.x_r - u.x_r) - (u.x_l - v.x_r)|)). \tag{6}$$

This choice of edge weights favors paths that are most parallel to the main diagonal of M along which all edge weights attain a maximum value. Each node u in V also has a positive weight $w_s(u)$ assigned to the matching score of the corresponding pixel pairs, which is equal to the matching score of their associated curves. Any path $\mathcal{P}$ in this graph from P to the bottom right corner of M gives

a set of feasible correspondences and the largest set of likely candidates are extracted by computing the longest path amongst these.

$$S_P = \arg\max_{\mathcal{P}} \big[\sum_{u \in \mathcal{P}} w_n(u) + \sum_{(u,v) \in \mathcal{P}} w_e(u,v) \big]. \tag{7}$$

The intra-scanline DP is performed for every interval on each scanline. The maximum score and the corresponding best paths for each interval is stored and used for propagating information across adjacent scanlines in the next stage.

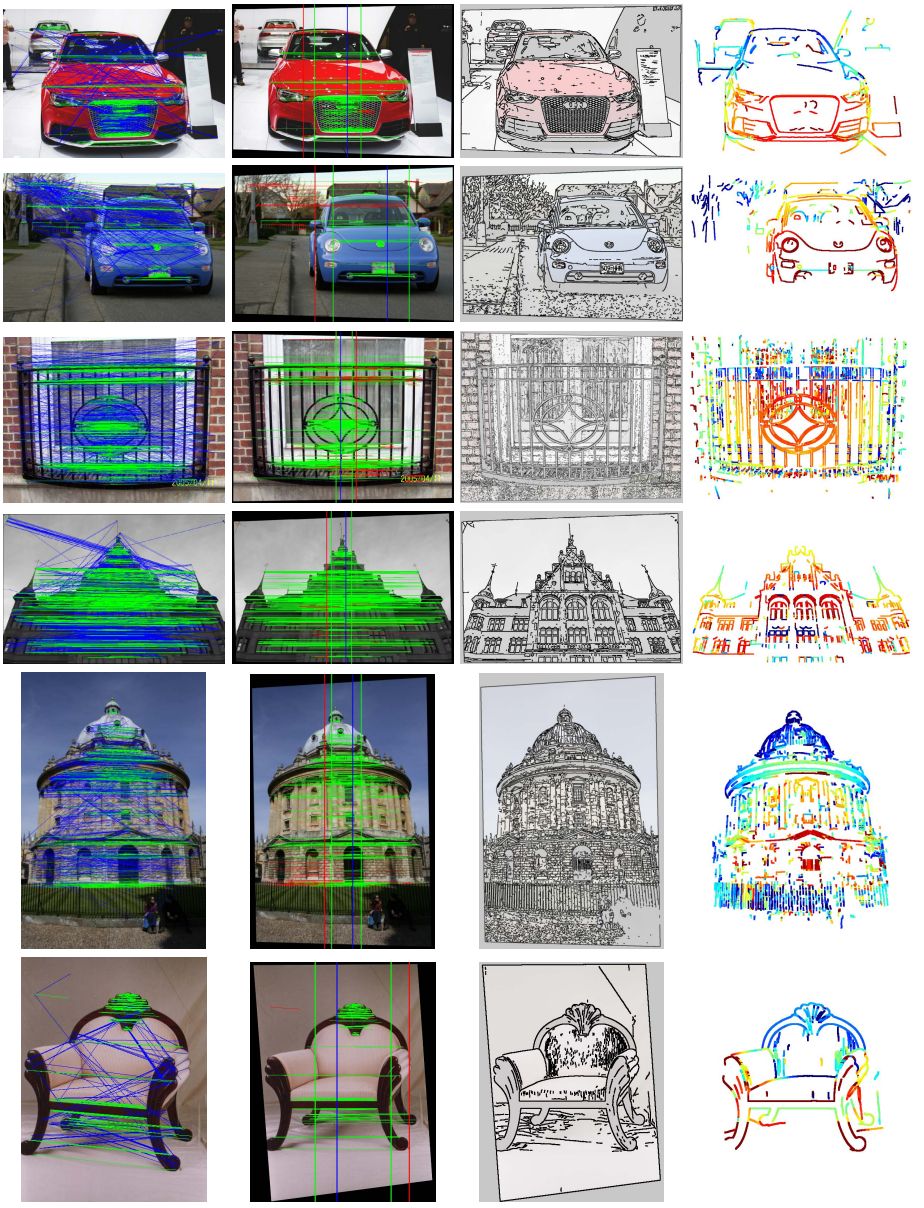
Dynamic Programming Across Scanlines: Each interval for which an intra-scanline DP problem was solved previously, now serves as a node in a new weighted directed acyclic graph. All intervals on a scanline are connected to intervals on the scanline below it via directed (top-down) edges. Each node is given a unary weight equal to the corresponding interval's score computed previously during the intra-scanline DP stage. For each edge between nodes j and k, we set the edge weight as $w_{jk} = \max(0, 1 - \frac{n_{jk}}{\kappa})$, where n_{jk} is the number of curve-pairs present in both intervals corresponding to j and k. We set the parameter $\kappa = 5$. This favor selecting interval pairs on adjacent scanlines having many curve pairs in common. For interval pairs that do not share common curve pairs, the weight is set inversely proportional to the x-displacement of the midpoints of the two intervals. This encourages similar disparity changes across scanlines. Finally, the longest path between nodes corresponding to the top and bottom scanline is computed and this produces the final set of curve matches.

Figure 6e shows the cost values computed for each scanline interval in this two-staged algorithm. The red lines overlaid on each scanline are the mid-lines corresponding to matched edges. The associated depth map shown in Figure 6f indicates that the new algorithm matches more edges in the curved regions of the car than the simpler one-pass algorithm.

7 Experimental Results

To determine the best feature detector for our application, we evaluated a variety of popular feature detectors (MSER, Laplace, Harris, FAST-ER, Hessian, Hessian-Laplace, Hessian-Affine, and Edge Foci) [21,28,29]. For detectors that do not estimate a characteristic scale, we computed features at different levels of an image pyramid. As noted in Section 4, we found that affine adaptation of the interest points hurts the repeatability of Hessian-Affine points. MSER, Edge Foci and the Laplace variants produced middle of the pack results while FAST-ER produced the most repeatable interest points.

In order to account for local affine distortions, we generated affine transformed versions [22] of the FAST-ER [28] interest point patches. This produced the best performance in terms of epipolar inliers. The affine transformations can be characterized by ellipses with varying tilt angles and scales. In our experiments, we sampled four different scalings of the ellipse along the horizontal axis in increments of 0.25 to generate the affine distortions. Since most of the images



(a) (b) (c) (d)

Fig. 8. Some additional results: (a) epipolar geometry from matched points; (b) rectified image with depth range and inlier matches; (c) extracted curves; (d) recovered depths for matched curves. See the supplementary materials [30] for even more results and some 3D animations.

were upright, varying the tilt angles did not provide much benefit. After interest point extraction, we perform nearest-neighbor matching, as described in Section 4. Figure 5 shows the point matches for the Hessian-Laplace, FAST-ER, and FAST-ER with affine distortions, along with the results for FAST-ER+Affine. Additional results can be found in the supplementary material [30].

Figure 8 shows some of the images we used for testing our algorithms along with the results of using our best variants, namely FAST-ER+Affine feature detector and descriptor, the inlier disparity range computation, and the two-stage DP (intra-scanline and inter-scanline) algorithm used to compute matching edges and their disparities. As you can see, our algorithms successfully estimate the direction of symmetry (epipole) for these images along with reasonable disparity ranges for the objects. They also correctly match a large number of edges under challenging conditions that include textureless regions and background clutter. For well textured scenes, such as for the fourth row in Figure 8, dense pixel-based matching may work better [8]. However, our technique is complementary to theirs and can compute a sparse set of matches more reliably which is important when the photometric cues are less reliable as in our example images.

The supplementary materials include additional examples, as well as video animations that can better visualize the recovered depth maps [30]. Our method sometimes fails to match enough features to accurately recover the axis of symmetry and can have difficulty dealing with highly repetitive patterns which may overwhelm the curve matcher. Although the ordering constraint in our curve matching approach is beneficial in most cases, it can produce erroneous results for certain 3D objects such as the chair shown in the sixth row in Figure 8. Also as discussed earlier, the occluding contours for symmetric rounded objects need not be symmetric and in such cases, the depths computed by our method at pixels on the occluding contour can be inaccurate.

8 Conclusions

In this paper, we have developed a system to detect and reconstruct bilaterally symmetric 3D objects with interesting 3D structure. Our system first extracts highly repeatable keypoints and then generates descriptors from affinely deformed patches around these keypoints. It then determines the direction of symmetry using pairs of corresponding points, rectifies the image, and then computes a plausible disparity range based on the spatial extent of the inliers. We then use a novel two-stage dynamic programming algorithm to match extended curves extracted from the rectified image. This allows us to reconstruct textureless, specular objects which are traditionally problematic for pixel-based stereo. In the future, we would like to combine curve-based matching with pixel-level matching and also explore foreground segmentation and the use of the recovered sparse 3D geometry to aid the recognition of 3D symmetric objects.

References

1. Liu, Y., Hel-Or, H., Kaplan, C.S., Gool, L.V.: Computational symmetry in computer vision and computer graphics. FnT CGCV 5, 1–195 (2010)
2. Tuytelaars, T., Turina, A., Van Gool, L.: Noncombinatorial detection of regular repetitions under perspective skew. IEEE PAMI 25, 418–432 (2003)
3. Loy, G., Eklundh, J.-O.: Detecting Symmetry and Symmetric Constellations of Features. In: Leonardis, A., Bischof, H., Pinz, A. (eds.) ECCV 2006. LNCS, vol. 3952, pp. 508–521. Springer, Heidelberg (2006)
4. Cornelius, H., Perdoch, M., Matas, J(G.), Loy, G.: Efficient Symmetry Detection Using Local Affine Frames. In: Ersbøll, B.K., Pedersen, K.S. (eds.) SCIA 2007. LNCS, vol. 4522, pp. 152–161. Springer, Heidelberg (2007)
5. Cho, M., Lee, K.M.: Bilateral symmetry detection and segmentation via symmetry-growing. In: BMVC 2009, pp. 4.1–4.1. BMVA Press (2009)
6. Rauschert, I., Brocklehurst, K., Kashyap, S., Liu, J., Liu, Y.: First symmetry detection competition: Summary and results. Technical Report CSE11-012, Department of Computer Science and Engineering, The Pennsylvania State University (2011)
7. Wu, C., Frahm, J.M., Pollefeys, M.: Repetition-based dense single-view reconstruction. In: CVPR, pp. 3113–3120 (2011)
8. Köser, K., Zach, C., Pollefeys, M.: Dense 3D Reconstruction of Symmetric Scenes from a Single Image. In: Mester, R., Felsberg, M. (eds.) DAGM 2011. LNCS, vol. 6835, pp. 266–275. Springer, Heidelberg (2011)
9. Mitsumoto, H., et al.: 3-D reconstruction using mirror images based on a plane symmetry recovering method. IEEE PAMI 14, 941–946 (1992)
10. François, A.R.J., Medioni, G.G., Waupotitsch, R.: Mirror symmetry ⇒ 2-view stereo geometry. IVC 21, 137–143 (2003)
11. Jiang, N., Tan, P., Cheong, L.-H.: Symmetric architecture modeling with a single image. ACM Transactions on Graphics 28 (2009)
12. Shimshoni, I., Moses, Y., Lindenbaum, M.: Shape Reconstruction of 3D Bilaterally Symmetric Surfaces. IJCV 39, 97–110 (2004)
13. Xue, T., Liu, J.: Symmetric piecewise planar object reconstruction from a single image. In: CVPR, pp. 2577–2584 (2011)
14. Sun, Y., Bhanu, B.: Reflection symmetry integrated image segmentation. IEEE PAMI (2011)
15. Hong, W., Yang, A.Y., Huang, K., Ma, Y.: On symmetry and multiple-view geometry: Structure, pose, and calibration from a single image. IJCV 60, 241–265 (2004)
16. Mitra, N.J., Guibas, L.J., Pauly, M.: Partial and approximate symmetry detection for 3D geometry. ACM Transactions on Graphics 25, 560–568 (2006)
17. Wu, C., Frahm, J.-M., Pollefeys, M.: Detecting Large Repetitive Structures with Salient Boundaries. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part II. LNCS, vol. 6312, pp. 142–155. Springer, Heidelberg (2010)
18. Okutomi, M., Kanade, T.: A multiple baseline stereo. IEEE PAMI 15, 353–363 (1993)
19. Hartley, R.I., Zisserman, A.: Multiple View Geometry. Cambridge University Press, Cambridge (2004)
20. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. IJCV 60, 91–110 (2004)
21. Mikolajczyk, K., Schmid, C.: Scale & affine invariant interest point detectors. IJCV 60, 63–86 (2004)

22. Morel, J.M., Yu, G.: ASIFT: A new framework for fully affine invariant image comparison. SIAM J. Imaging Sciences 2, 438–469 (2009)
23. Brown, M., Hua, G., Winder, S.: Discriminative learning of local image descriptors. IEEE PAMI 33, 43–57 (2011)
24. Fischler, M.A., Bolles, R.C.: Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM 24, 381–395 (1981)
25. Scharstein, D., Szeliski, R.: A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. IJCV 47, 7–42 (2002)
26. Ohta, Y., Kanade, T.: Stereo by intra- and inter-scanline search using dynamic programming. IEEE PAMI PAMI-7, 139–154 (1985)
27. Collins, R.T.: A space-sweep approach to true multi-image matching. In: CVPR, pp. 358–363 (1996)
28. Rosten, E., Porter, R., Drummond, T.: Faster and better: A machine learning approach to corner detection. IEEE PAMI 32, 105–119 (2010)
29. Zitnick, C.L., Ramnath, K.: Edge foci interest points. In: ICCV, pp. 359–366 (2011)
30. `http://research.microsoft.com/en-us/um/redmond/groups/ivm/SymmRecon3D`