Abstract

Task Specification Languages for Uncalibrated Visual Servoing

Zachary Bechman Dodds

Yale University

2000

In the field of vision-based robotic control, there is a large and growing body of work which investigates *how* to perform tasks based on direct image feedback. Work in this area, often called image-based visual servoing, has successfully demonstrated a variety of interesting and useful tasks.

Little attention, however, has been paid to *what* tasks the image-based servoing framework makes possible. Answering the question "What tasks are performable with an uncalibrated stereo rig?" is the underlying motivation for this thesis.

In particular, a formalism is presented that makes the above question precise and then provides a complete answer for a number of different possible meanings of "uncalibrated." From these results this work builds a complete enumeration of distinguishable feature configurations for the considered sets of uncalibrated cameras. Such enumerations, in turn, lead to concise and provably complete languages for specifying tasks. These languages offer a small number of primitive skills and composition operators that together span the set of all tasks specifiable with image-based constraints.

Applications of these formal results are also considered and tested. First, each of the primitive skills is shown to have an image-based constraint that is satisfied exactly when the underlying 3d geometric constraint is satisfied. The composition operators then provide a means for automatically generating image-based constraints for any specifiable task. Secondly, the full enumeration of distinct feature configurations allows a process to monitor whether or not enough feature points are available for a desired task specification. Example task specifications and results of their implementation are presented.

Task Specification Languages for Uncalibrated Visual Servoing

A Dissertation
Presented to the Faculty of the Graduate School
of
Yale University
in Candidacy for the Degree of
Doctor of Philosophy

by
Zachary Bechman Dodds

Dissertation Director: Gregory D. Hager

May 2000

Acknowledgments

I want to express my genuine thanks to the many friends, colleagues, and mentors that made this effort possible. First, I owe the greatest debt in the development of this work and in my development as a researcher to Greg Hager. He managed to balance friendly guidance and an encouragement to explore in a way that I aspire to with my own students. His influence made the undertaking of the past six years not only rewarding and worthwhile in hindsight, but enjoyable in its progression.

I am grateful to the balance of my committee for giving shape to the experience: Drew McDermott for his insights from often unexpected directions, Peter Belhumeur for helping keep things in perspective, and Seth Hutchinson for our friendly and supportive encounters. In addition, I want to thank Stanley Eisenstat for fostering, mentoring, and furthering my interest in teaching, Dana Angluin for our research conversations and for connecting me with many types of students, Hemant Tagare for our numerous and wide-ranging discussions, and Mike Fischer for encouraging a balance between technical precision and technical presentation.

As much as from established researchers and teachers, I have benefited from both the formal and informal interactions with fellow students, postdocs, and visitors. Kentaro Toyama, Chris Rasmussen, Chris Cantor, Martin Jagersand, and Jonas August pushed me to broaden my horizons and reminded me why I was working towards a degree. Finally, I want to acknowledge the enormous debt I owe to my mother, brother, and father for their consistent encouragement; and especially to my wife, Rita, for her constant motivation and support throughout the past four years.

Contents

List of Figures

Chapter 1

Introduction

Robotics strives to create devices that interact with the physical world. Whether assisting in an assembly task or demonstrating self-preserving behavior, robotic agents intend to have an impact on objects within their sphere of influence.

Unless in a perfectly engineered environment, a robot needs to sense the surrounding world in order to interact with it effectively. Vision holds enormous promise as a sensor in its flexibility (because it is passive), in its bandwidth (a camera provides more than 7 MB of data each second), and in its effective use by many biological systems (from bees to humans). Developing artificial systems which use image information as capably as we do, however, is a long-standing open problem.

One difficulty of the vision problem which affects robotic interaction with the world is the often uncertain relationship between *image space* (one or more 2d images) and *task space*, the environment in which the robot acts (the 3d world around us). This uncertainty between image space and task space presents itself in two related forms: uncertainty in the pose of the camera(s) with respect to a scene and the resulting uncertainty in the configuration of objects in task space. In essence, lack of knowledge about the imaging system leads to potential ambiguity in objects' locations.

To make these ideas more concrete, the next two sections present examples of this "potential ambiguity" between images of the world and the world itself.

1.1 An Everyday Example of Visual Ambiguity

Figure 1.1 shows an ordinary newspaper picture which demonstrates how even the human visual system can incorrectly extract 3d geometric information from an image.

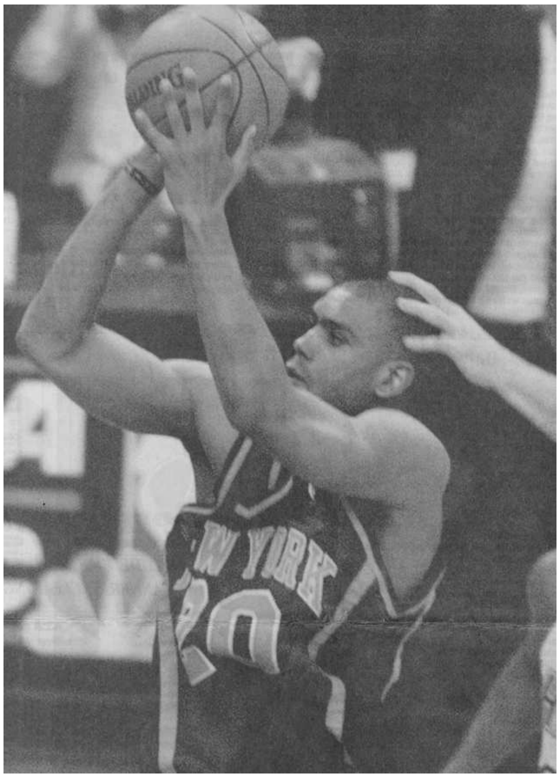


Figure 1.1: A photograph demonstrating ambiguity of the 3d locations of different parts of an imaged scene [26].

To many observers, the defender's hand in Figure 1.1 looks firmly clasped on the back of the shooter's head. In actuality, the hand is well away from the shooter and very close to the depth of his left forearm. Given this scenario, imagine if the defender had only the information from this image with which to evaluate the position of his hand — just as a visually controlled robot might. He would be unable to determine whether he was committing a foul or merely reaching out for the ball, since either situation yields the same image information.

One approach to disambiguating the different possible worlds which Figure 1.1 could depict is to use precise models of both the objects imaged and the camera's geometry. Knowing the exact dimensions of the shooter's head and defender's hand, along with an accurate estimate of the pose of the camera with respect to those objects, would suffice to determine the true 3d geometry of the scene. Only under extremely controlled conditions, however, are such object and camera models available. Many vision-based robotic applications fall outside such controlled environments, e.g., when manipulating objects from a mobile base, if operating remotely or in hazardous environments, or when using flexible or modular sensing systems.

1.2 Ambiguity in Stereo Vision

Even without a precise model of the imaging device, adding another eye is an alternative approach to disambiguating among possible 3d configurations of objects. Although the depth confusion in Figure 1.1 disappears (at least in theory) when a second camera observes the scene, stereo vision does not eliminate 3d ambiguity. For example, consider the tasks performed in Figure 1.2. Two cameras, with unknown pose with respect to each other and the scene, observe three tasks: a pair of scissors intending to divide an object in half, a probe approaching a conductor, and the ends of two wires about to make contact.

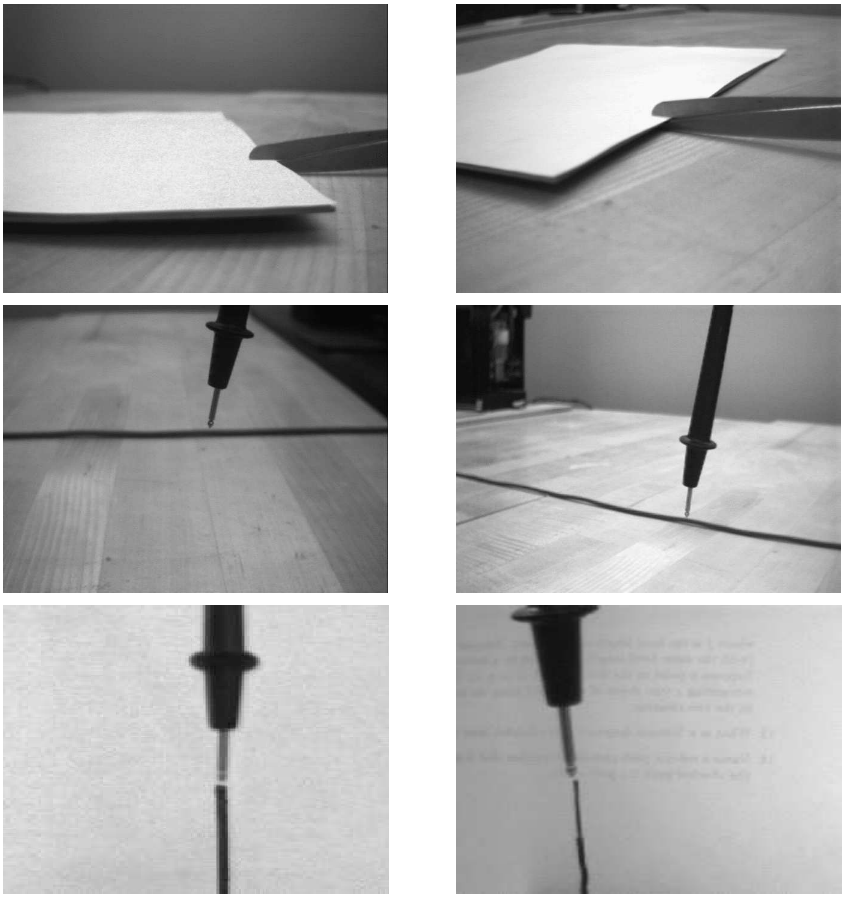


Figure 1.2: (**Top**) An example of two cameras observing a bisection task: the scissors are to intersect the midpoint of the leading edge of the paper. (**Middle, Bottom**) Two images of a probe touching a wire and two endpoints coinciding. All three of the intended tasks look accomplished in the left and right images simultaneously. In all of these cases, however, looks can be deceiving because of uncertainty in the cameras' locations. See Figure 1.3 .

In each of the left and right images these tasks look accomplished. Yet, because the cameras' locations are unknown, in each case the 3d world being imaged is not what it might seem. Figure 1.3 shows the scenes which produced the images of Figure 1.2 from viewpoints which reveal that those images do not guarantee that the intended tasks are satisfied.

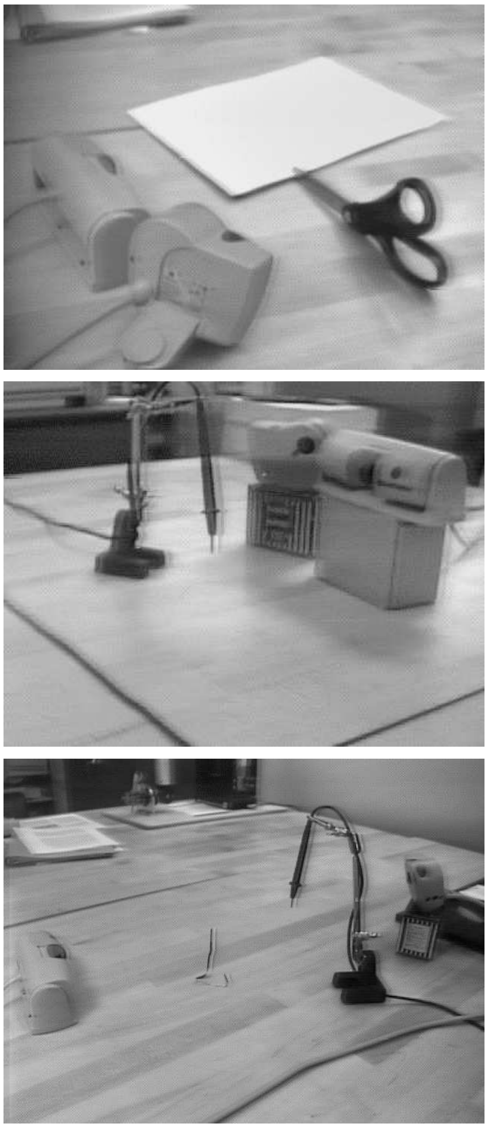


Figure 1.3: The scenes which gave rise to the images of Figure 1.2, shown from a revealing viewpoint. Each of the three intended tasks is not, in fact, satisfied.

These kinds of observations, as well as the desire to know conditions under which image information *does* guarantee the accomplishment of certain 3d tasks, motivate the work in this thesis. Through this exposition the examples of Figures 1.2 and 1.3 will appear in order to illustrate both theoretical points and practical concerns.

1.3 Theses

As suggested by Figures 1.1 and 1.2, certain physical tasks are not unambiguously specifiable with only limited available knowledge about the camera(s) observing those tasks. This dissertation provides a framework within which this notion of an "unambiguously specifiable" task (termed *decidable*) can be made precise.

Because the set of all such decidable tasks depends fundamentally on the level of certainty with which the observing cameras are known, several levels of certainty are explicitly considered. Attention is focused on stereo vision systems, although the framework presented applies to arbitrary sensor configurations. For four levels of camera certainty considered, the set of decidable tasks is fully characterized. These characterizations, in turn, underlie the following contributions:

- Algebraic rules are developed which can construct an explicit listing of all distinguishable (i.e., unambiguous) task-space configurations of features.

- Those explicit listings of configurations allow the development of concise languages for specifying any decidable task in terms of a few primitive skills and operators for combining them.

- These task-specification languages provide a means for automatically generating constraints on image information which, when satisfied, ensure that the intended 3d task is complete.

- An algorithm is presented which takes advantage of redundancy in task specification in order to lighten the burden on the supporting vision system.

Several demonstrations, implemented and tested in the context of robotic hand/eye coordination, validate these theoretical results. While a convenient testbed for experiments, robotic manipulation is only one of many applications for which the presented conclusions on task structure apply. Mobile robots exploring and navigating the world with a stereo rig or teams of agents viewing one another and their environment operate under the same constraints on visual task performance.

Thesis statement To summarize these contributions into a single thesis statement, this work advocates a principled approach to specifying tasks for robotic systems: one based in the distinguishing capabilities of the available sensors.

1.4 Outline

The following chapter begins by reviewing related research in order to place this thesis in the context of the large body of work on vision-based robot control. Additionally, Chapter 2 sets down conventions of notation and some of the mathematical background on which subsequent developments build. Chapter 3 then establishes the framework which formalizes the ideas of task, task specification, and decidable task — concepts which thus far have relied on an intuitive understanding. In addition, Chapter 3 details five levels of certainty for binocular vision systems by considering injective, uncalibrated projective, weakly calibrated, affine, and internally calibrated stereo rigs.

Chapters 4 and 5 present the theoretical heart of this thesis: characterizations of decidable tasks for four of the five considered sets of stereo cameras. The first characterization describes the invariance properties of decidable tasks; the second builds complete enumerations of these tasks one feature at a time. Chapter 6 carries these results a step further, presenting concise and complete languages for constructing any decidable task and its image-based specification.

Chapter 7 offers another application of Chapter 5's results: employing the redundancy in a task specification to inform the visual tracking system of the relative importance of particular tracked features. As a result, the vision system need not work perfectly to assure execution of certain tasks. The experiments of Chapter 8 validate the task specification languages presented in Chapter 6 and analyzed in Chapter 7. The thesis closes with a summary and concluding perspective on this work.

Chapter 2

Background

This thesis lies at the intersection of two fields of research: the geometry of stereo vision and visual control of robotic systems. This chapter limits its review of these widely studied fields to the content needed in this work. Within geometric vision this includes a synopsis of well-established theoretical results on which Chapters 3 to 7 build. From vision-based robotics an overview of influential systems — and their theoretical underpinnings — clarifies the primary difference in this work's objective: the analysis of system capabilities rather than their demonstration. Even so, those demonstrated systems inspire and support the application of visual servoing as a framework for performing tasks.

2.1 Notation

Throughout this thesis, $'$ denotes matrix transpose, $\{e_i\}$ is the canonical basis of $\mathbb{R}^m$, and $\mathbb{P}^m$ is m-dimensional projective space. Homogeneous coordinates identify points in $\mathbb{P}^m$ with 1d linear subspaces of $\mathbb{R}^{m+1}$. These homogeneous representations are equivalent up to nonzero scaling, e.g., for $\lambda \neq 0$, $\left[\begin{smallmatrix} u \\ v \\ w \end{smallmatrix}\right]$ and $\lambda \left[\begin{smallmatrix} u \\ v \\ w \end{smallmatrix}\right]$ represent the same point in $\mathbb{P}^2$.

In general, f and g denote distinguished feature points in task space ($\mathbb{P}^3$ or $\mathbb{R}^3$, as convenient), while y and m are points in image space ($\mathbb{P}^2$ or $\mathbb{R}^2$ for a single camera, $\mathbb{P}^2 \times \mathbb{P}^2$ or $\mathbb{R}^2 \times \mathbb{R}^2$ for two-camera systems). When needed, components of task space vectors will be uppercase $f = [X\ Y\ Z\ W]'$. Components of image space vectors are lowercase, e.g., if $y \in \mathbb{P}^2 \times \mathbb{P}^2$, $y = [y_1\ y_2]' = [u_1\ v_1\ w_1\ u_2\ v_2\ w_2]'$, where y_1 and y_2 are the images in the first and second camera, respectively. In general, m denotes a point in a metric image frame, while y denotes pixel coordinates, further described below and in Figure 2.1. Matrices appear in bold uppercase, e.g., a 3×4 camera projection matrix $\mathbf{P} = \mathbf{K}\,[\,\mathbf{R}\ \ t\,]$ with 3×3 factors $\mathbf{K}$ and $\mathbf{R}$ and three-component vector t.

The usual Euclidean scalar product of two vectors is denoted $v \cdot w$; $v \times w$ represents the cross product of vectors in $\mathbb{R}^3$ (or points in $\mathbb{P}^2$). $[v]_x$ is the 3×3 skew-symmetric matrix such that for any w, $[v]_x w = v \times w$. $\dot{y}$ equals the time derivative $\frac{\mathrm{d}y}{\mathrm{d}t}$. $\mathbf{I}_n$ represents the $n \times n$ identity matrix, and n will be omitted if clear from context.

2.2 The Geometry of Images

Effectively relating 2d image information with the 3d world drives the study of the geometric properties of vision. As Figure 2.1 shows, the pinhole camera models image formation as a projection of 3d points through a focal point C known as the camera center. Similar triangles then yield

$$x_c = \frac{X_c}{Z_c} \qquad \text{and} \qquad y_c = \frac{Y_c}{Z_c}, \tag{2.1}$$

where $m = \left[\begin{smallmatrix} x_c \\ y_c \\ 1 \end{smallmatrix}\right]$ and $f = [\, X_c \; Y_c \; Z_c \; 1\,]'$. Though nonlinear as a mapping from points in $\mathbb{R}^3$ to $\mathbb{R}^2$, when expressed in homogeneous coordinates Equation (2.1) becomes linear. Thus, $m = \mathbf{P}_c\, f$, or componentwise

$$\lambda \left[\begin{smallmatrix} x_c \\ y_c \\ 1 \end{smallmatrix}\right] = \left[\begin{smallmatrix} u_c \\ v_c \\ w_c \end{smallmatrix}\right] = \left[\begin{smallmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{smallmatrix}\right] \left[\begin{smallmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{smallmatrix}\right]. \tag{2.2}$$

In Equations (2.1) and (2.2) the mapping $\mathbf{P}_c$ assumes canonical coordinate systems for both world points and image points, depicted in Figure 2.1 (left).

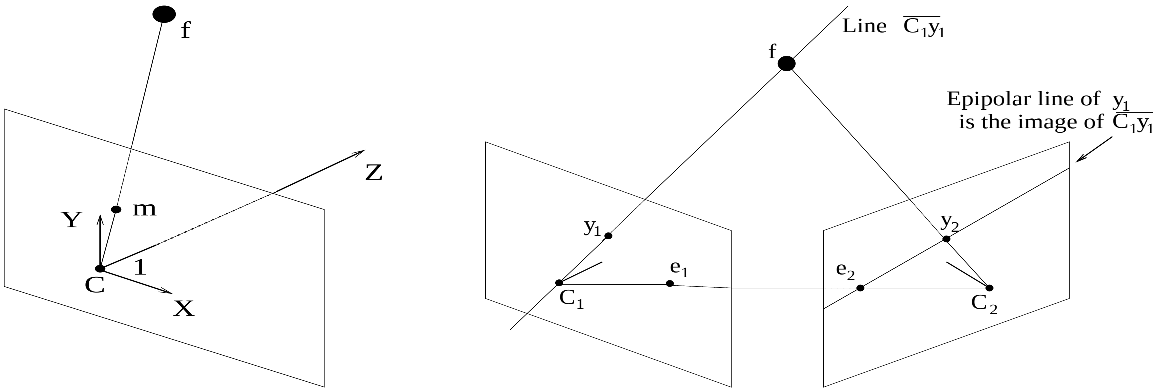


Figure 2.1: (**Left**) Projection in a pinhole camera. Suppose a world point f is expressed in a camera's *canonical (Euclidean) frame* with the indicated x, y, and z directions scaled to units of focal length and centered at the camera's focal point C. Then f's projection, y, satisfies Equations (2.1) and (2.2) when written in the *metric image frame* — identical to the camera's canonical frame. (**Right**) The epipolar constraint. Given the projection y_1 of a point f in an image, f's corresponding projection in a second image must lie along y_1's *epipolar line*, the projection of the line $\overline{C_1 y_1}$.

In general, world points might exist in any coordinate system rigidly displaced from the canonical one by a rotation $\mathbf{R}$ and translation t. In addition, image coordinates measured in pixels are modeled as an affine transformation $\mathbf{K}$ of the metric coordinates of Equations (2.1) and (2.2). Hence, in these more general frames image projection becomes

$$y = \mathbf{K}\,[\,\mathbf{R}\;\;t\,]\,f. \tag{2.3}$$

$\mathbf{R}$ and t hold six "external" parameters (the pose of the world frame with respect to the camera's canonical frame). $\mathbf{K}$ is an invertible, upper triangular matrix which models five "internal" camera parameters. Because f and y are homogeneous, the lower right entry of $\mathbf{K}$ can be scaled to 1. In this case

$$\mathbf{K} = \begin{bmatrix} a & b & c \\ 0 & d & e \\ 0 & 0 & 1 \end{bmatrix} \tag{2.4}$$

implies that (c, e) are the coordinates of the principal point, $\cot^{-1}(-\frac{b}{a})$ is the angle θ between the image's x and y axes, and there are a and $\frac{d}{\sin(\theta)}$ pixels in a single focal length in the x and y image directions, respectively. Cameras with known $\mathbf{K}$ are considered *internally calibrated*, and $\mathbf{K}$ provides the relationship

$$y = \mathbf{K}m \tag{2.5}$$

between m's metric image frame and y's *pixel image frame.*

Stereo

Unambiguously recovering world points from their projections requires more than a single image. Mutiple views have been employed on robotic systems since the Stanford Cart, with which Hans Moravec "broke the back" of the visual reconstruction problem with a translating camera [70]. The Cart, in line with most early systems, assumed the observing camera(s) were internally calbrated.

This thesis restricts its attention to the case of two simultaneously obtained views, e.g., from a stereo camera rig. Though a field of research unto itself, the correspondence between the two images of each task-space feature point is assumed available throughout this work. Figure 2.1 (right) depicts the basic constraint, the *epipolar constraint*, on points imaged in two views. Because a point f seen as y_1 in the first image must lie along the line between y_1 and that image's center C_1, the projection of f in the second image (y_2) can not simply appear anywhere in that image, but must lie along the projection of $\overline{C_1 y_1}$. Hence any point (from either image) gives rise to an *epipolar line* (in the other image) along which a corresponding image point must lie. The only exceptions are the *epipoles* e_1 and e_2, the projections of the cameras' baseline $\overline{C_1 C_2}$. Since the ray of every imaged point passes through a camera center, every epipolar line must pass through its image's epipole. Longuet-Higgins succinctly encapsulated this geometric constraint for two internally calibrated cameras as a single 3×3 matrix called the essential matrix [59].

The Calibrated Case

Suppose m_1 and m_2 are two images of a single point, expressed in the metric image frame of two internally calibrated cameras. The essential matrix $\mathbf{E}$ relates the two observed points according to

$$m_2' \mathbf{E} m_1 = 0. \tag{2.6}$$

Because m_1 and m_2 use homogeneous coordinates, Equation (2.6) defines $\mathbf{E}$ only up to a scale factor. One interpretation of this constraint is that $\ell = \mathbf{E} m_1$ is a representation of the epipolar line along which m_2 must lie. In fact, ℓ consists of the epipolar line's Plücker coordinates [20] – a parameterization in which a point p lies on the line if and only if $p \cdot \ell = 0$. Similarly, $\mathbf{E}' m_2$ yields the epipolar line of m_2 in the first image.

For an internally calibrated stereo rig the essential matrix captures all of the geometry available from only point corespondences. In fact, $\mathbf{E}$ decomposes as

$$\mathbf{E} = [t]_x\mathbf{R},$$

where $\mathbf{R}$ is the rigid rotation between the two cameras of the stereo rig and t is a vector in the direction of translation from one camera's center to the other's [20]. With $\mathbf{E}$ defined only up to a scale factor, the length of t can not be recovered. Hence $\mathbf{E}$ has five degrees of freedom: three in $\mathbf{R}$ and two in t. Researchers have exploited this structure to develop nonlinear algorithms finding $\mathbf{E}$ with as few as 5 point matches [22, 56] and linear methods using at least 8 matches [34, 59]. Related work has described the geometric configurations which defeat these algorithms or lead to ambiguities in the recovered essential matrix [44, 68].

The most important result regarding the essential matrix is that if $\mathbf{E}$ is known for an internally calibrated pair of cameras, the observed world can be reconstructed up to an unknown similarity, i.e., a scaled rigid displacement [21]. To see this, suppose that the first camera's projection matrix is

$$\mathbf{P_1} = [\mathbf{K_1} \;\; 0],$$

thus identifying the world coordinate system with the first camera's canonical frame. With a known $\mathbf{E} = [t]_x\mathbf{R}$, the second camera's projection matrix becomes

$$\mathbf{P_2} = [\mathbf{K_2}\mathbf{R} \;\; \mathbf{K_2}t],$$

where t (and the cameras' baseline) is scaled to unit length. $\mathbf{P_1}$ and $\mathbf{P_2}$ permit triangulation for the coordinates of any 3d point f, given images y_1 and y_2:

$$y_1 = \mathbf{P_1} f \quad \text{and} \quad y_2 = \mathbf{P_2} f \qquad \Rightarrow \qquad \mathbf{A} \left[\begin{smallmatrix} X \\ Y \\ Z \end{smallmatrix} \right] = b, \tag{2.7}$$

where $\mathbf{A}$ and b appear when the linear constraints are consolidated in terms of $f = [X\,Y\,Z\,1]'$. The resulting values express f's position in the first camera's canonical frame, scaled to accomodate a unit baseline. The actual coordinates of f in any world or object frame thus differ by at most a similarity, i.e., a rigid displacement and uniform scaling.

The Uncalibrated Case

Although many algorithms for the determination of a camera's internal parameters exist [38, 113], in general they require accurate knowledge about some feature points in the world, such as points on a calibration grid. With noise in the image extraction of those points [43, 54], the values recovered can vary considerably. In addition, internal parameters can change either through direct control of focus or zoom or indirectly by thermal or mechanical forces. For all of these reasons, research on the geometric properties of stereo rigs has swung away from the assumption of available internally calibrated cameras.

Suppose two cameras with unknown internal parameters observe a scene. The epipolar constraint depicted in Figure 2.1 continues to hold, but Equation (2.6) must now be expressed in homogeneous pixel image coordinates (y), rather than metric image coordinates (m):

$$y_2' \mathbf{F} y_1 = 0. \tag{2.8}$$

In this case the 3×3 matrix $\mathbf{F}$, termed the fundamental matrix [61], summarizes the geometric information available only from point correspondences without internal calibration. As with the essential matrix, $\mathbf{F}y_1$ and $\mathbf{F}'y_2$ represent y_1's epipolar line in the second image and y_2's epipolar line in the first image, respectively.

Compared to an internally calibrated stereo rig with known $\mathbf{E}$, a stereo rig with known $\mathbf{F}$ has considerable freedom in how it images the world. As a result, two-camera systems for which only the fundamental matrix is known are called *weakly calibrated* [90]. As with the essential matrix, research has uncovered accurate and efficient means for computing $\mathbf{F}$ from image information [34, 61, 112, 85, 119, 120]. Yet the most important application of knowing $\mathbf{F}$ was shown independently by Faugeras and Hartley in 1992: a weakly calibrated stereo rig can reconstruct an observed scene in the 3d world modulo some unknown projective transformation [19, 35].

Analogous to reconstruction up to a similarity using $\mathbf{E}$, projective reconstruction depends on estimating two projection matrices. If first camera effects the transformation

$$\mathbf{P_1} = [\,\mathbf{I} \;\; 0\,],$$

the world frame becomes a projective coordinate system attached to the first camera. Given a known $\mathbf{F}$, the epipoles e_1 and e_2 are the null spaces of $\mathbf{F}$ and $\mathbf{F}'$, respectively. To write a second camera's projection matrix consistent with $\mathbf{F}$ [61], let

$$\mathbf{P_2} = [\,[e_2]_x \mathbf{F}' \quad e_2\,].$$

Solving for the projective coordinates of $f = [Z\ Y\ Z\ W]'$ in

$$y_1 = \mathbf{P_1} f \quad \text{and} \quad y_2 = \mathbf{P_2} f \qquad \Rightarrow \qquad \mathbf{A} \begin{bmatrix} X \\ Y \\ Z \\ W \end{bmatrix} = 0, \tag{2.9}$$

which with real data requires finding the eigenvector of $\mathbf{A}$ with the smallest eigenvalue, e.g., by singular value decomposition. The resulting coordinates of f express that point's projective structure within the frame attached to the first camera; in any externally defined frame that structure will differ by at most a homography of $\mathbb{P}^3$. Of five reconstruction methods compared in [93], Equation (2.9) best maintained 3d accuracy in the presence of noise.

Other levels of calibration

Stereo vision thus provides different levels of geometric information about the 3d world, depending on the extent to which the parameters of the observing cameras are known. Faugeras presented this unified view in [21], in which he included the two examples of known $\mathbf{F}$ and known $\mathbf{E}$, as well as a third, called *affine calibration.* Although affine representations of an observed scene had been considered earlier [53, 110], that work relied on imaging models coarser than the pinhole camera, e.g., orthographic projection.

In short, two geometric objects are known for an affinely calibrated stereo rig: the fundamental matrix $\mathbf{F}$ and the *plane at infinity.* The plane at infinity consists of ideal points of $\mathbb{R}^3$, i.e., "directions" or vectors whose homogeneous coordinates' fourth component is a 0. With only weak calibration the 3d world can be recovered up to arbitrary projective transformations, under which the plane at infinity is indistinguishable from any other plane. Additional information, e.g., the images of three vanishing points or observed point translations, allow recovery of the projective coordinates a of the plane at infinity [14, 43]. With a known, projective structure from Equation (2.9) generates a scene's affine structure through

$$f_{\text{aff}} = \begin{bmatrix} \mathbf{I} & 0 \\ & a' \end{bmatrix} f_{\text{proj}}, \tag{2.10}$$

called the projective-affine upgrade [95].

Outside the community of research on geometric vision, some work has questioned the broad applicability of the pinhole, or projective, camera model [24]. This work will consider both the three well-studied levels of stereo calibration above, as well as two weaker levels of calibration: uncalibrated projective cameras, for which not even $\mathbf{F}$ is known, and stereo rigs assumed to be injective, but no more. To put Figure 1.2's three examples into perspective, the camera pairs which observe those scenes do not even guarantee injectivity, evidenced by the failure of the point-to-point task. Chapters 4 will prove minimum calibration levels needed to accomplish these tasks.

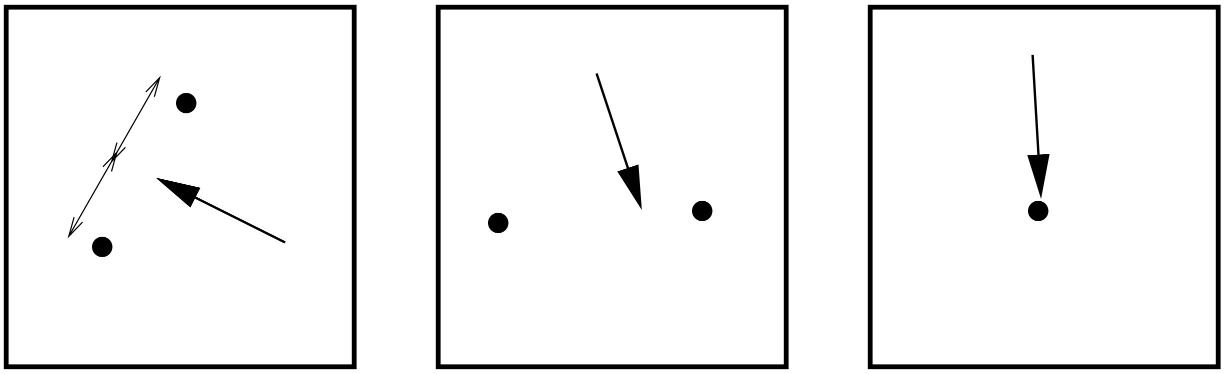


Figure 2.2: The three geometric tasks — midpoint positioning, collinearity, and coincidence — of Figure 1.2 phrased as image contraints on point features. Arrows' endpoints represent manipulation-controlled features; small circles indicate stationary references. Note that the collinearity task requires two distinguished points to specify the "finish line." Section 9.1 considers the relative expressiveness of pure line features.

Note that restricting attention to the properties of 3d point features, as this section has, does not rule out the consideration of any of Chapter 1's examples. Figure 2.2 depicts each of Figure 1.2's tasks in terms of point features alone. With two distinguished features on the wire, even the point-to-line task admits such a representation. Because of the expressiveness, widespread use, and theoretical simplicity of points, this thesis will consider only point features until Chapter 7, which explores some of the differences between point and line features, as well as other possible visual primtives.

One motivation for the various stereo rigs presented in this section lies in characterizing what tasks each level of calibration supports. In one sense, the reconstruction results of Equations (2.7), (2.9), and (2.10) answer this question. A task can not require a stereo rig to distinguish between two feature configurations which lie within that rig's reconstructive ambiguity, e.g., differing by only a similarity if **E** is known. On the other hand, stereo rigs *can* distinguish feature configurations which differ by more than the sensors' inherent uncertainty. Chapters 3 and 4 formalize these ideas and then builds on the reconstruction results of this section toward efficient languages for expressing tasks and compositional methods for automatic task execution.

2.3 Vision-based Robotics

For decades robotics has provided motivation for the field of computer vision. The overall goal of vision-guided robotics is general and robust object positioning via image information. Two broad — and increasingly overlapping — areas of robotics differ largely in the nature of the "object" positioned. Robot *navigation* concerns itself with the location of a self-contained, mobile platform, while tasks become robotic *manipulation* if a tool (a gripper or fixtured item) is the focus of attention. The development of control algorithms for navigation and manipulation testifies to their similarity: historically, work has progressed from almost completely independent vision and control to the very tightly coupled systems and hybrids seen today.

For instance, early vision-based navigators, such as Shakey and the Stanford Cart, used visual input to construct an internal representation of the 3d world as completely as possible. Afterward, a supervisory process planned appropriate actions and executed them blindly until stopping to repeat the cycle [70, 78]. On the other side of the spectrum (and more recently), the MIT robot Polly used low-level cues extracted at video frame rate in order to tour visitors around her stomping grounds [45]. Today sophisticated autonomous vehicles use both immediate image feedback to maintain a safe position, as well as visual indications from other cars to plan for future slowdowns or lane changes [16, 97].

Though applicable to a broad range of vision-based systems, the work of this thesis originated within the area of robotic manipulation. Figure 2.3 depicts one long-term objective of automatic hand/eye coordination. A pair of cameras observe a workcell which also includes a manipulator (or two...) and a number of items of interest. Such a cell would support tasks including diverse assembly, disassembly, and placement operations on known objects. As with robot navigation, "look-then-move" describes the traditional approach to manipulation. For example, HANDEY, a system by Grimson, Lozano-Perez, et al., used 3d reconstruction in order to locate a modeled part from an unstructured environment, select a grasp, plan for both robot and part motion, and then execute a specified positioning task [60]. As with other industrially motivated workcell systems, structured lighting such as laser striping simplified the recovery of 3d structure from visual information [69, 57].

If the relationships among workcell components were known perfectly, a task phrased in any of the cell's coordinate systems could be performed precisely with-

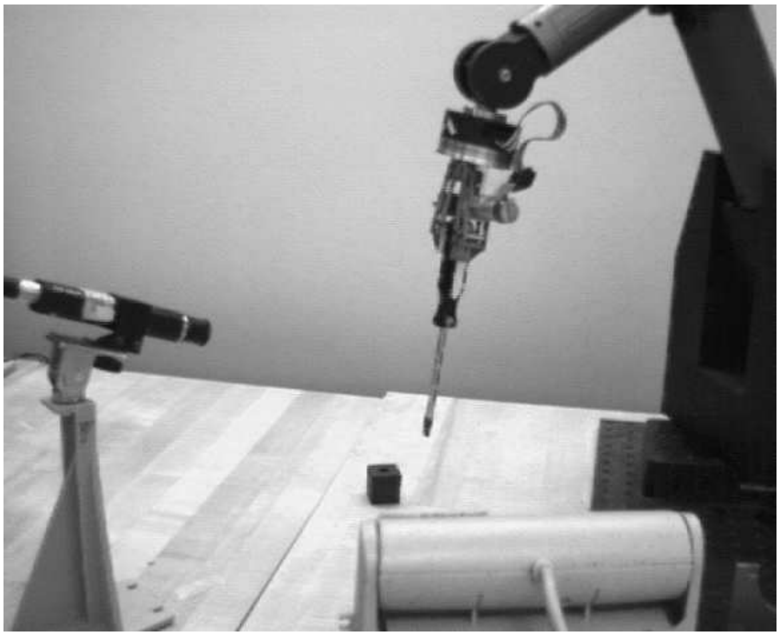


Figure 2.3: A modular workcell for robotic manipulation. Two cameras observe the workspace of a robot arm. The relationships among the parts of the workcell (cameras, arm, and objects of interest) may be known with varying levels of certainty. This level of "self-knowledge" affects the set of tasks the system can perform with precision. This set of tasks, in turn, provides a principled basis for specifying and monitoring workcell operations.

out recourse to the cameras. In many situations (mobile manipulation, modular setups such as Figure 2.3, time-delayed teleoperation, and systems handling poorly modeled objects) workcell calibration is not known with precision. The cameras fundamentally enable the system to tolerate *uncertainty* — uncertainty in the position of the manipulator or end-effector and uncertainty in the geometry and pose of the items to be manipulated. By observing both the destination and current configuration of visual features, direct image feedback or *visual servoing* can lead to precise task performance even without precise camera and object models, as outlined in the following section. At the same time, Section 2.2 has quantified how cameras can *introduce* uncertainty into a system if their relationship with each other, the task, or the manipulator is not fully known. Chapters 4 and 5 address these uncertainty tradeoffs by describing the tasks performable under a given level of camera uncertainty. The resulting constraint on performance grounds the principled approach to hand/eye system design of Chapters 6 and 7.

2.4 Visual Servoing

In contrast to look-then-move approaches, a great deal of research has been devoted to developing vision-based feedback control systems, i.e., visual servoing systems, which can accomplish positioning tasks without accurate estimates of camera calibration [2, 9, 11, 42, 47, 7, 114, 115, 118]. Weiss and Sanderson strengthened visual servoing as an area of research by classifying servo systems according to two criteria [99]. First, they distinguish between controllers which do and do not use joint feedback in the loop. The speed of today's joint controllers (up to several thousand hertz) compared to the relatively slow stream of images (usually 30 Hz) make this distinction rare in practice; most systems improve stability and performance through joint feedback. The second classification of servo systems describes the space in which task control is phrased. *Image-based servoing* achieves tasks by realizing a goal in image feature space; *position-based* systems compute and control the pose of an observed object. Though still used today, this distinction has blurred with the advent of hybrid servoing and visual tracking systems [9, 63].

With the problem well-defined by Sanderson and Weiss, research in visual servoing progressed with a focus on showing *how* specific tasks could be achieved. As an example, consider again the point-to-point task of Figure 1.2 (bottom). Because image-based approaches subsume position-based [39], assume an image-specified goal y^* and a current visual state y (see Figure 2.4). Define an image error $\Delta y = y^* - y$

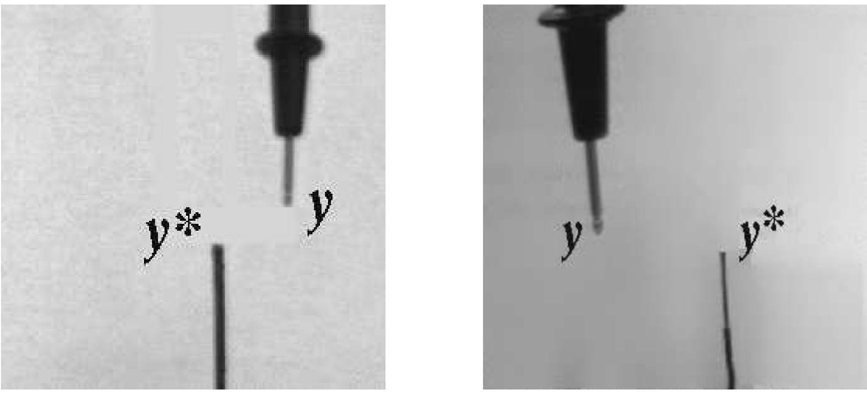


Figure 2.4: The point-to-point task revisited from a control standpoint. Suppose a manipulator controls the pose of the upper probe; y contains its image coordinates. The lower probe remains stationary; y^* hold the images' set points. Image-based visual servoing uses the observed error $y^* - y$ to generate a control signal which will reduce that error. Note that, as shown in Chapter 1, system convergence may not guarantee the accomplishment of the desired 3d constraint. (Figures 1.2 and 1.3)

which represents the desired change in image space. Presuming that a robot manipulator controls the downward-pointing probe, y is a function of the manipulator's pose r so that $y = y(r)$. Differentiating with respect to time yields

$$\frac{\mathrm{d}y}{\mathrm{d}t} = \frac{\mathrm{d}y}{\mathrm{d}r}\frac{\mathrm{d}r}{\mathrm{d}t}. \tag{2.11}$$

For any given pose the term $\frac{\mathrm{d}y}{\mathrm{d}r} = \mathbf{J}_r$ is $y(r)$'s Jacobian, the linear function mapping changes in robot state $\dot{r}$ to changes in visual state $\dot{y}$. Using the image error Δy as an estimate of the desired $\dot{y}$, the desired robot velocity obtains from (2.11):

$$\dot{r} = \mathbf{J}_r^+ \Delta y \ . \tag{2.12}$$

Here, $\mathbf{J}_r^+$ represents either a suitable inverse of $\mathbf{J}_r$ or merely that Equation (2.11) was solved for $\dot{r}$, e.g., through QR decomposition. Thus, with each computed visual error Δy Equation (2.12) generates a new robot control vector $\dot{r}$.

Control-level Considerations

Although something like Equation (2.12) represents a basic visual servoing loop, the details of its application provide numerous points of departure for investigation. For example, the form of $\mathbf{J}_r$ varies according to system configuration: either *eye-in-hand* with a camera attached to the end effector [6, 83, 88] or with externally mounted cameras [27, 30, 74]. More fundamentally, $\mathbf{J}_r$ depends on the parameterization of r, potentially consisting of individual joints [49, 84] or the twist coordinates representing a velocity in SE(3) [47, 73]. $\mathbf{J}_r^+$ typically includes a gain factor, but also can incorporate weighting matrices which reflect the certainty of the imaging model and/or the visual information available [77, 81, 83]. The features comprising Δy are often point image coordinates, as assumed here, but usable values include angles, lengths, region moments, and even optical flow parameters [13, 71]. Perhaps the widest variety of work has addressed the problem of finding $\mathbf{J}_r$. Techniques range from analytically derived expressions based on coarse camera calibration [10, 47, 76] or adaptive parameter estimation and control schemes [82, 102] to the use of canonical robot movements [96, 107] or online linear model-building [48, 84].

In this vein, it is worth noting that in order for any vision-based control system to succeed, the geometric relationship between the controlled system (in this case the robot manipulator) and the measuring apparatus (in this case the image planes of the two cameras) must be known either a priori or by online estimation with sufficient accuracy to guarantee stability of the closed-loop control system computed by Equation (2.12). In practice, this requirement is usually not difficult to guarantee [29, 31, 49] and therefore we take this as a given throughout. However, we emphasize that this does *not* presume we know this relationship with high accuracy. Thus, this information cannot be relied on to support, for example, reliable scene reconstruction. Put another way, calibration at this level directly affects a system's *dynamic performance*, but should not affect the system's *positioning precision*. Systems achieving dynamically-defined tasks [4, 80, 89, 117] often require far better estimates of vision/actuation relationships. This work restricts its focus to the large class of visually specified static positioning tasks [27, 42, 67, 114, 115, 118], though Chapter 7 suggests natural extensions to sequences of tasks and tasks involving trajectory constraints.

System-level Considerations

Visual servoing has supported demonstrations from industry-motivated filament placement [51], dragline control [92], and engine disassembly operations [111] to overarching applications such as scene exploration [64] and household tasks [30, 50]. The promise of the approach has spurred work in connecting servoing to higher-level considerations about tasks and task structure. One such consideration is planning, studied at the sensor level [1, 52, 76, 105], for robot trajectories [109], and at the task specification level [25, 50, 105]. A second consideration is the automatic evaluation of image features: both accuracy analyses and measures of feature "goodness," based on sensor and target configuration. Local techniques using the singular value decomposition of the linearized image/pose relationship include [23, 37, 74, 103]; a global performance measure is proposed in [108].

Many servoing demonstrations base their control strategies on the properties of the surrounding workcell and observing cameras. A third high-level consideration in constructing hand/eye systems is a modular framework for expressing tasks' structure. This objective has motivated research in task decomposition on a purely visual level [10, 50, 100], as well as in combination with joint-encoder information [7, 79] and with force feedback [71, 75]. In general, manipulation systems do exploit task-specific information in order to create set points and calculate a control signal. Yet the broader question of *what* can be done given a particular set of workcell uncertainties and constraints has received little, if any, attention. We use this question as a touchstone for developing easily programmable hand/eye systems which allow the composition of tasks from a small set of primitive skills.

Task Specification Languages

Several researchers have proposed systems for explicit task construction from simpler primitives. Schnackertz and Grupen, for instance, employ three control primitives and corresponding image constraints as a basis for visual homing tasks [100]. Morrow and Khosla start from a complete taxonomy of the constrainable degrees of freedom between two rigid bodies and progress to a partial list of vision- and force-based primitives for accomplishing those constraints [71]. Chaumette, Rives, and Espiau develop complete control details for several such DOF constraints, termed "virtual linkages," to form a library of visual servoing tasks [10]. Their framework handles points, lines, spheres, cylinders, and their combinations. The ViSP system [62] implements this library modularly, enabling composition through any desired conjunction of available primitives. It also permits the definition of secondary tasks which control degrees of freedom left unspecified by a primary visual specification [66]. However, ViSP commits at the control level to an eye-in-hand configuration, and few guarantees can be made regarding task performance [8]. In addition, all of these systems lack completeness, e.g., disjunctions of primitives are unavailable, and none consider the potential for visual ambiguity demonstrated by the examples of Chapter 1.

This thesis contributes concise and complete task-specification languages whose image-based error functions, when satisfied, guarantee that the desired task-space constraint is accomplished. Building from the "Graspomatic" and "XVision" compositional frameworks [30, 33], this thesis formalizes the ideas of task and task specification in order to derive these languages. Because each language is based on a workcell's fundamental sensing capabilities, the resulting tools for task-composition are provably complete, i.e., all tasks specifiable with point features lie within the span of the proposed languages.

One possible argument against this work's focus on how workcell uncertainty impacts task performability might stress that workcell calibration can be done online, obviating any constraints on performable tasks. Work such as [28, 48] provides methods for explicitly estimating workcell relationships from measured information to *some level of precision.* Although distinct enough to have been coined "camera-space manipulation" instead of visual servoing in [104], such approaches do not differ from servoing in their task specification [102]. Thus, the question addressed here — whether or not enough visual information is present to specify a task accurately — remains relevant. In addition, a designer presumably seeks systems which are both simple and robust. Minimizing the amount and/or degree of precision of calibration required for a system to function satisfies both criteria and hence should be a principal consideration for system design.

Chapter 3

Framework

This chapter formalizes the intuitive notions of task, task specification, and set of camera models as used thus far. As noted, attention is limited to the interaction of point features within a bounded workspace $\mathcal{W}$, considered a subset of an ambient space $\mathcal{X}$, either $\mathbb{P}^3$ or $\mathbb{R}^3$ as convenient. A stereo camera rig with a viewspace $\mathcal{V} \subset \mathcal{W}$ provides measurements of the image position of these features. To avoid trivial situations, assume that the viewspace contains some open set. Anticipating additional notation through this and subsequent chapters, $\mathcal{V}^n$ denotes the Cartesian product of a set $\mathcal{V}$ with itself n times. If $C : \mathcal{V} \to \mathcal{Y}$, the name C will continue to denote the function mapping $\mathcal{V}^n$ to $\mathcal{Y}^n$ which uses C componentwise. Bold, lowercase letters will indicate lists of points, delimited by braces: $\mathbf{f} = \{f_1 \dots f_n\}$ or $\mathbf{y} = \{y_1 \dots y_n\}$. (Recall that matrices use square brackets and uppercase names.)

3.1 Tasks

By a static positioning task or simply a "task" is meant, roughly speaking, the objective of bringing the 3d pose of a robot-controlled object to some "target" in

$\mathcal{W}$. Both the pose of the object under consideration and the target to which it is to be brought are determined by a list of simultaneously observed point features $\mathbf{f} = \{f_1, f_2, \ldots, f_n\}$ in $\mathcal{V}$. For succinctness, $\mathbf{f}$ will be called a *feature*. The admissible feature space is the set $\mathcal{V}^n$ of all such lists of interest.

As in [98, 47], tasks are represented mathematically using a *task function* T mapping $\mathcal{V}^n$ to $\{0, 1\}$. The *task* which T represents is the constraint

$$T(\mathbf{f}) = 0. \tag{3.1}$$

If (3.1) holds we say that the task is accomplished or *satisfied* at $\mathbf{f}$. Intuitively, the set of features $\mathbf{f}$ for which $T(\mathbf{f}) = 0$ constitute the desired poses of the manipulated object(s). T's codomain $\{0, 1\}$ abstracts the binary notion of satisfying or not satisfying a task specification, applications of which are this work's primary concern. In fact, only weak conditions on $\mathcal{V}$ and $\mathcal{Y}$ are needed to create a continuous task function from this abstraction [40]. Note that this definition of a task function — operating on sets points in the robotic workspace — differs from some of the visual servoing literature [9, 10, 18, 30, 47, 91], in which "task functions" take image information as input. Section 3.3 will address the cameras' rendering of Equation (3.1).

Tying this definition back to the example tasks of Chapter 1, recall that the bird's-eye views of Figure 1.3 revealed that the desired configurations, i.e. the tasks, were not actually satisfied even though the images of Figure 1.2 "looked right." This discrepancy underscores the 3d nature of a task: in the end, it is the 3d world that we would like to manipulate. After abstracting the 3d constraints as interactions of point features, as demonstrated in Figure 2.2, the formal definitions below can now supplement the original, more general descriptions of these three example tasks.

- **Midpoint Task**: Let $T_{\text{midpt}} : \mathcal{V}^3 \to \{0,1\}$ by the rule

$$\mathbf{f} = \{f_1, f_2, f_3\} \longmapsto \begin{cases} 0 & f_1 \text{ is the midpoint of } f_2 \text{ and } f_3 \\ 1 & \text{otherwise} \end{cases} .$$

  The constraint $T_{\text{midpt}}(\mathbf{f}) = 0$ describes the desired 3d configuration of features for the paper-cutting task of Figure 1.2.

- **Collinearity Task**: The collinearity task function, T_{col}, has domain $\mathcal{V}^3$ and maps features according to

$$\mathbf{f} = \{f_1, f_2, f_3\} \longmapsto \begin{cases} 0 & \text{the } f_i\text{'s are all collinear in } \mathbb{P}^3 \\ 1 & \text{otherwise} \end{cases} .$$

  This task is satisfied exactly when the feature points of interest lie on a single line; in fact, any two or even all three may be coincident. Chapter 6 formalizes the task accomplished when three feature points lie on a line and no two coincide.

- **Point-to-point Task**: Let T_{pp} be the task function defined on $\mathcal{V}^2$ by the rule

$$\mathbf{f} = \{f_1, f_2\} \longmapsto \begin{cases} 0 & f_1 \text{ and } f_2 \text{ are the same point in } \mathbb{P}^3 \\ 1 & \text{otherwise} \end{cases} .$$

  The point-to-point task is satisfied, $T_{\text{pp}}(\mathbf{f}) = 0$, exactly when $\mathbf{f}$'s two feature points are coincident.

3.2 Camera Models

Two cameras — a *camera model* or *stereo rig* — observe the features of interest on workcell objects and provide all of the information available (to a purely vision-based system) for task performance. Denote by C_{actual} the actual stereo rig observing $\mathcal{W}$ and assume that the component cameras within C_{actual} have a joint field of view $\mathcal{V}$ containing an open subset of $\mathcal{X}$. C_{actual} maps points from $\mathcal{V}$ to feature points in the two-camera image space $\mathcal{Y} \subset \mathbb{P}^2 \times \mathbb{P}^2$ or $\mathbb{R}^2 \times \mathbb{R}^2$ (the two image planes). Thus, for feature points and features respectively

$$y = C_{\text{actual}}(f) \quad \text{and} \quad \mathbf{y} = C_{\text{actual}}(\mathbf{f}). \tag{3.2}$$

In an uncalibrated environment, C_{actual} is unknown. To represent the extent of a system's uncertainty about C_{actual}, a known set of camera models $\mathcal{C}$ is assumed to contain C_{actual}. In $\mathcal{C}$ each camera model $C : \mathcal{D} \to \mathcal{Y}$ "sees" all points in the view-space $\mathcal{V}$, i.e., $\mathcal{V} \subset \mathcal{D}$, C's domain.

Chapter 2 introduced several sets of stereo rigs, distinguished by the level of calibration known about each. Such sets offer a precise description of the potential uncertainty affecting the imaging process. While this chapter's framework applies to any $\mathcal{C}$, five sets of camera models are considered in detail through this thesis:

1. **Injective Cameras** Let $\mathcal{C}_{\text{inj}}[\mathcal{V}]$ denote the class of all injective functions C such that $C : \mathcal{D} \to \mathcal{Y}$ with $\mathcal{V} \subset \mathcal{D}$. This class is broad enough to model stereo rigs with lens distortions, panoramic cameras, and even many nonvisual sensors. For projective camera models (Equation (2.2)) injectivity demands that the stereo rig's baseline lie outside $\mathcal{V}$.

2. **Uncalibrated (Projective) Cameras** Define $\mathcal{C}_{\text{uncal}}[\mathcal{V}]$ to be the set of all

uncalibrated (projective) two-camera models which are injective on $\mathcal{V}$. Each component camera of a stereo rig in $\mathcal{C}_{\mathrm{uncal}}[\mathcal{V}]$ can be modeled by a full-rank 3×4 matrix which maps points in $\mathcal{V}$ to points in an image plane according to Equation (2.3). As noted, to preserve injectivity $\mathcal{V}$ may not contain the cameras' baseline. The requirement of injectivity implies that uncalibrated camera models are a subset of injective camera models.

3. **Weakly Calibrated Cameras** The symbol $\mathcal{C}_{\mathrm{proj}}[\mathcal{V}, \mathbf{F}]$ denotes the subset of $\mathcal{C}_{\mathrm{uncal}}[\mathcal{V}]$ whose stereo rigs have fundamental matrix $\mathbf{F}$ (Equation (2.8)). Consider the number of degrees of freedom lost in the transition from $\mathcal{C}_{\mathrm{uncal}}[\mathcal{V}]$ to $\mathcal{C}_{\mathrm{proj}}[\mathcal{V}, \mathbf{F}]$. Since uncalibrated cameras have 22 parameters (eleven for each of two independent applications of Equation (2.3)) and $\mathbf{F}$ has seven degrees of freedom (the nine components of matrix constrained to be singular and defined up to a scale factor), elements of $\mathcal{C}_{\mathrm{proj}}[\mathcal{V}, \mathbf{F}]$ must have fifteen dimensions — exactly the number in a linear transformation of $\mathbb{P}^3$. The subscript $_{\mathrm{proj}}$ conveys both the projective mapping of each component camera and the fact that weakly calibrated stereo rigs allow reconstruction up to some linear projective transformation of task space.

4. **Affinely Calibrated Cameras** Affine camera models, $\mathcal{C}_{\mathrm{aff}}[\mathcal{V}, \mathbf{F}, a]$, are weakly calibrated projective cameras with a given plane at infinity, a. Since a is a four-vector defined up to scale, affine stereo rigs have twelve free parameters and permit scene recovery up to an unknown 3d affine transformation of features' coordinates (Equation (2.10)).

5. **Internally Calibrated Cameras** The most contrained set of camera models considered, called *perspective* cameras for short, are internally calibrated cameras ($\mathbf{K}_1$ and $\mathbf{K}_2$ known) with a particular essential matrix $\mathbf{E}$ (Equa-

tion (2.6)). $\mathcal{C}_{\text{persp}}[\mathcal{V}, \mathbf{E}, \mathbf{K}_i]$ denotes this set. Because the known quantities $\mathbf{E}$ and $\mathbf{K}_i$ uniquely determine both the fundamental matrix and the plane at infinity, each of the more widely defined sets above contain several sets of perspective cameras.

The balance of this thesis presumes that the cameras' viewspace $\mathcal{V}$, an arbitrary fundamental matrix $\mathbf{F}$, essential matrix $\mathbf{E}$, plane at infinity a, and sets of internal camera parameters $\mathbf{K}_i$ are understood and will suppress those parameters when unneeded or clear from context. Since each of these levels of calibration subsumes all of its predecessors, the above camera models form the following hierarchy:

$$\mathcal{C}_{\text{inj}} \supset \mathcal{C}_{\text{uncal}} \supset \mathcal{C}_{\text{proj}} \supset \mathcal{C}_{\text{aff}} \supset \mathcal{C}_{\text{persp}} \supset \{ C_{\text{actual}} \}. \tag{3.3}$$

Returning to the examples of Figure 1.2, the cameras observing those scenes span more of the set of all possible imaging functions that *any* of the above sets of camera models. The failure of the point-to-point task, for example, demonstrates a noninjective task-space to image-space mapping which must fall outside of the entire hierarchy of (3.3). The following sections (and chapters) explore the interplay between the amount of calibration available in a hand/eye system and the tasks which that system can visually specify.

3.3 Task Specifications

A task specification mediates between the 3d world of tasks and the 2d worlds of the cameras' image planes: a specification attempts to capture a task's geometric constraint in terms of the image features seen by the system. A specification is thus termed an *image-based task encoding*, or simply *encoding*, because it intends to translate the set of desired task-space feature configurations into a corresponding set of image-space feature configurations. Formally, an *encoded task function* E maps $\mathcal{Y}^n$ to $\mathbb{R}^m$, and the constraint $E(\mathbf{y}) = 0$ is an image-based task encoding. As such, encodings resemble the "task functions," i.e., image-defined error functions, of much of the servoing literature [9, 10, 18, 30, 47, 91].

With this definition the three task encodings used (unsuccessfully) to translate Chapter 1's example tasks into the image constraints of Figure 1.2 become

- **Midpoint Task**: The encoding for T_{midpt} implied by Figure 1.2 (top) is $E_{\text{midpt}}(\mathbf{y}) = 0$, with $\mathbf{y} = \{y_1, y_2, y_3\}$, each $y_i \in \mathbb{R}^2 \times \mathbb{R}^2$, and E_{midpt} defined by

$$\mathbf{y} = \{y_1, y_2, y_3\} \longmapsto \; y_1 - \frac{1}{2}(y_2 - y_3).$$

  That is, in each image the Cartesian coordinates of the manipulated feature point must equal the average of the coordinates of the statinoary feature points. Note that the codomain of E in this case is $\mathbb{R}^4$.

- **Collinearity Task**: A natural encoding for the collinearity task, used in Figure 1.2 (middle), is $E_{\text{col}}(\mathbf{y}) = 0$, with $\mathbf{y} = \{y_1, y_2, y_3\}$, each y_i in $\mathbb{P}^2 \times \mathbb{P}^2$, and E_{col} defined by

$$\mathbf{y} = \{y_1, y_2, y_3\} \longmapsto \begin{bmatrix} y_1^1 \cdot (y_2^1 \times y_3^1) \\ y_1^2 \cdot (y_2^2 \times y_3^2) \end{bmatrix}. \tag{3.4}$$

  The expressions y_i^1 and y_i^2 refer to the first and second cameras' image points of the two-camera feature y_i. In geometric terms this encoding is satisfied when the three points, y_1^j, y_2^j, and y_3^j, are collinear in each image ($j = 1, 2$). E_{col}'s codomain is $\mathbb{R}^2$.

- **Point-to-point Task**: An encoding for the point-to-point task is $E_{\text{pp}}(\mathbf{y}) = 0$ with $y = \{y_1, y_2\} \in \mathcal{Y}^2$, $y_i \in \mathbb{R}^2 \times \mathbb{R}^2$, and E_{pp} defined by

$$\mathbf{y} = \{y_1, y_2\} \longmapsto \; y_1 - y_2.$$

  Only configurations in which two image feature points are coincident in both component cameras satisfy this encoding. As with E_{midpt}, E_{pp} has codomain $\mathbb{R}^4$.

Note that while the codomain of tasks is the binary set $\{0, 1\}$, the codomain of task encodings is $\mathbb{R}^m$. This distinction anticipates the need to use image information for robot control. Ideally, an encoding E is continuous, differentiable (as the examples above), or even "better" behaved, so that during the execution of a task the values obtained from E can smoothly guide an observed feature configuration into a desired one where $E = 0$. The control problem of satisfying such image-based error functions has received considerable attention [2, 7, 9, 8, 11, 12, 42, 47, 55, 114, 115, 118].

More fundamental to task accomplishment, however, is the connection between a task encoding and the desired task-space configurations it tries to capture. All three of the encodings informally depicted in Figure 1.2 and defined mathematically above do *not* guarantee performance of the tasks they encode (Figure 1.3). The next section defines this critical property that a task encoding's accomplishment ensure the underlying task's accomplishment.

3.4 Task Decidability

Intuitively, the image information provided by, say, an internally calibrated stereo rig reveals more about the observed 3d configuration than the same information would if less were known about the cameras. The different levels of scene reconstruction described in Section 2.2 demonstrate the same dependence on the extent of camera-calibration knowledge. As a result, any connection between task-space constraints and image-space ones necessarily involves the uncertainty of the observing cameras. To formalize this connection, an encoding $E(\mathbf{y}) = 0$ is said to *verify* a task $T(\mathbf{f}) = 0$ on a set of camera models $\mathcal{C}$ if

$$\forall C \in \mathcal{C} \;\; \forall \mathbf{f} \in \mathcal{V}^n \quad E(\mathbf{y}) = 0 \Leftrightarrow T(\mathbf{f}) = 0, \tag{3.5}$$

where $\mathbf{y} = C(\mathbf{f})$. In turn, a task is *verifiable* or *decidable* on a set of camera models $\mathcal{C}$ if there exists some encoding which verifies that task on $\mathcal{C}$.

The notion of task decidability singles out precisely those tasks whose accomplishment (or lack thereof) can be deduced from measured data, without regard to particular encodings which verify them. Thus, this thesis treats decidability as the formal definition of "specifiable" or "performable" tasks, as considered informally in the introduction. Put another way, the essential characteristic of tasks decidable on $\mathcal{C}$ is the ability of the camera models in $\mathcal{C}$ to distinguish task success from task failure. The following proposition expresses this intuition mathematically.

Proposition 1 *T is decidable on $\mathcal{C}$ if and only if*

$$\forall C_1, C_2 \in \mathcal{C} \;\; \forall \mathbf{f}, \mathbf{g} \in \mathcal{V}^n \qquad T(\mathbf{f}) \neq T(\mathbf{g}) \implies C_1(\mathbf{f}) \neq C_2(\mathbf{g}) \tag{3.6}$$

Proof of Proposition 1: If configuration $\mathbf{f}$ satisfies a decidable task, but $\mathbf{g}$ does not, then no two cameras in $\mathcal{C}$ can image $\mathbf{f}$ and $\mathbf{g}$ in the same way. After all, if $C_1(\mathbf{f}) = C_2(\mathbf{g})$, the system could not tell whether it was seeing $\mathbf{f}$ through C_1 or $\mathbf{g}$ through C_2, since the image information is identical in each case. No deterministic algorithm (or encoding function) could then distinguish task success, $\mathbf{f}$, from task failure, $\mathbf{g}$.

To argue the "if" direction of implication, suppose the latter statement of the proposition. Define an encoding function E such that

$$E(\mathbf{y}) = 0 \quad \text{if} \quad \exists \mathbf{f} \in \mathcal{V}^n \;\; \exists C \in \mathcal{C} \;\; \text{s.t.} \;\; \mathbf{y} = C(\mathbf{f}) \;\; \text{and} \;\; T(\mathbf{f}) = 0 \; ,$$

otherwise $E(\mathbf{y}) = 1$. It remains to show that E is well-defined and that E verifies T. To demonstrate the former, consider any two visible features $\mathbf{f}$ and $\mathbf{g}$ and any two stereo rigs in $\mathcal{C}$, C_1 and C_2. Suppose $\mathbf{y} = C_1(\mathbf{f}) = C_2(\mathbf{g})$; for E to be well-defined, $E(\mathbf{y})$ must have a consistent definition for both means of observing $\mathbf{y}$. The assumption of Equation (3.6) guarantees that $T(\mathbf{f}) = T(\mathbf{g})$, so that $E(\mathbf{y}) = T(\mathbf{f}) = T(\mathbf{g})$. Thus, E is well-defined and $E(\mathbf{y}) = 0$ verifies $T(\mathbf{f}) = 0$. ■

While Proposition 1 will help characterize the decidable tasks on several sets of camera models, note that it does not offer usable task encodings for those decidable tasks. The E constructed in the proof lacks even continuity and so is wholly unsuitable as a basis for a robot controller.

3.5 A Task Hierarchy

Using the notation $\mathcal{T}_{\mathrm{x}}$ to represent the set of tasks decidable on $\mathcal{C}_{\mathrm{x}}$, Equation (3.3) and the universal quantifier in (3.5) imply that

$$\mathcal{T}_{\mathrm{inj}} \subset \mathcal{T}_{\mathrm{uncal}} \subset \mathcal{T}_{\mathrm{proj}} \subset \mathcal{T}_{\mathrm{aff}} \subset \mathcal{T}_{\mathrm{persp}} . \tag{3.7}$$

This task hierarchy merely formalizes the intuition that additional knowledge about an imaging system should increase the set of tasks that system can distinguish.

To make this formalism of decidable tasks and verifying encodings more concrete, we return to the three examples of Figure 1.2. Simply stated, none of the three encodings (described in Section 3.3) verify the tasks (defined mathematically in Section 3.1) on the very large set of camera models allowed to view the scenes of Figure 1.3. In fact, since the pictures in Figure 1.2 *could have* imaged configurations which satisfied each task, Proposition (1) shows that not one of these three tasks is decidable on the allowed set of camera models. Yet with more information about the observing cameras each of these tasks *is* decidable. The next chapter establishes how high each task sits in the hierarchy of (3.7).

Chapter 4

Characterizing Decidable Tasks

The hierarchy of task sets (3.7) begs the question of a simply stated characterization of each $\mathcal{T}_x$. This chapter presents one such characterization — in terms of tasks' invariance to a particular group of transformations. These invariance-based descriptions, in turn, will lead to a complete pointwise enumeration of all decidable feature configurations in Chapter 5.

The reconstruction results of Chapter 2 hold the key to characterizing decidable tasks. For the sets of camera models presented, the ambiguity in scene reconstruction turns out to be exactly the extent of invariance decidable tasks must exhibit. Before presenting formal definitions and proving this true, consider the following informal argument for the case of weakly calibrated stereo rigs.

Suppose a task is decidable on a set of weakly calibrated cameras, $\mathcal{C}_{\text{proj}}$. Then, if feature configurations $\mathbf{f}$ and $\mathbf{g}$ are projectively equivalent (one is a projective transformation of the other), there may be distinct stereo rigs C_1 and C_2 in $\mathcal{C}_{\text{proj}}$ such that the image of $\mathbf{f}$ in C_1 is exactly the image of $\mathbf{g}$ in C_2. If $\mathcal{C}_{\text{proj}}$ is large enough, in fact, such camera models C_1 and C_2 will exist. As a result, with $T(\mathbf{f}) = 0$ decidable on $\mathcal{C}_{\text{proj}}$, T will have to have the same value (0 or 1, satisfied or unsatisfied) on both $\mathbf{f}$ and $\mathbf{g}$. After all, there is no way for T's verifying encoding to "tell" from image information if it is looking at $\mathbf{f}$ through C_1 or $\mathbf{g}$ through C_2 — so, it must make the same satisfied/unsatisfied decision on each. In other words, T will have to be invariant to projective transformations.

In the other direction, suppose T is a task function which is invariant under projective transformations. Given that a weakly calibrated stereo rig observes the scene, some "reconstruction function" of image features which outputs a projective transformation of the scene's structure must exist. An encoding which verifies T, then, is T itself composed with this projective transformation and the reconstruction function — a composition which yields exactly $T(\mathbf{f})$ from input $\mathbf{y}$.

While this argument — that reconstructability defines the invariance properties of decidable tasks — lacks rigor, it captures the essence of the more formal reasoning provided in the balance of this chapter. The next three sections motivate mathematical definitions, and a more straightlaced treatment of the above "reconstructibility/task invariance" claim follows.

4.1 Task Invariance

The above argument hints at the formal definition for "task invariance." In essence, an invariant task has an invariant task function, i.e., one which does not change its value under certain transformations of its input. Formally, a task function T and the task T represents are called *$\mathcal{G}$-invariant* if for a group of transformations $\mathcal{G}$ which operate on $\mathcal{X}$,

$$\forall \mathbf{f} \in \mathcal{V}^n \quad \forall g \in \mathcal{G} \text{ s.t. } g(\mathbf{f}) \in \mathcal{V}^n \qquad T(\mathbf{f}) = T(g(\mathbf{f})). \tag{4.1}$$

If $\mathcal{G} = \mathrm{GL}(4)$, the set of invertible linear transformations of $\mathbb{P}^3$, tasks invariant to elements of $\mathcal{G}$ are *projectively invariant*; appropriate subgroups presented in Sections 5.5 and 5.6 define *affine invariance* and *similarity-invariance* in the same way. *Bijectively invariant* describes tasks invariant to all bijections of $\mathcal{X}$, of which GL(4) is a subgroup.

What tasks fall under these categories? Consider the three task examples from Chapter 1 whose formal definitions appear in Section 3.1. Since bijections preserve point-coincidence relations, the point-to-point task function T_{pp} is bijectively invariant, as is the task $T_{\mathrm{pp}}(\mathbf{f}) = 0$. Projective transformations maintain coincidence and collinearity (and coplanarity and cross-ratios, for that matter), so that the collinearity task $T_{\mathrm{col}}(\mathbf{f}) = 0$ is projectively invariant. The midpoint task $T_{\mathrm{midpt}}(\mathbf{f}) = 0$ is affinely invariant and similarity-invariant, as both preserve Euclidean length ratios: the latter uniformly in all directions, the former nonuniformly.

4.2 Reconstructability

Proving the "reconstructability/task invariance" hypothesis necessitates a mathematical description of reconstructibility. The results summarized in Equations (2.7), (2.9), (2.10) suggest that reconstructability provides a mapping from image information back to the original task space, $\mathcal{X}$. Thus, a set of camera models $\mathcal{C}$ provides scene reconstruction up to a group of transformations $\mathcal{G}$ if

$$\exists r_{\mathcal{C}} : \mathcal{Y}^n \to \mathcal{X}^n \;\; \forall C \in \mathcal{C} \;\; \forall \mathbf{f} \in \mathcal{V}^n \;\; \exists g \in \mathcal{G} \;\text{ such that }\; \mathbf{f} = g(r_{\mathcal{C}}(\underbrace{C(\mathbf{f})}_{\mathbf{y}})) \,. \tag{4.2}$$

The function $r_{\mathcal{C}}$ is the reconstruction function, mapping image features to scene structure in the appropriate ambient space $\mathcal{X}$. For any possible observing stereo rig, that structure is identical to the original observed scene $\mathbf{f}$ up to some (unknown) transformation $g \in \mathcal{G}$.

While (2.9), (2.10), and (2.7) define $r_{\mathcal{C}}$ for C_{proj}, C_{aff}, and C_{persp}, respectively, the balance of this thesis will also consider reconstruction up to a bijection. To exercise the above definition of reconstructability, the following will present a proof that injective camera models provide scene reconstruction up to a bijection.

Let $\mathcal{C} = \mathcal{C}_{\text{inj}}$ and let $\mathcal{G}_{\text{bij}}$ represent the group of bijections of $\mathbb{R}^3$. Define a "first appearance function" i_{fa} which takes a list of points and outputs a list of the index of the first appearance of each point. For instance, if $\mathbf{y} \in \mathcal{Y}^6$ and each $y_i \in \mathbb{R}^2 \times \mathbb{R}^2$ with

$$\mathbf{y} \quad = \quad \{\, y_1\ y_2\ y_3\ y_4\ y_5\ y_6 \,\} \quad = \quad \left\{ \begin{smallmatrix} 0 & 0 & 1 & 0 & 2 & 1 \\ 0 & 1 & 1 & 1 & 2 & 1 \\ 0 & 2 & 1 & 2 & 2 & 1 \\ 0 & 3 & 1 & 3 & 2 & 1 \end{smallmatrix} \right\} \ ,$$

then $i_{\text{fa}}(\mathbf{y}) \quad = \quad \{\, 1\ 2\ 3\ 2\ 5\ 3 \,\}$. With $\mathbf{y}$ given as context, this equation also defines the individual indices $i_{\text{fa}}(y_i)$. For example, $i_{\text{fa}}(y_2) = i_{\text{fa}}(y_4) = 2$ and $i_{\text{fa}}(y_3) = i_{\text{fa}}(y_6) = 3$. Next, for $\mathbf{y} \in \mathcal{Y}^n$ define $r_{\mathcal{C}}$ by

$$r_{\mathcal{C}}(\mathbf{y}) = \mathbf{f} \quad \text{with} \quad f_i = [\, i_{\text{fa}}(y_i)\ 0\ 0\,]' \ . \tag{4.3}$$

Finally, let $C \in \mathcal{C}_{\text{inj}}$ and $\mathbf{f} \in \mathcal{V}^n$ be arbitrary, with $\mathbf{y} = C(\mathbf{f})$. It remains to show an element $g \in \mathcal{G}_{\text{bij}}$ exists such that $\mathbf{f} = g(r_{\mathcal{C}}(\mathbf{y}))$. The bijection $g : \mathbb{R}^3 \to \mathbb{R}^3$ defined by

$$h \ \longmapsto \begin{cases} f_i & \text{if } \ h = \left[\begin{smallmatrix} i \\ 0 \\ 0 \end{smallmatrix}\right] \\ \left[\begin{smallmatrix} i \\ 0 \\ 0 \end{smallmatrix}\right] & \text{if } \ h = f_i \\ h & \text{otherwise} \end{cases} \ .$$

suffices; it is well-defined because C is guaranteed to be injective. In addition, g is seen to be bijective, because it is the identity transformation except at the points f_i and $\left[\begin{smallmatrix} i \\ 0 \\ 0 \end{smallmatrix}\right]$, which map to each other. Hence injective camera models do allow for scene reconstruction up to an unknown bijective transformation. ■

In one sense the above result seems one-sided: sets of injective cameras do provide reconstruction up to a bijection, but only some bijections arise from that reconstructive process. In a less stringent sense, however, completeness does hold: all of the sets of camera models thus far considered for their reconstructive capacity have been in some sense as large as they could be without losing that capacity. For example, $\mathcal{C}_{\text{inj}}$ contains *all* injective mappings from $\mathcal{V}$ to $\mathcal{Y}$. The reconstructability/task invariance result this chapter intends to show depends on this "completeness" property. The following section provides the formal machinery behind it.

4.3 Complete Sets of Camera Models

Recall that the informal reconstructability/task invariance argument relies on the *size* of the set of weakly calibrated cameras under consideration. This section makes that requirement precise. To motivate the definition of a reconstructively complete set of camera models, consider the following incomplete projective camera-model set $\mathcal{C}$:

$$\begin{aligned} \{ \quad \{ \mathbf{P}_1 = \left[\begin{smallmatrix} 1 & 0 & 0 & -2 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{smallmatrix}\right], \mathbf{P}_2 = \left[\begin{smallmatrix} 1 & 0 & 0 & -1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{smallmatrix}\right] \}, \quad & \quad (4.4) \\ \{ \mathbf{P}_1 = \left[\begin{smallmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{smallmatrix}\right], \quad \mathbf{P}_2 = \left[\begin{smallmatrix} 1 & 0 & 0 & 2 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{smallmatrix}\right] \} \quad \}. & \quad (4.5) \end{aligned}$$

$\mathcal{C}$ consists of only two stereo pairs, shown in Figure 4.1. Certainly these two camera models share the same viewspace and fundamental matrix. Thus, using image information from either of the stereo rigs in $\mathcal{C}$ and Equation (2.9), any configuration of

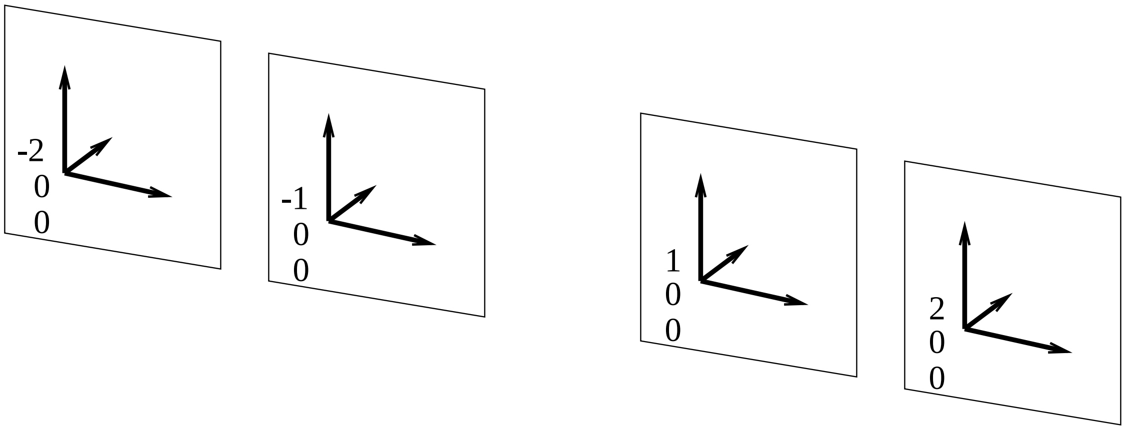


Figure 4.1: Four cameras comprising the two stereo rigs in a set $\mathcal{C}$. Their centers' coordinates differ only in the x component, as labeled. Both pairs of cameras have identical fundamental matrices $\mathbf{F}$, so that they provide reconstruction of scene points up to some projective transformation. Hence projectively invariant tasks are decidable on $\mathcal{C}$. Yet because much more than $\mathbf{F}$ is known about the stereo rigs in $\mathcal{C}$, projective invariance is sufficient but not necessary for task decidability on $\mathcal{C}$. If $\mathcal{C}$ included all other injective stereo rigs with the same $\mathcal{V}$ and $\mathbf{F}$, projective invariance does become necessary for task decidability. In this case $\mathcal{C}$ is considered *projectively complete*.

feature points in the viewspace can be recovered up to a projective transformation.

However, because $\mathcal{C}$ only contains two camera models, much more than the projective structure of a scene can be deduced from image information. In fact, exact 3d scene structure can be recovered up to a single binary ambiguity: either the reconstruction is correct or it is off by three units in the x direction of translation. Certainly projectively invariant tasks are only a small, proper subset of the decidable tasks on this $\mathcal{C}$; for that matter, so are affinely invariant and similarity-invariant tasks. The problem lies in the fact that many more camera models could be added to $\mathcal{C}$ without its losing the capability to recover scene structure up to a projective transforamtion.

To handle this problem, this thesis considers only reconstructively complete sets of camera models, where $\mathcal{C}$ is reconstructively $\mathcal{G}$-*complete* for a group $\mathcal{G}$ if

$$\forall g \in \mathcal{G} \quad \forall \mathbf{f} \in \mathcal{V}^n \text{ s.t. } g(\mathbf{f}) \in \mathcal{V}^n \quad \exists C_1, C_2 \in \mathcal{C} \qquad C_1(\mathbf{f}) = C_2(g(\mathbf{f})). \tag{4.6}$$

That is, for any two features in the viewpace equivalent up to some transformation in $\mathcal{G}$, there must exist camera models in $\mathcal{C}$ which image those two features in exactly the same way. The camera-model sets $\mathcal{C}_{\text{inj}}, \mathcal{C}_{\text{proj}}, \mathcal{C}_{\text{aff}}$, and $\mathcal{C}_{\text{persp}}$ are all reconstructively complete in this sense, a result shown in [41].

This definition completes the prerequisites for a formal proof of the reconstructability/task invariance claim informally introduced at the start of this chapter.

4.4 Reconstructibility $\Longrightarrow$ Decidability

Two statements lie at the heart of the connection between reconstructability and a characterization of decidable tasks:

Statement A A set of camera models $\mathcal{C}$ provides scene reconstruction up to a group of transformations $\mathcal{G}$ and is $\mathcal{G}$-complete.

Statement B Decidable tasks on $\mathcal{C}$ are exactly those tasks invariant to transformations in $\mathcal{G}$.

This section demonstrates that statement A implies statement B and thereby characterizes decidable tasks for four of the five sets of camera models in the hierarchy of (3.3).

Theorem 1 *Statement A, above, implies statement B.*

Proof of Theorem 1: Suppose statement A. We must show two things: that all decidable tasks on $\mathcal{C}$ are invariant to transformations in $\mathcal{G}$ and that all tasks invariant to transformations in $\mathcal{G}$ are decidable on $\mathcal{C}$.

For the first, suppose that $T(\mathbf{f}) = 0$ is decidable on $\mathcal{C}$. Choose g arbitrarily from $\mathcal{G}$ and $\mathbf{f} \in \mathcal{V}^n$ so that $g(\mathbf{f}) \in \mathcal{V}^n$. By the $\mathcal{G}$-completeness of $\mathcal{C}$, there exist $C_1, C_2 \in \mathcal{C}$ such that $C_1(\mathbf{f}) = C_2(g(\mathbf{f}))$. By Equation (3.6), $T(\mathbf{f}) = T(g(\mathbf{f}))$. Thus, T is invariant to transformations in $\mathcal{G}$.

Arguing in the opposite direction, suppose that T is invariant to transformations in $\mathcal{G}$. Let $C_1, C_2 \in \mathcal{C}$ and $\mathbf{f}^1, \mathbf{f}^2 \in \mathcal{V}^n$ such that $C_1(\mathbf{f}^1) = C_2(\mathbf{f}^2)$. Because $\mathcal{C}$ provides reconstruction up to $\mathcal{G}$, there exists some reconstruction function $r_{\mathcal{C}}$ and elements $g_1, g_2 \in \mathcal{G}$ with

$$\mathbf{f}^1 \; = \; g_1(r_{\mathcal{C}}(C_1(\mathbf{f}^1)) \quad \text{and} \quad \mathbf{f}^2 \; = \; g_2(r_{\mathcal{C}}(C_2(\mathbf{f}^2)).$$

Letting $\mathbf{y} = C_1(\mathbf{f}^1) = C_2(\mathbf{f}^2)$, it follows that $\mathbf{f}^1 = g_1(r_{\mathcal{C}}(\mathbf{y}))$ and $\mathbf{f}^2 = g_2(r_{\mathcal{C}}(\mathbf{y}))$, so that $\mathbf{f}^1 = g_1(g_2^{-1}(\mathbf{f}^2))$ with $g_1 \circ g_2^{-1} \in \mathcal{G}$. Because of the invariance of T to transformations in $\mathcal{G}$, $T(\mathbf{f}^1) = T(\mathbf{f}^2)$. By Proposition 1 T is decidable on $\mathcal{C}$. $\blacksquare$

It remains an open issue whether the implication in the opposite direction holds, i.e., whether statement B entails statement A. Fortunately, the implication demonstrated in Theorem 1 suffices to completely analyze the nature of decidable tasks for the camera sets of interest in this work. The next section begins this analysis through direct application of the above theorem.

4.5 Invariance-based Task Decidability

Theorem 1, along with the reconstructability results of Chapter 2 and Section 4.2, provides an immediate characterization of four sets of decidable tasks in the hierarchy of (3.7). The following table summarizes these sets.

$\mathcal{T}_{\text{inj}}$	A task $T(f) = 0$ is decidable on $\mathcal{C}_{\text{inj}}$ if T is invariant to the group of bijective transformations of $\mathbb{R}^3$, $\mathcal{G}_{\text{bij}}$.
$\mathcal{T}_{\text{proj}}$	A task $T(f) = 0$ is decidable on $\mathcal{C}_{\text{proj}}$ if T is invariant to elements of $\mathcal{G}_{\text{proj}} = \text{GL}(4)$, the group of invertible linear transformations of $\mathbb{P}^3$.
$\mathcal{T}_{\text{aff}}$	A task $T(f) = 0$ is decidable on $\mathcal{C}_{\text{aff}}$ if T is invariant to $\mathcal{G}_{\text{aff}}$, the group of affine transformations of $\mathbb{R}^3$.
$\mathcal{T}_{\text{persp}}$	A task $T(f) = 0$ is decidable on $\mathcal{C}_{\text{persp}}$ if T is invariant to the group $\mathcal{G}_{\text{sim}}$ of similarities (scaled Euclidean transformations) of $\mathbb{R}^3$.

Figure 4.2: Characterizations of four sets of decidable tasks.

Figure 4.2 provides a much stronger version of the hierarchy (3.7), quantifying how more tasks become decidable as knowledge about the observing cameras increases. Note that $\mathcal{T}_{\text{uncal}}$ does not appear. We do not have a simple characterization of tasks in $\mathcal{T}_{\text{uncal}}$, but Figure 4.2 does demonstrate upper and lower "bounds" on the tasks in that set. Chapter 8 considers task performance with the uncalibrated projective cameras of $\mathcal{C}_{\text{uncal}}$ in more detail.

Figure 4.2 answers the question alluded to in Chapter 1 and refined subsequently: what assumptions about the observing stereo rigs do the tasks of Figures 1.2 and 1.3 require to guarantee performance? Equivalently, where do those example tasks fall in the hierarchy of (3.7)? From the invariance properties discussed in Section 4.1, the above characterizations offer these answers:

- **Midpoint Task**: The midpoint task is decidable on affinely calibrated stereo rigs (as well as perspective cameras). In fact, any Euclidean length-ratio task constraint has a verifying encoding, given affine (or similarity-based) calibration.

- **Collinearity Task**: The collinearity task is decidable on weakly calibrated stereo rigs. Any projectively invariant task has a verifying encoding, provided the fundamental matrix of the observing stereo rig is known.

- **Point-to-point Task**: The point-to-point task is decidable on injective cameras. If a set of camera models is known to be injective, there exists a verifying encoding for any point-to-point task.

Although these results can guide a hand/eye system designer to provide the level of calibration needed to support certain tasks of interest, they leave critical questions unanswered for the user or programmer of a hand/eye system. First, what are these decidable tasks? Figure 4.2's characterizations provide a test to determine whether or not a task is decidable in each of four cases, but it does not provide a constructive method for generating any such task. The following chapter resolves this with an explicit enumeration of decidable tasks.

Second and equally important, what are verifying task encodings for the decidable tasks? As stated, verifying encodings must exist for the above tasks (as long as the appropriate calibration information is available), but nothing in Theorem 1 or Figure 4.2 suggests how to build such encodings. On a task-by-task basis, a programmer might construct encodings by hand, but problems can arise.

Consider, for instance, the encodings constructed in Section 3.3 for the T_{col} and T_{midpt} tasks. E_{col}, in fact, does *not* verify T_{col}, even on $\mathcal{C}_{\mathrm{proj}}$. The confounding configuration in Figure 1.3 continues to satisfy E_{col}, but not T_{col} when the observing cameras' fundamental matrix is known. Similarly, E_{midpt} does not verify T_{mipdt} even on the sets $\mathcal{C}_{\mathrm{aff}}$ and $\mathcal{C}_{\mathrm{persp}}$ on which T_{midpt} is decidable. For these two tasks, verifying encodings exist, but certainly not every encoding is a verifying one. Rather then requiring manually created encodings, a surer and more flexible system would provide an automatic means for generating verifying encodings for desired task-space constraints, based on only a few primitive encodings. Chapter 6 demonstrates languages which do so.

Chapter 5

Enumerating Decidable Tasks

Overview Chapters 5 and 6 present the bulk of the technical results of this thesis. With propositions 2 - 5 this chapter constructively characterizes the decidable tasks in $\mathcal{T}_{\text{inj}}, \mathcal{T}_{\text{proj}}, \mathcal{T}_{\text{aff}}$, and $\mathcal{T}_{\text{persp}}$. Sections 5.1 to 5.2 build necessary background for these characterizations. Sections 5.3 to 5.6 motivate and prove each proposition in turn. To prevent details from overwhelming the larger theme — the structure of decidable tasks — no subsequent developments will rely on the proofs appearing at the end of these sections. The example tasks of Section 5.7 recapitulate the chapter's results.

Although Figure 4.2 provides tests of the decidability for a given task, those tests do not help construct examples of decidable tasks. By identifying tasks with their satisfying feature configurations, this chapter derives algebraic rules for creating all decidable tasks in $\mathcal{T}_{\text{i}}$ for $\text{i} \in \text{inj}, \text{proj}, \text{aff}, \text{persp}$. Representing feature configurations in an efficient and unambiguous way is a critical step toward this constructive description of decidability.

5.1 Feature Equivalence Relations

Pairs of features related by a transformation in $\mathcal{G}$, $\mathbf{f}$ and $g(\mathbf{f})$, played an important role in the proof of Theorem 1. When $\mathcal{G}$ is a group of transformations, the relationship identifying $\mathbf{f}$ and $g(\mathbf{f})$ for $g \in \mathcal{G}$ defines an equivalence relation. Thus, the following four equivalence relations emerge from the four sets of camera models treated in Figure 4.2:

Definition 1 *For each* $\mathrm{i} \in \{\mathrm{bij}, \mathrm{proj}, \mathrm{aff}, \mathrm{persp}\}$ *define an equivalence relation on* $\mathcal{V}^n$, *called* $\simeq_{\mathrm{i}}$, *by*

$$\forall \mathbf{f}^1, \mathbf{f}^2 \in \mathcal{X}^n \quad \mathbf{f}^1 \simeq_{\mathrm{i}} \mathbf{f}^2 \iff \exists\, g \in \mathcal{G}_{\mathrm{i}} \; s.t.\; \mathbf{f}^2 = g(\mathbf{f}^1)\ . \tag{5.1}$$

Equation (5.1) defines conditions for bijective-, projective-, affine-, and similarity-equivalence for lists of feature points. Because of the completeness of the camera models considered, if two features are equivalent under $\simeq_{\mathrm{i}}$ and both are in the views-pace $\mathcal{V}^n$, they yield identical images under two distinct stereo rigs in $\mathcal{C}_{\mathrm{i}}$.

The equivalence relations $\simeq_{\mathrm{i}}$ impose an equivalence-class structure on the space of possible features $\mathcal{X}^n$. Because these equivalence classes underlie the structure of decidability, a concise representation for them will simplify the analysis of decidable tasks. As every feature in $\mathcal{X}^n$ lies in only one equivalence class, one such feature — a *canonical representative* — serves to denote an entire class. Recall that features consist of an ordered list of feature points with coordinates written in some frame of reference. When considering the affine or similarity-based structure of points, $\mathcal{X} = \mathbb{R}^3$ serves as the most natural representation of coordinates. With projective structure and homogeneous coordinates $\mathcal{X} = \mathbb{P}^3$. Either space serves equally well for describing a feature's bijective structure; here $\mathbb{R}^3$ will be used.

Canonical Features

Which element of each equivalence class is the canonical representative? Informally, call the canonical representative that feature whose feature points' coordinates consist of as many zeros as possible. This applies for each feature point individually, from left to right and bottom to top. To break remaining ties, smaller integral coordinates are preferred when possible. Each resulting feature list exposes the structure (up to the appropriate equivalence relation) of all the features in its equivalence class. More formal descriptions of canonical representatives will replace this informal definition in Sections 5.3 - 5.6. For the moment, this treatment suffices to motivate a few examples of canonical features.

- **Bijective Equivalence** Consider the following five features —

$$\underset{\mathbf{f}^1}{\left\{\begin{smallmatrix}0&1&1\\0&0&0\\0&0&0\end{smallmatrix}\right\}} \quad \underset{\mathbf{f}^2}{\left\{\begin{smallmatrix}1&4&4\\2&5&5\\3&6&6\end{smallmatrix}\right\}} \quad \underset{\mathbf{f}^3}{\left\{\begin{smallmatrix}1&0&0\\0&0&0\\0&0&0\end{smallmatrix}\right\}} \quad \underset{\mathbf{f}^4}{\left\{\begin{smallmatrix}0&0&1\\0&0&0\\0&0&0\end{smallmatrix}\right\}} \quad \underset{\mathbf{f}^5}{\left\{\begin{smallmatrix}0&1&0\\0&0&0\\0&0&0\end{smallmatrix}\right\}}$$

  Features $\mathbf{f}^1$ and $\mathbf{f}^2$ are bijectively equivalent, because the second and third point in each case are identical to one another and different from the first feature point in the lists. $\mathbf{f}^1$ is the canonical representative for all three-point features whose first two feature points differ and whose final two feature points coincide. Though $\mathbf{f}^3$ is bijectively equivalent to $\mathbf{f}^1$ and contains more zeros than $\mathbf{f}^1$, $\mathbf{f}^3$ is not in canonical form because the "zero-preference rule" is applied not to the feature list as a whole, but to the feature points individually from left to right. Canonical representatives of $\simeq_{\mathrm{bij}}$ always have zeros in their feature points' second and third components.

  Similar to $\mathbf{f}^1$, features $\mathbf{f}^4$ and $\mathbf{f}^5$ contain two identical and an additional distinct feature point, but because all feature lists are considered ordered, neither

$\mathbf{f}^4$ nor $\mathbf{f}^5$ is bijectively equivalent to $\mathbf{f}^1$. In fact, both $\mathbf{f}^4$ and $\mathbf{f}^5$ are themselves canonical representatives. Chapter 6 addresses permutation-equivalence, a relation which does consider $\mathbf{f}^1$, $\mathbf{f}^4$, and $\mathbf{f}^5$ equivalent.

- **Projective Equivalence** Consider another set of five features

$$\underset{\mathbf{f}^1}{\left\{\begin{smallmatrix} 0 & 1 & 3 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{smallmatrix}\right\}} \qquad \underset{\mathbf{f}^2}{\left\{\begin{smallmatrix} 5 & 2 & 8 \\ 0 & 0 & 0 \\ 3 & 6 & 0 \\ 1 & 1 & 1 \end{smallmatrix}\right\}} \qquad \underset{\mathbf{f}^3}{\left\{\begin{smallmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{smallmatrix}\right\}} \qquad \underset{\mathbf{f}^4}{\left\{\begin{smallmatrix} 0 & 2 & 0 \\ 0 & 0 & 2 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{smallmatrix}\right\}} \qquad \underset{\mathbf{f}^5}{\left\{\begin{smallmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 1 & 0 \\ 2 & 0 & 1 \end{smallmatrix}\right\}} \tag{5.2}$$

Assuming the usual embedding of $\mathbb{R}^3$ within $\mathbb{P}^3$, the points in $\mathbf{f}^1$ are three points along a single axis in $\mathbb{R}^3$, and those in $\mathbf{f}^2$ satisfy the midpoint task T_{midpt}. Both are projectively equivalent to the canonical representative $\mathbf{f}^3$ which expresses that its three feature points lie in a single line with no two coincident. Because T_{midpt} is satisfied for feature $\mathbf{f}^2$ but not $\mathbf{f}^1$ and both features are projectively equivalent to $\mathbf{f}^3$, this suggests (and Proposition 3 will prove) that T_{midpt} is *not* decidable on a weakly calibrated set of projective cameras. Note that the zeros are placed in the lower components before upper ones, so that $\mathbf{f}^3$ is in canonical form, while the projectively equivalent $\left\{\begin{smallmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{smallmatrix}\right\}$ is not.

The adjacent vertices of a square contained in $\mathbf{f}^4$, as well as the points in $\mathbf{f}^5$, are projectively equivalent to the canonical list $\left\{\begin{smallmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{smallmatrix}\right\}$, three points which span a plane. These five examples will appear again in Section 5.4 to demonstrate additional properties of projective canonical features.

- **Affine Equivalence** Consider the following features in an affine task space

$$\underset{\mathbf{f}^1}{\left\{\begin{smallmatrix} 0 & 1 & 3 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{smallmatrix}\right\}} \qquad \underset{\mathbf{f}^2}{\left\{\begin{smallmatrix} 0 & 1 & 2 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{smallmatrix}\right\}} \qquad \underset{\mathbf{f}^3}{\left\{\begin{smallmatrix} 0 & 1 & \pi \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{smallmatrix}\right\}} \qquad \underset{\mathbf{f}^4}{\left\{\begin{smallmatrix} 0 & \pi & \pi^2 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{smallmatrix}\right\}} \qquad \underset{\mathbf{f}^5}{\left\{\begin{smallmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{smallmatrix}\right\}}$$

Though $\mathbf{f}^1 \simeq_{\mathrm{bij}} \mathbf{f}^2$ and under $\simeq_{\mathrm{bij}}$-equivalence $\mathbf{f}^2$ is the canonical representative of $\mathbf{f}^1$'s equivalence class, when considering only affine transformations $\mathbf{f}^1 \not\simeq_{\mathrm{aff}} \mathbf{f}^2$. (Section 5.5 will prove this rigorously.) Similarly, $\mathbf{f}^3 \not\simeq_{\mathrm{aff}} \mathbf{f}^2$, even though all of $\mathbf{f}^1$, $\mathbf{f}^2$, and $\mathbf{f}^3$ are projectively equivalent (the first feature of (5.2) shows the homogeneous representation of $\mathbf{f}^1$). In terms of their affine structure, all three features are in canonical form. On the other hand, $\mathbf{f}^4$ *is* affinely equivalent to $\mathbf{f}^3$. Because each equivalence class has a single canonical representative and because of the preference for small integers over larger integers (or nonintegers), $\mathbf{f}^4$ is not in canonical form.

The inclusion of $\mathbf{f}^5$ serves to underscore the fact that two feature lists must have the same cardinality to be equivalent under any of the relations of Definition 1. Among features comprised of four points, $\mathbf{f}^5$ is a canonical representative.

- **Similarity Equivalence** Under similarity equivalence, the first three features of the following set are in canonical form.

$$\underset{\mathbf{f}^1}{\left\{\begin{matrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{matrix}\right\}} \qquad \underset{\mathbf{f}^2}{\left\{\begin{matrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 16 & 0 \\ 0 & 0 & 0 & 16 \end{matrix}\right\}} \qquad \underset{\mathbf{f}^3}{\left\{\begin{matrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0.1 & 0 \\ 0 & 0 & 0 & 36 \end{matrix}\right\}} \qquad \underset{\mathbf{f}^4}{\left\{\begin{matrix} 0 & 2 & 0 & 0 \\ 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 2 \end{matrix}\right\}}$$

Feature 1 represents points at adjacent vertices of a cube, $\mathbf{f}^2$ the corners of a "pizza carton" parallelipiped, and $\mathbf{f}^3$ the vertices of a yardstick-shaped solid. Feature 4 is similarity-equivalent to $\mathbf{f}^1$, but not $\mathbf{f}^2$ nor $\mathbf{f}^3$. All four are affinely equivalent, however, demonstrating a primary difference between affine and similarity-based representations: the former can confound nonuniform stretching of coordinates while the latter do not.

5.2 Elementary Tasks

Theorem 1 bridges the gap between equivalence classes and tasks. That theorem asserts that decidable tasks on $\mathcal{C}_{\mathrm{i}}$ are those invariant to the action of transformations in $\mathcal{G}_{\mathrm{i}}$ ($\mathcal{C}_{\mathrm{i}}$ and $\mathcal{G}_{\mathrm{i}}$ appear in Figure 4.2). Thus, decidable tasks are exactly those which are constant-valued on each $\simeq_{\mathrm{i}}$-equivalence class of features. Also, since $\mathcal{V}$ contains an open set, each equivalence class of $\mathcal{X}^n$ contains at least one feature in $\mathcal{V}^n$ [41]. Thus, any decidable task function with domain $\mathcal{V}^n$ uniquely defines an extended function on $\mathcal{X}^n$ which is constant-valued on $\simeq_{\mathrm{i}}$-equivalence classes.

Since decidable tasks are constant on each $\simeq_{\mathrm{i}}$-equivalence class, a natural basis for describing $\mathcal{T}_{\mathrm{i}}$ is the set of tasks satisfied by *exactly one* such equivalence class: call these *elementary tasks*. Each elementary task is decidable and satisfied by exactly one canonical representative in $\mathcal{X}^n$. Henceforth that canonical representative will denote not only its equivalence class of features, but also the elementary task those features satisfy. Any decidable task consists of a disjunction of the constraints of some number of elementary tasks. Thus, any set of canonical representatives (all of length n) gives rise to some decidable task and vice versa.

As examples, consider again the three tasks from Chapter 1. The point-to-point task, decidable on $\mathcal{C}_{\mathrm{inj}}$, is itself an elementary task:

$$T_{\mathrm{pp}} = \left\{ \begin{smallmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{smallmatrix} \right\} .$$

As the following section will prove, the only other two-point task decidable on $\mathcal{C}_{\mathrm{inj}}$ has canonical form $\left\{ \begin{smallmatrix} 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{smallmatrix} \right\}$, representing point-noncoincidence. Note that this example uses the name of a task function to represent a task; we will continue doing this as long as no confusion can result.

T_{midpt} is also an elementary task, decidable on $\mathcal{C}_{\text{aff}}$ rather than $\mathcal{C}_{\text{inj}}$, with

$$T_{\text{midpt}} = \left\{ \begin{smallmatrix} 0 & 1 & 2 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{smallmatrix} \right\} .$$

In fact, two more canonical representatives in $\mathcal{T}_{\text{aff}}$ equally well represent the midpoint-positioning task,

$$T^{*}_{\text{midpt}} = \left\{ \begin{smallmatrix} 0 & 1 & 0.5 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{smallmatrix} \right\}$$

$$T^{**}_{\text{midpt}} = \left\{ \begin{smallmatrix} 0 & 1 & -1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{smallmatrix} \right\} .$$

The distinction depends on which feature point is equidistant from the other two, i.e., which of the three is the midpoint: the second in T_{midpt}, the third in T^{*}_{midpt}, and the first in T^{**}_{midpt}. Chapter 6 builds languages which handle such feature-point-naming issues gracefully.

As pointed out in Section 3.1, the collinearity task of Figure 1.2 allows the coincidence of any or all of the feature points involved. Hence several projectively equivalent classes of features satisfy T_{col} (which is thus *not* elementary):

$$T_{\text{col}} = \left\{ \; \left\{ \begin{smallmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{smallmatrix} \right\} \; \left\{ \begin{smallmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{smallmatrix} \right\} \; \left\{ \begin{smallmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{smallmatrix} \right\} \; \left\{ \begin{smallmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{smallmatrix} \right\} \; \left\{ \begin{smallmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{smallmatrix} \right\} \; \right\} .$$

Canonical representatives allow the articulation of any elementary task in the same manner as T_{midpt} or T_{pp}. Enumerating all elementary tasks, then, completely accounts for the "raw material" for specifying more involved 3d constraints such as T_{col}. The next four sections provide this enumeration of elementary tasks in $\mathcal{T}_{\text{inj}}$, $\mathcal{T}_{\text{proj}}$, $\mathcal{T}_{\text{aff}}$, and $\mathcal{T}_{\text{persp}}$. In each case a set of rules governs the feature points which a canonical representative of size n may append to become a representative of size $n+1$. These algebraic rules capture all of the distinct methods of building $\mathcal{G}_{\mathrm{i}}$-invariant feature configurations one point at a time.

5.3 Elementary Tasks of $\mathcal{T}_{\mathrm{inj}}$

This section demonstrates that point-coincidence and point-noncoincidence encompass everything that camera models in $\mathcal{C}_{\mathrm{bij}}$ can distinguish. The rules for building the elementary tasks in $\mathcal{T}_{\mathrm{inj}}$ boil down to adding a new feature point or repeating one already seen. Before presenting those rules, a number of conventions which will extend through this and the following three sections need to be established.

If $\mathbf{f}^- = \{f_1 \ldots f_{n-1}\}$ is a feature in canonical form, f_n is considered a *legal successor* of $\mathbf{f}^-$ if the concatenated list $\mathbf{f} = \{f_1 \ldots f_n\} = \{f^-; f_n\}$ is also in canonical form. The rules below, in essence, construct the possible legal successors of every canonical representative. The designation $f \in \mathbf{f}$ will abbreviate the fact that the feature point f appears somewhere in the list $\mathbf{f}$.

For each of the four sets of decidable tasks considered, determining legal successors depends on properties of the canonical representatives. For $\mathcal{T}_{\mathrm{inj}}$ an important property of canonical representatives and, in fact, all features is the number of distinct feature points they contain, designated by D. Formally, $D(\mathbf{f})$ is the cardinality of a feature $\mathbf{f}$ when considered a set, rather than an ordered list; the dependence on $\mathbf{f}$ is dropped if clear or unimportant. For example, for the features

$$\left\{ \begin{smallmatrix} 0 & 1 & 1 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{smallmatrix} \right\} \quad \text{and} \quad \left\{ \begin{smallmatrix} -1 & 7 & -1 & 7 & -1 \\ 2 & 1 & 2 & 1 & 2 \\ 3 & -3 & 3 & -3 & 3 \end{smallmatrix} \right\},$$

D equals 3 and 2 respectively. With this background the following proposition enumerates the elementary tasks of $\mathcal{T}_{\text{inj}}$. In addition, Proposition 2 provides a formal definition of all features in canonical form, replacing the informal "zero-preference rule." $\mathcal{L}_{\text{bij}}$ will denote the set of all such canonical feature lists. The construction proceeds inductively, starting from the single possible canonical form which contains one feature point and imposing conditions on what feature points can be appended to a canonical list.

Proposition 2 *Let $\mathcal{X} = \mathbb{R}^3$. Suppose $\mathbf{f}^- = \{f_1 \dots f_{n-1}\} \in \mathcal{X}^{n-1}$ is in canonical form with respect to bijective equivalence. Let D equal the number of distinct points in $\mathbf{f}^-$. Also, suppose $\mathbf{f} = \{f_1 \dots f_n\} \in \mathcal{X}^n$ is in canonical form if and only if f_n is a legal successor of $\mathbf{f}^-$, i.e., if and only if f_n satisfies one of the following rules:*

Rule 0 *The only feature list of length $n = 1$ in canonical form is $\mathbf{f} = \left\{ \begin{smallmatrix} 0 \\ 0 \\ 0 \end{smallmatrix} \right\}$.*

Rule 1 $f_n = \left[\begin{smallmatrix} D \\ 0 \\ 0 \end{smallmatrix} \right]$.

Rule 2 $f_n = f_i$, *for some* $f_i \in \mathbf{f}^-$ $(1 \leq i \leq n-1)$.

Then, for each $\simeq_{\text{bij}}$-equivalence class of $\mathcal{X}^n$, there is a unique element of that class formed by successive application of these rules.

Proof of Proposition 2: As suggested, let $\mathcal{L}_{\text{bij}}$ represent the set of all features obtained by succesive application of these rules, i.e., $\mathcal{L}_{\text{bij}}$ is the set of canonical representatives. By Rule 0, $\mathcal{L}_{\text{bij}}$ is nonempty. We need to demonstrate two things:

1. **Statement 1** If $\mathbf{f}, \mathbf{g}$ are any two distinct elements of $\mathcal{L}_{\text{bij}}$, then $\mathbf{f}$ and $\mathbf{g}$ are *not* bijectively equivalent, i.e. $\mathbf{f} \not\simeq_{\text{bij}} \mathbf{g}$.

2. **Statement 2** If $\mathbf{f}$ is any feature in $\mathcal{X}^n$, then there exists a $\mathbf{g} \in \mathcal{L}_{\text{bij}}$ such that $\mathbf{f} \simeq_{\text{bij}} \mathbf{g}$.

Before proceeding to prove statements 1 and 2, we show a useful lemma about the structure of canonical representatives.

Lemma 1 *Suppose* $\mathbf{f} \in \mathcal{L}_{\text{bij}}$. *If* $\mathbf{f}$ *has* n *feature points and* D *distinct feature points, each* $f_i \in \mathbf{f}$ *has coordinates* $f_i = \left[\begin{smallmatrix} d_i \\ 0 \\ 0 \end{smallmatrix}\right]$ *for some integral* d_i *such that* $0 \leq d_i < D$.

Proof of Lemma 1: Rule 0 entails that Lemma 1 holds for $n = 1$. Assume the lemma holds for all positive integers less than n. Let $\mathbf{f}^-$ represent the initial sublist of $n - 1$ feature points of $\mathbf{f}$. In particular, the lemma holds for $\mathbf{f}^-$. Two cases apply. First, if $D(\mathbf{f}) = D(\mathbf{f}^-)$, then f_n was added through the application of Rule 2. Thus, $f_n = f_i$ with $0 < i < n$. By the assumption of Lemma 1 for $\mathbf{f}^-$, Lemma 1 must also hold for $\mathbf{f}$. On the other hand, if $D(\mathbf{f}) \neq D(\mathbf{f}^-)$, $\mathbf{f}$ contains more distinct points than $\mathbf{f}^-$. Since $\mathbf{f}$ and $\mathbf{f}^-$, considered sets, differ only by f_n, it must follow that $f_n \notin \mathbf{f}^-$ and that $D(\mathbf{f}) = D(\mathbf{f}^-) + 1$. Thus, f_n was added through application of Rule 1, so that $f_n = \left[\begin{smallmatrix} D(\mathbf{f}^-) \\ 0 \\ 0 \end{smallmatrix}\right]$. Since $D(\mathbf{f}) = D(\mathbf{f}^-) + 1$, the lemma still holds for $\mathbf{f}$. ■

Next, statements 1 and 2 are demonstrated in turn.

Statement 1 Suppose $\mathbf{f}$ and $\mathbf{g}$ are distinct elements of $\mathcal{L}_{\text{bij}}$. If they contain different numbers of feature points, they are not bijectively equivalent, so suppose they both contain n feature points. If $n = 1$, $\mathbf{f} = \mathbf{g}$ by Rule 0. We thus proceed by induction. Suppose $n > 1$ and that the proposition holds for all positive integers less than n. In particular, it holds for $\mathbf{f}^-$ and $\mathbf{g}^-$, the initial sublists of $n - 1$ features of $\mathbf{f}$ and $\mathbf{g}$. Thus, if $\mathbf{f}^-$ and $\mathbf{g}^-$ are distinct, they are not bijectively equivalent. This, in turn, implies that $\mathbf{f}$ and $\mathbf{g}$ are not bijectively equivalent.

It remains to consider the case in which $\mathbf{f}^- = \mathbf{g}^-$. From this we know that $D(\mathbf{f}^-) = D(\mathbf{g}^-) = D$. Hence f_n and g_n could not both have come from the application of Rule 1, or $\mathbf{f}$ would equal $\mathbf{g}$. Suppose that f_n came from Rule 1 and

$g_n = g_i \quad (0 < i < n)$ from Rule 2. Suppose, too, a bijection b exists such that $\mathbf{f} = b(\mathbf{g})$, i.e., $\forall i \;\; f_i = b(g_i)$. Because b is a well-defined function, $b(g_i) = b(g_n)$ implies that $f_i = f_n$, which can not be true by Lemma 1. The same contradiction arises if $f_n = f_j$ and $g_n = g_i$ both come through application of Rule 2, but $f_j = g_j \neq g_i$. This exhausts the possibilities for constructing $\mathbf{f}$ and $\mathbf{g}$, so statement 1 holds.

Statement 2 Again, this holds for $n = 1$; assume $n > 1$ and that the statement holds for all positive integers less than n. Let $\mathbf{f}$ be any feature in $\mathcal{X}^n$ and $\mathbf{f}^-$ its initial sublist of $n-1$ feature points. By inductive hypothesis there exists some feature $\mathbf{g}^- \in \mathcal{L}_{\text{bij}}$ with $n-1$ feature points and a bijection $b \in \mathcal{G}_{\text{bij}}$ such that $b(\mathbf{f}^-) = \mathbf{g}^-$. Consider the case in which f_n is distinct from all of the other $f_i \in \mathbf{f}^-$. Lemma 1 shows that $g_n = \begin{bmatrix} D(\mathbf{g}^-) \\ 0 \\ 0 \end{bmatrix}$ is distinct from all $g_i \in \mathbf{g}^-$. Let $\mathbf{g} = \{\mathbf{g}^-; g_n\}$. By Rule 1 $\mathbf{g}$ is in canonical form. Let b^* be the bijection defined by

$$b^*(h) = \begin{cases} g_n & \text{if } \; h = f_n \\ b(f_n) & \text{if } \; h = b^{-1}(g_n) \\ b(h) & \text{otherwise} \end{cases} .$$

This relation, b^*, is both a function and a bijection because b is a bijection and because b^* merely switches b's action on the domain elements f_n and $b^{-1}(g_n)$. (If they happen to be the same, $b^* = b$.)

Consider the other possibility, that f_n equals some f_i with $0 < i < n$. Then, letting $g_n = g_i$ (by Rule 2, $\mathbf{g} \in \mathcal{L}_{\text{bij}}$) and $b^* = b$ again demonstrates the statement. Hence, statement 2 and Proposition 2 hold. ■

5.4 Elementary Tasks of $\mathcal{T}_{\mathrm{proj}}$

To present rules for constructing the elementary tasks of $\mathcal{T}_{\mathrm{proj}}$, several additional properties of features must be defined. As examples, consider again the five feature lists which appeared in (5.2)

$$\underset{\mathbf{f}^1}{\left\{\begin{smallmatrix} 0 & 1 & 3 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{smallmatrix}\right\}} \qquad \underset{\mathbf{f}^2}{\left\{\begin{smallmatrix} 5 & 2 & 8 \\ 0 & 0 & 0 \\ 3 & 6 & 0 \\ 1 & 1 & 1 \end{smallmatrix}\right\}} \qquad \underset{\mathbf{f}^3}{\left\{\begin{smallmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{smallmatrix}\right\}} \qquad \underset{\mathbf{f}^4}{\left\{\begin{smallmatrix} 0 & 2 & 0 \\ 0 & 0 & 2 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{smallmatrix}\right\}} \qquad \underset{\mathbf{f}^5}{\left\{\begin{smallmatrix} 1 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 1 & 0 \\ 2 & 0 & 1 \end{smallmatrix}\right\}} \tag{5.3}$$

One fundamental attribute of a feature $\mathbf{f} \in \mathcal{X}^n$ ($\mathcal{X} = \mathbb{P}^3$) is $\dim(\mathbf{f})$, defined to be the number of dimensions spanned by $\mathbf{f}$'s consituent feature points when considered vectors in $\mathbb{R}^4$. Thus, $1 \leq \dim(\mathbf{f}) \leq 4$, as long as the feature list is nonempty. This value is well defined even though we are using homogeneous coordinates, because scaling (by a nonzero factor) any vector in a set does not change the span of that set. As an example, the dimension of features $\mathbf{f}^1, \mathbf{f}^2$, and $\mathbf{f}^3$ is 2. Because projective transformations are invertible, they preserve dimensionality, so that $\dim(\mathbf{f})$ is a constant for all features in a given equivalence class. From this, we can conclude that since $\dim(\mathbf{f}^4) = 3$ and $\dim(\mathbf{f}^3) = 2$, $\mathbf{f}^4$ and $\mathbf{f}^3$ are members of distinct equivalence classes of $\mathcal{X}^n$.

A "row" of $\mathbf{f}$ merely refers to the list of ith coordinates of $\mathbf{f}$, e.g., the first row of $\mathbf{f}^2$ is $\{\, 5 \;\, 2 \;\, 8 \,\}$. One property of rows of $\mathbf{f}$ which will turn out to be important is whether or not two rows have been "measured" with respect to one another. To define this property, we first define the relation $\sim_{\mathrm{m}}$ on the set $\{1, \ldots, 4\}$, the row indices of $\mathbf{f}$. We specify that $i \sim_{\mathrm{m}} j$ in $\mathbf{f}$ if and only if there is a feature point in $\mathbf{f}$ which contains a component of 1 both in row i and in row j ($1 \leq i, j \leq 4$). An example from (5.2) is that for $\mathbf{f}^1$, $1 \sim_{\mathrm{m}} 4$ and $4 \sim_{\mathrm{m}} 4$. Though the $\sim_{\mathrm{m}}$ relation depends on a particular feature $\mathbf{f}$, we will suppress writing that feature when clear from context.

Let $\simeq_{\mathrm{m}}$ be the transitive closure of $\sim_{\mathrm{m}}$ within a particular feature $\mathbf{f}$. Then we will say that two rows i and j are *measured in* $\mathbf{f}$ if and only if $i \simeq_{\mathrm{m}} j$. For example, $2 \simeq_{\mathrm{m}} 4$ in $\mathbf{f}^5$ above, although $2 \not\sim_{\mathrm{m}} 4$. The relation $\simeq_{\mathrm{m}}$ is an equivalence relation on the set $\{1, \ldots, 4\}$. Thus, it induces a partition on that set. We will call that partition the *row-partition of* $\mathbf{f}$ and denote it generally by $\{\mathcal{P}_1, \ldots, \mathcal{P}_N\}$, where the $\mathcal{P}_i$ are mutually disjoint and each $\mathcal{P}_i \subset \{1, \ldots, 4\}$. The row-partitions of $\mathbf{f}^4$ and $\mathbf{f}^5$ are thus $\{\,\{1,2\}\ \{3\}\ \{4\}\,\}$ and $\{\,\{1,2,4\}\ \{3\}\,\}$, respectively.

To remove arbitrary scale factors when necessary, define $\mathrm{norm}(f)$ to be the homogeneous coordinates of the feature point f, scaled so that the nonzero coordinate with the highest row index is a 1. Thus, $\mathrm{norm}\left(\begin{bmatrix} -1 \\ 14 \\ 2 \\ 0 \end{bmatrix}\right) = \begin{bmatrix} -0.5 \\ 7 \\ 1 \\ 0 \end{bmatrix}$.

The purpose of the following proposition is to demonstrate explicitly how to construct canonical representatives of the $\simeq_{\mathrm{proj}}$-equivalence classes of $\mathcal{X}^n$. As in Proposition 2 the construction proceeds inductively.

Proposition 3 *Let* $\mathcal{X} = \mathbb{P}^3$*. Suppose* $\mathbf{f}^- = [f_1 \ldots f_{n-1}] \in \mathcal{X}^{n-1}$ *is in canonical form. Let* $d = \dim(\mathbf{f}^-)$ *and* $\{\mathcal{P}_1, \ldots, \mathcal{P}_N\}$ *be the row-partition of* $\mathbf{f}^-$ *under* $\simeq_{\mathrm{m}}$*. Also, suppose that* $\mathbf{f} = [f_1 \ldots f_n] \in \mathcal{X}^n$ *is in canonical form if and only if* f_n *is a legal successor of* $\mathbf{f}^-$*, i.e., if and only if* f_n *satisfies one of the following rules:*

Rule 0 *The only feature list of length 1 in canonical form is* $\mathbf{f} = \left\{ \begin{smallmatrix} 1 \\ 0 \\ 0 \\ 0 \end{smallmatrix} \right\}$.

Rule 1 $f_n = e_i$*, where* $i = \dim(\mathbf{f}^-) + 1$ *and* $i \leq 4$.

Rule 2 $f_n = \mathrm{norm}(v)$*,* $v \in \mathbb{R}^4, v \neq 0$*, and* v *contains nonzero entries only in rows in* $\mathcal{P}_i$ *for some* $i \in \{1, \ldots, N\}$.

Rule 3 $f_n = s_1 + \ldots + s_M$ $(M \geq 2)$*, where each* $s_k \in \mathbb{R}^4$ *is a legal successor of* $\mathbf{f}^-$ *by Rule 2 and each* s_k *contains nonzero components only in rows in some distinct* *partition set $\mathcal{P}_{i_k}$, i.e., $j \neq k \Rightarrow i_j \neq i_k$.*

Then, for each $\simeq_{\mathrm{proj}}$-equivalence class of $\mathcal{X}^n$, there is a unique representative element of that class formed by successive application of these rules.

Proposition 3 Before proving Proposition 3, we investigate some of the structure of canonical lists of feature points. One property, true by inspection of the rules, is that the all-zero vector can not appear in a canonical list:

Property 1 *Let f_n be a feature point generated by any of the rules in Proposition 3. Then f_n contains a nonzero component, i.e. $f_n \neq 0$.* ■

Through the course of this proof, we will blur the distinction between lists of feature points $\mathbf{f}$ and the matrix which consists of exactly the coordinates of the points in $\mathbf{f}$. Interpreting a feature list (in homogeneous coordinates) as a matrix has the advantage that $\simeq_{\mathrm{proj}}$-equivalence can be expressed elegantly as a transformation of coordinates. The following property follows from the fact that for two feature lists to be projectively equivalent, each pair of corresponding feature points in those lists must be related by the same projective transformation $\mathbf{A}$. The diagonal matrix $\mathrm{Diag}(\rho_i)$ allows for arbitrary (nonzero) scalings of the points' homogeneous coordinates.

Property 2 *If $\mathbf{f}$ and $\mathbf{g}$ are lists of n feature points represented as columns of homogeneous coordinates, then*

$$\mathbf{f} \simeq_{\mathrm{proj}} \mathbf{g} \iff \exists \mathbf{A} \in GL(4),\ \rho_i \in \mathbb{R}: \quad \mathbf{g} = \mathbf{A}\ \mathbf{f}\ \mathrm{Diag}(\rho_i),$$

where the $\rho_i \neq 0$. Lists with different numbers of feature points are not projectively equivalent ($\simeq_{\mathrm{proj}}$-equivalent). ■

If f_n is a vector generated by Rule 3, we can write $f_n = s_1 + \ldots + s_M$, and by Property 1 we know that each $s_k \neq 0$. In fact, we know even more about the relationship between the components of f_n and those of the s_k's. Since s_j and s_k can not have nonzero entries in identical rows for any $j \neq k$, each of f_n's nonzero components must have come from exactly one of the s_k's. In other words, we can state

Property 3 *Let $f_n = [\alpha_1\ \alpha_2\ \alpha_3\ \alpha_4]'$ be a feature point generated by Rule 3. If row k's component, α_k, is nonzero, then there is a unique s_i $(1 \leq i \leq M)$ such that s_i's component in row k is also α_k.* ■

We can add another statement about the nonzero components of vectors generated by the rules of Proposition 3. If $[\alpha_1\ \alpha_2\ \alpha_3\ \alpha_4]'$ is in $\mathbb{R}^4$, we will call a component α_i "lower" than α_j if $i > j$. (When the row index is larger, it is "lower" on the page.) Analogously, we can refer to the "lowest" or "last" component of a vector which satisfies a certain condition. We can then show the following:

Property 4 *Let f_n be a feature point generated by one of the rules in Proposition 3. Then the lowest (last) nonzero component of f_n is a 1.*

Proof of Property 4: It is clear by construction that legal successors generated by Rules 0, 1, and 2 must have a 1 as their lowest nonzero component. If a legal successor $f_n = s_1 + \ldots + s_M$ is generated by Rule 3, then by Property 3, the final nonzero component of f_n is the final nonzero component of one of the s_k's. Because each of the s_k's are generated by Rule 2, the final nonzero components of each must be a 1. ■

We next consider how the dimension d of a canonical list affects the feature points which appear in that list. In particular, notice that only Rule 1 enlarges the dimensionality of the previous feature list. That is, if f_n is a legal successor (not generated by Rule 1) of the canonical list $\mathbf{f}^-$, then $\dim(\mathbf{f}) = \dim(\mathbf{f}^-)$, where $\mathbf{f}$ is the feature list consisting of f_n appended to $\mathbf{f}^-$, i.e. $\mathbf{f} = \{\mathbf{f}^-; f_n\}$.

We use these observations and other properties of canonical lists to demonstrate a few lemmas which will simplify the proof of Proposition 3.

Lemma 2 *Let* $d = \dim(\mathbf{f})$, *where* $\mathbf{f}$ *is a canonical list. Then* e_i *appears as one (or more) of the vectors in the list* $\mathbf{f}$ $(e_i \in \mathbf{f})$ *for* $i \in \{1, \ldots, d\}$.

Proof of Lemma 2: If $d = 1$, the lemma is true by Rule 0. Suppose $d = 2$. Let $\mathbf{f}^i$ represent the sublist of $\mathbf{f}$ consisting of $\mathbf{f}$'s first i feature points. Because $\mathcal{S}_1 \subset \mathcal{S}_2$ implies that $\mathrm{span}(\mathcal{S}_1) \subset \mathrm{span}(\mathcal{S}_2)$, it follows that if $i \leq j$, $\dim(\mathbf{f}^i) \leq \dim(\mathbf{f}^j)$. Thus, $\dim(\mathbf{f}^i)$ is nondecreasing. Since $\dim(\mathbf{f}^1) = 1$ and $\dim(\mathbf{f}) = d = 2$, there must exist an index t such that $\dim(\mathbf{f}^{t-1}) = 1$ and $\dim(\mathbf{f}^t) = 2$. Because Rule 1 is the only means for generating a vector which increases a list's dimensionality, f_t is generated by Rule 1. Hence $f_t = e_2$. A similar argument holds for $d = 3$ and $d = 4$, so the lemma holds. ■

Lemma 3 *If there is a nonzero entry in row* i *of a canonical list* f, *then* $\dim(\mathbf{f}) \geq i$.

Proof of Lemma 3: Let $d = \dim(\mathbf{f})$. By Lemma 2 $e_j \in \mathbf{f}$ for $j \in \{1, \ldots, d\}$. Suppose that there is a nonzero entry in row i of feature point $f_k \in \mathbf{f}$ and $i > d$. Then the vectors in the set $\{f_k, e_1, \ldots, e_d\}$ are all linearly independent and thus span $d + 1$ dimensions. Since that set is a subset of the vectors in f, this contradicts the assumption that $d = \dim(\mathbf{f})$, which proves the lemma. ■

Lemma 4 *Suppose* $\mathbf{f}$ *is in canonical form,* $\dim(\mathbf{f}) = d$, *and* $\mathbf{f} = \mathbf{A}\ \mathbf{f}\ \mathrm{Diag}(\rho_i)$ *for some* $\mathbf{A} \in \mathrm{GL}(4)$, $\rho_i \in \mathbb{R}$, $\rho_i \neq 0$, *and* $i \in \{1, \ldots, n-1\}$. *Then the upper* $d \times d$ *submatrix of* $\mathbf{A}$ *is diagonal and invertible.*

Proof of Lemma 4: Suppose the conditions of the lemma hold and assume $i \in \{1, \ldots, d\}$. Then Lemma 2 shows that because $\dim(\mathbf{f}) = d$, the vectors e_i must appear in the list of feature points $\mathbf{f}$. Since $\mathbf{f} = \mathbf{A}\mathbf{f}\text{Diag}(\rho_i)$, it follows that $e_i = \rho^i \mathbf{A} e_i$, where $\rho^i \in \{\rho_1, \ldots, \rho_{n-1}\}$ and hence $\rho^i \neq 0$. Thus, A_i, the ith column of $\mathbf{A}$, must be $\frac{1}{\rho^i} e_i$. ■

The above lemma shows that if $\mathbf{A}$ maps any canonical list $\mathbf{f}$ to itself (modulo a nonzero scaling of each feature point), then $\mathbf{A}$ acts componentwise on any vector which would not extend the dimension of that canonical list. Put another way, as long as $\alpha_i = 0$ for $i > \dim(\mathbf{f})$, there are $\kappa_i > 0$ such that

$$\mathbf{A} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \end{bmatrix} = \begin{bmatrix} \kappa_1\alpha_1 \\ \kappa_2\alpha_2 \\ \kappa_3\alpha_3 \\ \kappa_4\alpha_4 \end{bmatrix},$$

The following two results extend this constraint on the structure of A in the case that rows are $\simeq_{\mathrm{m}}$-equivalent.

Lemma 5 *If* $\mathbf{f}$ *is in canonical form,* $\mathbf{f} = \mathbf{A}\mathbf{f}\text{Diag}(\rho_i)$ *and* $j \sim_{\mathrm{m}} k$ *in* $\mathbf{f}$*, then the* j*th column of* $\mathbf{A}$ *is* κe_j *and the* k*th column of* $\mathbf{A}$ *is* κe_k *for for some* $\kappa \in \mathbb{R}, \kappa \neq 0$.

Proof of Lemma 5: If the conditions of Lemma 5 hold, then by Lemmas 2 and 3 and the definition of $\sim_{\mathrm{m}}$, the following three vectors must appear in $\mathbf{f}$: e_j, e_k, and $\hat{e}$, where $\hat{e} \in \mathbb{R}^4$ contains a component of 1 in rows j and k. Thus, Lemma 4 implies Lemma 5 immediately if $j = k$. For the balance of the proof, we suppose $j \neq k$.

Let A_i represent the ith column of $\mathbf{A}$. By Lemma 4, we know that $A_i = \kappa_i e_i$ with $\kappa_i \neq 0$ and $i \in \{1, \ldots, d\}$. Further, $\mathbf{A}$ acts componentwise on $\hat{e}$:

$$\hat{e} = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \end{bmatrix} \implies \mathbf{A}\hat{e} = \begin{bmatrix} \kappa_1\alpha_1 \\ \kappa_2\alpha_2 \\ \kappa_3\alpha_3 \\ \kappa_4\alpha_4 \end{bmatrix}. \tag{5.4}$$

By assumption, however, $\hat{e} = \hat{\rho}\mathbf{A}\hat{e}$ for some $\hat{\rho} \neq 0$. Recalling that for $\hat{e}$, $\alpha_j = \alpha_k = 1$ and combining with Equation (5.4), we find that $1 = \hat{\rho}\kappa_j$ and $1 = \hat{\rho}\kappa_k$. Thus, $\kappa_j = \kappa_k$, and Lemma 5 holds. ■

Lemma 5 holds even if its conditions are weakened by replacing $\sim_{\mathrm{m}}$ by $\simeq_{\mathrm{m}}$. This is true because by the definition of $\simeq_{\mathrm{m}}$,

$$j \simeq_{\mathrm{m}} k \quad \Rightarrow \quad \exists\ i_1, \dots, i_4 \in \{1 \dots 4\} : j = i_1, k = i_4, \text{ and } i_{n-1} \sim_{\mathrm{m}} i_n \ \text{ for } \ n \in \{2 \dots 4\}. \tag{5.5}$$

Even if $j = k$ or $j \sim_{\mathrm{m}} k$, this holds true by assigning each intermediate row index $i_n = j$. By Equation 5.5 and repeated application of Lemma 5, we obtain the following:

Lemma 6 *Lemma 5 holds even if $\sim_{\mathrm{m}}$ is replaced by $\simeq_{\mathrm{m}}$ in the conditions of that statement.* ■

Finally, we introduce a result on how the row-partition of a feature under $\simeq_{\mathrm{proj}}$ affects the feature points in that feature:

Lemma 7 *Suppose $\{\mathcal{P}_1, \dots, \mathcal{P}_N\}$ is the row-partition of $\mathbf{f}$ under $\simeq_{\mathrm{m}}$. Then every $f_k \in \mathbf{f}$ contains nonzero components in rows indexed by only one of the $\mathcal{P}_i$'s.*

Proof of Lemma 7: Suppose f_k contains nonzero components in rows i and j with $i \in \mathcal{P}_1$ and $j \in \mathcal{P}_2$ with $\mathcal{P}_1 \neq \mathcal{P}_2$. Let $\{\mathcal{P}_1^{k-1}, \dots, \mathcal{P}_K^{k-1}\}$ be the row-partition of $\mathbf{f}^{k-1}$ under $\simeq_{\mathrm{m}}$, where $\mathbf{f}^{k-1}$ indicates the list of the first $k-1$ feature points of $\mathbf{f}$. From the definition of the row-partition, it is clear that we can rearrange indices so that $\mathcal{P}_1^{k-1} \subset \mathcal{P}_1$ and $\mathcal{P}_2^{k-1} \subset \mathcal{P}_2$.

Because f_k contains components in both $\mathcal{P}_1^{k-1}$ and $\mathcal{P}_2^{k-1}$, f_k could not have been added by Rules 0, 1, or 2, so it must have been added by Rule 3. Thus, modulo a possible rearrangement of indices we can write $f_k = s_1 + \ldots + s_M$ such that s_1 and s_2 are legal successors of $\mathbf{f}^{k-1}$, s_1 has all of its nonzero entries in $\mathcal{P}_1^{k-1}$, s_2 has all of its nonzero entries in $\mathcal{P}_2^{k-1}$, the ith component of s_1 is nonzero, and the jth component of s_2 is nonzero.

However, by Property 4 there is a row $r_1 \in \mathcal{P}_1^{k-1}$ such that the component of s_1 in row r_1 is equal to 1. Similarly, there is a row $r_2 \in \mathcal{P}_2^{k-1}$ such that the component of s_2 in row r_2 is equal to 1. Since $r_1 \sim_{\mathrm{m}} r_2$ through f_k, and $i \simeq_{\mathrm{m}} r_1$ and $j \simeq_{\mathrm{m}} r_2$ in $\mathbf{f}^{k-1}$, we know that $i \simeq_{\mathrm{m}} j$ in $\mathbf{f}$. This contradicts our assumption and finishes the lemma. ■

With these results on the structure of canonical lists, we are able to prove Proposition 3.

Proof of Proposition 3: Let $\mathcal{L}_{\mathrm{proj}}$ be the set of all feature lists in canonical form. By Rule 0, $\mathcal{L}_{\mathrm{proj}}$ is nonempty. Again, we need to demonstrate two things:

1. **Statement 1** If $\mathbf{f}, \mathbf{g}$ are any two distinct elements of $\mathcal{L}_{\mathrm{proj}}$, then $\mathbf{f}$ and $\mathbf{g}$ are *not* projectively equivalent, i.e. $\mathbf{f} \not\simeq_{\mathrm{proj}} \mathbf{g}$.

2. **Statement 2** If $\mathbf{f}$ is any nonempty list in $\mathcal{X}^n$ in homogeneous coordinates, then there exists a $\mathbf{g} \in \mathcal{L}_{\mathrm{proj}}$ such that $\mathbf{f} \simeq_{\mathrm{proj}} \mathbf{g}$.

Statement 1 We first show Statement 1 by induction on the number (n) of feature points in $\mathbf{f}$ and $\mathbf{g}$. Suppose $\mathbf{f}$ and $\mathbf{g}$ are distinct elements of $\mathcal{L}_{\mathrm{proj}}$. If they do not have the same number of feature points, then certainly $\mathbf{f} \not\simeq_{\mathrm{proj}} \mathbf{g}$. So assume they each have n feature points. Rule 0 constitutes a base case for $n = 1$. As inductive hypothesis, assume that for any two distinct feature lists of $n-1$ feature points $\mathbf{f}^-, \mathbf{g}^- \in \mathcal{L}_{\mathrm{proj}}$, it is true that $\mathbf{f}^- \not\simeq_{\mathrm{proj}} \mathbf{g}^-$.

Let $\mathbf{f}, \mathbf{g} \in \mathcal{L}_{\text{proj}}$ be two distinct lists with n feature points ($n \geq 2$). To prove Statement 1 it remains to show that $\mathbf{f} \not\simeq_{\text{proj}} \mathbf{g}$.

We first show that $\mathbf{f}^- = \mathbf{g}^-$, where $\mathbf{f}^-$ and $\mathbf{g}^-$ denote the first $n-1$ feature points of $\mathbf{f}$ and $\mathbf{g}$, respectively. If $\mathbf{f}^-$ and $\mathbf{g}^-$ are not identical, by the inductive hypothesis there is no projective transformation $\mathbf{A}^-$ and invertible diagonal matrix $\mathbf{D}^-$ such that $\mathbf{g}^- = \mathbf{A}^-\mathbf{f}^-\mathbf{D}^-$. If $\mathbf{g} \simeq_{\text{proj}} \mathbf{f}$, however, then $\mathbf{g} = \mathbf{AfD}$ and $\mathbf{A}^- = \mathbf{A}, \mathbf{D}^- =$ the upper $(n-1) \times (n-1)$ submatrix of $\mathbf{D}$ would satisfy $\mathbf{g}^- = \mathbf{A}^-\mathbf{f}^-\mathbf{D}^-$. Thus, if $\mathbf{f}^-$ and $\mathbf{g}^-$ are not identical, $\mathbf{f} \not\simeq_{\text{proj}} \mathbf{g}$.

Then, since $\mathbf{f}$ and $\mathbf{g}$ are distinct and $\mathbf{f}^- = \mathbf{g}^-$, the nth feature points of $\mathbf{f}$ and $\mathbf{g}$ must be distinct, i.e. $f_n \neq g_n$. Since we need to show that $\mathbf{f} \not\simeq_{\text{proj}} \mathbf{g}$, to proceed by contradiction we assume that $\mathbf{f} \simeq_{\text{proj}} \mathbf{g}$ and in particular that $\mathbf{g} = \mathbf{A}\ \mathbf{f}\ \text{Diag}(\rho_i)$ as in Property 2. We now consider the different methods by which f_n and g_n might have been added as legal successors of $\mathbf{f}^-$ and $\mathbf{g}^-$.

- Suppose f_n was added by Rule 1. If g_n was also added by Rule 1, then $f_n = g_n$, because $\mathbf{f}^- = \mathbf{g}^- \;\Rightarrow\; \dim(\mathbf{f}^-) = \dim(\mathbf{g}^-)$. If g_n was not added by Rule 1, then $\dim(\mathbf{f}) \neq \dim(\mathbf{g})$. Since $\mathbf{A}$ and $\text{Diag}(\rho_i)$ are invertible, $\mathbf{f} \not\simeq_{\text{proj}} \mathbf{g}$. This contradiction shows that f_n was not added by Rule 1.

  Notice that $\mathbf{g} = \mathbf{A}\ \mathbf{f}\ Diag(\rho_i) \Leftrightarrow \mathbf{f} = \mathbf{A}^{-1}\ \mathbf{g}\ \text{Diag}(\frac{1}{\rho_i})$, since $\mathbf{A} \in \text{GL}(4)$ and the ρ's are nonzero. As a result, $\mathbf{f}$ and $\mathbf{g}$ play symmetric roles in their relationship. Thus, since f_n could not have been added by Rule 1, neither could g_n.

- Suppose f_n was added by Rule 2. Suppose also that α_i and α_j (not necessarily distinct) are any two nonzero components of f_n, guaranteed to exist by Property 1. It then follows by Rule 2 that $i \simeq_{\text{m}} j$. From Lemma 6 we know that for all rows i in which f_n has a nonzero component, the ith column of $\mathbf{A}$ is κe_i

for some constant $\kappa \neq 0$. So at least for f_n we have

$$f_n = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \end{bmatrix} \implies g_n = \rho_n \mathbf{A} f_n = \rho_n \kappa \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \end{bmatrix}.$$

with $\rho_n \kappa \neq 0$. In other words, g_n is a scale multiple of f_n and thus the last nonzero component of f_n must be in the same row as the last nonzero component of g_n. Property 4 then demonstrates that these last nonzero elements must both equal 1, and hence $f_n = g_n$. This contradiction shows that f_n could not have been added by Rule 2. By analogous reasoning, g_n also could not have been added by Rule 2.

- The final case to consider occurs when f_n and g_n are added by Rule 3. Since $\mathbf{f}^- = \mathbf{g}^-$, we know that the row-partition of $\mathbf{f}^-$ under $\simeq_{\mathrm{m}}$ is identical to the row-partition of $\mathbf{g}^-$ under $\simeq_{\mathrm{m}}$. Call that partition $\{\mathcal{P}_1, \ldots, \mathcal{P}_N\}$. Let $f_n = s_1 + \ldots + s_N$, where s_k has its nonzero elements, if any, in $\mathcal{P}_k$. Note that this differs from the expression in Rule 3, because we are allowing some, but not all, of the s_k's to be the zero vector. Similarly, let $g_n = t_1 + \ldots + t_N$.

  By Lemma 4 and an argument identical to the previous case, we know that $t_i = \rho_n \kappa_i s_i$ for $i \in \{1, \ldots, N\}$. Again, s_i and t_i have any nonzero components, they must have a component of 1 in the same row, so that $s_i = t_i$ for each i. Because $\mathbf{A}$ is linear, we can sum these component vectors to obtain $g_n = f_n$.

Thus, none of the rules provided in the proposition suffice to make $f_n \neq g_n$ as well as $\mathbf{f} \simeq_{\mathrm{proj}} \mathbf{g}$. This completes the proof by contradiction of Statement 1. No two canonical lists are projectively equivalent.

Statement 2 We next turn to Statement 2: If $\mathbf{f}$ is any nonempty list in $\mathcal{X}^n$ in homogeneous coordinates, we need to show that there exists a $g \in \mathcal{L}_{\text{proj}}$ such that $\mathbf{f} \simeq_{\text{proj}} \mathbf{g}$. We proceed by induction on the number of feature points in $\mathbf{f}$.

If $\mathbf{f}$ has only one feature point, we can choose three other vectors f_2, f_3, f_4 in $\mathbb{R}^4$ such that the f_i form a basis of $\mathbb{R}^4$. Create a 4×4 matrix $\mathbf{F} = [f_1; f_2; f_3; f_4]$. Let $\mathbf{A} = \mathbf{F}^{-1}$ and $\rho_i = 1$. Then,

$$\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad = \quad \mathbf{A}\ \mathbf{f}\ \mathrm{Diag}(\rho_i),$$

which establishes a base case for the inductive proof.

We assume that statement 2 holds for positive integers less than n. Suppose that $\mathbf{h}$ is any list in $\mathcal{X}^n$ in homogeneous coordinates. Represent the list of the first $n-1$ feature points of $\mathbf{h}$ by $\mathbf{h}^-$. By the inductive hypothesis, we know there exists a $\mathbf{g}^-$, $\mathbf{A}$, and $\mathrm{Diag}(\rho_i)$ with $\mathbf{g}^-$ in canonical form such that

$$\mathbf{g}^- \quad = \quad \mathbf{A}\ \mathbf{h}^-\ \mathrm{Diag}(\rho_i).$$

Define $\mathbf{f} = \mathbf{A}\ \mathbf{h}^-\ \mathrm{Diag}(\rho_i)$ with $\rho_n = 1$. And let $\mathbf{f}^-$ be the leading sublist of $n-1$ feature points of $\mathbf{f}$. Because $\simeq_{\text{proj}}$ is transitive, we only have to show that there exists a canonical list $\mathbf{g}$ such that $\mathbf{g} \simeq_{\text{proj}} \mathbf{f}$.

Thus, we proceed to show that a canonical list $\mathbf{g}$ exists with $\mathbf{g} \simeq_{\text{proj}} \mathbf{f}$. We now have a head start, as $\mathbf{f}^-$ is already in canonical form. We know, too, that $f_n \in \mathbb{R}^4$ and $f_n \neq \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$. We consider several cases, each making different assumptions about how f_n relates to that of $\mathbf{f}^-$.

- Consider the case in which f_n has a nonzero component in a row lower than $d = \dim(\mathbf{f}^-)$. Then there must exist a permutation matrix $\mathbf{P}$ such that $\mathbf{P}e_i$

$= e_i$ for $i \leq d$ and $\mathbf{P}f_n$ has a nonzero element in its $d+1$st row. Let $\mathbf{P}f_n = [\alpha_1\ \alpha_2\ \alpha_3\ \alpha_4]'$. We next define a matrix $\mathbf{A} = [A_1; A_2; A_3; A_4]$.

Since $\alpha_{d+1} \neq 0$, let

$$A_{d+1} = \frac{1}{\alpha_{d+1}} \begin{bmatrix} -\alpha_1 \\ \cdots \\ 1 \\ \cdots \\ -\alpha_4 \end{bmatrix},$$

where the 1 appears instead of α_{d+1} as the $d+1$st component. For $i \neq d+1$ define

$$A_i = e_i.$$

Since $A_{d+1} = e_{d+1} + v$ with $v \cdot e_{d+1} = 0$, the A_i are independent. Thus, $\mathbf{A} \in \mathrm{GL}(4)$. As a result, $\mathbf{AP} \in \mathrm{GL}(4)$, too. Also, by defining $\rho_i = 1$ for $i \leq n$, all of the ρ_i's are nonzero. Using the above definitions for ρ_i and $\mathbf{A}$, let $g = \mathbf{AP}\ \mathbf{f}\ \mathrm{Diag}(\rho_i)$. It remains to show that $\mathbf{g}$ is in canonical form.

Note that the upper $d \times d$ submatrix of $\mathbf{A}$ is the $d \times d$ identity matrix, the upper $d \times d$ submatrix of $\mathbf{P}$ is the $d \times d$ identity matrix, and $\rho_i = 1$ for $i < n$. Thus, letting $\mathbf{g} = \{\mathbf{g}^-; g_n\}$, $\mathbf{g}^- = \mathbf{f}^-$, from Lemma 3. Since $\mathbf{f}^-$ is in canonical form, so is $\mathbf{g}^-$. It suffices, then, to show that $\mathbf{g}_n = \rho_n AP\mathbf{f}_n$ is a legal successor of $\mathbf{g}^-$. Computing g_n componentwise,

$$g_n = (\rho_n)\mathbf{A}\frac{1}{\alpha_{d+1}} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \end{bmatrix} = \frac{1}{\alpha_{d+1}} \begin{bmatrix} \alpha_1 - \alpha_1 \\ \cdots \\ 1 \\ \cdots \\ \alpha_4 - \alpha_4 \end{bmatrix} = e_{d+1}.$$

Since e_{d+1} is a legal successor of $\mathbf{g}^-$ whose dimension is d, this first case is complete.

- Consider now that f_n does not have a nonzero element in a row lower than

$d = \dim(\mathbf{f}^-)$. Let $\{\mathcal{P}_1, \ldots, \mathcal{P}_N\}$ be the row-partition of $\mathbf{f}^-$ under $\simeq_{\mathrm{m}}$. Write $f_n = s_1 + \ldots + s_N$, where each s_i has all of its nonzero components in rows in $\mathcal{P}_i$. Some of the s_i's might be the zero vector. With a possible rearrangement of indices, however, we can assume that $f_n = s_1 + \ldots + s_M$ with each s_i $(1 \leq i \leq M)$ containing at least one nonzero component and each s_i only having nonzero components in rows in $\mathcal{P}_i$. Let $d = \dim(\mathbf{f}) = \dim(\mathbf{f}^-)$.

We will need to define several quantities. Let $f_n = [\alpha_1\ \alpha_2\ \alpha_3\ \alpha_4]'$. Let i_1 be the largest integer (row number corresponding to the physically lowest row) in $\mathcal{P}_1$ such that α_{i_1} is nonzero. Such an integer must exist from the previous decomposition of f_n. Similarly, define i_k to be the largest integer in $\mathcal{P}_k$ such that α_{i_k} is nonzero. Again, such an i_k exists for $k \in \{1, \ldots, M\}$).

We will also need to distinguish which of the $\mathcal{P}_i$'s contains a particular row number. To this end, we will use the notation $\mathcal{P}_{(1)}$ to denote that element of $\{\mathcal{P}_1, \ldots, \mathcal{P}_N\}$ which contains the integer (row number) 1. Similarly, $\mathcal{P}_{(j)}$ is that element of the partition containing row number j. Note that even if $j \neq k$, it is possible that $\mathcal{P}_{(j)} = \mathcal{P}_{(k)}$. Analogously, let $i_{(j)}$ represent the largest integer in $\mathcal{P}_{(j)}$ such that $\alpha_{i_{(j)}}$ is nonzero. Such an $i_{(j)}$ must exist for some, but not necessarily all, $j \leq d$. Note that by these definitions, for any $j, k \in \{1, \ldots, 4\}$

$$j \simeq_{\mathrm{m}} k \implies \mathcal{P}_{(j)} = \mathcal{P}_{(k)} \text{ and } i_{(j)} = i_{(k)}. \tag{5.6}$$

With this notation we can define a matrix $\mathbf{A}$ as follows. For $j \in \{1, \ldots, d\}$ such that $i_{(j)}$ exists, let

$$A_j = \frac{1}{\alpha_{i_{(j)}}} e_j.$$

For other $j \in \{1, \ldots, 4\}$, let $A_j = e_j$. In addition, for each feature-point index k ($k \in \{1, \ldots, n-1\}$) let the integer r_k denote the index of the lowest row of f_k containing a nonzero entry. (Since $\mathbf{f}^-$ is a canonical list, by Property 4 that entry must be a 1.) Then for $k \in \{1, \ldots, n-1\}$ define

$$\rho_k = \alpha_{i_{(r_k)}}.$$

When $k = n$, let $\rho_n = 1$. As this is unwieldy at first glance, we provide some intuition behind this assignment as follows.

The matrices $\mathbf{A}$ and $\mathrm{Diag}(\rho_i)$ are intended to transform f into canonical form. Since $\mathbf{A}$ is diagonal but not the identity matrix, it will scale some or all of the components of the feature points in $\mathbf{f}^-$. But $\mathbf{f}^-$ is already in canonical form. In particular, for each f_k ($k < n$), the final nonzero component of f_k is a 1. If that nonzero component is in row r_k, $\mathbf{A}f_k$ will have the value $\frac{1}{\alpha_{i_{(r_k)}}}$ in row r_k. Since g_k must have the value 1 in row r_k, we need to define ρ_k as we have above.

We are ready to check that $\mathbf{g}$ is a canonical list, where $\mathbf{g} = \mathbf{A}\ \mathbf{f}\ \mathrm{Diag}(\rho_i)$. We do this in two steps. First, we show that $g_k = f_k$ for $k < n$. In this case since $\mathbf{f}^-$ is a canonical list, so must $\mathbf{g}^-$ be. Second, we show that $g_n = \rho_n \mathbf{A} f_n$ must be a legal successor of $\mathbf{g}^-$, demonstrating that $\mathbf{g}$ is in canonical form.

By Lemma 7 we know that for $k < n$, f_k must have its nonzero components in a single element of $\{\mathcal{P}_1, \ldots, \mathcal{P}_M\}$. One row in which f_k has a nonzero element is r_k. From this, the definition of $\mathbf{A}$, and Equation (5.6), we have $\mathbf{A}f_k = \frac{1}{\alpha_{i_{(r_k)}}} f_k$. Thus, $g_k = \rho_k \mathbf{A} f_k = \frac{\alpha_{i_{(r_k)}}}{\alpha_{i_{(r_k)}}} f_k = f_k$, so that $\mathbf{g}^- = \mathbf{f}^-$.

Let $g_n = \rho_n \mathbf{A} f_n = A f_n$. Then by the linearity of $\mathbf{A}$, $g_n = A\mathbf{A}s_1 + \ldots \mathbf{A}s_M$.

By an argument similar to the previous paragraph's, we know that each for each i $(1 \leq i \leq M)$ such that s_i has nonzero components, $\mathbf{A}s_i$ is a legal successor of $\mathbf{g}^-$. Hence, by Rule 3 (or Rule 2 if $M = 1$), g_n is a legal successor of $\mathbf{g}^-$, which completes the proof.

As a result, $\mathcal{L}_{\text{proj}}$ is, in fact, the set of elementary tasks decidable on $\mathcal{C}_{\text{proj}}$. ■

We use the word "measured" to describe rows equivalent under the $\simeq_{\text{m}}$ relation because, as Proposition 3 shows, measured rows in a canonical list $\mathbf{f}$ have sufficient feature-point structure to admit general positioning within the dimension(s) corresponding to those rows (by Rule 2). Hence, the feature points involved form a basis for "measuring" those dimensions.

5.5 Elementary Tasks of $\mathcal{T}_{\text{aff}}$

The additional structure of $\simeq_{\text{aff}}$ makes the description of the elementary tasks of $\mathcal{T}_{\text{aff}}$ simpler than those of $\mathcal{T}_{\text{proj}}$. In this case $\mathcal{X} = \mathbb{R}^3$.

We use the same definition of dimension as in the projective case, $\dim(\mathbf{f})$ is the number of dimensions spanned by the feature points (i.e., the column vectors) in $\mathbf{f}$. Now, however, $0 \leq \dim(\mathbf{f}) \leq 3$, with $\dim(\text{span}(\left[\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right])) = 0$; e_i refers to the ith canonical basis vector of $\mathbb{R}^3$ and by convention we let $e_0 = \left[\begin{smallmatrix}0\\0\\0\end{smallmatrix}\right]$.

With these changes, we claim the following characterization of the set of elementary tasks in $\mathcal{T}_{\text{persp}}$.

Proposition 4 *Let $\mathcal{X} = \mathbb{R}^3$. Suppose $\mathbf{f}^- = \{f_1 \ldots f_{n-1}\} \in \mathcal{X}^{n-1}$ is in canonical form. Let $d = \dim(\mathbf{f}^-)$ Also, suppose $\mathbf{f} = \{f_1 \ldots f_n\} \in \mathcal{X}^n$ is in canonical form if and only if f_n is a legal successor of $\mathbf{f}^-$, i.e., if and only if f_n satisfies one of the following rules:*

Rule 0 *The only feature list of length 1 in canonical form is* $\mathbf{f} = \left\{ \begin{smallmatrix} 0 \\ 0 \\ 0 \end{smallmatrix} \right\}$ *.*

Rule 1 $f_n = e_i$*, with* $i = d + 1$ *and* $i \leq 3$*.*

Rule 2 $f_n = \sum_{i=0}^{d} \alpha_i e_i$*, for* $\alpha_i \in \mathbb{R}$*.*

Then, for each $\simeq_{\text{persp}}$*-equivalence class of* $\mathcal{X}^n$*, there is a unique representative element formed by successive application of these rules.*

Three properties will assist in the proof of Proposition 4. First, an explicit definition of affine transformations:

Property 5 *If* $\mathbf{f}$ *and* $\mathbf{g}$ *are lists of* n *feature points in* $\mathcal{X}$*, then*

$$\mathbf{f} \simeq_{\text{aff}} \mathbf{g} \iff \exists \mathbf{A} \in \mathrm{GL}(3),\ t \in \mathbb{R}^3 : \ \ \mathbf{g} = \mathbf{A}\mathbf{f} + t,$$

where $\mathrm{GL}(3)$ *consists of all* 3×3 *invertible matrices. Lists with different numbers of feature points are not* $\simeq_{\text{aff}}$*-equivalent.* ∎

Because the $\mathbf{A}$ in the above definition is invertible, like projective transformations affine transformations preserve dimensionality:

Property 6 *If* $\mathbf{f} \simeq_{\text{aff}} \mathbf{g}$*, then* $\dim(\mathbf{f}) = \dim(\mathbf{g})$*.* ∎

Comparable to Lemma 2 from the previous proposition, the following lemma asserts the presence of the basis vectors e_i in canonical representatives with dimension at least i. Recall that $e_0 = \left[\begin{smallmatrix} 0 \\ 0 \\ 0 \end{smallmatrix} \right]$.

Lemma 8 *Let* $d = \dim(\mathbf{f})$*, where* $\mathbf{f}$ *is a canonical list. Then* e_i *appears as at least one of the vectors in the list* $\mathbf{f}$ *for* $0 \leq i \leq d$

Proof of Lemma 2: If $d = 0$, the lemma is true by Rule 0. Suppose $d = 1$ and let $\mathbf{f}^i$ represent the sublist of $\mathbf{f}$ consisting of $\mathbf{f}$'s first i feature points. Because $\mathcal{S}_1 \subset \mathcal{S}_2$ implies that $\mathrm{span}(\mathcal{S}_1) \subset \mathrm{span}(\mathcal{S}_2)$, it follows that if $i \leq j$, $\dim(\mathbf{f}^i) \leq \dim(\mathbf{f}^j)$. Thus, as i increases, $\dim(\mathbf{f}^i)$ is nondecreasing. Since $\dim(\mathbf{f}^1) = 0$ and $\dim(\mathbf{f}) = d = 2$, there must exist a transition index t such that $\dim(\mathbf{f}^{t-1}) = 0$ and $\dim(\mathbf{f}^t) = 1$. Because Rule 1 is the only means for generating a vector which increases a list's dimensionality, f_t is generated by Rule 1. Hence $f_t = e_1$. A similar argument holds for $d = 2$ and $d = 3$, so the lemma holds. ■

The proof of the characterization of affine elementary tasks builds on these results.

Proof of Proposition 4: Define $\mathcal{L}_{\text{aff}}$ and statements 1 and 2 analogously with the previous two propositions.

Statement 1 Let $\mathbf{f}, \mathbf{g} \in \mathcal{L}_{\text{aff}}$ and proceed by induction on the number of feature points n in both $\mathbf{f}$ and $\mathbf{g}$. As in Proposition 3, by inductive hypothesis the initial sublists of $\mathbf{f}$ and $\mathbf{g}$ can be considered equal, $\mathbf{f}^- = \mathbf{g}^-$. Assume $f_n \neq g_n$, but that $\mathbf{f} \simeq_{\text{aff}} \mathbf{g}$. We proceed to a contradiction by considering the different ways f_n and g_n might have been added to $\mathbf{f}^-$ and $\mathbf{g}^-$. First, since $\dim(\mathbf{f}^-) = \dim(\mathbf{g}^-)$, if Rule 1 added both f_n and g_n then $f_n = g_n$, a contradiction. Thus, at least one of f_n and g_n came from Rule 2. If Rule 2 added *only one* of f_n and g_n, however, $\dim(\mathbf{f}) \neq \dim(\mathbf{g})$ — again a contradiction.

Consider the final possibility that Rule 2 added both f_n and g_n. Since $\mathbf{f} \simeq_{\text{aff}} \mathbf{g}$, use Property 5 to write $\mathbf{g} = \mathbf{Af} + t$ for $A \in GL(3)$ and $t \in \mathbb{R}^3$. It follows that $t = 0$ since $g_1 = f_1 = e_0$. In addition, the upper $d \times d$ submatrix of $\mathbf{A}$ must be the identity matrix $\mathbf{I}_d$. This follows from the presence of e_i (for $0 \leq i \leq d$) in $\mathbf{f}^-$ and $\mathbf{g}^-$ (Lemma 8) and the fact that $\mathbf{f}^- = \mathbf{g}^-$. Since Rule 2 added f_n, f_n has nonzero components only in its upper d rows. Since the upper $d \times d$ submatrix of $\mathbf{A}$ is $\mathbf{I}_d$,

$g_n = f_n$ — a contradiction which completes the proof of statement 1.

Statement 2 Let $\mathbf{h}$ be an arbitrary element of $\mathcal{X}^n$. Since statement 2 holds for $n = 1$, suppose $n > 1$ and assume that this statement holds for positive integers less than n. Let $\mathbf{h}^-$ be the leading sublist of $n-1$ points of $\mathbf{h}$. By the inductive hypothesis there exists an $\mathbf{f}^- \in \mathcal{L}_{\text{aff}}$ such that $\mathbf{h}^- \simeq_{\text{aff}} \mathbf{f}^-$, i.e., $\mathbf{f}^- = \mathbf{A}^*\mathbf{h}^- + t^*$. Let $f_n = \mathbf{A}^* h_n + t^*$ and $\mathbf{f} = \{\mathbf{f}^-; f_n\}$. By the transitivity of $\simeq_{\text{aff}}$, it suffices to show that some $\mathbf{g} \in \mathcal{L}_{\text{aff}}$ exists such that $\mathbf{g} \simeq_{\text{aff}} \mathbf{f}$. Consider cases based on the structure of f_n with respect to $\mathbf{f}^-$. Let $d = \dim(\mathbf{f}^-)$.

- If $d = 3$, $\mathbf{f}$ is in canonical form; letting $\mathbf{g} = \mathbf{f}$ satisfies $\mathbf{g} \simeq_{\text{aff}} \mathbf{f}$ and $\mathbf{g} \in \mathcal{L}_{\text{aff}}$.

- If $d = 2$ and $\mathbf{f}_n$ has a zero in its third component, $\mathbf{f}$ is in canonical form. Suppose, then, that $d = 2$ and $f_n = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{bmatrix}$ with $\alpha_3 \neq 0$. Let

$$\mathbf{g} = \begin{bmatrix} 1 & 0 & -\frac{\alpha_1}{\alpha_3} \\ 0 & 1 & -\frac{\alpha_2}{\alpha_3} \\ 0 & 0 & \frac{1}{\alpha_3} \end{bmatrix} \mathbf{f} \,.$$

  Note that this matrix has determinant $\frac{1}{\alpha_3}$ and so is invertible. Also, $g_n = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$. Lemma 8 implies that every feature point in $\mathbf{f}^-$ has nonzero components in only the first two coordinate rows. By the above relationship, then, each $g_i = f_i$ for $1 \leq i < n$. Hence $\mathbf{g}$ is in canonical form (by Rule 1) and $\mathbf{g} \simeq_{\text{aff}} \mathbf{f}$.

- If $d = 1$ and $\mathbf{f}_n$ has a zero in its second and third components, $\mathbf{f}$ is in canonical form. Suppose, then, that $d = 1$ and $f_n = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{bmatrix}$ with $\alpha_2 \neq 0$ or $\alpha_3 \neq 0$. Without loss of generality suppose $\alpha_2 \neq 0$ (otherwise replace future occurrences of α_2 with α_3 and rearrange the following matrix). Define

$$\mathbf{g} = \begin{bmatrix} 1 & -\frac{\alpha_1}{\alpha_2} & 0 \\ 0 & \frac{1}{\alpha_2} & 0 \\ 0 & -\frac{\alpha_3}{\alpha_2} & 1 \end{bmatrix} \mathbf{f} \,.$$

Again, this matrix is invertible since its three columns are linearly independent. Also, $g_n = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$. As in the above case Lemma 8 implies that every feature point in $\mathbf{f}^-$ has nonzero components in only the first coordinate row. By the above relationship, then, each $g_i = f_i$ for $1 \leq i < n$. Hence $\mathbf{g}$ is in canonical form (by Rule 1) and $\mathbf{g} \simeq_{\text{aff}} \mathbf{f}$.

- If $d = 0$ then all of the feature points of $\mathbf{f}^-$ are equal to e_0. If $f_n = e_0$, $\mathbf{f}$ is already in canonical form. Otherwise, let $r'_1 = \frac{f_n}{||f_n||^2}$, so that r_1 is a row vector in the same direction as f_n. Let r'_2 and r'_3 be two vectors orthogonal to f_n such that r'_1, r'_2, and r'_3 form a basis of $\mathbb{R}^3$. Build a matrix $\mathbf{A}$ from those rows:

$$\mathbf{A} = \begin{bmatrix} r_1 \\ r_2 \\ r_3 \end{bmatrix} .$$

Then $\mathbf{A}$ is invertible and $\mathbf{A}f_n = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$. Thus, if $\mathbf{g} = \mathbf{A}\mathbf{f}$, $\mathbf{g}$ is in canonical form and $\mathbf{g} \simeq_{\text{aff}} \mathbf{f}$.

Hence, statement 2 and Proposition 4 holds. ■

5.6 Elementary Tasks of $\mathcal{T}_{\text{persp}}$

Fundamentally, the structure obtainable from perspective cameras differs from affine structure in that only one dimension needs to provide a "measured point," rather than all three, before universal metrical positioning is possible. As a result, the rules for constructing similarity-based coordinate systems constrain legal successors even less than in the previous proposition. The following set of rules characterizes the set of elementary tasks in $\mathcal{T}_{\text{persp}}$.

Proposition 5 *Let* $\mathcal{X} = \mathbb{R}^3$*. Suppose* $\mathbf{f}^- = \{f_1 \dots f_{n-1}\} \in \mathcal{X}^{n-1}$ *is in canonical form. Let* $d = \dim(\mathbf{f}^-)$ *Also, suppose* $\mathbf{f} = \{f_1 \dots f_n\} \in \mathcal{X}^n$ *is in canonical form if and only if* f_n *is a legal successor of* $\mathbf{f}^-$*, i.e., if and only if* f_n *satisfies one of the following rules:*

Rule 0 *The only feature list of length 1 in canonical form is* $\mathbf{f} = \left\{ \begin{smallmatrix} 0 \\ 0 \\ 0 \end{smallmatrix} \right\}$.

Rule 1 $f_n = \left[\begin{smallmatrix} 0 \\ 0 \\ 0 \end{smallmatrix} \right]$ *or* $\left[\begin{smallmatrix} 1 \\ 0 \\ 0 \end{smallmatrix} \right]$*, if* $d = 0$.

Rule 2 $f_n = \left[\begin{smallmatrix} \alpha_1 \\ \alpha_2 \\ 0 \end{smallmatrix} \right]$*, with* $\alpha_i \in \mathbb{R}$*, if* $d = 1$.

Rule 3 $f_n = \left[\begin{smallmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{smallmatrix} \right]$*, with* $\alpha_i \in \mathbb{R}$*, provided* $d \geq 2$.

Then, for each $\simeq_{\text{persp}}$*-equivalence class of* $\mathcal{X}^n$*, there is a unique representative element formed by successive application of these rules.*

Proof of Proposition 5: Again, we show two things: first, that if $\mathbf{f}, \mathbf{g} \in \mathcal{L}_{\text{persp}}$, then $\mathbf{f} \simeq_{\text{persp}} \mathbf{g} \Rightarrow \mathbf{f} = \mathbf{g}$. Second, we show that if $\mathbf{f}$ is any list of feature points in $\mathcal{X}^n$, then there exists a $\mathbf{g} \in \mathcal{L}_{\text{persp}}$ such that $\mathbf{g} \simeq_{\text{persp}} \mathbf{f}$.

We use the following fact about $\simeq_{\text{persp}}$-equivalence, which follows directly from the definition of $\simeq_{\text{persp}}$:

Property 7 *If* $\mathbf{f}$ *and* $\mathbf{g}$ *are lists of* n *feature points represented as columns of homogeneous coordinates, then*

$$\mathbf{f} \simeq_{\text{persp}} \mathbf{g} \iff \exists \mathbf{R} \in \mathrm{SO}(3),\ \lambda \in \mathbb{R},\ t \in \mathbb{R}^3 : \mathbf{g} = \lambda \mathbf{R} \mathbf{f} + t,$$

where $\lambda \neq 0$*. Lists with different numbers of feature points are not* $\simeq_{\text{persp}}$*-equivalent.*

■

In addition, note that as legal successors are added to a feature, the dimension of the resulting feature either must remain constant or increase. Since the only points which are legal successors to features of dimension 0 are the vectors $[0\ 0\ 0]'$ and $[1\ 0\ 0]'$ and the former does not increase the dimension of the resulting list, we can assert the following:

Property 8 *If* $\mathbf{f} \in \mathcal{L}_{\text{persp}}$ *and* $\dim(\mathbf{f}) > 0$*, then* $[1\ 0\ 0]' \in \mathbf{f}$. ■

Property 8 implies that the span of any set of feature points within a canonical list of dimension 1 is the set of vectors $[\alpha\ 0\ 0]'$, with $\alpha \in \mathbb{R}$. The following property answers the question of what has to be in a canonical list of dimension 2.

Property 9 *If* $\mathbf{f} \in \mathcal{L}_{\text{persp}}$ *and* $\dim(\mathbf{f}) \geq 2$*, then there is some feature point* $f_k \in \mathbf{f}$ *such that* $f_k = \left[\begin{smallmatrix} \alpha_1 \\ \alpha_2 \\ 0 \end{smallmatrix}\right]$ *with* $\alpha_2 \neq 0$.

Proof of Property 9: Since the dimension of a canonical list is nondecreasing as feature points are added, there is some initial sublist of $\mathbf{f}$, $\mathbf{f}^k$, such that $\dim(\mathbf{f}^{k-1}) = 1$ and $\dim(\mathbf{f}^k) = 2$. Thus, f_k was added to the list by Rule 2, and so $f_k = \left[\begin{smallmatrix} \alpha_1 \\ \alpha_2 \\ 0 \end{smallmatrix}\right]$. Suppose $\alpha_2 = 0$. Then, from Property 8 and the observation about the span of canonical lists of dimension 1, it follows that $\dim(\mathbf{f}^k)$ would equal 1. Since $\dim(\mathbf{f}^k) = 2$, $\alpha_2 \neq 0$. ■

Statement 1 Suppose that $\mathbf{f}$ and $\mathbf{g}$ are elements of $\mathcal{L}_{\text{persp}}$ and that $\mathbf{f} \simeq_{\text{persp}} \mathbf{g}$. We know immediately that $\mathbf{f}$ and $\mathbf{g}$ have the same number of feature points, call it n. If $n = 1$ we know that $\mathbf{f} = \mathbf{g}$ from Rule 0. As an inductive hypothesis, suppose that any two $\simeq_{\text{persp}}$-equivalent elements of $\mathcal{L}_{\text{persp}}$ each with $n-1$ feature points must be identical. Let $\mathbf{f}^-$ and $\mathbf{g}^-$ be the leading lists of $n-1$ elements ($n > 1$) of $\mathbf{f}$ and $\mathbf{g}$, respectively. By inductive hypothesis, $\mathbf{f}^- = \mathbf{g}^-$. We only need to show that $f_n = g_n$.

Using Property 7, we will presume that for some $\mathbf{R} \in \mathrm{SO}(3), t \in \mathbb{R}^3, \lambda \in \mathbb{R}, \lambda \neq 0$

$$\mathbf{g} = \lambda \mathbf{R}\mathbf{f} + t.$$

Note that since $[0\ 0\ 0]'$ must be the first column in list $\mathbf{f}$ and list $\mathbf{g}$ by Rule 0, we can immediately conclude that for $\mathbf{g} = \lambda \mathbf{R}\mathbf{f} + t$, $t = 0$. Let $d = \dim(\mathbf{f}^-) = \dim(\mathbf{g}^-)$. As in the projective case, we consider several different cases

- Suppose $d = 0$. Then f_n and g_n were generated by Rule 1. If $f_n = [0\ 0\ 0]'$, then by Property 7 $g_n = [0\ 0\ 0]' = f_n$. Similarly, since $\mathbf{f}$ and $\mathbf{g}$ play symmetric roles under the equivalence relation $\simeq_{\text{persp}}$, $g_n = [0\ 0\ 0]' \Rightarrow f_n = [0\ 0\ 0]' = g_n$. Thus by the exclusion of any other possibilities, $f_n = [1\ 0\ 0]' \Leftrightarrow g_n = [1\ 0\ 0]'$. Hence, $g_n = f_n$.

- Suppose $d = 1$. Notice that by Property 8 this implies $[1\ 0\ 0]' \in \mathbf{f}^-$. Then $\mathbf{f}^- = \mathbf{g}^-$ implies that $e_1 = \lambda \mathbf{R} e_1$, so that using R_i to denote the ith column of $\mathbf{R}$, $R_1 = e_1$ and $\lambda = 1$.

  Because $d = 1$, we know that f_n and g_n are legal successors to $\mathbf{f}^- = \mathbf{g}^-$ by Rule 2. Let

  $$f_n = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ 0 \end{bmatrix} \text{ and } g_n = \begin{bmatrix} \beta_1 \\ \beta_2 \\ 0 \end{bmatrix},$$

  with the $\alpha_i, \beta_i \in \mathbb{R}$. Since $R_1 = e_1$ and $\lambda = 1$, we conclude that $\beta_1 = \alpha_1$ and

  $$\begin{bmatrix} \alpha_1 \\ \beta_2 \\ 0 \end{bmatrix} = \mathbf{R} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ 0 \end{bmatrix}.$$

  Since $\mathbf{R} \in \mathrm{SO}(3)$, $\mathbf{R}$ preserves Euclidean lengths. Thus, $\beta_2 = \alpha_2$, so that $f_n = g_n$.

- Suppose $d \geq 2$. By Properties 8 and 9, there are feature points $f_i, f_j \in \mathbf{f}^-$ such that $f_i = [0\ 0\ 0]'$ and $f_j = [\alpha_1\ \alpha_2\ 0]'$ with $\alpha_2 \neq 0$. By the reasoning for the case of $d = 1$, we know that $\lambda = 1$ and $R_1 = e_1$. Thus, we can write $\mathbf{R}$ as

  $$\mathbf{R} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{bmatrix}.$$

  But since

  $$\begin{bmatrix} \alpha_1 \\ \alpha_2 \\ 0 \end{bmatrix} = \mathbf{R} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ 0 \end{bmatrix},$$

  for some $\alpha_2 \neq 0$, we conclude that $\theta = 0$. Thus, $\mathbf{R}$ is the 3×3 identity matrix, so that $f_n = g_n$

As a result, any two elements of $\mathcal{L}_{\text{persp}}$ which are $\simeq_{\text{persp}}$-equivalent are also identical.

Statement 2 We turn next to the second statement we need to show – namely that for any $\mathbf{f} \in (\mathbb{R}^3)^n$, there exists some $\mathbf{g} \in \mathcal{L}_{\text{persp}}$ such that $\mathbf{f} \simeq_{\text{persp}} \mathbf{g}$. Again, we approach this by induction. Let $\mathbf{f}$ be some list of n feature points. If $n = 1$, $\mathbf{f} \simeq_{\text{persp}} \mathbf{g}$ via

$$g_1 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} = \lambda \mathbf{R} f_1 + (-f_1),$$

where $\lambda = 1$, $\mathbf{R}$ is the identity matrix, and we have let $t = -f_1$.

So, suppose $n > 1$. We take as an inductive hypothesis that for all features of $n - 1$ feature points, there is an element of $\mathcal{L}_{\text{persp}}$ which is $\simeq_{\text{persp}}$-equivalent to it. Suppose $\mathbf{h}$ is a list of n feature points, so that $\mathbf{h}^-$ has $n - 1$ feature points. There is some $\mathbf{f}^- \in \mathcal{L}_{\text{persp}}$ such that $\mathbf{f}^- \simeq_{\text{persp}} \mathbf{h}^-$ via $\mathbf{f}^- = \lambda \mathbf{R} \mathbf{h}^- + t$. Let $f_n = \lambda \mathbf{R} h_n + t$ and call $\mathbf{f} = \{\mathbf{f}^-; f_n\}$. By the transitivity of $\simeq_{\text{persp}}$, it suffices to show that there is some $\mathbf{g} \in \mathcal{L}_{\text{persp}}$ such that $\mathbf{g} \simeq_{\text{persp}} \mathbf{f}$. We consider different cases depending on the structure of f_n. Let $d = \dim(\mathbf{f}^-)$.

- If $d = 2$, we're done because $\mathbf{f}$ is already in canonical form.
- If $d = 1$, we're done if f_n has a 0 in its third component. So suppose

  $$f_n = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{bmatrix},$$

  with $\alpha_3 \neq 0$. Choose θ such that $\alpha_2 \sin(\theta) + \alpha_3 \cos(\theta) = 0$. If $\alpha_2 = 0$, let $\theta = \frac{\pi}{2}$. Otherwise, assign $\theta = \tan^{-1}(-\frac{\alpha_3}{\alpha_2})$.

  Then, letting

  $$\mathbf{g} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\theta) & -\sin(\theta) \\ 0 & \sin(\theta) & \cos(\theta) \end{bmatrix} \mathbf{f}$$

  creates a new list $\mathbf{g}$. Since $d = 1$, we know that every feature point in $\mathbf{f}^-$ is of the form $[\alpha\ 0\ 0]'$, for some $\alpha \in \mathbb{R}$. Thus, every point in $\mathbf{g}^-$ is identical to the corresponding point in $\mathbf{f}^-$. In addition, by our choice of θ, g_n has its third component equal to 0, so it is a legal successor to $\mathbf{g}^- = \mathbf{f}^-$ by Rule 2.
- Suppose $d = 0$. Then the only vector in $\mathbf{f}^-$ is $[0\ 0\ 0]'$. If $f_n = [0\ 0\ 0]'$, $\mathbf{f}$ is already in canonical form. Suppose the f_n has a nonzero component. Let $\mathbf{R} \in \mathrm{SO}(3)$ be a rotation which makes the last two components of $\mathbf{R}f_n$ equal to 0. Since R preserves Euclidean length, the first component of $\mathbf{R}f_n$ is nonzero, call it α. Let $\lambda = \frac{1}{\alpha}$. Then, let $\mathbf{g} = \lambda \mathbf{R}\mathbf{f}$. The first $n-1$ feature points in $\mathbf{g}$ are all $[0\ 0\ 0]'$, and $g_n = [1\ 0\ 0]'$, so that $\mathbf{g}$ is in canonical form.

Thus, any feature list in $\mathcal{X}^n$ is $\simeq_{\text{persp}}$-equivalent to some element of $\mathcal{L}_{\text{persp}}$, which completes the proof of Proposition 5. ■

The following section takes advantage of the computational means for generating all elementary (and thus all) tasks in $\mathcal{T}_{\text{inj}}$, $\mathcal{T}_{\text{proj}}$, $\mathcal{T}_{\text{aff}}$, and $\mathcal{T}_{\text{persp}}$. It presents examples of several elementary tasks and nonelementary tasks which Chapter 6 will press into service.

5.7 Examples of Decidable Tasks

The first three examples of this section describe tasks and verifying encodings which take advantage of the reconstructibility of task space. Each of these is *metrical* in nature, i.e., a task involving positioning a feature point to particular coordinate values with respect to several observed basis points. The fourth example describes a coplanarity task, a coordinate-free constraint similar to T_{pp} and T_{col}.

Affine Positioning The example task T_{midpt} of Chapter 1 specializes the more general positioning capability of affinely calibrated cameras. For example, the elementary task T_{aff} with canonical representative

$$\left\{ \begin{smallmatrix} 0 & 1 & 0 & 0 & \alpha & \beta_1 & \gamma_1 \\ 0 & 0 & 1 & 0 & 0 & \beta_2 & \gamma_2 \\ 0 & 0 & 0 & 1 & 0 & 0 & \gamma_3 \end{smallmatrix} \right\} ,$$

inlcudes all of the structure of affine space: a basis, a 1d constraint (a general Euclidean-ratio specification, such as the midpoint task's), a 2d constraint, and general 3d positioning.

Assume that the seven points comprising a feature fall in subspaces indicated by the above canonical list — f_1, f_2, and f_5 on a single line; f_1, f_2, f_3, and f_6 in a plane; f_1, f_2, f_3, and f_4 in general position. Then, reconstruction of the points' coordinates provides a verifying encoding for $T_{\text{aff}}(\mathbf{f}) = 0$. Let r_{aff} be the reconstructive function expressed by Equation (2.10). For any observed feature $\mathbf{y}$, let $\mathbf{f} = g(r_{\text{aff}}(\mathbf{y}))$, where g

is an affine transformation chosen so that the feature points of $\mathbf{f}$ are $\left\{\begin{smallmatrix} 0 & 1 & 0 & 0 & \alpha^* & \beta_1^* & \gamma_1^* \\ 0 & 0 & 1 & 0 & 0 & \beta_2^* & \gamma_2^* \\ 0 & 0 & 0 & 1 & 0 & 0 & \gamma_3^* \end{smallmatrix}\right\}$. Such a g exists as long as the subspace constraints listed above are satisfied. Then

$$E_{\text{aff}}(\mathbf{y}) = \begin{bmatrix} \alpha^* - \alpha \\ \beta_1^* - \beta_1 \\ \beta_2^* - \beta_2 \\ \gamma_1^* - \gamma_1 \\ \gamma_2^* - \gamma_2 \\ \gamma_3^* - \gamma_3 \end{bmatrix} \tag{5.7}$$

constitutes a verifying encoding for T_{aff} on $\mathcal{C}_{\text{aff}}$. This encoding ensures the affinely invariant coordinates of the feature points f_5, f_6, and f_7 match the desired positions for those points with respect to the first four.

In fact, the canonical representative for T_{aff} specifies more than a single task: each α, β_i, and γ_i can hold an arbitrary value in $\mathbb{R}$. Because any such instantiation of $T_{\text{aff}}(\mathbf{f}) = 0$ yields a corresponding $E_{\text{aff}}(\mathbf{y}) = 0$ which verifies it, the set of possible tasks in the above form provides a general-purpose *affine-positioning primitive.* The following two examples demonstrate analogous results for $\mathcal{C}_{\text{persp}}$ and $\mathcal{C}_{\text{proj}}$.

Similarity Positioning The *similarity-positioning primitive* T_{persp} consists of the elementary tasks of the form

$$\left\{\begin{smallmatrix} 0 & 1 & \xi & \alpha_1 & \gamma_1 \\ 0 & 0 & 0 & \alpha_2 & \gamma_2 \\ 0 & 0 & 0 & 0 & \gamma_3 \end{smallmatrix}\right\},$$

again for ξ, α_i, and $\gamma_i \in \mathbb{R}$ ($\alpha_2 \neq 0$). A reconstructive encoding verifies T_{persp}, provided the first three feature points lie in a single line and the first four in a plane. Let r_{persp} represent the reconstruction function of Equation (2.7) and $\mathbf{f} = g(r_{\text{persp}}(\mathbf{y}))$, where g is a similarity chosen so that $\mathbf{f} = \left\{ \begin{smallmatrix} 0 & 1 & \xi^* & \alpha_1^* & \gamma_1^* \\ 0 & 0 & 0 & \alpha_2^* & \gamma_2^* \\ 0 & 0 & 0 & 0 & \gamma_3^* \end{smallmatrix} \right\}$. Then

$$E_{\mathrm{sim}}(\mathbf{y}) = \begin{bmatrix} \xi^* - \xi \\ \alpha_1^* - \alpha_1 \\ \alpha_2^* - \alpha_2 \\ \gamma_1^* - \gamma_1 \\ \gamma_2^* - \gamma_2 \\ \gamma_3^* - \gamma_3 \end{bmatrix} \tag{5.8}$$

verifies T_{persp} on $\mathcal{C}_{\mathrm{persp}}$.

Projective Positioning To define the metrical positioning primitive for weakly calibrated cameras, more than one elementary task must be considered. Unlike affine or similarity-based coordinate systems, projective coordinate systems can measure the space around them in several distinct ways. Each of these projective coordinate systems necessitates an additional elementary task for fully general projective positioning.

Each of the five elementary tasks required begins with the list $\left\{ \begin{smallmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{smallmatrix} \right\}$; they conclude with one of the following:

$$\underset{\#1}{\left\{ \begin{smallmatrix} 1 & 0 & \alpha_1 & \beta_1 \\ 1 & 0 & 1 & \beta_2 \\ 0 & 1 & \alpha_3 & \beta_3 \\ 0 & 1 & 1 & 1 \end{smallmatrix} \right\}} \quad \underset{\#2}{\left\{ \begin{smallmatrix} 1 & \alpha & \xi_1 & \beta_1 & \gamma_1 \\ 1 & 1 & \xi_2 & \beta_2 & \gamma_2 \\ 0 & 1 & 1 & 1 & \gamma_3 \\ 0 & 0 & 0 & 1 & 1 \end{smallmatrix} \right\}} \quad \underset{\#3}{\left\{ \begin{smallmatrix} 1 & \alpha & \beta & \gamma_1 \\ 1 & 1 & 1 & \gamma_2 \\ 0 & 0 & 1 & \gamma_3 \\ 0 & 0 & 1 & 1 \end{smallmatrix} \right\}} \quad \underset{\#4}{\left\{ \begin{smallmatrix} 1 & \alpha_1 & \beta_1 & \gamma_1 \\ 1 & \alpha_2 & \beta_2 & \gamma_2 \\ 1 & 1 & 1 & \gamma_3 \\ 0 & 0 & 1 & 1 \end{smallmatrix} \right\}} \quad \underset{\#5}{\left\{ \begin{smallmatrix} 1 & \alpha_1 \\ 1 & \alpha_2 \\ 1 & \alpha_3 \\ 1 & 1 \end{smallmatrix} \right\}} \quad . \tag{5.9}$$

Call the set of elementary tasks in one of the above forms the *projective-positioning primitive*. It has a verifying reconstructive encoding analogous to the previous two metrical positioning primitives. Using the projective reconstruction function of Equation (2.9), the encoding function $E_{\mathrm{proj}}(\mathbf{y})$ consists of the differences of the desired and actual metrically defined coordinates. Again, the task's feature points must lie in appropriate subspaces for a meaningful comparison of coordinates.

Each of the three positioning primitives above requires that feature points lie in particular subspaces (along lines, in planes, or in general position). How can image information determine this? The point-to-point task and its encoding detect 0d coincidence; the collinearity task detects 1d coincidence. Testing 2d coincidence is the final capability needed; its failure indicates points in general position.

Coplanarity The examples of Chapter 1 introduced a coincidence (point-to-point) task and a collinearity (point-to-line) task. In a three-dimensional world there is one more *subspace-based task*: a point-to-plane constraint or *coplanarity* task. Because projective transformations preserve dimensionality, $T_{\text{copl}}(\mathbf{f}) = 0$ is in $\mathcal{T}_{\text{proj}}$. Similar to the task definitions of Section 3.1, T_{copl} has domain $\mathcal{V}^4$ and and maps features according to

$$\mathbf{f} = \{f_1, f_2, f_3, f_4\} \longmapsto \begin{cases} 0 & \text{the } f_i\text{'s are all coplanar in } \mathbb{P}^3 \\ 1 & \text{otherwise} \end{cases}.$$

Chapter 6 will demonstrate a verifying encoding for T_{copl} which relies on the weak calibration of the observing stereo rig.

Similar to the collinearity task, $T_{\text{copl}}(\mathbf{f}) = 0$ does not require that the four feature points be in general position in a plane. Three may be collinear or several may coincide. The possible combinations total twenty-three: the number of elementary tasks whose union comprise this coplanarity constraint. Thus $T_{\text{copl}}(\mathbf{f}) = 0$ for any feature among the following:

$$
\begin{array}{cccccc}
\left\{\begin{smallmatrix}1&0&0&1\\0&1&0&1\\0&0&1&1\\0&0&0&0\end{smallmatrix}\right\} &
\left\{\begin{smallmatrix}1&0&1&0\\0&1&1&0\\0&0&0&1\\0&0&0&0\end{smallmatrix}\right\} &
\left\{\begin{smallmatrix}1&0&0&1\\0&1&0&1\\0&0&1&0\\0&0&0&0\end{smallmatrix}\right\} &
\left\{\begin{smallmatrix}1&0&0&1\\0&1&0&0\\0&0&1&1\\0&0&0&0\end{smallmatrix}\right\} &
\left\{\begin{smallmatrix}1&0&0&0\\0&1&0&1\\0&0&1&1\\0&0&0&0\end{smallmatrix}\right\} &
\left\{\begin{smallmatrix}1&0&1&\alpha\\0&1&1&0\\0&0&0&0\\0&0&0&0\end{smallmatrix}\right\} \\
\left\{\begin{smallmatrix}1&0&1&1\\0&1&0&1\\0&0&0&0\\0&0&0&0\end{smallmatrix}\right\} &
\left\{\begin{smallmatrix}1&1&0&1\\0&0&1&1\\0&0&0&0\\0&0&0&0\end{smallmatrix}\right\} &
\left\{\begin{smallmatrix}1&0&0&1\\0&1&1&1\\0&0&0&0\\0&0&0&0\end{smallmatrix}\right\} &
\left\{\begin{smallmatrix}1&1&0&0\\0&0&1&0\\0&0&0&1\\0&0&0&0\end{smallmatrix}\right\} &
\left\{\begin{smallmatrix}1&1&0&0\\0&0&1&1\\0&0&0&0\\0&0&0&0\end{smallmatrix}\right\} &
\left\{\begin{smallmatrix}1&0&1&0\\0&1&0&0\\0&0&0&1\\0&0&0&0\end{smallmatrix}\right\} \\
\left\{\begin{smallmatrix}1&0&1&0\\0&1&0&1\\0&0&0&0\\0&0&0&0\end{smallmatrix}\right\} &
\left\{\begin{smallmatrix}1&0&0&1\\0&1&0&0\\0&0&1&0\\0&0&0&0\end{smallmatrix}\right\} &
\left\{\begin{smallmatrix}1&0&0&1\\0&1&1&0\\0&0&0&0\\0&0&0&0\end{smallmatrix}\right\} &
\left\{\begin{smallmatrix}1&0&0&0\\0&1&1&0\\0&0&0&1\\0&0&0&0\end{smallmatrix}\right\} &
\left\{\begin{smallmatrix}1&0&0&0\\0&1&0&1\\0&0&1&0\\0&0&0&0\end{smallmatrix}\right\} &
\left\{\begin{smallmatrix}1&0&0&0\\0&1&0&0\\0&0&1&1\\0&0&0&0\end{smallmatrix}\right\} \\
 &
\left\{\begin{smallmatrix}1&1&1&1\\0&0&0&0\\0&0&0&0\\0&0&0&0\end{smallmatrix}\right\} &
\left\{\begin{smallmatrix}1&1&1&0\\0&0&0&1\\0&0&0&0\\0&0&0&0\end{smallmatrix}\right\} &
\left\{\begin{smallmatrix}1&1&0&1\\0&0&1&0\\0&0&0&0\\0&0&0&0\end{smallmatrix}\right\} &
\left\{\begin{smallmatrix}1&0&1&1\\0&1&0&0\\0&0&0&0\\0&0&0&0\end{smallmatrix}\right\} &
\left\{\begin{smallmatrix}1&0&0&0\\0&1&1&1\\0&0&0&0\\0&0&0&0\end{smallmatrix}\right\}
\end{array}
$$

Every four-point list in $\mathcal{L}_{\text{proj}}$ except $\left\{\begin{smallmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\\0&0&0&1\end{smallmatrix}\right\}$ (four points in general position) appears in the above set. This multitude of canonical representatives exhibits the awkwardness of specifying general tasks through elementary tasks and the single mechanism for composing them — union, indicating a disjunction of task constraints. The following chapter explores more succinct means for expressing tasks: languages with fewer primitives and additional techniques for combining them.

Chapter 6

Languages for Task Specification

Our objective in this chapter is to provide a more intuitive set of building blocks for specifying hand/eye tasks than the canonical forms of Chapter 5. We seek a language more typical of modular specification systems: a set of primitive skills along with a number of operators for combining them. Proofs of completeness for the resulting languages build on the algebraic formulation of elementary tasks.

6.1 Elementary Tasks as a Specification Language?

The four characterizations of Propositions 2 - 5 provide languages for unambiguously specifying any decidable task in $\mathcal{T}_{\mathrm{inj}}$, $\mathcal{T}_{\mathrm{proj}}$, $\mathcal{T}_{\mathrm{aff}}$, and $\mathcal{T}_{\mathrm{persp}}$. The examples of nonelementary tasks, T_{col} and T_{copl}, depend on creating a union or disjunction of several elementary tasks' constraints. Thus far, that disjunction has ocurred informally by appealing to intuition; the following definition of the *disjunction operator* formalizes that intuitive treatment.

Definition 2 *For two task functions T_1 and T_2 with domain $\mathcal{V}^n$, define the disjunc-*

tion of T_1 *and* T_2 *as the task* $(T_1 \vee T_2)(\mathbf{f}) = 0$, *where*

$$(T_1 \vee T_2)(\mathbf{f}) = T_1(\mathbf{f})T_2(\mathbf{f}).$$

Definition 2 meets our intuitive requirements for disjunction as a "union" of constraints: if a feature satisfies either $T_1(\mathbf{f}) = 0$ or $T_2(\mathbf{f}) = 0$, it must satisfy $(T_1 \vee T_2)(\mathbf{f}) = 0$.

Specification Languages

How do the sets $\mathcal{L}_{\mathrm{i}}$ (sets of elementary tasks) and the disjunction operator form a language for specifying tasks? Mathematically, a *task-specification language* is a pair $\langle \mathcal{P}, \mathcal{O} \rangle$ consisting of a set of basis tasks ("primitives") $\mathcal{P}$ and a set of task operators $\mathcal{O}$. The language $\langle \mathcal{P}, \mathcal{O} \rangle$ *specifies* a task T, if T is in the closure of $\mathcal{P}$ under the operations in $\mathcal{O}$. The symbol $\mathcal{T}(\langle \mathcal{P}, \mathcal{O} \rangle)$ denotes the set of all tasks specified by the language $\langle \mathcal{P}, \mathcal{O} \rangle$. According to this formalism, elementary tasks in $\mathcal{L}_{\mathrm{i}}$ form a basis set and disjunction constitutes an operator set $\mathcal{O} = \{\vee\}$, so that $\langle\, \mathcal{L}_{\mathrm{i}}, \{\vee\}\,\rangle$ are the formal task-specification languages generated by the elementary tasks of Chapter 5.

With this background we can investigate properties of the tasks in $\mathcal{T}(\langle \mathcal{L}_{\mathrm{i}}, \mathcal{O} \rangle)$ — or any language's tasks. Of particular interest is the relationship of a language's tasks with the decidable tasks in $\mathcal{T}_{\mathrm{i}}$. To this end we characterize the two possible inclusion relationships between the tasks a language specifies and another task set.

Definition 3 *Given a task-specification language* $\mathcal{L} = \langle \mathcal{P}, \mathcal{O} \rangle$ *and a set of tasks* $\mathcal{T}$, *the language* $\mathcal{L}$ *is considered* concise *with respect to* $\mathcal{T}$ *if*

$$\mathcal{T}(\mathcal{L}) \subset \mathcal{T} \ .$$

In short, concise languages do not specify more than necessary with respect to some externally defined set. *Completeness* describes languages which can specify at least the tasks contained in a reference set:

Definition 4 *Given a task-specification language $\mathcal{L} = \langle \mathcal{P}, \mathcal{O} \rangle$ and a set of tasks $\mathcal{T}$, the language $\mathcal{L}$ is considered* complete *with respect to $\mathcal{T}$ if*

$$\mathcal{T} \subset \mathcal{T}(\mathcal{L}) \ .$$

By these definitions, languages that are both concise and complete with respect to some set enable the specification of any task in that set but no tasks outside of it. This capability arises naturally with respect to decidable tasks, as they constitute all of the tasks for which a particular family of cameras could possibly guarantee performance. The fact (Theorem 1) that every decidable task in $\mathcal{T}_{\mathrm{i}}$ must be constant on each $\simeq_{\mathrm{i}}$-equivalence class implies that the specification languages $\langle\, \mathcal{L}_{\mathrm{i}}, \{\vee\}\,\rangle$ are complete with respect to each $\mathcal{T}_{\mathrm{i}}$. On the other hand, since every task invariant under $\simeq_{\mathrm{i}}$-equivalence is decidable on $\mathcal{C}_{\mathrm{i}}$, those languages are also concise with respect to the appropriate sets of decidable tasks. This argument proves the following proposition.

Proposition 6 *Let* $\mathrm{i} \in \{\mathrm{inj}, \mathrm{proj}, \mathrm{aff}, \mathrm{persp}\}$. *The elementary tasks $\mathcal{L}_{\mathrm{i}}$ and the disjunction operator generate concise and complete languages $\langle\, \mathcal{L}_{\mathrm{i}}, \{\vee\}\,\rangle$ for specifying the decidable tasks in $\mathcal{T}_{\mathrm{i}}$.*

As the coplanarity example of Section 5.7 and the collinearity exmple of Section 5.2 suggest, the languages of Proposition 6 can be unwieldy in practice. In particular, they have large sets of primitives, $\mathcal{L}_{\mathrm{i}}$, and employ only a single method for combining them. Hence a natural strategy for creating simpler languages involves balancing the number and complexity of task operators with the number of task primitives needed to span each $\mathcal{T}_i$. The rest of this chapter exploits this strategy to develop specification languages for decidable tasks. The next section starts by introducing several additional operators for composing tasks.

6.2 Operators for Composing Tasks

To eliminate the need for such a large basis set of task primitives, we expand our repertoire of task operators to six, including disjunction, arranged in Figure 6.1. The symbol $\mathcal{O}_6$ will denote this set of six operators. A natural first question to ask about any task operator is whether or not it preserves decidability. The most direct approach of showing this entails defining composite encodings and demonstrating that as long as component encodings verify component tasks, the composite encodings verify their composite tasks. Figure 6.1 shows the action of the same six operators on task encodings, and Proposition 7 demonstrates that these operators do preserve decidability.

Proposition 7 *For any tasks T and $\hat{T}$, verified by E and $\hat{E}$ respecitvely, θE_T verifies θT for each unary θ in Figure 6.1 and $E\theta\hat{E}$ verifies $T\theta\hat{T}$ for each binary operator in Figure 6.1.*

Proof of Proposition 7: Fix two task functions T and $\hat{T}$ which are decidable on some set of camera models $\mathcal{C}$. Let verifying encoding functions be E and $\hat{E}$. Also, choose an element $C \in \mathcal{C}$ and a feature $\mathbf{f} \in \mathcal{V}^n$ with $\mathbf{y} = C(\mathbf{f})$. Then, since $T(\mathbf{f}) = 0 \Leftrightarrow E(\mathbf{y}) = 0$, we know that $1 - T(\mathbf{f}) = \neg T(\mathbf{f}) = 0 \Leftrightarrow \neg E(\mathbf{y}) = 0$. Hence complement preserves verifiability. Because C operates componentwise on $\mathbf{f}$, $\pi\mathbf{y} = C(\pi\mathbf{f})$ for any permutation π. Thus, permutation preserves verifiability.

Conjunction preserves verifiability because $\begin{bmatrix} E(\mathbf{y}) \\ \hat{E}(\mathbf{y}) \end{bmatrix}$ is the zero vector if and only if its components, $E(\mathbf{y})$ and $\hat{E}(\mathbf{y})$ are both 0. Similarly, because $(E \vee \hat{E})(\mathbf{y}) =$

Task Operators in $\mathcal{O}_6$

Complement	$\neg T(\mathbf{f}) = 1 - T(\mathbf{f})$	$\neg E(\mathbf{y}) = \begin{cases} 0 & \text{if } E(\mathbf{y}) \neq 0 \\ 1 & \text{if } E(\mathbf{y}) = 0 \end{cases}$
Permutation	$\pi T(\mathbf{f}) = T(\pi \mathbf{f})$	$\pi E(\mathbf{y}) = E(\pi \mathbf{y})$ with $\pi \in \mathcal{P}_n$
Disjunction	$(T \vee \hat{T})(\mathbf{f}) = T(\mathbf{f})\hat{T}(\mathbf{f})$	$(E \vee \hat{E})(\mathbf{y}) = E(\mathbf{y})\hat{E}(\mathbf{y})$
Conjunction	$(T \wedge \hat{T})(\mathbf{f}) = 1 - (1 - T(\mathbf{f}))(1 - \hat{T}(\mathbf{f}))$	$(E \wedge \hat{E})(\mathbf{y}) = \left[\begin{smallmatrix} E(\mathbf{y}) \\ \hat{E}(\mathbf{y}) \end{smallmatrix} \right]$ or $E^2(\mathbf{y}) + \hat{E}^2(\mathbf{y})$
Expansion	$\varepsilon T(f_1 \dots f_m) = T(f_1 \dots f_n)$	$\varepsilon E(y_1 \dots y_m) = E(y_1 \dots y_n)$ with $n < m$
Contraction	$cT(f_1 \dots f_l) = T(f_1 \dots f_n)$ $l < n$ and cT well-defined	$cE(y_1 \dots y_l) = E(y_1 \dots y_n)$ $l < n$ and cE well-defined

Figure 6.1: Six task operators constituting the set $\mathcal{O}_6$. Each operates also on task encodings. Constraints on the operator definitions appear below them. $\mathcal{P}_n$ represents the group of permutations of n objects.

$E(\mathbf{y})\hat{E}(\mathbf{y}) = 0$ is equivalent to either having $E(\mathbf{y}) = 0$ or $\hat{E}(\mathbf{y}) = 0$, disjunction preserves verifiability. Expansion and contraction follow by identical reasoning. Note that contraction is only defined when T is single-valued on equivalence classes in $\mathcal{V}^n$ under the following equivalence relation:

$$\{f_1 \dots f_l, f_{l+1} \dots f_n\} \equiv \{g_1 \dots g_l, g_{l+1} \dots g_n\} \Leftrightarrow \forall i \in \{1 \dots l\}\ f_i = g_i.$$

Essentially, T must "ignore" the last $n - l$ feature points in all of its possible inputs, In this case, however, contraction does preserve verifiability. ■

Because explicit image-based encoding functions are provided, the proof of Proposition 7 is a constructive proof of the following:

Corollary 1 *The six operators of Figure 6.1 preserve task decidability on any set of two-camera models.*

In addition, the continuity and differentiability of encoding functions — necessary for robot control — are maintained by five of the six operators in $\mathcal{O}_6$. Continuity and differentiability remain under conjunction and disjunction because multiplication preserves those properties and vector-valued functions are continuous and differentiable when their component functions are. Permutation, expansion, and contraction do not change encodings at all, so they also preserve continuity and differentiability. Thus, the following proposition obtains.

Proposition 8 *All of the encoding operators of Figure 6.1, except for the complement operator, preserve the continuity and differentiability of task encoding functions.*

■

What happens with the negation operator? In fact, there is no general definition for the encoded task function $\neg E(\mathbf{y})$ which preserves continuity while maintaining the fact that its kernel is the set of goal configurations. To see this, consider possible encodings of $\neg T_{\mathrm{pp}}$. Correctness imposes that $\neg E_{\mathrm{pp}}(\mathbf{y}) = 0$ for all $\mathbf{y}$ in the domain except image features with coincident feature points. Because we are assuming a compact viewspace, those features with coincident feature points are limit points of the kernel of $\neg E_{\mathrm{pp}}$. To preserve continuity, $\neg E_{\mathrm{pp}}$ would have to be identically zero on its domain, in which case it would cease to verify $\neg T_{\mathrm{pp}}$. Thus, no continuous encoding verifies $\neg T_{\mathrm{pp}}$. Fortunately, this is not a serious problem in practice: Section 8.3 presents approximations which can achieve negated task-space constraints.

We call the compositions (namely θT or $T\theta\hat{T}$) of Figure 6.1's operators *composite* task functions. Proposition 7 allows us to generate composite tasks with the confidence that composing them will not somehow "break" the imaging system's ability to distinguish success from failure. The next section presents several detailed examples of composite tasks.

6.3 Examples of Composite Tasks

As an example of a composite task, consider building a "pure" collinearity constraint similar to T_{col}. Section 3.1 noted that several configurations involving point-coincidence satisfy T_{col} and Section 5.2 enumerated exactly the elementary tasks for which $T_{\mathrm{col}}(\mathbf{f}) = 0$. Let $T_{\mathrm{3col}}(\mathbf{f}) = 0$ represent the task satisfied when three feature points are collinear and no two are coincident. The corresponding elementary task is $\left\{\begin{smallmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{smallmatrix}\right\}$. Using the operators of $\mathcal{O}_6$, the following composition of T_{col} and T_{pp} expresses T_{3col}:

$$T_{\mathrm{3col}}(\mathbf{f}) \quad = \quad T_{\mathrm{col}}(\mathbf{f}) \;\wedge\; \pi_1\varepsilon\neg T_{\mathrm{pp}}(\mathbf{f}) \;\wedge\; \pi_2\varepsilon\neg T_{\mathrm{pp}}(\mathbf{f}) \;\wedge\; \pi_3\varepsilon\neg T_{\mathrm{pp}}(\mathbf{f}) \tag{6.1}$$

Here, the extensions denoted by ε increase the number of domain feature points by one: T_{3col} takes features with three feature points while T_{pp}'s inputs have only two. The permutations $\pi_1, \pi_2 \in \mathcal{P}_3$ are $(1,2,3)$, $(2,3,1)$, and $(3,1,2)$. Combined with the negation operator, the last three terms effectively prohibit point-coincidence between each possible pair of feature points. Because each term's argument $(\mathbf{f})$ does not contribute to the structural description of composite tasks such as (6.1), it will be suppressed in subsequent compositions.

Note that this composite task assumes nothing about the roles of the features in the task. If the manipulator controlled only f_2 and left f_1 and f_3 distinct and stationary, the final term in the composition becomes unnecessary, since it merely guarantees that f_1 and f_3 do not coincide. Chapter 7 investigates the extent to which these kinds of assumptions about the scene can simplify task specifications.

At first glance the representation of Equation (6.1) might not seem a clear improvement over the elementary-task representation of T_{col} in Section 5.2. The real gain from the above approach is that we need only two (rather than five) primitive encodings. In addition, the composition which specifies the task $T_{3\mathrm{col}}$ automatically generates an encoding for it:

$$E_{3\mathrm{col}} \quad = \quad E_{\mathrm{col}} \ \wedge \ \pi_1\varepsilon(\neg E_{\mathrm{pp}}) \ \wedge \ \pi_2\varepsilon(\neg E_{\mathrm{pp}}) \ \wedge \ \pi_3\varepsilon(\neg E_{\mathrm{pp}}) \tag{6.2}$$

As long as the component encodings verify the component tasks, the composite encoding will verify the resulting composite task.

Analogous to $T_{3\mathrm{col}}$, we can build $T_{4\mathrm{copl}}$, satisfied only when four points are coplanar but no three are collinear. The appropriate composition is

$$T_{4\mathrm{copl}} \quad = \quad T_{\mathrm{copl}} \ \wedge \ \bigwedge \pi_i\varepsilon\neg T_{\mathrm{col}} \ , \tag{6.3}$$

with each T_{col} taking three of the four feature points as input. Again, E_{copl} and E_{col} (as defined in Section 6.7) create a verifying encoding for $T_{4\mathrm{copl}}$.

Extending one more dimension, define T_{gp} as a general positioning task, satisfied only when all its n ($n \geq 4$) feature points fall in general position, i.e., no four are coplanar. Similar to the latter term of $T_{4\mathrm{copl}}$, T_{gp} requires a conjunction of many negated primitive tasks:

$$T_{\mathrm{gp}} \quad = \quad \bigwedge \pi_i\varepsilon\neg T_{\mathrm{copl}} \ . \tag{6.4}$$

The conjunction indexes over all of the four-feature-point subsets of $\mathbf{f}$.

$T_{3\mathrm{col}}$, $T_{4\mathrm{copl}}$, and T_{gp} demonstrate expansion, conjunction, permutation, and negation operations; T_{col} and T_{copl} use the disjunction operator (and several elementary tasks). To demonstrate the lone remaining task operator, contraction, consider again the midpoint task T_{midpt} of Figure 1.2. The encoding E_{midpt} presented in Section 3.3 did not verify T_{midpt} on $\mathcal{C}_{\mathrm{aff}}$, although $T_{\mathrm{midpt}} \in \mathcal{T}_{\mathrm{aff}}$. Instead of developing an encoding from scratch (as with E_{midpt}), we can recognize T_{midpt} as a special case of the general affine positioning primitive, T_{aff}. The resulting definition of T_{midpt} employs the contraction operator:

$$T_{\mathrm{midpt}} \quad = \quad c(\pi T_{\mathrm{aff}}) \quad \wedge T_{\mathrm{col}} \ . \tag{6.5}$$

Here, $\pi = (1, 2, 4, 5, 3, 6, 7) \in \mathcal{P}_7$ rearranges f_1, f_2, and f_5 to occupy the first three feature-point positions in T_{aff}, and the contraction operator c ignores all but these three. Because of the contraction T_{aff} need not ensure the subspace configuration of any feature points except that the first three are collinear. The T_{col} term expresses that constraint. One parameter of T_{aff} must be specified $\alpha = \frac{1}{2}$; the remaining parameters must remain arbitrary in order to ensure that $c(\pi T_{\mathrm{aff}})$ is well-defined.

As in the previous case, the encoding for T_{midpt} mirrors its task specification:

$$E_{\mathrm{midpt}} \quad = \quad c(\pi E_{\mathrm{aff}}) \quad \wedge E_{\mathrm{col}} \ . \tag{6.6}$$

The contraction merely removes the ignored reconstructive parameters from the full version of E_{aff}. Unlike the encoding of Section 3.3, the E_{midpt} of Equation (6.6) does verify T_{midpt} on $\mathcal{C}_{\mathrm{aff}}$.

The following sections demonstrate concise and complete languages for specifying decidable tasks with far fewer primitves than the elementary-task languages of Section 6.1. Chapter 7 demonstrates how to remove much of the apparent complexity of specifications like (6.1). In particular, assigning consistent labels for feature points allows the hiding of expansion, contraction, and permutation operations from a programmer or user.

6.4 A Compositional Task Language for $\mathcal{T}_{\text{inj}}$

Because injective camera models preserve the distinctness of feature points, but guarantee no other structure, the point-to-point primitive T_{pp} seems the only building block $\mathcal{T}_{\text{inj}}$ would need. The following proposition proves that the point-to-point constraint and the operators of $\mathcal{O}_6$ do suffice to generate $\mathcal{T}_{\text{inj}}$. This result justifies the widespread use of compositions of point-coincidence primitives to specify visual servoing tasks for almost any uncalibrated visual system [9, 30, 46, 49].

Proposition 9 *Let $\mathcal{P} = \{T_{\text{pp}}\}$ and $\mathcal{L} = \langle \mathcal{P}, \mathcal{O}_6 \rangle$. Then $\mathcal{L}$ is a concise and complete task-specification language for $\mathcal{T}_{\text{inj}}$.*

Proof of Proposition 9: Proposition 7 shows that every task specifiable by $\mathcal{L}$ is decidable on $\mathcal{C}_{\text{inj}}$, since T_{pp} is. Thus, $\mathcal{L}$ is concise with respect to $\mathcal{T}_{\text{inj}}$. For completeness, because $\mathcal{O}_6$ contains the disjunction operator, it suffices to demonstrate that every elementary task in $\mathcal{T}_{\text{inj}}$ is also in $\mathcal{L}$. We proceed by induction on n, the number of feature points in $\mathbf{f}$, an arbitrary canonical representative from $\mathcal{L}_{\text{inj}}$.

For the base case $\mathcal{L}$ contains the always-satisfied and never-satisfied tasks

$$T_{\text{true}} = c(T_{\text{pp}} \vee \neg T_{\text{pp}}) \qquad T_{\text{false}} = c(T_{\text{pp}} \wedge \neg T_{\text{pp}}) \ ,$$

alternatively defined by $T_{\text{true}} \equiv 0$ and $T_{\text{false}} \equiv 1$. These are the only interpretations possible for the elementary tasks with $n = 0$ and $n = 1$, namely $\mathbf{f} = \left\{ \ \right\}$ and $\mathbf{f} = \left\{ \begin{smallmatrix} 0 \\ 0 \\ 0 \end{smallmatrix} \right\}$. The contraction operator above allows the reduction of the domain in either case. Assume, then, that $n > 1$ and that Proposition 9 holds for positive integers less than n. Let $\mathbf{f} \in \mathcal{L}_{\mathrm{inj}}$ have n feature points and $\mathbf{f}^-$ consist of $\mathbf{f}$'s initial $n-1$ feature points. Since the inductive hypothesis holds for $\mathbf{f}^-$, $\mathbf{f}^- \in \mathcal{L}$. As a result, some composition of the tasks in $\mathcal{P}$ and the operators in $\mathcal{O}_6$ yield a task satisfied on no canonical representative other than $\mathbf{f}^-$. Call this composite task $T_{\mathrm{n}-1}$.

Either Rule 1 or Rule 2 added f_n to $\mathbf{f}^-$. Suppose the latter. Then $f_n = f_i$ for some $f_i \in \mathbf{f}^-$. Let

$$T_{\mathrm{n}} \quad = \quad \varepsilon_1 T_{\mathrm{n}-1} \;\; \wedge \;\; \pi\varepsilon_2 T_{\mathrm{pp}} \;\; , \tag{6.7}$$

where ε_2 adds $n-2$ feature points to T_{pp}'s domain and $\pi \in \mathcal{P}_n$ is a permutation such that $\pi(1) = i$ and $\pi(2) = n$. The expansion ε_1 adds a single point to $T_{\mathrm{n}-1}$'s domain. We claim that the task with task function T_{n} is the task represented by $\mathbf{f}$. To show this, it suffices to demonstrate that $T_{\mathrm{n}}(\mathbf{f}) = 0$ and that $T_{\mathrm{n}}(\mathbf{g}) \neq 0$ for any canonical $\mathbf{g} \neq \mathbf{f}$. $T_{\mathrm{n}}(\mathbf{f}) = 0$ by construction: the permutation π merely ensures that f_i equal f_n, as $\mathbf{f}$ specifies. If $\mathbf{g} \neq \mathbf{f}$ and $\mathbf{g} \in \mathcal{L}_{\mathrm{inj}}$, either $g_n \neq g_i$ or $T_{\mathrm{n}-1}(\mathbf{g}^-) \neq 0$, so that $T_{\mathrm{n}}(\mathbf{g}) \neq 0$. This handles the case in which Rule 2 added f_n to $\mathbf{f}^-$.

Suppose Rule 1 added f_n. Then $f_n \neq f_i$ for $1 \leq i < n$. In this case let

$$T_{\mathrm{n}} \quad = \quad \varepsilon_1 T_{\mathrm{n}-1} \;\; \wedge \;\; \bigwedge_{i=0}^{n-1} \pi_i \varepsilon_2 \neg T_{\mathrm{pp}} \;\; . \tag{6.8}$$

Here, for each $i \in \{1 \ldots n-1\}$ the permutation $\pi_i \in \mathcal{P}_n$ satisfies $\pi_i(1) = i$ and $\pi_i(2) = n$. By construction, this composition is satisfied on $\mathbf{f}$. Suppose $\mathbf{g} \neq \mathbf{f}$ and $\mathbf{g} \in \mathcal{L}_{\mathrm{inj}}$. Then either $T_{\mathrm{n}-1}(\mathbf{g}^-) \neq 0$ or $g_n = g_i \in \mathbf{f}^-$, which will cause one of the conjuncts in T_{n} to fail. Either way, $T_{\mathrm{n}}(\mathbf{g}) \neq 0$

Since all possibilities for $\mathbf{f}$ have been considered, Proposition 9 holds. ■

6.5 A Compositional Task Language for $\mathcal{T}_{\text{proj}}$

The tasks in $\mathcal{T}_{\text{proj}}$, $\mathcal{T}_{\text{aff}}$, and $\mathcal{T}_{\text{persp}}$ provide the capacity both for coincidence-based positioning (as in $\mathcal{T}_{\text{inj}}$) and metrical positioning, as demonstrated in Section 5.7. Since transformations in $\mathcal{G}_{\text{proj}}$ preserve 1d and 2d subspace structure, as well as 0d structure (coincidence), the set of basis tasks for $\mathcal{T}_{\text{proj}}$ will include T_{col} and T_{copl} , along with T_{pp}. To handle metrical tasks the general-purpose projective positioning primitive T_{proj} suffices. The following proposition summarizes and proves these claims.

Proposition 10 *Let* $\mathcal{P} = \{T_{\text{pp}}, T_{\text{col}}, T_{\text{copl}}, T_{\text{proj}}\}$ *and* $\mathcal{L} = \langle \mathcal{P}, \mathcal{O}_6 \rangle$. *Then* $\mathcal{L}$ *is a concise and complete task-specification language for* $\mathcal{T}_{\text{proj}}$.

Proof of Proposition 10: We begin by characterizing additional structure of the elementary tasks of $\mathcal{T}_{\text{proj}}$. Consider an arbitrary feature $\mathbf{f}$ in canonical form. We examine the possible number of feature points of $\mathbf{f}$ which each of the four rules of Proposition 3 might have added. By definition only the first feature point (e_1) comes from Rule 0. Because $1 \leq \dim(\mathbf{f}) \leq 4$ and feature points added through Rule 1 necessarily increase $\dim(\mathbf{f})$, at most three feature points (e_2, e_3, e_4) within $\mathbf{f}$ could come from Rule 1. Similarly, the number of sets in the row-partition of a feature is between 1 and 4 (inclusive) and feature points added through Rule 3 decrease that number. Hence Rule 3 can contribute no more than three feature points.

As a result of these observations, we can imagine the following permutation of the feature points of $\mathbf{f}$. First, in order of increasing i, list once each appearing "basis" vector e_i. Second, list each "measuring" point added by Rule 3 in order of appearance in $\mathbf{f}$. Finally, list the balance of the feature points (considered "positioned") in any order. The resulting permutation of $\mathbf{f}$ is a feature in canonical form, because the points which determine the projective structure of $\mathbf{f}$ (those added by Rules 1 and 3) appear before those which depend on that structure. We summarize this result in the following lemma.

Lemma 9 *Let* $\mathbf{f}$ *be any feature in* $\mathcal{L}_{\text{proj}}$*. Then there exists a permutation of* $\mathbf{f}$*,* $\pi\mathbf{f}$*, such that* $\pi\mathbf{f} = \{\mathbf{f}_B; \mathbf{f}_M; \mathbf{f}_P\}$ *with* $\mathbf{f}_B$ *consisting of the 1 to 4 feature points added by Rules 0 and 1,* $\mathbf{f}_M$ *consisting of the 0 to 3 feature points added by Rule 3, and* $\mathbf{f}_P$ *consisting of the feature points added by Rule 2.* ■

Call $\pi\mathbf{f}$ a *canonical permutation* of $\mathbf{f}$. The basis points $\mathbf{f}_B$ of a canonical permutation have a simple structure: $\mathbf{f}_B = \{e_1 \ \ldots \ e_{\dim(\mathbf{f})}\}$. The positioned points $\mathbf{f}_P$ are any normalized vectors which have nonzero components in only one set of the row-partition of $\mathbf{f}$. The structure of $\mathbf{f}_M$ is more involved. Yet since $\mathbf{f}_M$ consists of at most three feature points, there are only a small number of distinct structures $\mathbf{f}_M$ can have. We now set out to list these possibilities.

To this end a useful characterization of a feature $\mathbf{f}$ is the *signature* of $\mathbf{f}$'s row-partition, which we define to be the decreasing list of the cardinalities of the individual sets $\{\mathcal{P}_1 \ldots \mathcal{P}_N\}$ of that row-partition. As an example, the row-partitions $\{\{1\},\{2,3\},\{4\}\}$ and $\{\{1,4\},\{3\},\{2\}\}$ have the same signature, namely, $\{2,1,1\}$. In total only five different signatures are possible: $\{1,1,1,1\}$, $\{2,1,1\}$, $\{2,2\}$, $\{3,1\}$, and $\{4\}$.

Recall that $\mathbf{f}^i$ represents the initial sublist of i feature points of $\mathbf{f}$ and that $\mathbf{f}^i$ is also a feature in canonical form. Suppose that $\mathbf{f}^i$ and $\mathbf{f}^{i+1}$ have different row-partitions. Because only feature points added through Rule 3 change the row-partition of a feature list, f_{i+1} must have come through Rule 3. On the other hand, if f_{i+1} is added through Rule 3, it reduces the number of sets in the feature's row-partition, so $\mathbf{f}^i$ and $\mathbf{f}^{i+1}$ have distinct signatures. Thus, by Lemma 9 there is a transition sequence of at most four distinct signatures for the $\mathbf{f}^i$s, $1 \leq i \leq n$. The following lemma enumerates all possible signature-sequences for a canonical feature $\mathbf{f}$ and provides a composite task equivalent to each. The notation $\#i$ indicates the ith elementary task in the projective positioning primitive (5.9).

Lemma 10 *There are ten distinct signature sequences possible within the initial sublists $\mathbf{f}^i$ $(1 \leq i \leq n)$ of a canonical feature $\mathbf{f}$ with n feature points. Each is listed here, along with a minimal elementary task which exhibits the same signature-sequence. A composite task satisfied only by that elementary task's equivalence class follows. Note that in the possibilities below, each variable $\alpha, \beta, \ldots$ is assumed nonzero. Additional disjunctions of subspace-coincidence tasks would handle any zero-valued variables.*

$$\{1,1,1,1\} \rightarrow \{2,1,1\} \rightarrow \{2,2\} \rightarrow \{4\} \qquad \left\{\begin{smallmatrix} 1&0&0&0&1&0&\alpha_1&\beta_1 \\ 0&1&0&0&1&0&1&\beta_2 \\ 0&0&1&0&0&1&\alpha_3&\beta_3 \\ 0&0&0&1&0&1&1&1 \end{smallmatrix}\right\}$$

$$\pi_1\varepsilon_1\neg T_{\text{copl}}(f_1,f_2,f_3,f_4) \ \wedge\ \pi_2\varepsilon_2 T_{\text{3col}}(f_1,f_2,f_5) \ \wedge\ \pi_3\varepsilon_3 T_{\text{3col}}(f_3,f_4,f_6)$$
$$\pi_4\varepsilon_4 T_{\text{gp}}(f_1,f_2,f_3,f_4,f_7) \ \wedge\ T^{\#1}_{\text{proj}}(\mathbf{f})$$

$$\{1,1,1,1\} \rightarrow \{2,1,1\} \rightarrow \{3,1\} \rightarrow \{4\} \qquad \left\{\begin{smallmatrix} 1&0&0&0&1&\alpha&\beta_1&\gamma_1 \\ 0&1&0&0&1&1&\beta_2&\gamma_2 \\ 0&0&1&0&0&1&1&\gamma_3 \\ 0&0&0&1&0&0&1&1 \end{smallmatrix}\right\}$$

$$\pi_1\varepsilon_1\neg T_{\text{copl}}(f_1,f_2,f_3,f_4) \ \wedge\ \pi_2\varepsilon_2 T_{\text{3col}}(f_1,f_2,f_5) \ \wedge\ \pi_3\varepsilon_3 T_{\text{4copl}}(f_1,f_2,f_3,f_6)$$
$$\pi_4\varepsilon_4 T_{\text{gp}}(f_1,f_2,f_3,f_4,f_7) \ \wedge\ T^{\#2}_{\text{proj}}(\mathbf{f})$$

$$\{1,1,1,1\} \rightarrow \{2,1,1\} \rightarrow \{4\} \qquad \left\{\begin{smallmatrix} 1&0&0&0&1&\alpha&\beta_1 \\ 0&1&0&0&1&1&\beta_2 \\ 0&0&1&0&0&1&\beta_3 \\ 0&0&0&1&0&1&1 \end{smallmatrix}\right\}$$

$$\pi_1\varepsilon_1\neg T_{\text{copl}}(f_1,f_2,f_3,f_4) \ \wedge\ \pi_2\varepsilon_2 T_{\text{3col}}(f_1,f_2,f_5) \ \wedge\ \pi_3\varepsilon_3 T_{\text{gp}}(f_1,f_2,f_3,f_4,f_6) \ \wedge\ cT^{\#3}_{\text{proj}}(\mathbf{f})$$

$$\{1,1,1,1\} \rightarrow \{3,1\} \rightarrow \{4\} \qquad \left\{\begin{smallmatrix} 1&0&0&0&1&\alpha_1&\beta_1 \\ 0&1&0&0&1&\alpha_2&\beta_2 \\ 0&0&1&0&1&1&\beta_3 \\ 0&0&0&1&0&1&1 \end{smallmatrix}\right\}$$

$$\pi_1\varepsilon_1\neg T_{\text{copl}}(f_1,f_2,f_3,f_4) \ \wedge\ \pi_2\varepsilon_2 T_{\text{4copl}}(f_1,f_2,f_3,f_5) \ \wedge\ \pi_3\varepsilon_3 T_{\text{gp}}(f_1,f_2,f_3,f_4,f_6) \ \wedge\ cT^{\#4}_{\text{proj}}(\mathbf{f})$$

$\{1,1,1,1\} \to \{4\}$ $\left\{\begin{smallmatrix} 1&0&0&0&1&\alpha \\ 0&1&0&0&1&\beta \\ 0&0&1&0&1&\gamma \\ 0&0&0&1&1&1 \end{smallmatrix}\right\}$

$\pi_1\varepsilon_1\neg T_{\mathrm{copl}}(f_1,f_2,f_3,f_4) \;\wedge\; \pi_2\varepsilon_2 T_{\mathrm{gp}}(f_1,f_2,f_3,f_4,f_5) \;\wedge\; T^{\#5}_{\mathrm{proj}}(\mathbf{f})$

$\{1,1,1,1\} \to \{2,1,1\} \to \{2,2\}$ $\left\{\begin{smallmatrix} 1&0&0&0&1&0 \\ 0&1&0&0&1&0 \\ 0&0&1&0&0&1 \\ 0&0&0&1&0&1 \end{smallmatrix}\right\}$

a contraction of the first example above

$\{1,1,1,1\} \to \{2,1,1\} \to \{3,1\}$ $\left\{\begin{smallmatrix} 1&0&0&1&\alpha&\xi_1 \\ 0&1&0&1&1&\xi_2 \\ 0&0&1&0&1&1 \\ 0&0&0&0&0&0 \end{smallmatrix}\right\}$

$$\pi_1\varepsilon_1\neg T_{\mathrm{col}}(f_1,f_2,f_3) \;\wedge\; \pi_2\varepsilon_2 T_{\mathrm{3col}}(f_1,f_2,f_4) \;\wedge\; \pi_3\varepsilon_3 T_{\mathrm{4copl}}(f_1,f_2,f_3,f_5)$$
$$\wedge\; \pi_4\varepsilon_4 T_{\mathrm{4copl}}(f_1,f_2,f_3,f_6) \;\wedge\; cT^{\#2}_{\mathrm{proj}}(\mathbf{f})$$

$\{1,1,1,1\} \to \{3,1\}$ $\left\{\begin{smallmatrix} 1&0&0&1&\alpha_1 \\ 0&1&0&1&\alpha_2 \\ 0&0&1&1&1 \\ 0&0&0&0&0 \end{smallmatrix}\right\}$

$\pi_1\varepsilon_1\neg T_{\mathrm{col}}(f_1,f_2,f_3) \;\wedge\; \pi_2\varepsilon_2 T_{\mathrm{4copl}}(f_1,f_2,f_3,f_4) \;\wedge\; \pi_3\varepsilon_3 T_{\mathrm{4copl}}(f_1,f_2,f_3,f_5) \;\wedge\; cT^{\#4}_{\mathrm{proj}}(\mathbf{f})$

$\{1,1,1,1\} \to \{2,1,1\}$ $\left\{\begin{smallmatrix} 1&0&1&\alpha \\ 0&1&1&1 \\ 0&0&0&0 \\ 0&0&0&0 \end{smallmatrix}\right\}$

$\pi_1\varepsilon_1 T_{\mathrm{3col}}(f_1,f_2,f_3) \;\wedge\; \pi_2\varepsilon_2 T_{\mathrm{3col}}(f_1,f_2,f_4) \;\wedge\; cT^{\#3}_{\mathrm{proj}}(\mathbf{f})$

$\{1,1,1,1\}$ $\left\{\begin{smallmatrix} 1&0&0&0 \\ 0&1&0&0 \\ 0&0&1&0 \\ 0&0&0&1 \end{smallmatrix}\right\}$

$T_{\mathrm{gp}}(\mathbf{f})$

Proof of Lemma 10: Regardless of $\mathbf{f}$, the signature of $\mathbf{f}^1$ is $\{1,1,1,1\}$. Lemma 9 shows that at most three feature points can be added by Rule 3 in building a feature.

Thus, no more than three transitions in the sequence of distinct signatures is possible. In addition, the transition $\{1,1,1,1\} \rightarrow \{2,2\}$ is impossible, as a single feature point cannot simultaneously combine two distinct pairs of partition sets without creating a single row-partition set, i.e., $\{1,1,1,1\} \rightarrow \{4\}$. These constraints give rise to the possible transitions in the "signature state machine" depicted by Figure 6.2.

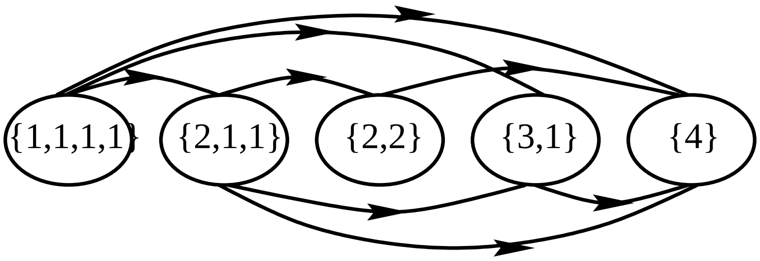


Figure 6.2: Possible signature transitions in a canonical representative of $\mathcal{L}_{\mathrm{proj}}$.

The sequences listed in the statement of this lemma represent every constraint-satisfying possibility for transition among the five signatures of Figure 6.2. ■

From these lemmas, we can proceed to prove Proposition 10. All of the arguments made at the start of the proof of Proposition 9 (e.g., for conciseness and the base case of completeness) hold true in this case, so the reasoning here begins with an arbitrary canonical feature $\mathbf{f}$ with $n > 1$ feature points. By inductive hypothesis $\mathbf{f}^-$ is a composition $T_{\mathrm{n}-1}$ of the tasks in $\mathcal{P}$ with the operators in $\mathcal{O}_6$. It remains to show the task represented by $\mathbf{f}$ is in $\mathcal{L}$.

Lemma 9 allows us to write $\pi^-\mathbf{f}^- = \{\mathbf{f}_B^-; \mathbf{f}_M^-; \mathbf{f}_P^-\}$ and $\pi\mathbf{f} = \{\mathbf{f}_B; \mathbf{f}_M; \mathbf{f}_P\}$. Only one of $\mathbf{f}_B \neq \mathbf{f}_B^-$, $\mathbf{f}_M \neq \mathbf{f}_M^-$, and $\mathbf{f}_P \neq \mathbf{f}_P^-$ holds. We consider each of these three cases separately.

- Suppose $\mathbf{f}_B \neq \mathbf{f}_B^-$. Then let

$$T_{\mathrm{n}}(\mathbf{f}) \quad = \quad \varepsilon_1 T_{\mathrm{n}-1}(\mathbf{f}^-) \ \wedge \ \pi\varepsilon_2 \neg T_{\mathrm{subsp}}(\mathbf{f}_B) \ .$$

Here, T_{subsp} (a subspace-coincident constraint) is T_{pp}, T_{col}, or T_{copl} when $\dim(\mathbf{f}^-) = 1, 2$, or 3, respectively. The expansion operators ε_1 and ε_2 ensure that the conjunction is performed on tasks with n feature ponts. The permutation π rearranges the feature points of $\mathbf{f}_B$ so that they occupy their original positions in $\mathbf{f}$. The only canonical feature satisfying $T_{\text{n}}(\mathbf{f}) = 0$ is the $\mathbf{f}$ under consideration.

- Suppose $\mathbf{f}_M \neq \mathbf{f}_M^-$. Then for each possible $\mathbf{f}_M$ there is a task in $\mathcal{L}$, T_{M}, which expresses the subspace constraints on the points of $\mathbf{f}_M$ and $\mathbf{f}_B$ Lemma 10 explicitly lists these tasks. As a result,

$$T_{\text{n}} \quad = \quad \varepsilon_1 T_{\text{n}-1} \ \wedge \ \pi\varepsilon_2 T_{\text{M}}$$

  demonstrates the element of $\mathcal{L}$ satisfied on $\mathbf{f}$ and no other canonical list.

- For the final possibility suppose that $\mathbf{f}_P \neq \mathbf{f}_P^-$. In this case $\{f_B; f_M; f_n\} = \{f_B^-; f_M^-; f_n\} \in \mathcal{L}_{\text{proj}}$. Because the elementary tasks of Lemma 10 include the most general positioned feature point supported by each signature sequence, one of the composite tasks of that lemma is satisfied on non canonical list other than $\{f_B; f_M; f_n\}$. Call the associated composite task T_{P}. Then the composite task we seek is

$$T_{\text{n}} \quad = \quad \varepsilon_1 T_{\text{n}-1} \ \wedge \ \pi\varepsilon_2 T_{\text{P}} \quad .$$

Hence, $\mathcal{L}$ is both concise and complete with respect to the decidable tasks in $\mathcal{T}_{\text{proj}}$. Thus, Proposition 10 holds. ■

6.6 Compositional Task Languages for $\mathcal{T}_{\text{aff}}$ and $\mathcal{T}_{\text{persp}}$

The languages for $\mathcal{T}_{\text{aff}}$ and $\mathcal{T}_{\text{persp}}$ differ from the compositional language for $\mathcal{T}_{\text{proj}}$ only in the positioning primitives needed. In the affine case the primitive T_{aff} replaces T_{proj}, as the following proposition states.

Proposition 11 *Let $\mathcal{P} = \{T_{\text{pp}}, T_{\text{col}}, T_{\text{copl}}, T_{\text{aff}}\}$ and $\mathcal{L} = \langle \mathcal{P}, \mathcal{O}_6 \rangle$. Then $\mathcal{L}$ is a concise and complete task-specification language for $\mathcal{T}_{\text{aff}}$.*

Perspective cameras introduce an analogous change.

Proposition 12 *Let $\mathcal{P} = \{T_{\text{pp}}, T_{\text{col}}, T_{\text{copl}}, T_{\text{persp}}\}$ and $\mathcal{L} = \langle \mathcal{P}, \mathcal{O}_6 \rangle$. Then $\mathcal{L}$ is a concise and complete task-specification language for $\mathcal{T}_{\text{persp}}$.*

Proof of Propositions 11 and 12: As in the projective case, the proof hinges on permuting any canonical list $\mathbf{f}$ into a concatenation of basis vectors and positioned vectors, $\mathbf{f} = \{\mathbf{f}_B; \mathbf{f}_P\}$. (No counterpart to $\mathbf{f}_M$ is needed.) The following two lemmas make this decomposition precise.

Lemma 11 *Every canonical feature $\mathbf{f}$ in $\mathcal{L}_{\text{aff}}$ has a permutation of the form $\pi\mathbf{f} = \{\mathbf{f}_B; \mathbf{f}_P\}$, where $\mathbf{f}_B$ is one of*

$$\left\{\begin{smallmatrix} 0 \\ 0 \\ 0 \end{smallmatrix}\right\} \quad \left\{\begin{smallmatrix} 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{smallmatrix}\right\} \quad \left\{\begin{smallmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{smallmatrix}\right\} \quad \left\{\begin{smallmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{smallmatrix}\right\}$$

and $\mathbf{f}_P$ is empty or consists of points added by Rule 2.

A counting argument demonstrates this. Rule 0 can contribute only one feature point: $\left\{\begin{smallmatrix} 0 \\ 0 \\ 0 \end{smallmatrix}\right\}$. Rule 1 contributes basis points e_i and increases the dimension of the resulting feature by one. Thus, the four lists above exhaust the possibilities for feature points added by Rules 0 and 1. ■

An argument of identical form proves the analogous lemma for perspective cameras. In this case $\begin{bmatrix} \alpha_1 \\ \alpha_2 \\ 0 \end{bmatrix}$ is the first appearing feature point whose second component is nonzero.

Lemma 12 *Every canonical feature* $\mathbf{f}$ *in* $\mathcal{L}_{\text{persp}}$ *has a permutation of the form* $\pi\mathbf{f} = \{\mathbf{f}_B; \mathbf{f}_P\}$*, where* $\mathbf{f}_B$ *is one of*

$$\left\{ \begin{smallmatrix} 0 \\ 0 \\ 0 \end{smallmatrix} \right\} \quad \left\{ \begin{smallmatrix} 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{smallmatrix} \right\} \quad \left\{ \begin{smallmatrix} 0 & 1 & \alpha_1 \\ 0 & 0 & \alpha_2 \\ 0 & 0 & 0 \end{smallmatrix} \right\}$$

(for any $\alpha_i \in \mathbb{R}$ *with* $\alpha_2 \neq 0$*);* $\mathbf{f}_P$*, if not empty, consists of points of the form* $\sum_{i=1}^{\dim(\mathbf{f})} \beta_i e_i$*.* ■

Consider the affine case of Proposition 11 first.

Each of the canonical lists of Lemma 11 can be expressed as a composition of the tasks in $\mathcal{P}$ as follows:

$$\left\{ \begin{smallmatrix} 0 \\ 0 \\ 0 \end{smallmatrix} \right\} \quad : \quad T_{\text{true}} \text{ or } T_{\text{false}} \tag{6.9}$$

$$\left\{ \begin{smallmatrix} 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{smallmatrix} \right\} \quad : \quad T_{\text{pp}} \tag{6.10}$$

$$\left\{ \begin{smallmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{smallmatrix} \right\} \quad : \quad T_{\text{3col}} \tag{6.11}$$

$$\left\{ \begin{smallmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{smallmatrix} \right\} \quad : \quad T_{\text{4copl}} \quad . \tag{6.12}$$

Every addition of a point by Rule 2 is equivalent (up to permutation and the possible omission of feature points) to one of

$$\left\{ \begin{smallmatrix} 0 & 1 & \alpha \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{smallmatrix} \right\} \quad \left\{ \begin{smallmatrix} 0 & 1 & 0 & \beta_1 \\ 0 & 0 & 1 & \beta_2 \\ 0 & 0 & 0 & 1 \end{smallmatrix} \right\} \quad \left\{ \begin{smallmatrix} 0 & 1 & 0 & 0 & \gamma_1 \\ 0 & 0 & 1 & 0 & \gamma_2 \\ 0 & 0 & 0 & 1 & \gamma_3 \end{smallmatrix} \right\} \quad . \tag{6.13}$$

Each of these is the only satisfying canonical list for one contraction of T_{aff}: the first removes feature points 3, 4, 6 and 7; the second, 4, 5, and 7; the third 5 and 6. As a result, the task which any canonical list $\mathbf{f}$ represents is equivalent to the composite task

$$T_{\mathrm{n}} \quad = \quad \varepsilon_1 T_{\mathrm{n}-1} \wedge \pi\varepsilon_2 T_* \ ,$$

where the expansions and permutation assure identical feature point lists as the domain of each component. T_* represents one of the contractions from (6.13) or one of the tasks listed beside (6.9).

Consider next the perspective camera case of Proposition 12.

Each of the canonical lists of Lemma 12 can be expressed as a composition of the tasks in $\mathcal{P}$ as follows:

$$\left\{\begin{smallmatrix} 0 \\ 0 \\ 0 \end{smallmatrix}\right\} \quad : \quad T_{\mathrm{true}} \text{ or } T_{\mathrm{false}} \tag{6.14}$$

$$\left\{\begin{smallmatrix} 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{smallmatrix}\right\} \quad : \quad T_{\mathrm{pp}} \tag{6.15}$$

$$\left\{\begin{smallmatrix} 0 & 1 & \alpha_1 \\ 0 & 0 & \alpha_2 \\ 0 & 0 & 0 \end{smallmatrix}\right\} \quad : \quad T_{\mathrm{3col}} \wedge cT_{\mathrm{persp}} \ . \tag{6.16}$$

The contraction of the positioning primtive T_{persp} in the last case ensures that the components of the third feature point are α_1 and α_2, as needed.

A feature point in $\mathbf{f}_P$, along with f_B, yields (up to permutation and the possible omission of feature points) one of

$$\left\{\begin{smallmatrix} 0 & 1 & \xi \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{smallmatrix}\right\} \quad \text{or} \quad \left\{\begin{smallmatrix} 0 & 1 & \alpha_1 & \gamma_1 \\ 0 & 0 & \alpha_2 & \gamma_2 \\ 0 & 0 & 0 & \gamma_3 \end{smallmatrix}\right\} \ . \tag{6.17}$$

Each of these is the only satisfying canonical list for a particular contraction of T_{persp}: the first removes feature points 4 and 5; the second elminates feature point 3. As a result, the task which any canonical list $\mathbf{f}$ represents is equivalent to the composite task

$$T_{\mathrm{n}} \quad = \quad \varepsilon_1 T_{\mathrm{n}-1} \ \wedge \ \pi\varepsilon_2 T_* \ \ ,$$

where the expansions and permutation assure identical feature point lists in the domain of each component. T_* represents one of the contractions from (6.17) or one of the tasks listed by (6.14).

Hence, Propositions 11 and 12 hold. ■

6.7 Verifying Encodings for Task Primitves

With these languages in place the final piece of the task-specification puzzle becomes finding a set of verifying encodings for the primitive tasks of Propositions 9 - 12. Section 5.7 provided differentiable verifying encodings for the positioning primitives based on the reconstructive use of visual data. Such reconstructive techniques work for the subspace-based tasks, too, but do not retain the positioning primitives' smoothness. For instance, projectively reconstructing three noncollinear points will yield coordinates of $\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$, $\begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$, and $\begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$. Even as the third point becomes arbitrarily close to the line formed by the other two, as long as they are not collinear the reconstructed coordinates will remain the same. When collinear, the third point will abruptly jump to $\begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}$, $\begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$, or $\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$.

Thus, explicitly image-based relationsips must play a role in developing smooth verifying encodings for the subspace-based tasks. For example, the point-to-point encoding E_{pp} of Section 3.3 verifies T_{pp} on $\mathcal{C}_{\mathrm{inj}}$ with an error function equal to the component image distances of the two feature points. As noted in Section 4.5, the E_{col} defined by Equation (3.4) does not verify T_{col} on $\mathcal{C}_{\mathrm{proj}}$. The following derivations provide encodings E_{col} and E_{copl} which do verify T_{col} and T_{copl} on $\mathcal{C}_{\text{proj}}$.

Collinearity

The collinearity encoding E_{col} of Section 3.3 fails to verify T_{col} in only one situation. When the three points involved in the task lie in a single plane with the two cameras' optical centers (an epipolar plane), they look collinear in each image regardless of their actual configuration. The following encoding remedies this problem.

$$E_{\text{col}}(\mathbf{y}) = \begin{cases} \begin{bmatrix} y_1^1 \cdot (y_2^1 \times y_3^1) \\ y_1^2 \cdot (y_2^2 \times y_3^2) \end{bmatrix} & \text{if the } y_i \text{ are outside an epipolar plane} \\ cr(y_1^1, y_2^1, y_3^1, e_1) - cr(y_1^2, y_2^2, y_3^2, e_2) & \text{otherwise} \end{cases} \tag{6.18}$$

Here, cr represents the cross ratio of four collinear points [20].

Using this definition of E_{col} instead of Section 3.3's, E_{col} verifies T_{col} on $\mathcal{C}_{\text{proj}}$. To see this, consider three points in an epipolar plane; in each image they must lie on a line with the epipole. Suppose the three feature points are collinear and let ℓ represent the line in task space through those three points. Then ℓ intersects the pencil of four lines extending from each camera's center through the three feature points and the image epipoles. The cross ratio of the lengths formed in each image by this pencil of lines must equal the cross ratio of lengths formed by that pencil's intersection with ℓ. Hence collinearity implies equal cross rations. Because only two of the three feature points are needed to uniquely determine ℓ, equal cross ratios also imply collinearity.

One issue remains: do image information and knowledge of $\mathbf{F}$ suffice to determine when image lines lie in an epipolar plane? Suppose ℓ_1 and ℓ_2 are lines in image 1 and image 2 of a stereo rig, respectively. Two conditions hold if and only if ℓ_1 and ℓ_2

lie in an epipolar plane. First, the epipole of each image i must lie on ℓ_i. Second, ℓ_2 must be the epipolar line (Equation (2.8) of any points on ℓ_1 other than the epipole and vice-versa. Thus, image information and knowledge of $\mathbf{F}$ do suffice to distinguish among the cases of (6.18). E_{col} does verify T_{col}.

Coplanarity

Four coplanar points form a quadrilateral whose diagonals must meet in the plane formed by the vertices. Depending on the pairing of the original feature points, there are three such sets of diagonals. On the other hand, the lines formed by two pairs of points in general position do not meet, because two lines through a single point are coplanar. In an image, however, even skew lines will seem to meet. That apparent coincidence suggests a means for coplanarity testing: if the apparent meeting points of the diagonals satisfy the epipolar constraint, they form actual points in task space and the feature is coplanar. Otherwise, the feature points lie in general position. The encoding

$$E_{\text{copl}}(\mathbf{y}) = \sum_{i=1}^{3} d_i^{2\prime}\mathbf{F}d_i^1 + \sum_{i,j} E_{\text{col}}(y_i, y_j, d_{ij}) \tag{6.19}$$

expresses this constraint. Each d_i^j represents the apparent intersection of the ith set of diagonals (as noted, there are three such sets) in the jth image ($j = 1$ or 2). The latter sum ensures the collinearity of the diagonals' (apparent) intersections and their vertices. It prevents noncoplanar configurations which "accidentally" satisfy the first term from satisfying the whole encoding.

Note that smoothness problems arise even in the two encodings presented here because of the two conditions in the definition of E_{col}. The next chapter will present approximations to these encodings which are everywhere smooth and continue to guarantee task-space accomplishment when satisfied. It will also explore applications and implementation details of the languages of Propositions 9 - 12. Chapter 8 then validates these theoretical results experimentally.

Chapter 7

Applying the Framework

Chapters 5 and 6 present the principal contribution of this thesis: provably complete and concise languages for specifying tasks under four well-known camera models. These languages offer a solid theoretical foundation for systems which program and execute visual tasks. As Chapter 2 notes, a great deal of research has investigated the implementation of such systems. Although reproducing that work is inappropriate here, this chapter and the next present several benefits, caveats, and examples arising from the languages of Chapter 6.

This chapter bridges the theory behind Propositions 9 - 12 and their implementation. It addresses several *system-level* considerations of a compositional task-specification system: simplifying the user interface, integrating a priori object knowledge, and exploiting specification redundancy. Chapter 8 handles further concerns at the level of individual task performance and demonstrates several implemented tasks.

7.1 The User Interface

The objective of simplicity motivated the development of Chapter 6's task-specification framework from the elementary tasks of Chapter 5. This section sets out to further simplify the programming of tasks. To start, consider a programmer's relationship with the operators of $\mathcal{O}_6$ (Figure 6.1). Task complement, conjunction, and disjunction express semantic connections among primitive constraints. An intention-providing agent, such as a human programmer or a high-level planning process, must choose a combination of these operators that yields a desired object configuration. Permutation, expansion, and contraction, on the other hand, guard the consistency of a task's functional representation. They ensure that conjuncts or disjuncts have domains of equal size and that the appropriate feature points of those primitives correspond.

While the intention behind a set of constraints must come from outside a task-specification system, the system itself can bear the burden of maintaining a consistent internal representation for those constraints. With a labeling for each feature point involved, component primitives can refer to their inputs by name, rather than by argument position, obviating the need for explicitly permuting argument lists. In the same manner, a system can implicitly handle expansion and contraction by adding or removing uninvolved feature points as necessary. The result is a three-operator programming language with familiar Boolean semantics.

As an example of a simplified composite task, consider aligning a wrench with the head of a bolt. Figure 7.1 suggests the desired configuration: two of the wrench's inner edges line up with outer edges of the bolt, and corresponding end points coincide. In the formalism of the previous chapter, the following expression specifies this

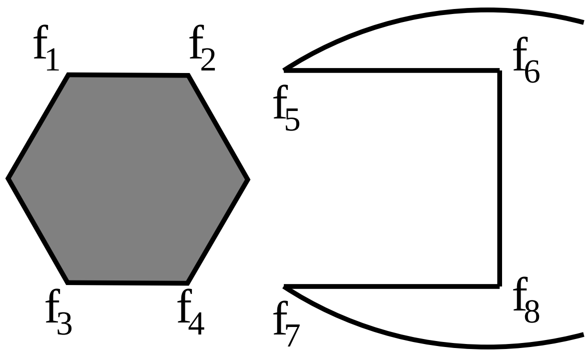


Figure 7.1: Feature points involved in an alignment task of a wrench to a bolt. In the desired configuration f_2 and f_5 coincide, f_4 and f_7 coincide, f_1, f_2, f_6 lie along a single line, and f_3, f_4, f_8 are also collinear.

task:

$$T_{\mathrm{wrench}}(\mathbf{f}) \;=\; \pi_1\varepsilon_1 T_{\mathrm{pp}}(\mathbf{f}) \,\wedge\, \pi_2\varepsilon_1 T_{\mathrm{pp}}(\mathbf{f}) \,\wedge\, \pi_3\varepsilon_2 T_{\mathrm{col}}(\mathbf{f}) \,\wedge\, \pi_4\varepsilon_2 T_{\mathrm{col}}(\mathbf{f}) \tag{7.1}$$

with

$$\begin{aligned}
\pi_1 &= (5,2,3,4,1,6,7,8) \quad & \pi_2 &= (4,7,3,1,5,6,2,8) \\
\pi_3 &= (1,2,6,4,5,3,7,8) \quad & \pi_4 &= (8,4,3,2,5,6,7,1) \\
\varepsilon_1(T(f_1,f_2)) &= T(f_1\ldots f_8) \quad & \varepsilon_2(T(f_1,f_2,f_3)) &= T(f_1\ldots f_8)
\end{aligned}$$

$$\text{and} \quad \mathbf{f} \;=\; \{f_1\ldots f_8\}\ .$$

The permutations and expansions here ensure that each primitive affects the desired feature points. With the labeling proposed in Figure 7.1, however, the simpler expression

$$T_{\mathrm{wrench}}(\mathbf{f}) \;=\; T_{\mathrm{pp}}(f_2,f_5) \,\wedge\, T_{\mathrm{pp}}(f_4,f_7) \,\wedge\, T_{\mathrm{col}}(f_1,f_2,f_6) \,\wedge\, T_{\mathrm{col}}(f_3,f_4,f_8) \tag{7.2}$$

suffices. Equation (7.2) distills the formal functional equivalence of (7.1) into only the desired task-space constraints and the connections between them. With a consistent labeling the system can translate between Equations (7.1) and (7.2), so that a programmer can use the latter specification without loss of precision. The balance of this work will express tasks in the simpler and more intuitive form of Equation (7.2).

To ensure correctness of both (7.2) and (7.1), the programmer and the system must agree on the feature-point labels in Figure 7.1. Several approaches might provide such consensus. With object models available, the system could assign names to feature points and display them as needed or as requested by a programmer. Alternatively, a human programmer could designate features by clicking on their images in a specified order. The examples of Chapter 8 employ the latter technique.

7.2 Incorporating A Priori Knowledge

In addition to offering modularity and simplicity, a compositional approach to task specification easily incorporates several kinds of a priori information. By adjusting the number and type of task primitives used, a programmer can acknowledge the natural hierarchy among possible constraints, the grouping of features into rigid objects, and the flexibility of redundant task specifications.

7.2.1 Constraint Subsumption

Consider again the eight feature points of the task illustrated in Figure 7.1. Only a fraction of the constraints attained by the feature $\mathbf{f} = \{f_1 \ldots f_8\}$, when in the goal configuration, appear in Equation (7.2). For example, the feature points f_1, f_2, and f_5 lie on a single line, yet no $T_{\mathrm{col}}(f_1, f_2, f_5)$ conjunct appears. Its absence does not diminish the specification's power, however, because satisfaction of $T_{\mathrm{pp}}(f_2, f_5) = 0$ necessarily entails that $T_{\mathrm{col}}(f_1, f_2, f_5) = 0$. In a word, $T_{\mathrm{pp}}(f_2, f_5)$ *subsumes* $T_{\mathrm{col}}(f_1, f_2, f_5)$.

More formally, one task T_a subsumes another, T_b, provided the accomplishment of T_a implies the accomplishment of T_b, i.e.,

$$T_a(\mathbf{f}) = 0 \implies T_b(\mathbf{f}) = 0 \, . \tag{7.3}$$

If (7.3) holds, then the single term T_a can replace the conjunction $T_a \wedge T_b$ without changing the meaning of a specification. By the same token, T_b can replace the disjunction $T_a \vee T_b$ in a specification without affecting the underlying task.

When they share feature points the subspace-coincidence primitives T_{pp}, T_{col}, and T_{copl} form a natural chain of subsuming constraints: $\pi_1 T_{\mathrm{pp}}(f_1, f_2)$ subsumes $\pi_2 T_{\mathrm{col}}(f_1, f_2, f_3)$, which in turn subsumes $\pi_3 T_{\mathrm{copl}}(f_1, f_2, f_3, f_4)$, for any permutations π_i. Thus, a task programmer may omit $T_{\mathrm{col}}(f_1, f_2, f_5)$ and $T_{\mathrm{copl}}(f_1, f_2, f_5, f_6)$ from the specification of the wrench task, because the conjunct $T_{\mathrm{pp}}(f_2, f_5)$ subsumes both of them. While a human programmer might easily, even unconsciously, excise such subsumed constraints from a specification, an automated system could employ (7.3) to distill an exhaustively generated list of constraints into a minimal specifying set for a particular task.

What would an exhaustive set of satisfied constraints look like? In the wrench task the only two point-to-point constraints (modulo permutations of arguments) appear in Equation (7.2). As for collinearity, the two lines of four feature points give rise to $2 \cdot \binom{4}{3} = 8$ possible constraints:

$$\begin{array}{cccc} (f_1, f_2, f_5)^* & (f_1, f_2, f_6) & (f_1, f_5, f_6) & (f_2, f_5, f_6)^* \\ (f_3, f_4, f_7)^* & (f_3, f_4, f_8) & (f_3, f_7, f_8) & (f_4, f_7, f_8)^* \end{array}$$

Of these eight, the two point-to-point constraints subsume the four marked with asterisks. The other four produce the two distinct collinearity constraints in (7.2); the doubling occurs because coincident points are freely interchangeable.

Letting f_2 represent f_5 and f_4 represent f_7, the wrench task admits $\binom{6}{4} = 15$ distinct coplanarity constraints:

$$\begin{array}{ccc}
(f_1, f_2, f_6, f_3)^* & (f_1, f_2, f_6, f_4)^* & (f_1, f_2, f_6, f_8)^* \\
(f_1, f_2, f_3, f_4) & (f_1, f_2, f_3, f_8)^* & (f_1, f_2, f_4, f_8) \\
(f_1, f_6, f_3, f_4)^* & (f_1, f_6, f_3, f_8)^* & (f_1, f_6, f_4, f_8)^* \\
(f_2, f_6, f_3, f_4) & (f_2, f_6, f_3, f_8)^* & (f_2, f_6, f_4, f_8) \\
(f_3, f_4, f_8, f_1)^* & (f_3, f_4, f_8, f_2)^* & (f_3, f_4, f_8, f_6)^*
\end{array}$$

Of these, subsumption by collinearity or coincidence constraints eliminates all but the four unstarred. Instead of subsumption, it is the rigidity of the wrench and bolt, along with those tools' intrinsic geometry, which obviates the need for the final four T_{copl} conjuncts. The following section treats such cases in more detail.

7.2.2 Rigidity

As presented, the languages of Chapter 6 do not presume any rigidity constraints on the feature points involved in a task. In essence, those languages are assuming the most flexible possible scenario, one in which the task-performing system controls each feature point independently of the others. If known, a fixed relationship among some feature points can simplify a specification in a manner similar to subsumption — by excluding from consideration constraints which hold a priori.

For instance, because the feature points f_5, f_6, f_7, f_8 on the wrench always form a single plane, as do f_1, f_2, f_3, f_4 on the bolt, the corresponding coplanarity constraints, $T_{\text{copl}}(f_1 \ldots f_4) = 0$ and $T_{\text{copl}}(f_5 \ldots f_8) = 0$ (identical to $T_{\text{copl}}(f_2, f_6, f_4, f_8) = 0$, given the point-coincidence coinstraints present) always hold. Thus, they need not appear in the specification of (7.2). Subsumption then handles the final two coplanarity constraints on the above list, $T_{\text{copl}}(f_2, f_6, f_3, f_4)$ and $T_{\text{copl}}(f_1, f_2, f_4, f_8)$, as

$$T_{\text{copl}}(f_1, f_2, f_3, f_4) \;\wedge\; T_{\text{col}}(f_1, f_2, f_6) = 0 \quad\Longrightarrow\quad T_{\text{copl}}(f_2, f_6, f_3, f_4) = 0$$
$$T_{\text{copl}}(f_1, f_2, f_3, f_4) \;\wedge\; T_{\text{col}}(f_3, f_4, f_8) = 0 \quad\Longrightarrow\quad T_{\text{copl}}(f_1, f_2, f_4, f_8) = 0 \;\; .$$

Hence the four terms of Equation (7.2) guarantee all coincidence, collinearity, and coplanarity constraints satisfied by the feature $\mathbf{f}$ in Figure 7.1.

7.2.3 Necessary sets of constraints

The previous sections demonstrate that subsumption and rigidity ensure that the conjuncts in Equation (7.2) suffice to accomplish the wrench-positioning task. That sufficiency begs the question of necessity: are all four conjuncts necessary, or would an even more concise specification also suffice? While a complete answer to the question of minimal specifying sets of constraints requires the power of a geometric planner [3, 5, 58, 86, 101, 116], here we point out several rough heuristics for determining specification sufficiency.

To a first approximation, primitives in which a point on one rigid object coincides with a point on a target object constrain three degrees of freedom of the objects' relative pose. Point-to-line primitives constrain two degrees of freedom and point-to-plane primitives, one. A simple but imperfect calculus results from adding together conjuncts' constraints until all of a task's degrees of freedom are specified. For instance, the wrench task of Figure 7.1 specifies a six-DOF constraint. By the above reasoning any three of the four conjuncts of (7.2) will suffice to specify that task, e.g.,

$$T_{\mathrm{pp}}(f_4, f_7) \;\wedge\; T_{\mathrm{col}}(f_1, f_2, f_6) \;\wedge\; T_{\mathrm{col}}(f_3, f_4, f_8) \;=\; 0 \;, \tag{7.4}$$

which specify 3, 2, and 2 degrees of freedom, respectively, according to this calculus. That the primitives' constrained freedoms sum to seven suggests that an even more minimal specification would also suffice. In fact, the conjunction

$$T_{\mathrm{pp}}(f_4, f_7) \;\wedge\; T_{\mathrm{copl}}(f_1, f_3, f_6, f_7) \;\wedge\; T_{\mathrm{col}}(f_3, f_4, f_8) \;=\; 0 \;, \tag{7.5}$$

when satisfied, does guarantee the accomplishment of the same task.

These alternative specifications can lighten the burden on a vision system which supports task execution. If, for instance, a tracker following feature f_5 becomes occluded or lost, the specification of Equation (7.4) and its corresponding encoding can continue to guide the task to completion. Moreover, Equation (7.5) ensures the desired final configuration without using either f_5 or f_2. If additional features beyond these two are lost, the full wrench-alignment task can not be verified. Remaining visual information may, however, continue to control a subtask consisting of fewer degrees of freedom. Work such as [17, 50] explores such possibilities.

The point-to-point constraints alone,

$$T_{\mathrm{pp}}(f_2, f_5) \;\wedge\; T_{\mathrm{pp}}(f_4, f_7) \;=\; 0 \;, \tag{7.6}$$

do not fully specify the wrench alignment task — Equation (7.6) does not ensure that the bolt top and wrench face are coplanar. This failure underscores the very rough reasoning provided by the above DOF-counting calculus. Additional rules, such as noting that two point-coincidence primitives constrain five, not six, degrees of freedom, only temporarily stave off the need for a true geometric planner in the general case. Even the original constraint of Equation (7.2) admits two satisfying configurations: the one depicted in Figure 7.1 and one in which the wrench is reflected around the line $\overline{f_5 f_7}$. Initial conditions and control parameters decide, in effect, which of these poses to achieve.

This section has pointed out the flexibility of Chapter 6's specification languages and acknowledged the geometric complexity possible in compositions of their primitives. A full accounting of that complexity lies outside those languages and depends fundamentally on the details of a particular task. Yet the framework of Chapters 5 and 6 do fully determine the visual information necessary for one aspect of individual tasks: that of specifying metrically defined set points. The following section explores the structure of such measurement-based constructions.

7.3 Calibration and Set Point Construction

How can a task specification system take advantage of object information more quantitative than sets of rigidity constraints? Knowledge of both rigidity and relative distances permits the construction of a coordinate system around an object's known visual features. The resulting object-centered frame frees a system from reliance on visible features and allows the use of setpoints defined metrically, i.e., based on absolute measurements. Because measurement-based tasks are often naturally phrased in terms of Euclidean ratios or absolute distances, object-centered calibration most extends the practical capabilities of weakly calibrated systems, which are otherwise limited to cross-ratio-based metrical constraints.

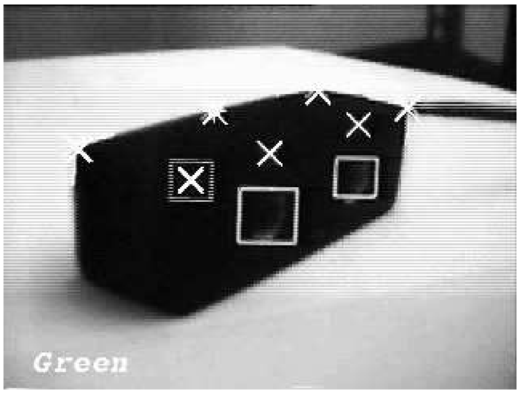
Many methods are available for calibrating to an object's coordinate system [15, 32, 113, 120]; the accuracy of a number of different techniques are compared in [94]. In general, redundant visual information improves calibration accuracy and decreases sensitivity to noise. Even without redundancy, however, there exists an information threshhold below which no metrical estimation is possible at all. That lower bound depends intimately on the geometry of the feature points available, and it is exactly this dependence that the elementary-task construction rules of Chapter 5 capture. Those rules categorize precisely and completely the conditions under which enough visual information exists to construct metrically defined set points for each set of camera models considered.

As an example, consider a task involving the block shown in Figure 7.3. The block has dimensions of 81mm × 30mm × 30mm, with two evenly spaced holes extending through its body. A weakly calibrated pair of cameras observes the block. The visible corners of the block's upper surface provide four coplanar reference points with known relative geometry. In addition, the front side's holes can be used to construct two more feature points on the edge closest to the cameras, resulting in six coplanar feature points (Figure 7.3).

Taking the corner nearest the cameras as the origin, those six points form an object-specific coordinate frame with the following homogeneous coordinates

$$\begin{bmatrix} 0 & 27 & 54 & 81 & 0 & 81 \\ 0 & 0 & 0 & 0 & 30 & 30 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \quad . \tag{7.7}$$

Through the calibration technique proposed by Faugeras in [19], the coordinates in (7.7) and their projections yield a 3×4 projection matrix M for each camera. M translates any desired set point $f_* = \begin{bmatrix} \alpha \\ \beta \\ 0 \\ 1 \end{bmatrix}$, expressed in the block's coordinate frame, into an image feature with homogeneous coordinates Mf_*. Figure 7.2 shows the projection of $\begin{bmatrix} 40.5 \\ 30 \\ 0 \\ 1 \end{bmatrix}$, the midpoint of the block's far edge, along with the six fiducial points. As Section 5.4 proves, without the calibration made possible by the fiducial points' known geometry, a weakly calibrated pair of cameras can not recover this midpoint. The additional information in $\{f_1 \ldots f_6\}$, in effect, upgrades the weakly calibrated system to one fully calibrated with respect to the provided coordinate frame.



$$\begin{bmatrix} 1 & 0 & 1 & 1.33 & 0 & 2 \\ 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Figure 7.2: Six fiducial points on the top surface of a block and canonical coordinates for those points. The feature point closest to the camera (lowest in the image) is the origin of the object-centered coordinate system. Because $[\alpha\ \beta\ 1\ 0]'$ is a legal successor to the six fiducial points according to Proposition 3, any metrically defined point in the plane of the block's top surface can be constructed. Here the estimate of the midpoint of the far edge is shown with an asterisk.

Propositions 3, 4, and 5 demonstrate how each point of a feature in canonical form constrains either the projective, affine, or similarity-based coordinate frame describing that feature. (A system known only to be bijective gains little from seeing additional feature points.) The proofs of those propositions delineate a procedure for converting points' known coordinates to a canonical form. This procedure creates a task-space transformation between object-centered and canonical coordinates. Each proposition's rules then determine the extent to which metrical positioning is possible. Legal successors of a visible feature can be constructed from that feature's information; points that are not legal successors can not.

For the example coordinate system of (7.7), the following equation relates object-centered coordinates (on the left) and canonical coordinates ($\mathbf{f}$, in the center):

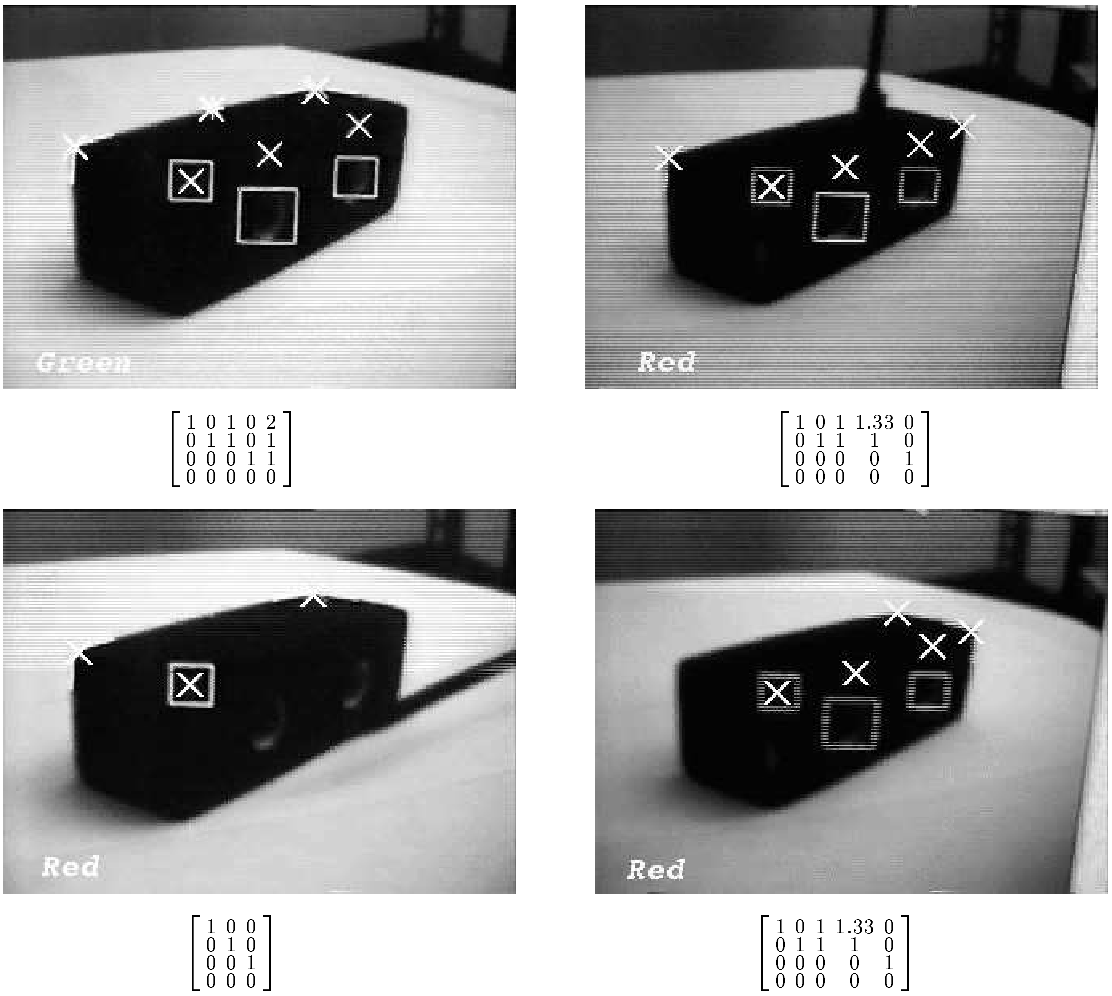
$$\begin{aligned} \begin{bmatrix} 0 & 27 & 54 & 81 & 0 & 81 \\ 0 & 0 & 0 & 0 & 30 & 30 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} &= \begin{bmatrix} 0 & 54 & 0 & 0 \\ 0 & 0 & 20 & 0 \\ 0 & 0 & 0 & 1 \\ -1 & 2 & 0.66 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 1 & 1.33 & 0 & 2 \\ 0 & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} -1.0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.5 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1.0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1.5 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1.5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1.5 \end{bmatrix} \\ &= \mathbf{A}\,\mathbf{f}\,\mathbf{D}\ . \end{aligned} \tag{7.8}$$

Because by Proposition 3 any point in the form $\begin{bmatrix} \alpha \\ \beta \\ 1 \\ 0 \end{bmatrix}$ is a legal successor of $\mathbf{f}$, this object-specific frame supports general metrical positioning in the plane spanned by f_1, f_2, and f_5 – the block's upper face. By the same token, because $\begin{bmatrix} \alpha \\ \beta \\ \gamma \\ 1 \end{bmatrix}$ is not, in general, a legal successor of $\mathbf{f}$, $\mathbf{f}$ does not suffice to define general 3d setpoints in the block's frame. This restriction is a natural result of the lack of fiducial points outside the block's top surface.

Figure 7.3 shows a number of situations in which some of $\mathbf{f}$'s feature points are occluded. When applied to the visible feature points remaining, the procedure which produced $\mathbf{f}$ in Equation 7.8 yields the canonical forms shown beneath each picture. When $\begin{bmatrix} \alpha \\ \beta \\ 1 \\ 0 \end{bmatrix}$ is a legal successor (above and to the left), the information available suffices to recover the desired setpoint f_*, drawn in its estimated location. Two examples in Figure 7.3 demonstrate single-point occlusions for which the remaining information does not suffice; in these cases calibration to the block's 2d coordinate frame is impossible. As long as $\begin{bmatrix} \alpha \\ 1 \\ 0 \\ 0 \end{bmatrix}$ is a legal successor of the remaining feature (the situations to the left), metrical positioning in the line of the block's near edge stays possible.

It is well-known that four coplanar points (with no three collinear) are required to perform metrical positioning with respect to those points in the plane. This rule and others governing coordinate system construction are concisely summarized by Propositions 3-5 for different sets of cameras. By maintaining a list of visible object features, converting to canonical form, and employing the propositions' feature-successor rules, a task-monitoring process can appropriately continue with a task or stop and search for additional visual information.

This chapter has examined some of the system-level implications of Chapter 6's languages: a simple user interface, the automatic culling of feature constraints, and minimum-information requirements for constructing metrical set points. Other components of a complete system for task performance and evaluation lie outside the faculties provided by Propositions 3-5. For example, gauging the effects of rigidity constraints and determining minimally sufficient compositions of primitives for particular tasks require the power of a general-purpose geometric planner. As is, however, the presented compositional framework does support systems in which all feasible tasks can be specified. The next chapter examines the implementation of such systems at the level of the individual task.



$$\begin{bmatrix} 1 & 0 & 1 & 0 & 2 \\ 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad \begin{bmatrix} 1 & 0 & 1 & 1.33 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix} \qquad \begin{bmatrix} 1 & 0 & 1 & 1.33 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Figure 7.3: A series of partial occlusions and the resulting reconstructive capabilities of the remaining feature points. The observing cameras are weakly calibrated. In each case, tracked (or computed) image features are marked with an overlaid X. Canonical coordinates for those features appear below each image. The point of interest is the midpoint of the far block edge. When the available information suffices to reconstruct that midpoint (at the upper left), an asterisk appears there. The image overlays, when green, convey the presence of sufficient information to construct that setpoint; red indicates insufficient information. Here, the colors appear printed in the image corners; color MPEGs are available at `www.cs.hmc.edu/~dodds/thesis.html`.

Chapter 8

Implementation and Experiments

This chapter explores how the use of compositional languages impacts the setup and execution of particular tasks. Implemented examples — for each primitive and operator, as well as several applications — demonstrate the feasibility of Chapter 6's task-specification framework both for the camera models considered and, with some caveats, for completely uncalibrated projective cameras.

8.1 Control-level implementation

An advantage of the compositional nature of Chapter 6's languages is that after a user has specified a task, the details of the low-level controller used to perform it can be automatically generated. In fact, even within a single hand/eye system distinct tasks need not take the same approach to control. Once the operators of Figure 6.1 yield a verifying image-based encoding for a task, any controller which can drive the resulting image error to zero will achieve that task. Developing such controllers has been the focus of considerable work in the field of image-based visual control [10, 47, 83, 104].

Examples in this chapter rely on two control strategies: in the first, the relationship between the end effector and cameras is approximately known; in the second, nothing is assumed about that relationship. Equation (2.12), repeated here, underlies both strategies:

$$\dot{r} = \mathbf{J}_r^{+} \Delta y \ .$$

This equation describes a linear relationship between image error (Δy) and robot joint velocities ($\dot{r}$). In the case of online model updating, $\mathbf{J}$ is continually estimated during task performance through first-order differencing [48]

$$\mathbf{J}_{\mathrm{new}} = \mathbf{J}_{\mathrm{old}} + \frac{(\Delta y - \mathbf{J}_{\mathrm{old}} \Delta r)\Delta r'}{\Delta r' \Delta r} \ ,$$

where encoders provide Δr, the change in joint position, and the cameras provide Δy, the change in the locations of the image features. With an a priori model, on the other hand, the system uses a precomputed estimate of $\mathbf{J}$ which does not change through the execution of the task.

A Zebra Zero manipulator operating under Linux performed all of the tasks in this chapter. Within that system a custom driver resolves velocity commands — the $\dot{r}$ in Equation 2.12 — into positional setpoints for the robot's joints at 100 hz; each joint's HCTL-1100 motor control chip then achieve those setpoints through a 2000 hz PID control loop. The system gain is kept low enough to avoid the need to account for the arm's dynamics.

The following section presents examples using online model estimation; Section 8.3's tasks rely on a priori hand/eye models as well. None of the tasks exhibited, however, *require* either of these control strategies. Other techniques, e.g., more strongly modeled position-based control, hybrid strategies, or high-performance controllers that account for robot dynamics, could also be used. It is the separation of task specification and task execution provided by Chapter 6's languages that gives these hand/eye systems such flexibility in their underlying control implementation.

8.2 Implementing Task Primitives

Point-coincidence T_{pp} Figure 8.1 demonstrates the initial and final configurations of two floppy disks controlled by the equation

$$\dot{r} = \mathbf{J}^{+} E_{\mathrm{pp}}(\mathbf{y}) \ , \tag{8.1}$$

which is Equation (2.12) instantiated with a verifying encoding for the point-coincidence task. In this instantiation the manipulator brings the lower right corner of a controlled floppy disk to coincide with the upper left corner of a stationary disk.

Because robot joint velocities ($\dot{r}$) are proportional to the visual error (E_{pp}), convergence to the set point is not absolute. As long as the control loop runs, small perturbations in the estimates of the corners' positions lead to corresponding small motions about the goal. In this case final positioning accuracy remains within 2 millimeters. The cameras have 11mm focal lengths and observe the scene from a distance of 40cm. Figure 8.2 shows the final configuration from a viewpoint that highlights the accuracy achieved.

Note that although several of the geometric parameters of the cameras and the workcell were measured, none contributed to the accomplishment of the task. Nor did the system take advantage of the fact that a projective transformation well approximates each camera's imaging transformation, i.e., that the camera pair is contained

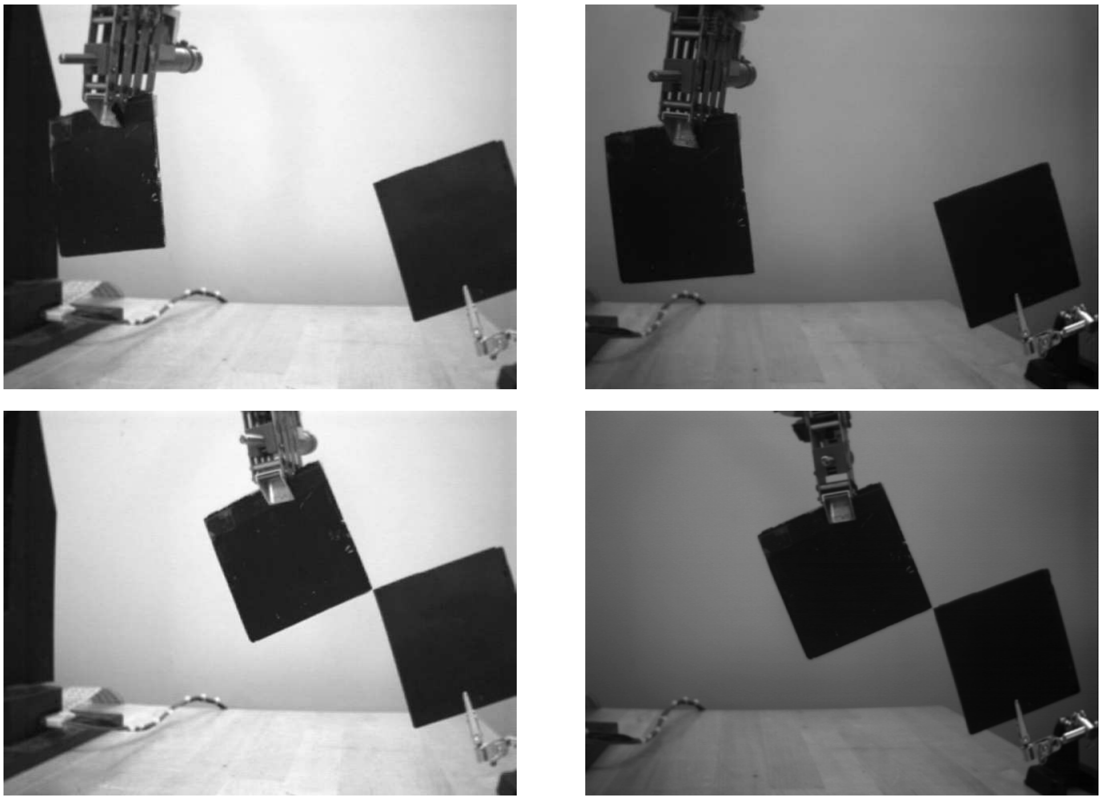


Figure 8.1: **Point-coincidence**: (**Top**) The stereo rig's initial and final views of the execution of a point-to-point constraint. The lower right corner of the robot-held floppy is made coincident with the upper left corner of the target disk. Figure 8.2 presents another view of the final configuration.

in $\mathcal{C}_{\text{uncal}}$. Thus, the cameras were assumed to produce only an injective transformation of the workcell. Although T_{pp} can be *verified* on any injective stereo rig, practical task performance does rely on more than injectivity. Specifically, Equation (8.1)'s success in actually achieving the desired constraint (rather than merely testing its truth or falsehood) depends on the smoothness of the task-space-to-image-space transformation.

In addition to providing a basis for composite tasks, simple primitives can be cast as applications in their own right. For example, a needle-threading task involves no more than a point-coincidence constraint. Figure 8.3 shows the initial and final configurations of an implementation of a needle-threading task. An endstopped

Figure 8.2: The accuracy of the point-coincidence task depends on the accuracy of the vision system's estimates of the salient features' positions. Accuracy additionally depends on the cameras' effective resolution within the task's workspace. Here, with a focal length of 11 mm and at a distance of about 400mm, the disk positioning task of Figure 8.1 admits an error of less than 2 mm.

line tracker maintains the image position of the tip of the wire being inserted, while the needle eye is tracked with a sum-of-squared-differences template. The XVision tracking system [33], with some task-specific tuning, provides these image-processing algorithms, as well as the corner-tracking used in the disk-positioning task. Using uncalibrated cameras at a distance of 200mm, the system achieves a final configuration error accurate enough to perform the insertion.

Collinearity T_{col} Achieving a collinearity constraint does require more than just a smooth transformation from task space to image space. As the results of Chapter 5 show, the observing stereo rig must be weakly calibrated in order to guarantee that T_{col} can be verified in general.

Only a small subset of the possible goal configurations for a collinearity task requires weak calibration, however. The verifying encoding E_{col} provided in Equa-

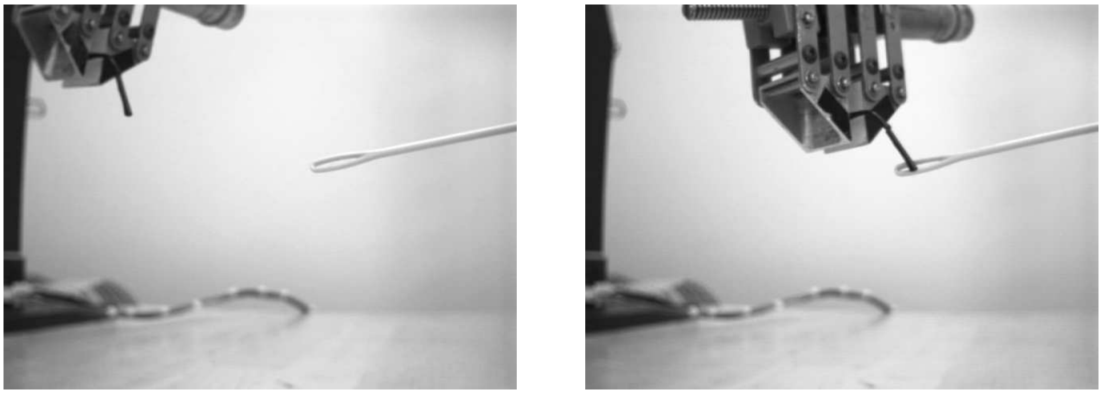


Figure 8.3: Initial and final configurations in a needle-threading application of a point-coincidence constraint. With a camera distance of 20cm the image constraints suffice to prepare the wire for insertion.

tion (6.18) and reproduced here:

$$E_{\text{col}}(\mathbf{y}) = \begin{cases} \begin{bmatrix} y_1^1 \cdot (y_2^1 \times y_3^1) \\ y_1^2 \cdot (y_2^2 \times y_3^2) \end{bmatrix} & \text{if the } y_i \text{ are outside an epipolar plane} \\ cr(y_1^1, y_2^1, y_3^1, e_1) - cr(y_1^2, y_2^2, y_3^2, e_2) & \text{otherwise} \end{cases}$$

distinguishes this subset exactly: when outside an epipolar plane, image collinearity suffices. When in an epipolar plane, the images of the three points and and the stereo rig's epipoles necesarily lie on a single line in each image. Those four collinear image points have equal cross ratios in each image if and only if the three task-space points are collinear.

As long as the three feature points involved in the T_{col} task do not lie in an epipolar plane of the observing stereo rig, weak calibration is not needed to verify the task. As an example, Figure 8.4 shows the initial and final configurations of a task in which the intersection of two scissor blades is brought to the line formed by a taut wire. Because that line is known a priori not to lie in an epipolar plane, the control does not require calibration information beyond the local, online estimates of

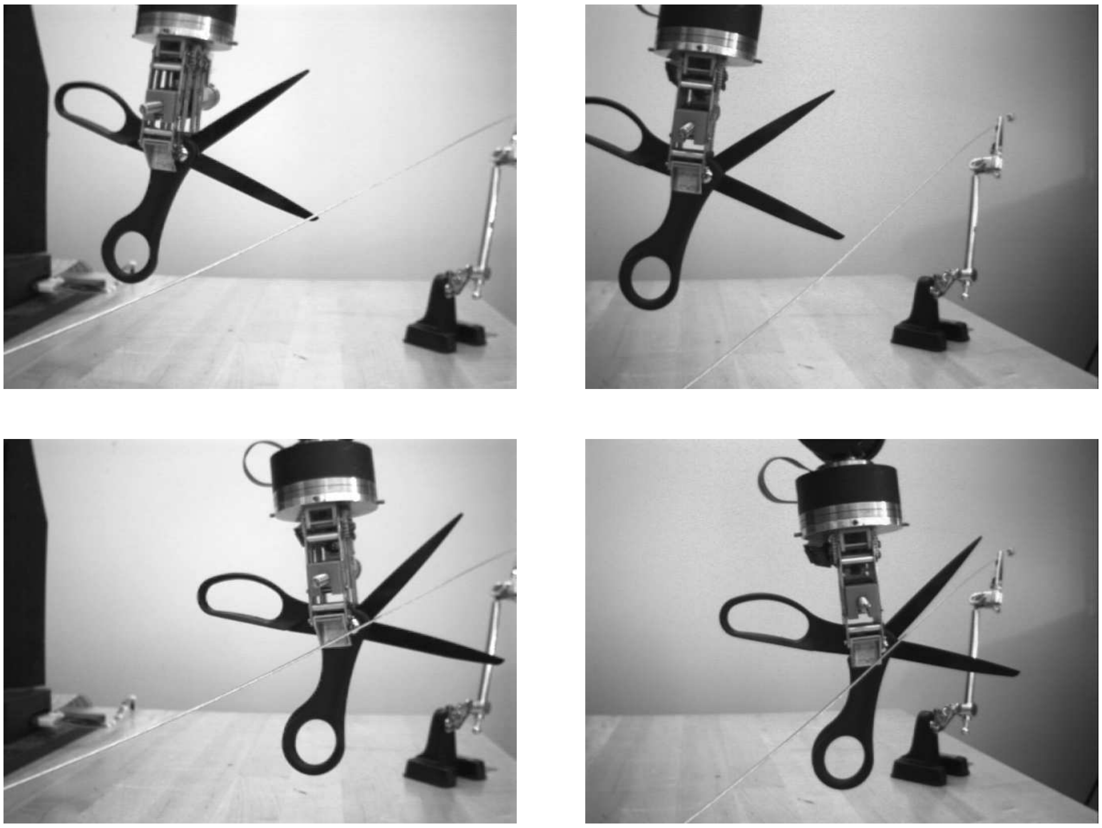


Figure 8.4: Two cameras' views of the initial and final configuration of a point-to-line task. Here a pair of scissors is servoed to a point suitable for cutting a wire. This task did not require the weak calibration of the cameras, because the line of points which satisfy the task do not lie near an epipolar plane.

$\mathbf{J}$ needed for convergence and stability. Further, the verifying encoding in this case

$$E_{\mathrm{col}}(\mathbf{y}) = \begin{bmatrix} y_1^1 \cdot (y_2^1 \times y_3^1) \\ y_1^2 \cdot (y_2^2 \times y_3^2) \end{bmatrix} = \begin{bmatrix} y^1 \cdot \ell^1 \\ y^2 \cdot \ell^2 \end{bmatrix} \tag{8.2}$$

does not require knowledge of two distinct points on the target line in each image. Each observed line feature yields exactly the information provided by the cross product of any two distinct points on that line [72].

The situation depicted in Figure 1.3 in the introduction is a stylized example of what can go wrong when the target line does lie in or near an epipolar plane.

Figure 8.5 reinforces this observation with the results of a collinearity task in which the manipulator has achieved a very small error in each image, but the task-space error remains large. In this case the manipulated endpoint is within 0.2 pixels of the finish line's projection in the left image and within 0.5 pixels in the right image. In the robot's workspace, however, the manipulated point is about 106mm from its desired destination. In this task the cameras have a baseline of 275mm and stand approximately 230mm from the stationary feature points.

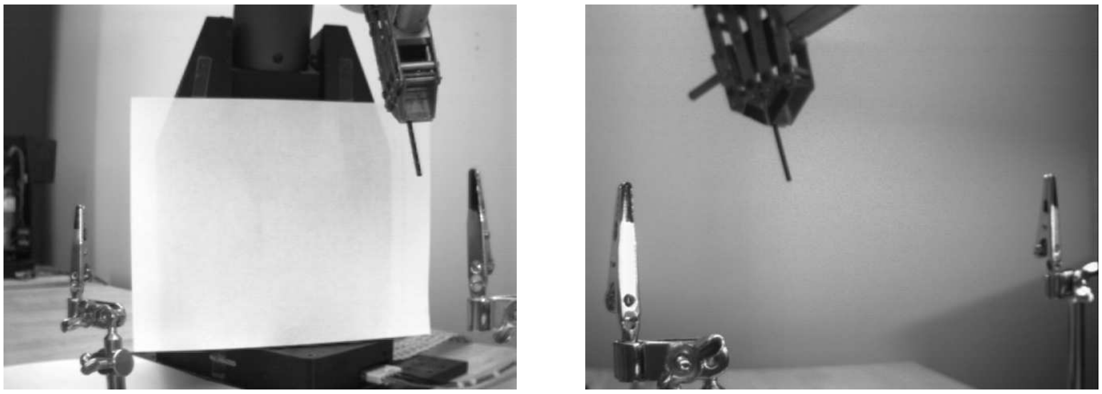


Figure 8.5: T_{col} **not verified**: The final configuration is shown for each of the two cameras' images. Because the probe's destination line lies in an epipolar plane of the stereo rig, collinearity in each image does not ensure collinearity in task space. The image errors here are 0.2 and 0.5 pixels, respectively, but the task-space error is about 106mm. That the gripper appears as small as it does hints at its distance from the target line.

The verifying collinearity encoding of Equation (6.18) overcomes this problem by insisting on the equality of the cross ratios of the epipoles and feature-point images, when those four image points are collinear. While mathematically correct, this encoding is not a suitable basis for a controller. In a real system noise in the measured visual data as well as calibration incaccuracies will conspire to make the measured locations of those four image points *never* lie on a single image line. As a result, the control system will never check the points' cross ratios for equality — ultimately leading to results such as Figure 8.5's.

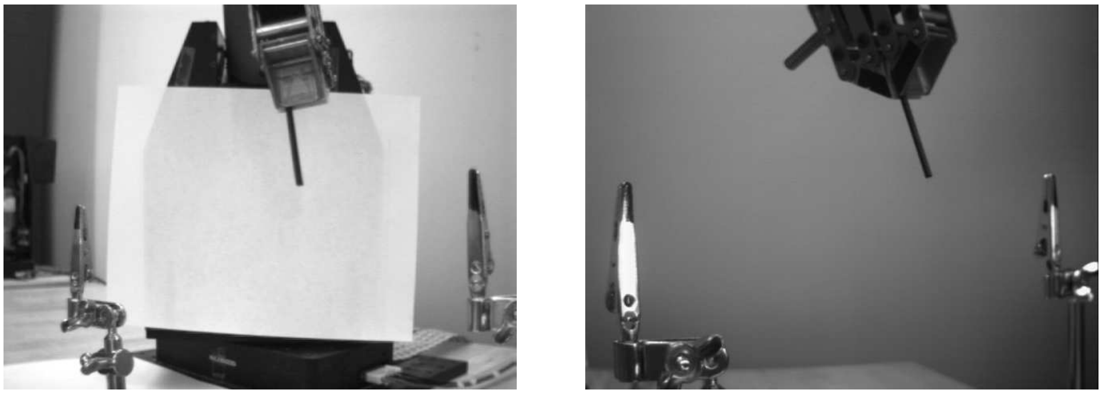


Figure 8.6: T_{col} **verified**: Using a continuous, "safe" approximation to the verifying encoding E_{col}, the system achieved a collinear configuration of three points in an epipolar plane. The images of the manipulated point are 0.4 and 0.3 pixels from the reference lines, respectively, and the difference in cross ratios along the nearest epipolar line is 0.02. (In Figure 8.5, the difference in cross ratios is 1.79.) In task-space the manipulated point is 1mm from the line in the vertical direction (parallel to the image planes) and 3mm from the line in depth.

"Safe" approximations to verifying encodings To handle this sensitivity to noise, we consider approximations to the verifying encoding of Equation (6.18). Intuitively, we would like an approximation which does not relinquish the guarantee that zero visual error implies zero task-space error. More formally, suppose that the encoding $E(\mathbf{y}) = 0$ verifies the task $T(\mathbf{f}) = 0$. Define $\hat{E}(\mathbf{y}) = 0$ to be a *safe* approximation of $E(\mathbf{y}) = 0$ if the following two conditions hold:

(1) $\quad \hat{E}(\mathbf{y}) = 0 \quad \Longrightarrow \quad T(\mathbf{f}) = 0$

(2) $\quad \exists\, \mathbf{f}$ such that $\hat{E}(\mathbf{y}) = \hat{E}(C_{\text{actual}}(\mathbf{f})) = 0$

By Condition 1 completion of the visually defined task entails completion of the physical task. Condition 2 ensures that the visually defined task not be impossible, a case which would make Condition 1 vacuously true. Note that for a safe approximation to a verifying encoding, $T(\mathbf{f}) = 0$ need not imply that $\hat{E}(\mathbf{y}) = 0$, i.e., there may be configurations in which the underlying task is satisfied, but the visual task is not. As long as a control system can reduce the visually defined error to zero, however, a safe approximation will suffice to accomplish the task.

In the case of the collinearity task, we can construct a safe approximation to E_{col} as follows. Let ℓ_1 and ℓ_2 represent epipolar lines in the respective cameras' images which best fit the estimated location of the epipoles and the two stationary feature points. That is, ℓ_1 and ℓ_2 are the corresponding epipolar lines that minimize their squared distances from the reference points on the target line. The work of Hartley and Sturm [36] details the construction of such lines for the purpose of finding optimal correspondence estimates in the presence of noise. Let y_i^j represent the image of the ith feature point ($i \in \{1, 2, 3\}$) in the jth image ($j \in \{1, 2\}$), where y_1^1 and y_1^2 indicate the images of the manipulated feature point. Finally, let $\hat{y}_i^1$ and $\hat{y}_i^2$ denote the points on ℓ_1 and ℓ_2, respectively, that are closest to y_i^1 and y_i^2.

With these conventions in place, the encoding

$$\hat{E}_{\text{col}}(\{y_1, y_2, y_3\}) = \begin{pmatrix} y_1^1 \cdot (y_2^1 \times y_3^1) \\ y_1^2 \cdot (y_2^2 \times y_3^2) \\ cr(\hat{y}_1^1, \hat{y}_2^1, \hat{y}_3^1, e^1) - cr(\hat{y}_1^2, \hat{y}_2^2, \hat{y}_3^2, e^2) \end{pmatrix} \tag{8.3}$$

defines a safe approximation to E_{col}. To show this rigorously, note that when the first two components of $\hat{E}_{\text{col}}$ are zero, the feature points' images are collinear in each camera's retina. Either the task-space collinearity constraint is satisfied or the three feature points lie in an epipolar plane. In the latter case the third component of $\hat{E}_{\text{col}}$ forces the three feature points to lie on a single line, just as in E_{col}. In the former case the third component of $\hat{E}_{\text{col}}$ is an additional constraint on the relative positions of f_1, f_2, and f_3. Thus, not all configurations satisfying the task T_{col} satisfy the approximate encoding $\hat{E}_{\text{col}}$. Yet $\hat{E}_{\text{col}}$ does meet Condition 2 of the definition of a safe approximation, because the constraint

$$cr(\hat{y}_1^1, \hat{y}_2^1, \hat{y}_3^1, e^1) - cr(\hat{y}_1^2, \hat{y}_2^2, \hat{y}_3^2, e^2) = 0$$

holds when the manipulated feature point f_1 coincides with one of the stationary feature points f_2 or f_3. In general there will be additional configurations satisfying $\hat{E}_{\text{col}}$ that do not involve feature point coincidence, such as the one realized in Figure 8.6.

Figure 8.6 demonstrates the result of using the safe approximation $\hat{E}_{\text{col}}$ under the same conditions depicted in Figure 8.5. Here the small visual error present in the final configuration (0.4 and 0.3 pixels from the image point to the target lines and 0.02 difference in cross ratios) corresponds to a small task error (1mm parallel to the image planes and 3mm in depth).

Coplanarity T_{copl} Whereas a system can often achieve and verify the collinearity of three points without recourse to the cameras' weak calibration, verifying the coplanarity of four feature points requires weak calibration in all but trivial cases (e.g., when two of the points coincide). To see this, consider a 3×4 full-rank transformation C, which models a single camera's imaging function. Let $\mathbf{f}$ be a 4×4 matrix of general-position feature points' homogeneous coordinates; $\mathbf{f}$ must have rank 4. Let $\mathbf{g}$ be the homogeneous coordinates of four coplanar points with no three collinear. Then $\mathbf{g} = \mathbf{G} \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$ for some invertible 4×4 $\mathbf{G}$. Finally, let $\begin{bmatrix} y_1 & y_2 & y_3 & y_4 \end{bmatrix} = \mathbf{y} = C\mathbf{f}$; because $\mathbf{y}$ has full rank, in general there exist nonzero scalars α_i such that $\sum_{i=1}^{3} \alpha_i y_i = y_4$. It follows that

$$\mathbf{y} = C\mathbf{f} = C\mathbf{f} \begin{bmatrix} \alpha_1 & 0 & 0 & 0 \\ 0 & \alpha_2 & 0 & 0 \\ 0 & 0 & \alpha_3 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \mathbf{G}^{-1}\mathbf{g}$$

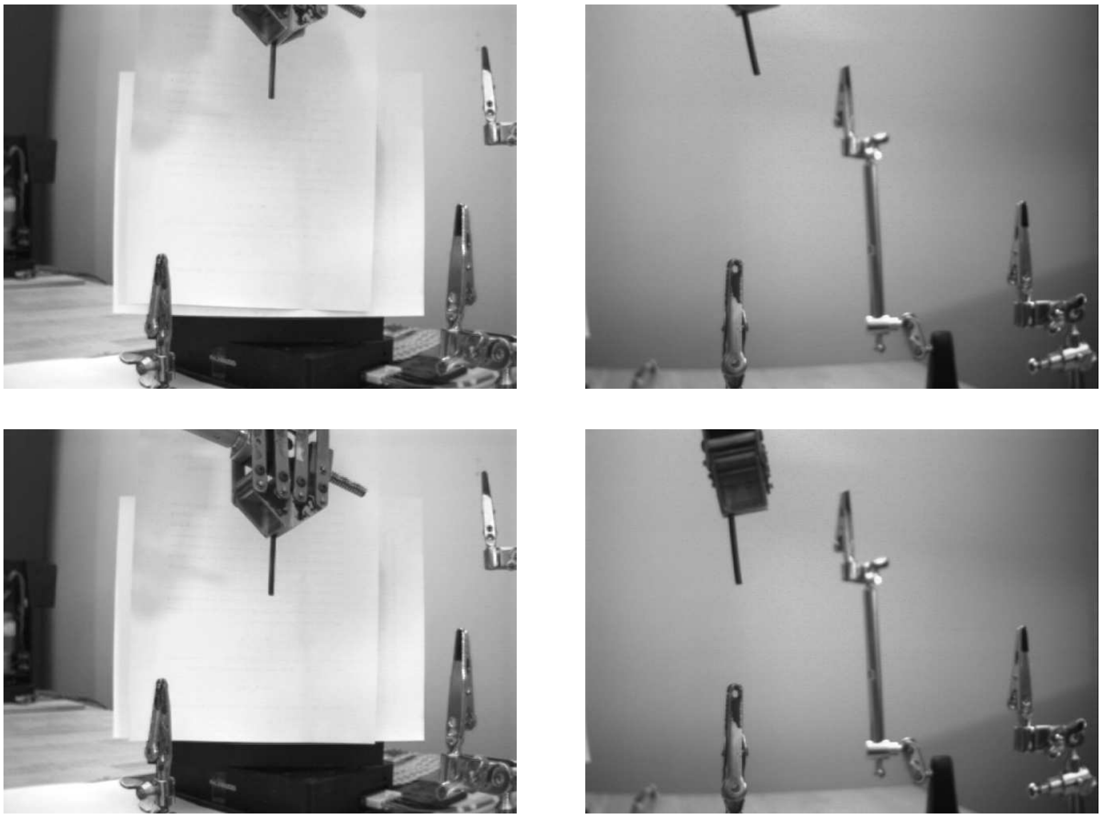


Figure 8.7: T_{copl} **verified**: (**top**) The initial configuration of a task in which a manipulated point is to be made coplanar with three stationary points. The robot-controlled feature point lies approximately 100mm from the destination plane. (**bottom**) The final configuration in which the manipulated point settles within 4mm of its goal.

Thus, the image of the feature $\mathbf{f}$ under the camera C is identical to the image of the feature $\mathbf{g}$ under the camera whose 3×4 full-rank transformation is $C\mathbf{f}\begin{bmatrix} \alpha_1 & 0 & 0 & 0 \\ 0 & \alpha_2 & 0 & 0 \\ 0 & 0 & \alpha_3 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}\mathbf{G}^{-1}$. Put another way, for any observation of four coplanar points, there exists a projective camera that images four noncoplanar points in exactly the same way. As this result holds for any number of observing cameras, without weak calibration coplanarity is impossible to determine.

The encoding E_{copl} thus depends more fundamentally on the weak calibration of the observing cameras than E_{col}. Indeed, the verifying encoding E_{copl} of Equation (6.19)

$$E_{\text{copl}}(\mathbf{y}) = \sum_{i=1}^{3} d_i^{2\prime} \mathbf{F} d_i^1 + \sum_{i,j} E_{\text{col}}(y_i, y_j, d_{ij}) \tag{8.4}$$

depends intimately on the stereo rig's fundamental matrix $\mathbf{F}$.

Using this encoding from an initial configuration shown at the top of Figure 8.7, the online-estimation control strategy of Equation (2.12) directed the manipulated point into the final state depicted below. Though difficult to ascertain in those pictures, the system reduces the task error from over 100mm to less than 4mm in the direction perpendicular to the target plane.

Metrical Positioning: T_{cr} and T_{midpt} Like T_{copl}, metrical positioning tasks such as T_{cr} and T_{midpt} require calibration estimates for the observing cameras. Because the collinearity task demonstrated the achievement of a cross-ratio constraint, this section focuses on the role of reconstruction in accomplishing T_{midpt}, the midpoint-positioning task.

Because $T_{\text{midpt}} \in \mathcal{T}_{\text{aff}}$, midpoint positioning requires at least affine calibration of the observing stereo rig. Figure 8.8 shows the calibration object used to obtain that affine information. The software packages Susan [106] and XVision [33] performed edge extraction and corner detection from the images; Tsai's calibration algorithm [113] used the relative Cartesian positions of each fiducial point to estimate the parameters of the imaging system. Although this algorithm provides a full metrical reconstruction of the cameras' internal and external parameters with respect to the calibration object, only the affine information — the matrix $\mathbf{A}$ from Equation (2.10) — is used for manipulator control.

With the computed calibration and Equations (2.9) and (2.10), the system re-

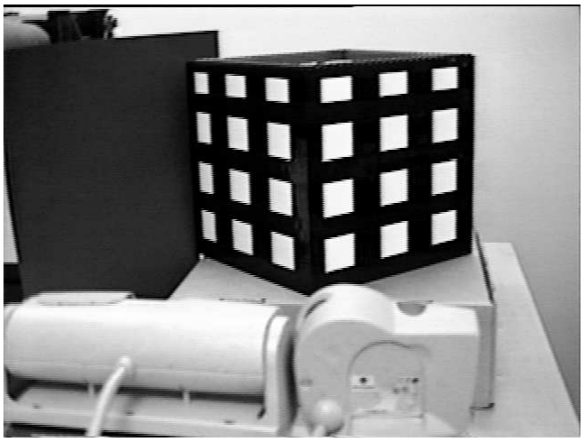


Figure 8.8: An overview of the calibration object used to obtain the affine calibration of the cameras observing from the foreground. The ninety-six corners visible on the object comprise a set of fiducial points with known relative geometry. Tsai's algorithm [113] was used to estimate the rig's affine calibration.

constructs an affine representation of the left and right markings on a ruler. It then uses a controller identical to that of Equation (8.1) to position the manipulated endpoint to the computed midpoint of those two visible markings. The initial and final configurations appear in Figure 8.9. With the stereo rig's baseline of 30mm, focal lengths of 11mm, and at a depth of 420mm, the system positioned the tooltip 8mm from the actual midpoint. Views of the accuracy obtained appear in Figure 8.10.

This result falls in line with work, such as [54, 93], that demonstrates the sensitivity of metrical positioning to small inaccuracies in calibration estimates. Because the accuracy of affine (or similarity-based) reconstruction and the resulting setpoints contribute to the task-space error, direct image feedback without reconstruction will avoid those errors and produce more accurate results. The tasks of Figures 8.1 and 8.9 have comparable system parameters and identical control laws. Yet, the former leads to a final task-space error on the order of a millimeter, rather than the centimeter in the latter case.

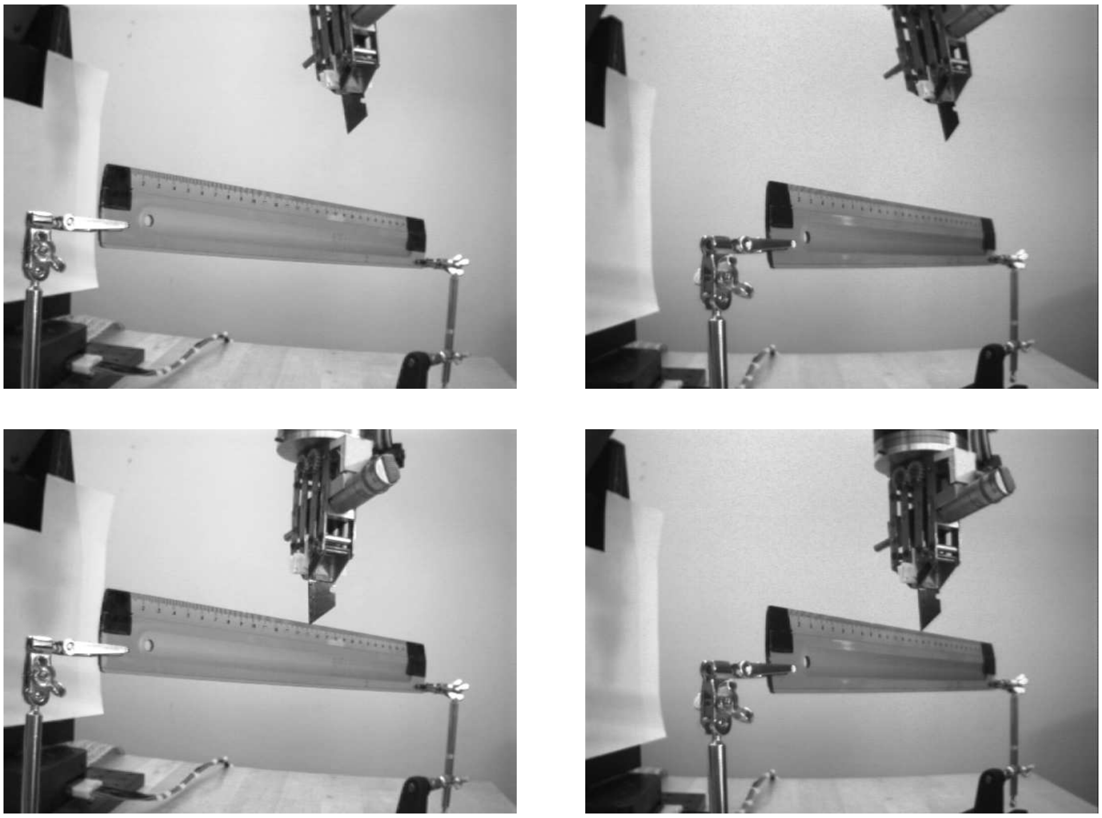


Figure 8.9: T_{midpt} **verified**: (**top**) The initial configuration of a task in which a manipulated point is to coincide with the center of the two marked stationary points. The robot-controlled feature point lies approximately 150mm from its destination. (**bottom**) The final configuration in which the manipulated point settles within a centimeter of its goal.

8.3 Implementing composite tasks

This section presents the implementation details associated with the six operators of $\mathcal{O}_6$. When combined with the previous section's primitives, these examples form a complete instantiation of the task-specification languages of Chapter 6.

Conjunction and Disjunction The conjunction operator expresses an intent to satisfy several task constraints simultaneously. As such, it is ubiquitous and — in a sense — appears even within the implementation of the coincidence and collinearity primitives. For instance, Equation (8.1)'s implementation of T_{pp} uses conjunction by

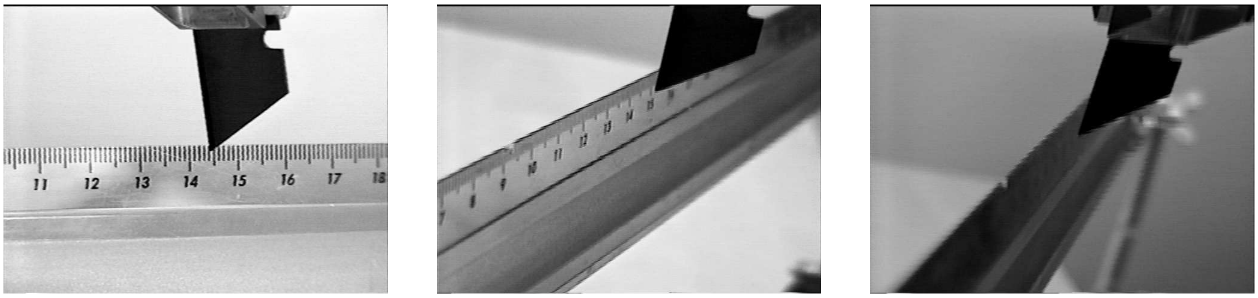


Figure 8.10: T_{midpt}**'s accuracy**: Three views of the tooltip's final configuration. The actual midpoint is the top of the 15cm mark on the ruler. In this case, the final positioning error is approximately 8mm.

gathering one-dimensional error signals into a single vector-valued error: $E_{\mathrm{pp}}(\mathbf{y}) = \begin{bmatrix} y_1 - y_1^* \\ y_2 - y_2^* \\ y_3 - y_3^* \\ y_4 - y_4^* \end{bmatrix}$, with each y_i representing a single image coordinate and each y_i^* a goal coordinate.

Disjunctive constraints, on the other hand, do not so readily translate into control equations. In particular, because conjunctions are so common, a practical implementation of disjunction must handle the case in which the two component disjuncts in a task are themselves conjunctions, or vectors, possibly with distinct numbers of constraints. For example, consider two point-to-point primitives, each represented as the conjunction of four one-dimensional error signals, $E_1 = \begin{bmatrix} e_1 \\ e_2 \\ e_3 \\ e_4 \end{bmatrix}$ and $E_2 = \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \epsilon_3 \\ \epsilon_4 \end{bmatrix}$. The one-dimensional disjunction of these two encodings:

$$E_1^t E_1 \cdot E_2^t E_2$$

does provide a definition of image error that will verify the underlying task. Yet this error function is difficult to use to control the three degrees of freedom necessary to achieve a point-coincidence constraint.

An alternative encoding exploits the fact that a target configuration will occur only when all of the e_i and ϵ_i vanish. Instead of combining these components into a single, complicated real-valued error function, a "cross product" vector of constraints will also verify the disjunctive task. More precisely, the vector-valued encoding

$$E_1 \vee E_2 \quad = \quad [\, e_i \cdot \epsilon_j \,] \quad = 0 \quad , \tag{8.5}$$

which has sixteen components as i and j take values in $\{1, 2, 3, 4\}$, also verifies the disjunctive task. It has the advantage of overspecifying the number of degrees of freedom available in a typical manipulation system and generalizes to disjuncts with arbitrarily many components.

Figure 8.11 demonstrates the control produced with the encoding of Equation (8.5). The system converges to one or the other set point from a variety of starting configurations, though the paths taken to reach those set points are not straightforward to characterize. In particular, the system did not always converge to the goal closest to the starting point, nor did it monotonically approach one or the other targets. Figures 8.11 and 8.12 sample the trajectory of several such system runs.

Expansion, Contraction, and Permutation As Chapter 7 attests, these three operators do not affect the set of configurations a system can attain. Instead, they provide the mathematical flexibility required to demonstrate that conjunction, disjunction, and negation, along with an appropriate set of primitives, span the set of verifiable tasks for the sets of camera models considered in this work. All of the implemented tasks in this chapter depend on the maintenance of consistent label names and consistent feature point cardinality provided by the expansion, contraction, and permutation operators.

Negation From a theoretical standpoint the negation operator's failure to preserve the differentiability or continuity of encodings complicates the automatic derivation of controllers from task specifications. In practice, however, the lack of a continuous encoding for tasks involving negation does not undermine Chapter 6's framework for the following reasons:

- A large class of realistic tasks do not involve the negation operator at all. These *positively-defined* tasks, in fact, comprise many of the constraints of interest in everyday applications. The vast majority of the example tasks found in the literature, whether involving contrived patterns or unadorned objects, involve only compositions of unnegated primitives.

- Approximation strategies can be used to an arbitrary level of precision. Discontinuous task encodings can be smoothed through convolution with a suitable kernel (e.g., a Gaussian). The spatial extent of that kernel can be tuned to offer as sharp a distinction between satisfying and unsatisfying feature configurations as desired.

- Often the intent of a negated task constraint is not merely to exclude certain poses from the goal configuration, but to exclude those poses from any possible trajectory of the image features. For example, negated task constraints may stipulate that the system avoid image-based or mechanical singularities, as in [65, 66]. Such trajectory constraints are not within the scope of the compositional approach to specifying goals presented here, yet a "trajectory-conscious" approach does easily and naturally extend this framework.

In particular, the approximation strategies noted above effectively penalize proximity to certain configurations and allow a task programmer to constrain the path taken in achieving a task. Figure 8.13 shows samples along the image trajectories of the conjunction of a point-to-point goal (to the left) with the negation of another point-to-point constraint (the the right); the following encoding provided the control:

$$\begin{aligned} E(\mathbf{y}) &= E^2_{\mathrm{ppLeft}}(\mathbf{y}) + E^2_{\neg\mathrm{ppRight}} \ , \\ \text{with} \quad E_{\neg\mathrm{ppRight}} &= \begin{cases} \frac{1}{|\mathbf{y}-\mathbf{y}_{\mathrm{Right}}|} & \text{when} \ \ |\mathbf{y}-\mathbf{y}_{\mathrm{Right}}| < \tau \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

This encoding uses a term inversely proportional to the image distance from the point to be avoided. The threshhold τ limits this term's effect to a small region around the right-hand point, thereby ensuring that the task-space constraint is satisfied when the image-space constraint becomes zero. The resulting path avoids passing too near the point to be avoided, even when the control would otherwise do so (Figure 8.13). Work such as [12, 87] derives and demonstrates the use of potential functions that guarantee the achievement of such objectives.

8.3.1 The wrench-alignment task

As an example of the application of the primitives and operators of Sections 8.2 and 8.3, consider the implementation of the wrench-positioning task initially suggested by Figure 7.1. While Chapter 7 analyzes this task's specification in detail, Figure 8.14's snapshots show the execution of the constraints of Equation (7.2).

Because Figure 8.14's wrench-alignment task requires the control of six degrees of task-space freedom and the monitoring of sixteen degrees of image-space freedom (four points in two images), ensuring a stable and convergent control required more careful modeling than the simpler tasks above. In particular, for several values of r, $\mathbf{J}_r$ was estimated offline; the system then used those models in conjunction with an online update as the manipulator approached the appropriate part of its configuration space.

From a specification point of view the programming of this composite task requires no additional machinery or consideration beyond that of the simpler applications above. The specification of Equation (7.2) generates the verifying encoding

$$\begin{aligned} E_{\text{wrench}}(\mathbf{y}) &= E_{\text{pp}}(y_2, y_5) \wedge E_{\text{pp}}(y_4, y_7) \;\wedge\; E_{\text{col}}(y_1, y_2, y_6) \;\wedge\; E_{\text{col}}(y_3, y_4, y_8) \qquad (8.6) \\ &= \begin{bmatrix} y_2 - y_5 \\ y_4 - y_7 \\ y_6 \cdot (y_1 \times y_2) \\ y_8 \cdot (y_3 \times y_4) \end{bmatrix} , \end{aligned}$$

where the dot products suffice for the collinearity constraints because the destination lines are not near epipolar planes. A control strategy analogous to Equation 8.1 used this encoding to achieve the results in Figure 8.14. In effect, the building blocks of task primitives and their combining operators modularize the construction of hand/eye systems, separating concerns about task description from those of task execution.

MPEG videos of both the visual tracking and the execution of the wrench-alignment and the other tasks described in theis chapter are available at `www.cs.hmc.edu/~dodds/thesis.html` .

8.4 Task Accuracy

The languages of Chapter 6 guarantee that, in the absence of noise, satisfying a task's verifying encoding will also satisfy the underlying physical constraint. All real systems, however, operate with noise arising from a variety of sources: inaccurate image processing, limited sensor resolution, and errors in modeling and calibrating cameras and manipulators. These factors beg the question of how noise ultimately affects the accuracy of task-space constraints. This question has received considerable attention within the visual servoing community; here we relate that work to this thesis's results.

Estimating the accuracy of a hand/eye system's final configuration requires a model for the uncertainty of the observing cameras' poses. Three possibilities for such uncertainty models include (1) knowledge that a stereo rig falls into one of the sets $\mathcal{C}_{\text{bij}}, \mathcal{C}_{\text{proj}}, \mathcal{C}_{\text{aff}}, \mathcal{C}_{\text{sim}}$, as defined in Chapter 2, (2) knowledge of interval bounds on certain parameters of a stereo rig's calibration, and (3) an approximate (full) calibration used to create a linear relationship between image errors and task-space errors. We consider each of these uncertainty models in turn.

The first possibility is the uncertainty model considered in this work – that the observing stereo rig has a known weak, affine, or similarity calibration. Because none of these levels of certainty bounds the system's "depth," the distance from cameras to a task's feature points, image errors of any size could, in theory, lead to arbitrarily large errors in final task-space configuration. This possibility is inherent in any system which does not operate at a fixed scale; in essence it amounts to setting an arbitrary unit for error measurement.

In practice the scale of the usable robot workspace will limit the distance at which any task is observed. Upper and lower bounds will generally be known for other system parameters, such as focal length, baseline, relative orientation, and pixel resolution. With a system's calibration guaranteed to fall within specified intervals, conservative, worst-case error bounds replace the potential for arbitrarily large task error.

Such worst-case error is computed by assuming the feasible camera geometry which maximizes task space error. For point-coincidence tasks, by considering the greatest possible distance (d) to the task, the shortest possible baseline (b), and the shortest allowed focal lengths (f), the resulting model will overestimate the actual system's sensitivity to image error. Using values of $d = 600mm$, $b = 50mm$, and $f = 10mm$ with a pixel size of $13\mu m$, for instance, yields a worst-case task-space error of 20mm (in depth) for an image uncertainty of one pixel. In the situation depicted by Figure 8.1 the values produced during tracking a stationary corner fell within ± 0.2 pixels of the estimated image position. This image-space error ensures a task-space error of less than 4mm, certainly a conservative estimate of the error seen in Figure 8.2.

Naturally, knowing d and other system parameters allows for more precise estimates than the above worst-case analysis. To this end an approximation of a full system calibration can be used to model the relationship between image and task error. Linearizing from the approximate calibration, Gaussian-distributed image errors propagate as Gaussian task-space errors. Such techniques provide a measure of system sensitivity (termed both "resolvability" and "perceptibility"). These measures are suited to handle both symmetric and anisotropic estimates of image error, and they can be used during task execution to maintain visual tracking and avoid control singularities. Research such as [76, 77, 103] has explored this strategy in depth; [93] estimates sensitivities to calibration errors experimentally.

Such work applies directly to the tasks that the specification languages presented here can describe. Within the control framework of image-based visual servoing Chapter 6's modular approach to specifying tasks supports task accuracy in two ways. First, by categorizing the level of camera certainty necessary to verify a task-space constraint, the languages offer only capabilities that their corresponding camera models can handle. Second, since each automatically generated encoding verifies its underlying task, whatever task error does arise is guaranteed not to come from either the task specification or its representation as an image-based constraint.

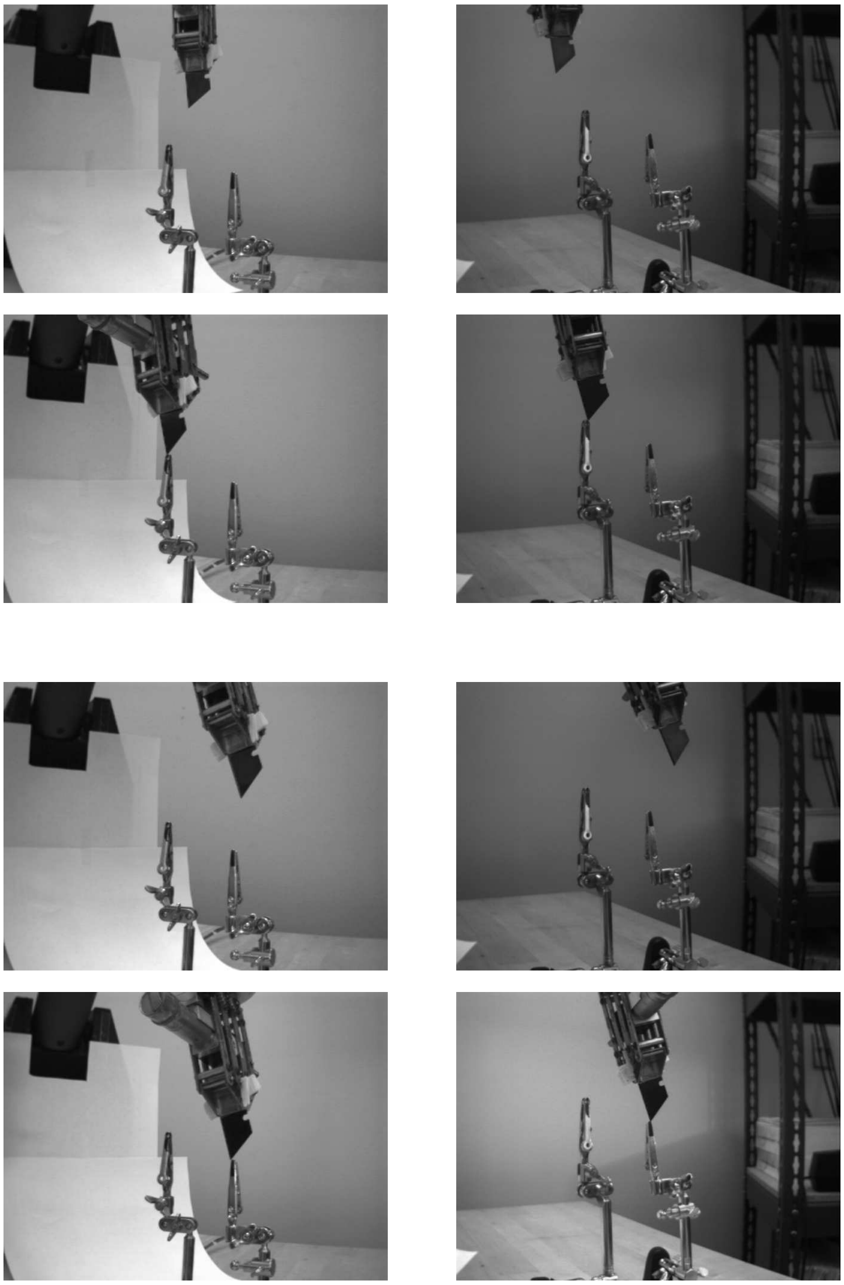


Figure 8.11: **Performing** $T_{\mathrm{pp1}} \vee T_{\mathrm{pp2}}$: (**Top**) The initial and final configurations (viewed from each camera) of the tooltip and the two target points, both of which are suitable destinations. Because the control commences with the manipulated point closer to the left-hand setpoint, ultimately the system converges there. (**Bottom**) Another run of the same task from a right-biased initial configuration. Figure 8.12 demonstrates that the paths generated in these kinds of disjunctive tasks do not always converge in a straightforward manner.

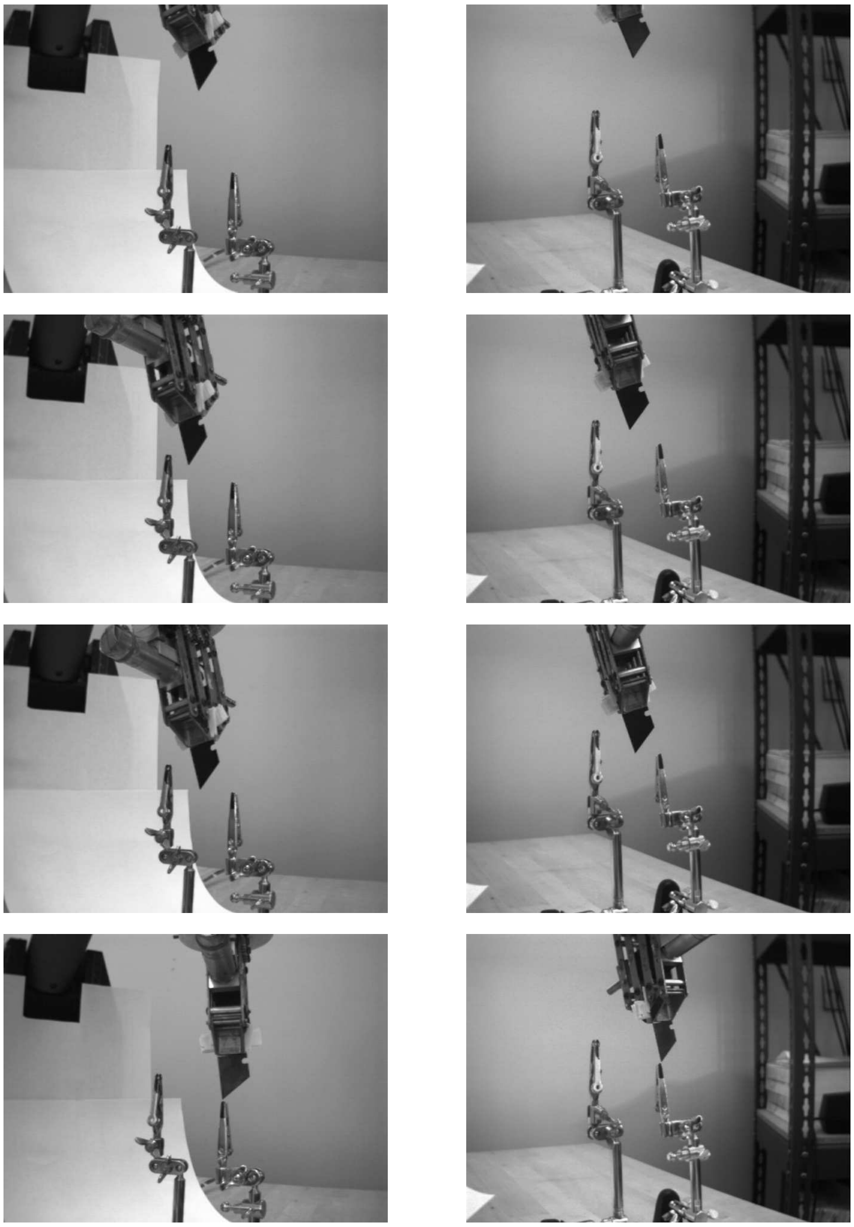


Figure 8.12: **Samples along a trajectory in $T_{\mathrm{pp1}} \vee T_{\mathrm{pp2}}$**: The manipulated point does not necessarily follow a path that consistently approaches a single setpoint in either image or task space. This figure demonstrates four snapshots from each camera along a single convergent trajectory.

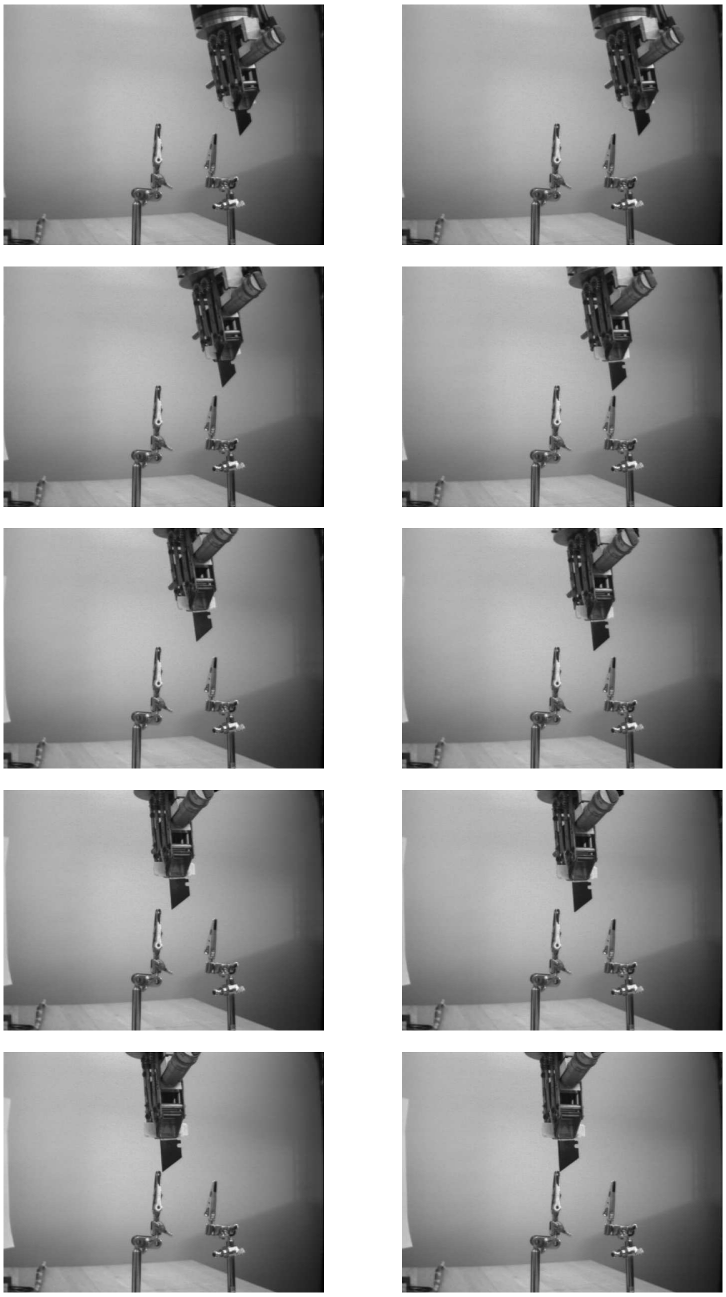


Figure 8.13: **Incorporating negation** (**Right**) Snapshots (proceeding from top to bottom) from the execution of a point-to-point constraint in which a manipulated point is made coincident with the left-hand set point in each image. (**Left**) Adding a term that penalizes proximity to the right-hand feature point produces a more cautious trajectory. In general such penalty terms better express the desired semantics of negated task constraints than the mathematically more precise but physically less constraining definition of $\neg E$.

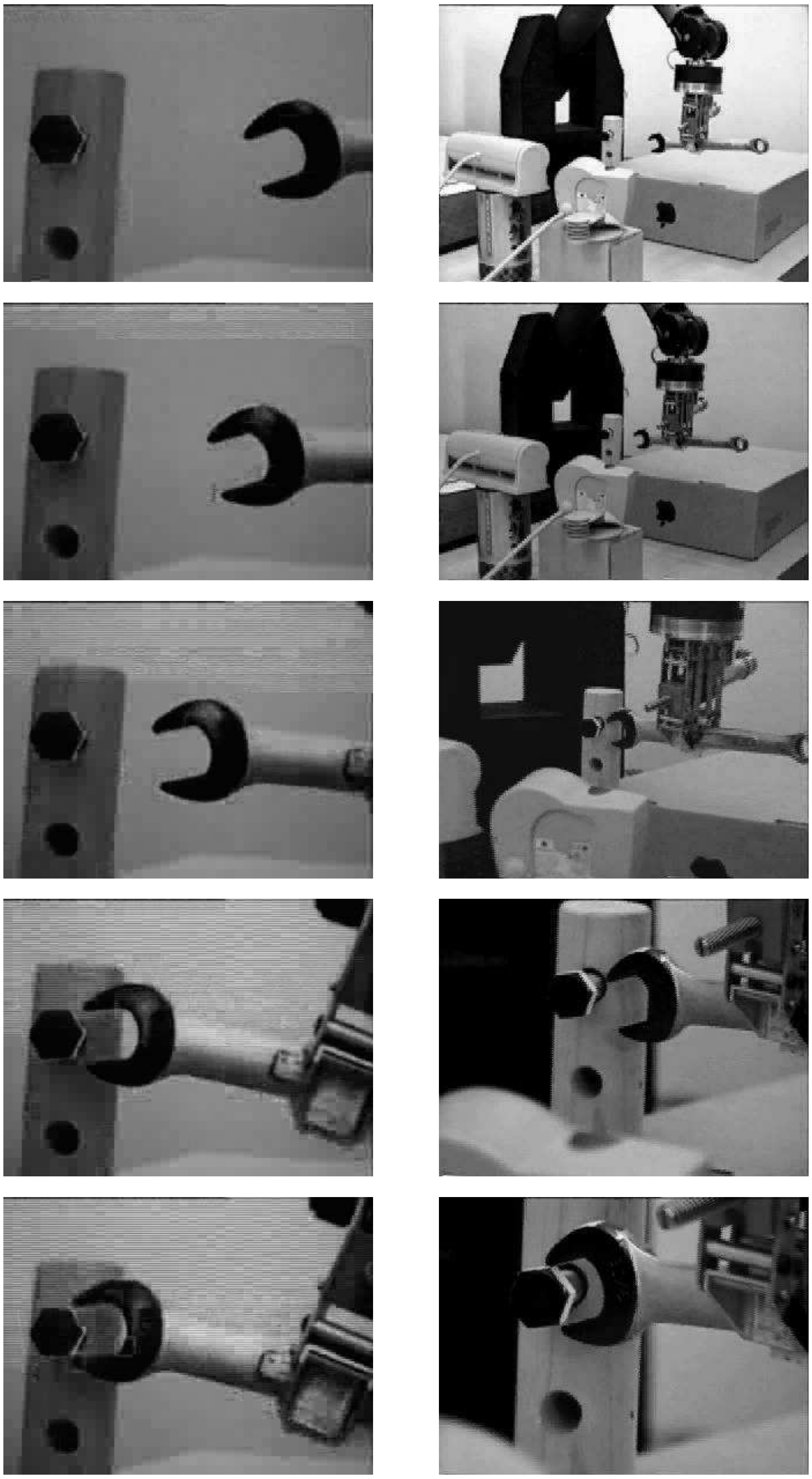


Figure 8.14: Snapshots of a wrench-alignment task from two viewpoints: (**Left**) one of the observing camera's views; (**Right**) a viewpoint distinct from the cameras involved in task control, zooming as the wrench approaches its destination pose.

Chapter 9

Conclusions

Broadly construed, this thesis assesses the impact of a fundamental characteristic of robotic systems — namely, the ability of its sensors to distinguish among different world states. The consequences of this ability resonate across a spectrum of theoretical and practical system considerations. By making it a touchstone for design, resulting systems recognize their inherent limitations and fully exploit available capabilities.

In the progression from theory to practice the idea of a task — or at least the extent to which that idea is fleshed out — evolves along with the system for specifying and executing that task. At the outset a task simply represents a target set of relative object poses: a purely geometric constraint in the 3d world. In order to use vision to decide when such a task is accomplished, families of relative poses indistinguishable to the system's cameras must be either wholly acceptable or wholly unacceptable in unison.

Decidable tasks, then, are guaranteed to have a set of distinguishing relative feature poses in image space as well as in 3d task space. As long as the combining operators preserve decidability, this dual existence extends to compositions of decidable tasks. Thus, whenever a number of 3d constraints specifies a desired object pose, so does a parallel set of image-based constraints.

Achieving these image constraints requires additional assurances of smoothness, along with on- or off-line estimates of how a robot's actions effect image change. At this level of implementation tasks become differentiable error functions of image data, a view often taken as a starting point within the field of visual servo control. Building these error functions from the fundamental objective of controlling the configuration of the physical world, however, provides a framework which neither undercuts system capabilities nor overstates them. The resulting languages for specifying tasks represent only — and all of — what can be done.

9.1 Open Issues

Though complete in its descriptions of the families of tasks considered, this work leaves a number of lines of inquiry only partially explored.

Additional Visual Features The assumption that observable features are image points, while common, begs the question of how easily this thesis's framework incorporates other types of visual information. The wire-cutting example of Figure 8.4 shows that some uses of line features, e.g., a point-to-line constraint, emerge directly from tasks such as T_{col}. The structure of projective transformations — that they preserve collinearity — might suggest that tasks involving line-segment features, such as the point-to-line constraint, are in $\mathcal{T}_{\text{proj}}$; in fact, weakly calibrated cameras can not, in general, determine when a point and line feature coincide. When both the point and the line lie in the observing cameras' epipolar plane, image information can not verify their coincidence (or noncoincidence). A complete characterization of verifiable tasks involving line-segment features remains an open extension of the results of Chapter 6.

Other approaches to visual servoing have used photometric, rather than geometric, features. Aligning an image patch into some known, desired appearance can lead to very accurate positioning. Although it is possible to regard such tasks as large compositions of point-to-point constriants by identifying points with pixels, the analogy ignores substantial differences between the two techniques. Performing photometric alignment requires that the system has seen (or has rendered a guess at) the goal image, whereas geometric positioning needs a less exacting description of the goal configuration in terms of either coincidence or metrical constraints. The high dimensionality of photometric data also leads to control strategies in which image errors are projected to a lower-dimesional representation before applying control, rather than in the control computation itself. In a sense, the geometric approach presented here does simply perform image-matching, but in doing so it separates the task-specific image constructions needed to obtain feature points from a more broadly applicable set of task-independent hand/eye skills.

Characterizing Other Camera Models The four levels of calibration represented by $\mathcal{C}_{\mathrm{bij}}$, $\mathcal{C}_{\mathrm{proj}}$, $\mathcal{C}_{\mathrm{aff}}$, and $\mathcal{C}_{\mathrm{persp}}$ occupy a prominent place within the vision community because they arise naturally from observations of the world (correspondences) and varying strata of internal knowledge (calibration). Yet they comprise only four possibilities within a continuum of imaginable levels of calibration. What can we say about the other sets of camera models in that continuum? In some cases, such as with $\mathcal{C}_{\mathrm{uncal}}$, the four camera models considered here provide upper and lower bounds on distinguishable feature configurations. In other cases, an arbitrary stereo rig will require additional point matches before it can be compared directly to the capabilities of these four.

The example of $\mathcal{C}_{\text{uncal}}$ suggests that it is very unlikely that an arbitrarily chosen set of camera models will admit the clean characterizations that $\mathcal{C}_{\text{bij}}$, $\mathcal{C}_{\text{proj}}$, $\mathcal{C}_{\text{aff}}$, and $\mathcal{C}_{\text{persp}}$ do. The group properties of those four sets' feature equivalence classes seem fundamental to their concise description. Ambiguity in $\mathcal{C}_{\text{uncal}}$, on the other hand, does not even obey transitivity: there exist features $\mathbf{f}_1, \mathbf{f}_2$, and $\mathbf{f}_3$ such that uncalibrated cameras can distinguish $\mathbf{f}_1$ from $\mathbf{f}_3$, but neither between $\mathbf{f}_1$ and $\mathbf{f}_2$ nor $\mathbf{f}_2$ and $\mathbf{f}_3$. This holds, for instance, when $\mathbf{f}_1$ consists of four collinear points, $\mathbf{f}_2$ of four coplanar (but not collinear) points, and $\mathbf{f}_3$ of four points in general position. This lack of transitivity casts doubt on the existence of a characterization for $\mathcal{T}_{\text{uncal}}$ — and for the vast majority of the possible sets of camera models — as concise as that of $\mathcal{T}_{\text{proj}}$.

Other Task-specification Languages The languages presented in Chapter 6 are by no means unique, as the unwieldy example of elementary-tasks and the disjunction operator demonstrates. Even with the task primitives presented, however, the operators in $\mathcal{O}_6$ can be pared down: DeMorgan's laws permit the definition of either conjunction or disjunction in terms of the other and negation. Moreover, additional operators allow for more concise languages in the sense that fewer primitives are needed to span the set of performable tasks. For example, consider a feature-construction operator that allows the creation of points at the intersection of any two lines visible by cameras in $\mathcal{C}_{\text{proj}}$. With the addition of this operator, the one-dimensional cross-ratio-positioning primitive $T_{\text{cr}\alpha}$ along with the subspace-coincidence primitives spans the tasks in $\mathcal{T}_{\text{proj}}$. Similar construction operators can be defined for the other classes of tasks analyzed here. Thus, the operators and primitives of Chapter 6 do not constitute minimal specification languages, but proposals for intuitive and useful ones, as supported by the examples of Chapters 7 and 8.

9.2 Perspective

Stepping back from its details, this work offers an analysis of vision-based tasks at a decidedly low level of abstraction. It poses an organizational framework for determining when available visual information suffices to perform a task and how to leverage image data to do so. The philosophy underlying these low-level results is that of a modular and skill-based approach to hand-eye coordination that relieves a task programmer from the burden of explicitly formulating control laws. In the future, however, systems should further unencumber programmers of visual tasks.

The current state of specifying vision-based robotic tasks resembles that of computational tasks several decades ago. At that time, human programmers decomposed complex computations into sequences of very low-level steps — the machine-language primitives supplied by the architecture in use. The burden of extracting common functionality from different pieces of code or creating useful abstractions fell entirely into human hands. Feature-based methods for visual task specification present programmers with a similar, low-level interface. Transferring the work of low-level specification from human to machine will require a thorough understanding of these basic skills.

Much as today's programming languages and user interfaces provide abstractions to support high-level descriptions of many basic computations, the future design of robot programming systems will allow a programmer to specify vision-based tasks in terms of objects to be manipulated, rather than their constituent features. In adressing and answering several fundamental questions about the capabilities of feature-based servoing systems, this work contributes to the foundation of such a design.

Bibliography

[1] P. Allen and K. Tarabanis. Sensor planning in computer vision. *IEEE Transactions on Robotics and Automation*, 11(1):96–105, 1995.

[2] P. Allen, A. Timcenko, B. Yoshimi, and P. Michelman. Automated tracking and grasping of a moving object with a robitc hand-eye system. *IEEE Transactions on Robotics and Automation*, 9(2):152–165, 1993.

[3] A. P. Ambler and R. J. Popplestone. Inferring the positions of bodies from specified spatial relationships. *Artificial Intelligence*, 6(2):157–174, 1975.

[4] R. L. Anderson. Dynamic sensing in a ping-pong playing robot. *IEEE Transactions on Robotics and Automation*, 5(6):723–739, December 1989.

[5] D. F. Baldwin, T. E. Abell, M.-C. Lui, T. L. De Fazio, and D. E. Whitney. An integrated computer aid for generating and evaluating assembly sequences for mechanical products. *IEEE Trans. on Robotics and Automation*, 7(1):78–94, 1991.

[6] F. Berry, P. Martinet, and J. Gallice. Trajectory generation by visual servoing. In *Proc., Internationl Conference on Intelligent Robots and Systems*, pages 1066–1072, 1997.

[7] A. Castano and S. Hutchinson. Visual compliance: Task-directed visual servo control. *IEEE Transactions on Robotics and Automation*, 10(3):334–342, June 1994.

[8] F. Chaumette. Potential problems of stability and convergence in image-based and position-based visual servoing. *Lecture Notes in Control and Information Sciences*, 237:66–78, 1998.

[9] F. Chaumette, E. Malis, and S. Boudet. 2d 1/2 visual servoing with respect to a planar object. In *Workshop on New Trends in Image-Based Robot Servoing*, pages 45–52, 1997.

[10] F. Chaumette, P. Rives, and B. Espiau. Classification and realization of the different vision-based tasks. In K. Hashimoto, editor, *Visual Servoing*, pages 199–228. World Scientific, 1994.

[11] P. I. Corke. Visual control of robot manipulators—a review. In K. Hashimoto, editor, *Visual Servoing*, pages 1–32. World Scientific, 1994.

[12] N. J. Cowan and D. E. Koditschek. Planar image based visual servoing as a navigation problem. In *Proc., Internationl Conference on Robotics and Automation*, pages 611–617, 1999.

[13] A. Cretual and F. Chaumette. Image-based visual servoing by integration of dynamic measurements. In *Proc., Internationl Conference on Robotics and Automation*, pages 1994–2001, 1998.

[14] G. Csurka, D. Demirdjian, A. Ruf, and R. Horaud. Closed-form solutions for the euclidean calibration of a stereo rig. In *Proc., European Conference on Computer Vision*, pages 426–442, 1998.

[15] F. Devernay and O. D. Faugeras. From projective to euclidean reconstruction. In *Proc., Computer Vision and Pattern Recognition*, pages 264–269, 1996.

[16] E. Dickmanns. Dynamic vision merging control engineering and AI methods. In *Lecture Notes in Control and Information Sciences*, volume 237, pages 210–229, 1998.

[17] Z. Dodds, M. Jagersand, G. D. Hager, and K. Toyama. A hierarchical vision architecture for robotic manipulation tasks. *Lecture Notes in Computer Science: First Int. Conf. of Computer Vision Systems*, 1542:312–330, 1999.

[18] B. Espiau, F. Chaumette, and P. Rives. A new approach to visual servoing in robotics. *IEEE Transactions on Robotics and Automation*, 8:313–326, 1992.

[19] O. D. Faugeras. What can be seen in three dimensions with an uncalibrated stereo rig? In *Proc., European Conference on Computer Vision*, pages 563–578, 1992.

[20] O. D. Faugeras. *Three-Dimensional Computer Vision.* MIT Press, 1993.

[21] O. D. Faugeras. Stratification of 3D vision: Projective, affine, and metric representations. *Journal of the Optical Society of America, A*, 12(7):465–484, 1995.

[22] O. D. Faugeras and S. Maybank. Motion from point matches: Multiplicity of solutions. *International Journal of Computer Vision*, 4(3):225–246, 1990.

[23] J. T. Feddema, C. S. G. Lee, and O. R. Mitchell. Feature-based visual servoing of robotic systems. In K. Hashimoto, editor, *Visual Servoing*, pages 105–138. World Scientific, 1994.

[24] M. Fleck. Perspective projection: The wrong imaging model. Technical Report TR 95-01, University of Iowa, Iowa City, IA, 1994.

[25] A. Fox and S. Hutchinson. Exploiting visual constraints in the synthesis of uncertainty-tolerant motion plans. *IEEE Transactions on Robotics and Automation*, 11:56–71, 1995.

[26] Agence France-Presse. Photograph accompanying "It's Up, It's Good: Houston Sends Knicks to Round 2". *New York Times*, CXLVIII(51,525):D1, 5/17/1999.

[27] B. Ghosh, T. Tarn, N. Xi, Z. Yu, and D. Xiao. Calibration-free visually controlled manipulation of parts in a robotic manufacturing workcell. In *Proc., Internationl Conference on Robotics and Automation*, pages 3197–3202, 1996.

[28] E. J. Gonzalez-Galvan, S. B. Skaar, U. A. Korde, and W. Z. Chen. Application of a precision enhancing measure in 3d rigid-body positioning using camera-space manipualtion. *International Journal of Robotics Research*, 16(2):240–257, 1997.

[29] G. D. Hager. Calibration-free visual control using projective invariance. In *Proc., Internationl Conference on Computer Vision*, pages 1009–1015, 1995.

[30] G. D. Hager. A modular system for robust positioning using feedback fromstereo vision. *IEEE Transactions on Robotics and Automation*, 13(4):582–595, August 1997.

[31] G. D. Hager, W-C. Chang, and A. S. Morse. Robot hand-eye coordination based on stereo vision. *IEEE Control Systems Magazine*, 15(1):30–39, February 1995.

[32] G. D. Hager and Z. Dodds. A projective framework for constructing accurate hand-eye systems. In *Proc. IROS Workshop on New Trends in Image-based Robot Servoing*, pages 71–82, 1997.

[33] G. D. Hager and K. Toyama. XVision: Combining image warping and geometric constraints for fast visual tracking. In *Proc., European Conference on Computer Vision*, pages 507–517. Springer Verlag, 1996.

[34] R. Hartley. In defence of the 8-point algorithm. In *Proc., Internationl Conference on Computer Vision*, pages 1064–1070, 1995.

[35] R. Hartley, R. Gupta, and T. Chang. Stereo from uncalibrated cameras. In *Proc., Computer Vision and Pattern Recognition*, pages 761–764, 1992.

[36] R. Hartley and P. Sturm. Triangulation. In *Proc. of the ARPA Image Understanding Workshop*, pages 957–966, November 1994.

[37] K. Hashimoto and T. Noritsugu. Performance and sensitivity in visual servoing. In *Proc., Internationl Conference on Robotics and Automation*, pages 2321–2326, 1998.

[38] J. Heikkila and O. Silven. Calibration procedure for short focal length off-the-shelf ccd cameras. In *Proc., International Conference on Pattern Recogition*, pages 166–170, 1996.

[39] J. Hespanha, Z. Dodds, G. D. Hager, and A. S. Morse. What tasks are performable with a weakly calibrated stereo rig? In *Lecture Notes in Control and Information Sciences*, volume 237, pages 79–89, 1998.

[40] J. Hespanha, Z.Dodds, G. D. Hager, and A. S. Morse. What can be done with an uncalibrated stereo system? In D. J. Kriegman, G. D. Hager, and A. S. Morse, editors, *The Confluence of Vision and Control*, number 237 in Lecture Notes in Control and Information Sciences, pages 79–89. Springer-Verlag, London, 1998.

[41] J. P. Hespanha. *Logic-Based Switching Algorithms in Control*. PhD thesis, Yale University, New Haven, CT, 1998.

[42] N. Hollinghurst and R. Cipolla. Uncalibrated stereo hand eye coordination. *Image and Vision Computing*, 12(3):187–192, 1994.

[43] R. Horaud and G. Csurka. Self-calibration and euclidean reconstruction using motions of a stereo rig. In *Proc., Internationl Conference on Computer Vision*, pages 96–103, 1998.

[44] B. K. P. Horn. Motion fields are hardly ever ambiguous. *International Journal of Computer Vision*, 1(3):263–278, 1987.

[45] I. Horswill. Polly: A vision-based artificial agent. In *Proc., American Conference on Artificial Intelligence*, pages 824–829, 1992.

[46] K. Hosoda and M. Asada. Versatile visual servoing without knowledge of true jacobian. In *Proc., Internationl Conference on Intelligent Robots and Systems*, pages 186–191. IEEE Computer Society Press, 1994.

[47] S. Hutchinson, G. D. Hager, and P. Corke. A tutorial on visual servo control. *IEEE Transactions on Robotics and Automation*, 12(5):651–670, 1996.

[48] M. Jagersand, O. Fuentes, and R. Nelson. Acquiring visual-motor models for precision manipulation with robot hands. In *Proc., European Conference on Computer Vision*, 1996.

[49] M. Jagersand, O. Fuentes, and R. Nelson. Experimental evaluation of uncalibrated visual servoing for precision manipulation. In *Proc., Internationl Conference on Robotics and Automation*, pages 2874–2880, 1997.

[50] M. Jagersand and R. Nelson. Visual task specification, planning, and control. In *Proc., IEEE International Symposium on Computer Vision*, pages 521–526, 1995.

[51] R. Joshi and A. Sanderson. Application of feature-based multi-view servoing for lamp filament alignment. In *IEEE Robotics and Automation Magazine*, pages 25–32, 1998.

[52] F. Keçeci, M. Tonko, H.-H. Nagel, and V. Gengenbach. Improving visually servoed disassembly operations by automatic camera placement. In *Proc., Internationl Conference on Robotics and Automation*, pages 2947–2952, 1998.

[53] J. Koenderink and A. Van Doorn. Affine structure from motion. *Journal of the Optical Society of America A*, 8(2):377–385, 1991.

[54] S. K. Kopparapu and P. Corke. The effect of measurement noise on intrinsic camera calibration parameters. In *Proc., Internationl Conference on Robotics and Automation*, pages 1281–1286, 1999.

[55] P. Krautgartner and M. Vincze. Performance evaluation of vision-based control tasks. In *Proc., Internationl Conference on Robotics and Automation*, pages 2315–2320, 1998.

[56] E. Kruppa. Zur ermittlung eines objektes aus zwei perspektiven mit innerer orientierung. *Sitz.-Ber. Akad. Wiss., Wien, Math. Naturw. Kl., Abt. IIa*, 122:1939–1948, 1913.

[57] P. Liang, J.-F. Lee, and S. Hackwood. A general framework for robot hand/eye coordination. In *Proc., Internationl Conference on Robotics and Automation*, pages 1082–1087, 1988.

[58] Y. Liu and R. J. Popplestone. Planning for assembly from solid models. In *Proc., Internationl Conference on Robotics and Automation*, 1989.

[59] H. C. Longuet-Higgins. A computer algorithm for reconstructing a scene from two projections. *Nature*, 293:133–135, 1981.

[60] T. Lozano-Perez, J. L. Jones, E. Mazer, P. A. O'Donnell, and W. E. L. Grimson. Handey: a robot system that recognizes, plans, and manipulates. In *Proc., Internationl Conference on Robotics and Automation*, pages 843–849, 1987.

[61] Q. Luong and O. D. Faugeras. The fundamental matrix: Theory, algorithms, and stability analysis. *International Journal of Computer Vision*, 17:43–75, 1996.

[62] E. Marchand. ViSP: A software environment for eye-in-hand visual servoing. In *Proc., Internationl Conference on Robotics and Automation*, pages 3224–3229, 1999.

[63] E. Marchand and P. Bouthemy. Robust real-time tracking using a 2d-3d model-based approach. In *Proc., Internationl Conference on Computer Vision*, to appear, 1999.

[64] E. Marchand and F. Chaumette. Active sensor placement for complete scene exploration and reconstruction. In *Proc., Internationl Conference on Robotics and Automation*, pages 743–751, 1997.

[65] E. Marchand, F. Chaumette, and A. Rizzo. Using the task function approach to avoid robot joint limits and kinematic singularities in visual servoing. In *Proc., Internationl Conference on Intelligent Robots and Systems*, volume 3, pages 1083–1090, 1996.

[66] E. Marchand and G. D. Hager. Dynamic sensor planning in visual servoing. In *Proc., Internationl Conference on Robotics and Automation*, pages 1988–1993, 1998.

[67] N. Maru, H. Kase, A. Nishikawa, and F. Miyazaki. Manipulator control by visual servoing with the stereo vision. In *Proc., Internationl Conference on Intelligent Robots and Systems*, pages 1866–1870. IEEE Computer Society Press, 1993.

[68] S. J. Maybank. The projective geometry of ambiguous surfaces. In *Proc. Royal Society of London, A*, volume 332, pages 1–47, 1990.

[69] H. G. McCain. A hierarchically controlled, sensory interactive robot in the automated manufacturing research facility. In *Proc., Internationl Conference on Robotics and Automation*, pages 931–939, 1985.

[70] H. Moravec. *Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover*. PhD thesis, Stanford University, 1980.

[71] J. D. Morrow and P. K. Khosla. Manipulation task primitives for composing robot skills. In *Proc., Internationl Conference on Robotics and Automation*, pages 3354–3359, 1997.

[72] J. L. Mundy and A. Zisserman. *Geometric Invariance in Computer Vision*. MIT Press, Cambridge, MA, 1992.

[73] R. Murray, Z. Li, and S. Sastry. *A Mathematical Introduction to Robotic Manipulation*. CRC Press, 1994.

[74] B. Nelson and P. Khosla. Vision resolvability for visually servoed manipulation. *Journal of Robotic Systems*, 13(2):75–93, 1996.

[75] B. Nelson and P. Khosla. Integrating force and vision feedback within virtual environments for telerobotic systems. In *Proc., Internationl Conference on Robotics and Automation*, pages 1588–1593, 1997.

[76] B. Nelson, N. Papanikolopoulos, and P. Khosla. Robotic visual servoing and robotic assembly tasks. *IEEE Robotics and Automation Magazine*, 3(2):23–31, 1996.

[77] K. Nickels and S. Hutchinson. Weighting observations: The use of kinematic models in object tracking. In *Proc., Internationl Conference on Robotics and Automation*, pages 1677–1682, 1998.

[78] N. Nilsson. A mobile automaton: An application of artificial intelligence techniques. In *Proc., International Joint Conference on Artificial Intelligence*, pages 233–244, 1969.

[79] P. Y. Oh and P. K. Allen. Design of a partitioned visual feedback controller. In *Proc., Internationl Conference on Robotics and Automation*, pages 1360–1365, 1998.

[80] D. E. Okhotsimsky, A. K. Platonov, I. R. Belousov, A. A. Boguslavsky, S. N. Emelianov, V. V. Sazonov, and S. M. Sokolov. Real time hand-eye system: Interaction with moving objects. In *Proc., Internationl Conference on Robotics and Automation*, pages 1683–1688, 1998.

[81] N. Papanikolopoulos. Selection of features and evaluation of visual measurements during robotic visual servoing tasks. *IEEE Transactions on Robotics and Automation*, 13:279–304, 1995.

[82] N. Papanikolopoulos, P. K. Khosla, and T. Kanade. Adaptive robot visual tracking. In *Proc., American Control Conference*, pages 962–967, 1991.

[83] N. Papanikolopoulos, B. Nelson, and P. K. Khosla. Six degree-of-freedom hand/eye visual tracking with uncertain parameters. In *Proc., Computer Vision and Pattern Recognition*, pages 174–179, 1994.

[84] J. Piepmeier, G. McMurray, and H. Lipkin. A dynamic quasi-newton method for uncalibrated visual servoing. In *Proc., Internationl Conference on Robotics and Automation*, pages 1595–1600, 1999.

[85] J. Ponce and Y. Genc. A new approach to weak calibration. *Image and Vision Computing*, 16(3):223–243, 1998.

[86] A. A. G. Requicha and S. Chan. Representation of geometric features, tolerances and attributes in solid modelers based on csg. *IEEE Trans. on Robotics and Automation*, 2(3):156–166, 1986.

[87] E. Rimon and D. Koditschek. The construction of analytic diffeomorphisms for exact robot navigation on star worlds. *Transactions of the American Mathematical Society*, 327(1):71–115, September 1991.

[88] P. Rives, F. Chaumette, and B. Espiau. Positioning of a robot with respect to an object, tracking it and estimating its velocity by visual servoing. In *Proc., Internationl Conference on Robotics and Automation*, pages 2248–2254, 1991.

[89] A. Rizzi. *Dexterous Robot Manipulation*. PhD thesis, Yale University, November 1995.

[90] L. Robert and O. D. Faugeras. Relative 3d positioning and 3d convex hull computation from a weakly calibrated stereo pair. In *Proc., Internationl Conference on Computer Vision*, pages 540–543, 1993.

[91] L. Robert, C. Zeller, and O. Faugeras. Applications of non-metric vision to some visually guided robotics tasks. Technical Report 2584, INRIA, Sophia-Antipolis, June 1995.

[92] J. Roberts, F. Pennerath, P. Corke, and G. Winstanley. Robust sensing for a 3,500 tonne field robot. In *Proc., Internationl Conference on Robotics and Automation*, pages 2723–2728, 1999.

[93] C. Rothwell, G. Csurka, and O. D. Faugeras. A comparison of projective reconstruction methods for pairs of views. *Computer Vision and Image Understanding*, 68(1):37–49, 1997.

[94] C. Rothwell, O. D. Faugeras, and G. Csurka. A comparison of projective reconstruction methods for pairs of views. *Computer Vision and Image Understanding*, 68(1):37–58, 1997.

[95] A. Ruf, G. Csurka, and R. Horaud. Projective translations and affine stereo calibration. In *Proc., Computer Vision and Pattern Recognition*, pages 475–481, 1998.

[96] A. Ruf, G. Csurka, and R. Horaud. Rigid and articulated motion seen with an uncalibrated stereo rig. In *Proc., Internationl Conference on Computer Vision*, to appear, 1999.

[97] G. Salgian and D. H. Ballard. Visual routines for autonomous driving. In *Proc., Internationl Conference on Computer Vision*, pages 876–882, 1998.

[98] C. Samson, M. Le Borgne, and B. Espiau. *Robot Control: The Task Function Approach*. Clarendon Press, Oxford, 1991.

[99] A. C. Sanderson and L. E. Weiss. Image-based visual servo control using relational graph error signals. In *Proceedings, IEEE*, pages 1074–1077, 1980.

[100] T. J. Schnackertz and R. A. Grupen. A control basis for visual servoing tasks. In *Proc., Internationl Conference on Robotics and Automation*, pages 445–451, 1995.

[101] A. Schweikard. Detecting geometric infeasibility. Technical Report TUM-I9626, Technical University of Munich, Munich, Germany, 1996.

[102] M. Seelinger, S. Skaar, and M. Robinson. An alternative approach for image-plane control of robots. *Lecture Notes in Control and Information Sciences*, 237:41–65, 1998.

[103] R. Sharma and S. Hutchinson. Motion perceptibility and its application to active vision-based servo control. *IEEE Transactions on Robotics and Automation*, 13(4):607–617, Aug. 1997.

[104] S. B. Skaar, W. H. Brockman, and W. S. Jang. Three-dimensional camera space manipulation. *International Journal of Robotics Research*, 9(4):22–39, 1990.

[105] C. Smith and N. Papanikolopoulos. Vision-guided robotic grasping: Issues and experiments. In *Proc., Internationl Conference on Robotics and Automation*, pages 3203–3208, April 1996.

[106] S. M. Smith and J. M. Brady. Susan - a new approach to low level image processing. *International Journal of Computer Vision*, 23(1):45–78, May 1997.

[107] D. E. Stevenson and M. M. Fleck. Robot aerobics: Four easy steps to a more flexible calibration. In *Proc., Internationl Conference on Computer Vision*, pages 34–39, 1995.

[108] H. Sutanto and R. Sharma. Global performance evaluation of image features for visual servo control. *Journal of Robotic Systems*, 13:243–258, 1996.

[109] H. Sutanto and R. Sharma. Efficient multi-strategic hierarchical motion planning with visual servoing constraints. In *Proc., Internationl Conference on Robotics and Automation*, pages 1373–1378, 1998.

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

[111] M. Tonko, K. Schafer, F. Heimes, and H.-H. Nagel. Towards visually servoed manipulation of car engine parts. In *Proc., Internationl Conference on Robotics and Automation*, 1997.

[112] P. H. S. Torr and D. W. Murray. The development and comparison of robust methods for estimating the fundamental matrix. *International Journal of Computer Vision*, 24(3):271–300, 1997.

[113] R. Y. Tsai. A versitile camera clibration technique for high-accuracy 3D machine vision metrology using off-the-shelf tv cameras and lenses. *IEEE Transactions on Robotics and Automation*, 3(4):323–344, 1987.

[114] L. Weiss, A. Sanderson, and C. P. Neuman. Dynamic sensor-based control of robots with visual feedback. *IEEE Transactions on Robotics and Automation*, RA-3(5):404–417, October 1987.

[115] S. W. Wijesoma, D. F. H. Wolfe, and R. J. Richards. Eye-to-hand coordination for vision-guided robot control applications. *International Journal of Robotics Research*, 12(1):65–78, February 1993.

[116] J. D. Wolter. On the automatic generation of assembly plans. In *Proc., Internationl Conference on Robotics and Automation*, 1989.

[117] D. Xiao, B. K. Ghosh, Ning Xi, and T. J. Tarn. Intelligent robotic manipulation with hybrid position/force control in an uncalibrated workspace. In *Proc., Internationl Conference on Robotics and Automation*, pages 1671–1676, 1998.

[118] B. Yoshimi and P. K. Allen. Active, uncalibrated visual servoing. In *Proc., Internationl Conference on Robotics and Automation*, pages 156–161, San Diego, CA, May 1994.

[119] Z. Zhang. Determining epipolar geometry and its uncertainty: A review. *International Journal of Computer Vision*, 27(2):161–195, 1998.

[120] Z. Zhang, O Faugeras, and R. Deriche. An effective technique for calibrating a binocular stereo through projective reconstruction using both a calibration object and the environment. *Videre*, 1(1):58–68, 1997.