symmetry

MDPI

Article

Binocular 3D Object Recovery Using a Symmetry Prior

Aaron Michaux [1,*], Vikrant Kumar [2], Vijai Jayadevan [1], Edward Delp [1] and Zygmunt Pizlo [2]

[1] School of Electrical and Computer Engineering, Purdue University, 465 Northwestern Avenue, West Lafayette, IN 47907, USA; vthottat@purdue.edu (V.J.); ace@ecn.purdue.edu (E.D.)
[2] Department of Psychological Sciences, Purdue University, 703 3rd Street, West Lafayette, IN 47907, USA; vikrant1998@gmail.com (V.K.); zpizlo@purdue.edu (Z.P.)
* Correspondence: amichaux@purdue.edu

Academic Editor: Marco Bertamini
Received: 16 January 2017; Accepted: 24 April 2017; Published: 28 April 2017

Abstract: We present a new algorithm for 3D shape reconstruction from stereo image pairs that uses mirror symmetry as a biologically inspired prior. 3D reconstruction requires some form of prior because it is an ill-posed inverse problem. Psychophysical research shows that mirror-symmetry is a key prior for 3D shape perception in humans, suggesting that a general purpose solution to this problem will have many applications. An approach is developed for finding objects that fit a given shape definition. The algorithm is developed for shapes with two orthogonal planes of symmetry, thus allowing for straightforward recovery of occluded portions of the objects. Two simulations were run to test: (1) the accuracy of 3D recovery, and (2) the ability of the algorithm to find the object in the presence of noise. We then tested the algorithm on the *Children's Furniture Corpus*, a corpus of stereo image pairs of mirror symmetric furniture objects. Runtimes and 3D reconstruction errors are reported and failure modes described.

Keywords: symmetry prior; symmetry detection; stereo; two-view geometry; 3D recovery

1. Introduction

The central importance of symmetry is well established in diverse fields such as physics, chemistry, mathematics, and art; however, it has received comparatively little attention in computer vision. Early computer vision researchers were interested in 3D recovery (for examples, see: [1–4]), and, as will be discussed later in this paper, symmetry is a powerful regularity that allows for 3D reconstruction. However, perhaps owing to the difficulty of the problem, interest in 3D recovery has been supplanted by machine learning approaches that almost always rely on purely 2D image features. (In the case of deep learning, instead of merely learning from 2D features, the 2D features themselves are learned from the training data [5].) Despite this trend in computer vision, it is widely recognized that human vision does perform 3D recovery [6], and, to this day, 3D recovery continues to be an important topic in computer vision.

Camera image formation is the forward problem of vision, and it is well-posed and well understood. Indeed, we know how to produce realistic looking images from 3D models [7]. However, there are an infinity of possible 3D models that can produce any given camera image, and that makes the inverse problem, 3D recovery from camera images, ill-posed [8]. In a classic 1985 paper, Poggio et al. [9] introduced the notion of computational vision, which uses regularization to solve early vision problems, such as 3D recovery. In essence, this means that all solutions to 3D recovery must involve a priori assumptions that constrain the possible recoveries. If we adopt a computational model of human cognition [10], as was done by David Marr [4], then the existence of innate a priori constraints also applies to human vision [11,12]. Considering that humans have definite ideas about the 3D shapes of objects that they perceive—Shepard and Metzler [13] proved as much with their

classic experiment on the mental rotation of objects—it makes sense to investigate and understand what priors the human vision system uses in 3D shape perception.

Recent advances in human perception have shown that symmetry, and mirror symmetry in particular, is the most important prior in 3D shape perception [12,14,15]. Li et al. [16] argued further that an object's shape is defined by its symmetries. That is, an object has as much shape as it has symmetry or regularity. We consider it natural, therefore, to investigate mirror symmetry as an informative prior for 3D shape recovery in the spirit of Poggio et al.'s paradigm of computational vision.

In this paper, we present a new algorithm for performing 3D recovery for mirror symmetric objects from a pair of stereo images. There has been a small amount of persistent interest in using mirror symmetry to perform 3D recovery; however, almost no attempts use both mirror symmetry and two-view geometry at the same time. However, there is psychophysical evidence that the human visual system does precisely this: using both mirror symmetry and two-view geometry to aid 3D shape perception [17]. The advantage of combining mirror symmetry with two-view geometry is that each problem disambiguates the other, making it simpler to solve both simultaneously. (This is developed in Sections 3.2 and 3.3). For the sake of computational efficiency, we added the additional constraint that objects stand on a flat surface, or floor. This is easily arranged when capturing images in controlled conditions; however, most real world objects do stand on a flat surface [12], and there is psychophysical evidence that the human vision system takes advantage of this prior [18].

2. Related Works

As early as 1978, Marr and Nishihara [19] emphasized the importance of 3D representations in computer vision. In particular, Marr and Nishihara advocated for symmetric parts based on Binford's generalized cones [20]. However, this work did not address the mathematical problem of recovering 3D symmetric shapes from camera images. Kanade, 1981, did provide details for 3D reconstruction [3], but only for skew-symmetry (i.e., orthographic projections of planar symmetric figures). It was not until 1990 that Gordon first published details for 3D reconstruction from the perspective images of mirror symmetric objects [21], although Gordon credits the Oxford University engineer Guy Scott with the idea of calculating depth information from symmetry.

Since then, the field has grown immensely richer. A key observation was that mirror symmetry is a variant of two-view geometry [22]. That is, the image of a symmetric object allows for computing a second view of the same object. This is true except for degenerate cases and can be generalized to other types of symmetry. For example, the left and right side of the face are roughly identical, and thus the image of a face can be thought of as two aligned views of a single half. This provides a variety of mathematical invariants that form a theoretical basis for identifying symmetry, and for performing 3D reconstruction [23–25]—a field which could perhaps be called *structure from symmetry* [26].

The key practical problem in solving structure from symmetry is identifying which pairs of image points "go together". Returning to the example of the face, a point on the tip of the left ear goes with a specific point on the tip of the right ear: no other point will do. Registering these points in the 2D camera image is called the *symmetry correspondence problem*, and once done, the positions of the 2D points can be corrected at the subpixel level [27]. However, a general solution to the symmetry correspondence problem is currently beyond the state of the art. Few invariants exist for the general case problem, and any pair of imaged 2D curves is consistent with some 3D mirror symmetric interpretation [28].

It is natural to try to restrict the problem (following in the footsteps of Kanade) to orthographic images of planar symmetric figures, such as symmetric designs on walls and patterns on carpets. In this case, correspondences can be found using local 2D features that are robust, or invariant, to local affine distortions [29–32].

Despite challenges, solving symmetry correspondence for more general symmetric figures has been a topic of ongoing interest. One approach noted that mirror symmetry extends to the surface normals of a shape, providing a useful invariant for performing dense reconstruction using shape from

shading [33]. Dense reconstruction has also been performed on mirror symmetric scenes by matching texture patches under smoothness constraints [34]. For scenes characterized by translational symmetry, dense reconstruction was performed using a graph-cut, with an energy function that incorporates symmetrical repetition [35].

More typically, researchers solve symmetry correspondences on edge maps by trying to register pairs of smooth contours. One early approach identified polygons in the 2D image that obey invariants for some form of 3D symmetry. These polygons are then registered with each other, allowing for 3D reconstruction [36]. Alternatively, it is possible to match line features directly in a loop that validates the match geometry for some form of 3D symmetry [37]. Instead of matching line features, one can attempt to trace pairs of contours, matching key features along the way, such as turning angle. Two studies follow this approach, using sets of human generated contours as input [38,39]. Öztireli et al. [40] extracted feature points and contours from rasterized images; however, human intervention is required to select points in order to facilitate curve matching. Sinha et al. [41] put forward an interesting dynamic programming approach that simultaneously extracts and matches contours; however, it can only work when the viewing angle of the 3D mirror symmetric object is restricted.

The algorithm presented in this paper solves symmetry correspondences on rasterized edge maps, and without human intervention, by using the two-view geometry of stereo image pairs to disambiguate possible correspondences. It is possible to perform 3D reconstruction from pairs of stereo images alone [22]; however, the results are usually noisy due to image pixelation. This paper presents an alternative point of view for using symmetry and two-view geometry together: symmetry can be used to improve the quality of 3D reconstructions, in effect performing subpixel accurate correspondence. This approach was pursued to clean up the 3D point clouds generated from the application of structure from motion [42,43].

Symmetry, however, does more than just provide subpixel correction, or accurate 3D reconstruction. Symmetry is a global property, and thus it can be used to perform *figure/ground organization*: localizing individual objects in a scene, and separating them from the background. In previous work, we investigated using mirror symmetry with two-view geometry to perform figure/ground organization [44]. However, as far back as 1997, Zabrodsky and Weinshall [45] observed that mirror symmetry and two-view geometry can be used together. They developed a graph based method for matching corresponding points, and tested it mainly on synthetic images. However, they also ran two tests using, respectively, three and five camera views of a 3D symmetric object and reported the average 3D reconstruction error on sparse sets of manually selected interest points. This can be taken as part of the inspiration for the algorithm presented in this paper.

3. Proposed Algorithm

We present an algorithm that performs 3D reconstruction of mirror symmetric objects from a pair of stereo images, where the symmetry is evident in the contours of the object. In many ways, the algorithm is a successor to the algorithm presented in [44]; however, it has been repurposed to perform 3D object reconstruction. Comparisons with [44] are detailed in Sections 4.3.1 and 4.3.3. While it is possible that this algorithm may have practical uses, it is a proof of concept for a family of methods that perform 3D object recovery using symmetries. The specific types of objects recovered depend on how the object's shape is defined. For this proof of concept, we use Definition 1 for the shape of an object.

Definition 1. *An object's shape is a set of 3D points that are mirror symmetric about two orthogonal planes of symmetry.*

This definition covers a variety of manufactured objects, including some pieces of furniture. One advantage of this definition is that the two planes of symmetry can be used to reconstruct the backs of objects. This is made clear in Sections 3.6 and 4.3.2. The reasons for restricting the algorithm

to objects with definite edges, and the formulation of the 3D recovery as an optimization problem is developed in the rest of this section.

3.1. Notation

We use lowercase and uppercase bold symbols for points in $\mathbb{R}^2$ and $\mathbb{R}^3$, respectively. Therefore, for example, $\boldsymbol{x} \in \mathbb{R}^2$, and $\boldsymbol{X} \in \mathbb{R}^3$. Furthermore, we add a superscript asterisk for points in the respective projective spaces. Thus, $\boldsymbol{x}^* \in \mathbb{P}^2$ and $\boldsymbol{X}^* \in \mathbb{P}^3$. The exception is unit vectors in $\mathbb{R}^3$, which are written $\tilde{\boldsymbol{n}}$ (lowercase because they have only two degrees of freedom). Subscripts refer to a particular camera, if relevant. For example, $\boldsymbol{x}_1^*$, and $\boldsymbol{x}_2^*$ refer to point $\boldsymbol{x}^*$ in camera 1 and camera 2. The function $E(\boldsymbol{x}^*)$ is used to convert homogeneous coordinates into equivalent Cartesian points. Thus, $\|E(\boldsymbol{x}^*)\|$ refers to the L^2 norm of the Cartesian equivalent to the homogeneous coordinate $\boldsymbol{x}^*$.

In this paper, we use a pinhole camera model and only consider calibrated cameras. We use K for the intrinsic camera matrix, and $P = K[R \mid \boldsymbol{t}]$ for the camera matrix. R and $\boldsymbol{t}$ are the extrinsic camera parameters: R a rotation matrix, and $\boldsymbol{t} = -R\boldsymbol{C}$, where $\boldsymbol{C}$ is the coordinates of the camera center. Again, we use subscripts if a particular camera is indicated. Thus, $\boldsymbol{x}_1^* = P_1\boldsymbol{X}^*$ gives the projection of point $\boldsymbol{X}^*$ in camera 1. All intrinsic parameters are assumed to be known, as is the relative rotation and translation between the camera pair, and thus we use a calibrated stereo system.

Sometimes, we need to refer to the reprojection error of an image point relative to some estimate of a 3D point. For example, point $\boldsymbol{X}^*$ may be imaged in camera 1 as point $\boldsymbol{x}^*$, but with some errors. In this case, the reprojection error is expressed as: $d(\boldsymbol{x}^*, P_1\boldsymbol{X}^*) = \|E(\boldsymbol{x}^*) - E(P_1\boldsymbol{X}^*)\|$. For those unfamiliar with homogeneous coordinates, camera matrices, and projective spaces, please refer to Hartley and Zisserman [22].

3.2. 3D Mirror Symmetry and Projective Geometry

An object is mirror symmetric in 3D if the object is invariant (unchanged) when reflected through a single plane of symmetry, $\boldsymbol{\pi} = \begin{bmatrix} \tilde{\boldsymbol{n}}^\top & d \end{bmatrix}^\top$. Thus, Equation (1) holds true for any point on the object S:

$$\forall \boldsymbol{U} \in S, \quad \exists \boldsymbol{V} \in S \quad s.t. \quad \boldsymbol{V} = \boldsymbol{U} - 2(\tilde{\boldsymbol{n}}^\top \boldsymbol{U} + d)\tilde{\boldsymbol{n}}. \tag{1}$$

In this formulation of the plane equation, $\boldsymbol{\pi} = \begin{bmatrix} \tilde{\boldsymbol{n}}^\top & d \end{bmatrix}^\top$, the point plane distance is given by $\tilde{\boldsymbol{n}}^\top \boldsymbol{U} + d$, and $\tilde{\boldsymbol{n}}$ (a unit vector) gives the normal to the symmetry plane, which will henceforth be referred to as the *direction of symmetry*. A mirror symmetric object is shown in Figure 1.

Object Definition 1 refers to two orthogonal planes of symmetry. In this case, Equation (1) is true for each plane of symmetry, and furthermore, the planes of symmetry are orthogonal to each other. Planes $\boldsymbol{\pi}_1$ and $\boldsymbol{\pi}_2$ are orthogonal to each other if, and only if, the respective directions of symmetry are orthogonal to each other.

We use a pinhole camera model for a calibrated camera with intrinsic matrix K, and with neither radial nor skew distortion. Let P be the camera matrix, and let $\boldsymbol{U}^*$ and $\boldsymbol{V}^*$ be mirror symmetric points obeying the relationship given in Equation (1) (i.e., when expressed as Cartesian points in $\mathbb{R}^3$). Thus $\boldsymbol{u}^* = P\boldsymbol{U}^*$ and $\boldsymbol{v}^* = P\boldsymbol{V}^*$ are the points imaged by the camera.

The 3D line joining $\boldsymbol{U}$ and $\boldsymbol{V}$ has direction $\tilde{\boldsymbol{n}}$. Imagine starting at point $\boldsymbol{U}$ and traveling toward and through point $\boldsymbol{V}$ and continuing an infinite distance: to a point on $\boldsymbol{\pi}_\infty$, the plane at infinity in $\mathbb{P}^3$. This creates a new point $\boldsymbol{N}^* = \begin{bmatrix} \tilde{\boldsymbol{n}}^\top & 0 \end{bmatrix}^\top$, which has no Cartesian equivalent in $\mathbb{R}^3$. Then $\boldsymbol{n}^* = P\boldsymbol{N}^*$ is the *vanishing point* associated with the plane of symmetry. It is a rule of projective geometry that points on lines in $\mathbb{P}^3$ are imaged to points on lines in $\mathbb{P}^2$ (see [22]), and because $\boldsymbol{U}^*$, $\boldsymbol{V}^*$, and $\boldsymbol{N}^*$ are all co-linear, then so are $\boldsymbol{u}^*$, $\boldsymbol{v}^*$, and $\boldsymbol{n}^*$. We call $\boldsymbol{u}^*$ and $\boldsymbol{v}^*$ *corresponding points*, since they are related to each other by the mirror symmetry of $\boldsymbol{U}^*$ and $\boldsymbol{V}^*$. All corresponding points for the image of a mirror-symmetric object are co-linear with the vanishing point.

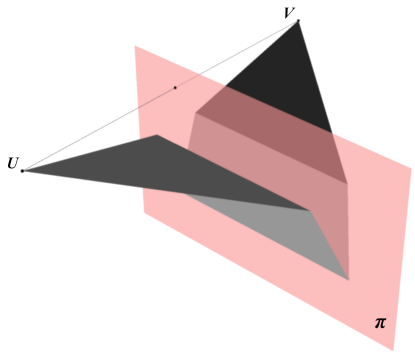


Figure 1. The figure is mirror symmetric about the plane $\boldsymbol{\pi} = \begin{bmatrix} \tilde{\boldsymbol{n}}^\top & d \end{bmatrix}^\top$. This means that every point $\boldsymbol{U}$ is related to some point $\boldsymbol{V} = \boldsymbol{U} - 2(\tilde{\boldsymbol{n}}^\top \boldsymbol{U} + d)\tilde{\boldsymbol{n}}$—for example, the wing-tips, as shown above. The line joining $\boldsymbol{U}$ to $\boldsymbol{V}$ is parallel to the direction of symmetry, $\tilde{\boldsymbol{n}}$, the normal of the symmetry plane.

This brings us to the mathematical relationship for reconstructing 3D points $\boldsymbol{U}$ and $\boldsymbol{V}$ up to scale from their imaged points $\boldsymbol{u}$ and $\boldsymbol{v}$, given that the vanishing point, $\boldsymbol{n}$, is known. First, write $\boldsymbol{U} = \|\boldsymbol{U}\|\tilde{\boldsymbol{u}} + \boldsymbol{C}$, and $\boldsymbol{V} = \|\boldsymbol{V}\|\tilde{\boldsymbol{v}} + \boldsymbol{C}$, where $\boldsymbol{C}$ is the camera center, and $\tilde{\boldsymbol{u}}$ and $\tilde{\boldsymbol{v}}$ are unit vectors. Let KR be the left-most 3×3 block of the camera matrix $P = K\,[R \mid t]$. This makes $\tilde{\boldsymbol{u}} = (KR)^{-1}\boldsymbol{u}^* / \|(KR)^{-1}\boldsymbol{u}^*\|$, and $\tilde{\boldsymbol{v}} = (KR)^{-1}\boldsymbol{v}^* / \|(KR)^{-1}\boldsymbol{v}^*\|$. Let $\tilde{\boldsymbol{n}} = (KR)^{-1}\boldsymbol{n}^* / \|(KR)^{-1}\boldsymbol{n}^*\|$ be the direction vector in $\mathbb{R}^3$ derived from the vanishing point. Note that $\tilde{\boldsymbol{n}}$ is the direction of symmetry set by the symmetry plane normal. Now, consider the angles between $\tilde{\boldsymbol{n}}$, and $\tilde{\boldsymbol{u}}$ and $\tilde{\boldsymbol{v}}$. Specifically, let $\theta = \cos^{-1}(\tilde{\boldsymbol{u}}^\top \tilde{\boldsymbol{n}})$ and $\phi = \cos^{-1}(\tilde{\boldsymbol{v}}^\top \tilde{\boldsymbol{n}})$. The angles θ and ϕ are illustrated in Figure 2.

Referring again to Figure 2, the line joining $\boldsymbol{U}$ and $\boldsymbol{V}$ is, by definition, parallel to the line extending from the camera center, $\boldsymbol{C}$, down the direction of symmetry $\tilde{\boldsymbol{n}}$. Let $\boldsymbol{U}'$ be the point on the ray through $\tilde{\boldsymbol{n}}$ such that $\boldsymbol{C}\boldsymbol{U}'\boldsymbol{U}$ makes a right angle triangle, and let $\lambda = \|\boldsymbol{U} - \boldsymbol{U}'\|$ be the distance between $\boldsymbol{U}$ and $\boldsymbol{U}'$. Then, by construction, $\sin\theta = \lambda / \|\boldsymbol{U}\|$. Similarly, if $\boldsymbol{V}'$ is the point on the ray through $\tilde{\boldsymbol{n}}$ such that $\boldsymbol{C}\boldsymbol{V}'\boldsymbol{V}$ makes a right angle triangle, then $\sin\phi = \lambda / \|\boldsymbol{V}\|$. The relationship $\lambda = \|\boldsymbol{U} - \boldsymbol{U}'\| = \|\boldsymbol{V} - \boldsymbol{V}'\|$ holds because the four points $\{\boldsymbol{U}, \boldsymbol{V}, \boldsymbol{U}', \boldsymbol{V}'\}$ form a rectangle. This gives Equation (2), which relates the length $\|\boldsymbol{V}\|$ relative to $\|\boldsymbol{U}\|$, with the angle formed by their direction vectors with the direction of symmetry. This relationship is shown in Figure 2:

$$\frac{\sin\theta}{\sin\phi} = \frac{\|\boldsymbol{V}\|}{\|\boldsymbol{U}\|}. \tag{2}$$

To solve for $\boldsymbol{U}$ and $\boldsymbol{V}$, note that their midpoint lies on the symmetry plane $\boldsymbol{\pi} = \begin{bmatrix} \tilde{\boldsymbol{n}}^\top & d \end{bmatrix}^\top$. That is: $\frac{1}{2}(\boldsymbol{U} + \boldsymbol{V})^\top \tilde{\boldsymbol{n}} + d = 0$. Thus, $(\boldsymbol{U} + \boldsymbol{V})^\top \tilde{\boldsymbol{n}} = \left(\|\boldsymbol{U}\|\tilde{\boldsymbol{u}} + \boldsymbol{C} + \frac{\sin\theta}{\sin\phi}\|\boldsymbol{U}\|\tilde{\boldsymbol{v}} + \boldsymbol{C}\right)^\top \tilde{\boldsymbol{n}} = -2d$, giving Equation (3):

$$\|\boldsymbol{U}\| = \frac{-2(d + \boldsymbol{C}^\top \tilde{\boldsymbol{n}})}{\tilde{\boldsymbol{u}}^\top \tilde{\boldsymbol{n}} + \frac{\sin\theta}{\sin\phi}\tilde{\boldsymbol{v}}^\top \tilde{\boldsymbol{n}}}. \tag{3}$$

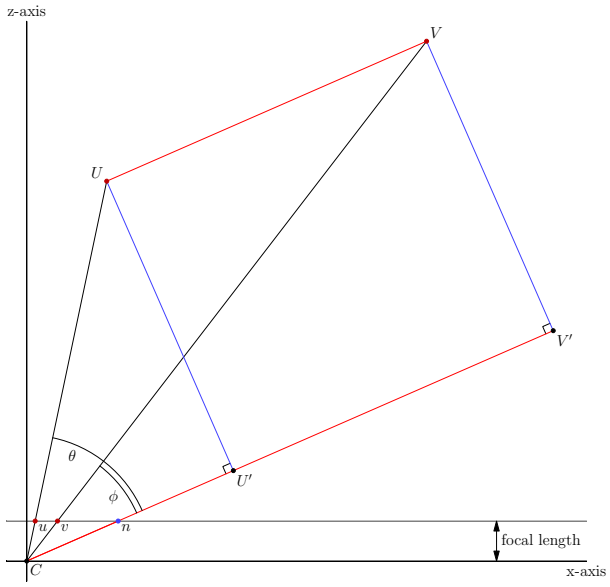


Figure 2. Diagram that shows the relationship expressed in Equation (2). Specifically because the points $\{\boldsymbol{U}, \boldsymbol{V}, \boldsymbol{U}^{'}, \boldsymbol{V}^{'}\}$ form a rectangle, the length $\lambda = \|\boldsymbol{U} - \boldsymbol{U}^{'}\| = \|\boldsymbol{V} - \boldsymbol{V}^{'}\|$. Then, we have the ratio $\sin\theta / \sin\phi = \|V\| / \|U\|$. The diagram is patterned after [44]. An interactive 3D version of this diagram is available in the supplementary materials.

