Computer Science
Technical Report

Colorado State University

Simultaneous Refinement of Pose and Sensor Registration

A. Schwickerath

October 25, 1996

Technical Report CS-yy-nnn

Computer Science Department
Colorado State University
Fort Collins, CO 80523-1873

Phone: (970) 491-5792 Fax: (970) 491-2466
WWW: http://www.cs.colostate.edu

Abstract

Typically, image registration and 3D object pose determination are treated as isolated problems. Image registration is the task of determining a transformation which maps pixels from one image to pixels in another. Object pose determination is the task of recovering the 3D position and orientation of an object relative to a sensor. However, when multiple sensors from an essentially common point in the world view a scene, the separation of these two tasks is artificial and harmful. What is needed is a single uniform algorithm which positions the object relative to the sensor suite while simultaneously refining the registration of the imagery. This process of simultaneous pose and registration refinement is here called *coregistration*. Presented in this thesis are new algorithms for solving coregistration problems between optical and range imagery using both least-mean squares and least-median squares optimization. A sensitivity analysis using synthetic data is presented for the least-mean squares algorithm. Both algorithms are demonstrated on real data and the least-median squares algorithm is shown to be robust with respect to outliers. Two additional extensions to the approach are proposed: one based upon feature grouping and the other upon Extended Kalman Filters.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

The key contribution of this thesis is an algorithm for the simultaneous refinement of both the sensor suite-to-model pose and sensor-to-sensor registration, or *coregistration*. Model-to-sensor pose determination (e.g., [27, 22]) and homogeneous image registration (e.g., [15]) have both already be dealt with individually. However, there is no simple combination of these two techniques yielding both pose and registration for suites of **heterogeneous** sensors.

Why is such an algorithm necessary? Whenever discrete, inexpensive sensors are used in conjunction, there will be variations in intersensor parameters. The variability of intersensor calibration parameters can be traced to vibrations, thermal changes, and other sources of sensor drift. To compensate for these variations, the sensor-to-sensor transformation should be updated for each **image** suite, rather than only once for the **sensor** suite. Without continual recalibration, the resulting registration error can cause algorithms that rely upon accurate registration to fail. For example, in the case where range data helps predict occlusion in optical data, poor sensor registration can result in incorrect identification of occluded and unoccluded features. This is true, even in the case of a few pixel misregistration, when the total number of pixels making up the object in either image is small.

We believe that registration refinement as part of the matching of model features-to-image features and the pose determination process is essential in many consumer and industrial domains. In the case of assembly lines, infrared sensors coupled with color sensors can help detect flaws, but only if the images are correctly registered. Vibrations from neighboring equipment or variations in temperature will vary the relative sensor positions, yielding poor results if a static image registration were used.

This work on geometric model based sensor fusion exists at the junction of three subfields of Computer Vision: *geometric model matching*, *model-based pose determination* and *sensor fusion*. Model matching and model-based pose are already closely tied in the Vision literature (e.g., [34, 1, 2, 29, 19]). In model matching, a correspondence between features (e.g., lines, corners) in the image and features in the model is sought. In pose determination, the orientation and position of the model relative to the sensor is estimated, often depending upon a presupposed correspondence. A matching algorithm can provide this correspondence. However, determining which model and image features should be paired often requires evaluating the goodness of fit *given the best pose for the match*. In this

way, these two tasks are intertwined. Few cases exist of matching or pose determination being tied to sensor fusion (e.g., [19, 13]).

1.2 Model-Based Pose

Different approaches have been taken to model-based pose determination (e.g., [27, 29, 19, 8]). This thesis is concerned with those which act directly on 3D model features, rather than 2D deformable graphs [35], or weak perspective (2D rigid projections) methods [8, 2]. Methods which deal directly with 3D model features can utilize full 3D perspective and implicitly deal with any view of the object. 2D methods can correct only somewhat for perspective projections. They also must explicitly describe changes in appearance due to changes in viewpoint. This can result in either large databases of potential views or a restricted set of poses. Most 2D methods also provide only coarse pose determination, due to the partitioning of the views. The 3D methods can yield results accurate to the limits of the perspective sensor model, tolerances in the object modeling, and the accuracy of the image measurements.

In 3D model-based pose determination, the standard approach is to cast the problem in terms of optimization. The basic optimization approach can be outlined as:

- Propose a criterion function: an error measure or fitness function (one is just the inverse of the other).
- Either:
  - Find a closed form solution. [23]
  - Use a standard iterative optimization procedure such as Newton or Quasi-Newton methods. [27]

Notice that this outline is general for all optimization problems and as such, will be used as a point of reference for both previous work and the algorithms presented in Chapters 2 and 4.

Horn [22] (pages 303-307) deals with the pose (which he calls "absolute orientation", using the term from photogrametry) of a 3D model relative to 3D data, with all features being 3D points. The error function is the sum of squared Euclidean distances between paired model and data points. He solves this system using Newton's method. In [23], he provides an elegant closed form solution.

Kumar [27] considers pose determination based upon 3D model lines and 2D image lines, assuming a perspective camera model. He proposes 3 different error functions, one of which will be described in Chapter 2 and another in Chapter 4. These criteria functions are optimized using the Levenberg-Marquardt extension to the Newton Method.

Common to both Horn and Kumar is the use of a least-mean squares framework. Another approach, and a more general one, is Kalman Filtering. [10, 39] Hel-Or [19] and Hel-Or and Werman [18, 20] use this approach to fuse heterogeneous measurements into a pose estimation. This is, in fact, just another way of casting the sensor fusion problem. The premise of Kalman Filtering is that you have measurements that are a linear function of state, but are contaminated by noise of a known quality. The purpose of the Kalman Filter is to minimize the expected error in the state estimate. In terms of pose determination, the state is the pose, the noise is determined by the sensor and the feature extraction method, and the linear function is an approximation of the model-to-sensor

transformation. As with previous uses of Kalman Filtering in computer vision (e.g., [1]), this approach elegantly incorporates matching into the pose refinement process. This will be discussed in greater depth in Chapter 4.

Not all pose estimation processes deal with explicit correspondences. For breadth, I will briefly introduce one of these. Huttenlocher's Hausdorf Distance [24] deals with overall shape properties, rather than explicit pairings between model and data features. Implicit pairings are used instead. While this removes the usually difficult matching problem, it yields other problems. The distance measure is not always an intuitive measure of the goodness of fit between two shapes. In addition, occlusion is poorly dealt with unless *a priori* information about the amount of the model missing from the image is available or can be estimated. Also, the use of higher-level features is not included in the Hausdorf distance, so only point data can be used. This limits the amount of information that can be exploited by the method.

1.3 Model-Based Matching

While it is possible to compute pose without an explicit correspondence as was shown in the case of the Hausdorf Distance, the 3D-model based pose estimation methods of interest in this thesis require correspondences. By making explicit the pairing between model and image features, it is possible to deal with features missing in the image (omission) or appearing in the image but unexplained by the model (clutter). Both as examples and to motivate the matching work in Chapter 3, let us review some common approaches to establishing correspondences.

A classic matching method in computer vision is tree search, espoused by Grimson [16]. In this method, a tree is considered, where each node denotes a match and each link, the addition of a pair to the correspondence. Each level of the tree represents the coupling of a specific image feature. This limits matches to include only pairings of one model feature to many data features, not allowing multiple model features to correspond to one data feature. Grimson does introduce a method for handling clutter, however, by allowing the matching of an image feature to a *null* model feature.

Beveridge [2] defines a combinatorial search space and uses local search to find locally optimal matches. Unlike the tree search methods, the search space used here takes into account mappings of potentially many model features to potentially many data features. Local search in this large space is robust, but can be a slow process. We will revisit this approach again in Chapter 3.

In Kalman Filtering approaches, such as those proposed by Ayache [1] and Hel-Or and Werman [19, 18, 20], the matching process is interleaved with the pose estimation process. Rather than finding a complete correspondence and then recovering the pose, individual feature pairs are selected based on consistency with the current match, similar to tree matching but allowing many-to-many mappings. These pairs are included in the match, updating the pose estimate so that another consistent feature pair can be selected.

1.4 Sensor Fusion

Eason and Gonzalez [13] propose a general least-squares framework for multi-sensor fusion. At its heart, this is simply a direct extension of the basic least-squares approach for finding the pose between a single sensor and a model instance. The sensor suite is considered to

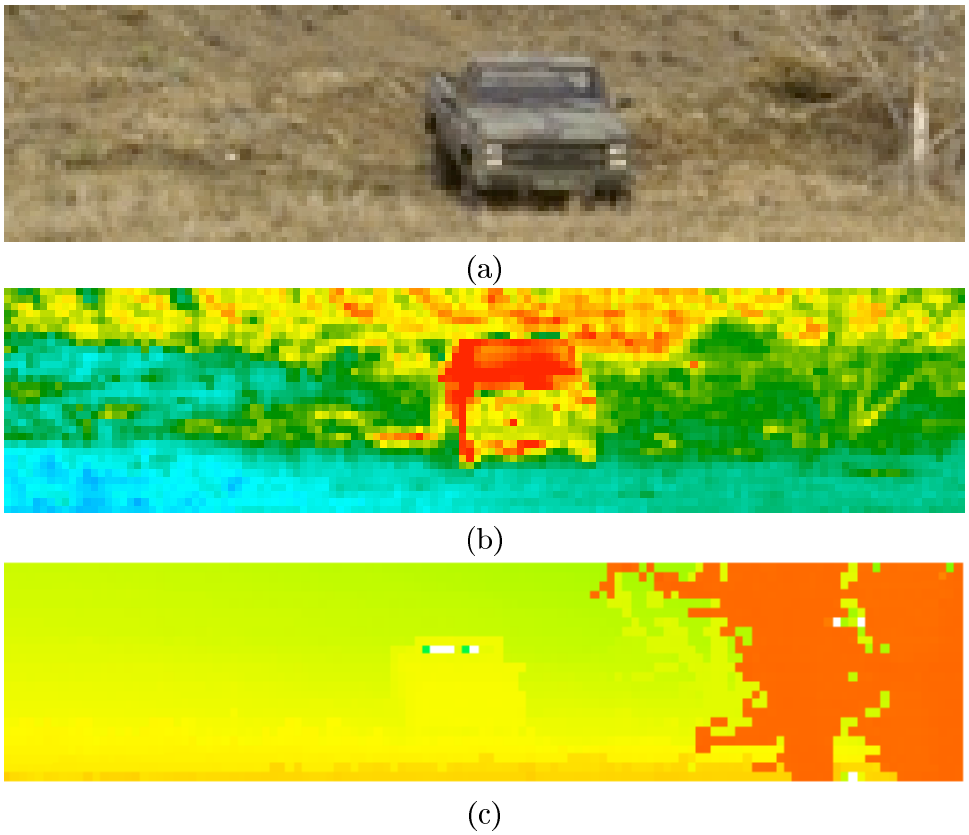


Figure 1.1: An example of (a) color, (b) infrared, and (c) range data of a single scene (Shot 20, Array 8 from the Fort Carson Data Collection [5]).

be completely calibrated, so the exact pose of the sensors relative to each other is deemed known. All feature pairs are point-to-point correspondences, which we will see in Chapter 3 leads to unwieldy search spaces.

A similar approach, already introduced in section 1.2, is Hel-Or and Werman's Kalman Filtering method [19, 18, 20]. This method is even further constrained, assuming that the sensors are coincident (i.e., the focal point and optical axis of the sensors are coincident), the intersensor registration is the identity transformation.

Both of these approaches miss a key issue: sensor suites comprised of discrete sensors are not coincident and do not remain precisely registered [17]. This shortcoming could be ignored if a precise sensor-to-sensor correspondence could be computed from the raw data. This is not an easy task when the nature of the sensors is diverse. The images returned by a color camera, an infrared camera, and a range sensor (see Figure 1.1) do not necessarily correlate.[1] As such, registration algorithms must deal with homogeneous sensor suites [15].

[1]These images are part of a set of IR, color, and range images collected at Fort Carson in Fall 1994. They will be used for testing throughout this thesis.

1.5 Contributions

As stated earlier, the concept of coregistration is the main contribution of this thesis. In Chapter 2, I present a simple least-squares coregistration solution. The criterion function draws upon previous work by Kumar [27] and Horn [22, 23], with similarities to Eason and Gonzalez's model based sensor fusion [13]. An empirical sensitivity analysis is performed upon an implementation of this algorithm. Finally the algorithm is applied to real data.

The application of the least-squares coregistration algorithm to real data points out a shortcoming of all least-squares algorithms: outlier sensitivity. In Chapter 2, hand-picked correspondences produce unstable results. To correct for this problem and place the least-mean squares algorithm in a more useful context, two methods of correspondence construction are presented in Chapter 3: a robust statistical method and a standard search method. Results on real images are discussed for both of these methods.

The least-mean squares algorithm provides a proof of concept for coregistration. In Chapter 4, I sketch two, more sophisticated coregistration algorithms. The first is a direct extension of the simple least-squares formulation. The criterion function is replaced with one which utilizes higher level, linear and planar features in the range correspondence, rather than a simple point-to-point mapping. This should address least-mean squares stability issues as well as reduce the correspondence search space.

The second approach, a Kalman Filtering solution to coregistration, directly extrapolated from Hel-Or and Werman's articulated model formulation [18, 20], is also outlined in Chapter 4. This formalization provides a more general framework for encoding intersensor constraints in much the same way as it provided a general method of encoding constraints for geometric models.

The purpose of this thesis is to provide an introduction to the concept of coregistration: a concept which I consider important in the progression of multisensor data interpretation. All of the algorithms, whether implemented and tested or simply sketched, provide both an investigation into the difficulties of the coregistration problem as well as methods for lessening these hardships.

Chapter 2

Least-Mean Squares Coregistration

2.1 Coregistration: Pose plus Sensor Registration

Previously I defined *coregistration* as the process of simultaneously recovering sensor suite-to-model pose and sensor-to-sensor registration. Such a definition allows for constrained and unconstrained sensor suites, but the latter is not very interesting. The constrained case takes advantage of knowledge about the physical sensing system such as coplanarity, relative orientation and relative distance. These constraints combine the independent pose estimation processes into a single optimization problem, though they may also increase the difficulty of solving the optimization problem.

I also framed the pose (and hence, coregistration) task as an optimization problem. In this chapter, a least-mean squares optimization solution for coregistering a specific sensor suite is presented as an example.

2.1.1 Sensing Geometry for Optical and Range Sensors

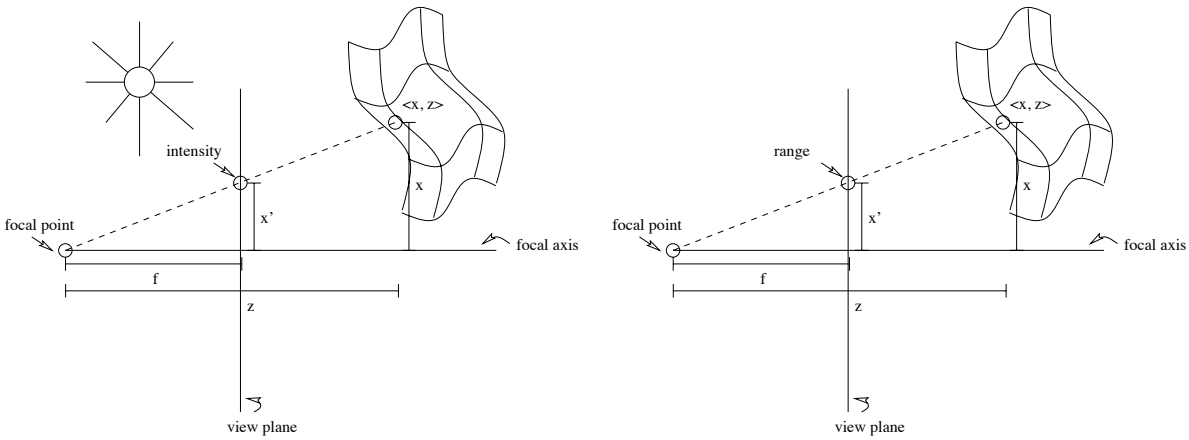
Before any meaningful high-level interpretation of the image can be made, the sensing technology must be understood. The sensor suite considered in this thesis consists of a CCD (color) sensor and a LADAR (range) sensor.

The projective model for both sensors is approximated by a radial geometry. Figure 2.1(a) shows a cross-section for the mapping of a 3D point in the scene to a 2D image point for the optical (CCD) sensor.[1] The basic model consists of a focal point, a view plane, and a scene in the world. Each point in the scene is mapped onto the view plane by a ray passing from the focal point through the view plane and contacting an imaged point in the scene.

Mathematically, the position on the view plane can be computed by similar triangles. First we can note that

$$\frac{x'_{meters}}{f} = \frac{x}{z}$$

[1]Here I am simply considering the (x, z) coordinates, since the (y, z) mapping is derived in the same way.



(a) (b)

Figure 2.1: Perspective projection from scene into an image. The color sensor is shown in (a) and the range in (b).

This gives x' in world measurement units, such as meters, rather than in pixels. To convert meters into pixels, a new factor, s_x, can be defined as $s_x = f\frac{pixels}{meters}$ yielding

$$\begin{aligned} \frac{x'_{pixels}}{s_x} &= \frac{x}{z} \\ x' &= s_x\frac{x}{z} \end{aligned}$$

This maps real world 2D coordinates into 1D pixel coordinates.

For the other dimension, we add a similar equation for y'. It is important to note that s_x is not necessarily the same as s_y, due to the possible difference in aspect ratio.

A simplified version of the LADAR sensor is shown in Figure 2.1(b). The 3D range sensor is very similar to the 2D optical sensor. The perspective mapping is approximated by the same pinhole projection model. The only addition is that of depth; the depth is the "time of flight" along the projection ray, or $r = \sqrt{x^2 + y^2 + z^2}$.

An additional aspect of LADAR data over optical data is the ability to reconstruct the original 3D world coordinates of the point being imaged.

$$\begin{aligned} x &= \frac{x'}{s_x}z \\ y &= \frac{y'}{s_y}z \\ z &= \sqrt{\frac{r^2}{1 + \frac{x'^2}{s_x^2} + \frac{y'^2}{s_y^2}}} \end{aligned}$$

In the case of optical (CCD) data, this is not true. 2D data can be mapped to the set of points on the ray from the focal point through the image point, but the exact 3D point cannot be reconstructed from a single natural image.

2.1.2 Constraints

The color and range sensors are assumed to have coplanar, co-oriented focal planes, as shown in Figure 2.2. To implicitly codify these constraints, the pose+registration param-

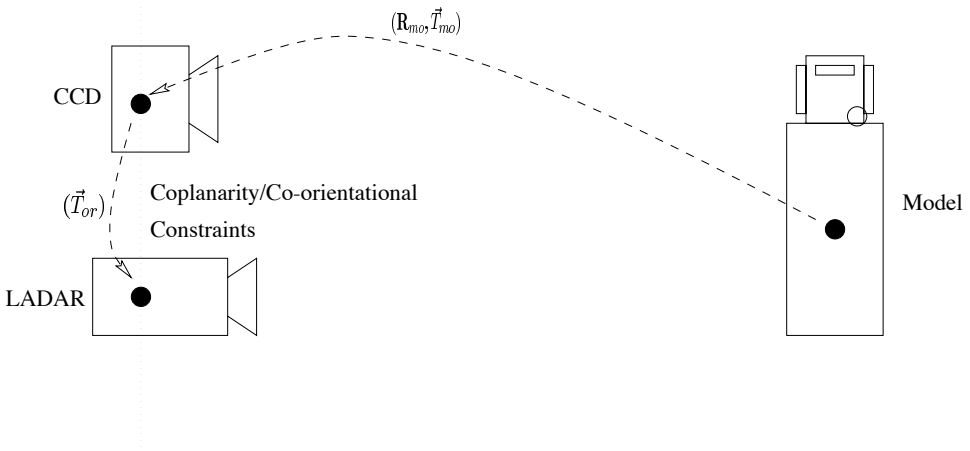


Figure 2.2: Color and Range sensor suite discussed in this thesis.

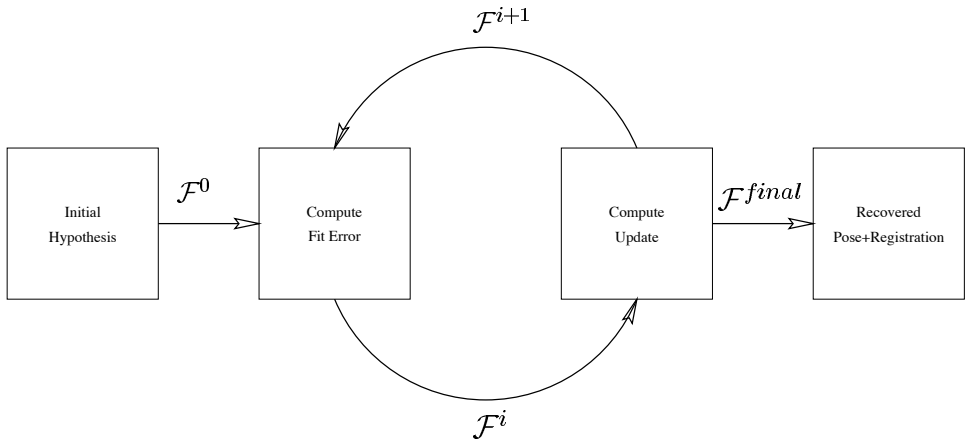


Figure 2.3: A block diagram of least-mean squares optimization in coregistration.

eters are chosen to be the 3D model-to-optical sensor mapping, $(\mathbf{R}_{mo}, \vec{T}_{mo})$[2], and the 2D translation $\vec{T}_{or}$ between the optical sensor coordinate frame and the range sensor frame. Collectively, these pose+registration parameters are referred to as $\mathcal{F}$.

Strictly speaking, the sensors in this suite were not guaranteed to be coplanar, but only *nearly* coplanar. Beveridge, et. al. [7] showed that small rotations, such as the ones in our sensor suite, can be approximated by translations in the view plane.

2.2 Overview of the Coregistration Refinement Algorithm

The coregistration refinement task can be cast as an optimization problem, following the outline in Section 1.2. First we must define a criterion function (Section 2.3), then we must

[2]A subscripting convention is adopted in this thesis to denote both data and transformations. Each transformation and point is subscripted by a chain of letters indicating, from left to right, the coordinate systems being used: m indicates model, o indicates optical, and r indicates range. For example, $\vec{T}_{mo}$ is the translation from model to optical coordinate frames.

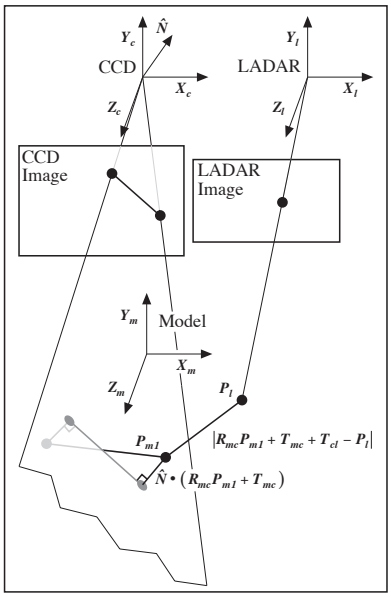


Figure 2.4: Illustrating distance errors which define optimal coregistration.

choose an optimization method (Section 2.4). The coregistration algorithm presented here couples the criterion functions used by Horn [22] and Kumar [27]. Coupling these through the constrained multisensor geometry yields a new, more complex and higher dimensional criterion function.

2.3 The Criterion Function

When casting pose refinement as an optimization problem, the criterion function measures the goodness or badness of fit between the model and the data for a given pose. The model is represented by 3D features, such as lines and points, extracted from a CAD model. The data consists of features extracted from a color image and a corresponding range image of a scene containing the modeled object.[3]

Since the range and optical data are obtained from independent sensors, the criterion function, which I will refer to as the *fit error*, can be decomposed into

$$E_{fit}(\mathcal{F}, \mathcal{C}) = \alpha_{fit} E_{fit,o}(\mathcal{F}, \mathcal{C}_o) + (1 - \alpha_{fit}) E_{fit,r}(\mathcal{F}, \mathcal{C}_r) \tag{2.1}$$

where E_{fit} is the total fit error for the sensor suite, $E_{fit,o}$ and $E_{fit,r}$ are the fit errors for the optical and range data individually. The weight $\alpha_{fit} \in [0, 1]$ tunes the relative importance of the sensors. The intersensor constraints are not seen in this measure, since they are implicit in the parameterization.

---

[3]Another aspect of the computer vision problem is the decision of whether a specific model can be found in an image. I assume that this determination has already been made by a hypothesis generation phase such as [12, 9]. While the criterion function could be used to test for the existence of a model in a specific pose, I am limiting the treatment in this Chapter to the case where an instance of the model is known to exist in an image.

2.3.1 The Optical Sensor Error Term

The optical sensor fit error is the same as Kumar's E_1 measure [27]. This measures the endpoint-to-plane error between a 3D model line and a 2D data line, assuming a pinhole camera model. The basic idea stems from the perspective mapping process (shown in Figure 2.4). Rays are cast starting at the focal point of the camera and passing through the endpoints of the image line. These two rays define a plane. The fit error is the weighted squared distance from model line endpoints to the plane.

Mathematically, Kumar's fit error can be written as the weighted sum of these individual errors:

$$E_{fit,o} = \frac{1}{2\tau_{mo}^2 \sum_{i=1}^{n_o} \lambda_{oi}} \sum_{i=1}^{n_o} \sum_{j=1}^{2} \lambda_{oi} \left(\hat{N}_{oi} \cdot \left(R_{mo} \vec{P}_{mij} + \vec{T}_{mo} \right) \right)^2 \tag{2.2}$$

where

n_o The number of model-to-optical pairs.
τ_{mo} The threshold distance within model-to-optical pairs.
λ_{oi} Weight for model-to-optical pair i.
$\hat{N}_{oi}$ The unit normal of the plane formed by optical line i and the focal point.

The individual pair weights, λ_i, can be used to adjust the penalty for poor fit. A common assignment for this weight is based upon the distance to the line.

$$\lambda_i = \frac{1}{\left| \frac{\vec{P}_{i1} - \vec{P}_{i2}}{2} \right|}$$

Selection of weight values will be taken up in Section 2.3.3.

2.3.2 The Range Sensor Error Term

The range data correspondence is composed of 3D data points matched to 3D model points. A standard criterion function (used by Horn in [22, 23]) and the one used here is the squared Euclidean distance between the paired points. This is represented mathematically as

$$\begin{aligned} E_{fit,r} &= \frac{1}{\tau_{mr}^2 \sum_i \lambda_{ri}} \sum_{i=1}^{n_r} \lambda_{ri} \left| \left(\left(\vec{P}_{moi} + \vec{T}_{or} \right) - \vec{P}_{ri} \right) \right|^2 \\ \vec{P}_{moi} &= R_{mo} \vec{P}_{mi} + \vec{T}_{mo} \end{aligned} \tag{2.3}$$

where

n_r The number of model-to-range pairs.
τ_{mr} The threshold distance within model-to-range pairs.
λ_{ri} Weight for model-to-range pair i.

2.3.3 Selecting Weights to Normalize and Combine Error Terms

The weighting term $\frac{1}{2\tau_{mo}^2 \sum_i \lambda_{oi}}$ normalizes the $E_{fit,o}$ measure based upon the assumption that the image line endpoints fall within τ_{mo} of the associated model line. That is, if we assume that the underlying noise process is Gaussian, then we can set τ_{mo} to be 2 times the standard deviation of the Gaussian noise process. This will guarantee 98% of the points due to the model will fall within τ_{mo} of the perfect model projection. Notice that this τ_{mo}

is in Euclidean units (such as meters) rather than in pixels. An exact upper bound can be found based upon a pixel-based threshold, but the formulation is linear in the number of correspondences, rather than constant time. Hence, the simpler τ_{mo}, based on an estimate of total noise on target is computationally necessary in the current implementation. Also notice that while this normalization does not absolutely guarantee that $E_{fit,o}$ will remain in the range $[0,1]$, it does guarantee it assuming that the correspondence does not contain any pairs with image endpoint-to-model-line distances greater than the tolerance τ_{mo} (i.e., outliers). Similarly, $\frac{1}{2\tau_{mr}^2 \sum_i \lambda_{ri}}$ is used to weight the range data. As in the optical case, τ_{mr} is the maximum expected distance between a model point-range point pair and all of the previous comments hold.

One method for selecting τ_{mo} and τ_{mr}, as stated above, is to assume a Gaussian noise model and set the thresholds to 2σ. Methods have been proposed for approximating the overall noise in an image [25, 42, 37].

Another method for selecting these thresholds (both τ_{mo} and τ_{mr}), and the one used in the experiments in Chapter 3, is to observe the distribution of unweighted pair errors given the initial hypothesis. Given a ranked list of these residuals for each sensor, a threshold can be selected based upon the amount of good data we believe we have. In practical terms, this means selecting a percentile or count for each of the sensors and using that residual from the ranked list as the threshold.

2.4 Optimizing Coregistration Through Iterative Update

The criterion function defined in Equations 2.1, 2.2, and 2.3 is non-linear due to rotation. There are many general techniques for optimizing non-linear functions [43, 36], all of which are iterative.

The optimization technique used here is similar to the Newton method, called Levenberg-Marquardt [36]. The basic idea is to optimize the nonlinear function, linearized about the current estimate, as in the Newton method. Levenberg-Marquardt effectively trades off control between the Newton second order technique and a first order technique. This deals with the unstable tendencies of the Newton method. Levenberg-Marquardt has proven useful in solving pose determination problems by Lowe [29, 30], Kumar [27], and Beveridge [2].

The first step in optimizing the fit error is to obtain a linearization of the criterion function. If we treat R_{mo} as a 3×3 rotation matrix, solving for R_{mo} by minimizing Equation 2.1 and allowing all 9 terms to vary independently violates the constraint that R_{mo} be an ortho-normal matrix. While the matrix terms $(r_{mc_{1,1}} \ldots r_{mc_{3,3}})$ could be constructed in such a way as to allow only rigid rotations, this would yield an even more nonlinear equation. Kumar [26, 28] suggests another approach: Rodriguez's formula, which is an approximation appropriate for small rotations.[4] To rotate a point $\vec{P_{mi}}$ by an amount R_{mo}, we write

$$\begin{aligned} R_{mo}\vec{P}_{mi} &= R^e_{mo}\vec{P}_{mi} + \delta\vec{\omega}_{mo} \times \left(R^e_{mo}\vec{P}_{mi}\right) \\ &= \vec{P}^e_{mi} + \delta\vec{\omega}_{mo} \times \vec{P}^e_{mi} \end{aligned}$$

[4] Another approach employed by Horn [23] and Hel-Or [19] is the use of quaternions.

(Note: to provide a more compact notation the vector $\vec{P}^e_{mi}$ is introduced as the current estimate of the transformed model point.) Here, R^e_{mo} is the current estimated rotation (3×3) matrix and $\delta\vec{\omega}_{mo}$ is the small rotation update represented as a unit rotational axis scaled by the rotational magnitude.

The error terms in equations 2.2 and 2.3 may now be rewritten as follows.

$$E_{fit,o} = \frac{1}{2\tau^2_{mo}\sum_i \lambda_{oi}} \cdot \sum_{i=1}^{n_o}\sum_{j=1}^{2} \lambda_{oi}\left(\hat{N}_{oi} \cdot \left(\vec{P}^e_{mij} + \delta\vec{\omega}_{mo} \times \vec{P}^e_{mij} + \vec{T}^e_{mo} + \Delta\vec{T}_{mo}\right)\right)^2 \tag{2.4}$$

$$E_{fit,r} = \frac{1}{\tau^2_{mr}\sum_i \lambda_{ri}} \cdot \sum_{i=1}^{n_r} \lambda_{ri}\left|\left(\left(\vec{P}^e_{mi} + \delta\vec{\omega}_{mo} \times \vec{P}^e_{mi} + \vec{T}^e_{mo}\right) + \Delta\vec{T}_{mo} + \vec{T}^e_{or} + \Delta\vec{T}_{or} - \vec{P}_{ri}\right)\right|^2 \tag{2.5}$$

In order to minimize the linearized E_{fit} in equations 2.4 and 2.5 with respect to $\delta\vec{\omega}_{mo}$, $\Delta\vec{T}_{mo}$ and $\Delta\vec{T}_{or}$, the partial derivatives with respect to each are set to zero and the resulting system of equations solved for the coregistration update parameters. These partial derivatives are

$$\begin{aligned}\frac{\partial E_{fit}}{\partial\delta\vec{\omega}_{mo}} &= w_{mo}\sum_{i=1}^{n_o}\sum_{j=1}^{2}\lambda_{oi}\left(\left(\vec{P}^e_{mij} \times \hat{N}_{oi}\right) \cdot \delta\vec{\omega}_{mo} + \hat{N}_{oi} \cdot \Delta\vec{T}_{mo}\right)\left(\vec{P}^e_{mij} \times \hat{N}_{oi}\right) \\ &+ w_{mr}\sum_{i=1}^{n_r}\lambda_{ri}\left(M^2_{moi}\delta\vec{\omega}_{mo} + M_{moi}\Delta\vec{T}_{mo} + M_{moi}\Delta\vec{T}_{or}\right) \\ &+ w_{mo}\sum_{i=1}^{n_o}\sum_{j=1}^{2}\lambda_{oi}\left(\hat{N}_{oi} \cdot \left(\vec{P}^e_{moij} + \vec{T}^e_{mo}\right)\right)\left(\vec{P}^e_{mij} \times \hat{N}_{oi}\right) \\ &+ w_{mr}\sum_{i=1}^{n_r}\lambda_{ri}M_{moi}\left(\vec{P}^e_{mi} + \vec{T}^e_{mo} + \vec{T}^e_{or} - \vec{P}_{ri}\right)\end{aligned} \tag{2.6}$$

$$\begin{aligned}\frac{\partial E_{fit}}{\partial\Delta\vec{T}_{mo}} &= w_{mo}\sum_{i=1}^{n_o}\sum_{j=1}^{2}\lambda_{oi}\left(\left(\vec{P}^e_{mij} \times \hat{N}_{oi}\right) \cdot \delta\vec{\omega}_{mo} + \hat{N}_{oi} \cdot \Delta\vec{T}_{mo}\right)\hat{N}_{oi} \\ &+ w_{mr}\sum_{i=1}^{n_r}\lambda_{ri}\left(M_{moi}\delta\vec{\omega}_{mo} + \Delta\vec{T}_{mo} + \Delta\vec{T}_{or}\right) \\ &+ w_{mo}\sum_{i=1}^{n_o}\sum_{j=1}^{2}\lambda_{oi}\left(\hat{N}_{oi} \cdot \left(\vec{P}^e_{mij} + \vec{T}^e_{mo}\right)\right)\hat{N}_{oi} \\ &+ w_{mr}\sum_{i=1}^{n_r}\lambda_{ri}\left(\vec{P}^e_{mi} + \vec{T}^e_{mo} + \vec{T}^e_{or} - \vec{P}_{ri}\right)\end{aligned} \tag{2.7}$$

$$\begin{aligned}\frac{\partial E_{fit}}{\partial\Delta\vec{T}_{or}} &= w_{or}\sum_{i=1}^{n_r}\lambda_{ri}\left(M_{moi}\delta\vec{\omega}_{mo} + \Delta\vec{T}_{mo} + \Delta\vec{T}_{or}\right) \\ &+ w_{or}\sum_{i=1}^{n_r}\lambda_{ri}\left(\vec{P}^e_{mi} + \vec{T}^e_{mo} + \vec{T}^e_{or} - \vec{P}_{ri}\right)\end{aligned} \tag{2.8}$$

In these equations, we simplify by rewriting the weighting terms as

$$w_{mo} = \alpha_{fit} \frac{1}{2\tau_{mo}^2 \sum_i \lambda_{oi}} \tag{2.9}$$

$$w_{mr} = (1-\alpha_{fit}) \frac{1}{\tau_{mr}^2 \sum_i \lambda_{ri}} \tag{2.10}$$

The following matrix M_{moi} is introduced to simplify the expressions.

$$M_{moi} = \begin{pmatrix} 0 & \left(\vec{P}^e_{mi}\right)_z & -\left(\vec{P}^e_{mi}\right)_y \\ -\left(\vec{P}^e_{mi}\right)_z & 0 & \left(\vec{P}^e_{mi}\right)_x \\ \left(\vec{P}^e_{mi}\right)_y & -\left(\vec{P}^e_{mi}\right)_x & 0 \end{pmatrix} \tag{2.11}$$

The matrix M_{moi} is the vector product with $\vec{P}^e_{mi}$. In other words: $M_{moi}\vec{V} = \vec{V} \times \vec{P}^e_{mi}$. It is also the partial derivative of $\delta\vec{\omega}_{mo} \times \vec{P}^e_{mi}$ with respect to $\delta\vec{\omega}_{mo}$.

Setting the partial derivatives to zero yields 9 linear equations in 9 unknowns. These linear equations may be written as:

$$\begin{pmatrix} A & B & C \\ D & E & F \\ G & H & J \end{pmatrix} \begin{pmatrix} \delta\vec{\omega}_{mo} \\ \Delta\vec{T}_{mo} \\ \Delta\vec{T}_{or} \end{pmatrix} = \begin{pmatrix} \vec{K} \\ \vec{L} \\ \vec{M} \end{pmatrix} \tag{2.12}$$

where the constants A through M are defined in

$$A = w_{mo} \sum_{i=1}^{n_o} \sum_{j=1}^{2} \lambda_{oi} \left(\vec{P}^e_{mij} \times \hat{N}_{oi}\right) \left(\vec{P}^e_{mij} \times \hat{N}_{oi}\right)^T + w_{mr} \sum_{i=1}^{n_r} \lambda_{ri} M^2_{moi} \tag{2.13}$$

$$B = w_{mo} \sum_{i=1}^{n_o} \sum_{j=1}^{2} \lambda_{oi} \left(\vec{P}^e_{mij} \times \hat{N}_{oi}\right) \hat{N}^T_{oi} + w_{mr} \sum_{i=1}^{n_r} \lambda_{ri} M_{moi} \tag{2.14}$$

$$C = w_{mr} \sum_{i=1}^{n_r} \lambda_{ri} M_{moi} \tag{2.15}$$

$$D = w_{mo} \sum_{i=1}^{n_o} \sum_{j=1}^{2} \lambda_{oi} \left(\hat{N}_{oi} \left(\vec{P}^e_{mij} \times \hat{N}_{oi}\right)^T\right) + w_{mr} \sum_{i=1}^{n_r} \lambda_{ri} M_{moi} \tag{2.16}$$

$$E = w_{mo} \sum_{i=1}^{n_o} \sum_{j=1}^{2} \lambda_{oi} \hat{N}_{oi} \hat{N}^T_{oi} + w_{mr} \sum_{i=1}^{n_r} \lambda_{ri} I_3 \tag{2.17}$$

$$F = w_{mr} \sum_{i=1}^{n_r} \lambda_{ri} I_3 \tag{2.18}$$

$$G = w_{mo} \sum_{i=1}^{n_r} \lambda_{ri} M_{moi} \tag{2.19}$$

$$H = w_{mo} \sum_{i=1}^{n_r} \lambda_{ri} I_3 \tag{2.20}$$

$$J = w_{mo} \sum_{i=1}^{n_r} \lambda_{ri} I_3 \tag{2.21}$$

$$\vec{K} = -w_{mo} \sum_{i=1}^{n_o} \sum_{j=1}^{2} \lambda_{oi} \left(\hat{N}_{oi} \cdot \left(\vec{P}^e_{mij} + \vec{T}^e_{mo} \right) \right) \left(\vec{P}^e_{mij} \times \hat{N}_{oi} \right)$$
$$-w_{mr} \sum_{i=1}^{n_r} \lambda_{ri} M_{moi} \left(\vec{P}^e_{mi} + \vec{T}^e_{mo} + \vec{T}^e_{or} - \vec{P}_{ri} \right) \quad (2.22)$$

$$\vec{L} = -w_{mo} \sum_{i=1}^{n_o} \sum_{j=1}^{2} \lambda_{oi} \left(\hat{N}_{oi} \cdot \left(\vec{P}^e_{mij} + \vec{T}^e_{mo} \right) \right) \hat{N}_{oi}$$
$$-w_{mr} \sum_{i=1}^{n_r} \lambda_{ri} \left(\vec{P}^e_{mi} + \vec{T}^e_{mo} + \vec{T}^e_{or} - \vec{P}_{ri} \right) \quad (2.23)$$

$$\vec{M} = -w_{mr} \sum_{i=1}^{n_r} \lambda_{ri} \left(\vec{P}^e_{mi} + \vec{T}^e_{mo} + \vec{T}^e_{or} - \vec{P}_{ri} \right) \quad (2.24)$$

Since $\Delta\vec{T}_{or}$ has the form $\left(\delta t_{ol_x}, \delta t_{ol_y}, 0\right)$, we can drop the rightmost column and bottom row of the 9×9 matrix in equation 2.12. The result is an 8×8 linear system which is used to iteratively solve for the optimal set of coregistration parameters.

Each time through the loop, the resulting delta updates ($\delta\vec{\omega}_{mo}$, $\Delta\vec{T}_{mo}$, and $\Delta\vec{T}_{or}$) are added to the current estimate (R^e_{mo}, $\vec{T}^e_{mo}$, and $\vec{T}^e_{or}$). The constants in Equations 2.13-2.24 are recomputed each time through the loop (see Figure 2.3). The algorithm converges when iterative improvement in E_{fit} drops below a threshold value. Unsuccessful termination occurs if the total number of iterations exceeds a maximum number of iterations.

2.5 Sensitivity Analysis

A key problem with iterative non-linear optimization techniques is the possibility of returning a local, rather than a global optima. Part of this sensitivity is due to the initial estimate. In the case where a model is being fit to data, the function space may be warped so that a nearby but non-global optima attracts the solution.

To investigate the sensitivity of the coregistration algorithm, a pair of synthetic tests were designed: one to test sensitivity to the initial estimate and the other to test the sensitivity to noisy data. For both of these tests, the same configuration is used.

For both of these tests, a sensor suite consisting of an optical and a range sensor was used. The characteristic parameters of these sensors is based upon the sensor suite planned for the DARPA Unmanned Ground Vehicle (UGV) as of early 1994. The synthetic optical sensor has a 4° field of view and generates a 512×512 image; the range images, 6 pixels per meter at 500 meters. The sensors are separated by 1 meter. Each model is located 500 meters from the sensors along the focal axis of the optical sensor. The ground truth image data for these tests is obtained for each sensor by synthetically projecting the appropriate model features (lines for optical, points for range) onto the sensor view plane.

Algorithm tuning parameters such as error weighting terms and convergence criteria are constant throughout both experiments. The weights in the coregistration error, λ_{oi}, λ_{ri}, w_{mo} and w_{mr}, are all set to 1.0. In the case of the λ weights, this simply means that the distance to the 3D features does not affect the noise. This basically means that all features, regardless of distance, are taken to be equally important. At the time this experiment was performed, the τ and α version of the weighting terms had not been formulated; w_{mo} and w_{mr} combined α, τ_{mo}, and τ_{mr}. The w weights are set to 1 since roughly equal amounts of data are available from each sensor. The convergence threshold

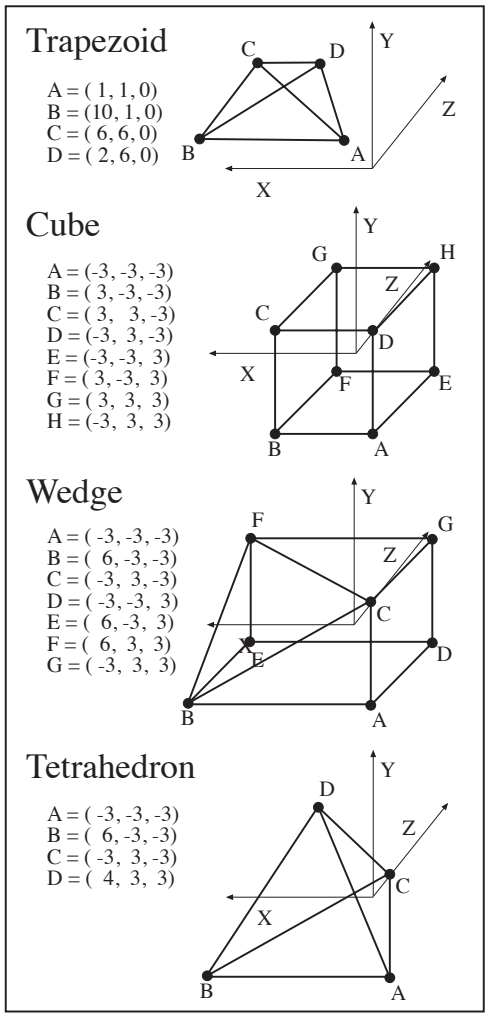


Figure 2.5: Four models used to test coregistration code.

for E_{fit} is 10^{-4}. The maximum number of iterations is 20, since experience showed that the algorithm converged within 20 iterations, or else converged to a suboptimal solution.

Two sets of experiments were conducted: I) sensitivity to noise in initial coregistration estimate, and II) sensitivity to noisy image data. Both tests were run on four synthetic models, shown in Figure 2.5. The models exhibit different geometric characteristics including planarity or lack of planarity, symmetry or lack of symmetry, and few versus many features. The exact coordinates of the points comprising the models are given and line segments used for model-to-optical component are also shown. The *Trapezoid* is an example of a planar asymmetric object, testing coregistration performance on sets of coplanar points. The *Cube* is an example of a non-planar symmetric object. The *Wedge* and the *Tetrahedron* both test stability on asymmetric, non-planar objects, with the *Wedge* representing more dense data and the *Tetrahedron* representing sparse data.

2.5.1 Test Suite I

The goal of test I is to probe the sensitivity of the algorithm to noise in the initial coregistration estimate. The convergence properties of the algorithm are tested with regard

to both the number of iterations and the quality of the final solution for a set of noisy initial coregistration estimates. Noise is introduced into the model-to-sensors orientation estimate by rotating a fixed amount (given in Table 2.1) around a randomly selected axis, the model-to-sensors translation estimate by adding a fixed amount along a random 3D direction, and the relative translation between the sensors by adding a fixed amount along a random *2D* direction. To characterize the inherent roundoff error, a baseline error E_{fit} is determined by running the algorithm with perfect pose+registration for each model.

A test consists of a specific model and a specific error parameter setting. Each test was run for 100 trials, with each trial starting with a different noisy initial estimate. The initial estimates were generated by rotating the sensor suite ΔR about a random axis, then translating it ΔT in a random direction. Then, the registration between the range and optical sensor is translated ΔT in the xy plane.

The results for each test are listed in Table 2.1. Runs which did not achieve a final E_{fit} less than 1.0 are assumed to be failures and are not considered in the averages.[5] The first test ($\Delta R = 0$ and $\Delta T = 0$) for each model established the baseline error, and was only run once. For all other tests, N is the number of successful runs out of 100, $\bar{Its}$ is the average number of iterations to convergence (for successful trials), and $\bar{E}_{fit}$ is the average error after convergence.

The first thing to be observed is that this algorithm tends to converge in very few iterations (< 5 for reasonable starting conditions). This is good, since solving the system of equations (at each iteration) is a relatively expensive operation. It also allows us to set the *maximum iterations* to a relatively small value, terminating unpromising evaluations rapidly.

For large translation errors (even $\Delta T = 500$), our algorithm successfully finds the correct solution 100% of the time. This is not surprising, since the translation portion of the error measure is linear. This indicates that we do not need to be terribly accurate in our initial registration between optical and range sensors.

The algorithm is noticeably less stable under large rotations. For rotations of 0.5 radians (28.6°), a modest success rate ($> 85\%$) was observed. For the more moderate rotation error 0.25 (14.3°), performance for three of the four models is excellent. However, the average error E_{fit} indicates that the *Trapezoid* for $\Delta R = 3.14$ and $\Delta T = 0$ and the *Wedge* for all but $\Delta R = 0$ exhibit some non-optimal convergences near the global minima.

These instabilities mean near-optimal solutions close to the true optima are being found. The error values for these are less than 1.0 and they are being included in the average E_{fit}. There are two explanations for this behavior: 1) local minima near the global minima or 2) plateaus in the error space. If there are local minima near the global minima, then not much can be done. However, plateaus can be dealt with by modifying Levenberg-Marquardt to allow small increases in error (due to roundoff error as the plateau is being traversed), while avoiding loops through the parameter space. This modification has not been tested.

---

[5] This experiment was performed early in the development of the coregistration algorithm. In retrospect, the criteria for selecting "correct" pose+registration recoveries should have been based upon the closeness to the correct pose, rather than upon E_{fit}.

2.5.2 Test Suite II

While the results from Test Suite I are encouraging, they assume perfect image data. This is never the case, since, even with the best feature extraction algorithm, there will always be some sort of discretization error. Test Suite II investigates the effects of noisy image data on the accuracy of the recovered coregistration parameters. Varying amounts of noise are added to the image measurements and then the recovered coregistration parameters are compared to the ground truth coregistration.

As in Test Suite I, each test consists of 100 trials for a given model and parameter set. However in this case, each coordinate in the image data was perturbed by Gaussian noise. This means that the range data has noise added to the distance component, as well as to the x and y coordinates. The amount of noise used in each test is listed in Table 2.2. The first two columns contain the standard deviation for the noise applied to the optical (Opt) and range (Rng) data.

The remaining six columns show the mean and standard deviation for the recovered model-to-sensors rotation error (from the ground truth), model-to-sensors translation error, and optical-to-range translation error. Rotations are reported in radians and translations in meters. Unlike Test Suite I, no success criteria was imposed on these error values, so the statistics include all 100 trials rather than only ones that are determined to have succeeded. This was done because we are interested in the movement of the global minimum when noise is present.[6]

The first test ($\sigma_C = 0.5$ and $\sigma_L = 0.5$) tested the case where discretization error is introduced into the sensor. This is a minimal amount of error, where the majority of the features fall within 0.5 pixels of their correct location. All of the errors were relatively small for this base amount of noise.

The remaining tests investigate noise introduced by feature extraction errors. As expected, the global minima in the parameter space moved more as the amount of image noise increased. For 5 pixels of noise in the optical, we have a reasonably stable solution. However, for 5 pixels of noise in the range, we have a solution with too much error for our purposes. This disproportionately large amount of error introduced by range error is due both to the more direct 3D constraint of the range as well as the larger arc length covered by a range pixel compared to an optical pixel.

When observing the standard deviation relative to the average, it can be seen that one of two things is happening, since the standard deviation is often nearly as large as the mean: either the distribution is nearly uniform or there are some extreme outliers. Given the nature of coregistration as an optimization algorithm, it would seem most likely that the variance is being caused by a few outliers.

Also, there is a noticeable difference between the models. The planar model (*Trapezoid*) has mean errors up to twice the level of the other models. This could be caused by the high levels of errors relative to the depth (which is 0). When image features are jittered by enough noise to move them to alternate sides of the object, many additional local optima can be created, or very shallow linear approximations may drag the solution off significantly.

---

[6] In retrospect, this data should be partitioned based upon closeness to the correct pose parameters. At the time of this experiment, it was not realized that a shallow basin of attraction could result in a large parameter update, yielding outliers.

2.6 Coregistering Real Data

As a demonstration, an M60 model and images nov31553c (color) and nov3120511 (LADAR) images from the Fort Carson data set [5] were coregistered from both good and poor initial estimates. Four corresponding points on the object model, in the color image, and in the LADAR image were hand picked using the Rangeview program described in [14]. The four pairs of corresponding points were used to generate six corresponding pairs of line segments for the model-to-CCD error term. These corresponding features, intrinsic camera parameters, and an initial coregistration estimate were passed to the coregistration algorithm (see Figures 2.6(a) and 2.7(a)).

In the case of a good (i.e., what we hope to be normal) initial estimate, as is seen in Figure 2.6(a), we see reasonable convergence to a nearly correct[7] solution. The initial estimate places the model nearly aligned with the LADAR data, with a moderate rotation. The color data is noticeably misregistered. In 8 iterations, the final solution shown in Figure 2.6(b) is obtained. This is close, though slightly rotated under. This is due to the feature selection. The model features were hand-computed, and so may not lie exactly on the vehicle surface. In the case of the image data, the hand selected points may not correspond exactly with the model features. Also, the model features are nearly planar, resulting in increased instability, as we saw in the synthetic Test II.

Even in the extreme case (Figure 2.7(a)), the coregistration algorithm converged to a solution (Figure 2.7(b)) in 6 iterations. In the initial estimate, the model-to-optical alignment and the optical-to-range registration are far off. There is only a small rotational error. After coregistration, a similar, but not identical set of coregistration parameters has been found. The convergence to different, but similar solutions is due to the hand-picked features.

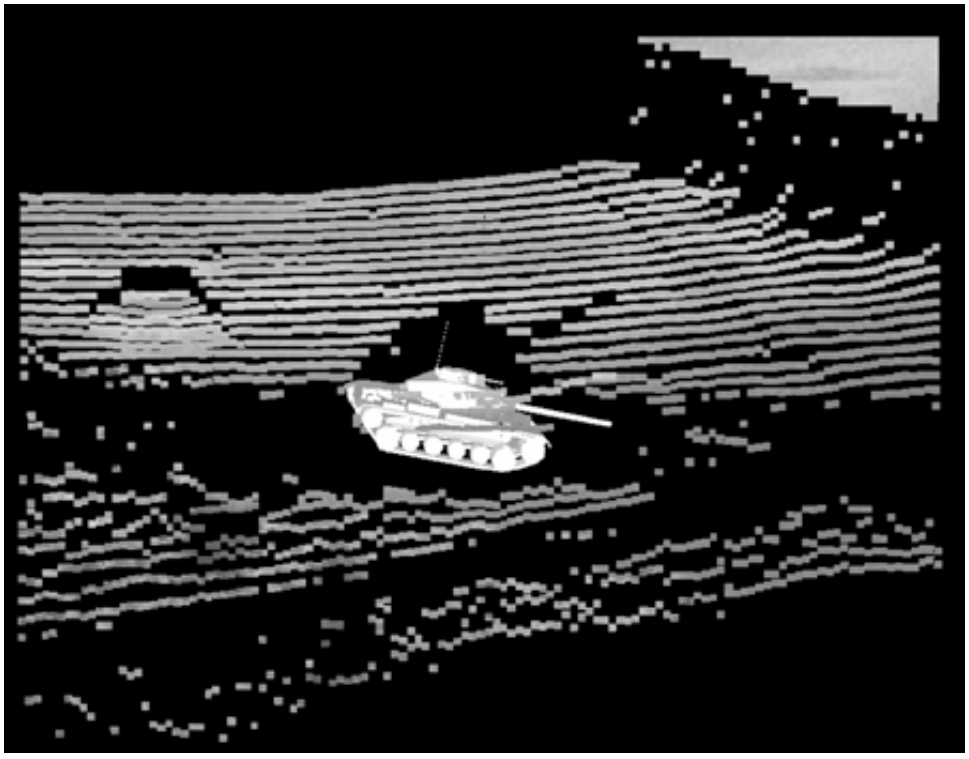
2.7 Conclusions

The simple coregistration algorithm has been shown to converge even with a very large error in the initial pose+registration estimate. Though there is slight variation between the models, the algorithm converges well for all of them. However, when noise is introduced into the image features, differences appear between models. Most noticeably, the planar model (*Trapezoid*) appeared up to twice as sensitive as any of the other models. This indicates that model feature selection insuring a large degree of non-planarity is important to the convergence of our algorithm.

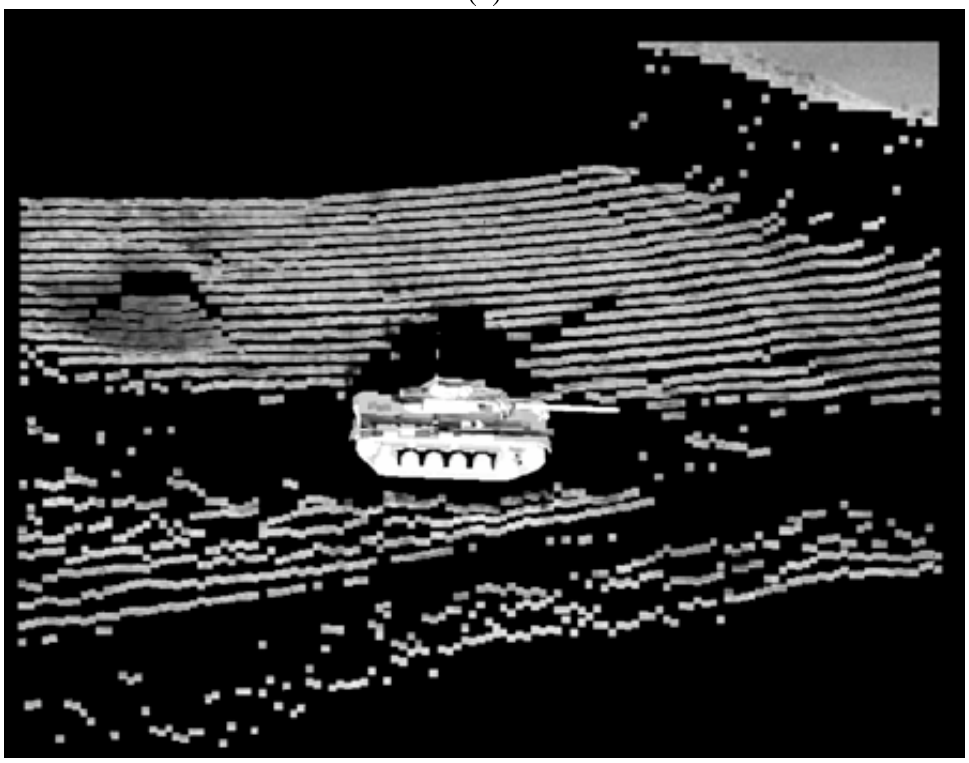
Also, coregistration appears very sensitive, as all least-mean squares algorithms are, to large amounts of noise (e.g., outliers). Starting from two initial estimates, two close but different final pose+registration results were achieved. This indicates that hand selected correspondences are a poor choice. To reasonably use coregistration, automatic correspondence construction is a must, and this is the topic of the next chapter.

---

[7]The term "correct" simply means that the final solution "looks good." There is no recorded ground truth for the Fort Carson data set.

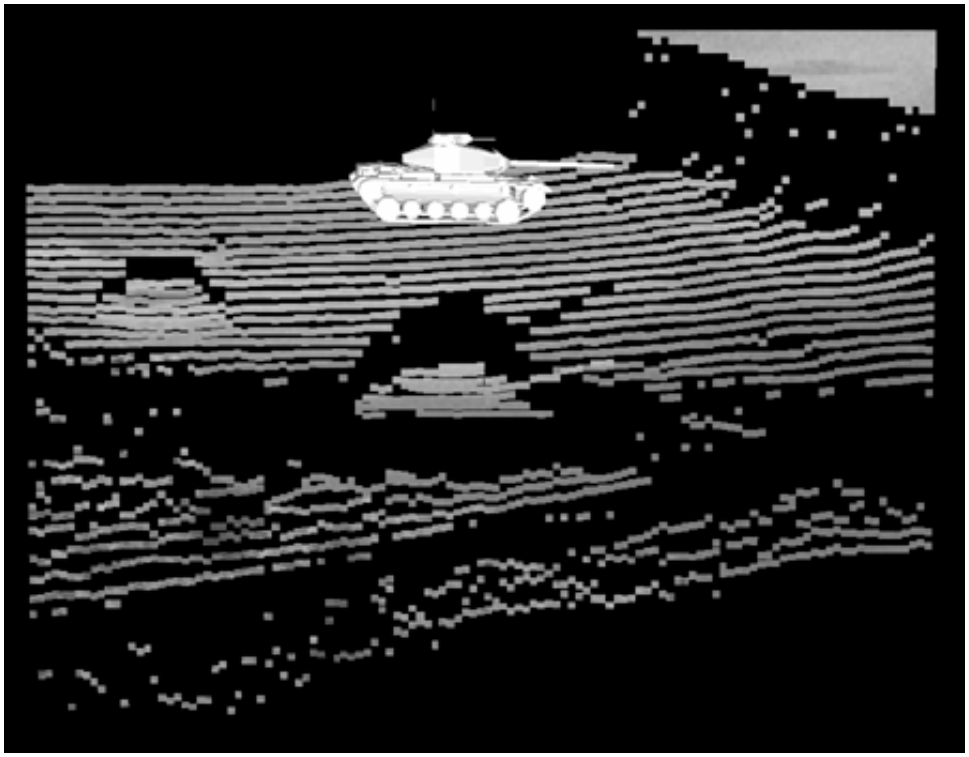


(a)

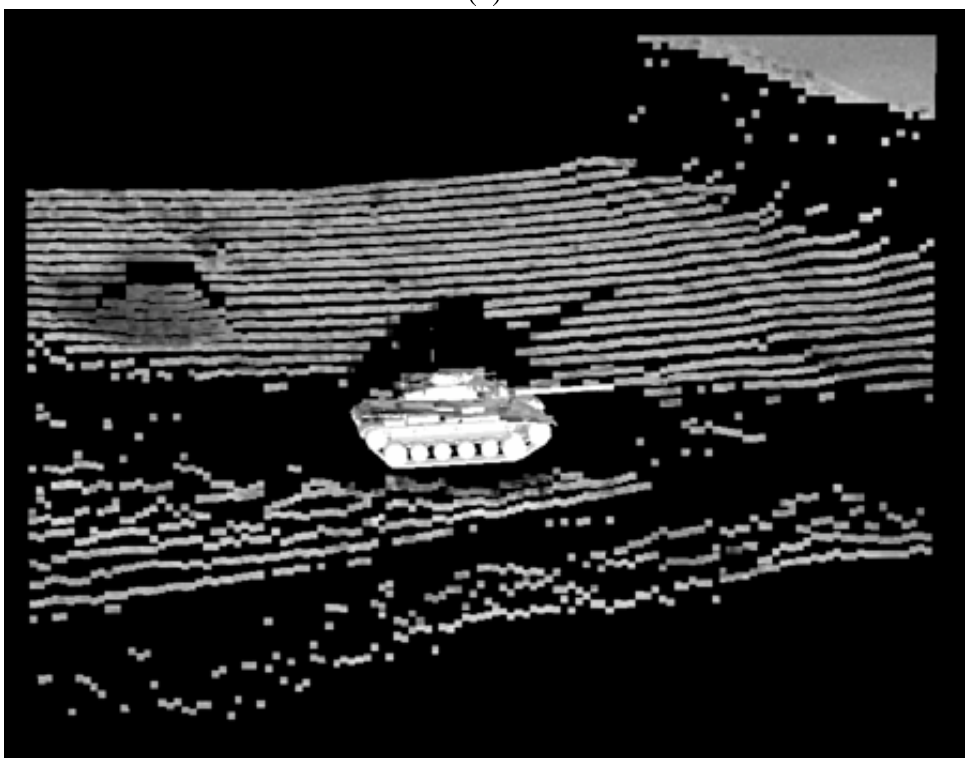


(b)

Figure 2.6: (a) Normal initial and (b) final pose+registration using coregistration and hand-selected features.



(a)



(b)

Figure 2.7: (a) Extreme initial and (b) final pose+registration using coregistration and hand-selected features.

Model	ΔR	ΔT	N	$\overline{\text{Its.}}$	$\overline{E_{fit}}$
Trapezoid	0	0	1	-	6.016e-06
”	0	40	100	3.00	5.987e-06
”	0.50	0	100	3.97	5.976e-06
”	0.25	20	100	3.00	5.992e-06
”	0.50	40	100	3.96	5.966e-06
”	0.90	100	88	4.95	5.968e-06
”	3.14	0	16	9.00	6.275e-03
”	0	500	100	3.68	6.011e-06
Cube	0	0	1	-	3.593e-07
”	0	40	100	3.00	3.594e-07
”	0.50	0	100	3.95	3.560e-07
”	0.25	20	100	3.22	3.623e-07
”	0.50	40	100	3.95	3.560e-07
”	0.90	100	100	5.78	3.586e-07
”	3.14	0	91	10.20	3.596e-07
”	0	500	100	3.00	3.594e-07
Wedge	0	0	1	-	2.172e-06
”	0	40	100	3.00	2.120e-06
”	0.50	0	90	4.85	4.179e-01
”	0.25	20	93	4.23	3.355e-01
”	0.50	40	92	4.90	4.281e-01
”	0.90	100	79	5.91	5.102e-01
”	3.14	0	59	10.27	4.115e-01
”	0	500	100	3.84	1.252e-01
Tetrahedron	0	0	1	-	8.417e-07
”	0	40	100	3.00	8.414e-07
”	0.50	0	86	4.10	8.379e-07
”	0.25	20	100	3.42	8.445e-07
”	0.50	40	86	4.18	8.378e-07
”	0.90	100	92	5.10	8.406e-07
”	3.14	0	35	8.28	8.412e-07
”	0	500	100	3.46	8.414e-07

Table 2.1: Test Suite I Results

Noise σ		$\lvert E_{R_{mo}} \rvert$		$\lvert E_{T_{mo}} \rvert$		$\lvert E_{T_{or}} \rvert$	
Opt	Rng	Avg.	Sd.	Avg.	Sd.	Avg.	Sd.
Trapezoid							
0.5	0.5	0.009	0.006	4.2	3.3	0.04	0.03
1	1	0.019	0.013	8.4	6.6	0.08	0.07
5	0	0.033	0.021	0.9	0.7	0.17	0.10
0	5	0.095	0.074	43.3	36.7	0.40	0.32
5	5	0.102	0.070	43.8	34.8	0.45	0.34
20	0	0.121	0.080	3.7	4.3	0.70	0.41
0	20	0.390	0.294	178.6	142.4	1.61	1.31
Cube							
0.5	0.5	0.005	0.004	2.4	2.0	0.00	0.01
1	1	0.009	0.007	4.3	3.6	0.01	0.02
5	0	0.006	0.010	2.0	3.0	0.01	0.03
0	5	0.059	0.032	28.6	16.5	0.10	0.10
5	5	0.045	0.034	20.0	16.1	0.07	0.10
20	0	0.031	0.044	9.8	14.4	0.07	0.22
0	20	0.239	0.137	114.7	67.6	0.48	0.65
Wedge							
0.5	0.5	0.004	0.003	1.9	1.9	0.01	0.01
1	1	0.007	0.007	3.8	3.7	0.01	0.03
5	0	0.005	0.009	2.2	3.9	0.01	0.03
0	5	0.049	0.026	24.4	13.7	0.06	0.11
5	5	0.042	0.035	20.3	17.6	0.08	0.15
20	0	0.024	0.037	9.9	15.9	0.01	0.07
0	20	0.188	0.094	91.6	48.5	0.43	0.78
Tetrahedron							
0.5	0.5	0.008	0.005	3.9	2.7	0.02	0.02
1	1	0.016	0.011	7.9	5.4	0.04	0.05
5	0	0.010	0.015	3.6	5.3	0.02	0.07
0	5	0.087	0.055	42.2	26.9	0.20	0.27
5	5	0.084	0.058	39.7	27.6	0.21	0.30
20	0	0.062	0.090	21.0	33.6	0.19	0.43
0	20	0.339	0.276	158.9	127.2	1.00	1.21

Table 2.2: Test Suite II Results.

Chapter 3

Building Correspondences

The notion of correspondences arises naturally from the physical sensing system (sensor and scene). Each element in the image, whether it is low-level (e.g., pixel) or higher-level (e.g., line), can trace its origin back to a physical process in the scene. In the case of model-based matching, this means that an image feature can be paired with the portion of the model which generated it.

As we saw in the previous chapter, robustness of a least-*mean* squares method for computing pose is dependent upon establishing the correct (if incomplete) correspondence set. A poor match will likely produce a exceedingly poor pose and registration.

Here, when we talk of a "poor" match, we are referring to the distribution of the residual error, given the ground truth pose+registration. As with all least-mean squares methods, the noise process is assumed to be both additive and Gaussian. If we assume this process model, we can label all measurements/correspondences not well described by it as *outliers*. It appears that, even in the case where the Gaussian noise has a large standard deviation, it is better to set the threshold lower, to increase convergence of the least-squares algorithm to the correct pose (see Table 2.2).

This chapter considers two main paradigms of generating outlier-free matches. First, a robust statistical outlier removal method, such as *least-median squares* (*median filtering*), may be applied to a hypothesized match. Given enough time, such a method can guarantee the correctness of the match, assuming that enough of the data (50% for median filtering) comprises a good correspondence. Such methods rely heavily on a fairly accurate initial pose+registration hypothesis, so they are not appropriate for matching when the object pose is highly uncertain.

Another option is to apply a non-statistical search technique, such as *local search*. This approach trades off the statistical robustness of median filtering for usability in realistic search spaces. It can deal with more than 50% outliers in the correspondence space.

3.1 Features

One of the key issues only hinted at up to this point is the extraction of features from a CAD model and from an image pair. The main point of building good correspondences (i.e., matching) is to pick the important features out of the clutter and associate these image features with their corresponding model features. A key question, though, is how to find these features in the first place.

While feature detection is not the main point of this thesis, the methods of feature extraction are pivotal to the interpretation of the results. As a consequence, I will present here and again in Section 4.1.1 feature extraction methods.

By dealing with both color and range data, a large disparity appears in what is represented between the two image types. In the color data, intensity is a function of the light source (the sun), as well as other objects in the environment (trees, clouds, tank barrel) and the pigmentation (camouflage paint, chlorophyll). In range data, measurements of the surface distance are directly taken using a laser. This means that the data is primarily a function of the distance and secondarily of the surface material. It also has the advantage of being 3D data, so a "true" point in the world can be reconstructed. Given the differences in the optical and the range modalities, we chose two separate extraction methods, one for color and one for range.

For color, we utilized Stevens's model driven line extraction [41]. Here, the silhouette and important interior lines are first extracted from the model given the hypothesized model-to-camera pose. These lines are then placed over the image and decoupled (a version with coupled model features is introduced in [40]). Each line is locally moved over the color/optical image to optimize the gradient under the edge. The 2D mapping of the optimal model line placement is the reported image feature.

We attempted to use other line extraction methods, such as Burns lines [11], but found that the high texture of these outdoor images produced too many features and often significantly pulled the true silhouette lines off target. We used Burns line extraction on the raw color images and on median filtered images and found that the median filtered images were more unusable than the lines from raw images.

For range, we simply used all the raw range points within a window about the hypothesized vehicle position. While these are within the 3D perspective system of the LADAR sensor we used, they are easily unwarped into a Cartesian system. A key point to remember when unwarping the data is that the uncertainty varies with depth, which is not necessarily true when dealing with the raw, perspective data. This issue will be addressed again when we consider Kalman Filtering solutions in Section 4.2.

3.2 Least Median Squares

Least-*mean* squares optimization minimizes (or maximizes) the mean of the squared residual errors by selecting advantageous values for the function parameters (e.g., pose+registration parameters). Analogously, least-*median* squares (also called "median filtering") minimizes the median squared residual error in the correspondence. In this case, however, the parameter space being optimized is the set of combinations of correspondence pairs, rather than the pose+registration parameters.

The least-median squares algorithm is outlined in Figure 3.2 for the specific task of constructing correct correspondences for coregistration. First, an initial correspondence is established. This match contains both correct and incorrect feature pairs, and at least 50% of these pairs must be correct feature pairs.[1] This initial correspondence can be based upon the initial pose+registration estimate, as is done in this paper.

---

[1]The closer to 50% outliers you get, the less chance you have of getting an outlier-free subset. This means that you want to have as few outliers in the initial correspondence as possible. If you cannot start

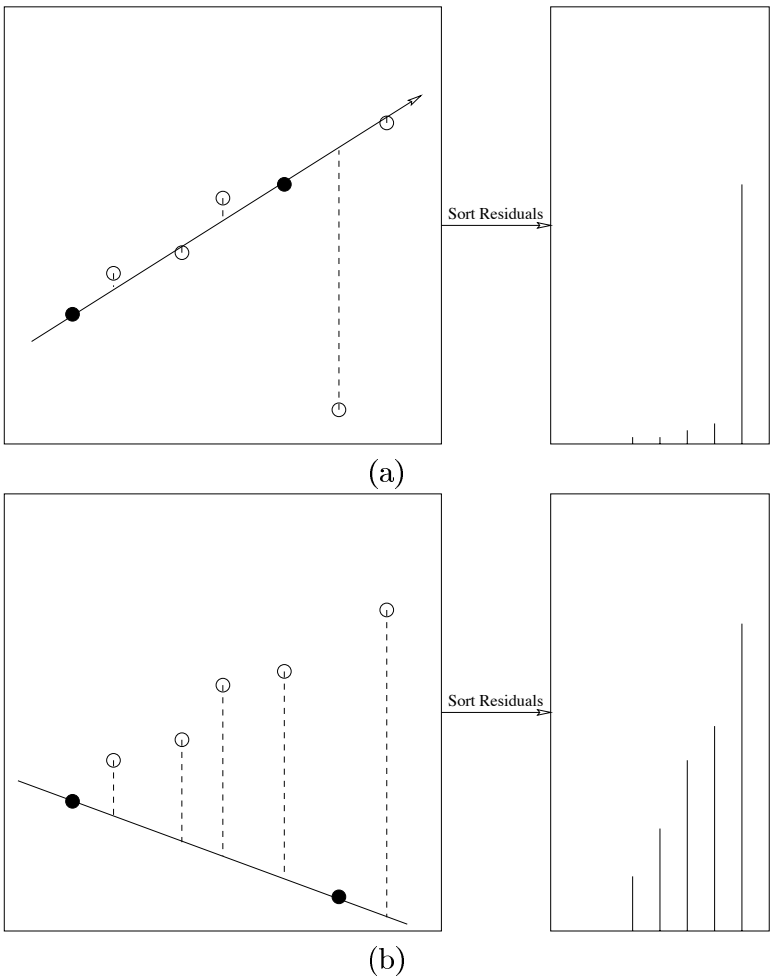


Figure 3.1: Fitting a line to subsets of data. Subset comprised of (a) all good data and (b) some outliers.

Next, subsets of the initial correspondence are selected. The idea is that, if enough small subsets of the initial correspondence are picked, one of them will probably contain no incorrect (outlier) pairs. Coregistration is run with each of these subsets. If a subset contains no outliers, then this will yield the correct pose+registration. Figure 3.1(a) shows an example of this. Notice that the data which corresponds well to the line results in low residual errors, while the outlier yields a large residual error. If a subset contains outliers, then the recovered pose+registration parameters should be incorrect. In Figure 3.1(b), the inclusion of an outlier yields a very poor fit. While the two points used to fit the line have zero residual error, the remaining points contribute very large errors.

From these previous two examples, we can see that we want to pick the pairs with low residual errors based upon the good subset. But how do we pick the outlier-free subset? As in Figure 3.1, the model (the line) is fit to the subset data. Once this is done, the

---

with few outliers, then you must try more subsets. As the percent outliers increases, the number of subsets increases exponentially for a given confidence level.

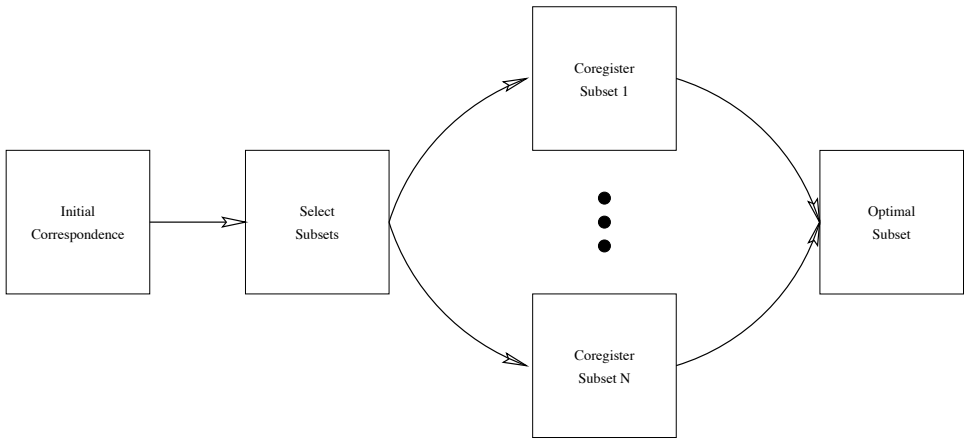


Figure 3.2: Block diagram of the least-median squares (median filtering) algorithm.

residual error (squared Euclidean distance) can be computed for each pair in the initial correspondence, using the new "optimal" positioning of the line. The median residual is then selected. If the residuals are thought of as the square of a Gaussian distribution, then the median can be thought of as the bounds (positive and negative) of the interquartile range. All pairs with a residual error[2] above $\texttt{cutoff} = (a \times s)^2$ where $s = \frac{\min \tilde{E}_{fit}}{0.6745}$ is an approximation of the standard deviation for a Gaussian distribution based upon the interquartile range. a is the number of standard deviations to set the cutoff at and is commonly set to $a = 2$. [27, 38]

3.2.1 Results

To demonstrate least-median squares optimization, an image suite was selected from the Fort Carson data set [5]: Shot 20 Array 5. The initial pose+registration was selected to fit the data as accurately as possible. This was necessary to ensure an initial correspondence with less than 50% outliers.

The initial correspondence was selected by first mapping the model features into the color and range image space based upon the initial pose+registration. In the color image, lines within $\tau_{mo\theta}$ degrees of the model line and with distance less than τ_{mod} were paired in the initial match. In the range image, model-image point within $\vec{tau}_{mrd}$[3] are included in the initial match.

The coregistration and median filtering parameters are shown in Table 3.1. The thresholds τ_{mo} and τ_{mr} were based upon the 50 percentile for color and the 50 percentile for range, as described in Section 2.3.3. These values were tuned slightly (keeping the same order of magnitude) to trade off omission of good features the inclusion of questionable features.

---

[2]Officially this should read "squared error," but the error used here is already the square distance.

[3]The x, y coordinates are in pixels and the z coordinate is in meters.

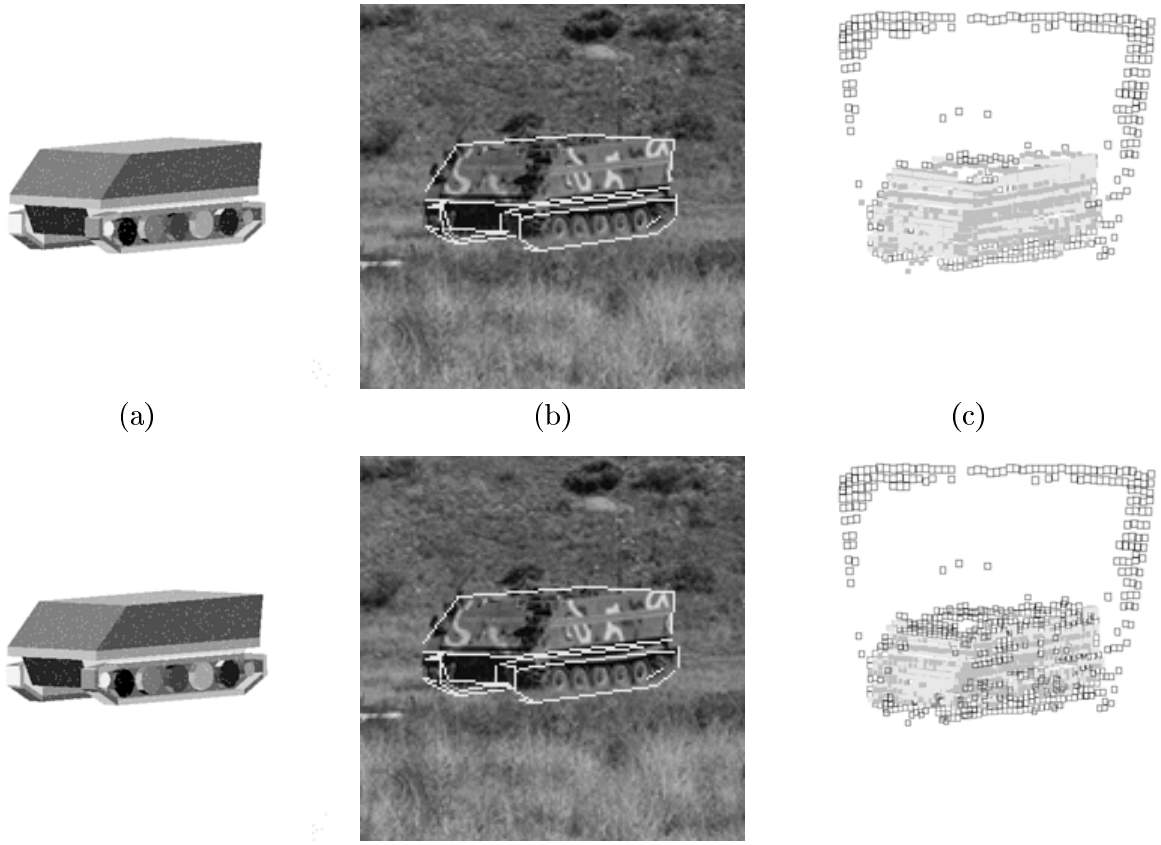


Figure 3.3: Least-median squares constructed correspondence using Shot 20 Array 5. (a) Initial model pose, (b) initial color features, (c) initial range features, (d) final model pose, (e) final color features, (f) final range features.

The initial and final pose+registration is shown in Figure 3.3. In Figures 3.3(b) and (c), the initial positioning of the model features is shown in red and the data features, in blue). All of the median filtering and matching images use this same color coding, with the addition that model features which are in the match are shown in yellow and model features in the match are shown in light blue. The correspondence between individual features is not explicitly shown.

Notice that, in this demonstration, the pose does not significantly alter. This is a good indication that the selection of features is indeed outlier-free.

Notice in 3.3(c), that the roof of the vehicle contains very sparse data, due to the viewing angle. The data points which are paired are significantly further apart than in the rest of the data. Median filtering correctly removes these from the matching (see Figure 3.3(f)).

Not only does it remove features which should be removed, but it also preserves features which are a good match. In the color imagery (see Figures 3.3(b) and (e)), the line segments are very good matches, due to the model driven feature selection. None of these features are dropped.

Coregistration	
α_{fit}	0.5
τ_{mo}	3.0
τ_{mr}	0.5
Initial Correspondence	
$\tau_{mo\theta}$	20°
τ_{mod}	10
$\vec{\tau}_{mrd}$	$(0.5, 0.5, 10)$
Least-Median Squares	
a	2.0
subsets	500

Table 3.1: Coregistration and least-median squares parameters.

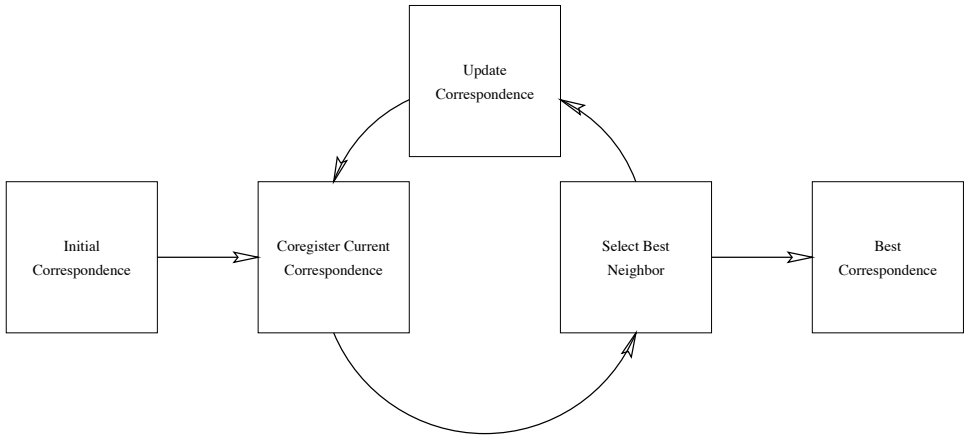


Figure 3.4: Block diagram of the local search algorithm.

However, this required that the initial pose+registration parameters be very good. In tests where the estimation was worse and a larger initial correspondence was considered, the number of outliers was too large and the system did not converge. This means that least-median squares, while good for validation processes, does not prove useful for matching tasks where the pose+registration parameters are not well known.

3.3 Local Search

The least-median squares method offers assurances as to the probability of finding the optimal correspondence. In fact, the longer this statistical method is run, the more likely it is to have found the optimum. Unfortunately, it requires a nearly perfect pose+registration estimate to fulfill the $< 50\%$ outliers assumption.

An alternative is to trade off the assurances of least-median squares for the search space traversing properties of local search. This search mechanism makes no demands on the amount of outliers, allowing even the complete space of all possible matches (image features $\times$ model features) to be considered. [2] This power does come with its price: the convergence to non-global optima may be likely. This can be partially overcome by restarting the search from a number of local positions, but does not make as strong of

a statement about convergence as median filtering, since the portion of the search space containing the global optimum may not be on many paths. [2]

3.3.1 Defining a Match Error

So far we have defined a measure of how well the model fits the data, given a set of coregistration parameters and a correspondence. If we allow the number of corresponding features to vary using only this measure, we would soon find that the optimal correspondence consisted of a small number of correspondences with zero total fit error. As a result, we also need to consider another error term which offers a penalty for not explaining model features. As Beveridge [2] did, we combine the omission error (E_{om}) with the fit error to form a total match error per sensor of

$$E_{match,sensor} = E_{fit,sensor} + E_{om,sensor}$$

and the total match error for the suite of

$$E_{match} = \alpha_{match} E_{match,o} + (1 - \alpha_{match}) E_{match,r}$$

Note here that α_{match} is the same as α_{fit}; it is simply the proportional importance of each sensor.

We consider omission to be the process by which some features or portions of features present in the model are not observed in the image. Computing the penalty for omitted range points is the simpler of the two. We simply consider the percentage p of points in the model which do not have corresponding points in the image. Since we expect not to observe every single model point in the image, we don't want to overly penalize the system for omitting a few points. however, as increasingly more model points go unobserved, we need to severely increase the error measure. We do this using an exponential function.

$$E_{om,r} = \left\{ \begin{array}{rl} \frac{e^{\alpha p}-1}{e^{\alpha}-1} & \alpha \neq 0 \\ p & \alpha = 0 \end{array} \right\}$$

We use an attenuation parameter a to control the curvature of the exponential function.

$$\alpha = 2 \ln \left(\frac{2}{a} - 1 \right)$$

If we set a to 1.0, we observe a linear omission function, but increasing a decreases the penalty for small amounts of omission. We consider the linear form in the absence of domain knowledge such as expected sensor dropout rate, as this avoids uninformed bias.

We similarly consider the omission error for the optical sensor. Here, however, rather than considering the collection of all model points, we consider the omission related to individual line segments. By applying the exponential error weighting function for each line, we penalize the exclusion of complete lines more than the omission of small fractions of many lines, even though the total percent features lost is the same. This is desirable, since we expect to have difficulty extracting the ends of lines, but we also expect to not completely overlook features. Consider, for example, the case where we have a square as the model. We expect an image with 75% of each of these four lines to have a lower omission measure than with 100% of each of three lines, even though the total line coverage is equal. We can state a line omission error which embodies these requirements

$$E_{om,o} = \frac{1}{|M_o|} \sum_{i \in} \left\{ \begin{array}{rl} \frac{e^{\alpha p_i}-1}{e^{\alpha}-1} & \alpha \neq 0 \\ p_i & \alpha = 0 \end{array} \right\}$$

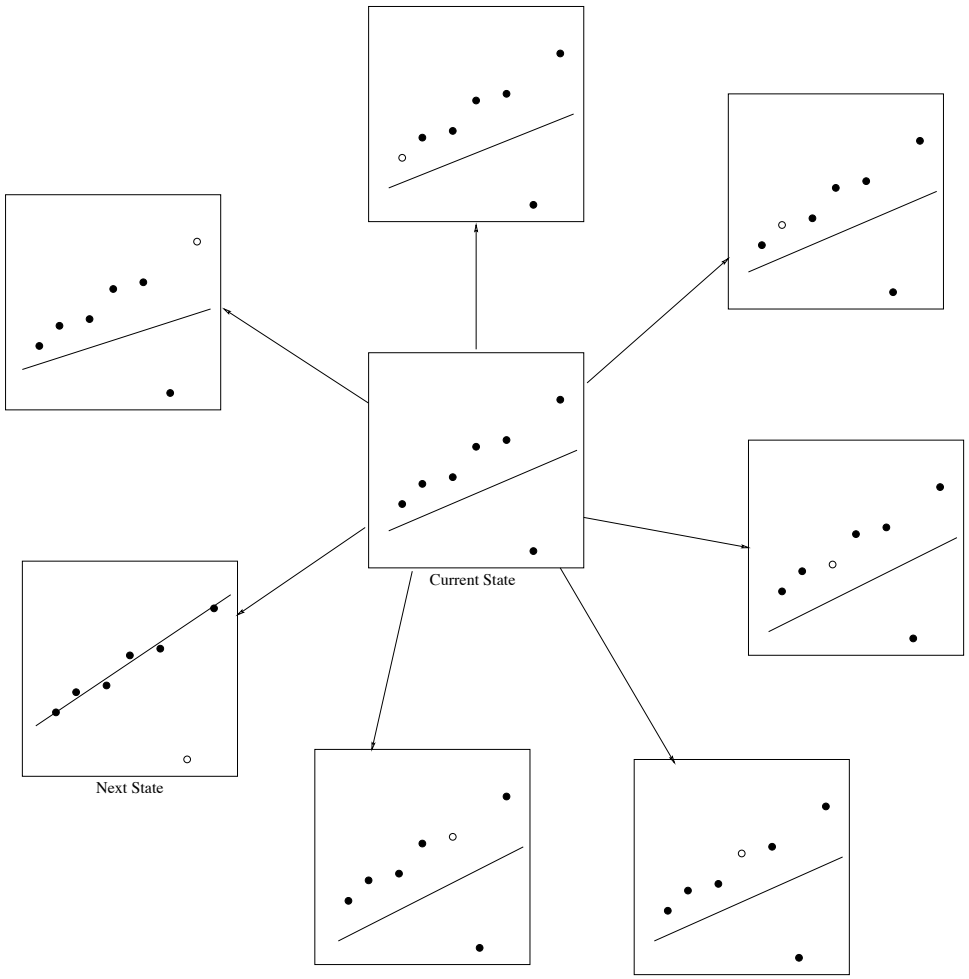


Figure 3.5: Example of local search in the line domain.

Using this match error formulation leaves us with five parameters: τ_o and τ_r for controlling the expected noise threshold, a_o and a_r for attenuating the omission response, and $alpha_{match}$ adjusting for the importance of each sensor.

3.3.2 Local Search

In local search, the state space is traversed using a series of moves. In our case, the state space is the space of all correspondences ($2^{|M|\times|D|}$ states). First, we need to define an initial state. We generate our initial state based upon the initial coregistration parameters, projecting the model into the range and optical sensor reference frames. Using distance and rotation thresholds for the features, we pair all features that fall within the thresholded area. The set of all of these correspondences makes up our initial state. This reduces the size of the search space, yielding a more tractable search space. An alternative is to randomly select pairs in the complete correspondence to construct the initial match [2].

Given this initial state, we can define a set of moves for traversing the search space to an optimal solution: in our case, removing a single correspondence (see Figure 3.5). This simple minded operator allows us to thin out our correspondence, removing outlier data. This also has the advantage that, as the algorithm runs, the neighborhood shrinks, short-

Coregistration	
α_{fit}	0.5
τ_{mo}	3.0
τ_{mr}	0.5
Initial Correspondence	
$\tau_{mo\theta}$	20°
τ_{mod}	10
$\vec{\tau}_{mrd}$	$(0.5, 0.5, 10)$
Local Search	
α_{mo}	0.5
α_{mr}	0.5

Table 3.2: Coregistration and local search parameters.

ening the time required to select the next move. This is desirable from the standpoint of a first pass, although a more complete neighborhood would prove more robust and practical once the inner loop (i.e., coregistration) has been optimized. Some other neighborhoods have been proposed, such as 1- and 2- Hamming distance [2], but these search a more interconnected space (a mesh, rather than a tree).

Once we have evaluated every possible move we can make from the current state, we need to select the move which we believe will bring us closer to an optimum. There are several heuristics such as first improvement (which lessens search time per move) and steepest descent. We choose to use steepest descent, since previously Beveridge empirically found this to work well on local search matching involving a single sensor.[2]

When we discover that none of the neighbors for our current state offer an improvement, we know that we have reached a local optimum and we stop. While we have not implemented this, it is possible to restart the search using a number of random subsets of the initial correspondence. This would improve our odds of finding the global optimum, if we were using a more connected search space. However, since we can only remove correspondences, we simply start from the complete thresholded correspondence set and only search once.

3.3.3 Results

To investigate the performance of local search, compared to median-filtering, Shot 20 Array 5 was used in conjunction with local search to construct "optimal" correspondences. The initial pose+registration parameters and initial correspondence are the same as those used in the least-median squares method. The parameters used for both coregistration and local search are found in Table 3.2.

Local search begins with a set of 672 feature pairs. This causes local search to be infeasible. Local search requires more invocations of coregistration to determine a single next state than were required for an entire run of least-median squares for this particular problem. The time to coregister the data is also linear in the number of features. The combination of these two yielded a problem which I could not solve, due to lack of computational resources.

3.4 Conclusions

Correspondence building methods such as least-median squares and local search are necessary to utilize a coregistration algorithm, such as the one proposed in Chapter 2. Each of these methods is appropriate in specific domains. Least-median squares is appropriate when the initial estimate is accurate. Local search has the capacity for dealing with more outliers, but is impractical given the current search space size.

Chapter 4

More Coregistration Methods: Feature Grouping and Kalman Filtering

The simple coregistration algorithm from Chapter 2 contains several limiting features. First, some convergence problems exist in the case where the data is nearly planar and contains noise. Also, there appears to be general problems with convergence to the global minima when moderate noise exists in the data. Second, in Chapter 3 it was shown that, while primitive matching can be performed on such dense data, it is not stable or tractable. Since most of the data comes from the range data, point-to-point matching is considered to be a poor representation.

While not the emphasis of this thesis, two more approaches other than the previously described Least-Mean Squares coregistration formulation are presented in this chapter to address these problems. Due to time constraints, these formulations presented here have not been implemented yet, but they are included to suggest solutions to some weaknesses in the current coregistration formulation and to point towards future enhancements.

First, we will overcome the feature-space problem described in Chapter 3 through the use of feature grouping. We consider the grouping of range pixels into linear segments as a precursor to planar surface extraction. These linear segments are then used in a line-to-plane based criterion function.

Then we will discuss the use of Kalman Filtering as a replacement for explicit Least-Mean Squares methods. A brief tutorial on Kalman Filtering is provided to minimize hardship. Then an extension to Hel-Or and Werman's sensor fusion [19, 20] will be introduced, which considers constrained articulation between sensors. This both addresses the problem of converging to good solutions [1] and the inclusion of arbitrarily constrained sensor suites.

---

[1]There is, of course, still the problem of converging to local minima, but this appears to be less of a problem in the Kalman Filtering framework using the notes on stable solutions pointed to in [19].

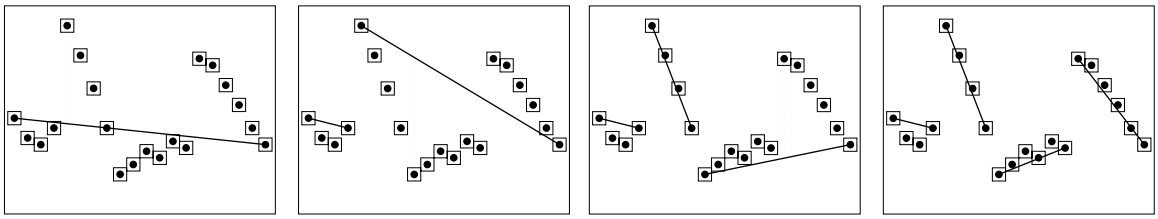


Figure 4.1: Four iterations of scanline segmentation.

4.1 Range Feature Grouping and Least-Mean Squares

Due to the density of features in the range/point representation addressed in Chapter 2, generation of correct matches is difficult, if not impossible. Removal or addition of a single feature has little effect, even assuming that, within a given domain, a good enough initial hypothesis can be generated, the time necessary to evaluate and prune the space using median filtering is impractical.

To address this problem, the grouping of range features is introduced into the coregistration error. Historically, range data is collected into planar patches. Hoffman and Jain [21], Jiang and Bunke [25], Masuda and Yokoya [31], Nadabar and Jain [32], and Parvin and Medioni [33] all provide variations on the theme of range image segmentation. In general, these methods are based around region growing with the grouping criteria based on curvature, slope, and continuity. Unfortunately all of these methods require significantly more data on target than we have available.

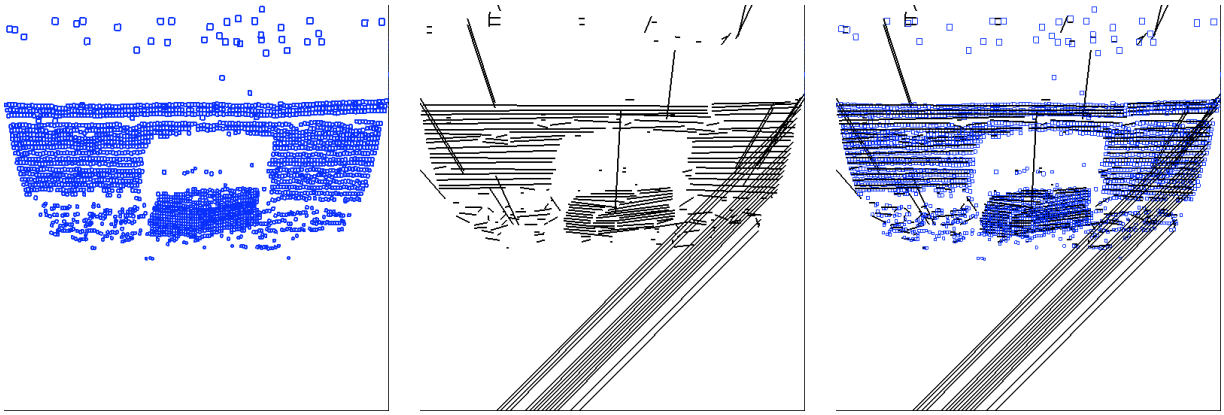
4.1.1 Linear Segment Extraction

Jiang and Bunke [25] provide the key to overcoming this data quantity problem. Their two-phase algorithm begins by collecting near-linear segments along scanlines. These scanline segments are then grouped into planar surfaces based upon a region growing criteria, similar to that found in more traditional region segmentation methods.

The algorithm for extracting linear segments from range data is shown in Figure 4.3. This is the same as the first phase as Jiang and Bunke's [25] planar surface segmentation algorithm. In their algorithm, after they have extracted linear segments along scanlines, they use region growing to produce planar regions. We often do not have large enough planar regions in our images to stably grow regions. As a consequence, I propose to simply use the linear segments.

Looking at the algorithm more closely, we see that the algorithm can be broken into three main steps: calculation of the threshold based on the image variation (sort of a preprocessing step), a splitting stage where scanlines are broken into linear segments, and a cleanup phase where breakpoints are adjusted to decrease error in the linear fit.

To better explain the algorithm, we will walk through the example shown in Figure 4.2. In these images, we are looking down on a single scanline, so the pixel number varies across the x-axis and the range value varies along the y-axis. The first step is to create a line segment between the endpoints of the lines, as shown in the leftmost image. Here, a dotted line is drawn to show the greatest point-to-line distance. This point is selected as the breakpoint and two new segments replace the old one. The breakpoint is included in the segment to which it is closest (i.e., the breakpoint is grouped with the neighbor



(a) (b) (c)

Figure 4.2: Linear segmentation of real data. (a) contains the original range data, (b) contains the line segments extracted using the first phase of Jiang and Bunke's [25] algorithm, and (c) combines (a) and (b).

with the nearest range value). Moving to the second image, we see the two new segments. Again, we see the point furthest from the line segments are marked with a dotted line. The potential breakpoint in the leftmost segment is within the distance threshold, so this segment is considered a good approximation of its data. The right segment, however, does not fit its data well, so we break this line again. In the next two images, we see another breakpoint found and two more segments formed more breaks occurring. In the final image, we see that all of the data is well fit by the segments: no more new segments can be formed.

A cleanup procedure is then introduced to shift the breakpoints, thereby improving the fidelity of the segments. The basic idea is to see whether the line fit is improved by shifting the breakpoint left or right on pixel. This is repeatedly done for all segments until no more corrections are made. In the example, the first iteration shows us that shifting the endpoints would not improve the fit. The only place where moving the endpoints will improve the fit is in the case of curved or polygonal scanline signatures.

4.1.2 A Line-to-Plane Fit Error

Similar to the error equation for the optical line data, we will be defining a line to plane distance as the sum of the distances for each of the line endpoints to the plane. Now, however, the error is considered in terms of *data* line endpoints to *model* planes. This is the reverse of the situation found in the previous optical error. The new range fit error for feature pair c and model-to-sensor pose $\mathcal{F}$ is denoted

$$E_{fit,r}(\mathcal{F}, c_i) \quad = \quad \sum_{j=1}^{2}(((\vec{p}_{ij} - \mathcal{F}(\vec{\rho}_i)) \cdot \mathcal{F}(\hat{(n)}_i))^2$$

Here, we have denoted the model plane (m_i) in point-normal form as $(\vec{\rho}_i, \hat{n}_i)$. Transforming this measure will compute the distance between each of the endpoints (1 and 2) and the model plane. It is important to note here that we treat $\vec{\rho}_i$ as a point and $\vec{n}_i$ as a vector. This means that transforming $\vec{\rho}_i$ with $\mathcal{F}$ applies both rotational and translational components of

- For each pixel:
  - Incorporate its variance from its neighbors if it doesn't appear to be an edge.
- For each scanline:
  - Add a line running from the leftmost to the rightmost measurement.
  - Repeat until $d_{\max} < \tau$ for all segments:
    - Compute $d_{\max}$, the maximum distance from the current segment to the data.
    - If $d_{\max} \geq \tau$:
      - Break the segment at the column where d_{max} was found, creating two new segments to replace the old.
  - For each pair of adjacent segments:
    - If moving the breakpoint left or right improves the line fit:
      - Move the breakpoint to better fit the two segments.

Figure 4.3: Linear segment extraction algorithm for range data.

the transformation to the point. Transforming $\vec{n}_i$ simply applies the rotational component of $\mathcal{F}$, since vectors are not positional, but only directional.

When we compare this measure against the similar fit error proposed in Chapter 2, we find one key difference: the transformation is being applied to the plane rather than to the point. The problem with this is that it creates additional nonlinearities. In the old optical error, these nonlinearities were not introduced, since the transformation was from the line data to the plane data coordinate system.

To correct for this, we can consider the inverse of the model-to-sensor pose, $\mathcal{F}^{-1}$. This is actually the transformation we need to be solving for in Kumar's more robust R_AND_T_MOD described in [27]. We adopt this measure when formulating the new joint measure. This allows us to characterize the range fit error as

$$E_{fit,r}(\mathcal{F}^{-1}, c_i) = \sum_{j=1}^{2}(((\mathcal{F}^{-1}\vec{p}_{ij} - (\vec{\rho}_i)) \cdot (\hat{(}n)_i))^2$$

and the optical fit error as

$$E_{fit,o}(\mathcal{F}^{-1}, c_i) = \sum_{j=1}^{2} \frac{\lambda_i}{M_i^2} \left(\mathcal{F}_R^{-1}(\frac{d_{ij,x}}{s_x}, \frac{d_{ij,y}}{s_y}, 1) \cdot (\vec{m}_{i2} - \mathcal{F}_{\vec{T}}^{-1}) \times (\vec{m}_{i1} - \mathcal{F}_{\vec{T}}^{-1}) \right)^2$$

where

$$M = \sqrt{\left(\frac{A_x}{s_x}\right)^2 + \left(\frac{A_y}{s_y}\right)^2}$$
$$A = \mathcal{F}_R(\vec{m}_2 - \mathcal{F}_{\vec{T}}) \times \mathcal{F}_R(\vec{m}_1 - \mathcal{F}_{\vec{T}})$$

These fit errors can be combined into a single weighted combination, as in Equation 2.1. Such a combination yields

$$E_{fit}(\mathcal{F}^{-1}) = \alpha_{fit} E_{fit,o}(\mathcal{F}^{-1}) + (1 - \alpha_{fit}) E_{fit,r}(\mathcal{F}^{-1})$$

$$= \alpha_{fit} \sum_{i \in C_o} \sum_{j=1}^{2} \frac{\lambda_i}{M_i^2} \left(\mathcal{F}_R^{-1}(\frac{d_{ij,x}}{s_x}, \frac{d_{ij,y}}{s_y}, 1) \cdot (\vec{m}_{i2} - \mathcal{F}_{\vec{T}}^{-1}) \times (\vec{m}_{i1} - \mathcal{F}_{\vec{T}}^{-1}) \right)^2$$
$$+(1 - \alpha_{fit}) \sum_{i \in C_r} \sum_{j=1}^{2} (((\mathcal{F}^{-1} \vec{d}_{ij} - (\vec{\rho}_i)) \cdot (\hat{(n)}_i))^2$$

While this new form is similar to the previous form, there are a few key differences. First, the mapping ($\mathcal{F}^{-1}$) that is found in this formulation is the inverse of that found in the previous formulation. Second, the values A and M must be updated at each step of the quasi-Newton optimization. Rather than attempting to leave the F terms in A, which would increase the degree of nonlinearity, it is reasonable to use F^e, the estimate of F in the current iteration. This keeps with the spirit of expanding the linearization about the current estimate and calculating the direction of improvement.

While this new form is similar to the previous form and should prove simple to implement, due to time constraints, this final step has not been completed. Possible future work would include the same analysis of this form as was provided in Chapter 2 with respect to the original coregistration algorithm.

The reader should also be reminded of the motivation for this new form: by grouping features the size of the correspondence space is drastically decreased. For example, it is not uncommon to have an object with 100 pixels in image. Matching this 100 pixels with 100 pixels in the model would yield $100 \times 100 = 10,000$ pairs. By grouping pixels into linear segments can easily decrease the number of image features by an order of magnitude (i.e.., 10 image features) and so can using planar surfaces in the model. This now decreases the potential number of correspondences from 10,000 to $10 \times 10 = 100$.

4.2 Optimization and Kalman Filtering

The goal of Kalman Filtering, as with all optimization techniques, is to find the "optimal" set of parameters for a criterion function. When a process model for the system being modeled exists, then "optimal" means most likely. Part of the assumptions made by Kalman Filtering includes notions of what properties the noise within the measurements will have. While some efforts have been made to compute the noise properties of image features [42, 37], these are not proven. So, while Kalman Filtering literature will sometimes refer to the solution being "most likely," I will only refer to it as being "optimal."

In more specific terms, the goal in Kalman Filtering is to construct a recursive estimator of the current system state, s. The system we are considering evolves in a noisy, but linear fashion over time. Each time we receive a new measurement, we want to incorporate this knowledge into our estimate of the system state.

4.2.1 The General Kalman Filter

The system we are considering is a noisy linear process defined by a state s_t at time t[2]. The system evolves as

$$s_{t+1} = as_t + w_t \tag{4.1}$$

where w is a white noise function[3].

When we measure the system, we cannot directly measure s_t. Instead we can measure y_t, which is a linear function of s_t. We can state this transformation

$$y_t = cs_t + v_t \tag{4.2}$$

where v is a white noise function.

Assumptions we make at this point are that we know the variance of the noise variables, σ_w^2 and σ_v^2, and we know the dynamics of the system, a. We also assume that we know the parameters of the sensor[4], c.

As a reminder, our goal is to find the "optimal" recursive estimator of the state, s_t. We write this estimator as

$$\hat{s}_t = a_t\hat{s}_{t-1} + b_t y_t$$

In other words, we weight how much we think the previous estimate ($\hat{s}_{t-1}$) and the current measurement (y_t) reflect the current state (s_t) when we are computing our current state estimate ($\hat{s}_t$). We use the values a_t and b_t to control this trade-off between past experience and current observation. So these are the parameters we want to alter to find the "optimal" estimate of the current system state.

Well, we keep saying "optimal" (in quotes), because we could define any of a number of optimality criteria. In the case of Kalman Filters, though, we consider the optimal solution to minimize the expected squared error. This can be written

$$p_t = E[e_t^2]$$

where the error, e_t, is the difference between the estimate and the true value. We can write this

$$\begin{aligned} e_t &= \hat{s}_t - s_t \\ &= a_t\hat{s}_{t-1} + b_t y_t - s_t \end{aligned}$$

---

[2]Here we will talk about things as they normally are, in terms of fusing knowledge over *time*. However, in the pose determination methods we will discuss, time is replaced by *paired measurement*, so that we are incrementally fusing individual feature pairs.

[3]Whenever we say that a variable is a "white noise" function, we mean that the variable w has the qualities that $E[w_t] = 0$ and that $E[w_{t1}w_{t2}] = 0$ if $t1 \neq t2$ and $E[w_{t1}w_{t2}] = \sigma_w^2$ if $t1 = t2$.

[4]Here, the term "sensor" is introduced to stand for the linear mapping between the world state and the measurement. When Kalman Filtering is extended to the vector domain, this notion of a measurement being mapped by a linear transformation of the entire system state. In real systems, the sensor observing the world embody the linear transformation.

So, we can rewrite the object function as

$$p = E[(a_t \hat{s}_{t-1} + b_t y_t - s_t)^2]$$

We have now cast this as a standard unconstrained optimization problem. To find the optimum, we simply take the partial derivative of the objective function (p_t) with respect to the parameters we can alter (a_t and b_t).

$$\begin{aligned} \frac{\partial p_t}{\partial a_t} &= 2E[(a_t \hat{s}_{t-1} + b_t y_t - s_t)\hat{s}_{t-1}] \\ \frac{\partial p_t}{\partial b_t} &= 2E[(a_t \hat{s}_{t-1} + b_t y_t - s_t) y_t] \end{aligned}$$

Then we set these to 0 and solve for a_t and b_t.

We can now define the scalar Kalman Filtering Estimator as

$$\begin{aligned} \hat{s}_t &= a\hat{s}_{t-1} + b_t(y_t - ac\hat{y}_{t-1}) \\ b_t &= cq_t(c^2 q_t + \sigma_v^2)^{-1} \\ q_t &= a^2 p_{t-1} + \sigma_w^2 \\ p_t &= q_t - cb_t q_t \end{aligned}$$

These equations are, in order, the *recursive filter estimator*, the *filter gain*, the q, and the *mean square error*.

A system with a vector state, $\vec{s}_t$ can also be defined similar to the scalar system from Equation 4.1.

$$\vec{s}_{t+1} = matA\vec{s}_t + \vec{w}_t$$

This system is observed through a series of measurements

$$\vec{z}_t = \mathbf{H}\vec{s}_t + \vec{v}_t$$

A similar formulation can be followed to construct the vector Kalman Filter Equations.

$$\hat{s}_t = \hat{s}_{t-1} + K_k(\hat{z}_t - \mathbf{H}_t \hat{s}_{t-1}) \tag{4.3}$$

$$\mathbf{\Sigma}_t = \mathbf{\Sigma}_{t-1} - \mathbf{K}_t \mathbf{H}_t \mathbf{\Sigma}_{t-1} \tag{4.4}$$

$$\mathbf{K}_t = \Sigma_{t-1}\mathbf{H}_t^T(\mathbf{H}_t \Sigma_{t-1} \mathbf{H}_t^T + \Lambda_t)^{-1} \tag{4.5}$$

These equations are, in order, the *state estimate update*, the *state covariance update*, and the *Kalman gain matrix*. In this form, $\hat{s}_t$ is now a vector estimating the state $\vec{s}$ and $\mathbf{\Lambda_t}$ is the covariance matrix of $\vec{w}$.

This can be shown to subsume Gauss's standard least-squares estimation [39].

Beyond this, people also speak of Extended Kalman Filters. This notion is the intuitive cross between gradient decent in standard least squares optimization and Kalman Filtering. If one has a nonlinear model equation, the idea is to take the first order Taylor series expansion of the model equation expanded around the current state estimate. This linear approximation serves us now as a model as we formulate the standard Kalman Filter.

4.2.2 Application in Pose Determination

Kalman Filtering is no stranger to Computer Vision. Ayache [1] used Kalman Filtering in a 2D domain. The basic idea was to use the iterative nature of Kalman Filtering to incorporate the next, best model-data pair into the match, updating both the pose hypothesis and the covariance of the pose while performing matching.

Hel-Or [19] and Hel-Or and Werman [18, 20] use an Extended Kalman Filtering framework to deal with both 3D data (including perspective sensors such as standard optical sensors) and the integration of data from heterogeneous sources (i.e., multi-sensor fusion). In the spirit of Ayache, they also interleave the matching process with the pose+covariance update.

Remember from the last section that a Kalman Filter is based around five components: measurements, covariance/noise characterization, state, and a linear relationship between the measurements and the state. Hel-Or and Werman define each of these components in their model. In our treatment, we will use mostly their notation, with some adaptations to the symbols used in this thesis (where appropriate).

The model will be represented as a set, M, of n 3D points. We can write this formally as

$$M = \{\vec{u}_i | \vec{u}_i = (x_i, y_i, z_i)^T \forall i \in [1, n]\}$$

This is mapped into perfect measurements, $\vec{u}'_i$, by the transformation $\mathcal{F}$. $\mathcal{F}$ is an affine transformation and does not include any perspective transformations. As described in the previous section, true measurements are not available, so noisy measurements must be relied upon. The set of noisy measurements, M', is denoted

$$M' = \{(\hat{u}'_j, \Lambda_j) | \forall j \in [1, m]\}$$

where $\hat{u}'_j = \vec{u}'_j + n_j$, with n_j as a white noise variable. The white noise's covariance, Λ_j, varies based upon the measurement type and sensor characteristics. For all measurements, however, $\hat{u}'_j$ and Λ_j are considered to be 3D, even if the measurement is inherently 2D.

Given these measurements, M', and a model, M, as well as a correspondence set, $\{(i, j)\}$[5], the goal is to recover the model pose, $\mathcal{F}$. In Kalman Filtering terms, $\mathcal{F}$ becomes the state. Here, the state is static, even though the Kalman Filtering framework can accommodate dynamic state.

$\mathcal{F}$ is defined as $\{\vec{T}, \vec{s}\}$, where $\vec{T}$ is the translation portion of the mapping and $\vec{s}$ is the rotation. If we represent the rotation as the quaternion, $\tilde{q} = (q_0, \vec{q})$, then $\vec{s}$ can be written $\frac{\vec{q}}{q_0}$. Conversely, $\vec{s}$ can be converted to quaternion form by

$$\begin{aligned} q_0 &= \frac{1}{\sqrt{1 + \vec{s}^T \vec{s}}} \\ \vec{q} &= q_0 \vec{s} \end{aligned}$$

All that is left to cast this as a Kalman Filtering problem is to define the relationship between the measurements and the state. The measurement model is defined by the

[5] As Hel-Or and Werman did, the convention here will be to let $i = j$ implicitly, simplifying the formulation.

following nonlinear equation using quaternion notation.

$$\tilde{u}'_i = \tilde{q}\tilde{u}_i\tilde{q}^* + \tilde{t} \tag{4.6}$$

$\tilde{q}^*$ is the conjugate of $\tilde{q}$. After some manipulations, this yields the (non-linear) mapping, $\vec{h}$ from world state to (noise-free) measurement.

$$\vec{h}_i(\vec{u}_i, \vec{u}'_i, \mathcal{F}) \equiv < \vec{u}'_i + \vec{u}_i > \vec{s} + (\vec{u}'_i - \vec{u}_i) - (I_3 - < \vec{s} >)\vec{T} \;=\; \vec{0}$$

with the 3×3 identity matrix denoted I_3. The matrix form of the cross product is represented by the operator $< \cdot >$ (i.e., $\vec{u} \times \vec{v} =< \vec{u} > \vec{v}$).

$$< \vec{v} > \;=\; \begin{pmatrix} 0 & \vec{v}_z & -\vec{v}_y \\ -\vec{v}_z & 0 & \vec{v}_x \\ \vec{v}_y & -\vec{v}_x & 0 \end{pmatrix}$$

Kalman Filtering requires a linear measurement model, and the one above is definitely nonlinear. To solve this, an Extended Kalman Filter is used, so h_k is linearized around the current estimate of $(\hat{\mathcal{F}}^{k-1}, \hat{u}'_k)$ using a first order Taylor series expansion (where $\hat{\mathcal{F}}^{k-1}$ is just the estimate of the state, $\mathcal{F}$, after measurement $k-1$ has been fused).

$$\begin{aligned} h_k(\vec{u}_k, \vec{u}'_k, \mathcal{F}) &= 0 \\ &\approx h_k(\vec{u}_k, \hat{u}'_k, \hat{\mathcal{F}}^{k-1}) + \frac{\partial h_k}{\partial \mathcal{F}}(\mathcal{F} - \hat{\mathcal{F}}^{k-1}) \end{aligned}$$

After some manipulation, this can be rewritten in the standard Kalman Filtering notation as

$$\vec{z}_k \;=\; H_k\mathcal{F} + \nu_k$$

where

$$\vec{z}_k = < \hat{s}^{k-1} > \hat{T}^{k-1} + \vec{u}_k - \hat{u}'_k \tag{4.7}$$

$$H_k = (< \hat{u}'_k + \vec{u}_k - \hat{T}^{k-1} >, < \hat{s}^{k-1} > -I_3) \tag{4.8}$$

$$\nu_k = (I_3 - < \hat{s}^{k-1} >)(\vec{u}'_k - \hat{u}'_k) \tag{4.9}$$

With the Kalman Filtering form of pose for rigid 3D objects in place, the only remaining task is to determine the covariance matrices for the various sensor types. Each of these is fully derived in [19]. Here we will simply cover the two sensor types of interest to us: perspective optical and perspective range sensors.

The basic geometry used for perspective sensors is shown in Figure 4.4.

If we assume that the distance from the $x \times y$ plane to the $v \times w$ plane is 1, we can make the following statements.

$$\begin{aligned} \hat{\phi} &= \arctan\sqrt{\hat{v}^2 + \hat{w}^2} \\ \hat{\theta} &= \arccos\frac{\hat{v}}{\sqrt{\hat{v}^2 + \hat{w}^2}} \end{aligned}$$

$$\begin{aligned} \kappa &= \frac{1}{\sqrt{\hat{v}^2 + \hat{w}^2 + 1}} \\ \psi &= \frac{1}{\sqrt{\hat{v}^2 + \hat{w}^2}} \end{aligned}$$

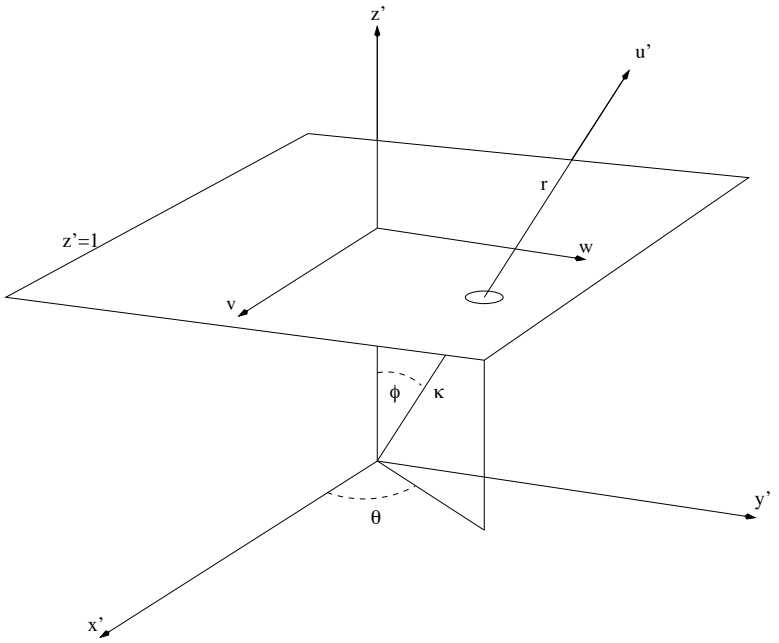


Figure 4.4: Perspective projections in Hel-Or's framework.

$\hat{r}$ is undefined in the 2D case. For a range sensor, $\hat{r} = |\hat{u}|$. Notice here that all measurements up to this point have been defined as estimates. As estimates, there is some uncertainty in the values. To define the covariance matrix characterizing the noisiness of these estimates, the uncertainty of $(\hat{v}, \hat{w})$ is cast in terms of (ϕ, θ).

$$\Lambda_{\phi\theta} = \left(\frac{\partial(\phi,\theta)}{\partial(v,w)}\right) \Sigma_{vw} \left(\frac{\partial(\phi,\theta)}{\partial(v,w)}\right)^T$$

$\frac{\partial(\phi,\theta)}{\partial(v,w)}$ is the Jacobian transform from (v, w) to (ϕ, θ).

Transforming the covariance matrix into spherical coordinates yields

$$\Lambda_{r\phi\theta} = \begin{pmatrix} \infty & 0 & 0 \\ 0 & & \\ 0 & & \Lambda_{\phi\theta} \end{pmatrix}$$

for the 2D perspective case. The ∞ reflects that we have no idea from this single measurement what the range is. In the case of perspective range sensing, we know this value, so the covariance matrix in spherical coordinates is

$$\Lambda_{r\phi\theta} = \begin{pmatrix} \sigma_r^2 & 0 & 0 \\ 0 & & \\ 0 & & \Lambda_{\phi\theta} \end{pmatrix}$$

Equations 4.3, 4.4, 4.5, 4.7, 4.8, and 4.9, in addition to the covariance matrices defined above, are enough to implement pose determination for a sensor suite where the sensors have coincident focal points. [6]

[6]Though not covered here, computational stability issues are addressed in [19].

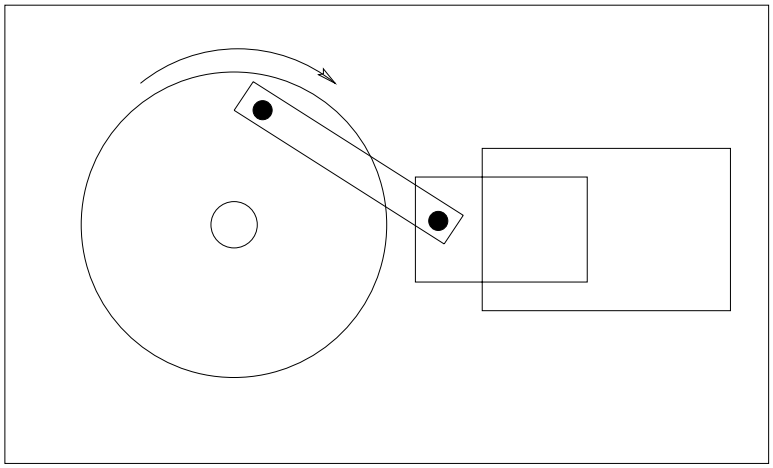


Figure 4.5: An example of an articulated model.

4.3 Kalman Filtering and Articulated Models

In [19, 20], articulated objects and constraints in general are added to this framework. While this is not directly the coregistration problem addressed throughout this thesis, its solution provides the basis for the Kalman Filtering coregistration method to be proposed in Section 4.3.1.

In Figure 4.5, we find an example of an articulated object. The joints on the bar coupling the piston and the wheel are revolute joints, allowing rotation, but keeping a fixed distance (0) between the two connected points. Using a parametric (implicit) form as we did with the sensor suite in earlier sections, we would have the total pose of the suite of model components, $\mathcal{F}_{suite}$, and the internal parameter, $\mathcal{F}_\theta$. In this case, we would be required to perform a fair amount of work to formulate a mathematical representation of the coupling of the wheel, bar, and piston.

Hel-Or and Werman formulate their method around the non-parametric (explicit) constraint model. In this paradigm, the model suite is composed of model components, each having its own pose $\mathcal{F}_i$. Constraints can then be placed on the transformations to embody the physical connections between the components. The general form of a constraint is

$$\psi_j(\mathcal{F}_l, \mathcal{F}_m) = 0 \tag{4.10}$$

In other words, there is some constraint function that, given two legal transformations, one for model component l and one for component m, is always 0.[7] For example, in the case of a revolute joint, the constraint would be

$$\begin{aligned} \psi_j(\mathcal{F}_l, \mathcal{F}_m) &= |\mathcal{F}_l(\vec{P}_l) - \mathcal{F}_m(\vec{P}_m)| \\ &= 0 \end{aligned} \tag{4.11}$$

where $\vec{P}_l$ and $\vec{P}_m$ is the point about which the joint rotates on component l and m.

[7] To deal with greater-than or less-than constraints, slack variables can be introduced, just as in linear programming.

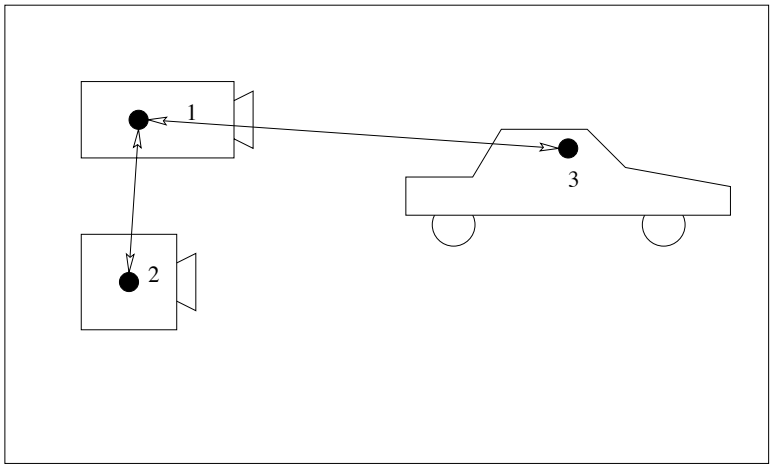


Figure 4.6: An example of an articulated sensor suite.

The constraints are used to introduce artificial measurements into the Kalman Filter system, based upon estimates derived from actual measurements. These constraint measurements are assumed to have **0** noise. Hel-Or **??** proposes three methods for determining the order in which the real and artificial measurements are fused: incremental, dynamic, and batch methods. These can result in different state estimates being recovered, a ordering effect inherent in Kalman Filtering.

4.3.1 Kalman Filtering and Coregistration

As a concrete but fanciful example, consider the articulated sensor suite found in Figure 4.6. Here, sensors *1* (a color camera) and *2* (a monochrome camera forming a stereo pair with *1*) are co-mounted on the metal bar above a street intersection, allowing them a view of traffic. The rough location of these is known, but is imperfect due to the wind shaking the bar. The distance between the two, however, can be known with relatively good accuracy. Driving on the street is an automobile with a camera looking out through the windshield. This camera uses a differential GPS (Global Positioning System), yielding high positional accuracy, but no orientational information. By fusing the information from these sensors, the traffic pattern can be ascertained, helping the automobile to suggest a low-traffic route home.[8]

What do we know about the constraints on this system? We know the distance between *1* and *2*, as well as a rough distance between *1* and *3*. These relationships are easily cast as constraints according to the form Hel-Or and Werman used.

$$\begin{aligned} \psi_{12}(\mathcal{F}_1, \mathcal{F}_2) &= |\mathcal{F}_1(\vec{0}) - \mathcal{F}_2(\vec{0})| - d_{12} && (4.12) \\ &= 0 \\ \psi_{13}(\mathcal{F}_1, \mathcal{F}_3) &= |\mathcal{F}_1(\vec{0}) - \mathcal{F}_3(\vec{0})| - d_{13} + s_{13} && (4.13) \\ &= 0 \\ & && (4.14) \end{aligned}$$

[8] I said this was a fanciful example.

where $\vec{0} = (0,0,0)$, d_{12} is the known distance between *1* and *2* and d_{13} is the rough distance between *1* and *3* with s_{13} taking up the slack in this measurement. A constraint can even be formed to anchor sensor *3* to the known GPS location with some slack.

$$\begin{aligned} \psi_{3g}(\mathcal{F}_3, \mathcal{F}_g) &= |\mathcal{F}_3(\vec{0}) - \mathcal{F}_g(\vec{0})| + s_{3g} \qquad (4.15) \\ &= 0 \end{aligned}$$

$$(4.16)$$

where s_{3g} captures the inaccuracies in GPS technology.

Once this system has been modeled, the system can be solved using any of the techniques used with articulated objects (i.e., incremental, dynamic, or batch fusion methods). In fact, the coregistration constraints and the articulated object constraints could also be combined, yielding a coregistration system capable of determining an objects internal parameters from unregistered multi-sensor data.

4.4 Tradeoffs

While an arbitrarily complex system could be encoded in the fashion proposed by Hel-Or and expanded upon in the previous section, the more constraints are introduced into a system, the more difficult it is to solve and the less likely a stable solution can be found. Part of this is an artifact of the explicit constraints used in Hel-Or's formulation. In the system for the two sensor suite used in Chapters 2 and 3, a transformation is necessary for each sensor, yielding 12 degrees of freedom. This is a considerably more complex system to solve than the 8 degree of freedom formulation from Chapter 2 using implicit constraints.

However, the formulation of the coregistration problem using implicit constraints was painstaking. Given the specific sensor suite, the parameter reduction required considerable thought. Using explicit constraints, this would have been straight forward and nearly algorithmic.

This tradeoff between simplicity of formulating a new problem and simplicity of solving the formulation is the main reason for presenting both methods. In situations where sensor configurations contain complex constraints which cannot be easily cast in an implicit framework, the use of explicit constraints is obvious. However, the potential for a more stable solution can make the price of finding an implicit formulation worth while.

Chapter 5

Conclusions

5.1 Recap

I assumed that incorporating the sensor registration process into the pose determination task is necessary to fully exploit the information inherent in heterogeneous sensor suites. To test the feasibility of such a *coregistration* system, an error function based on both Kumar's and Horn's measures was introduced. This error implicitly used coplanarity and co-orientational constraints to protect the linearity of the system. Testing the least-squares coregistration algorithm on a suite of perfect synthetic data, it was found that the correct pose+registration parameters were found at least 95% of the time when the initial pose+registration estimate was off by no more than 25° and 500 meters. As with all Least-Mean Squares methods, this coregistration formulation is very susceptible to outliers. In both tests on synthetic data and on real data with hand-picked feature correspondences, we found that outliers created a number of local optima around the true solution, yielding poor convergence.

To deal with the problem of outliers, two techniques were investigated: a statistically robust method (median filtering) and a traditional search method (local search). While both of these methods could be used to refine the match, given an accurate initial pose+registration hypothesis, reasonably sized errors introduced more than 50% outliers (the maximum amount that median filtering can cope with) and intraversable search spaces (in the case of local search). The problem with median filtering cannot be overcome, since the majority of an image will usually be non-target, pointing towards large numbers of outliers.

When using local search, the problem is with both the large number of range features (both model and data) and with the definition of the search neighborhood. Two possible approaches present themselves: alter the topology of the search space or increase the sophistication of the data/model features. I chose the latter of these. By using a modification of range image segmentation, I produced 3D linear surface segments and matched these to planes representing the model faces. While this was motivated and a criteria function was developed, demonstration of a working implementation was not performed.

Another approach to coregistration is to move a level lower than Least-Mean Squares and formulate the problem within the Kalman Filtering framework. A more general method allowing for more sophisticated explicit constraints is sketched, based upon Hel-Or and Werman's model for articulated objects. Again, while I proposed a method based on these ideas, an implementation has yet to be build.

5.2 Future Work

While a method for performing coregistration has been proposed, the system demonstrated is only a simple proof of concept. The constraints are implicitly, rather than explicitly stated, making it unsuitable for generalization. It does serve as both a call to arms and a demonstration of the philosophy I am proposing.

The current system implementation described in Chapters 2 and 3 is limited due to the large search spaces and awkward search neighborhoods. Also, the coregistration solution space was unstable due to the large number of features in the range component. To solve these problems, the next step would be to implement the line-to-plane range measure formulated in Section 4.1.2. Doing so would yield a usable multisensor matching system.

Even this implementation would contain an implicit constraint. Explicitly casting the intersensor constraints appears straightforward in the Kalman Filtering framework. Based on Hel-Or and Werman's method for articulated objects and the analogy to sensor-to-sensor constraints, a general coregistration+matching system could be formulated and constructed.

Another direction which has not been touched on in this thesis is the *necessity* of coregistration. So far, I have assumed that coregistration will yield better results than solving the pose of each decoupled sensor, individually. When a suitably robust coregistration system has been constructed (either the line-to-plane (range) system or the Kalman Filtering system), a comparison against pose based matching for individual sensors will be necessary. Conducted on synthetic data, the recovered pose's distance from ground truth could be computed, revealing the amount of improvement or degradation caused by coupling the sensors.

Also, remember that the construction of coregistration systems was based on the assumption that the accuracy derived from model-driven registration was necessary. To demonstrate this and to provide a useful tool, our system should be used as a front-end to a higher-level sensor fusion system. If the higher-level sensor fusion does well assuming that the images are registered, and does not when images are left with default registration, then coregistration can be said to be necessary. For example, an occlusion detection sensor fusion module could be used as the higher-level system.

This level of testing will soon be possible. The work on hypothesis driven matching by Beveridge, et. al. [3, 4, 6] lays out a framework for driving a multisensor matching system, including a coregistration phase. The work by Stevens and Beveridge [40] demonstrates coregistration through pose+registration search, rather than correspondence search. Combining these with the coregistration algorithms presented in this thesis would provide a complete system. This integration is already in the works.

5.3 Final Remarks

In closing, the utilization of data from multiple modalities and multiple discrete sensors will become more important in the future. The amount of information provided by any one sensor does not offer sufficient disambiguation in many challenging domains, such as RSTA or navigation in a cluttered environment. The use of discrete sensors allows both more flexibility in selection of modality and the opportunity to utilize low cost, off the shelf parts.

When dealing with information from multiple sensors, information is lost if the sensors are treated in isolation: some connection must be drawn between the images in a suite, requiring some form of registration. This requires the sensors to be registered. While the intersensor parameters can be calibrated, these parameters will drift as the operating environment affects the sensors. These changes can translate to a several pixel registration error, which can ruin the interpretation of the image suite.

Ultimately, self-calibrating algorithms are necessary in this domain. While the method implemented for this thesis has serious limitations, extensions such as the one proposed in Chapter 4 can answer these problems and provide the stability necessary for tomorrow's multi-sensor vision systems.

Bibliography

[1] Nicholas Ayache and Olivier D. Faugeras. HYPER: A New Approach for the Recognition and Positioning of Two-Dimensional Objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(1):44–54, January 1996.

[2] J. Ross Beveridge. *Local Search Algorithms for Geometric Object Recognition: Optimal Correspondence and Pose*. PhD thesis, University of Massachusetts, 1993.

[3] J. Ross Beveridge, Allen Hanson, and Durga Panda. Integrated color ccd, flir & ladar based object modeling and recognition. Technical report, Colorado State University and Alliant Techsystems and University of Massachusetts, April 1994.

[4] J. Ross Beveridge, Allen Hanson, and Durga Panda. Model-based Fusion of FLIR, Color and LADAR. In Paul S. Schenker and Gerard T. McKee, editors, *Proceedings: Sensor Fusion and Networked Robotics VIII*, Proc. SPIE 2589, pages 2 – 11, October 1995.

[5] J. Ross Beveridge, Durga P. Panda, and Theodore Yachik. November 1993 Fort Carson RSTA Data Collection Final Report. Technical Report CSS-94-118, Colorado State University, Fort Collins, CO, January 1994.

[6] J. Ross Beveridge, Mark R. Stevens, and Anthony N. A. Schwickerath. Toward target verification through 3-d model-based sensor fusion. *IEEE Transactions on Image Processing*, page (Submitted), 1996.

[7] J. Ross Beveridge, Mark R. Stevens, Zhongfei Zhang, and Mike Goss. Approximate Image Mappings Between Nearly Boresight Aligned Optical and Range Sensors. Technical Report CS-96-112, Computer Science, Colorado State University, Fort Collins, CO, April 1996.

[8] James Bevington, Randy Johnston, and Joel Lee. A Modular Target Recognition Algorithm for LADAR. In *2nd Automatic Target Recognizer Systems and Technology Conference*, pages 91–104, March 1992.

[9] James E. Bevington. Laser Radar ATR Algorithms: Phase III Final Report. Technical report, Alliant Techsystems, Inc., May 1992.

[10] S. M. Bozic. *Digital and Kalman Filtering*. Halsted Press, 2nd edition, 1994.

[11] J. B. Burns, A. R. Hanson, and E. M. Riseman. Extracting Straight Lines. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(4):425–456, July 1986.

[12] Bruce A. Draper, Carla E. Brodley, and Paul E. Utgoff. Goal-Directed Classification using Linear Machine Decision Trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(9):888–893, September 1994.

[13] R. O. Eason and R. C. Gonzalez. Least-Squares Fusion of Multisensory Data. In Mongi A. Abidi and Rafael C. Gonzalez, editors, *Data Fusion in Robotics and Machine Intelligence*, chapter 9. Academic Press, 1992.

[14] Michael E. Goss, J. Ross Beveridge, Mark Stevens, and Aaron Fuegi. Visualization and Verification of Automatic Target Recognition Results Using Combined Range and Optical Imager. In *1994 ARPA Image Understanding Workshop Proceedings*, 1994.

[15] L. Gottesfeld-Brown. A survey of image registration techniques. *ACM Computing Surveys*, 24(4):325, December 1992.

[16] W. Eric L. Grimson and Tomás Lozano-Pérez. Localizing overlapping parts by searching the interpretation tree. *Transactions on pattern Analysis and Machine Intelligence*, PAMI-9(4):469–482, July 1987.

[17] Martial Hebert. Presentation of the Mobility Group, UGV Demo II, Killeen, Texas. Future Recommendations of the Stereo Group, June 1996.

[18] Y. Hel-Or and M. Werman. Absolute Orientation from Uncertain Data: A Unified Approach. In *Proceedings: Computer Vision and Pattern Recognition*, pages 77 – 82. IEEE Computer Society Press, June 1993.

[19] Yacov Hel-Or. *Model Based Pose Estimation from Uncertain Data.* PhD thesis, Hebrew University in Jerusalem, 1993.

[20] