Calculation of Object Pose Constraints from Sparse, Erroneous Sensory Data [1]

Gordon Dakin
Robin Popplestone

COINS Technical Report 89-64

July 31, 1989

Laboratory for Perceptual Robotics
Department of Computer and Information Science
A305, Graduate Research Center
University of Massachusetts
Amherst, MA 01003

[1]Preparation of this paper was supported in part by grant number N00014-84-K-0564 from the Office of Naval Research.

Abstract

This report presents a method for calculating bounds on a sensed object's uncertain position and orientation. Interpreted in the context of an object model, each nominal sense datum acquired from a rigid object's surface restricts the pose of the object up to the symmetry of the sensed feature. When sensor error is present, each sense datum restricts the pose to a volume in 6-dimension pose space. If enough distinct surface features are sensed, their corresponding pose volumes intersect to form a finite uncertainty volume of plausible object poses. We approximate this irregularly-shaped uncertainty volume as a convex polyhedron by linearizing the relationship between sensor error and pose error. A set of hyperplanar pose constraints is formed to represent the polyhedron's boundary, and the maximum plausible extent of the pose along any axis in pose space may then be calculated through linear programming.

1 Introduction

Pose calculations and estimations of pose uncertainty are frequently required in robot applications. Calculation of an end-effector's position and orientation relative to task sites and obstacles is essential for the planning and execution of robot trajectories and manipulation tasks. Since the calculated locations of the manipulator and the objects in its environment typically rely on error-prone sensory data, it is often useful to characterize the spatial uncertainty associated with each pose calculation. By accommodating all plausible or likely poses that may be inferred from the noisy sense data, a robot motion or task can be planned in a more robust manner.

Pose uncertainty considerations have been incorporated into path planning among obstacles [9], robotic assembly planning [5][10], mobile robot navigation [13], and object recognition through tactile probing [6]. The related issues of choosing sensor locations [2] and deciding when to allocate additional sensors [1] have also been addressed.

One approach to representing pose uncertainty is to surround the nominal, expected pose with uncertainty bounds that contain all poses consistent with known sensory error bounds. A technique representative of this approach is Brooks' propagation of spatial errors through nonlinear constraint equations [1]. Another example of the pose-bounding approach is Fleming's propagation of dimension tolerances and pose uncertainties in a structure of interrelated parts [4]. Other approaches to characterizing pose uncertainty employ probability distributions [2][13] and possibility theory [3]. The technique presented in this report adopts the constraint-bounding approach, reformulated here as a linear programming problem.

In the paradigm of *sparse sense data* acquisition, surface data consisting of isolated points, normals, and curvature axes are acquired from an object's surface through touch, range, or visual sensing. The data acquired from sensed object features may be matched to their counterpart features in an

object model, in order to arrive at a semantic interpretation of the sense data [7]. Since there may be more than one consistent interpretation of the sensed features at any given time during the acquisition of the sense data, several tentative object identities may be hypothesized, as well as several discrete poses for a given identity.

The pose uncertainty algorithm described in this report operates on a *single* semantic interpretation of the sense data. The problem of matching features to their counterparts in the object model is not addressed here, and it is assumed that each sensed feature has been matched, either tentatively or definitively, to a counterpart in the object model. By considering only one set of correspondences between scene and model features, we avoid the complication of dealing with several disjoint uncertainty volumes in pose space.

Interpreted in the context of an object model, each sense datum acquired from a rigid object's surface restricts the pose of the object up to the symmetry of the sensed feature. Thus, if we ignore sensor error, a point and direction vector acquired from a linear edge restrict the sensed object's origin to lie on a cylinder of radius equal to the origin's known distance from the edge. If enough distinct surface features are sampled, the object's pose will be restricted to a single location, which may be calculated from the intersection of cosets of the sensed features' symmetry groups [12]. If the sense data are erroneous and redundant, no single pose is likely to be consistent with the data, so a *nominal*, or expected pose must be obtained by averaging, or by the method of least squares.

When sensor error is present, the set of plausible object poses to be inferred from each sense datum must encompass the entire range of poses propagating from the uncertain sense values. To quantify sensor error, uncertainty bounds are associated with each datum's variables, through empirical testing of the sensor. For instance, a point vertex datum may be surrounded by an uncertainty box, and a surface normal vector surrounded

by an uncertainty pyramid. Allowing for sensor error, each sense datum restricts the pose of a 3-dimensional object to a *volume* in 6-dimensional pose space. If the surface feature's symmetry is assumed infinite, the corresponding pose volume is infinite, as well. Provided that enough distinct surface features are sensed, the pose volumes intersect to form a finite uncertainty volume of plausible object poses.

In order to meet real-time computational demands, we approximate each sensed feature's irregularly-shaped pose uncertainty volume as a convex polyhedron. A nominal pose is first calculated from the combined features' sense data, to serve as a point in pose space around which to linearize the relationship between the uncertainty in a feature's sense and symmetry variables, and object pose uncertainty. A *pose Jacobian* is thus constructed for each sensed feature. The pose Jacobians are used to project the features' linear sensor error bounds into pose space, forming a set of hyperplanar pose constraints that bound a polyhedral pose uncertainty volume. The maximum possible extent of the pose along any desired axis in pose space is then calculated through linear programming.

Since linearizing the relationship between sense and pose variables approximates the relationship with increasing inaccuracy as sense variables deviate from their nominal values, the validity of this technique depends on very small sensor error.

2 An illustrative example in 2-D

This section introduces the pose uncertainty algorithm from an intuitive persepective, by way of a simple example involving a 2-dimensional object. The pose of a 2-dimensional object has three degrees of freedom (*dofs*): two for its position and one for its orientation. A 2-dimensional object's pose space is therefore 3-dimensional and easier to visualize than the 6-dimensional pose space of a 3-dimensional object. The algorithm extends to

the 3-dimensional case, which is presented in later sections.

The objective of the uncertainty algorithm is to determine the region in pose space in which a modelled object might reside, given sparse, erroneous sense data acquired from the surface of the object. The sparse sense data consist of any combination of point vertices, edge points and direction vectors, and surface points and normals. As mentioned earlier, we assume that all of the sensed features have been matched to a single set of model counterparts, so only one semantic interpretation of the sense data is considered at a time. From the relationship between the sensed points and direction vectors to their model counterparts, we obtain the nominal object pose. The uncertainty in the calculated pose arises from sensor error, which is characterized by uncertainty bounds surrounding the nominal points and direction vectors comprising the sense data.

From the perspective of a single sense datum, pose uncertainty arises from (1) the inherent symmetry of each sensed surface feature and (2) sensor error. Let us consider the first of these sources of uncertainty.

Figure 1 illustrates a polygonal object, modelled with an embedded coordinate frame that represents the object's location with respect to the global coordinate frame. The object frame's origin (x, y), together with the angle θ between the object frame's axis $\mathbf{x}_1$ and the global frame's axis $\mathbf{x}_0$, describe the three *dofs* in the object's pose. Boundary features, such as points and edges, may be described in the object model at locations specified relative to the object's embedded frame. Suppose, for example, a point vertex is sensed, returning sense datum $\mathbf{p} = (p_x, p_y)$. The vertex's location in the object model reveals its distance r from the object's origin. The point datum and its known distance to the origin may be used to constrain possible locations of the object, relative to the global coordinate frame.

Ignoring sensor error, the sense datum $\mathbf{p}$ constrains the the object's origin (x, y) to lie somewhere on the circle of radius r centered at point $\mathbf{p}$. The

object's pose is confined to a circular symmetry group of locations, which leaves one rotational *dof* unconstrained. The pose uncertainty arising from a sensed feature's inherent symmetry is referred to as *symmetry uncertainty*. The symmetry uncertainty of a polyhedral vertex feature is illustrated more explicitly in the polyhedron's 3-dimensional pose space, shown in figure 2. The object's pose lies somewhere on the helix H described by the following equations for the three *dofs* x, y, and θ:

$$f_x(p_x, p_y, \theta) = p_x + r\cos\theta$$

$$f_y(p_x, p_y, \theta) = p_y + r\sin\theta$$

$$f_\theta(p_x, p_y, \theta) = \theta$$

where pose *dof* θ doubles as the *dof* of the point feature's inherent symmetry. Without sensor error, *sense variables* p_x and p_y are constants, while *symmetry variable* θ may take on any real value, modulus 2π.