Substitute from Equation (3) into Equation (2) to calculate the length $\|\boldsymbol{V}\|$. Note that this formula requires a calibrated camera with a finite focal length because, otherwise, the angles θ and ϕ cannot be determined. Furthermore, the image points $\boldsymbol{u}$ and $\boldsymbol{v}$ must be distinct. There is a degenerate case when the plane of symmetry goes through the camera center, in which case both the numerator and denominator of Equation (3) become zero. (i.e., $d + \boldsymbol{C}^{\top}\tilde{\boldsymbol{n}} = 0$, and the denominator can be rearranged to show that $\tan\theta + \tan\phi = 0$.) Except for this degenerate case, a set of 3D mirror symmetric points can be recovered up to scale from a vanishing point, and a set of corresponding points in the image. The scale of the recovered object is set by d, the distance between the symmetry plane and the origin. For further discussion, see [44]. See [21,28,46] for alternative formulations.

3.3. Solving the Symmetry Correspondence Problem with Two-View Geometry

Solving the symmetry correspondence problem is all that is required to solve for the 3D shape of a mirror symmetric object. However, as described in Section 2, there is no known robust and general purpose method for solving symmetry correspondence, and it remains a topic of ongoing research. Indeed, so long as they are co-linear with the vanishing point, one can select arbitrary pairs of corresponding points and reconstruct a 3D object [12]. This applies in the continuous case as well, and any pair of 2D curves is consistent with some pair of 3D mirror symmetric curves [28]. One key issue is that a degenerate viewing angle can hide details in the images of one or both of the 3D curves, as is shown in Figure 3.

If we consider 3D mirror symmetry and two-view geometry together, the situation is simplified. For a two-view camera system, a single point in 3D is imaged in two different cameras. That is, point $\boldsymbol{X}^*$ is imaged in camera 1 as $\boldsymbol{x}_1^* = P_1\boldsymbol{X}^*$ and in camera 2 as $\boldsymbol{x}_2^* = P_2\boldsymbol{X}^*$. This pair of points, $\boldsymbol{x}_1^*$ and $\boldsymbol{x}_2^*$, is also known as a *corresponding pair*, but for the binocular correspondence problem. (for complete coverage, see: [22].)

Figure 3. These four images are generated from the same pair of 3D mirror symmetric curves but seen from different vantage points. On the far left, the viewing angle is such that the curve images do not suggest 3D symmetry. Nonetheless, the relationship is there, and it becomes clear as the view is rotated, especially in the two right-most images. These images were published in [28].

Consider shape S as a set of points in $\mathbb{R}^3$. Each camera gives a unique image of the points in S, and thus a prospective pair of 3D symmetric curves are imaged uniquely in each camera. This observation underlies Equation (4), which states that when the points $\boldsymbol{X} \in S$ are projected using camera matrix P_i, the reprojection error (defined in Section 3.1) to a point in the relevant binary edge map $\texttt{EdgeMap}_i$ is less than some threshold ϵ. (A binary edge map is a binary image where each pixel indicates the presence or absence of an edge.). For two-view geometry, we have $i \in \{1, 2\}$:

$$\forall \boldsymbol{X}^* \in S \quad \exists \boldsymbol{x}_i^* \in \texttt{EdgeMap}_i, \quad s.t. \quad d(\boldsymbol{x}_i^*, P_i \boldsymbol{X}^*) < \epsilon. \tag{4}$$

Taken together, Equations (1) and (4) specify a 3D mirror symmetric shape that is consistent with evidence in two (or perhaps more) camera images. Only two visible edge points are required for each image. Figure 4 illustrates what is gained by combining mirror symmetry with two-view geometry. Let $\boldsymbol{U}$ and $\boldsymbol{V}$ be 3D points that are mirror symmetric about a symmetry plane $\boldsymbol{\pi}$ with direction of symmetry $\tilde{\boldsymbol{n}}$. Let $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$ be the images of $\boldsymbol{U}$ in cameras 1 and 2, and likewise, let $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ by the images of $\boldsymbol{V}$ in cameras 1 and 2. The points $\boldsymbol{u}_1$ and $\boldsymbol{v}_1$ are symmetric corresponding points, and must be co-linear with $\boldsymbol{n}_1$, the image of $\tilde{\boldsymbol{n}}$ on the plane at infinity (as described in Section 3.2). Likewise, $\boldsymbol{u}_2$ and $\boldsymbol{v}_2$ are co-linear with $\boldsymbol{n}_2$ in the second camera image. Note also that $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$ correspond in the binocular sense, and so do $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$. Each individual point participates in two different epipolar geometries, and thus the entire system is over-constrained, with the locations of each of the four points constrained by both the direction of symmetry, $\tilde{\boldsymbol{n}}$, and *Fundamental matrix* [22], which constrains the locations of binocularly corresponding points according to the relative rotation and translation between the two cameras.

Thus, Equations (1) and (4) together specify 3D points that are consistent with this over-constrained geometry and provide an avenue to solve both the binocular and the symmetry correspondence problem at the same time. In effect, the symmetry correspondence problem disambiguates the binocular correspondence problem, and vice versa. This restricts the algorithm to those objects whose mirror symmetry is evident in the binary edge maps of both images. When one or both projected points are missing (for example, they are occluded, or not picked up by edge detection), then that pair of 3D points are omitted in shape S.

The proposed algorithm requires an efficient method for validating if a given shape S satisfies the image evidence. We first calculate binary edge maps for each image using the Canny operator with an adaptive filter [47]. The edge points are loaded into an R-tree [48], thus creating a spatial index for the edge points in each image. Then, for each point $\boldsymbol{X}^* \in S$, the spatial index is used to query the closest point to $E(P_i X^*)$ in camera i. The reprojection error is given by the L^2 distance between the queried point and the projected point, and this error must be less than ϵ. This procedure has average time complexity $O(mnk)$, where m is the number of cameras, $n = |S|$ is the number of 3D points in shape S, and k is the expected time complexity for looking up single points from the spatial index.

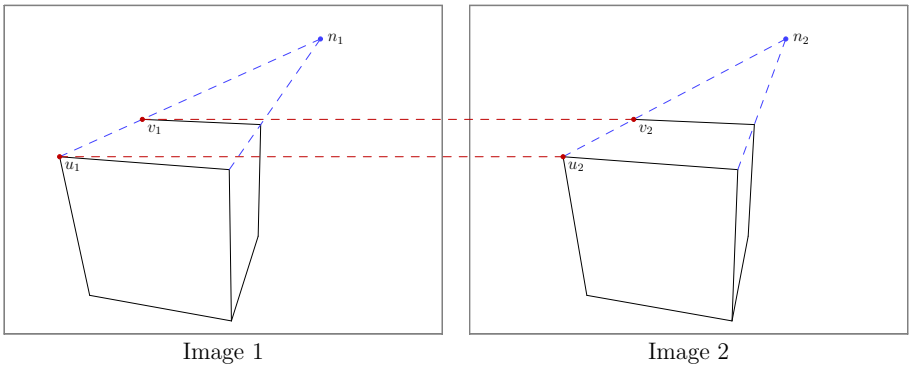


Figure 4. Points $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$ are the images of some point $\boldsymbol{U} \in \mathbb{R}^3$, and points $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ are the images of some point $\boldsymbol{V} \in \mathbb{R}^3$. Let $\tilde{\boldsymbol{n}}$ be the direction vector between $\boldsymbol{U}$ and $\boldsymbol{V}$, and $\boldsymbol{N}^*$ the intersection of $\tilde{\boldsymbol{n}}$ with the plane at infinity (see Section 3.2). Let $\boldsymbol{n}_1$ and $\boldsymbol{n}_1$ be the images of $\boldsymbol{N}^*$ in camera 1 and 2. Then, as described in Section 3.2, $\boldsymbol{u}_1$, $\boldsymbol{v}_1$, and $\boldsymbol{n}_1$ are co-linear, and likewise $\boldsymbol{u}_2$, $\boldsymbol{v}_2$, and $\boldsymbol{n}_2$ are co-linear. Furthermore, $\boldsymbol{u}_1$ and $\boldsymbol{u}_2$ are co-linear with the binocular epipole, a point in $\mathbb{P}^2$ calculated from the relative rotation and translation between the two cameras. See [22] for a full treatment. Likewise, $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ are also co-linear with the binocular epipole. This creates an over-constrained system, where the image points $\boldsymbol{u}_1$, $\boldsymbol{v}_1$, $\boldsymbol{u}_2$, and $\boldsymbol{v}_2$ are restricted by the direction vector between the two 3D points that generated them (i.e., $\boldsymbol{U}$ and $\boldsymbol{V}$), and the relative rotation and translation between the two cameras.

3.4. Using a Floor Prior

Most objects stand on a floor, or flat surface, and there is psychophysical evidence that the human visual system uses some sort of floor prior [12,18]. Using a floor prior is similar to the "Manhattan World" assumption [49], where a scene is assumed to have three global vanishing points related to an orthogonal basis in $\mathbb{R}^3$. In this case, the normal to the floor plane corresponds to the "up direction" in a Manhattan World. However, the floor prior by itself is more flexible, admitting objects with a single plane of symmetry (thus two vanishing points in total), or perhaps other symmetries. Furthermore, floor plane estimation is robust and stable under two-view geometry when many floor points are visible—a typical scenario when a particular application allows for the control of camera viewing angles.

We assume that objects stand (or lie) on the floor, and that their symmetry planes are orthogonal to the floor plane. This restricts the object's direction of symmetry to the one-dimensional subspace orthogonal to the floor plane normal. The projection of this subspace is called the *horizon line*, and is the intersection of the image plane with the plane parallel to the floor that also contains the camera center.

The floor prior is found using a procedure developed by Li et al. [46]. First, a disparity map is calculated using OpenCV's StereoBM algorithm [50], which matches patches of texture between a pair of rectified stereo gray-scale images. Triangulation [22] is then applied to the disparity map to generate a 3D point cloud. Finally, *Random sample consensus* (RANSAC [51]) is used to find the equation of the plane that contains the most 3D points within a specified error threshold. This method works well when the floor is visible and is robust to the noise typical of 3D reconstructions from stereo image pairs.

3.5. Finding Symmetry Planes

As pointed out in Section 3.2, finding vanishing points is equivalent to finding the direction of symmetry for a symmetry plane. The proposed algorithm extends existing feature based approaches to vanishing point detection by taking advantage of both 3D symmetry and two-view geometry. Equation (3) shows that the vanishing point alone would be enough to recover the 3D object up to scale; however, scale is important in two-view geometry and is required for Equation (4). Thus, we must estimate both the direction of symmetry and d, the distance between the symmetry plane and the origin.

We use the Harris operator [52] with parameters set to over detect corner features. Corner features are then registered with each other across both input images using the disparity map previously calculated when finding the floor prior. Let $\boldsymbol{\eta} = \begin{bmatrix} \tilde{\boldsymbol{g}}^\top & d_\eta \end{bmatrix}^\top$ be the equation for the floor, where $\tilde{\boldsymbol{g}}$ is suggestive of the direction of gravity. Let $\boldsymbol{x}_i^*$ be an image point in camera i. Equation (5) is the ray-plane intersection that denotes the point on the floor plane that images to $\boldsymbol{x}_i^*$. This situation is illustrated in Figure 5:

$$\mathtt{F}(\boldsymbol{x}_i^*, i) = \boldsymbol{C}_i - \frac{\tilde{\boldsymbol{g}}^\top \boldsymbol{C}_i + d_\eta}{\tilde{\boldsymbol{g}}^\top (KR)_i^{-1} \boldsymbol{x}_i^*} (KR)_i^{-1} \boldsymbol{x}_i^*. \tag{5}$$

If we assume that $\boldsymbol{u}_i^*$ and $\boldsymbol{v}_i^*$ are a corresponding pair of points, then the direction of symmetry, $\tilde{\boldsymbol{n}}$, can be estimated in camera i by finding the unit vector between $F(\boldsymbol{u}_i^*, i)$ and $F(\boldsymbol{v}_i^*, i)$. Equation (6) gives the estimate for $\tilde{\boldsymbol{n}}$ by averaging across both cameras:

$$\tilde{\boldsymbol{n}} = \frac{1}{2} \sum_{i=1}^{2} \frac{F(\boldsymbol{u}_i^*, i) - F(\boldsymbol{v}_i^*, i)}{\| F(\boldsymbol{u}_i^*, i) - F(\boldsymbol{v}_i^*, i) \|}. \tag{6}$$

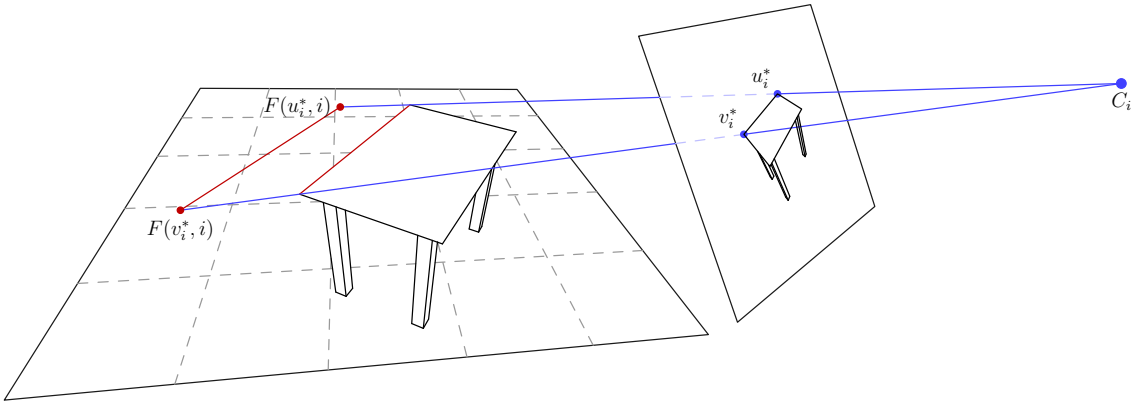


Figure 5. Method for estimating the direction of symmetry $\tilde{\boldsymbol{n}}$ from assumed corresponding points $\boldsymbol{u}_i^*$ and $\boldsymbol{v}_i^*$. Equation (5) is used to find $F(\boldsymbol{u}_i^*, i)$, and $F(\boldsymbol{v}_i^*, i)$, the points on the floor plane that image to $\boldsymbol{u}_i^*$ and $\boldsymbol{v}_i^*$. The vector $F(\boldsymbol{u}_i^*, i) - F(\boldsymbol{v}_i^*, i)$ is an estimate for the direction of symmetry, but for a single pair of points in a single camera. Equation (6) estimates $\tilde{\boldsymbol{n}}$ by averaging the normalized estimates across all cameras.

To estimate d, note that if $\boldsymbol{U}$ and $\boldsymbol{V}$ are symmetric about $\boldsymbol{\pi}$, then the midpoint $\boldsymbol{M} = \frac{1}{2}(\boldsymbol{U} + \boldsymbol{V})$ must lie on the symmetry plane. That is, $\boldsymbol{M}^\top \tilde{\boldsymbol{n}} = d$. To find $\boldsymbol{M}$, we first select an arbitrary d, say $d' = 1$. Then, using Equations (2) and (3), we recover $\boldsymbol{U}_i$ and $\boldsymbol{V}_i$ to scale, but for each camera image. Then, $\hat{\boldsymbol{m}}_i = \frac{1}{2}(KR)_i^{-1}(\boldsymbol{U}_i + \boldsymbol{V}_i)$ is the back projection from camera center $\boldsymbol{C}_i$ that images to point $\boldsymbol{m}_i^*$ in camera image i. That is, $\boldsymbol{m}_i^* = P_i(\hat{\boldsymbol{m}}_i + \boldsymbol{C}_i)$. We then use triangulation [22] in Equation (7) to estimate d by first finding $\boldsymbol{M}$:

$$d = \mathtt{triangulate}(P_1(\hat{\boldsymbol{m}}_1 + \boldsymbol{C}_1),\ P_2(\hat{\boldsymbol{m}}_2 + \boldsymbol{C}_2))^\top \tilde{\boldsymbol{n}}. \tag{7}$$

Symmetry planes are recovered for all pairs of detected corners; however, many solutions can be discarded immediately. The equations above are over-determined, and while every solution is guaranteed to be orthogonal to the floor, the resulting reprojection error may be large for 3D points recovered using Equations (2) and (3). Thus, we only keep those solutions that have reprojection errors for all corner features less than a threshold ϵ, as specified in Equation (8):

$$\Pi = \{\boldsymbol{\pi} : (\boldsymbol{U}_i, \boldsymbol{V}_i) = \xi(\boldsymbol{\pi}, P_i, \boldsymbol{u}_i^*, \boldsymbol{v}_i^*), \quad d(\boldsymbol{u}_i^*, P_i \boldsymbol{U}_i),\, d(\boldsymbol{v}_i^*, P_i \boldsymbol{V}_i) < \epsilon, \quad i \in \{1, 2\}\}, \tag{8}$$

where $\boldsymbol{u}_i^*, \boldsymbol{v}_i^*$ is every pair of corner features registered across both images by the disparity map calculated in Section 3.4, $\boldsymbol{\pi}$ is the resulting symmetry plane found by solving Equations (6) and (7), and $(\boldsymbol{U}_i, \boldsymbol{V}_i) = \xi(\boldsymbol{\pi}, P_i, \boldsymbol{u}_i^*, \boldsymbol{v}_i^*)$ gives the two solutions for Equations (2) and (3) in camera image i. This operation has worst time complexity $O(n^2)$ in the number of corner features detected, since we are testing all unique sets of two corner features chosen from the set of all detected corner features.

3.6. Two Orthogonal Symmetry Planes

To find orthogonal symmetry planes, we consider all pairs of hypotheses in Π specified in Equation (8). The worst case time complexity is $O(n^4)$, since, in theory, we could be testing all unique sets of four corner features from the set of all detected corner features. In practice, however, most pairs of corners are already discarded by using Equation (8) when generating the set Π. We specify the reprojection error of a pair of planes as the maximum reprojection error of one of the points in one of the cameras that was involved in generating one of the symmetry plane hypotheses. This is specified by Equation (9), where $\boldsymbol{u}_{ij}^*$ and $\boldsymbol{v}_{ij}^*$ are the pair of image points that originally generated symmetry plane hypothesis j in image i, and $\boldsymbol{U}_{ij}$ and $\boldsymbol{V}_{ij}$ are calculated using symmetry plane hypothesis j in image i:

$$\texttt{error}(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2) = max\left(d(\boldsymbol{u}_{ij}^*, P_i\boldsymbol{U}_{ij}), d(\boldsymbol{v}_{ij}^*, P_i\boldsymbol{V}_{ij})\right) \quad i, j \in \{1, 2\}. \tag{9}$$

Every pair of symmetry plane hypotheses in Π is considered, and Nelder–Mead [53] is applied to minimize the error specified in Equation (9) after forcing the symmetry planes to be orthogonal. The set $\Pi^{'}$ is then the set of all the hypothesis pairs whose optimized error is less than some threshold ϵ.

Note that using two planes of symmetry allows for reconstructing the backs of objects, which are normally occluded in the camera images. Because all the points in object S obey Equation (1) for every symmetry plane, then every point $X \in S$ can be reflected in one or both symmetry planes, relating it to three other points in S. This means that points on the visible front of the object are related to occluded points on the back of the object, and thus that back of the object can be recovered. This is not possible when objects have only a single plane of symmetry; however, even in this case, the backs of objects can be recovered using additional assumptions, such as planarity [12].

This observation requires that we modify Equation (4) so that we do not attempt to try to find image evidence for points that we expect to be occluded. To do this, we take each set of four points that are related through the two symmetry planes, and we remove the point that is furthest from the camera center. We then combine these sets of three points to make $S^{'} \subset S$, and apply Equation (4) to the points in $S^{'}$.

3.7. Recovering Objects with Short Curves

A 3D shape, defined by Definition 1, can be recovered for every symmetry-plane-pair in $\Pi^{'}$, by using Equations (2) and (3) on all pairs of edge points in a single image. According to Equation (4), the recovered 3D points must project to an edge point within a set threshold, ϵ, for all camera images. In a method similar to RANSAC [51], the final recovered 3D shape is simply that with the largest number of 3D points that are consistent with Equation (4).

In theory, this has worst case time complexity $O(n^2)$ in the number of edge points, since we could be considering all unique sets of two edge points from the set of all detected edge points. However, as outlined in Section 3.2, the images of symmetric points must be co-linear with the vanishing point derived from the relevant symmetry plane. Therefore, the proposed algorithm sorts all edge points by the angle subtended with the vanishing point, and then a local search is performed about each point to find co-linear points that solve for 3D points that obey the threshold specified in Equation (4). The expected time complexity is $O(n\,log\,n + nm)$, where $O(n\,log\,n)$ is the time complexity for sorting edge points, and nm is the expected number of comparisons made across all m camera images in the local search for co-linear points.

In practice, incidental arrangements of edge points lead to some false positive 3D points that are not filtered out by applying Equation (4). Almost all of these incidental points can be removed by considering short continuous runs of edge points, which we call contours. Instead of finding "best" or "meaningful" contours—an expensive operation that may introduce instabilities—we split the runs of edge points at approximately equally spaced intervals. We then select several points on each contour, and solve for the 3D points as before; however, in addition to filtering the results by Equation (4), we also add the additional constraint that the corresponding edge points must lie within a set distance along some contour. Not only does this procedure reduce the number of false positive points in the recovered shape, it also dramatically improves the speed of the computation by reducing the number of edge points being considered with each recovery.

3.8. Overview of Algorithm

Figure 6 gives a flow chart, and dependency graph, for the algorithm. Many steps can be executed in parallel, and individual steps can also make good use of parallelism. In particular, estimating the floor (Section 3.4), finding symmetry hypotheses (Sections 3.5 and 3.6), and calculating solutions (Section 3.7) can each have their time complexities divided by the available parallelism.

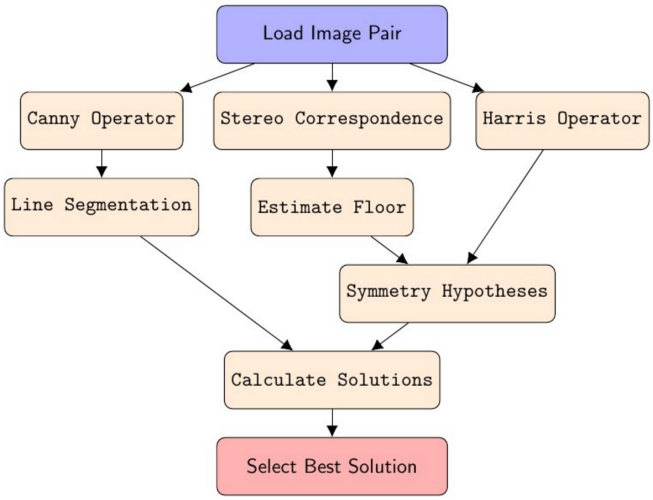


Figure 6. Flow chart of proposed 3D recovery algorithm.

Table 1 lists the algorithm parameters, which are chosen with reference to the properties of the relevant equations. Increasing the reprojection thresholds results in more false positives in the recovered 3D shape. The algorithm is robust to changes in the other parameters, including contour length.

Table 1. Algorithm parameters. For details on the *random sample consensus* algorithm (RANSAC), see: [51].

Parameter	Reference	Value
RANSAC iterations for floor estimation	Section 3.4	500 iterations
RANSAC threshold for floor estimation	Section 3.4	0.1 m
Harris block size	Section 3.5	3
Harris k	Section 3.5	0.01
Object point reprojection threshold	Equation (4)	$\epsilon = 1.5$ pixels
Symmetry plane reprojection threshold	Equation (8) and Section 3.6	$\epsilon = 1.5$ pixels
Contour length	Section 3.7	15 pixels

4. Results

We have presented an algorithm for performing 3D object recovery for mirror symmetric objects from pairs of images. We ran two simulations to test aspects of this algorithm. In simulation 1, Section 4.1, we compared Triangulation [22]—the standard 3D recovery algorithm for two-view geometry—to the formula derived in Section 3.2, which performs 3D recovery using mirror symmetry. In simulation 2, Section 4.2, we generated synthetic images of random mirror symmetric shapes with one versus two planes of symmetry, and with random noise features, in order to examine how well the algorithm solves the correspondence problem. In Section 4.3, we present results for the algorithm on a corpus of real image pairs. Since the algorithm is a repurposed refinement of previous work on figure/ground organization [44], we compare the new algorithm to this previous work.

4.1. Simulation 1

The algorithm presented in this paper addresses, in part, two different problems: the correspondence problem and the 3D recovery problem. The correspondence problem involves selecting sets of corresponding image points that share some geometric property. For example, consider Figure 4. The points u_1 and u_2 are corresponding points for the binocular correspondence problem, and u_1 and v_1 are corresponding points for the symmetry correspondence problem. Furthermore, all four points, u_1, v_1, u_2, and v_2, are corresponding points for both the symmetry and binocular correspondence problems together. Solving a given correspondence problem involves identifying a set of image points.

The 3D recovery problem involves calculating points in 3D from sets of corresponding points and is usually performed with a mathematical formula. By contrast, solving a correspondence problem usually involves some type of search. Triangulation [22] is the standard method for performing 3D recovery from a pair of binocular corresponding points. In Section 3.2, we developed an equation for 3D recovery from a pair of corresponding points for the symmetry correspondence problem. The reader may guess that this 3D recovery should be more accurate, on average, than any 3D reconstruction performed on corresponding points for the binocular correspondence problem, including Triangulation, because the latter is inherently sensitive to noise [22].

The supplementary materials include animations of a figure recovered from real image pairs (described fully in Section 4.3), using either Triangulation, or Equations (2) and (3) from Section 3.2. In all cases, the correspondence problem was solved using the symmetry based algorithm presented in this paper; only the last step—3D recovery—is different. The 3D figure recovered using Triangulation features peculiar "planar" clustering, which is caused by pixelation in the camera images. The recovered 3D points jump from plane to plane as the difference between the corresponding points changes by one pixel. This is a well known problem and can only be ameliorated by subpixel correction before Triangulation is performed. The animation files show qualitatively that recovery based on symmetry is unquestionably better than Triangulation. The pixel-related artifacts of binocular recovery are almost never symmetrical with respect to the common symmetry plane of the object, and so, these artifacts are corrected by adding the symmetry constraint. However, as we show below, Triangulation is more sensitive to noise even when controlling for pixelation.

In this simulation, we compare 3D recovery using Triangulation, versus using Equations (2) and (3) for pairs of points, as opposed to whole objects. We generated one random pair of 3D points at a time. The points were uniformly distributed over a $4 \times 4 \times 4$ m^3 box centered 3 m directly in front of the simulated camera. A single pair of points is obviously mirror symmetrical with respect to the plane that bisects them. This plane was computed and used for the recovery based on Equations (2) and (3). These two 3D points were projected, as continuous variables, to a pair of simulated images using a calibrated stereo camera with a 12-cm baseline, a 66-deg horizontal field of view. After the two image points (in $\mathbb{R}^2$) were calculated, a variable amount of Gaussian white noise was added. The standard deviation of the Gaussian white noise was specified in pixels, which is relative to the assumed 800 $\times$ 600 image format. The simulated image points were never rounded

(to the nearest pixel) so that the simulation is a valid approximation for other image resolutions. Thus, the noise in our experiment simulates other natural processes, other than pixelation, that affect image formation. Either Triangulation or Symmetry (Equations (2) and (3)) were used to perform 3D recovery from a pair of image points. Figure 7 shows the error in recovery (in meters) as a function of the standard deviation of the added Gaussian white noise (in pixels). Each data point in the graph is the average error from 1 million replications. In each replication, one pair of random 3D points was generated and recovered. Note that the use of the true symmetry plane in this simulation when a single pair of points was recovered represents the fact that the estimated symmetry plane of an object consisting of hundreds of points is very stable precisely because many pairs of points are used to estimate that plane. This is the unique advantage of incorporating a spatially global constraint.

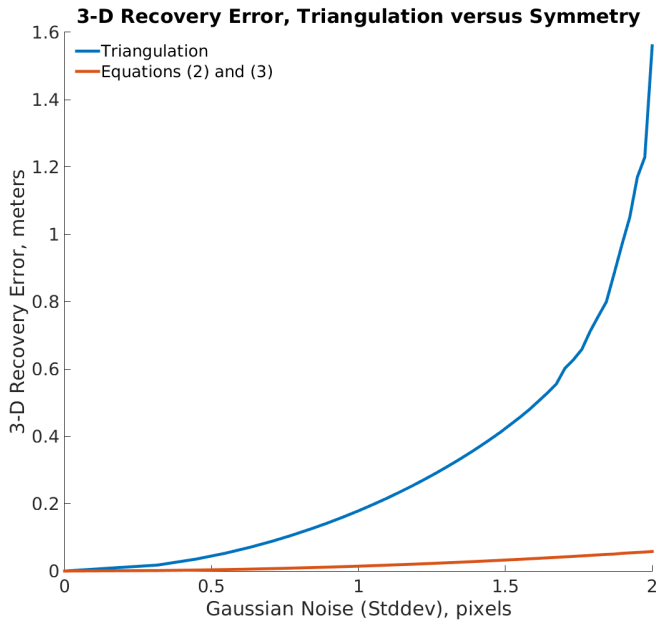


Figure 7. Top curve: error in 3D recovery using Triangulation [22] as a function of image noise. Bottom curve: error in 3D recovery using symmetry (Equations (2) and (3)) as a function of image noise.

Figure 7 clearly shows that 3D recovery using Triangulation is more sensitive to noise, which could come from image pixelation, or other aspects of camera system, such as uncorrected lens distortion. Recovery using Equations (2) and (3) was more than an order of magnitude more accurate at all levels of noise, except zero noise, when recovery was identical for both methods.

4.2. Simulation 2

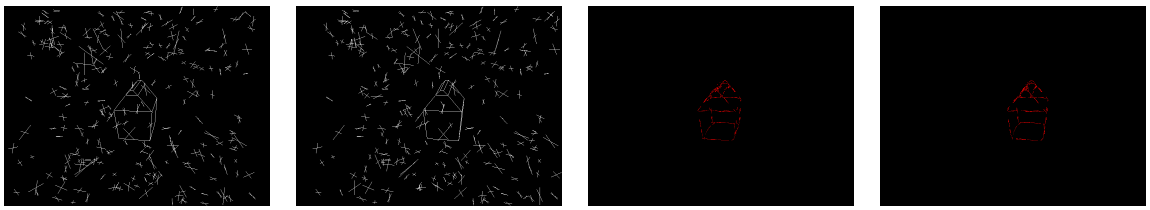
In this simulation, we used random synthetic figures to assess how well the present algorithm solves the correspondence problem for shapes with one or two planes of symmetry, and in the presence of noise. We followed [17] to generate random synthetic shapes with a single plane of symmetry. These shapes are bilaterally symmetric, have planar surfaces, a flat base, and are made out of three rectangular boxes joined to each other. The same procedure was used for shapes with two planes of symmetry; however, the shape was flatted across the top, with the height chosen to keep the total volume unchanged. The shapes averaged $35 \times 47 \times 34$ cm^3. Please refer to [17] for precise details on how these shapes are generated.

To generate random noise, we sampled 3D points uniformly in a $2 \times 2 \times 2$ m^3 box centered at a point 2.5 m in front of the camera center. Each 3D point became the point of intersection for two 3D lines, each 10-cm long, and sitting on a plane with a random orientation. The orientation of the

plane was generated by sampling from a uniform distribution for the orientation vector's inclination and azimuth.

The random shape and noise features were then projected to a pair of 800 × 600 simulated images using a calibrated stereo camera with a 12-cm baseline, and a 66-deg horizontal field of view. A random shape with two planes of symmetry and 250 noise features is shown in Figure 8a,b. The crossed noise features were specifically chosen to generate lots of corner detections, since these are used to find symmetry planes, as described in Section 3.5. Furthermore, the lines were made long enough so that most of the projected lines would not be easily filtered out when finding correspondences, as described in Section 3.7.

The proposed algorithm then attempted 3D recovery on the composited image pairs. The image pairs contained no texture information with which to estimate the floor plane, so the orientation of the floor—a plane orthogonal to the shapes one or two symmetry planes—was given to the algorithm. Correct correspondence (true positives), incorrect correspondences (false positives), and missed correspondences (false negatives) were then counted, and used to generate a *precision* and *recall* score for each image pair. Figure 8c,d shows the recovered 3D shape as seen from the two camera views.



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

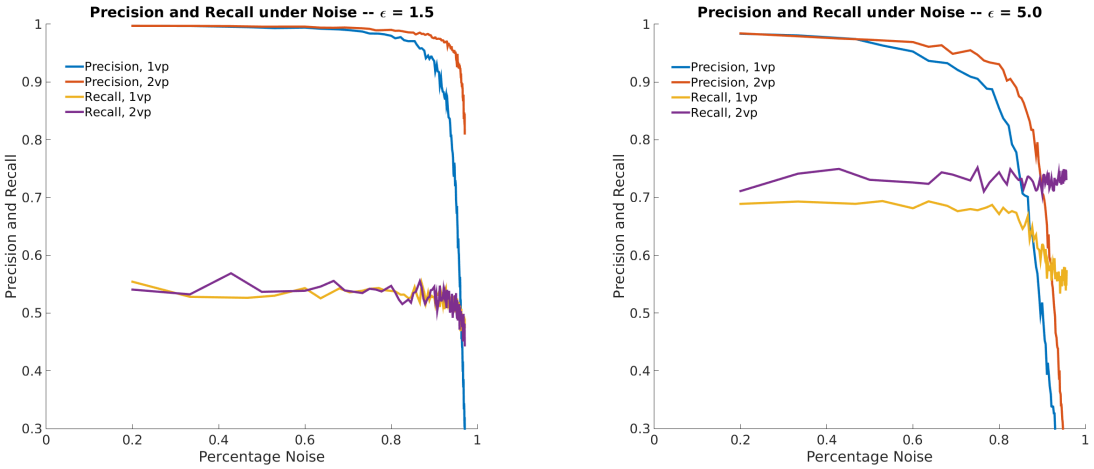
Figure 8. Input shape with two planes of symmetry, and random noise. (**a**,**b**) show the right and left camera image, respectively. The shape was recovered by the proposed algorithm, and (**c**,**d**) show the recovered 3D shape as seen by the left and right camera, respectively. An animation of the recovered shape is included in the supplementary materials.

Figure 9a shows precision and recall scores as a function of the percentage of noise features in the images, for shapes with one or two planes of symmetry. Ten random scenes were generated for each level of noise, and in each condition. (i.e., one or two planes of symmetry.) The percentage noise is calculated as 1 minus the number of visible corners in the synthetic shape (a maximum of 16) divided by the number noise features (each cross is a single noise feature). This data was generated using $\epsilon = 1.5$, the default "object point reprojection threshold" listed in Table 1, which is the ϵ used in Equation (4). We expected that as this threshold was loosened, recall would rise, and precision would fall. This is indeed the case, as shown in Figure 9b, which gives the same results but for $\epsilon = 5$.

We conclude that the method for solving the correspondence problem is robust to incidental noise. Furthermore, adding a second plane of symmetry does improve the robustness of the algorithm: precision is better, and recall does not suffer, or is better as in Figure 9b.

4.3. Experiment

To assess the algorithm on real images, we needed a corpus of stereo image pairs of objects, with two orthogonal planes of symmetry, standing on a clearly visible floor. The authors of [46] kindly made available the *Children's Furniture Corpus*, which is now available in the supplementary materials. This corpus consists of eighty-nine 800 × 600 grayscale image pairs of 11 exemplars, nine of which have two orthogonal planes of symmetry, and two with single planes of symmetry. Images were captured under typical indoor lighting conditions using a Point Grey Bumblebee2® stereo camera (Point Grey, Richmond, BC, Canada), with a 12-cm baseline, and a 66-deg horizontal field of view. Each exemplar averaged 1.87 m from the camera, sitting in the middle of the frame on flat blue carpet, and was captured from roughly equally spaced orientations.



(**a**) (**b**)

Figure 9. Precision and recall scores as a function of the percentage of noise features in the input images, for randomly generated shapes with one or two symmetry planes (vanishing points). (**a**) shows precision and recall for the algorithm's default parameters, listed in Table 1. We see that recall is similar for both types of shapes; however, precision is better for shapes with two symmetry planes. Furthermore, precision is more robust to noise for shapes with two symmetry planes; (**b**) shows precision and recall when $\epsilon = 5$. This is the ϵ in Equation (4). Loosening this threshold increases recall at the cost of lowered precision. Both precision and recall are better for two symmetry planes, indicating that the extra symmetry plane improves the algorithms performance due to the additional constraints that it provides.

The physical exemplars (pieces of furniture) were measured by hand with a tape measure, and Autodesk® Maya® (Maya 2016, Autodesk Inc., San Rafael, CA, USA) was used to create 3D models. Custom made software was then used to annotate each corpus image. Annotations consisted of extrinsic and intrinsic camera parameters, and the hand made Autodesk® Maya® model file (a triangle mesh) along with rotation, translation, and scale parameters. This is sufficient to place the mesh in 3D space, and project it to each camera image.

To measure the accuracy of the 3D recovery of shape S, we summed the average minimum distances between each point $X \in S$ and some point on triangle Δ in the hand-made 3D ground truth mesh, Ξ, with the average of the minimum distance between each vertex $V \in \Xi$ and some point $X \in S$. This is specified by Equation (10):

$$\texttt{error}(S,\Xi) = \left(\frac{1}{|S|}\sum_{X\in S} \underset{\Delta\in\Xi}{arg\,min}\, d(X,\Delta)\right) + \left(\frac{1}{|\Xi|}\sum_{V\in\Xi} \underset{X\in S}{arg\,min}\, d(V,S)\right). \quad (10)$$

Recovery was then performed on four 16 core Intel® Xeon® E7-8800v3 processors (Intel, Santa Clara, CA, USA), running Ubuntu® 14.04 (Canonical, London, UK). The algorithm produced output for all but two pairs of input images. Failure modes are discussed in Section 4.3.2. The average runtime was 2.41 s, and the average error, according to Equation (10), was 2.79 cm, or just over 1 inch. This error is the approximate thickness of the furniture surfaces, and, in part, contains errors in the hand made models. The results are summarized in Table 2, which shows the average speed and error for each exemplar. Table 3 shows sample 3D reconstructions.

4.3.1. Baseline for Comparison

This study is unique in that it reports the error in 3D object recovery over a corpus of images, when using projective geometry, and a mirror symmetry prior. To the authors knowledge, there is

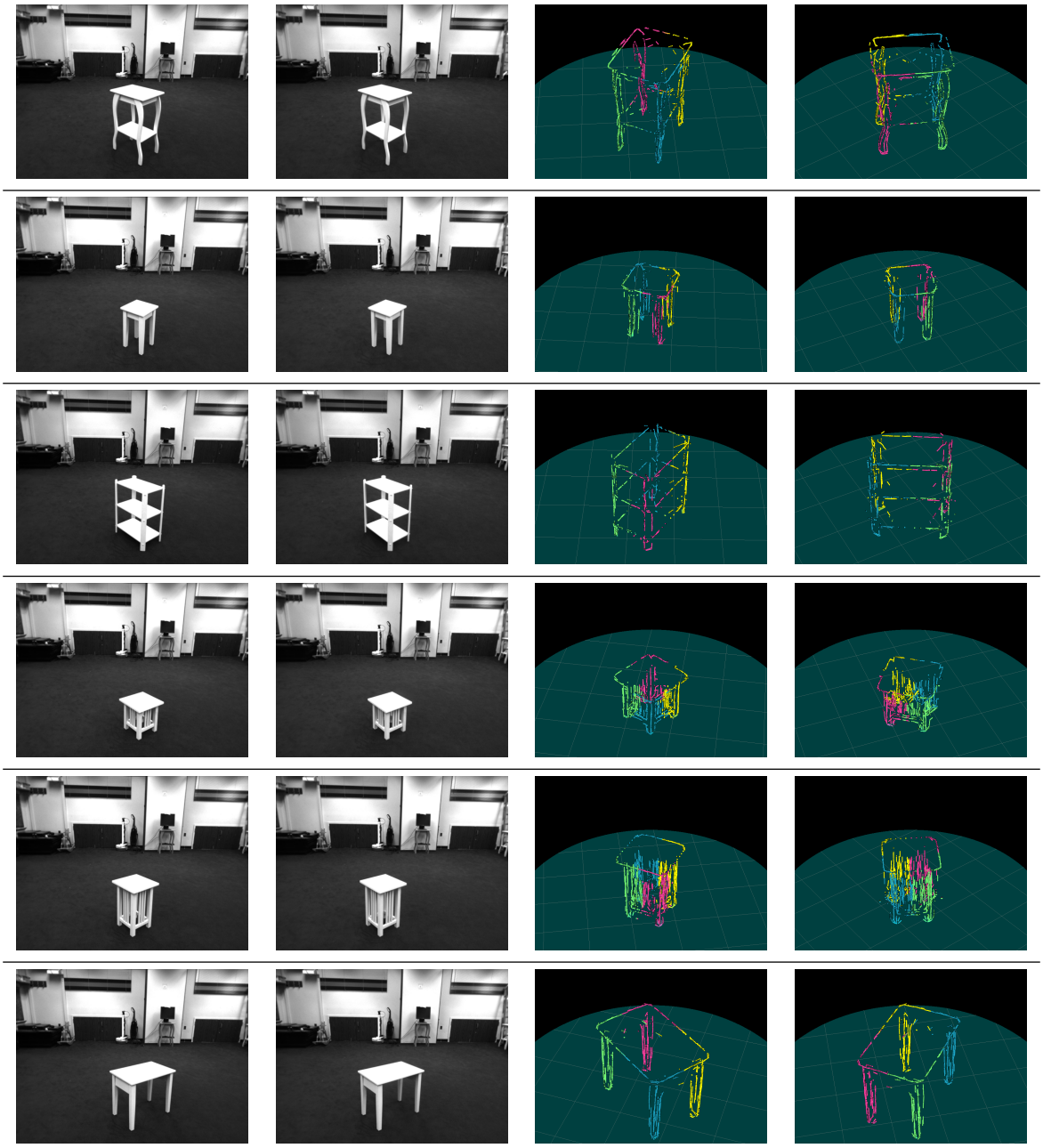
no comparable study, with either qualitative results being presented [25,30,34,37,41,54], errors only being reported on a few keypoints for only one or two images [45,55,56], or the approach is otherwise incomparable [29,31–33,35,42,43], or too restricted to be able to produce 3D recoveries from raster images such as those found in the Children's Furniture Corpus [3,21,26,38–40,57]. The authors suggest that more quantitative studies on corpora of camera images are needed, and have made the *Children's Furniture Corpus* available in the supplementary materials of this paper.

Michaux et al. [44] utilizes a similar method for disambiguating the symmetry and binocular correspondence problems, described in Section 3.3, and also uses a floor prior. Although this work aims to localize multiple objects in a scene at once, it can be adapted to perform 3D object recovery. The algorithm in this paper is an elaboration of [44], and thus we chose [44] as a baseline for comparison. The nature of those elaborations, and their effect on the speed and accuracy of 3D recovery, is discussed in Section 4.3.3.

Table 2. Experimental results were collected for the *Children's Furniture Corpus*, and a modified version of Michaux et al. [44] was used as a baseline for comparison. The second column denotes the number of image pairs for the given exemplar. Errors were calculated using Equation (10) and are reported in centimeters, along with a 95% confidence interval. Runtimes are reported in seconds, also with a 95% confidence interval. The algorithm failed to generate results for two image pairs of the Rubbish Bin, as discussed in Section 4.3.2.

Exemplar	#	Proposed Method	Michaux et al. [44]
Curved Stand	8	1.90 ± 1.69 cm 1.97 ± 0.41 s	3.42 ± 1.49 cm 8.96 ± 4.28 s
Short Stand	7	2.16 ± 0.27 cm 1.56 ± 1.47 s	4.94 ± 4.44 cm 3.52 ± 1.65 s
Bookshelf	8	2.13 ± 2.59 cm 2.30 ± 0.50 s	2.81 ± 2.86 cm 9.75 ± 3.25 s
Short Dense Stand	5	3.04 ± 1.17 cm 2.89 ± 3.90 s	6.21 ± 2.67 cm 4.14 ± 4.08 s
Mid Dense Stand	6	3.38 ± 2.48 cm 3.37 ± 4.14 s	4.74 ± 3.06 cm 8.25 ± 6.71 s
Tall Dense Stand	7	3.46 ± 5.76 cm 5.69 ± 7.34 s	5.20 ± 1.64 cm 18.17 ± 17.46 s
Short Table	8	2.96 ± 3.19 cm 1.45 ± 1.09 s	5.39 ± 5.36 cm 4.03 ± 1.53 s
Long Table	7	2.61 ± 4.08 cm 1.51 ± 1.25 s	5.52 ± 7.61 cm 5.68 ± 2.40 s
Rubbish Bin	6	2.54 ± 1.64 cm 0.89 ± 1.85 s	9.67 ± 8.43 cm 6.40 ± 5.56 s
Total/Average	63	2.66 ± 3.50 cm 2.41 ± 4.00 s	5.13 ± 5.65 cm 7.86 ± 10.71 s

Table 3. 3D reconstructions. The left two columns show the pair of the input images. The right two columns show 3D reconstructions, where the view has been rotated. The displayed objects are, from top to bottom: Curved Stand, Short Stand, Bookshelf, Short Dense Stand, Mid Dense Stand, and Long Table. Animation files are available in the supplementary materials. The recovered objects have two symmetry planes, and are thus naturally divided into four quadrants.

Input (Left Image)	Input (Right Image)	120-deg Rotation	240-deg Rotation
			

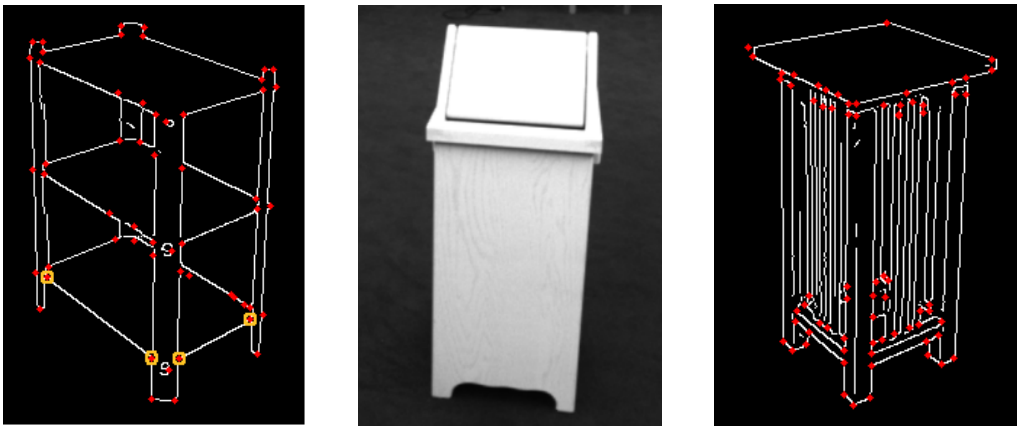
4.3.2. Discussion

The algorithm produced 3D reconstructions for all input image pairs, except for two image pairs of the Rubbish Bin. 3D reconstructions were, in general, excellent, with most recovered 3D points close to edges or surfaces on the annotated ground truth 3D meshes. The backs of objects were recovered, even though they were occluded, because the occluded points were related to points on the front of the object by reflections through one or both symmetry planes, as described in Section 3.6.

This contributes to the structure of the objects being clearly visible and identifiable to human viewers, who, presumably also recover the backs of the objects. Despite these successes, several failure modes were identified, as were opportunities for improvement.

Reducing the object point reprojection threshold, ϵ (see Table 1), resulted in significantly cleaner looking objects; however, recoveries were also sparser without affecting errors calculated by Equation (10). This is caused by the algorithm finding all the correspondences for edge points that are less than ϵ. Most edge points participate in multiple correspondence pairs, including erroneous correspondences that are "close enough", according to Equation (4). Solving this problem is not as simple as limiting edge points to single correspondences, since pixelated images require multiple correspondences for single edge points. This happens when the pair of mirror-symmetric curves are not equidistant to the camera, and thus image to 2D contours of different lengths. The pixels on the shorter contour must then participate in multiple correspondences in order to recover all of the points on the longer contour. This particular problem deserves consideration, and is important for generating more accurate and aesthetically pleasing 3D recoveries; however, it does not affect the basic structure of the recovery.

3D reconstruction depends on corner detection producing the required two pairs of corners that characterize the symmetry planes of the object, as shown in Figure 10a. Furthermore, these corners must appear in both images, and be registered with each other successfully. If no such set of four corners is found, then the algorithm fails. This happened for two images pairs of the Rubbish bin, one shown in Figure 10b. Corner detection worked well for the other corpus images; however, it does represent a potential algorithmic instability.

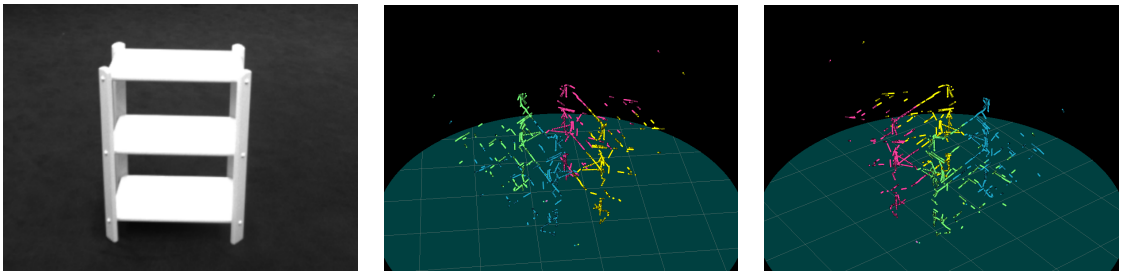


(**a**) (**b**) (**c**)

Figure 10. The importance of corner detection. (**a**) 3D recovery fails unless at least two pairs of corner features appear that reveal the required symmetry planes. In this edge map of an image of the Bookshelf, detected corners are marked in red, and a sufficient set of four corners is marked by yellow circles; (**b**) degenerate view of the Rubbish Bin. Recovery fails because the back of the object is not visible, and corner detection cannot be used to determine one of the planes of symmetry; and (**c**) runtime suffers when corner detection produces many corner pairs that generate symmetry hypotheses orthogonal to the floor, as is the case in this image of the Tall Dense Stand. These hypotheses are not filtered out by Equation (8), and runtime tends towards $O(n^4)$ in the number of detected corners. Runtime for this image was 11.48 s.

As described in Section 3.6, the worst case runtime is $O(n^4)$ in the number of detected corners. (The worst runtime was 11.48 s, for an image pair of the Tall Dense Stand shown in Figure 10c.) The algorithm can be modified to use any method for detecting symmetry planes, and there are many fast methods for finding vanishing points that can be used to calculate directions of symmetry for prospective symmetry planes.

Some 3D recoveries were poor even when symmetry planes were detected correctly. This was due to instabilities caused by degenerate cases inherent in the geometrical foundations of the algorithm. Consider the image of the Bookshelf shown in Figure 11a. The image itself is almost mirror symmetric in 2D, indicating that the symmetry plane contains (or runs close to) the center of perspective projection. As discussed in Section 3.2, this degenerate case prevents recovering 3D depth information from symmetry. As such, 3D recovery must rely on two-view geometry, which, in general, is noisy due to pixelation in the image [22]. However, two-view geometry also has a degenerate case. The Bumblebee2® camera (Point Grey, Richmond, BC, Canada) produces pairs of rectified images [22], which confounds solving binocular correspondence for points on horizontal lines. This means that Equation (4) is not particularly useful in disambiguating symmetry correspondences. Nonetheless, the algorithm produced recognizable output even in this pathological case, as shown in Figure 11b,c.



(**a**) (**b**) (**c**)

Figure 11. Degenerate views. (**a**) 3D recovery is poor even though symmetry planes can be estimated via corner detection. The image of the object is almost symmetrical in 2D, which means that the symmetry plane contains (or runs close to) the camera center. This makes 3D recovery numerically unstable, as described in Section 3.2. Furthermore, many of the lines are almost horizontal, a degenerate case when solving binocular correspondence for two-view geometry with rectified cameras [22], which is the case here. (**b**,**c**) show 3D recoveries with the view rotated by 120-deg and 240-deg. An animation file is available in the supplementary materials.

4.3.3. Comparison to Baseline

As shown in Table 2, the proposed method is both faster and more accurate than the baseline for comparison, [44]. The reasons for this lie in the differences between the two algorithms. We believe that the main reason for the more consistent and accurate results of the proposed algorithm lies in how vanishing points were detected. Firstly, Equation (9) uses the L^∞ norm, while [44] utilizes an equation similar to Equation (8), but based on the L^2 norm. Our analysis found that the L^∞ was much better suited to filtering out erroneous correspondences and should always be preferred to the L^2 norm.

Secondly, the baseline utilizes Equation (4) over the entire edge map of both input images, while the proposed algorithm also utilizes short pieces of curves. Using short pieces of curves proved to be an important innovation, because, as noted in Section 3.7, many incidental arrangements of edge points will still satisfy Equation (4). By utilizing short curves, we ensure that Equation (4) applies to contiguous sets of points—a powerful restriction. Our analysis showed that it was sufficient to work on the end points of the curve segments, and simply ensure that they are connected. This vastly improved the stability and performance of the algorithm.

Thirdly, the baseline identifies symmetry planes using RANSAC, while the proposed algorithm, searched across sets of four automatically detected corner features, and uses a closed form expression to make initial estimates of symmetry planes (Equations (5)–(7)). This means that the proposed algorithm depends on corner detection, as described in Section 4.3.2; however, we found that, in general, the symmetry plane detection was much better than the baseline.

Finally, the most obvious difference is that the baseline uses a shape definition based on just a single plane of symmetry, as opposed to two planes of symmetry. The choice of two planes of

symmetry provides additional constraints on recovered points, as specified by Equation (9), and also provides a simple mechanism for recovering occluded surfaces. This results in more accurate estimates of the vanishing points (more points are involved when optimizing Equation (9)), as well as more complete figures. However, note that the backs of most exemplars are not completely occluded in the corpus images, and when [44] performed well, it was almost as accurate as the proposed algorithm, even though it utilized just one plane of symmetry. The exception was the Rubbish Bin, an object whose rear facing contours are always occluded. In this case, the baseline's performance compared least favorably to the proposed algorithm. In brief, the refined method presented in this paper is faster and more robust than the baseline.

5. Conclusions

The results suggest that mirror symmetry—a biologically motivated prior—is useful in performing 3D recovery from pairs of stereo images. However, the algorithm was designed and tested on a specific class of symmetrical objects, namely, those with two orthogonal planes of symmetry. Although this presented potential performance issues, discussed in Section 3.6, it also allowed for straightforward recovery of the backs of objects. The formulation relied on the 3D mirror symmetry of the objects being evident in edge maps of input image pairs. Practically speaking, this limits the application of the algorithm to objects with clearly visible contours—a typical restriction for solving the symmetry correspondence problem.

The failure modes discussed in Section 4.3.2 suggest that the algorithm could benefit from numerous small improvements, such as better or faster vanishing point detection. Furthermore, the algorithm has good precision (it rarely recovers false points); however, recall is limited (it tends to miss parts of objects). It is reasonable, therefore, to use the algorithms output as a starting point for completing the entire figure, perhaps using a smoothness constraint to do contour completion.

Larger improvements are also possible. First, there is no reason why the algorithm cannot be reformulated for an object definition for shapes with single planes of symmetry, while still recovering the backs of objects. This involves adding extra constraints to the object; however, these constraints may be useful in other ways. For example, as pointed out in [12], combining mirror symmetry with planarity is sufficient to recover occluded surfaces under common conditions. Many objects have planar surfaces, and psychophysical evidence suggests that humans use a planarity prior when recovering the shape of mirror symmetric curves [58].

Relaxing the use of the floor prior raises interesting questions as well. The algorithm can be modified to do this by simply letting Π (from Equation (8)) be generated from all pairs of corner features, and Π' from all pairs of symmetry hypotheses in Π. Although the runtime would suffer, the algorithm would still recover 3D shapes, including objects floating through the air. One caveat is that part of the accuracy of the symmetry hypotheses in Π (and thus Π') comes from the floor prior. This is because direct 3D recovery from stereo image pairs tends to be noisy, and the direction of symmetry would have to be calculated from those noisy recoveries. One way around this is to use pairs of hypotheses (from Π') as the initialization point for some optimization procedure that modifies the symmetry plane parameters in order to maximize the number of recovered 3D points that obey Equation (4). This could potentially mean orders of magnitude more evaluations of Equation (4). These computational problems would be alleviated by finding better methods for estimating the vanishing points of objects, or perhaps other innovations with how Equation (4) is handled.

Another interesting problem is to consider solving the symmetry correspondence problem for single uncalibrated images. Solving this problem will have many immediate applications because the Internet is full of uncalibrated images of symmetric objects. The focal length of an uncalibrated image can be estimated from two orthogonal directions, if we assume that the principal point is the center of the image [59]—usually close enough if the image has not been cropped. Therefore, in theory, camera calibration could be performed from images of symmetrical objects. This may be related to how the human vision system calibrates itself [60].

The authors believe that advances in computer vision lie in a deep understanding of the priors used by the human visual system. There is a straightforward logical argument for why this is so: if we assume a computational model of human cognition, then the powerful capabilities of the human vision system are founded in the innate priors that it uses. We have used two-view geometry as a crutch to explore possible ways for solving symmetry correspondence and pointed out several small and large problems that remain to be solved. A robust and general purpose solution for the symmetry correspondence problem would significantly advance the state of the art and also have immediate applications.

Supplementary Materials: The following are available online at www.mdpi.com/2073-8994/9/5/64/s1: The *Children's Furniture Corpus*. Animation files for reconstructions shown in Table 3, and Figures 8 and 11 can be found at https://osf.io/qdzz6/. This url also contains an animation of the curved stand with 3D recovery performed using Triangulation, as described in Section 4.1. An interactive 3D version of Figure 2 is available at http://www.pageofswords.net/symmetry-equation/.

Acknowledgments: This research was supported by the National Eye Institute of the National Institutes of Health under award number 1R01EY024666-01. The content of this paper is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.

Author Contributions: Aaron Michaux designed and implemented the proposed algorithm, and co-wrote the paper. Vikrant Kumar handcrafted the 3D models, annotated the data set, and assisted in writing. Zygmunt Pizlo, Vijai Jayadevan and Edward Delp participated in theoretical discussions that laid the basis for the paper's algorithm, reviewed the math, and co-wrote the paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Horn, B. Understanding image intensities. *Artif. Intell.* **1977**, *8*, 201–231.
2. Witkin, A. Recovering surface shape and orientation from texture. *Artif. Intell.* **1981**, *17*, 17–45.
3. Kanade, T. Recovery of the three-dimensional shape of an object from a single view. *Artif. Intell.* **1981**, *17*, 409–460.
4. Marr, D. *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*; Henry Holt and Co., Inc.: New York, NY, USA, 1982.
5. Hinton, G. Learning multiple layers of representation. *Trends Cogn. Sci.* **2007**, *11*, 428–434.
6. Dickinson, S. Challenge of image abstraction. In *Object Categorization: Computer and Human Vision Perspectives*; Cambridge University Press: New York, NY, USA, 2009.
7. Chalmers, A.; Reinhard, E.; Davis, T. *Practical Parallel Rendering*; CRC Press: Boca Raton, FL, USA, 2002.
8. Tikhonov, A.N.; Arsenin, V.Y. *Solutions of Ill-Posed Problems*; Winston: Washington, DC, USA, 1977.
9. Poggio, T.; Torre, V.; Koch, C. Computational vision and regularization theory. *Nature* **1985**, *317*, 314–319.
10. Pinker, S. *How the Mind Works*; W. W. Norton & Company: New York, NY, USA, 1997.
11. Tsotsos, J.K. *A Computational Perspective on Visual Attention*; MIT Press: Cambridge, MA, USA, 2011.
12. Pizlo, Z.; Li, Y.; Sawada, T.; Steinman, R. *Making a Machine That Sees Like Us*; Oxford University Press: Oxford, UK, 2014.
13. Shepard, S.; Metzler, D. Mental rotation: Effects of dimensionality of objects and type of task. *J. Exp. Psychol. Hum. Percept. Perform.* **1988**, *14*, 3–11.
14. Vetter, T.; Poggio, T.; Bülthoff, H. The importance of symmetry and virtual views in three-dimensional object recognition. *Curr. Biol.* **1994**, *4*, 18–23.
15. Sawada, T.; Li, Y.; Pizlo, Z. Shape Perception. In *The Oxford Handbook of Computational and Mathematical Psychology*; Oxford University Press: Oxford, UK, 2015; p. 255.
16. Li, Y.; Sawada, T.; Shi, Y.; Steinman, R.; Pizlo, Z. Symmetry Is the sine qua non of Shape. In *Shape Perception in Human and Computer Vision*; Springer: London, UK, 2013; pp. 21–40.
17. Li, Y.; Sawada, T.; Shi, Y.; Kwon, T.; Pizlo, Z. A Bayesian model of binocular perception of 3D mirror symmetrical polyhedra. *J. Vis.* **2011**, *11*, 11.
18. Palmer, E.; Michaux, A.; Pizlo, Z. Using virtual environments to evaluate assumptions of the human visual system. In Proceedings of the 2016 IEEE Virtual Reality (VR), Greenville, SC, USA, 19–23 March 2016; pp. 257–258.

19. Marr, D.; Nishihara, H. Representation and Recognition of the Spatial Organization of Three-Dimensional Shapes. *Proc. R. Soc. Lond. B Biol. Sci.* **1978**, *200*, 269–294.
20. Binford, T. Visual perception by computer. In Proceedings of the IEEE Conference on Systems and Control, Miami, FL, USA, 15–17 December 1971; Volume 261, p. 262.
21. Gordon, G. Shape from symmetry. In Proceedings of the Intelligent Robots and Computer Vision VIII: Algorithms and Techniques, Philadelphia, PA, USA, 1 March 1990; Volume 1192, pp. 297–308.
22. Hartley, R.; Zisserman, A. *Multiple View Geometry in Computer Vision*; Cambridge University Press: Cambridge, UK, 2003.
23. Rothwell, C.; Forsyth, D.; Zisserman, A.; Mundy, J. Extracting projective structure from single perspective views of 3D point sets. In Proceedings of the Fourth IEEE International Conference on Computer Vision, Berlin, Germany, 11–14 May 1993; pp. 573–582.
24. Carlsson, S. *European Conference on Computer Vision*; Springer: Berlin/Heidelberg, Germany, 1998; pp. 249–263.
25. Van Gool, L.; Moons, T.; Proesmans, M. Mirror and Point Symmetry under Perspective Skewing. In Proceedings of the Computer Vision and Pattern Recognition, San Francisco, CA, USA, 18–20 June 1998; pp. 285–292.
26. Hong, W.; Yang, A.; Huang, K.; Ma, Y. On symmetry and multiple-view geometry: Structure, pose, and calibration from a single image. *Int. J. Comput. Vis.* **2004**, *60*, 241–265.
27. Penne, R. Mirror symmetry in perspective. In *International Conference on Advanced Concepts for Intelligent Vision Systems*; Springer: Berlin/Heidelberg, German, 2005; pp. 634–642.
28. Sawada, T.; Li, Y.; Pizlo, Z. Any pair of 2D curves is consistent with a 3D symmetric interpretation. *Symmetry* **2011**, *3*, 365–388.
29. Cham, T.; Cipolla, R. Symmetry detection through local skewed symmetries. *Image Vis. Comput.* **1995**, *13*, 439–450.
30. Cham, T.; Cipolla, R. Geometric saliency of curve correspondences and grouping of symmetric contours. In *European Conference on Computer Vision*; Springer: Berlin/Heidelberg, Germany, 1996; pp. 385–398.
31. Cornelius, H.; Loy, G. Detecting bilateral symmetry in perspective. In Proceedings of the IEEE Computer Vision and Pattern Recognition Workshop, New York, NY, USA, 17–22 June 2006; p. 191.
32. Cornelius, H.; Perd'och, M.; Matas, J.; Loy, G. Efficient symmetry detection using local affine frames. In *Scandinavian Conference on Image Analysis*; Springer: Berlin/Heidelberg, Germany, 2007; pp. 152–161.
33. Shimshoni, I.; Moses, Y.; Lindenbaum, M. Shape reconstruction of 3D bilaterally symmetric surfaces. *Int. J. Comput. Vis.* **2000**, *39*, 97–110.
34. Köser, K.; Zach, C.; Pollefeys, M. Dense 3D reconstruction of symmetric scenes from a single image. In *Joint Pattern Recognition Symposium*; Springer: Berlin/Heidelberg, Germany, 2011; pp. 266–275.
35. Wu, C.; Frahm, J.; Pollefeys, M. Repetition-based dense single-view reconstruction. In Proceedings of the IEEE Computer Vision and Pattern Recognition, Colorado Springs, CO, USA, 20–25 June 2011; pp. 3113–3120.
36. Yang, A.; Huang, K.; Rao, S.; Hong, W.; Ma, Y. Symmetry-based 3D reconstruction from perspective images. *Comput. Vis. Image Underst.* **2005**, *99*, 210–240.
37. Bokeloh, M.; Berner, A.; Wand, M.; Seidel, H.; Schilling, A. Symmetry detection using feature lines. *Comput. Graph. Forum* **2009**, *28*, 697–706.
38. Cordier, F.; Seo, H.; Park, J.; Noh, J. Sketching of mirror-symmetric shapes. *IEEE Trans. Vis. Comput. Graph.* **2011**, *17*, 1650–1662.
39. Cordier, F.; Seo, H.; Melkemi, M.; Sapidis, N. Inferring mirror symmetric 3D shapes from sketches. *Comput.-Aided Des.* **2013**, *45*, 301–311.
40. Öztireli, A.; Uyumaz, U.; Popa, T.; Sheffer, A.; Gross, M. 3D modeling with a symmetric sketch. In Proceedings of the Eighth Eurographics Symposium on Sketch-Based Interfaces and Modeling, Vancouver, BC, Canada, 5–7 August 2011; pp. 23–30.
41. Sinha, S.; Ramnath, K.; Szeliski, R. Detecting and Reconstructing 3D Mirror Symmetric Objects. In *European Conference on Computer Vision*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 586–600.
42. Gao, Y.; Yuille, A. Symmetric bon-rigid structure from motion for category-specific object structure estimation. In *European Conference on Computer Vision*; Springer: Berlin/Heidelberg, Germany, 2016; pp. 408–424.

43. Cohen, A.; Zach, C.; Sinha, S.; Pollefeys, M. Discovering and exploiting 3D symmetries in structure from motion. In Proceedings of the IEEE Computer Vision and Pattern Recognition, Providence, RI, USA, 16–21 June 2012; pp. 1514–1521.
44. Michaux, A.; Jayadevan, V.; Delp, E.; Pizlo, Z. Figure-ground organization based on three-dimensional symmetry. *J. Electron. Imaging* **2016**, *25*, 061606.
45. Zabrodsky, H.; Weinshall, D. Using bilateral symmetry to improve 3D reconstruction from image sequences. *Comput. Vis. Image Underst.* **1997**, *67*, 48–57.
46. Li, Y.; Sawada, T.; Latecki, L.; Steinman, R.; Pizlo, Z. A tutorial explaining a machine vision model that emulates human performance when it recovers natural 3D scenes from 2D images. *J. Math. Psychol.* **2012**, *56*, 217–231.
47. Canny, J. A computational approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **1986**, *6*, 679–698.
48. Beckmann, N.; Kriegel, H.; Schneider, R.; Seeger, B. The R*-tree: An efficient and robust access method for points and rectangles. *ACM Sigmod Rec.* **1990**, *19*, 322–331.
49. Coughlan, J.; Yuille, A. Manhattan world: Orientation and outlier detection by bayesian inference. *Neural Comput.* **2003**, *15*, 1063–1088.
50. Bradski, G. The OpenCV Library. Avaiable online: http://www.drdobbs.com/open-source/the-opencv-library/184404319 (accessed on 16 January 2017).
51. Fischler, M.; Bolles, R. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM* **1981**, *24*, 381–395.
52. Harris, C.; Stephens, M. A combined corner and edge detector. In Proceedings of the Fourth Alvey Vision Conference, Manchester, UK, 31 August–2 September 1988; pp. 147–151.
53. Olsson, D.; Nelson, L. The Nelder–Mead simplex procedure for function minimization. *Technometrics* **1975**, *17*, 45–51.
54. François, A.; Medioni, G.; Waupotitsch, R. Reconstructing mirror symmetric scenes from a single view using 2-view stereo geometry. In Proceedings of the IEEE International Conference on Pattern Recognition, Quebec, QC, Canada, 11–15 August 2002; Volume 4, pp. 12–16.
55. Fawcett, R.; Zisserman, A.; Brady, M. Extracting structure from an affine view of a 3D point set with one or two bilateral symmetries. *Image Vis. Comput.* **1994**, *12*, 615–622.
56. Mitsumoto, H.; Tamura, S.; Okazaki, K.; Kajimi, N.; Fukui, Y. 3D reconstruction using mirror images based on a plane symmetry recovering method. *IEEE Trans. Pattern Anal. Mach. Intell.* **1992**, *14*, 941–946.
57. Huynh, D. Affine reconstruction from monocular vision in the presence of a symmetry plane. In Proceedings of the Seventh IEEE International Conference on Computer Vision, Kerkyra, Greece, 20–27 September 1999; Volume 1, pp. 476–482.
58. Sawada, T.; Li, Y.; Pizlo, Z. Detecting 3D mirror symmetry in a 2D camera image for 3D shape recovery. *Proc. IEEE* **2014**, *102*, 1588–1606.
59. Li, B.; Peng, K.; Ying, X.; Zha, H. Simultaneous vanishing point detection and camera calibration from single images. In *International Symposium on Visual Computing*; Springer: Berlin/Heidelberg, Germany, 2010; pp. 151–160.
60. Pirenne, M. *Optics, Painting & Photography*; JSTOR; Cambridge University Press: Cambridge, UK, 1970.