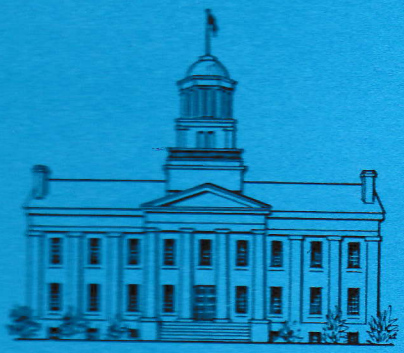
TECHNICAL REPORT 94-04

Shape and the Wide-Angle Image



RESEARCH REPORT

May 1994

by

Margaret Fleck

DEPARTMENT OF COMPUTER SCIENCE

THE UNIVERSITY OF IOWA • IOWA CITY

Shape and the Wide-Angle Image

Margaret M. Fleck[1]
Department of Computer Science
University of Iowa
Iowa City, IA 52242
USA
mfleck@cs.uiowa.edu
May 1994

Abstract:

This paper presents a new two-stage model of image formation: projection onto a sphere, followed by any of several projections onto a flat image. On the sphere, 2D shape does not depend on viewing direction. Close relationships between 3D and 2D exist not only for lines, but also for spheres and for local symmetries. Simple classes of 3D shapes can be reconstructed with few ambiguities.

Traditional perspective projection of the spherical image preserves straightness, thus features such as bitangent lines and curvature sign. Circles can be preserved by switching to stereographic projection. Stereographic projection is conformal (angle-preserving), handles very wide-angle views, reduces variation in brightness, and approximates a fish-eye lens. Using both projections, both straightness and circularity can be detected in wide-angle images. A fast linear algorithm for local symmetries is presented.

Keywords: stereographic projection, fisheye lenses, wide-angle images, imaging geometry, shape representation, object recognition, local symmetries, camera calibration

[1]This research was carried out at the Department of Computer Science, University of Iowa. It was supported in part by Research Initiation grant IRI-9209728 and Research Instrumentation grant CDA-9121985 from the NSF.

1 Introduction

In computer vision, perspective projection is generally accepted as the ideal model of camera imaging. Deviations from this ideal are analyzed as distortion, to be removed by suitable lens design [62] or software calibration [40, 57]. In perspective projection, straight lines project as straight.[2] This implies close relationships between the 3D notions of convexity, bitangency, and curvature sign and their counterparts in the 2D image [21, 22, 34, 36, 46, 51, 53, 54, 65]. It also implies that the search paths in stereo matching and the paths of features under camera translation [9] are linear (not curved).

However, perspective projection distorts the shape of objects if they are not centered in the field of view. For example, spherical objects appear circular in the center of the field of view, but elliptical in the periphery. Shape representations can be made invariant to these changes in shape [21, 22, 43, 47, 54, 64, 65], but at the cost of losing information about the shape and orientation of objects. Furthermore, a perspective image must cover less than a 180 degree field of view. Near 180 degrees, peripheral objects are grossly enlarged; beyond 180 degrees, perspective projection is no longer one-to-one.

These difficulties can be avoided by projecting onto a sphere (the *viewing sphere*) rather than onto a plane. Information is not lost, the full field of view is represented uniformly, and rotation about the camera center is made easy. A curved image surface also simplifies lens design [35]. Spherical projection is used (approximately) in the human eye, it is occasionally used (e.g. [14, 32]) for theoretical analyses in computer vision, and a few photographic systems have used spherical plates [35]. However, 2D images are much easier to work with: they are conveniently stored as arrays, low-level operations such as convolution are easy to implement, and one can use 2D rather than 3D geometry. Furthermore, neither spherical film nor spherical CCD arrays are available. These factors have prevented spherical projection from becoming popular.

This paper introduces an alternative imaging model, based on stereographic projection of the viewing sphere. In this model, spherical objects always appear circular. This implies a close relationship between local symmetries of a 3D object and local symmetries of its 2D image. The viewing sphere is mapped conformally onto the image, implying that small objects approximately preserve their shape no matter where they appear in the field of view and that the intersection angles of curves are preserved exactly. Finally, almost all of the viewing sphere can be represented using a single 2D image.

Both stereographic and perspective projection are plausible models for some real lenses [28, 31, 35, 50]. For narrow-angle lenses, the two projections are essentially identical. Good approximations to perspective projection can be obtained for fields of view up to about 110 degrees. Lenses with extremely wide fields of view–up to 220 degrees–are "fisheye" lenses, whose imaging geometry is similar to stereographic projection. Cheap wideangle lenses may fall somewhere in between these two extremes (figure 1). In computer vision and optics, such lenses are described as having significant "barrel distortion" [48, 62].

Thus, stereographic projection provides a second, alternative imaging model which may be preferable to perspective projection for some applications. If calibration information is available, a computer vision algorithm can use both projections simultaneously. For example, a shape algorithm can use properties based on circles and angles, together with properties based on straightness and properties based on smoothness (e.g. tangency, corners [1]). This better exploits the information

[2] In the interests of clarity, I will ignore degenerate cases and technical conditions except where they are directly relevant to my argument. Interested readers should have no difficulty filling in such details as how a line through the camera center appears in perspective projection, or which points cannot be projected because projection would involve dividing by zero.

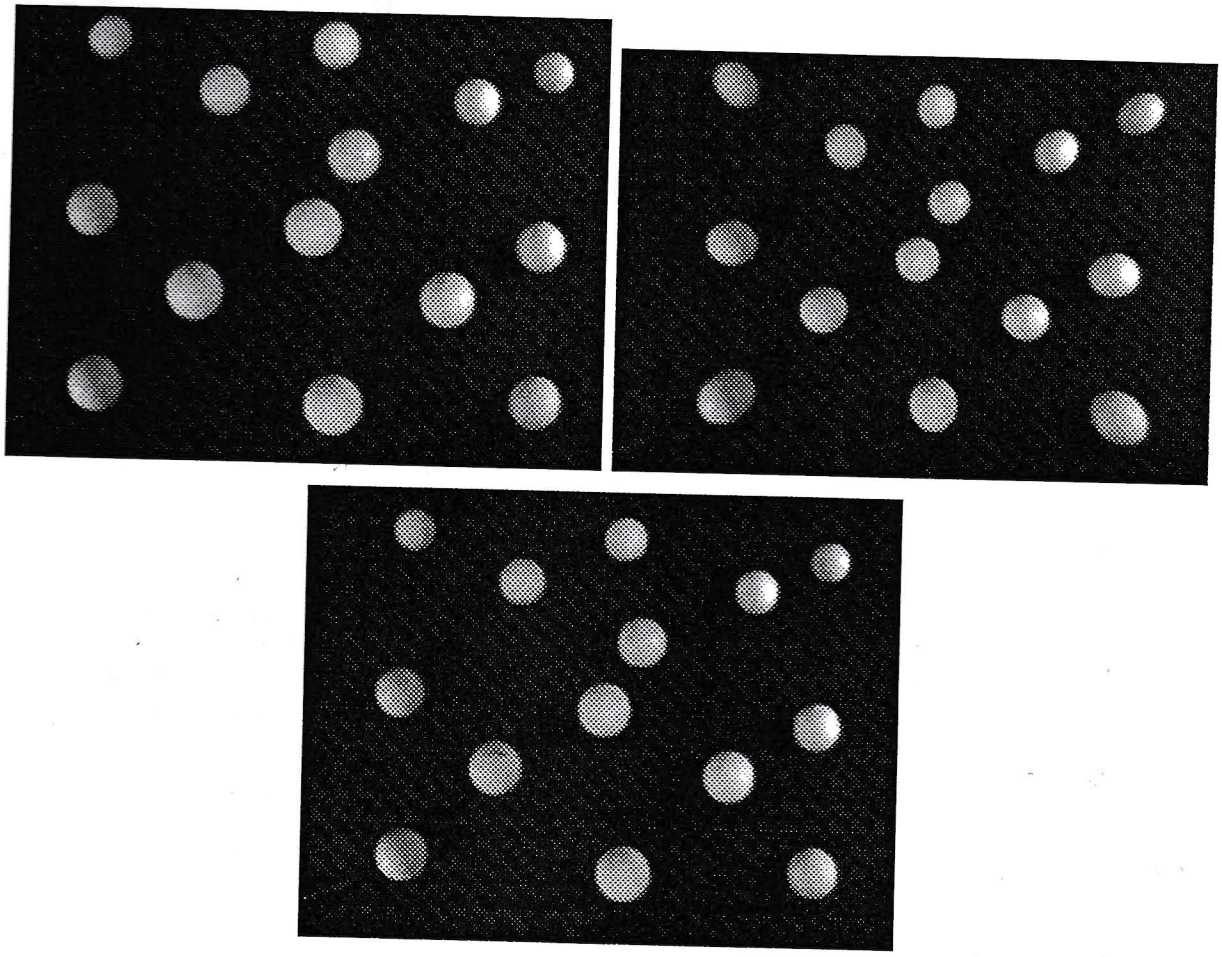


Figure 1: The outlines of spheres (table-tennis balls) appear as circles in stereographic projection (top left), but as a variety of ellipses in perspective projection (top right). The output of our wide-angle lens (93 by 72 degree field of view) is somewhere between these two projections (bottom).

present on the viewing sphere, allows richer representations of 2D regions and should eventually lead to more robust recognition of 3D objects.

The next section will define several camera models and their properties. Section 3 will discuss some key issues in more depth. Section 4 presents a theoretical analysis of local symmetry representations and section 5 shows how they can be implemented using a combination of perspective and stereographic coordinates.

2 Rethinking the concept of a camera model

Computer vision algorithms model ideal lenses using perspective projection, a suitable model for narrow-angle lenses [32, 48]. The other model in common use, orthographic projection plus scaling (sometimes called "weak perspective"), only approximates the behavior of a real lens when an object is near the optical axis and its diameter is small compared to its distance from the lens [48]. In this section, we will see how to construct a wider class of projection models, while retaining a close match to real lenses.

2.1 A traditional camera model

The most familiar equations for perspective projection map a 3D point (x, y, z) directly onto the 2D image location:

$$(x_p, y_p) = (\frac{f(x - c_x)}{z}, \frac{g(y - c_y)}{z}) \tag{1}$$

where c_x and c_y are the coordinates of the image center (the intersection of the optical axis with the image plane) and f and g are constants incorporating lens focal length, CCD element spacing, and the sampling rate in the framegrabber.[3] For narrow-angle lenses, this model is fairly accurate. For low-precision applications, any errors in the projection model will be smaller than errors from other sources, such as misplacement of edges because of smooth shading, shape variation within a class of objects, approximation of edges by polynomial curves, etc.

Good approximations to the parameters of perspective projection–f, g, c_x, and c_y–can be obtained from the specifications supplied with video cameras and lenses. For high-precision applications, camera calibration [40, 57] can supply precise values and correct small errors in the imaging geometry. Much of this calibration can also be done using two views of feature points under an unknown camera transformation [16] or one view of a known object [23]. Furthermore, it has recently been shown [21, 22, 47, 54, 64, 65] that some types of objects can be distinguished using properties which are invariant to changes in the projection parameters. These algorithms do not require values for the projection parameters or require them only to lie within broad ranges (e.g. an upper bound on the field of view). However, they require a close approximation to ideal perspective projection.

Wide-angle lenses, however, may diverge significantly from perspective projection. The distortion is typically radial. That is, it changes the distance, but not the direction, of each image point, relative to the image center. This causes straight lines (except those through the image center) to appear bent; the curvature increases with distance from the image center. Figure 2 shows that the output of our camera [4] contains substantial radial distortion (figure 2). The image of a straight line may

[3] The following discussion does not depend on how one handles points with $z = 0$.

[4] A NEC NX18A Color CCD Camera with a Computar 3.6 mm lens, 93 by 72 degree field of view

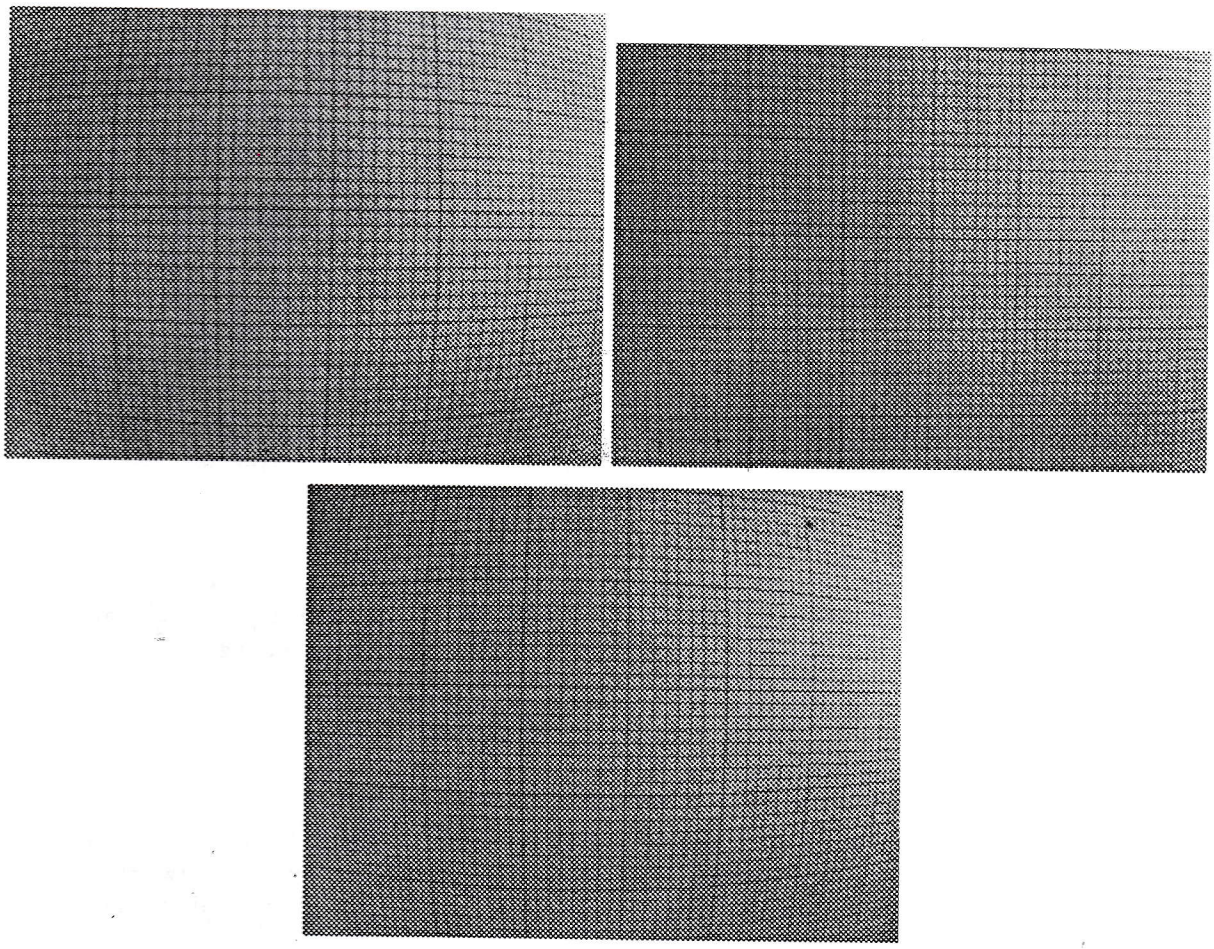


Figure 2: Lines in 3D (in this case, lines on a flat piece of graph paper) appear as lines in perspective projection (top right). In stereographic projection, they appear as circular arcs (top left) unless they pass through the image center. Compared to the perspective image, the stereographic image has "barrel" distortion. The output of our wide-angle lens is shown for comparison (bottom): its field of view is about 93 by 72 degrees.

contain as much as a 20 degree range of orientations. The apparent width of a graph paper square, which should be constant, varies by a factor of two across the image. These errors are too large for most vision applications to ignore.

Therefore, wide-angle lenses must be calibrated to remove radial distortion. For high-precision applications, high-precision calibration targets are required: the reader is referred to [40, 57]. A similar result can be obtained by using a robot arm to move the camera at medium precision (cf. [57]). For low-precision applications, most of the distortion can be removed with a simple, low-technology procedure: see Appendix A. In any case, it seems difficult or impossible to estimate the distortion without also estimating f, g, c_x, and c_y.

In this paper, I will assume that the camera system has been calibrated, at least approximately. First, calibration seems impossible to avoid if wide-angle lenses are to be used, because of radial distortion. Second, approximate parameter values are easy to obtain. Third, a non-zoom lens only needs to be calibrated once, because the calibration does not change over time. Finally, a calibrated

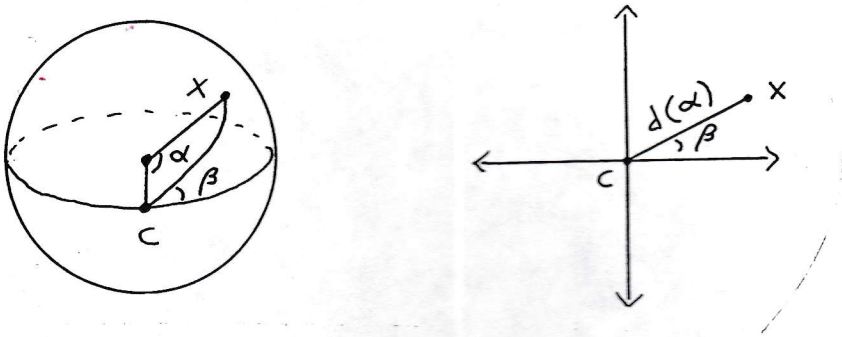


Figure 3: A position X on the viewing sphere can be represented by two angles α and β relative to a reference point C on the sphere (left). Projection of the sphere onto the plane (right) will preserve β. It maps each value of α onto a distance $d(\alpha)$ from the image center.

image provides more information about the shape of objects than an uncalibrated image (see below, section 3.2). To simplify the following theoretical discussion, I will assume that $f = g = 1$ and $c_x = c_y = 0$. The implementer of a real system must adapt the analysis by adding appropriate scaling constants.

2.2 Spherical projection

The traditional camera model projects the world onto the image in one step. In order to develop and analyze alternative imaging models, I will decompose the perspective projection into two steps. First, the 3D scene will be projected onto a sphere. Then, the spherical image will be projected onto a plane, to produce the final (flat) camera image.

I will define the *viewing sphere* to be a unit sphere centered at the origin. The 3D scene is projected onto the viewing sphere through the center of the sphere. That is, the point (x, y, z) is projected onto

$$(x_u, y_u, z_u) = (\frac{x}{r}, \frac{y}{r}, \frac{z}{r}) \tag{2}$$

where $r = \sqrt{x^2 + y^2 + z^2}$. This spherical representation of the scene contains all the information that is potentially available to a computer vision algorithm.

Alternatively, suppose that we choose $C = (0, 0, 1)$ to be a reference point on the sphere, marking the direction of gaze. We can then describe the image (x_u, y_u, z_u) of (x, y, z) in terms of its angular position, (α, β) relative to C (figure 3). The latitude angle α from C to (x_u, y_u, z_u) is $\text{atan}(q, z)$, where $q = \sqrt{x^2 + y^2}$ and atan is a 2-parameter inverse tangent function of the sort found in many programming languages.[5] The longitude angle β measures the position of (x_u, y_u, z_u) in a circle around C and is given by $\text{atan}(x, y)$. Angular representations of the field of view are often used in optics, in opthalmology, and visual psychophysics.

[5] Specifically $\text{atan}(a, b)$ returns the angle whose sine is a and whose cosine is b. I will assume that its output lies in the range $[-180, 180]$ degrees.

Projection onto the viewing sphere preserves two key features of 3D shape. A straight line in 3D, not passing through the origin, projects onto half a great circle on the viewing sphere. To see this, intersect the viewing sphere with the plane that contains the 3D line and the origin. Conversely, every half great circle corresponds to (among other possibilities) a set of 3D lines. Secondly, a 3D sphere S which does not contain the origin projects onto a region whose outline is a (non-great) 2D circle, because the situation is symmetric about the line which contains the origin and the center of S. Conversely, every non-great circle on the viewing sphere corresponds to a set of 3D spheres (among other possibilities).

In general, projection of objects onto the viewing sphere is simpler than projecting them onto a plane, because the shape of an object does not depend on its position in the field of view. Rather, the apparent shape of an object depends only on its distance and orientation relative to the center of the viewing sphere. If an object is small relative to its distance from the center of the viewing sphere, its projection onto the viewing sphere can be approximated by scaled orthographic projection onto a plane. The apparent shape of an object can vary less under orthographic projection than under perspective projection, e.g. parallel lines remain parallel, the projections of a flat object vary by affine transformations [34]. Consequently, a wider range of shape analysis algorithms can be used (e.g. [6, 34, 47, 58, 59, 60, 64]).

Given some specification of the accuracy required from our orthographic approximation, we can verify whether an observed object is sufficiently small in two dimensions by measuring its angular diameter on the viewing sphere. Its size in the third (depth) dimension, relative to its distance from the observer, can be constrained using *a priori* knowledge of the object, e.g. a bound on the elongation of objects in our modelbase. Alternatively, we can impose a bound on the tilt of the object relative to the observer and accept that the algorithm will fail if an object exceeds this tilt. This approach is standard in stereo matching. In either case, we can establish a maximum angular diameter below which orthographic projection is sufficiently accurate.

2.3 Projections of the viewing sphere

The viewing sphere could be projected onto the plane in a variety of ways. I will consider only projections that are symmetric about the z-axis and preserve β (at least for the front half of the field of view). Real lens systems have these properties, because it is difficult to manufacture lenses which are not rotationally symmetric. Therefore, suppose that we represent points on the output image using polar coordinates (r, θ) about the image center. The conditions on our projection imply that $\theta = \beta$. Thus, to specify a projection, we need only specify how each value of α is mapped onto a radius r (figure 3).

There are (at least) four reasonable, qualitatively different ways to map values of α onto values of r:

- $r = \tan(\alpha)$ (perspective projection)
- $r = 2\tan(\frac{\alpha}{2})$ (stereographic projection)
- $r = \alpha$ (equidistant fish-eye projection)
- $r = \sin(\alpha)$ (orthographic projection)

Graphs of these functions, normalized so that they are all identical at 45 degrees, are shown in figure 4. Narrow angle or corrected wide-angle lenses are designed to emulate perspective projection.

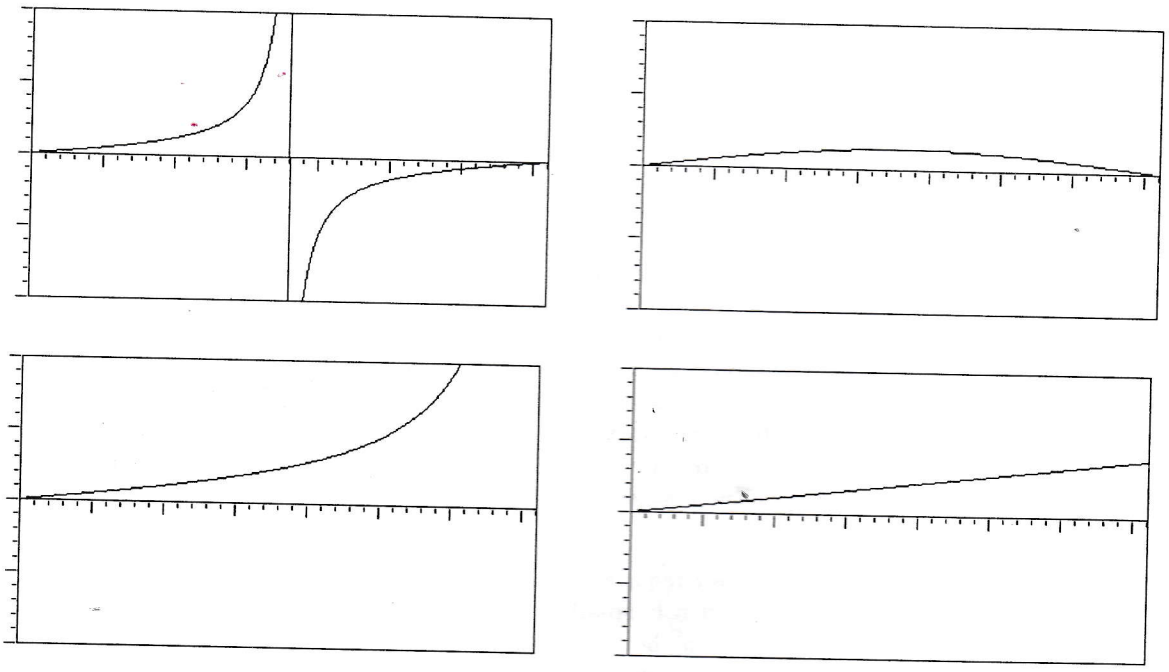


Figure 4: Four methods of mapping angles (from 0 to 180 degrees) onto distances from the image center. Perspective and orthographic projection (top). Stereographic and equidistant fish-eye projection (bottom). Vertical units have been normalized so that all four functions have the same value at 45 degrees.

Fisheye lenses are designed to emulate equidistant fish-eye or orthographic (also called sine law) projection [35] of the viewing sphere.[6]

Stereographic projection is widely used in mathematics [27, 44]. It is known as a way to emulate fish-eye lenses in graphics [31] and as an alternative representation for shape from shading in computer vision [32]. It is not (as far as I know) used in the design of lenses. However, it is very similar to equidistant fish-eye projection for a wide range of angles. Table 1 shows the ratio of perspective, stereographic, and orthographic radii to the equidistant fish-eye radii (i.e. the angles). These ratios have been normalized so that they are all 1.0 at 45 degrees. Stereographic coordinates are very close to the equidistant fish-eye coordinates for angles up to about 100 degrees from the image center (200 degree field of view).

[6] Notice that this has an effect quite different from orthographic projection of the 3D scene directly onto the image plane.

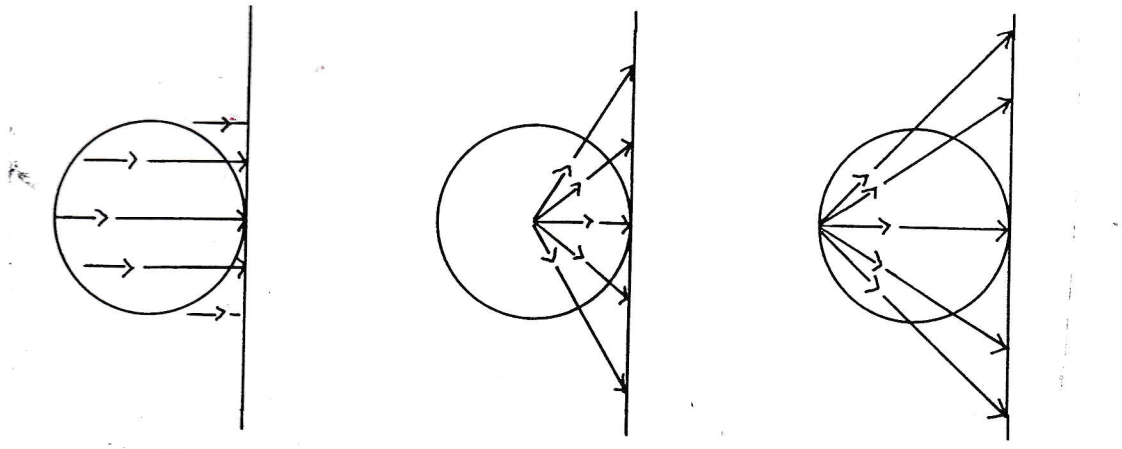


Figure 5: Geometrical constructions of the image of a point X under orthographic projection (left), perspective projection (middle), and stereographic projection (right).

Table 1: Comparison to equidistant fish-eye radii (Normalized to be 1.0 at 45 degrees)

angle	perspective	stereographic	orthographic
10	0.79	0.95	1.11
20	0.82	0.96	1.09
30	0.87	0.97	1.06
40	0.94	0.99	1.02
50	1.07	1.01	0.98
60	1.30	1.05	0.92
70	1.77	1.09	0.85
80	3.19	1.14	0.78
90	$\pm\infty$	1.21	0.71
100	-2.55	1.29	0.63
110	-1.12	1.41	0.54
120	-0.65	1.57	0.46
130	-0.41	1.79	0.38
140	-0.27	2.13	0.29
150	-0.17	2.70	0.21
160	-0.10	3.85	0.14
170	-0.05	7.30	0.07
180	-0.00	∞	0.00

Three of these projections correspond to simple geometrical constructions (figure 5). Suppose that the image plane is tangent to the viewing sphere at $(0, 0, 1)$. The orthographic projection of a point X on the sphere is the intersection of the image plane with a line through X perpendicular to the image plane. The perspective projection of X is the intersection of the image plane with the line containing X and the center of the viewing sphere. The stereographic projection of X is the intersection of the image plane with the line containing X and the far pole $(0, 0, -1)$.

Using these constructions, it is easy to derive specific equations for projecting a point (x_u, y_u, z_u) on the sphere to a point (x', y') on the plane:

- perspective: $(x', y') = (\frac{x_u}{z_u}, \frac{y_u}{z_u})$
- stereographic: $(x', y') = (\frac{2x_u}{z_u+1}, \frac{2y_u}{z_u+1})$
- equidistant fish-eye: $(x', y') = (px_u, py_u)$ where $p = \text{atan}(\sqrt{x_u^2 + y_u^2}, z_u)$
- orthographic: $(x', y') = (x_u, y_u)$

From these equations, it is clear that perspective projection only preserves the angle β for the front half of the viewing sphere. In projecting the back half, each angle β is replaced by the opposite angle (because z_u is negative). By contrast, the other three projections preserve the angle β for all points on the sphere.

2.4 Properties of projections

Each of these four projections offers some advantages but has some disadvantages, summarized in Table 2. These properties fall into three categories. First, the projection should represent as much of the viewing sphere as possible, preserving its topology.[7] Second, the brightness of the image of an object should not depend on its position in the field of view. Large variations in brightness mean that some part of the image will be either very dark or saturated, no matter how the aperture is set. Finally, the shape of regions on the viewing sphere should be preserved in the planar image, because this leads to simpler algorithms for shape description, texture analysis, and stereo matching.

Table 2: Properties of projections

	perspective	stereographic	equidistant	orthographic
one-to-one?	no	yes	yes	no
continuous except $(0, 0, -1)$	no	yes	yes	yes
front-side image finite?	no	yes	yes	yes
back-side image finite?	no	no	yes	yes
uniformity of illumination?	bad	ok	ok	perfect
preserves straightness?	yes	no	no	no
conformal?	no	yes	no	no
preserves circles?	no	yes	no	no

In order to represent wide-angle views with planar images, the ideal projection should map the entire viewing sphere one-to-one onto the plane, preferably onto a bounded portion of the plane. Perspective projection maps the front half of the viewing sphere onto the entire plane, there is a singularity at 90 degrees from the viewing direction, and the back half is projected on top of the front half. This limits perspective images to less than a 180 degree field of view. By contrast, stereographic projection maps the sphere (minus one point at the back) one-to-one onto the entire plane. Equidistant fish-eye projection maps the sphere (minus one point) onto a bounded region of the plane. Finally, orthographic projection maps the front half of the viewing sphere onto a bounded region of the plane and the back half is projected onto the same region.

Ideally, a uniformly illuminated piece of paper should generate an image filled with a constant value. This is not typically the case for real lenses [35, 48, 62]. With an ideal perspective lens, the apparent intensity of a patch of surface at angle α from the optical axis will be diminished by a factor of $\cos^4(\alpha)$. A drop of $\cos^2(\alpha)$ is created because the image location is $\frac{1}{\cos(\alpha)}$ further from the lens than a location on the optical axis and intensity drops as the square of distance. Two further

[7] Exact preservation of the entire viewing sphere is clearly impossible: a sphere is not homeomorphic to a plane.

factors of $\cos(\alpha)$ are introduced because the image location sees the exit pupil at an angle and light hits the image plane at an angle. If the lens has a 90 degree field of view, this means the edges of the image will be only one fourth as bright as the center. Further variation in illumination may be caused by vignetting or stop distortion [35].

The transformation from perspective to any of the other projections tends to make illumination more uniform across the image. First, a unit square near the periphery of the image represents a larger portion of the viewing sphere than a similar square near the middle of the image. Second, for a particular angle α on the viewing sphere, the corresponding image location is closer to the optical axis. Therefore, images from stereographic or equidistant fisheye lenses are more evenly illuminated than perspective images. Ideal orthographic (sine law) projection of the viewing sphere apparently creates perfectly even illumination [35].

Finally, perspective and stereographic projection create useful relationships between 3D and 2D shape. Perspective projection preserves straightness. That is, it maps great circles on the viewing sphere–the images of straight lines in 3D–onto straight lines in the planar image. Stereographic projection preserves circularity: it maps circles on the viewing sphere–images of spheres in 3D–onto circles and lines on the image plane (see section 3.4) for details). Furthermore, stereographic projection is *conformal* [27, 44]. That is, if two curves intersect at angle θ on the viewing sphere, they intersect at the same angle θ on the image plane. This implies that shape is approximately preserved for small patches of the viewing sphere, with the approximation improving as the patch size is decreased. I am not aware of any similar shape-preservation properties for orthographic and equidistant fisheye projection of the viewing sphere.

Taking all these properties as a whole, stereographic projection seems to be the best choice for a general-purpose vision representation. It maps most of the sphere[8] one-to-one onto a bounded region of the plane. It preserves circularity and local shape. Although it does not preserve global straightness, this can be remedied by using perspective as a secondary coordinate system. Finally, stereographic projection provides fairly uniform illumination of the image. For intensity-based algorithms (e.g. color constancy, shape from shading, texture, stereo matching) the remaining variation can be removed by calibration.

3 Selected topics in more details

Because these projection methods are unfamiliar, certain issues deserve more detailed discussion.

3.1 Why use wide-angle lenses?

Many of the issues discussed above could, of course, be avoided by using only narrow-angle lenses (e.g. field of view less than 50 degrees). For these lenses, radial distortion is not substantial and the four projections of the viewing sphere are approximately the same. However, the lack of peripheral vision is a handicap for visual searching, navigation, and detecting objects moving towards the observer. Although a human eye has good resolution for only about 50 degrees, the full visual field extends at least 90 degrees [4, 63]. The peripheral areas deliver only low-resolution information, but this is useful for detecting motion and finding large objects. Inexpensive 35mm and C-mount lenses can deliver a field of view as large as 110 degrees.[9] Wide-angle lenses have a very large depth of

[8] Specifically: the sphere minus any ball larger than a point around $(0, 0, -1)$ is mapped onto a bounded region of the plane.

[9] 180 degree lenses are expensive; 220 degree lenses prohibitively so.

field, an asset for identifying objects when the camera and/or objects may change position.

Furthermore, people can assemble wide-angle images of the world by rotating their eyes and combining information from both eyes. Without moving your head, your field of view is nearly 180 degrees. This panoramic view provides a context for deciding where to direct your high-resolution foveal vision. It also allows you to detect unexpected events (e.g. an attacking lion, an approaching car, a softball) coming from many directions. The human field of view is limited by the fact that both eyes point forwards, so as to provide stereo depth information. Animals for whom panoramic view is more important than stereo (e.g. many herbivores) have eyes that point sideways and an even wider field of view.

Finally, in mapping obstacles for robot navigation, it might be useful to represent the distance to the nearest obstacle in all directions, including behind the observer. Although perspective projection is limited to views less than 180 degrees, the three fisheye projection models could be used to create a stereo depth or disparity image covering most of the entire viewing sphere. Or, alternatively, they could provide a uniform set of coordinates for a more compact, symbolic representation of obstacles (e.g. in terms of regions and their boundaries).

3.2 Why calibrate field of view?

Data from any of the four projection models can be converted to spherical projection, if the camera system is calibrated. In particular, one must know the field of view. [10] This can vary substantially, depending on the lens chosen. Compact, affordable 35mm and C-mount lenses have fields of view ranging from 110 degrees down to 5-10 degrees,[11] a range of more than an order of magnitude. Matching the resolution of human foveal vision with a standard CCD camera, e.g. for reading or close inspection, would require a field of view between 4 and 7 degrees (based on data in [63]).

Camera calibration is required whenever the vision system must interact with a 3D positioning system, such as a robot gripper or a robot camera mount. If the lens is perspective, without significant radial distortion, many aspects of 3D shape can be computed without calibration using representations based on projective invariants [16, 21, 22, 43, 47, 54, 65]. However, potentially useful information is lost when calibration data is not available. For example, the ellipsoid in figure 6 can only be distinguished from the spheres if the image's aspect ratio (ratio of horizontal to vertical distances in the image) is known. If the image is perspective, the image center must also be known (cf. [47]).

The field of view parameter is required to establish a relationship between the distance to an object and the object's 3D size. If the camera is calibrated, we can scale positions returned by the edge finder so that they lie in the ideal perspective or stereographic images defined in Section 2. The projection equations can then be inverted to derive coordinates on the viewing sphere. For example, we can compute the angle subtended by a spherical object. If the field of view is unknown, however, we can only derive a 1-dimensional family of possible shapes on the viewing sphere.

Specifically, a sphere projects to a circle on the viewing sphere. If its angular radius is α and the distance between its center and the origin is d, then its 3D radius is $d\sin(\alpha)$ (figure 7). Consider an object made of two identical spheres of unknown radius r, subtending (known) angles 2α and $2\alpha'$, with centers separated by (known) angle β. The distances to the centers of the spheres are $\frac{r}{\sin(\alpha)}$ and $\frac{r}{\sin(\alpha')}$. The distance between the centers of the spheres is then rk, where k is given by:

[10] It would be more traditional to refer to this parameter as focal length, but using focal length in shape analysis requires additional parameters to represent the physical size and pixel resolution of the CCD array, as well as any adaptors used to connect the lens to the camera.

[11] Lenses with even smaller fields of view exist, but they are expensive and so long as to be unwieldy.

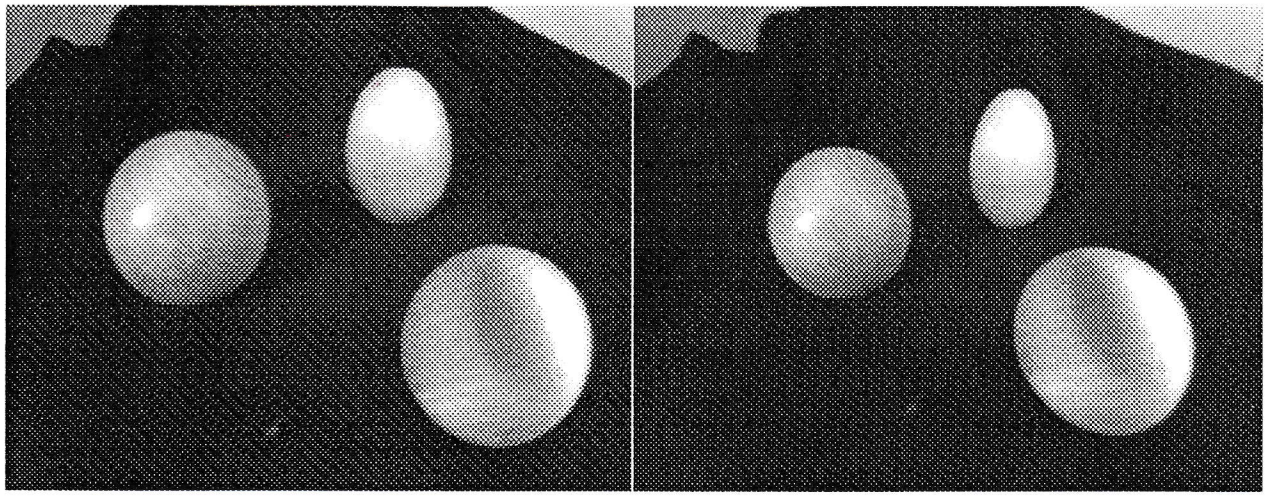


Figure 6: A image containing an object that is approximately ellipsoidal and two spheres. If we know that the image has not been compressed vertically or horizontally ($f = g$ for perspective) and that the optical axis runs through the center of the image, the uppermost figure cannot be an image of a sphere. In stereographic projection (left), it would project as a circle. In perspective projection (right), its axis would point towards the center of the image.

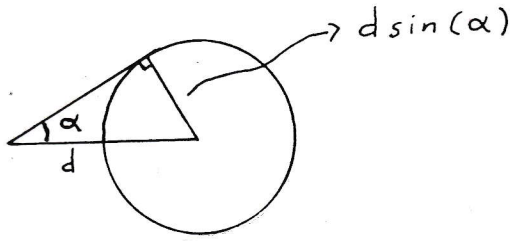


Figure 7: If a sphere lies at distance d from the center of the viewing sphere and it subtends an angle 2α, then its 3D radius is $d\sin(\alpha)$.

$$k^2 = \frac{1}{\sin^2(\alpha)} + \frac{1}{\sin^2(\alpha')} - \frac{2\cos\beta}{\sin(\alpha)\sin(\alpha')} \tag{3}$$

Thus, from the calibrated image, we can derive the ratio of the sphere radius to the inter-sphere distance. This information could be used, for example, to distinguish two barbells with the same length but different-sized ends. We can also compute the angle γ by which the barbell is tilted away from a frontal view:

$$\cos\gamma = \frac{\sin(\alpha)}{2k}(k^2 + \frac{1}{\sin^2(\alpha)} - \frac{1}{\sin^2(\alpha')}) \tag{4}$$

(See e.g. [5] for the required formulas.) To achieve reasonable accuracy, the distance between the spheres must be large relative to their distance from the observer.

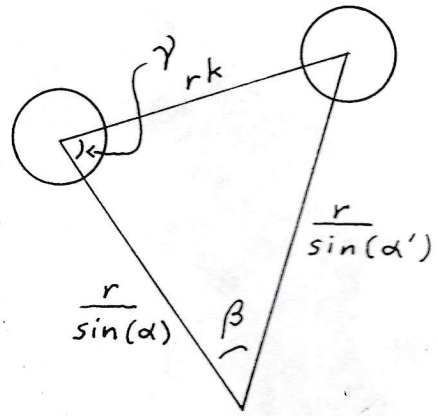


Figure 8: The distance between two identical spheres, relative to their radii, can be calculated from their images on the viewing sphere. We can also derive the tilt angle γ.

This information cannot be derived without the field of view information. If we vary the focal length of the lens, adjusting the viewing distance so that the front sphere remains a constant size, the image of the back sphere will change in size.[12] The same variation in apparent shape can also be created by changing the distance between the two spheres. Thus, an algorithm which is invariant to changes in calibration cannot distinguish barbells with different proportions.

3.3 Why is a conformal projection desirable?

An important property of stereographic projection is that it is conformal. Thus, the shape of a small region on the viewing sphere is approximately preserved in the image, with the errors in the approximation decreasing towards zero as one considers smaller neighborhoods on the viewing sphere [27]. As we saw in Section 2.2, projection of small 3D objects onto the viewing sphere approximates scaled orthographic projection onto a plane. Combining these two facts, we see that the apparent shape of a small 3D object in a stereographic image is approximately an orthographic projection of the object onto a plane.

This approximation does not hold for perspective projection of the viewing sphere. In perspective images, the apparent shape of a small object depends on its location in the field of view. This *perspective distortion* is significant for objects which maintain a relatively constant shape on the viewing sphere, regardless of viewpoint. For example, a sphere always appears circular on the viewing sphere and in a stereographic image, but can appear as a variety of conics in the perspective image. A rotationally symmetric object must appear symmetric on the viewing sphere, because everything involved is reflectionally symmetric in the plane containing the 3D axis of the object and the center of the viewing sphere. If the object is small, this symmetry is approximately preserved in a stereographic image. In a perspective image, however, the two sides of a small object may diverge significantly from symmetry if the object lies far from the optical axis (see [43] and figures 9-10).

For flat 3D objects (or flat faces of polyhedra), the situation is more subtle. A flat 3D object projects to a wide variety of shapes on the viewing sphere, depending on how it is tilted relative to the observer. The shape can vary by any perspective transformation [21, 54, 65] or, if the object

[12] Unless, of course, the spheres happen to be at the same distance from the center of the viewing sphere.

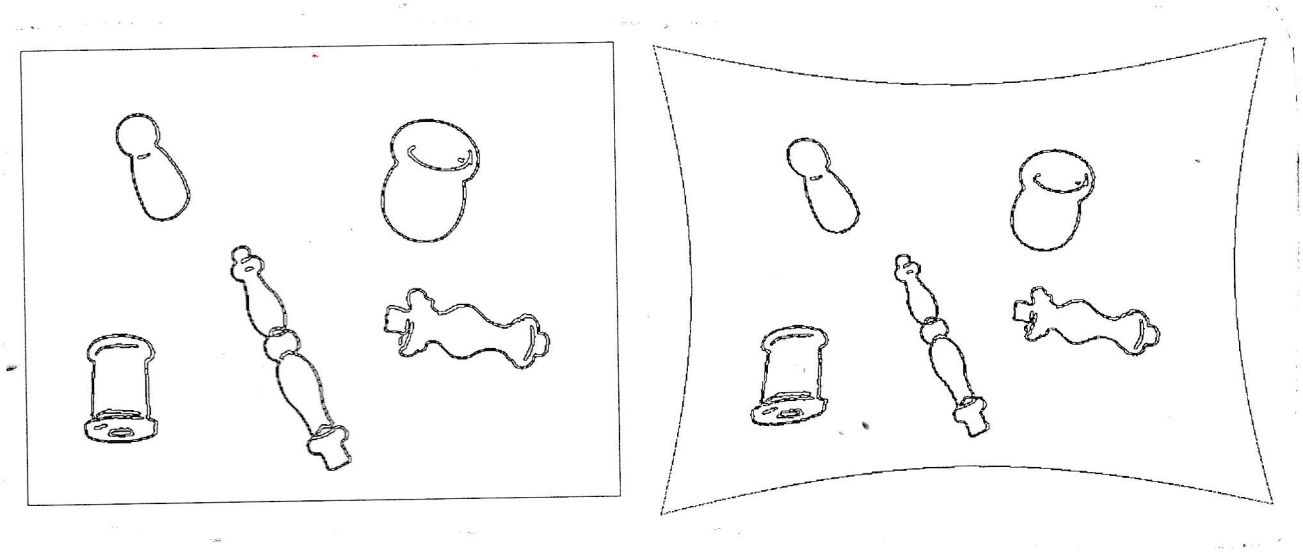


Figure 9: The edges of a set of rotationally symmetric objects, in stereographic (left) and perspective (right) coordinates. Notice how the symmetries of the lefthand objects are distorted in the perspective edges.

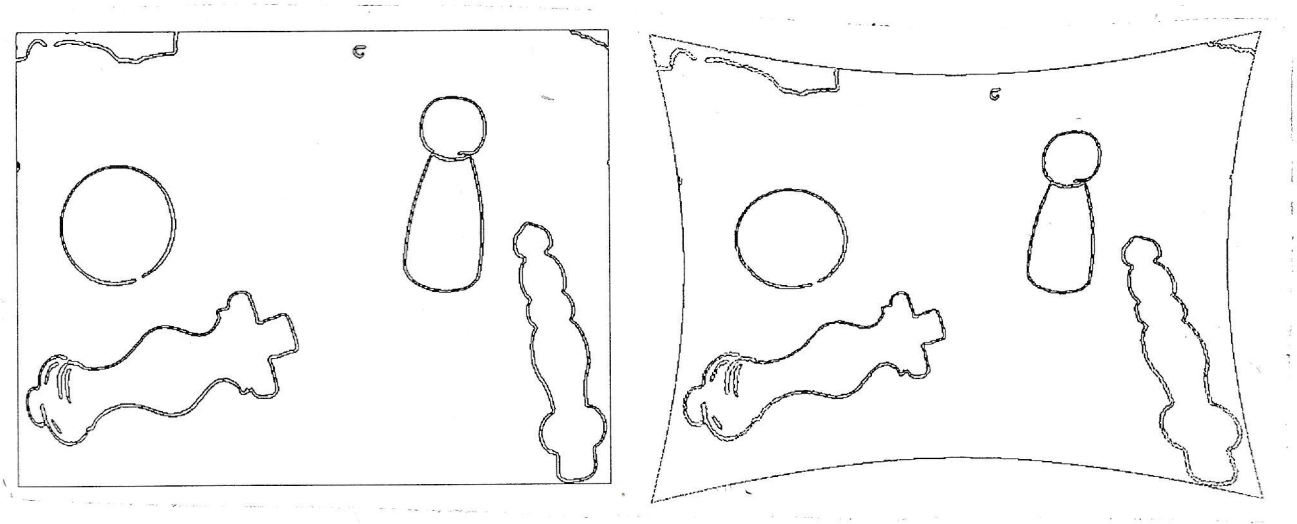


Figure 10: Edges for a second set of rotationally symmetric objects. The lower right object shows distortion of symmetry in the perspective image (right) and curving of its axis in the stereographic image (left). This object also appears (lower center) in the previous figure.

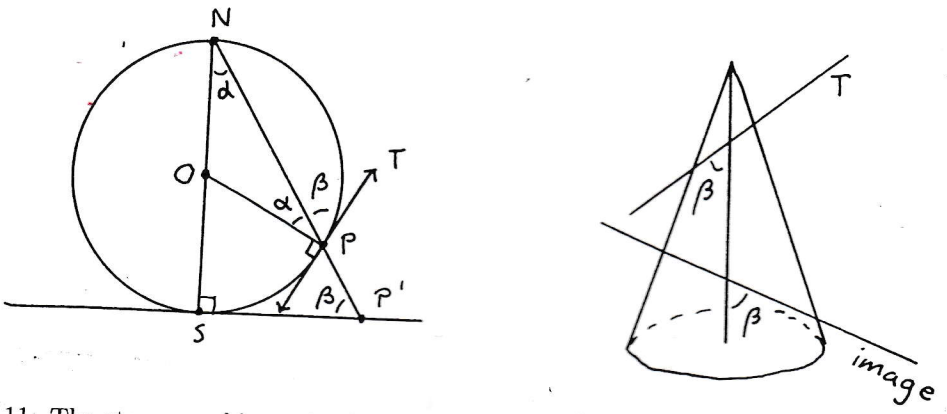


Figure 11: The stereographic projection line NP' intersects the tangent plane at P and the image plane at the same angle (β) (left). Consider the elliptical cone from N through a circle centered at P. The two planes cut this cone at equal angles (right), so their intersections have the same shape.

is small, by a transformation that is approximately affine [34, 47]. Perspective distortion creates approximately the same class of shape distortions. Thus, it does not introduce any new variety in the apparent shape of a flat object but it makes it more difficult to recover the object's 3D orientation (tilt).

Furthermore, suppose that one rotates a camera about its lens center. Each object, whether flat or rounded, retains the same apparent shape on the viewing sphere. Because stereographic projection is conformal, small objects keep the same shape in each stereographic image, though they may change size, translate, or rotate. This property could be useful in matching corresponding bits of two images, e.g. to glue them together into one ultra-wide-angle view. In perspective images, by contrast, even small objects change shape as the rotation moves them from the center of the field of view to the periphery.

3.4 Why does stereographic projection preserve circles?

In addition to the approximate preservation of shape implied by being conformal, stereographic projection of the viewing sphere projects circles onto circles or straight lines, exactly and no matter what size the circle is. Recall that a circle on the viewing sphere is the intersection of the viewing sphere with a plane. A circle containing the point $(0, 0, -1)$ must project onto a line because all the projection rays are coplanar.

To understand the general case, consider figure 11, which shows the projection of point P to point P' through the far pole $N = (0, 0, -1)$ of the viewing sphere. Let S be the near pole and O be the origin, i.e. the center of the sphere. Let T be the plane tangent to the viewing sphere at P. The two angles labelled α must be the same because the segments OP and ON both have length 1. The angle β between NP and T must be $90 - \alpha$ because a tangent to a sphere (T) is perpendicular to the corresponding radius (OP). The angle between NP' and the image plane is also $90 - \alpha$ (i.e. β) because the SNP' is a right triangle.

Now choose a circle C on the viewing sphere, centered at P. Assume that C does not contain N. Consider the cone of rays that start at N and pass through points of C. The cross-section of this cone must be an ellipse. The image plane and T cut the cone at equal, but opposite, angles.

The plane containing C is parallel to the tangent plane at P. Therefore, the intersections of these three planes with the cone must be the same shape. Since C is circular, they are all circular. This proof is somewhat informal; see [27] for more details.

3.5 How are stereographic and perspective coordinates related?

If the camera is calibrated, image coordinates can easily be converted from stereographic to perspective projection, or vice versa. The stereographic projection (x_s, y_s) and the perspective projection (x_p, y_p) of a point on the viewing sphere are related via the following equations (adapted from [32]):

$$(x_s, y_s) = \left(\frac{2x_p}{1+\sqrt{1+r_p^2}}, \frac{2y_p}{1+\sqrt{1+r_p^2}}\right) \tag{5}$$

where $r_p = \sqrt{x_p^2 + y_p^2}$ and

$$(x_p, y_p) = \left(\frac{4x_s}{4-r_s^2}, \frac{4y_s}{4-r_s^2}\right) \tag{6}$$

where $r_s = \sqrt{x_s^2 + y_s^2}$.

The transformation between perspective and stereographic coordinates is a special type of radial distortion. Specifically, suppose that a feature point X lies on a ray R starting at the center of the image. Then equation 5 or 6 changes the distance between X and the image center but does not move X off R. This is the defining characteristic of radial distortion. This property implies that most of Tsai's high-precision camera calibration algorithm [57]–including the critical decomposition into two stages and the entire first stage–does not depend on whether one's choice of ideal output projection is perspective or stereographic.

4 Application to local symmetries

The methods developed in the previous sections can be used to set local symmetry representations of shape on a firm theoretical footing. Specifically, we will see that the 3D symmetries of an object project onto symmetries of its 2D outline on the viewing sphere, whenever certain visibility conditions are satisfied.

4.1 What are local symmetries?

In defining local symmetries, I will assume that we are given a set of edges in the image or a set of surfaces in 3D. One side of each edge or surface is designated as the "object side" and one as the "background side." This designation may come from prior information (e.g. the object is darker than the background). Alternatively, each object boundary may generate two parallel edges or surfaces, representing the two ways to label the sides [18, 49]. I will assume that a normal direction has been computed for each edge or surface point and that the normal vectors always point towards

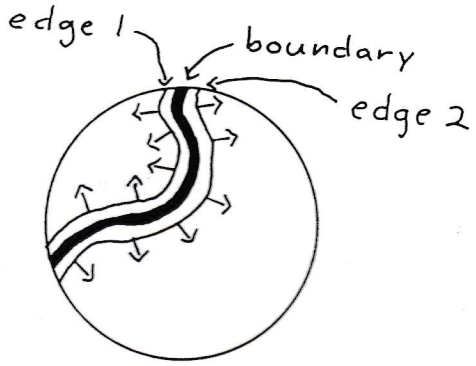


Figure 12: In the viewing sphere, the covariant normal to an edge point is tangent to the sphere, perpendicular to the edge's tangent vector, and points away from the boundary.

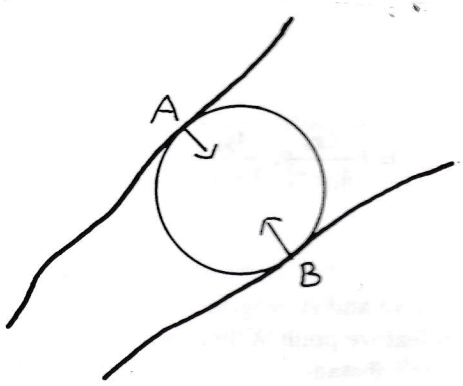


Figure 13: Two edge points A and B have a local symmetry if the edges at both points are tangent to a common circle and both normals point inwards.

the object side of the edge or surface.[13] If our edge lies in the viewing sphere, there is a well-defined 3D tangent direction T at each edge location P. I will define the *covariant normal* at P to be the (unique) unit vector which is (a) normal to T, (b) tangent to the sphere, and (c) points toward the object side (figure 12).

Traditionally [11, 12, 13, 24], two distinct edge points A and B in a 2D image are said to have a *local symmetry* if the edge is tangent to a common circle (figure 13) at these points. Some authors [11, 12] also require that both normal vectors point towards the center of the circle. (See Appendix B for discussion of the degenerate case.) Equivalently, the normals at A and B must be reflections of one another in the perpendicular bisector L of the line segment AB and both normals must point (strictly) towards L. I will refer to such points as *symmetry pairs* and to the tangent circle as the *symmetry circle*. The symmetries of a flat 3D object are the 2D symmetries of its outline in the plane containing it. It is well known that 3D symmetries of flat objects do not typically project to symmetries of the 2D outline.

To obtain a useful relationship between 3D and 2D symmetries, we need to switch our attention

[13] If your Platonic object model has sharp corners, either exclude an infinitely thin line of points along the corner or treat the corners as slightly rounded.

to a more general class of 3D objects and do our 2D geometry on the viewing sphere rather than on a flat image. In general, two points on a 3D surface have a local symmetry if the surface is tangent to a common sphere (the *symmetry sphere*) at both points, and both surface normals point towards the center of the sphere (cf. [13, 25]). So, for example, there is a local symmetry between any pair of points in the same cross-section of a rotationally symmetric object and the symmetry sphere is centered on the object's axis. To use this definition on flat objects, we must assume that they have some non-trivial thickness, e.g. replace each point of the planar object with a tiny filled sphere. Each symmetry circle in the traditional analysis is then replaced by a symmetry sphere with the same center.

I will say that two distinct edge points on the viewing sphere have a local symmetry if

(a) the edges at both points are tangent to a common circle C,

(b) C is not a great circle, and

(c) their covariant normals point into the smaller of the two regions bounded by C.

(Again, see appendix B for special cases.) If objects are transparent and our camera has a 360 degree field of view, then a 3D symmetry between points A and B creates a 2D symmetry whenever A and B project onto the object's outline on the viewing sphere. Conversely, every 2D symmetry on the viewing sphere corresponds to an infinite class of 3D symmetries (among other possibilities).

4.2 Visibility of individual symmetries

For opaque objects and a real camera system, a symmetry (A, B) is visible in the viewing sphere image if

(1) The rays from the lens center to A and B are tangent to the surface (equivalently, tangent to the symmetry sphere),

(2) The curvature of the surface along these rays is positive at A and B (so neither point is occluded locally),

(3) No other part of the surface lies in front of A or B, and

(4) The angular positions of A and B both lie within the camera's field of view.

Suppose that S is the symmetry sphere for A and B. Ignoring rotation about the center of the viewing sphere, the space of possible viewpoints is three-dimensional. At any viewing position, a circular set of points on the symmetry sphere are visible. In order for A and B to satisfy condition (1), they must lie on this circle. This will be true for a one-dimensional family of viewing positions. Specifically, the center of the viewing sphere must lie in the plane equidistant from A and B. Given an angular position in this plane, the required viewing distance (relative to the radius of the symmetry sphere) can be computed from the angular separation of A and B on the symmetry sphere. Condition (4) imposes a lower-bound on the viewing distance.

Condition (2) will always be satisfied if A and B are positive curvature elliptic points. It is satisfied for almost all viewpoints if they are parabolic. If they are hyperbolic, condition (2) is satisfied for a restricted range of viewpoints, see [37] for details. Except for the obvious special case of points on the convex hull (cf. [37]), condition (3) is global and, thus, hard to control.

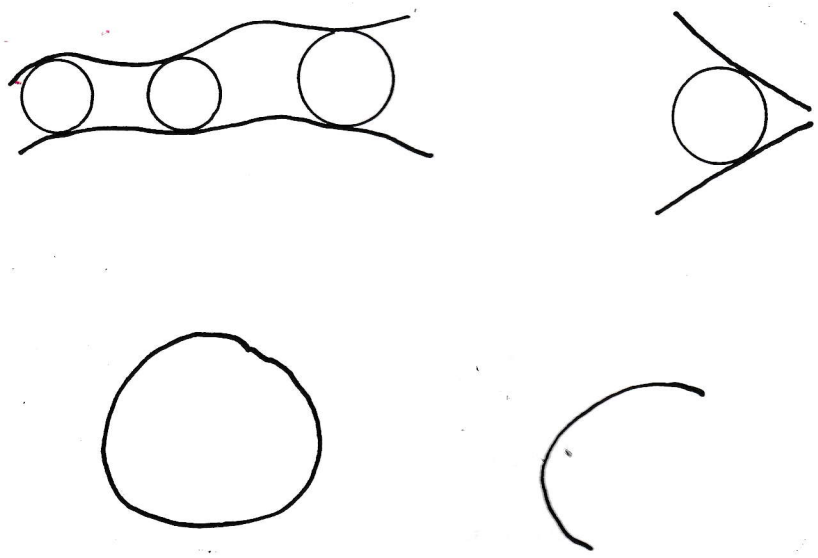


Figure 14: An elongated region (top) has two boundaries; opposite pairs of points are related by a local symmetry. In a round region (bottom), there is a single boundary; most or all pairs of points are related by local symmetries.

4.3 Symmetry regions

An isolated symmetry pair is of little use. A single 2D symmetry pair may have occurred by random chance. A single 3D symmetry pair almost never appears as a symmetry of the 2D outline. Therefore, symmetry algorithms primarily interpret symmetries involving extended sections of edge or extended patches of surface. Algorithms for detecting 2D symmetries [15, 17] often require that each region be long relative to its width to filter out symmetries due to random alignments of edges.

Specifically, I will define a *local symmetry region* to be a connected set of symmetry pairs. In 2D, the left sides of the symmetry pairs form a connected curve, as do the right sides. In 3D, the left sides and the right sides must each form a connected patch of surface. This follows the spirit of the definitions in [11, 12, 15, 17]; some previous authors [7, 8, 13, 24] have concentrated instead on axis connectivity. Some types of later processing may impose additional "smoothness" constraints which bound the rate at which shape parameters change along the region (e.g. [11, 12, 15]): such constraints are beyond the scope of this paper.

Figure 14 shows some simple 2D symmetry regions. In an elongated region, the left and right sides are disjoint. Each symmetry pair contains one point from each side. In a near-circular region, the left and right sides are the same curve. Many pairs of points are related by symmetries, with similar symmetry circles. A more complex symmetry region may display both types of pairing patterns, e.g. an elongated region with semi-circular ends. Because there may be errors in computed normals, practical implementations require only approximate symmetry. Thus, large connected symmetry regions can be formed despite irregularities in the contour.

Because the normals in a symmetry pair must point inwards, a region terminates when two corresponding sections of edge become tangent to a common line (rather than a common circle). This constraint prevents an elongated symmetry region from doubling back on itself. It also means

Figure 15: Because normals must point inwards, an elongated region cannot double back on itself (left) and an undulating boundary is divided into a series of regions (right).

that an undulating boundary tends to be decomposed into a series of round regions on alternating sides of the boundary (figure 15).

Simple 3D symmetry regions fall into three rough categories. On a round object, most patches on the surface are approximately tangent to a common sphere. If the object is (approximately) a surface of rotation, the surface along each cross-section curve is (approximately) tangent to a sphere. Furthermore, if the surface is smooth, this sphere must also be approximately tangent to the surface at points on nearby cross-sections.[14] Finally, on a flat object, most symmetry spheres are tangent to the surface at only two points. I will call a set of points X a *symmetry patch* if there is a sphere S such that the surface is approximately tangent to S at all points of X and all the surface normals point inwards.

4.4 When will a symmetry region be visible?

Unlike an individual 3D symmetry pair, a symmetry patch may be visible from a wide range of viewpoints. For example, consider a cross-section C of a surface of rotation. Assume that the points of C are elliptic, that C is contained in the camera's field of view, and that it is not occluded by a distant part of the surface. The points of C form a circle W on the symmetry sphere, of angular radius α. The rays from the viewpoint are tangent to the symmetry sphere along another circle Y. If the symmetry sphere subtends an angle of 2β on the viewing sphere, then the radius of Y is $90 - \beta$ degrees.

The cross-section C will generate a symmetry on the 2D image if W and Y intersect in at least two points. This will happen if the center of W lies in the band of radius α around Y. Using the standard formula for the surface area of a zone of a sphere (e.g. [5]), we can calculate the surface area covered by this band as $2\pi h$, where

$$h = \min(1, \sin(\beta + \alpha)) - \max(-1, \sin(\beta - \alpha))$$

The fraction of the sphere covered by this band, i.e. the fraction of all viewpoints from which C is visible, is $\frac{h}{2}$. If α is large (the cross-section radius is not changing very fast along the axis of the region) and β is large (the object subtends a small visual angle), then the symmetry will be visible

[14] This band is widest when the curvature of the surface parallel to the axis of rotation is the same as the curvature of the symmetry sphere.

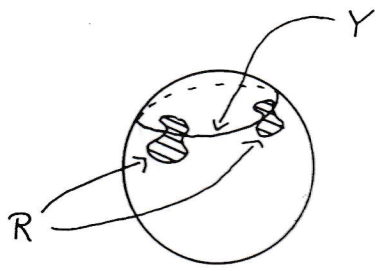


Figure 16: A symmetry patch is visible if the area R it covers on the symmetry sphere intersects the circle Y of points tangent to rays from the viewpoint, and the intersection contains at least two points with sufficient angular separation.

a large percentage of the time.[15]

It is sufficient to see two points from a symmetry patch, because this is enough information to reconstruct the patch's 3D symmetry sphere up to the usual size/distance ambiguity. Specifically, we first construct the symmetry circle from the two points and their normals. This circle is the image of the symmetry sphere on the viewing sphere. As we saw in section 3.2, we can then find the relation between the size of the symmetry sphere and its distance. So, I will say that a symmetry patch is visible if any pair of points (A, B) from the patch is visible.

It is beyond the scope of this paper to quantify visibility for other types of symmetry patches. To see how such an analysis might proceed, suppose that G is a symmetry patch and imagine the points of G as lying on the symmetry sphere (figure 16), covering some region R. To determine how often G will be visible, we consider the intersection of R with all circles Y with diameter at least $180 - \alpha$ degrees, where α is the camera's field of view.[16] The larger an area of the sphere covered by R, the more often Y will intersect R in two points. To generate useful symmetries, R must also contain subsections that are separated by a sufficient angular distance on the sphere. If the two symmetric points in the image are separated by too small an angle on their symmetry circle, the parameters of the symmetry circle cannot be reconstructed reliably from edge information in the presence of noise. Thus, a cross-section of an SHGC[17] will generate a symmetry from many viewpoints if it contains two extended sets of points approximately tangent to a common circle W and separated by a wide angle on W (e.g. the ends of an elliptical cross-section, if the ellipse is not too elongated). On the other hand, the symmetries of flat objects are not typically visible because each symmetry patch occupies only two tiny patches on the symmetry sphere.

To generate an extended symmetry region on the viewing sphere–one which is easily distinguished from a random alignment of edges–the 3D surface must contain either (a) an extended near-spherical region or (b) a long connected set of symmetry patches, each of which is visible from many viewpoints. Surfaces with symmetry regions visible from many viewpoints include:

- Spheres, partial spheres, and ellipsoids,
- Surfaces of rotation,

[15] I'm side-stepping the issue of what measure one should put on the space of all viewpoints. In particular, it isn't at all clear how one should rate the relative importance of nearby views versus distant views, nor whether one should assume that viewing distance can be made arbitrarily large.

[16] If Y is larger than this, the symmetry sphere cannot lie entirely within our image.

[17] Straight homogeneous generalized cylinder.

- Surfaces formed by sweeping a sphere along a space curve, with or without smooth changes in the sphere's radius,
- A SHGC whose cross-section meets the conditions described in the previous paragraph, and
- Complex objects formed from the above components.

Surfaces approximately matching one of these models (e.g. a finger, a standard hexagonal pencil) will generate symmetries that persist over a range of viewpoints but with variation in parameters. For flat objects and other types of SHGC's, other techniques are more appropriate: see [21, 47, 51, 54, 58, 59, 60, 65].

4.5 Symmetries in object recognition

Conversely, an extended symmetry region with a non-trivial pattern of changes in circle parameters[18] is unlikely to arise by accident. Thus, an observed 2D symmetry region can be treated as good evidence of a 3D symmetry region. From any 2D symmetry region, we can reconstruct the angular position and angular radius of each 3D symmetry sphere in the corresponding 3D symmetry region. Uncertainty about the shape of the rest of the symmetry patch, together with the depth/size ambiguity, implies that every 2D outline corresponds to an infinite-dimensional space of 3D objects. However, if we restrict our attention to a limited class of 3D objects, the ambiguities can be restricted and the symmetry information converted into useful features for recognition.

Suppose, for example, that our objects are known to be surfaces of rotation. Except where the surface approximates a sphere locally, each two-dimensional symmetry set contains two edge points belonging to the same cross-section. Once this correspondence is established, we can convert the outline to perspective coordinates and use the method in [22, 43, 65] to derive projective invariants of the outline. The local symmetry algorithm would then replace the narrow-angle approximations used in [43, 64] to locate points on the same cross-section of a generalized cylinder with circular cross-section and locate the axis of the region.

As we saw in section 3.2 and figure 6, however, using projective invariants can involve some loss of information about 3D shape. To avoid this, we can directly construct a canonical representation of the shape. First, locate a collection K of symmetries whose symmetry circles are centered on a common great circle of the viewing sphere (a common line in perspective coordinates). For example a straight axis in a symmetry region[19] or a set of adjacent regions could be found using a RANSAC-type algorithm [52]. Also choose an ordered pair of two distinguished symmetries P and Q from K (e.g. the first and last symmetries, two pairs sharing a bitangent line, two locations where the symmetry parameters change abruptly). In general, there will not be a unique choice for P and Q, so descriptors may need to be computed for several pairs.

Without loss of generality (just rotate the viewing sphere), we can assume that P's symmetry circle is centered at $(0, 0, 1)$ and that the center of Q's symmetry circle lies in the plane $x = z = 0$. We have two free parameters, the radius r of P's symmetry sphere and the ratio s of the radius of P's symmetry sphere to that of Q's symmetry sphere. These cannot be determined from a single image prior to model matching. To collapse the first degree of freedom, set $r = 1$, equating all objects which differ only in their size. Then collapse the second degree of freedom by setting $s = 1$, equating objects which look like foreshortened versions of one another.

[18] Specifically, because straight line segments are so common in man-made environments, a single region with straight non-parallel sides is not, by itself, useful in object recognition.

[19] A single region may have more than one axis. See figures 18-19.

Using the construction from section 3.2, we now have 3D coordinates for the centers of the symmetry spheres at P and Q. This gives us the 3D axis of symmetry. For each symmetry in K, we can then reconstruct the 3D location and radii of its symmetry sphere and then the radius of the corresponding circular cross-section perpendicular to the axis of symmetry. We now have a parameterized description of the 3D surface, from which we can compute any descriptors we like (cf. [54, 65]). This surface will typically consist of several connected components, separated by gaps where one cross-section occluded another in the original image. These occlusions often cause internal boundaries and/or cusps in the occluding boundary, so that the surface may have already been divided into several symmetry regions by earlier processing.

An advantage of this analysis over previous methods such as [22, 43, 64, 65] is that it extends in obvious ways to cases where other types of additional information are available to resolve the depth/size ambiguity. For example, we can remove the restriction that the axis be straight if we know that the axis is planar or the object is lying on a plane. If the parameters of the plane are known (e.g. the floor), the ambiguity is entirely resolved. Thus, if a snake crawls onto the floor of my kitchen, I can confidently determine how large it is and whether it has eaten recently. If it is lying on a surface that is horizontal but at unknown height (a table), there is a one degree of freedom ambiguity. If the orientation of the planar surface is also unknown (e.g. a cutting board propped on a book), we can still solve for the shape, but with a three degree of freedom ambiguity.

5 Finding symmetries in images

The definition of local symmetry given above consists of two distinct components: tangency to a circle and direction of normals. Because circles on the viewing sphere project to circles in a stereographic image, pairs meeting only the first condition (see the definitions in e.g. [25]) can be detected using only stereographic coordinates. If both conditions are imposed (as in [12, 15, 17, 29]), however, it is necessary to test for both circularity and straightness. Previous algorithms have done both tests in one image coordinate system (typically perspective), which is accurate only for narrow-angle images. In this section, we will see that symmetries can be computed for wide-angle images using a combination of stereographic and perspective coordinates.

Historically, local symmetries and similar low-level grouping operations (e.g. bitangents, edge segments with parallel tangents) have proved very difficult to compute. The full set of symmetries (or bitangents, or parallel segments) is clearly $O(n^2)$ worst-case: consider a set of many tiny circles of unit radius, no three centers colinear. Thus, algorithms which compute all symmetry pairs or bitangents [6, 12, 29, 36, 43, 45] are at least $O(n^2)$ worst-case.[20] It has been suspected for some time [15, 17, 19] that the complexity could be reduced by imposing additional constraints, so as to eliminate unwanted long-range symmetries, but no algorithms have achieved this cleanly. In addition, many algorithms are slow [17], complicated [17, 55], have a structure which makes running time analysis difficult [29, 45] (compare also [10]), or do only very early stages of the symmetry computation [56]. In this section, we will see that symmetry pairs can be found efficiently by approximating the edges with near-straight sections (a simplified version of the curvelinear approximation in [12]) and by incorporating a local aspect ratio constraint.

[20] The Hough transform algorithm in [43] is not linear in the number of edge pixels, because of its interpolation for high-curvature points.

5.1 Contour segmentation

The new local symmetry algorithm runs on the output of our Marr-Hildreth edge finder [18, 30]. The edge finder supplies connected chains of edge pixels, parameterized by arclength, together with normal vectors and signed intensity slope across the edge. Each edge is stored as an array. Each object boundary generates two edges, with normals pointing in opposite directions. Edges shorter than 20 pixels are ignored. Orientations are smoothed by convolving the orientation unit vectors with a Gaussian of standard deviation 4.0 pixels. (The scale of the representation can be changed by varying this standard deviation, as in [12] and/or the amount of smoothing used by the edge finder.)

The edges are computed from an image corrected to stereographic projection. At each edge location, the equations in section 3.5 are then used to compute perspective coordinates. Because our camera's field of view is not very large (90 degrees), a single perspective coordinate system is sufficient. For very wide-angle images, it may be necessary to compute coordinates in several perspective projections. Symmetry analysis for a pair of points would then choose a projection in which both points lie on the front side of the viewing sphere. Figures 9 and 10 show two sets of edges in both coordinate systems.

Each edge is then subdivided into segments that are nearly straight in both perspective and stereographic coordinates. Each edge is first divided into segments, each of which spans a 30 degree range of orientations in stereographic coordinates. When edge orientation changes rapidly, there may be multiple segments containing the same pixels but having different orientation ranges. Adjacent segments overlap by two the first and last segments in an edge may overlap more substantially. If any points on a segment are more than 10 pixels from the line joining its endpoints in perspective coordinates, the segment is recursively subdivided at its midpoint until this is no longer the case.

Each segment is decorated with various useful information:

- A unique integer ID.
- The ID numbers of the preceding segment and following segment.
- Its orientation range, with ± 5 degrees added for noise tolerance, stored as two unit vectors.
- Its *search length*, i.e. the maximum of its length and the lengths of the preceding and following segments.
- Its endpoints in perspective and stereographic coordinates, and the middle point along the segment in stereographic coordinates.
- The maximum deviation between a location on the segment and the line joining its endpoints in perspective coordinates.

Segments shorter than 5 pixels (e.g. in the tips of sharp corners) are not used in symmetry computations. This breaks the chain of pointers to preceding and following segments, implying that symmetry regions terminate when the curvature becomes very high.

5.2 Testing for symmetry

To determine whether two segments have a local symmetry and to isolate the symmetric subsections, each segment is approximated by the line segment joining its endpoints. Two tests are performed:

- Along what subsegments do the segments face one another (in perspective coordinates)?
- Along what subsegments do they have a reflectional symmetry (in stereographic coordinates)?

The subsections meeting both tests are returned as symmetric.

Extracting the facing subsections is straightforward computational geometry. There are only two subtleties. First, since segments are too short to measure curvature reliably, every segment is considered to face itself. Second, each segment may curve away from its approximating line segment. Therefore, to assert that an endpoint of one segment is on the correct side of the other segment, one must ensure that the endpoint is sufficiently far away from the segment on the correct side, compared to the maximum deviations for the segments.

To understand the symmetry test, first suppose that each segment has exactly one orientation. The direction of the symmetry axis can be computed as the average of the two orientations. The symmetric sections are computed by projecting both segments onto a line in the axis direction, taking their intersection, and backprojecting the intersection onto each segment. Since each segment actually covers a 40 degree range of orientations, compute the extreme clockwise and counter-clockwise possible axis orientations. For each extreme orientation, compute the symmetric sections. For each segment, the connected union of the section from the clockwise axis and the section from the counter-clockwise axis contains all points with a possible symmetry to the other segment.

This test will not work if the segments have overlapping orientation ranges. If the segments are identical or overlapping, they are assumed to be symmetric along their entire length. If the segments A and B are disjoint, the algorithm computes the range R of possible orientations of line segments joining a point on A to a point on B. It then reflects the orientation range of A in each extreme orientation of R and takes the connected union T of the two resulting ranges. A and B have a possible symmetry (as a whole; the algorithm does not attempt to extract symmetric subsections) if T intersects the orientation range of B.

Because all points in a segment are collectively assigned a 40 degree range of orientations, this algorithm computes an over-estimate of the set of symmetry pairs. This ensures that symmetry pairs can be assembled into connected regions in the presence of image noise or shape irregularity. It also provides initial estimates of symmetries very quickly. However, shape algorithms may wish to refine its results by post-processing, e.g. subdivide each segment into sections with smaller orientation ranges and recompute the symmetries or adapt the algorithms in [29, 45] to refine the pairing.

5.3 Symmetry parameters

If the segments have a local symmetry, the algorithm then computes the parameters of their symmetry circles using stereographic coordinates. Each edge segment is approximated by the two straight line segments joining three points: the two ends of the segment and its midpoint. From these six points, the center and radii of three symmetry circles are computed. The full set of symmetry circles can be estimated by linearly interpolating between the first and second, also the second and third, circle estimates. For each pair, we also compute its *normalized length*, i.e. the average length of the sides divided by the average symmetry circle radius.

This algorithm estimates the parameters of the symmetry circle, because the symmetry circle in stereographic projection is the image of the symmetry circle on the viewing sphere. Previous local symmetry algorithms have concentrated on computing the axis of an elongated region or the center of a round region. The axis point for two symmetric edge points A and B can be defined as the center of the tangent circle [7, 8, 13, 24, 17], the midpoint of AB [11, 12], or the midpoint of the circular

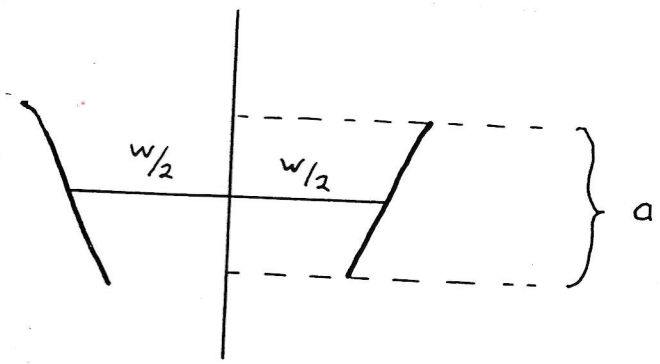


Figure 17: The local aspect ratio of this symmetry pair is $\frac{a}{w}$.

arc from A to B [41, 42]. In general, none of these points is the image of the corresponding point on the viewing sphere. However, existing methods for high-level descriptions of symmetry regions, reuniting regions separated by occlusions, building subpart structures and the like [3, 15, 29] could be reworked using the circle parameters.

Not all symmetric pairs of segments generate reliable estimates of the symmetry circles. For example, the segments may be too short to produce reliable orientations. If the segments are too close to parallel (less than 60 degrees difference between the orientations of the left and right sides),[21] the center of the circle cannot be located reliably. Such *non-supporting* symmetries are useful as connectors when building symmetry regions, particularly circular or semi-circular regions. However, a symmetry region cannot be entirely composed of non-supporting pairs.

5.4 Local aspect ratio

Suppose that we search for symmetries to a fixed segment A. The larger a neighborhood of A we search, the greater the chance that a symmetry to A will occur by accident. Furthermore, symmetries between short, widely-separated segments do not seem salient to human observers.[22] Therefore, local symmetry algorithms [15, 17] require symmetry regions to have a good *aspect ratio*, i.e. that they be long relative to their width.

The new algorithm uses a *local aspect ratio* constraint to avoid computing most of these unwanted symmetries. Specifically, the local aspect ratio of two symmetric segments is

- the length of the shorter segment, projected onto the axis of the symmetry, divided by
- the average distance between corresponding points of the two segments

(figure 17). In the current implementation, the aspect ratio of each local symmetry is required to be at least $M = \frac{360}{30\pi}$. Since each segment spans a 30 degree range of orientations, this is the aspect ratio of segments in a circular region. This constraint subsumes the curvature constraint used in [12].

[21] In a region with parallel sides, the orientations are opposite because both normals point inwards.

[22] This is not a subtle judgement. Axes generated from such symmetries look like random algorithm mistakes. It takes some effort to convince a researcher working in another area of computer science that these are symmetries of the input edges.

A symmetry pair can pass the aspect ratio test in two ways. The symmetry may pass because the actual lengths of its two segments are large enough. Alternatively, it may pass if the search lengths of its segments (see definition in section 5.1) are used in place of the segment lengths. Such symmetries are allowed because the decomposition of an edge into straight segments occasionally creates short segments at the ends of otherwise acceptable regions. These symmetries are used to connect other symmetries or extend symmetry regions, but they are considered non-supporting.

5.5 Bucket retrieval

The local aspect ratio test is used to limit the search for symmetry pairs. The new algorithm considers only symmetry pairs which have some hope of passing the aspect test, reducing the worst-case number of pairs considered to $O(L)$, where L is the length of the input edges. Without loss of generality, we can find each pair of segments by searching a neighborhood of the segment with shorter search length. Suppose that the shorter segment A has search length s. To satisfy the aspect ratio test, some point on the second segment must be $\leq Ms$ pixels from some point on A. Distinct edges do not overlap and the edge finder limits the density of edge pixels.[23] Therefore, the number of segments with length $\geq s$ within Ms pixels of A grows linearly with s.

Suppose that all segments are disjoint and s is the actual length of each segment. Then by integrating this limit over all segments A, we see that the total number of symmetry pairs passing the aspect ratio test is $O(L)$. The overlap between adjacent segments adds, at worst, a multiplier of two. The difference between search length and segment length multiplies the estimate, at worst, by three. So neither of these considerations affects the asymptotic complexity.

These pairs can be located efficiently using a pyramid structure and a bucketing algorithm similar to that in [2]. Let $T(k) = 10(\sqrt{2})^k$ for $k \geq 0$ and $T(-1) = 0$. At each level k of the pyramid, the image is divided into non-overlapping square buckets of width $(M + 0.5)T(k)$. Level k will be used to find pairs for segments whose search length is between $T(k-1)$ and $T(k)$. A segment A of search length s is stored into each level k for which $T(k-1) < s$. Whenever A intersects a bucket X, A is stored into X and its eight neighbors. This implies that A is put into a number of buckets proportional to its search length. Thus, the total work to put all segments into buckets is $O(L)$.

For each segment A of search length s, we retrieve the segments from the bucket containing the midpoint of A, from level k where $T(k-1) < s \leq T(k)$. This set contains segments from a wider area than A's ideal search neighborhood, so we must check that each candidate pair actually satisfies the aspect ratio constraint. However, the set of candidates differs from the idea by only a constant factor, so the difference does not affect the asymptotic running time. To avoid computing the same symmetry twice, we reject any candidates that are shorter than A and any candidates of the same search length whose ID number is less than A's.

5.6 Region formation

When a symmetry pair is detected between parts of two segments, new segments are created for the symmetric subsections. These shortened segments inherit the ID of the original segment,[24] as well as the ID numbers of the next and previous segments. The new pair of segments is stored in a

[23] Even using inter-cell boundaries the digitized representation prevents you from packing more than 4 pixels of boundary length per image pixel. Fine-scale edge finders typically pack at most 1 unit of boundary per two image pixels, due to e.g. the Nyquist limit. Edgefinders run with larger smoothing scales [26] or texture edgefinders (e.g. [20]) may force adjacent boundaries to be separated by many pixels.

[24] This is important; the following algorithm will not work if a new ID is assigned.

hash table, indexed by the list of ID's of its segments. The pair is also stored in reverse order. Two symmetry pairs are considered connected if their left segments overlap and their right segments also overlap. This can be true only if the left segment of the first pair has the ID number given in the next or previous field of the left segment of the second pair, and similarly for the right segments. Using the hash table, we can retrieve the (at most 4) pairs connected to a pair (A, B) in expected linear time.

Each symmetry region is a maximal connected set of symmetries which contains at least one supporting symmetry. The symmetry pairs (A, B) and (B, A) are not regarded as distinct in the definition of a region, but the region-finding algorithm represents them as distinct. To find the regions, we iterate through all supporting symmetries. For each supporting symmetry (A, B), we check whether it is still in the hash table. If so, we remove its connected component. If this doesn't contain (B, A), we also remove the connected component for (B, A) and return the two components together. This algorithm is linear in the number of symmetry pairs.

The connected component algorithm is based on a function that retrieves the maximal connected set of pairs with first element X, including a given symmetry (X, Y). I will refer to such a set as a *rib*. The connected component starting at (A, B) is represented as a binary tree, where each node is a rib. The left (right) child of a rib with left side X is a rib whose left sides is the segment preceding (following) X. Within a rib, symmetries are listed in order of their second elements. The connected component starting from (A, B) and the one starting from (B, A) represent the same region, with the same unordered symmetry pairs, but organize the symmetries into different rib structures.

The rib structure can be used to compute basic shape properties. For example, we can define the normalized length of a rib to be the maximum of the normalized lengths of the symmetry pairs in it. The normalized length of a region is then the sum of the normalized lengths over all ribs (from both halves, if the region consists of two connected components). This is closely related to the aspect ratio used in [15] but is incremental: adding more pairs to a region always increases its normalized length. The algorithm filters the raw connected components and returns a region only if its normalized length is at least 1.0 and both of its sides have lengths at least 20 pixels. This post-filtering can, of course, easily be tuned to the requirements of the user.

5.7 Results

Figures 9-10 show the symmetries computed for the edges in figures 18-19. For display purposes, each symmetry circle is represented by its center location. To make the display easier to understand, symmetries have been sorted by the contrast of their edges: both sides dark, both sides light, various hybrid cases. Because all the objects happen to be lighter than the background in these examples and there are few internal boundaries, this approximately separates the object regions from the background regions. The symmetry algorithm has successfully marked the axes of elongated regions and the centers of round regions. It has grouped the edges into a small number of connected regions, suitable for computing region descriptors, assembling into complex regions, and identifying objects.

The algorithm is worst-case expected time $O(L+A+N)$ where L is the length of the input edges, A is the area of the smallest rectangle containing these edges, and N is the number of segments. The number of symmetries is proportional to the density of edges in the image, so it should ideally be combined with a texture edge finder that guarantes a substantial spacing between adjacent edges as in [20, 38, 39]. The code is written in Lucid Common LISP and not heavily optimized. Our IBM RS6000 workstation requires 18 seconds to process the example in figure 9 (417 segments, 3483 symmetry pairs) and 16 seconds for example in figure 10 (345 segments, 2319 symmetry pairs).[25]

[25] These times do not include the time required to correct images using camera calibrations or run the edge finder.

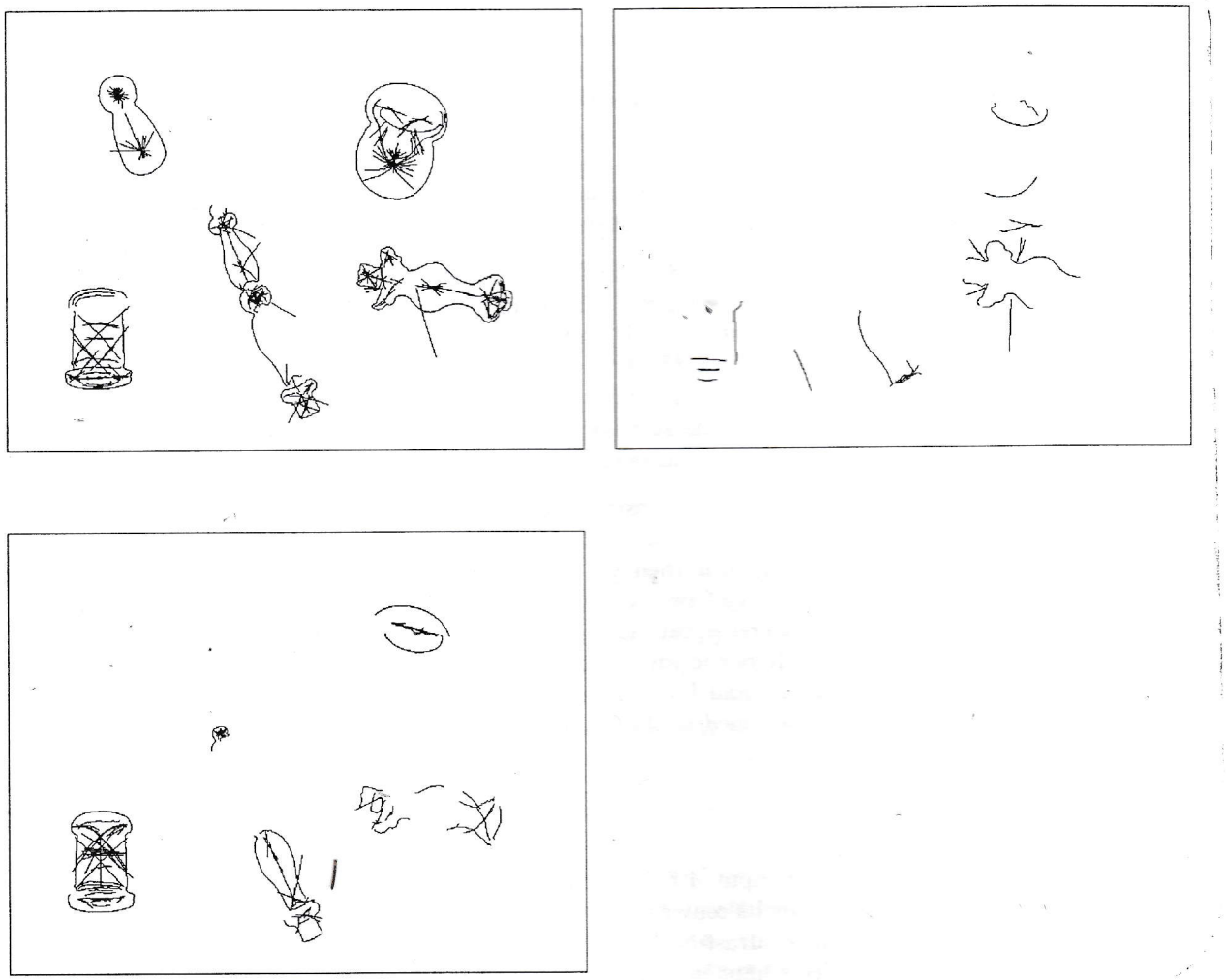


Figure 18: The 71 symmetry regions computed for the first example. The edges and the centers of the symmetry circles are shown for each region. These symmetry centers include (only approximately, see section 5.3) the centers of round regions and the axes of elongated regions and corners.

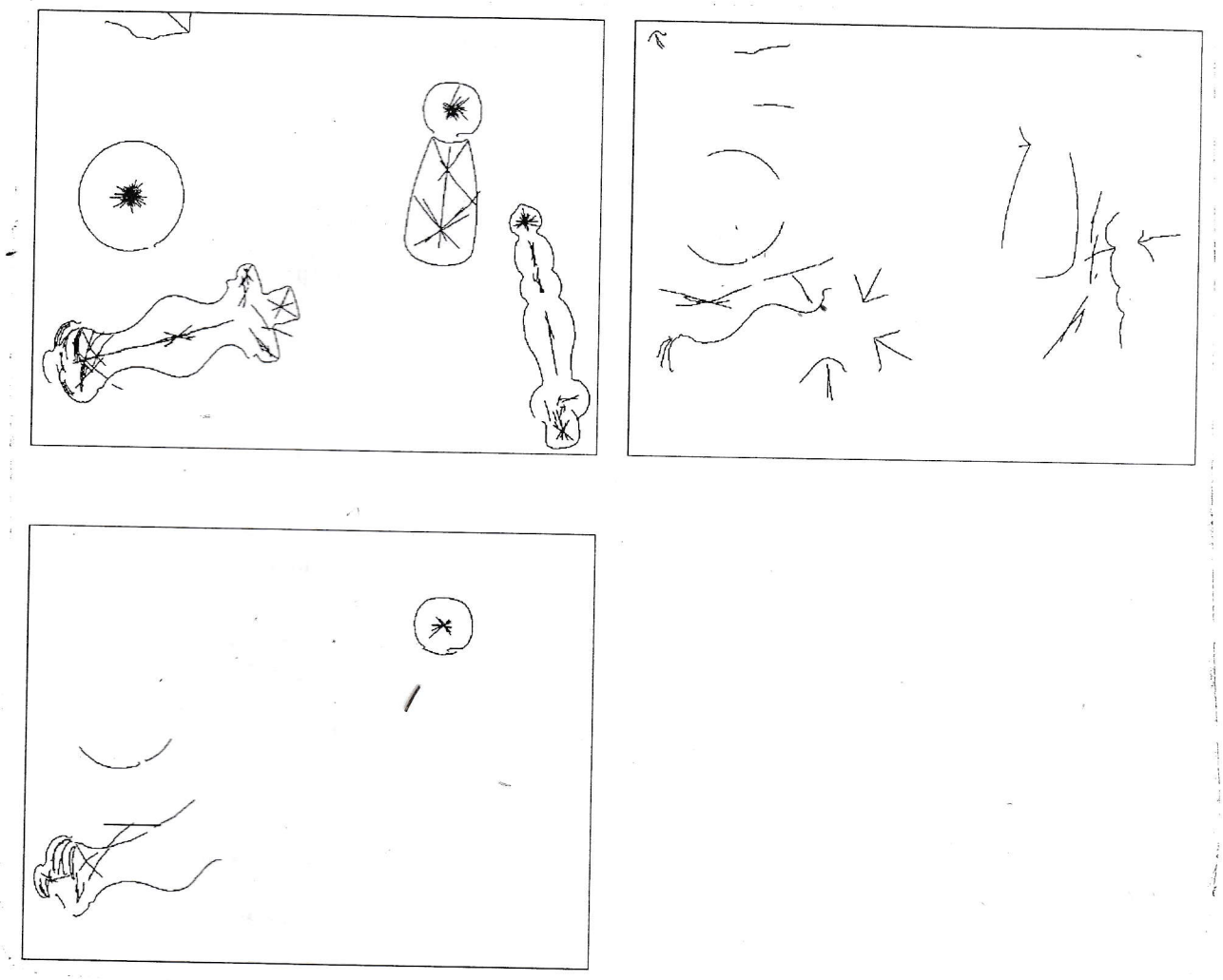


Figure 19: The 48 symmetry regions computed for the second example.

The new local symmetry algorithm has a slight asymptotic advantage over the local convexity grouping algorithm in [61], though they seem to run at about the same speed for the input sizes typical in vision applications. Unlike local convexity grouping, curvature sign descriptions (e.g. [34, 46, 53]), or corner detection (e.g. [1]), local symmetries can relate pairs of segments that are not adjacent. The new algorithm is substantially faster (for large inputs) than the $O(n^2)$ algorithms used to compute bitangents [36, 43, 51], local symmetries [12, 15], and skew symmetries [47]. The change in asymptotic complexity between [12, 15] and the new algorithm is primarily due to the local aspect ratio constraint.

6 Conclusions

We have seen three things in this paper. First, spherical projection represents, in a uniform manner, all the information available in a calibrated image. This makes it a convenient imaging model for theoretical analysis of shape algorithms. In particular, abstracting away from perspective projection helps in analyzing properties which may be distorted in flat images and in assessing how much information (section 3.2) is lost when the camera is not calibrated. We have seen that exact relations between 3D and 2D exist not only for properties derived from straightness, but also for properties derived from circularity and for hybrid properties such as local symmetries.

Second, there are several distinct ways to project the spherical image onto a plane, each preserving different properties (straightness, circularity, uniform lighting). For wide-angle images, stereographic projection may be a better general-purpose model than the traditional perspective projection, because it is conformal, distorts intensity values less, is at least as accurate a model of available wide-angle lenses, and can represent very wide-angle views ($\geq$ 180 degrees).

Finally, we have seen how local symmetry representations can be computed from a calibrated image using a combination of stereographic and perspective coordinates. The algorithm is asymptotically linear, fast in practice, and produces only a small number of output regions. Thus, local symmetries could be used in the early stages of shape analysis, supplementing features such as bitangent lines, corners, and zeros of curvature.

Acknowledgements

The author would like to thank Michael Covington, David Forsyth, Steve Osborne (of Electronic Video Systems), and Dan Stevenson for useful pointers and discussion.

Appendix A: Camera calibration

Camera calibration is required to convert image coordinates to coordinates on the viewing sphere. Researchers may be discouraged from calibrating their cameras because they believe it is difficult. For example, the methods of Tsai and Lenz [40, 57] require high-precision calibration targets and/or micrometer stages. However, for low-precision applications (e.g. many object recognition systems), it is sufficient to remove visible, gross distortions from the image. I will present a simple, low-tech procedure for doing this, to ensure that no one thinks they have an excuse for tolerating gross errors in images (e.g. visible radial distortion) when they have physical access to the camera.

To prepare the pictures in this paper, I used estimates of the intrinsic camera parameters: f, g, image center, radial distortion, and intensity drop-off. Camera manufacturers attempt to put

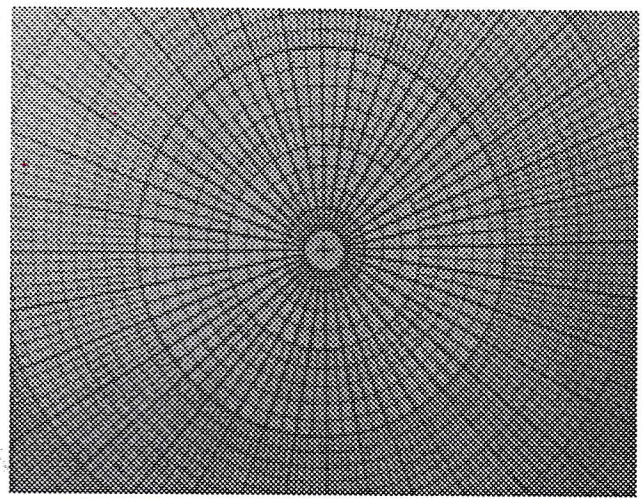


Figure 20: An image of radial graph paper used to calibrate radial distortion.

the image center in the middle of the digitized image. Tsai's results[40, 57] suggest that the image center can be relatively far (e.g. 25 pixels) from this ideal position. However, his analysis suggests that the errors in 3D measurement created by the misplaced center are no worse than those created by edge finder errors (digitization, shadows, smooth shading). Therefore, I simply used the center of the image.

From the standard proportions of a video image (4:3) and the dimensions of the image produced by the framegrabber (505 by 470), the ratio of g to f should be about 1.24. The same value (± 0.01) was obtained by placing a table-tennis ball in the center of the field of view and computing its elongation (using moments). The field of view (which determines f) was given by the lens supplier as 93 by 72 degrees. I confirmed this by measuring the apparent size of a planar set of hand-ruled markings,[26] held perpendicular to the viewing direction at a known distance from the lens. The estimates were very close (within 2 degrees).

This left the radial distortion parameters to be calibrated from scratch. A sheet of radial graph paper, held flat and perpendicular to the viewing direction, was photographed approximately centered in the digitized image (figure 20). I marked (by hand) the center of the pattern and four points at each of two rings (one near the corners of the image, one intermediate), as well as the center of the pattern. Radial distortion was modelled by an odd symmetric, 5th order polynomial (as in Tsai). The two distortion parameters[27] were computed (by direct algebra) from the average distance from the center to each ring. Figure 2 shows that this calibration removes most of the curvature from straight lines.

Finally, for some applications, it may be useful to calibrate the drop-off in intensity between the image center and the edges. As discussed in Section 2.4, wide-angle perspective lenses suffer substantial changes in illumination between the center of the image and the periphery. To measure the variation in illumination, a uniform intensity field must be projected onto the lens. I used images of a blank, evenly illuminated piece of paper taken through a diffuser (cut from a translucent plastic jug) over the lens.[28] While professional equipment would be desirable, even this procedure reveals that images from our narrow-angle lens have substantial variation in illumination level. The lens

[26] The markings on rulars aren't readable in our digitized images; black magic marker lines are.

[27] The polynomial has three coefficients, but one degree of freedom determines the size of the output image.

[28] In retrospect, a light table might provide more uniform lighting.

used for the figures in this paper, however, has substantial barrel distortion and, thus, its output has nearly uniform illumination even before calibration.

Appendix B: Special cases for symmetry definitions

The local symmetry definitions given in section 4.1 do not handle the case where the two edge or surface points are the same. These degenerate pairs are important because their presence or absence affects the connectivity of the space of symmetry pairs, which changes the qualitative structure of which connected regions are created (as in section 4.3). A partial round region (figure 14, bottom right) contains one connected set of symmetry pairs if all edge points symmetric to themselves. If they are not, it contains two connected sets, one consisting of the symmetry pairs with the first point clockwise of the second and one containing the rest. On the other hand, if every point forms a local symmetry with itself, near-round symmetry regions will not terminate when the curvature doubles back on itself.

We can define when a pair (A, A) has a local symmetry using an infinitesimal model of space [33]. I will say that a point A on the viewing sphere has a local symmetry with itself if it has a local symmetry with some point B infinitesimally close to A. This is equivalent to saying that A has a two-point contact with a (non-great) circle and the normal at A points into the circle. If you orient the boundary with the normal pointing left, this happens whenever the curvature at A is positive.

Similarly, a surface point A has a local symmetry with itself if it has a local symmetry with some point B infinitestimally close to A. In this case, A may have a variety of local symmetries with distinct symmetry circles, depending on the direction of B relative to A. The symmetry between A and B projects onto a symmetry of the outline on the viewing sphere if both A and B are on the boundary of the object (the rays from the lens center are tangent to the surface at both A and B).

This is one possible way to handle these degenerate pairs, but not the only way. It is beyond the scope of this paper to determine whether it is the best.

References

[1] Asada, Haruo and J. Michael Brady (1986) "The Curvature Primal Sketch," *IEEE Transactions on Pattern Analysis and Machine Intelligence* 8/1, pp. 2–14.

[2] Ayache, Nicholas and Bernard Faverjon (1987) "Efficient Registration of Stereo Images by Matching Graph Descriptions of Edge Segments," *International Journal of Computer Vision* 1/2, pp. 107–131.

[3] Bagley, Steven (1980) "Using Models and Axes of Symmetry to Describe Two-dimensional Polygonal Shapes," MS thesis, Elec. Eng. and Comp. Sci, Mass. Inst. of Techn.

[4] Barlow, H.B. and J.D. Mollon, eds., *The Senses*, Cambridge Univ. Press, Cambridge, 1982.

[5] Beyer, William H., ed. (1984) *CRC Standard Mathematical Tables*, 27th edition, CRC Press, Boca Raton, FL.

[6] Blake, Andrew, Michael Tayloer, and Adrian Cox (1993) "Grasping Visual Symmetry," *Proceedings of the International Conference on Computer Vision* 1993, pp. 724–733.

[7] Blum, Harry (1973) "Biological Shape and Visual Science (Part I)," *Journal of Theoretical Biology* 38, pp. 205–287.

[8] Blum, Harry and Roger N. Nagel (1978) "Shape Description using Weighted Symmetric Axis Features," *Pattern Recognition* 10, pp. 167–180.

[9] Robert C. Bolles, H. Harlyn Baker, and David H.Marimont (1987) "Epipolar-Plane Image Analysis: An Approach To Determining Structure from Motion," Intern. Journ. of Computer Vision 1/1, 7–55.

[10] Bookstein, Fred L., "The Line Skeleton," Computer Graphics and Image Processing 11 (1979) 123–137.

[11] Brady, J. Michael (1983) "Criteria for Representations of Shape," in Jacob Beck, Barbara Hope, and Azriel Rosenfeld, eds., *Human and Machine Vision,* Academic Press, New York, pp. 39–84.

[12] Brady, J. Michael and Haruo Asada (1984) "Smoothed Local Symmetries and Their Implementation," *International Journal of Robotics Research* 3/3, 36–61.

[13] J. W. Bruce, Peter J. Giblin and C. G. Gibson, "Symmetry Sets," Proc. Roy. Soc. Edinburgh 101A (1985) 163–186.

[14] Cipolla, Roberto and Andrew Blake (1992) "Surface Shape from the Deformation of Apparent Contours," *Intern. Journ. of Computer Vision* 9/2, 83–112.

[15] Connell, Jonathan H. and J. Michael Brady (1987) "Generating and Generalizing Models of Visual Objects," *Artificial Intelligence* 31/2, pp. 159–183.

[16] Faugeras, Olivier D. (1992) "What can be seen in three dimensions with an uncalibrated stereo rig?" *European Conf. on Comp. Vision* 1992, 563–578.

[17] Fleck, Margaret M. (1986) "Local Rotational Symmetries," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* 1986, pp. 332–337.

[18] Fleck, Margaret M. (1992) "Some Defects in Finite Difference Edge Finders," IEEE Trans. Patt. Analy. Mach. Intell. 14(3), pp. 337–345.

[19] Fleck, Margaret M. (1990) "Classifying Symmetry Sets," Proc. British Machine Vision Conf., Oxford, 297–302.

[20] Fleck, Margaret M. (1994) "Practical edge finding with a robust estimator," to appear CVPR 1994.

[21] Forsyth, David, Joseph Mundy, Andrew Zisserman, Chris Coelho, Aaron Heller, and Charles Rothwell (1991) "Invariant Descriptors for 3-D Object Recognition and Pose," *IEEE Transactions on Pattern Analysis and Machine Intelligence* 13/10, pp. 971–991.

[22] Forsyth, David, Joseph Mundy, Andrew Zisserman, and Charles Rothwell (1992) "Recognizing Rotationally Symmetric Surfaces from the Outlines," *European Conf. on Comp. Vision* 1992, 639–647.

[23] Forsyth, David, Joseph Mundy, Andrew Zisserman, and Charles Rothwell (1993) "Using Global Consistency to Recognize Euclidean Objects with an Uncalibrated Camera," to appear, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* 1993.

[24] Peter J. Giblin and S. A. Brassett, "Local Symmetry of Plane Curves," American Mathematical Monthly 92/10 (1985) 689–707.

[25] Peter J. Giblin and Donal B. O'Shea, "The Bitangent Sphere Problem," American Mathematical Monthly 97/1 (1990) 5–23.

[26] Grimson, W. Eric L. (1981) "A Computer Implementation of a Theory of Human Stereo Vision," *Philosophical Transactions of the Royal Society of London B* 292, pp. 217–253.

[27] Hilbert, D. and S. Cohn-Vossen, *Geometry and the Imagination,* Chelsea Publishing Co., New York, second edition, 1990, translation by P. Nemenyi of *Anschauliche Geometrie*, Springer-Verlag, Berlin, 1932.

[28] Hedgecoe, John, *The Photographer's Handbook*, Alfred Knopf, New York, 1993.

[29] Scott Spenser Heide (1984) "A Hierarchical Representation of Shape from Smoothed Local Symmetries," MS thesis, Dept. of Mech. Eng., Mass. Inst. of Techn..

[30] Hildreth, Ellen (1983) "The Detection of Intensity Changes by Computer and Biological Vision System," *Computer Vision, Graphics, and Image Processing* 22, 1–27.

[31] Hill, Francis S., Jr., *Computer Graphics*, MacMillan, New York, 1990.

[32] Horn, Berthold K.P., "Robot Vision," MIT Press, Cambridge MA, 1986.

[33] R. F. Hoskins, Standard and Nonstandard Analysis (Ellis-Horwood, Chichester, England, 1990).

[34] Huttenlocher, Daniel P. (1988) "Three-Dimensional Recognition of Solid Objects from a Two-Dimensional Image," Ph.D. thesis, Elec. Eng. and Comp. Sci., Mass. Inst. of Techn.

[35] Kingslake, Rudolf, *A History of the Photographic Lens,* Academic Press, San Diego, 1989.

[36] Kriegman, David J., B. Vijayakumar, and Jean Ponce (1993) "Reconstruction of HOT Curves from Image Sequences," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* 1993, pp. 20–26.

[37] Kiriakos N. Kutulakos and Charles R. Dyer (1992) "Recovering Shape by Purposing Viewpoint Adjustment," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* 1992 pp. 16–22.

[38] Leclerc, Y. and Zucker, S.W. 1987. The Local Structure of Image Discontinuities in One Dimension. *IEEE Trans. Patt. Anal. Mach. Intell.* 9(3) 341–355.

[39] Leclerc, Y. 1985. Capturing the Local Structure of Image Discontinuities in Two Dimensions. *IEEE Conf. Comp. Vis. Patt. Recogn.* pp. 34–38.

[40] Lenz, Reimar K. and Roger Y. Tsai, "Techniques for Calibration of the Scale Factor and Image Center for High Accuracy 3-D Machine Vision Metrology," IEEE PAMI 10/5 (1988) pp. 713–720.

[41] Michael Leyton, "Symmetry-Curvature Duality," Computer Vision, Graphics, and Image Processing 38 (1987) 327–341.

[42] Michael Leyton, "A Process-Grammar for Shape," Artificial Intelligence 34/2 (1988) 213–247.

[43] Jane Liu, Joe Mundy, David Forsyth, Andrew Zisserman, and Charlie Rothwell (1993) "Efficient Recognition of Rotationally Symmetric Surfaces and Straight Homogeneous Generalized Cylinders," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* 1993, pp. 123–128.

[44] Mac Lane, Saunders (1986) *Mathematics, Form and Function*, Springer-Verlag, New York.

[45] Morris, Richard (1990) "Symmetry of Curves and the Geometry of Surfaces: Two Explorations with the Aid of Computer Graphics," Ph.D. thesis, Mathematics, Univ. of Liverpool.

[46] Mokhtarian, Farzin and Alan Mackworth, "Scale-Based Description and Recognition of Planar Curves and Two-Dimensional Shapes," *IEEE Transactions on Pattern Analysis and Machine Intelligence* 8/1 (1986) pp. 34–43.

[47] Mukherjee, Dipti Prasad, Andrew Zisserman, and Michael Brady (1993) "Shape from Symmetry–Detecting and Exploiting Symmetry in Affine Images," TR OUEL 1988/93, Engineering Science, University of Oxford.

[48] Nalwa, Vishvjit, *A Guided Tour of Computer Vision*, Addison Wesley, Reading MA, 1993.

[49] Noble, Julia Alison (1989) "Descriptions of Image Surfaces," D. Phil thesis, Department of Engineering Science, Oxford University.

[50] Peterson, B. "Moose," *Nikon System Handbook*, Images Press, New York, 1991.

[51] Ponce, Jean, David Chelberg, and Wallace B. Mann (1989) "Invariant Properties of Straight Homogeneous Generalized Cylinders and Their Contours," *IEEE Transactions on Pattern Analysis and Machine Intelligence* 11/9, pp. 951–966.

[52] Fischler, Martin A. and Robert C. Bolles (1981) "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography," *Comm. ACM* 24/6, 381–395.

[53] Richards, Whitman and Donald D. Hoffman (1985) "Codon Constraints on Closed 2D Shapes," *Computer Vision, Graphics, and Image Processing* 31/2, pp. 156–177.

[54] Rothwell, Charles, Andrew Zisserman, David Forsyth, and Joseph Mundy (1992) "Canonical Frames for Planar Object Recognition," *European Conf. on Comp. Vision* 1992, 757–772.

[55] Saund, Eric (1992) "Putting Knowledge into a Visual Shape Representation," *Artificial Intelligence* 54/1–2, pp. 71–119.

[56] Scott, G. L., S.C. Turner, and A. Zisserman, "Using a Mixed Wave/Diffusion Process to Elicit the Symmetry Set," *Im. and Vis. Comp.* 7/1 (1990) pp. 63–70.

[57] Tsai, Roger Y., "An Efficient and Accurate Camera Calibration Technique for 3D Machine Vision," CVPR 1986, pp. 364–374.

[58] Fatih Ulupinar and Ramakant Nevatia (1988) "Using Symmetries for Analysis of Shape from Contour," *Proceedings of the International Conference on Computer Vision* 1988, pp. 414–426.

[59] Fatih Ulupinar and Ramakant Nevatia (1990) "Inferring Shape from Contour for Curved Surfaces," *Intern. Conf. on Patt. Recogn.*, 1990, pp. 147–154.

[60] Fatih Ulupinar and Ramakant Nevatia (1990) "Shape from Contour: Straight Homogeneous Generalized Cones," *Proceedings of the International Conference on Computer Vision* 1990 pp. 582–586.

[61] Huttenlocher, Daniel P. and Peter C. Wayner (1992) "Finding Convex Edge Groupings in an Image," *International Journal of Computer Vision* 8/1, pp. 7–27.

[62] Welford, Walter T., "Useful Optics," U. Chicago Press, Chicago, 1991.

[63] Wyszecki, G. and Stiles, W.S. 1982. *Color Science.* John Wiley, NY, second edition.

[64] Zerroug, Mourad and Ramakant Nevatia (1993) "Quasi-Invariant Properties and 3-D Shape Recovery of Non-Straight, Non-Constant Generalized Cylinders," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* 1993, pp. 96–103.

[65] Zisserman, Andrew, David A. Forsyth, Joseph L. Mundy, Charles A. Rothwell. (1992) "Recognizing General Curved Objects Efficiently," in Joseph Mundy and Andrew Zisserman, eds., *Geometric Invariance in Computer Vision* pp.232–254.