With the acquisition of additional sense data (say, another point vertex), the pose of the object in figure 1 is constrained to lie at a single point $(\bar{x}, \bar{y}, \bar{\theta})$ in pose space. For example, a point datum sampled from a different vertex constrains the pose to a second helix, which intersects the original helix H at a single pose $(\bar{x}, \bar{y}, \bar{\theta})$. As shown in figure 2, we can linearize helix H about $(\bar{x}, \bar{y}, \bar{\theta})$ to form a line L that approximates H well in the vicinity immediately surrounding $(\bar{x}, \bar{y}, \bar{\theta})$. The slope of L is obtained by differentiating f_x, f_y, and f_θ with respect to symmetry variable θ.

$$\frac{\partial f_x}{\partial f_\theta} = -r\sin\theta$$

$$\frac{\partial f_y}{\partial f_\theta} = r\cos\theta$$

$$\frac{\partial f_\theta}{\partial f_\theta} = 1$$

The helix may then be approximated as a line L by the following parametric equation over free variable λ:

$$\begin{bmatrix} x \\ y \\ \theta \end{bmatrix} = \begin{bmatrix} \bar{x} \\ \bar{y} \\ \bar{\theta} \end{bmatrix} + \lambda \begin{bmatrix} \frac{\partial f_x}{\partial f_\theta} \\ \frac{\partial f_y}{\partial f_\theta} \\ \frac{\partial f_\theta}{\partial f_\theta} \end{bmatrix}$$

The line approximation L of H is very close to the helix in the vicinity immediately surrounding $(\bar{x}, \bar{y}, \bar{\theta})$. We will exploit this linear approximation of H later on, in order to build linear pose constraints surrounding $(\bar{x}, \bar{y}, \bar{\theta})$.

With sensor error present, the sense variables p_x and p_y in the above pose functions f_x, f_y, and f_θ are no longer constant, but vary within an uncertainty range bounded by sensor-dependent error limits. For example, p_x and p_y might be confined to a rectangle:

$$\bar{p}_x - \epsilon_{p_x} \leq p_x \leq \bar{p}_x + \epsilon_{p_x}$$

$$\bar{p}_y - \epsilon_{p_y} \leq p_y \leq \bar{p}_y + \epsilon_{p_y}$$

where ϵ_{p_x} and ϵ_{p_y} denote the maximum absolute deviations of $\mathbf{p}$'s components p_x and p_y from their nominal values $\bar{p}_x$ and $\bar{p}_y$, respectively. These sensory constraints, when propagated to pose space via pose functions f_x, f_y, and f_θ, effectively constrain the pose of the sensed object to a bounded volume in pose space, with a nonlinear surface. In order to form linear constraints on the pose from the nonlinear pose functions, the pose is linearized about the nominal pose $(\bar{x}, \bar{y}, \bar{\theta})$. This is accomplished with the aid of the *pose Jacobian* J:

$$J = \begin{bmatrix} \frac{\partial f_x}{\partial p_x} & \frac{\partial f_x}{\partial p_y} & \frac{\partial f_x}{\partial \theta} \\ \frac{\partial f_y}{\partial p_x} & \frac{\partial f_y}{\partial p_y} & \frac{\partial f_y}{\partial \theta} \\ \frac{\partial f_\theta}{\partial p_x} & \frac{\partial f_\theta}{\partial p_y} & \frac{\partial f_\theta}{\partial \theta} \end{bmatrix} = \begin{bmatrix} 1 & 0 & -r\sin\theta \\ 0 & 1 & r\cos\theta \\ 0 & 0 & 1 \end{bmatrix}$$

The columns of J, denoted hereafter as as J_{p_x}, J_{p_y}, and J_θ, describe linearized deviations from the nominal pose $(\bar{x}, \bar{y}, \bar{\theta})$ that correspond to unit deviations of the sense and symmetry varables p_x, p_y, and θ from their respective nominal values $\bar{p}_x$, $\bar{p}_y$, and $\bar{\theta}$. Thus, after linearization about the nominal pose, the vectors $\pm\epsilon_{p_x} J_{p_x}$ and $\pm\epsilon_{p_y} J_{p_y}$ give the maximum deviations of the pose from $(\bar{x}, \bar{y}, \bar{\theta})$ resulting from the individual uncertainties in p_x and p_y, respectively. Recall that J_θ, the third column of J, was employed earlier to describe the instantaneous slope of the helix H in figure 2.

As shown in figure 3, the pose uncertainty volume arising from the sense and symmetry uncertainty of point datum $\mathbf{p}$ can be approximated by four constraint planes $\Gamma^+_{p_x}$, $\Gamma^-_{p_x}$, $\Gamma^+_{p_y}$, and $\Gamma^-_{p_y}$, which bound a parallelopiped-shaped, prismatic volume around line L. The linearized pose deviations $\pm\epsilon_{p_x} J_{p_x}$ corresponding to the maximum absolute error ϵ_{p_x} in p_x generate the pose constraint plane $\Gamma^+_{p_x}$ and its negative counterpart $\Gamma^-_{p_x}$. Similarly, pose deviations $\pm\epsilon_{p_y} J_{p_y}$ attributable to maximum error in p_y generate $\Gamma^+_{p_y}$ and $\Gamma^-_{p_y}$.

These pose constraint planes can be represented by normal-scalar pairs $(\hat{\mathbf{n}}, d)$ which are calculable from the components of J and the sensory error bounds. Plane $\Gamma^+_{p_x}$, for example, has normal $\hat{\mathbf{n}}_{p_x}$, which is perpendicular to J_y and J_θ (since both J_y and J_θ must lie in $\Gamma^+_{p_x}$). Hence, $\hat{\mathbf{n}}_{p_x}$ can be calculated as the cross product

$$\hat{\mathbf{n}}_{p_x} = J_y \times J_\theta$$

The scalar component d_{p_x} of plane $\Gamma^+_{p_x}$'s description is obtained by taking the dot product of any point on $\Gamma^+_{p_x}$ with $\hat{\mathbf{n}}_{p_x}$. Such a point is

$$\begin{bmatrix} \bar{x} \\ \bar{y} \\ \bar{\theta} \end{bmatrix} + \epsilon_{p_x} J_{p_x}$$

Hence,

$$d_{p_x} = \left(\begin{bmatrix} \bar{x} \\ \bar{y} \\ \bar{\theta} \end{bmatrix} + \epsilon_{p_x} J_{p_x} \right) \cdot \hat{\mathbf{n}}_{p_x}$$

The normal-scalar descriptors for the other three constraint planes may be derived similarly. Once derived, the constraint plane descriptors $\hat{\mathbf{n}}_i = [n_{ix}, n_{iy}, n_{i\theta}]^T$ and d_i will serve as linear constraints of the form

$$n_{ix}x + n_{iy}y + n_{i\theta}\theta \leq d_i$$

to be passed as arguments to a linear programming algorithm. When constraint planes generated from enough sensed features are combined (eg, two point vertices yielding eight constraint planes), the constraints will bound a finite volume in 3-dimensional pose space. The linear programming algorithm can then be invoked to determine the maximum extent of the pose along any axis in pose space.

3 Object localization in 3-D

The pose of a 3-dimensional object can be represented by six variables $(x, y, z, \phi, \theta, \psi)$, where x,y, and z describe the origin, and ϕ, θ, and ψ parameterize a composition of rotations that describes the object's orientation. In

the Euler representation, for example, the three *Euler angles* ϕ, θ, and ψ parameterize the composition of rotations

$$rot(\mathbf{k}, \phi)\ rot(\mathbf{j}, \theta)\ rot(\mathbf{k}, \psi)$$

In the remainder of this report, we will assume that the Euler representation is employed, although other schemes for representing orientation (such as *roll-pitch-yaw*) could be readily substituted.

3.1 Pose functions

Sensory information describing the local geometries of a modelled object's surfaces constrains the object's location up to the accuracy of the sense data and the symmetries of the sensed features. In section 2, for example, it was shown that sensing a vertex on a 2-dimensional object serves to constrain the object's pose in 3-dimensional pose space to a helicoid volume. The helical relationship between plausible origin positions and object orientations arises from the circular symmetry inherent in the vertex's geometry. The symmetry uncertainty combines with the uncertainty in the sensed point to form a volume in pose space.

As with the development of pose constraints for 2-dimensional objects, we begin our derivation of pose constraints for 3-dimensional objects by constructing pose functions over the sense and symmetry variables of surface features. We focus first on the translational symmetry variables.

A surface point and normal acquired from an identified face on a modelled polyhedral object constrain the object's pose up to the planar symmetry's three *dofs*, and up to the uncertainty in the point and normal values. A sensed planar surface point $\mathbf{p}$ may be expressed in local face coordinates. When the object coordinate frame coincides with the global frame, as illustrated in figure 4, $\mathbf{p}$'s global position may be described as the vector sum of $\mathbf{u}$, $d_x\mathbf{m}_x$, and $d_y\mathbf{m}_y$, where $\mathbf{u}$ is the origin of the face frame, $\mathbf{m}_x$, and $\mathbf{m}_y$

are surface tangential face frame axes, and d_x and d_y are coordinates along those axes.

Conversely, the object's origin (in this case the global origin) may be expressed in terms of the sensed point and the point's surface coordinates:

$$\mathbf{0} = \mathbf{p} - d_x\mathbf{m}_x - d_y\mathbf{m}_y - \mathbf{u}$$

When the object frame and global frame do not coincide, as illustrated in figure 5, the above equation for the object's origin generalizes to

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \mathbf{p} - d_x R\mathbf{m}_x - d_y R\mathbf{m}_y - R\mathbf{u} \qquad (1)$$

where R is a rotation matrix describing the orientation of the object. In figure 5, the rotated versions of face frame axes $\mathbf{m}_x$, and $\mathbf{m}_y$ are denoted more concisely as $\mathbf{n}_x$, and $\mathbf{n}_y$, and the rotated version of $\mathbf{u}$ is denoted as $\mathbf{v}$. Since d_x and d_y are unknowns when the surface point is sensed, we may regard them as symmetry variables, representing the two translational *dofs* allowed by the planar surface's symmetry.

We now consider the rotational symmetry variable δ_z of the planar feature. In order to express the object's orientation as a function of the sensed normal $\mathbf{n}_z$, rotation matrix R can be written as the following composition, involving a rotation by unknown angle δ_z about the sensed surface normal.

$$R = rot(\mathbf{m}_z \times \mathbf{n}_z, cos^{-1}(\mathbf{m}_z \cdot \mathbf{n}_z))\ rot(\mathbf{m}_z, \delta_z) \qquad (2)$$

The first rotation aligns the sensed normal with the model normal by rotating about their mutual perpendicular, and the second rotation comprises the third *dof* in the planar face's symmetry. The three orientation components of the pose are given by

$$\begin{pmatrix} \phi \\ \theta \\ \psi \end{pmatrix} = Euler^{-1}(R) \tag{3}$$

which together with equation (2) describes the three angular components of the pose in terms of sense datum $\mathbf{n}_z$ and symmetry varable d_z.

Implicit in the vector equations (1) and (3) are six position functions f_x, f_y, f_z, f_ϕ, f_θ, and f_ψ over the sense data $\mathbf{p}$ and $\mathbf{n}_z$ and symmetry variables d_x, d_y, and δ_z.

We may similarly derive object pose equations for linear edge features. Shown in figure 6, a surface point sampled from a linear edge of an object located at the global frame may be expressed as the vector sum of local edge frame origin $\mathbf{u}$, and $d_z\mathbf{m}_z$, where d_z is the point's ordinate along edge frame $\mathbf{m}_z$. Conversely, the object's origin is located at

$$\mathbf{0} = \mathbf{p} - d_z\mathbf{m}_z - \mathbf{u}$$

when the object and global frames coincide, and in general at

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \mathbf{p} - d_z R\mathbf{m}_z - R\mathbf{u} \tag{4}$$

where rotation matrix R is again comprised of a composition, involving a rotation by an unknown angle δ_z about the sensed edge vector:

$$R = rot(\mathbf{m}_z \times \mathbf{n}_z, cos^{-1}(\mathbf{m}_z \cdot \mathbf{n}_z))\ rot(\mathbf{m}_z, \delta_z)$$

As in the case of the planar feature,

$$\begin{pmatrix} \phi \\ \theta \\ \psi \end{pmatrix} = Euler^{-1}(R) \tag{5}$$

The vector equations (4) and (5) together describe six object pose functions f_x, f_y, f_z, f_ϕ, f_θ, and f_ψ over the sense data $\mathbf{p}$ and $\mathbf{n}_z$ and two symmetry variables d_z, and δ_z.

Finally, a point vertex sense datum $\mathbf{p}$ locates the origin of the object at

$$\mathbf{0} = \mathbf{p} - \mathbf{u}$$

when the object is located at the global frame, or in general at

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \mathbf{p} - R\mathbf{u} \tag{6}$$

for an object at an arbitrary location (see figure 7). In the case of vertex features, rotation matrix R is entirely unconstrained, expressed in the vertex's three symmetry variables ϕ, θ, and ψ:

$$R = Euler(\langle \phi, \theta, \psi \rangle)$$

$$\begin{pmatrix} \phi \\ \theta \\ \psi \end{pmatrix} = Euler^{-1}(R) \tag{7}$$

Again, vector equations (6) and (7) supply six pose functions f_x, f_y, f_z, f_ϕ, f_θ, and f_ψ over the sense data $\mathbf{p}$ and $\mathbf{n}$ and symmetry variables d_x, d_y, and δ_z.

4 The pose Jacobian

The pose functions derived above vary over error-bounded sense variables and unbounded symmetry variables. We may propagate the linearized effects of sense and symmetry uncertainty on the object's pose with the aid of

a pose Jacobian. If enough sense data are acquired to calculate an expected, or *nominal*, pose, the Jacobian will enable us to approximate bounds on the actual pose's deviation from the nominal pose, through the propagation of sense variable bounds.

The six *dofs* of an object's pose may be represented by six symmetry and sensory *dofs* in any sensed feature. For example, a sensed planar face's three symmetry *dofs* d_x, d_y, and δ_z combine with three additional *dofs* embodied in the uncertainty of the sensed point $\mathbf{p}$ and surface normal $\mathbf{n}_z$ to form a 6-dimensional uncertainty volume in pose space. In order to represent the sense data more concisely in terms of *dofs* which are independent of the complementary symmetry *dofs*, we will adopt the following conventions. A sensed planar face point $\mathbf{p}$ is denoted by its ordinate p_n along the normal $\mathbf{n}_z$

$$p_n = \mathbf{p} \cdot \mathbf{n}_z$$

Since $\mathbf{p}$'s components along the surface tangent axes are redundant with respect to the unknown translational symmetry variables d_x and d_y, the only truly relevant information contained in the sense datum $\mathbf{p}$ is p_n.

In a similar vein, polar coordinates provide a more concise representation of a sensed normal or edge vector $\mathbf{n}_z$ than its three components n_{z_x},n_{z_y}, and n_{z_z} (see figure 8):

$$n_\alpha = cos^{-1}\, n_{z_z}$$

$$n_\beta = sgn(n_{z_x} n_{z_y}) * tan^{-1}\, \frac{n_{z_y}}{n_{z_x}}$$

The new sense variables n_α and n_β correspond to the two pose *dofs* attributable to the uncertainty in the sense datum $\mathbf{n}_z$.

Now that the sense variables are nonredundant with respect to each other and the symmetry variables, we may form a square, 6×6 *pose Jacobian*:

$$J = \begin{bmatrix} \frac{\partial f_x}{\partial p_n} & \frac{\partial f_x}{\partial d_x} & \frac{\partial f_x}{\partial d_y} & \frac{\partial f_x}{\partial n_\alpha} & \frac{\partial f_x}{\partial n_\beta} & \frac{\partial f_x}{\partial \delta_x} \\ \frac{\partial f_y}{\partial p_n} & \frac{\partial f_y}{\partial d_x} & \frac{\partial f_y}{\partial d_y} & \frac{\partial f_y}{\partial n_\alpha} & \frac{\partial f_y}{\partial n_\beta} & \frac{\partial f_y}{\partial \delta_x} \\ \frac{\partial f_z}{\partial p_n} & \frac{\partial f_z}{\partial d_x} & \frac{\partial f_z}{\partial d_y} & \frac{\partial f_z}{\partial n_\alpha} & \frac{\partial f_z}{\partial n_\beta} & \frac{\partial f_z}{\partial \delta_x} \\ \frac{\partial f_\phi}{\partial p_n} & \frac{\partial f_\phi}{\partial d_x} & \frac{\partial f_\phi}{\partial d_y} & \frac{\partial f_\phi}{\partial n_\alpha} & \frac{\partial f_\phi}{\partial n_\beta} & \frac{\partial f_\phi}{\partial \delta_x} \\ \frac{\partial f_\theta}{\partial p_n} & \frac{\partial f_\theta}{\partial d_x} & \frac{\partial f_\theta}{\partial d_y} & \frac{\partial f_\theta}{\partial n_\alpha} & \frac{\partial f_\theta}{\partial n_\beta} & \frac{\partial f_\theta}{\partial \delta_x} \\ \frac{\partial f_\psi}{\partial p_n} & \frac{\partial f_\psi}{\partial d_x} & \frac{\partial f_\psi}{\partial d_y} & \frac{\partial f_\psi}{\partial n_\alpha} & \frac{\partial f_\psi}{\partial n_\beta} & \frac{\partial f_\psi}{\partial \delta_x} \end{bmatrix}$$

Conceptually, the elements of J may be thought of as object frame velocities resulting from unit velocities of the sensed surface features along the sense and symmetry variables' *dofs*. The derivation of J's elements is therefore based on standard velocity propagation principles. We first derive Jacobian elements corresponding to derivatives of f_x, f_y, and f_z, the pose functions for the object's origin $\langle x, y, z \rangle^T$.

A derivation of derivatives in the pose Jacobian for a sensed planar face follows. Derivatives for pose Jacobians pertaining to sensed linear edges and point vertices are found in Appendix I.

Figure 5 shows a surface point $\mathbf{p}$ and surface normal $\mathbf{n}_z$, sensed on a modelled object's surface. The origin's rate of change with respect to surface point variable p_n is just the sensed normal along which p_n is defined:

$$\frac{\partial \langle f_x, f_y, f_z \rangle^T}{\partial p_n} = \mathbf{n}_z$$

Similarly, the origin's rates of change with respect to symmetry variables d_x and d_y are the transformed face frame axes along which they are defined (negated as in equation (1)):

$$\frac{\partial \langle f_x, f_y, f_z \rangle^T}{\partial d_x} = -\mathbf{n}_x$$

$$\frac{\partial \langle f_x, f_y, f_z \rangle^T}{\partial d_y} = -\mathbf{n}_y$$

As may be apparent from figure 8, a perturbation in the surface normal's polar coordinate n_β causes a rotation about the global z-axis. Since the center of rotation occurs at surface point $\mathbf{p}$, the origin's rate of change with respect to n_β is given by the cross product of the axis of rotation and the lever arm from $\mathbf{p}$ to the origin:

$$\frac{\partial \langle f_x, f_y, f_z \rangle^T}{\partial n_\beta} = \mathbf{z} \times (\langle x, y, z \rangle^T - \mathbf{p})$$

Somewhat less apparent from figure 8 is that a perturbation in n_α causes a rotation about the axis

$$\left\langle cos(n_\alpha + \frac{\pi}{2}), sin(n_\alpha + \frac{\pi}{2}), 0 \right\rangle^T$$

The origin's rate of change with respect to n_α is therefore given by the cross product

$$\frac{\partial \langle f_x, f_y, f_z \rangle^T}{\partial n_\alpha} = \left\langle cos(n_\alpha + \frac{\pi}{2}), sin(n_\alpha + \frac{\pi}{2}), 0 \right\rangle^T \times (\mathbf{f} - \mathbf{p})$$

We now turn to the derivation of Jacobian elements corresponding to derivatives of angular pose functions f_ϕ, f_θ, and f_ψ. Because sense variable p_n and symmetry variables d_x and d_y provide only translational pose information and do not influence the calculated orientation of the object, the corresponding entries in J are zero:

$$\frac{\partial(\langle f_\phi, f_\theta, f_\psi \rangle^T)}{\partial p_n} = \frac{\partial(\langle f_\phi, f_\theta, f_\psi \rangle^T)}{\partial d_x} = \frac{\partial(\langle f_\phi, f_\theta, f_\psi \rangle^T)}{\partial d_y} = \mathbf{0}$$

Finally, let vector function

$$\xi(\mathbf{r}) = \begin{pmatrix} d\phi \\ d\theta \\ d\psi \end{pmatrix}$$

denote the differential changes in the Euler angles resulting from a differential rotation about axis $\mathbf{r}$. Then the derivatives of pose orientation functions f_ϕ, f_θ, and f_ψ with respect to the angular sense and symmetry variables are

$$\frac{\partial(f_\phi, f_\theta, f_\psi)}{\partial n_\alpha} = \xi\left(\left\langle cos(n_\alpha + \frac{\pi}{2}), sin(n_\alpha + \frac{\pi}{2}), 0\right\rangle^T\right)$$

$$\frac{\partial(f_\phi, f_\theta, f_\psi)}{\partial n_\beta} = \xi(\mathbf{z})$$

$$\frac{\partial(f_\phi, f_\theta, f_\psi)}{\partial \delta_z} = \xi(\mathbf{n}_z)$$

5 Constructing the linear pose constraints

In section 2, constraint planes in 3-dimensional pose space were derived to bound the set of plausible poses of a 2-dimensional object (see figure 3). The constraint planes are the boundaries of a parallelopided-shaped prism, whose axes are the columns of the pose Jacobian formed by differentiating the nominal pose with respect to uncertain sense and symmetry variables. In this section, we derive constraint hyperplanes to bound the plausible poses of a sensed object in n-dimensional pose space.

The pose of a sensed object in n-dimensional pose space may be expressed as n functions over m sense variables and $n - m$ symmetry variables. As

shown in section 3, for instance, the sense data sampled from a planar surface of a 3-dimensional object give rise to six pose equations over three sense variables and three symmetry variables. Differentiating the n pose functions with respect to the n sense and symmetry variables yields an $n \times n$ pose Jacobian J. The evaluation of J at the nominal pose $\bar{x}$ enables us to propagate sense variable bounds to pose space, yielding hyperplanar pose constraint planes. We evaluate J at $\bar{x}$ by plugging into J the sense data acquired from the surface feature in question, and symmetry values that are the most consistent with the nominal pose and the sense data [2].

The technique for constructing two pose constraint hyperplanes $\Gamma^+_{s_i}$ and $\Gamma^-_{s_i}$ for each sense variable follows. Let us denote the columns of J by $J_1, J_2, \ldots, J_n$. Column vector J_i contains the derivatives of the n pose functions with respect to the sense or symmetry variable s_i. Sense variable s_i's maximal deviation from its nominal value is denoted by the absolute error bound ϵ_{s_i}. The propagation of sense variable bounds to pose space is achieved by scaling each sense variable's Jacobian column J_i by $\pm\epsilon_{s_i}$, yielding linearized perturbations of the pose from its nominal, $\bar{x}$. Thus, $\bar{x} \pm \epsilon_{s_i}$ provide points on the respective constraint planes $\Gamma^+_{s_i}$ and $\Gamma^-_{s_i}$.

The normals $\pm\hat{n}_i$ of $\Gamma^+_{s_i}$ and $\Gamma^-_{s_i}$ are derived by noting that every column of J (i.e., every parallelopiped axis) *except* J_i lies parallel to the hyperplanes $\Gamma^+_{s_i}$ and $\Gamma^-_{s_i}$ in pose space. Hyperplane normals $\pm\hat{n}_i$ are therefore easily calculated by employing the Gram-Schmidt technique (see [8]) to produce an a vector ($\hat{n}_i$ or $-\hat{n}_i$) that is orthogonal to the $n-1$-dimensional subspace spanned by the Jacobian columns $J_1, \ldots, J_{i-1}, J_{i+1}, \ldots, J_n$.

Finally, each pair of pose constraints $\Gamma^+_{s_i}$ and $\Gamma^-_{s_i}$ give rise to two inequalities in a system of $2m$ linear pose constraints $A\mathbf{x} \leq \mathbf{b}$. The hyperplanar

[2] The derivation of these symmetry values is performed by iterative projection, as described in Appendix IV.

constraint $\Gamma^+_{s_i}$ is represented by the inequality

$$\hat{n}_i \cdot \mathbf{x} \leq (\bar{x} + \epsilon_{s_i} J_i) \cdot \hat{n}_i$$

which asserts that the ordinate $\hat{n}_i \cdot \mathbf{x}$ of a plausible pose $\mathbf{x}$ along $\Gamma^+_{s_i}$'s normal $\hat{n}_i$ is no greater than the ordinate of $\Gamma^+_{s_i}$'s point $\bar{x} + \epsilon_{s_i} J_i$. This inequality is entered into the system $A\mathbf{x} \leq \mathbf{b}$ of linear constraints by placing $\hat{n}_i$ in a row of A and placing $(\bar{x} + \epsilon_{s_i} J_i) \cdot \hat{n}_i$ in the corresponding row of $\mathbf{b}$. The pose constraint for $\Gamma^-_{s_i}$ is similarly derived as

$$-\hat{n}_i \cdot \mathbf{x} \leq (\bar{x} - \epsilon_{s_i} J_i) \cdot -\hat{n}_i$$

which is also entered into the constraint system $A\mathbf{x} \leq \mathbf{b}$.

6 The pose uncertainty algorithm

Here we present a concise description of our technique for constructing linear pose constraints $A\mathbf{x} \leq \mathbf{b}$ for the pose of a sensed object in n-dimensional pose space. The sense data from l sensed surface features are first combined in the manner described in Appendix III to arrive at a nominal object pose $\bar{\mathbf{x}}$. Every ith surface feature's pose function $\mathbf{f}(\vec{\sigma}_i, \vec{\psi}_i)$ over its m_i sense variables σ_{ij} and $n - m$ symmetry variables ψ_{ij} is linearized about values for σ_{ij} and ψ_{ij} corresponding to $\bar{\mathbf{x}}$. Linearization of $\mathbf{f}(\vec{\sigma}_i, \vec{\psi}_i)$ is accomplished by forming a pose Jacobian J (see section 4), whose columns serve to define the axes of a parallelopiped-shaped constraint prism in pose space.

The $2m_i$ hyperplanar walls of the ith feature's pose constraint prism represent the error bounds, propogated to pose space, on the feature's m_i sense variables. The Gram-Schmidt technique is employed to obtain the normals $\pm\hat{n}_{ij}$ for each opposing pair of hyperplanar prism walls. All hyperplanar constraints are finally expressed as inequalities and entered as coefficients

into the appropriate rows of a linear constraint system $A\mathbf{x} \leq \mathbf{b}$. The resulting pose constraints may be used as input to a linear programming package to determine the maximum possible extent of the pose along any objective function axis $\mathbf{c}$ in pose space.

As mentioned above, the m_i features' pose functions $\mathbf{f}(\vec{\sigma}_i, \vec{\psi}_i)$ are linearized about values for σ_{ij} and ψ_{ij} corresponding to $\bar{\mathbf{x}}$. A complication arises from the fact that the nominal pose of the sensed object is generally inconsistent with any given feature's sense data. This is due to the averaging of pose calculations performed on redundant and erroneous sense data. Strictly speaking, there is *no* well-defined value for $\vec{\psi}_i$ corresponding to $\bar{\mathbf{x}}$, since there is no $\vec{\psi}_i$ for which $\mathbf{f}(\vec{\sigma}_i, \vec{\psi}_i) = \bar{\mathbf{x}}$. Moreover, the object pose $\bar{\mathbf{y}} = \mathbf{f}(\vec{\sigma}_i, \vec{\psi}_i)$ corresponding to the inverse pose function result $\vec{\psi}_i = \mathbf{f}^{-1}(\vec{\sigma}_i, \bar{\mathbf{x}})$ is not necessarily the closest pose to $\bar{\mathbf{x}}$ which is consistent with the ith feature's sense data. We therefore employ a numerical technique, described in Appendix IV, to arrive at a value for $\vec{\psi}_i$ for which $\bar{\mathbf{y}} = \mathbf{f}(\vec{\sigma}_i, \vec{\psi}_i)$ is as near as possible to the nominal pose $\bar{\mathbf{x}}$.

NOMENCLATURE

l	Number of sensed object features.
m_i	Number of sense variables in the ith feature.
$m = \sum_{i=1}^{l} m_i$	Total number of sense variables
n	Number of pose space dimensions.
$\vec{\sigma}_i$	Vector of m_i sense variables in the ith feature.
$\vec{\psi}_i$	Vector of $n - m_i$ symmetry variables in the ith feature.
$\epsilon_{\sigma_{ij}}$	Absolute error bound on the jth sense variable of the ith feature.
$\mathbf{x}$	Pose vector of length n.
$\bar{\mathbf{x}}$	Nominal pose vector of length n.
$\bar{\mathbf{y}}$	Projection of nominal pose vector $\bar{\mathbf{x}}$ onto the hypersurface of poses which are consistent with a feature's sense data.
$\mathbf{f}(\vec{\sigma}_i, \vec{\psi}_i)$	Pose function over the ith feature's sense and symmetry

	variables, returning pose vector $\mathbf{x}$.
$\mathbf{f}^{-1}(\vec{\sigma}_i, \mathbf{x})$	Inverse pose function which returns symmetry variables $\vec{\psi}_i$ such that $\mathbf{f}(\vec{\sigma}_i, \vec{\psi}_i) = \mathbf{x}$.
$\mathbf{f}^*(\vec{\sigma}_i, \mathbf{x})$	Function which returns symmetry variable vector $\vec{\psi}_i$ such that $\mathbf{f}(\vec{\sigma}_i, \vec{\psi}_i)$ is the closest possible approximation to $\bar{\mathbf{x}}$ when there is no $\vec{\psi}_i$ for which $\mathbf{f}(\vec{\sigma}_i, \vec{\psi}_i)$.
$NOM(\sigma_1, \ldots, \sigma_l)$	Nominal pose function returning the pose $\bar{x}$ most consistent with the l features' sense data.
J	$n \times n$ Jacobian obtained by differentiating a feature's pose functions with respect to its sense and symmetry variables.
J_i	The ith column vector of pose Jacobian J.
$GS(\mathbf{v}_1, \ldots, \mathbf{v}_{n-1})$	Modified Gram-Schmidt procedure which returns vectors $\pm\mathbf{v}_n$ orthogonal to the subspace spanned by $\mathbf{v}_1, \ldots, \mathbf{v}_{n-1}$.
A	$2m \times n$ matrix in the linear constraint system $A\mathbf{x} \leq \mathbf{b}$.
A_i	The ith row of matrix A.
$\mathbf{b}$	$2m$-length vector in the linear constraint system $A\mathbf{x} \leq \mathbf{b}$.
$\mathbf{b}_i$	The ith element of vector $\mathbf{b}$.

Begin /* Construct pose constraints $A\mathbf{x} \leq \mathbf{b}$ */

$NOM(\vec{\sigma}_1, \ldots, \vec{\sigma}_l) \rightarrow \bar{\mathbf{x}}$

$0 \rightarrow ROW$

For $i \in 1 \ldots l$

$\mathbf{f}^*(\vec{\sigma}_i, \mathbf{x}) \rightarrow \vec{\psi}_i$
$\mathbf{f}(\vec{\sigma}_i, \vec{\psi}_i) \rightarrow \bar{\mathbf{y}}$

$\left[\frac{\partial \mathbf{f}(\vec{\sigma}_i, \vec{\psi}_i)}{\partial(\vec{\sigma}_i, \vec{\psi}_i)}\right] \rightarrow J$

For $j \in 1 \ldots m_i$

$GS(J_1, \ldots, J_{j-1}, J_{j+1}, \ldots, J_n) \rightarrow \hat{n}_{ij}, -\hat{n}_{ij}$

$ROW + 1 \rightarrow ROW$
$\hat{n}_{ij} \rightarrow A_{ROW}$
$(\bar{\mathbf{y}} + \epsilon_{\sigma_{ij}} J_i) \cdot \hat{n}_{ij} \rightarrow \mathbf{b}_{ROW}$

$ROW + 1 \rightarrow ROW$
$-\hat{n}_{ij} \rightarrow A_{ROW}$
$(\bar{\mathbf{y}} - \epsilon_{\sigma_{ij}} J_j) \cdot -\hat{n}_{ij} \rightarrow \mathbf{b}_{ROW}$

Endfor

Endfor

Return A, $\mathbf{b}$

End

7 Discussion

The pose constraint construction procedure described in section 5 returns matrix A and vector $\mathbf{b}$ of the linear constraint system $A\mathbf{x} \leq \mathbf{b}$. These coefficent structures may in turn be submitted to a linear programming procedure, along with any desired pose space axis $\mathbf{c}$, to determine the greatest possible extent of the pose along axis $\mathbf{c}$. Such calculations can be useful during the real-time execution of free or manipulator motions in which a tool or assembly part is grasped by a touch-sensitive end-effector. By calculating the greatest possible extent of the object's pose in the direction of an obstacle's surface, collisions may be avoided. Compliance strategies that relate object-centered forces to object displacements should also take into account the affect of object pose uncertainty on the resulting forces applied to the object.

In the interest of achieving real-time computability, some properties of the linear constraint system $A\mathbf{x} \leq \mathbf{b}$ output by our algorithm should be observed. First, note that the $2m \times n$ matrix of coeffients A has a constant

number of columns, i.e., the number of pose space dimensions n. Thus, for 3-dimensional objects, A typically has 6 columns. The linear programming algorithm employed could be designed with a constant $n = 6$ dimensions in mind, with the loss of generality resulting in greater efficiency. In fact, Megiddo [11] has introduced a linear programming algorithm with a time requirement that is linear in the number of constraints when the dimensionality is fixed. Although the algorithm is double-exponential in the number of dimensions, its application toward the maximization of pose could be advantageous when a large number of sensed features yields a large number of constraints.

Second, the algorithm presented in section 5 lends itself well to parallel implementation. Each of the m_i iterations of the inner loop, as well as the l iterations of outer loop, may be executed in parallel, with few computational dependencies to be found among them. Moreover, very few access and storage conflicts occur. For example, the pair of pose constraints arising from each sense datum are placed in the datum's own pair of rows in $A\mathbf{x} \leq \mathbf{b}$, so no storage bottleneck takes place.

Owing to the pose function linearizations involved in our constraint construction technique, the sensors utilized for acquiring the surface points and normals must be very accurate. The linearized pose functions provide decreasingly valid approximations of the propagated pose deviations as the sense variable error ranges increase in size. The algorithm is currently being evaluated for various sensory error magnitudes. We are planning a robot implementation of this algorithm using the JPL/Stanford hand as a testbed, with fingertip-shaped force sensors supplied by Brock Research, Inc.

8 Conclusion

We have presented a technique for constructing a constraint volume in pose space which contains the set of object poses that are consistent with sensory

data acquired from the object's surface. Pose functions defined over sense and feature symmetry variables are linearized about the nominal object pose, so that linear sensory error bounds may be mapped into pose space to form hyperplanar pose constraints. The pose constraints $A\mathbf{b} \leq \mathbf{x}$ are then submitted to a linear programming algorithm, together with any desired pose direction $\mathbf{c}$, in order to determine the maximum possible extent of the pose along $\mathbf{c}$. The resulting pose optimizations will be useful in the realms of path planning, obstacle avoidance, and object-based force control.

Appendix I

DERIVATIVES OF POSE FUNCTIONS FOR LINEAR EDGE FEATURES

$$\frac{\partial \langle f_x, f_y, f_z \rangle^T}{\partial p_x} = \mathbf{n}_x$$

$$\frac{\partial \langle f_x, f_y, f_z \rangle^T}{\partial p_y} = \mathbf{n}_y$$

$$\frac{\partial \langle f_x, f_y, f_z \rangle^T}{\partial d_z} = -\mathbf{n}_z$$

$$\frac{\partial \langle f_x, f_y, f_z \rangle^T}{\partial n_\alpha} = \left\langle cos(n_\alpha + \frac{\pi}{2}), sin(n_\alpha + \frac{\pi}{2}), 0 \right\rangle^T \times (\mathbf{f} - \mathbf{p})$$

$$\frac{\partial \langle f_x, f_y, f_z \rangle^T}{\partial n_\beta} = \mathbf{z} \times (\langle x, y, z \rangle^T - \mathbf{p})$$

$$\frac{\partial(\langle f_\phi, f_\theta, f_\psi \rangle^T)}{\partial p_x} = \frac{\partial(\langle f_\phi, f_\theta, f_\psi \rangle^T)}{\partial p_y} = \frac{\partial(\langle f_\phi, f_\theta, f_\psi \rangle^T)}{\partial d_z} = \mathbf{0}$$

$$\frac{\partial(\langle f_\phi, f_\theta, f_\psi \rangle^T)}{\partial n_\alpha} = \xi \left(\left\langle cos(n_\alpha + \frac{\pi}{2}), sin(n_\alpha + \frac{\pi}{2}), 0 \right\rangle^T \right)$$

$$\frac{\partial(\langle f_\phi, f_\theta, f_\psi \rangle^T)}{\partial n_\beta} = \xi(\mathbf{z})$$

$$\frac{\partial(\langle f_\phi, f_\theta, f_\psi \rangle^T)}{\partial \delta_z} = \xi(\mathbf{n}_z)$$

Appendix II

DERIVATIVES OF POSE FUNCTIONS FOR POINT VERTEX FEATURES

Let vector function $\varsigma(\gamma) = \mathbf{r}$ denote the axis of a differential rotation resulting from a differential increase in Euler angle γ.

$$\frac{\partial \langle f_x, f_y, f_z \rangle^T}{\partial \langle p_x, p_y, p_z \rangle^T} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\frac{\partial \langle f_x, f_y, f_z \rangle^T}{\partial \phi} = \varsigma(\phi) \times (\mathbf{f} - \mathbf{p})$$

$$\frac{\partial \langle f_x, f_y, f_z \rangle^T}{\partial \theta} = \varsigma(\theta) \times (\mathbf{f} - \mathbf{p})$$

$$\frac{\partial \langle f_x, f_y, f_z \rangle^T}{\partial \psi} = \varsigma(\psi) \times (\mathbf{f} - \mathbf{p})$$

$$\frac{\partial \langle f_\phi, f_\theta, f_\psi \rangle^T}{\partial \langle p_x, p_y, p_z \rangle^T} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$\frac{\partial \langle f_\phi, f_\theta, f_\psi \rangle^T}{\partial \langle \phi, \theta, \psi \rangle^T} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Appendix III

CALCULATING THE NOMINAL POSE

Calculation of a modelled object's pose from sparse surface points $\mathbf{p}_i$ and normals or edge directions $\mathbf{n}_i$ proceeds in two stages [7]. First the object's orientation is derived, described as a rotation $R = rot(\mathbf{k}, \theta)$ of θ radians about an axis $\mathbf{k}$. The position of the object is then calculated as a displacement vector $\mathbf{v}$.

A nominal rotation $R = rot(\mathbf{k}, \theta)$ of θ may be calculated using any two distinct normals $\mathbf{n}_1$ and $\mathbf{n}_2$ sampled from the object. If there are more than two distinct normals available, the rotation descriptors $\mathbf{k}$ and θ may be computed for each pair of normals, and the nominal rotation R can be obtained by averaging the results.

To derive rotation axis $\mathbf{k}$ from sensed normals $\mathbf{n}_1$ and $\mathbf{n}_2$, we compare the normals with their nonrotated counterparts $\mathbf{m}_1$ and $\mathbf{m}_2$ in the object model. Since the displacements $\mathbf{m}_1 - \mathbf{n}_1$ and $\mathbf{m}_2 - \mathbf{n}_2$ resulting from the rotation about axis $\mathbf{k}$ must both be perpendicular to $\mathbf{k}$, the rotation axis may be calculated as their cross product:

$$\mathbf{k} = (\mathbf{m}_1 - \mathbf{n}_1) \times (\mathbf{m}_2 - \mathbf{n}_2)$$

The angle of rotation θ is computed as the angle separating the projections of any $\mathbf{m}_i$ and $\mathbf{n}_i$ onto a plane whose normal is $\mathbf{k}$, or equivalently:

$$\theta = (\mathbf{m}_i \times \mathbf{k}) \cdot (\mathbf{n}_i \times \mathbf{k})$$

Having parametrized the nominal rotation matrix $R = rot(\mathbf{k}, \theta)$, the translatory component of the pose may be calculated as follows. Given any sensed surface point $\mathbf{p}_i$ residing in a plane whose equation in the object model is $\mathbf{m}_i \cdot [x\ y\ z]^T = d_i$, we obtain the following planar equation in scene coordinates:

$$R\mathbf{m}_i \cdot (\mathbf{p}_i - \mathbf{v}) = d_i$$

Such equations in three unknowns v_x, v_y, and v_z may be generated by each surface point $\mathbf{p}_i$ and combined to form a system of equations whose solution provides the pose component $\mathbf{v}$. Each point acquired from a planar surface generates one equation of this form, while a point acquired from a polyhedral edge or vertex generates two or three equations, respectively. Three distinct equations in v_x, v_y, and v_z are sufficient to determine $\mathbf{v}$. When more than three equations are produced, $\mathbf{v}$ may be obtained as the best least squares solution to the overdetermined system of equations (see [8]).

Appendix IV

PROJECTING THE NOMINAL POSE ONTO A SINGLE FEATURE'S HYPERSURFACE OF POSES

The set of object poses consistent with the ith feature's m_i nominal sense variables σ_{ij} may be described as an $(n - m_i)$-dimensional hypersurface in n-dimensional pose space, where $n - m_i$ is the number of surface symmetry variables ψ_{ij} This hypersurface of poses is depicted schematically in figure 9 as the pose function curve $\mathbf{f}(\vec{\sigma}_i, \vec{\psi}_i)$ The nominal pose $\bar{\mathbf{x}}$ does not generally lie on a feature's pose hypersurface, so we opt to choose the closest point on the hypersurface to $\bar{\mathbf{x}}$.

The following numerical technique has been employed to compute the symmetry variable vector $\vec{\psi}_i$ for which $\mathbf{f}(\vec{\sigma}_i, \vec{\psi}_i)$ is the closest point on the feature's pose hypersurface to the nominal pose $\bar{\mathbf{x}}$. The nominal pose is iteratively projected onto the hyperplane defined by the current "best" hypersurface point $\bar{\mathbf{y}}$ and the slope at $\bar{\mathbf{y}}$. Subroutine $HPROJ$ computes the

hyperplanar projection, using the columns of a pose Jacobian to represent a basis for the vector space coset comprising the hyperplane. Each projection $\bar{\mathbf{w}}$ yields a new symmetry variable vector $\vec{\psi}_i$ by way of the inverse pose function, and $\bar{\mathbf{y}}$ is updated via the forward pose function. Figure 9 illustrates the calculation of $\bar{\mathbf{y}}$ for two iterations. In our experience, $\bar{\mathbf{y}}$ converges rapidly toward the closest point on the hypersurface to $\bar{\mathbf{x}}$. Three or four iterations are sufficient to obtain a precise projection.

Begin /* Compute $\mathbf{f}^*(\vec{\sigma}_i, \mathbf{x})$ */

$\bar{\mathbf{x}} \rightarrow \bar{\mathbf{w}}$

Repeat

$\mathbf{f}^{-1}(\vec{\sigma}_i, \mathbf{x}) \rightarrow \psi_i$
$\mathbf{f}(\vec{\sigma}_i, \vec{\psi}_i) \rightarrow \bar{\mathbf{y}}$

$\left[\frac{\partial \mathbf{f}(\vec{\sigma}_i, \vec{\psi}_i)}{\partial(\vec{\psi}_i)}\right] \rightarrow J$

$HPROJ(\bar{\mathbf{x}}, \bar{\mathbf{y}}, J) \rightarrow \bar{\mathbf{w}}$

Until $\bar{\mathbf{y}}$ converges

Return $\vec{\psi}_i$

End

References

1. Brooks, R.A. (1982), "Symbolic error analysis and robot planning", *International Journal of Robotics Research*, Vol. 1, No. 4 pp. 29-68.

2. Durrant-Whyte, H.F. (1987), "Uncertain geometry in robotics", *Proceedings of the IEEE Conference on Robotics and Automation*, Vol. 2, pp. 851-856.

3. Farreny, Prade (1986), "Tackling uncertainty and imprecision in robotics" in Faugeras, O.D., Giralt, G. (eds), *Robotics Research*, MIT Press, pp. 85-92.

4. Fleming, Alan (1985), "Analysis of uncertainties in a structure of parts", *International Journal of Computing and Artificial Intelligence*, pp. 1113-1115.

5. Gordon, S.J. (1986), "Automated Assembly Using Feature Localization", Technical Report 932, MIT Artificial Intelligence Laboratory.

6. Grimson, W.E.L. (1986), "Sensing strategies for disambiguating among multiple objects in known poses", *IEEE Journal of Robotics and Automation*, Vol. RA-2, No. 4, pp. 196-213.

7. Grimson, W.E.L., Lozano-Perez, T. (1984), "Model-based recognition and localization from sparse range data or tactile data", *International Journal of Robotics Research*, Vol. 3, No. 3, pp. 3-35.

8. Johnson, L.W., Riess, R.D. (1981), *Introduction to Linear Algebra*, Reading, Mass.: Addison-Wesley Publishing Co.

9. Losano-Perez, T. (1981), "Automatic planning of manipulator transfer movements", *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. SMC-11, No. 10, pp. 681-697.

10. Lozano-Perez, T., Mason, M., Taylor, R.H. (1984), "Automatic synthesis of fine-motion strategies for robots", *International Journal of Robotics Research*, Vol. 3, No. 1, pp. 3-24.

11. Megiddo, N. (1984), "Linear programming in linear time when the dimension is fixed", *J. AMC*, Vol.31, No. 1, pp. 114-127.

12. Popplestone, R., Weiss, R., Liu, Y. (1988), "Using characteristic invariants to infer new spatial relationships from old", *Proceedings of the 1988 IEEE Conference on Robotics and Automation*, Vol. 2, pp. 1106-1112.

13. Smith, R.C., Cheeseman, P. (1987), "On the representation and estimation of spatial uncertainty", *International Journal of Robotics Research*, Vol. 5, No. 4, pp. 56-68.

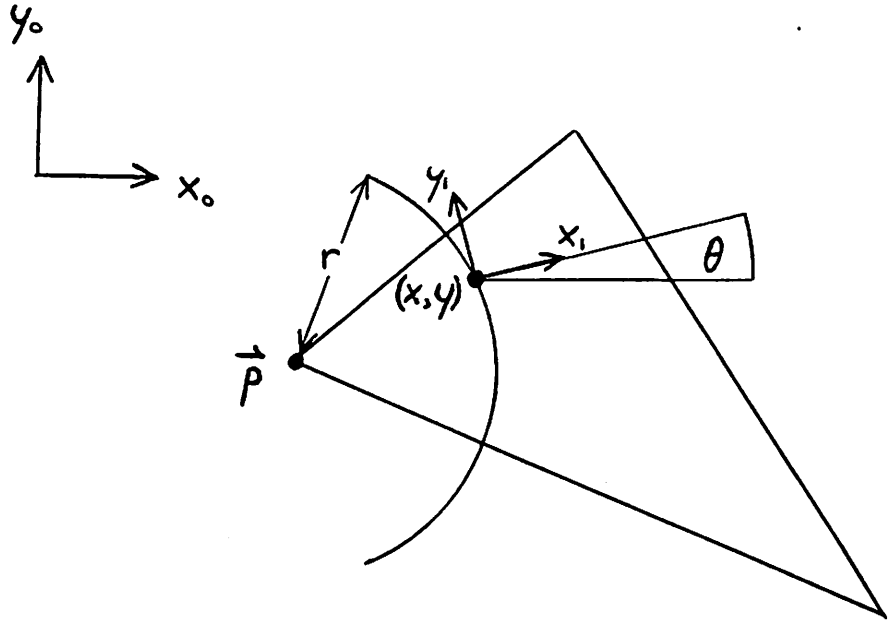

Figure 1. Sensing a 2-D object vertex

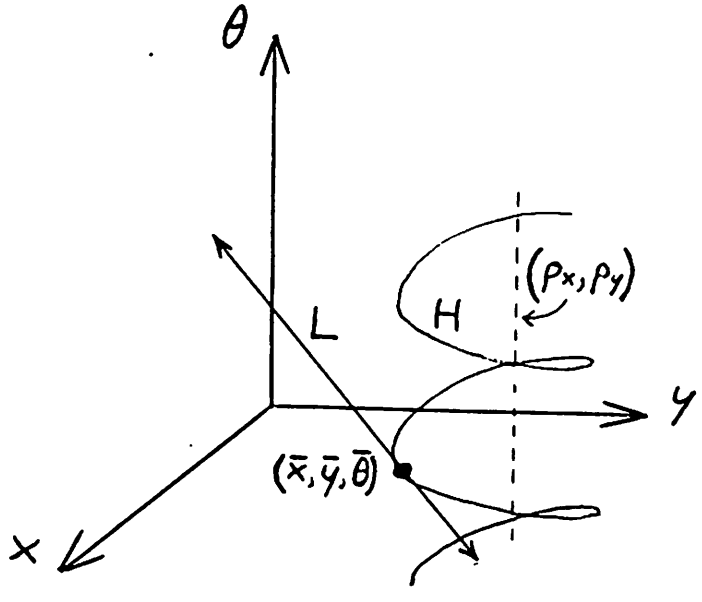

Figure 2. Symmetry uncertainty in 3-D pose space

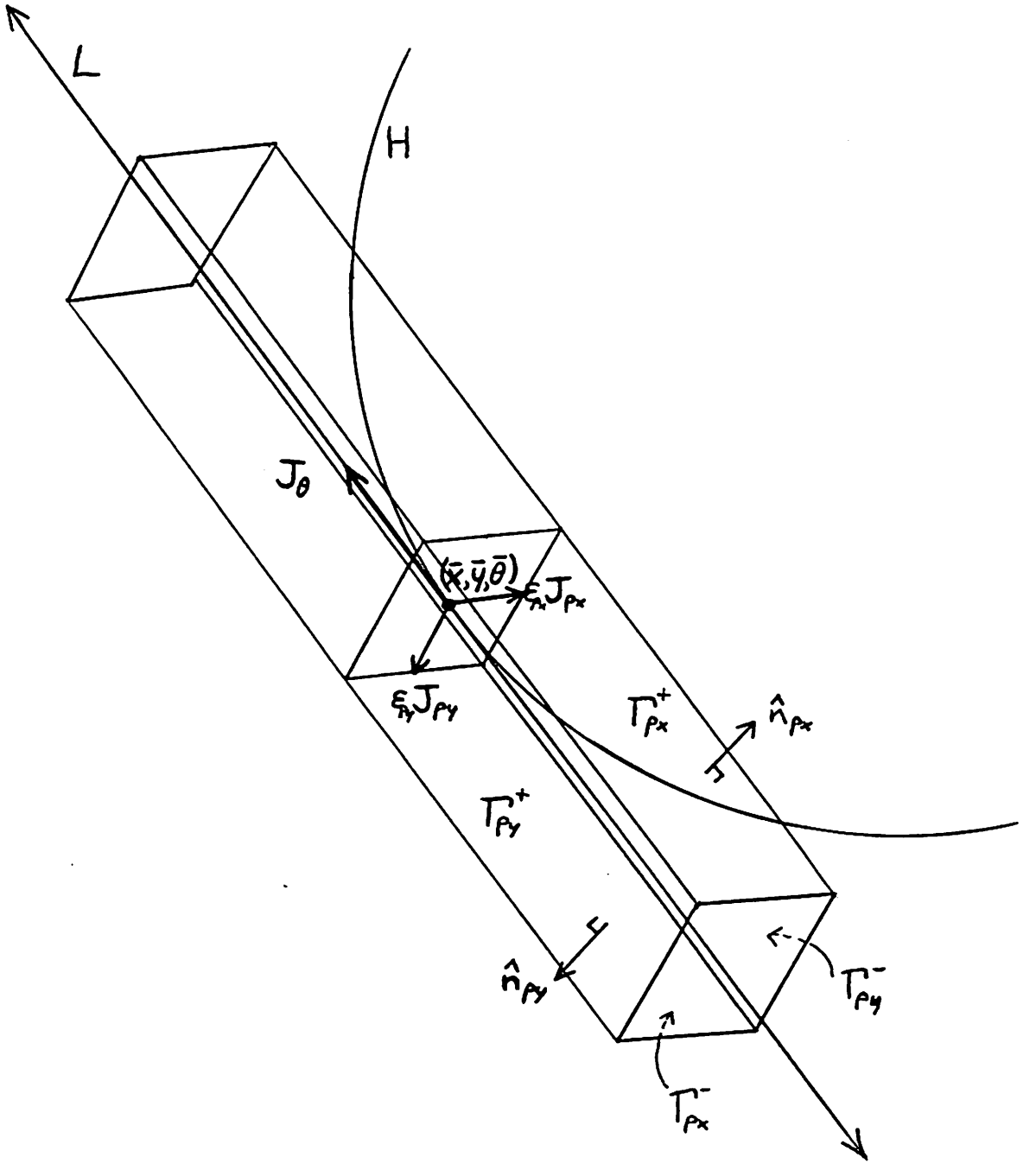

Figure 3. Pose constraint prism in 3-D pose space

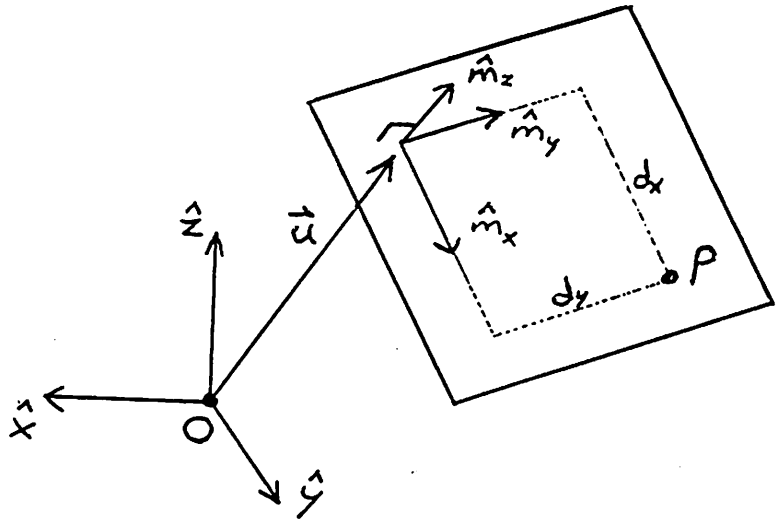

Figure 4. Surface point in planar face coordinates (object at origin)

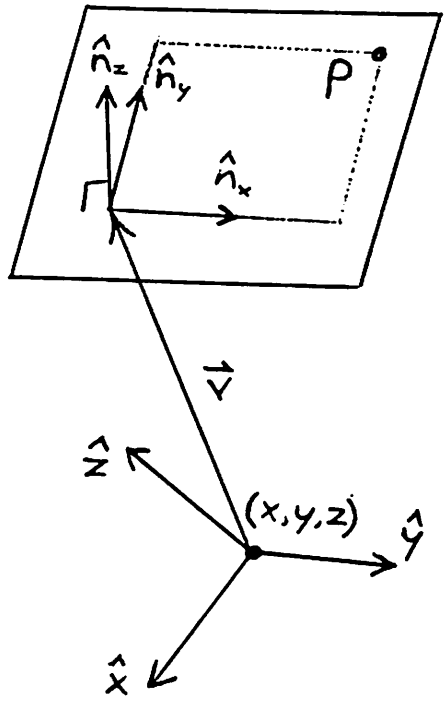

Figure 5. Surface point in planar face coordinates (object not at origin)

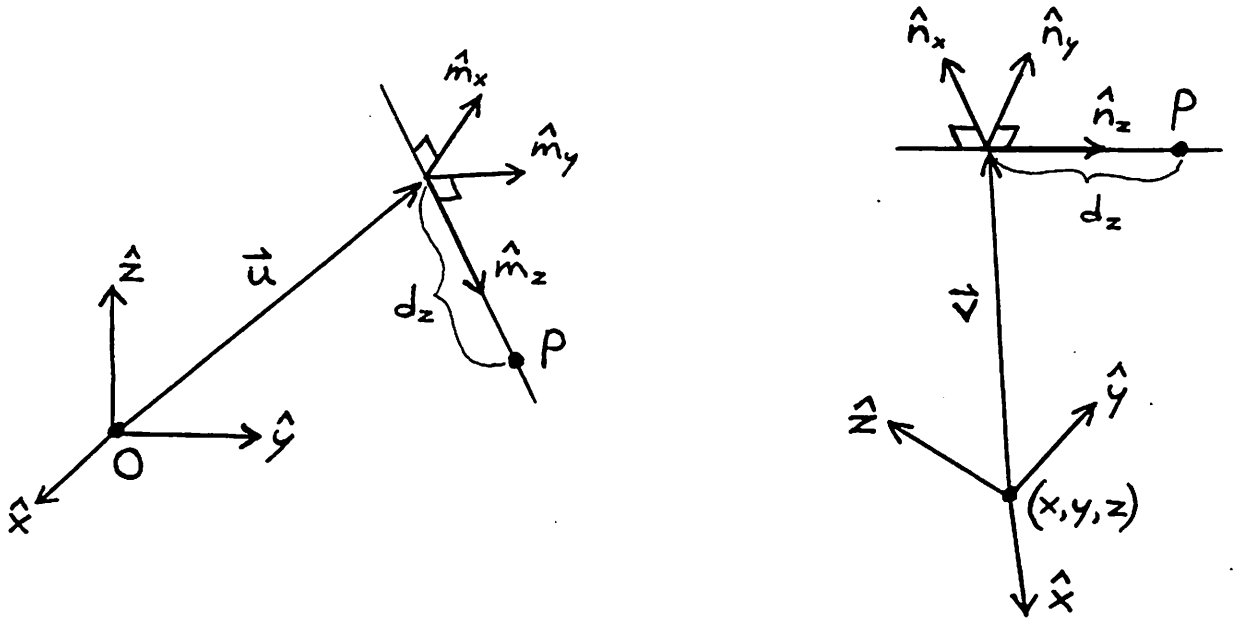

Figure 6. Surface point in linear edge coordinates

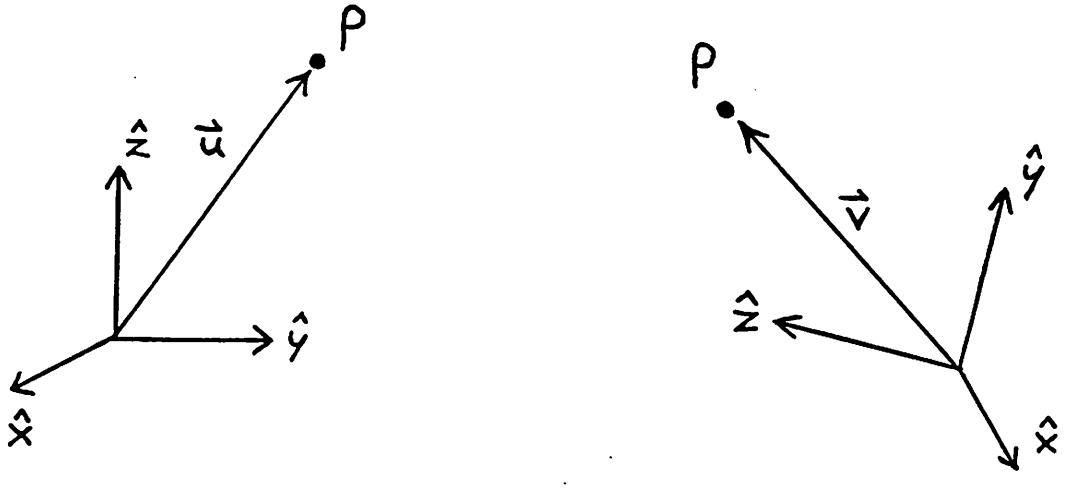

Figure 7. Surface point in vertex coordinates

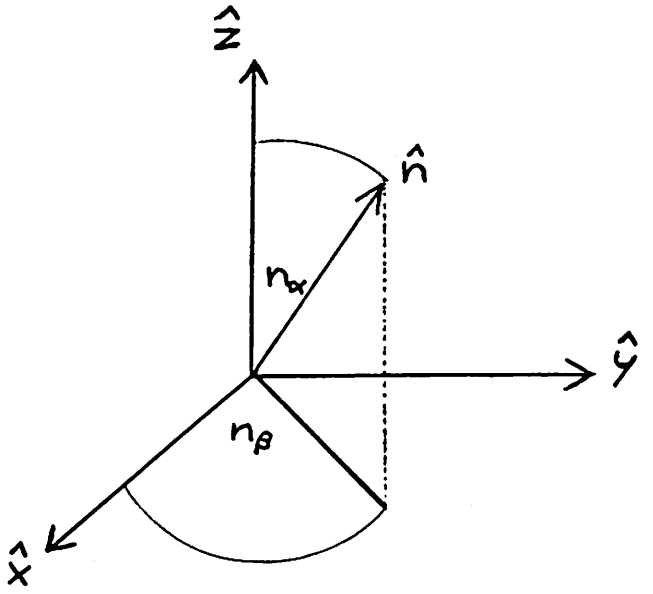

Figure 8. Polar coordinate represention of a surface normal

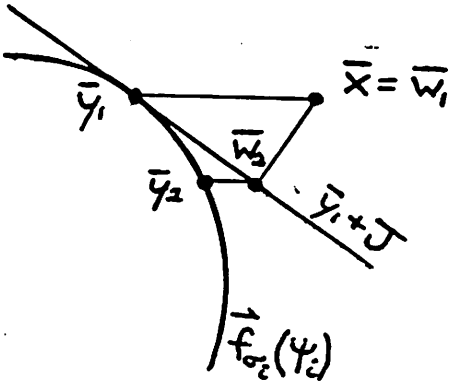

Figure 9. Projecting the nominal pose onto a feature's pose hypersurface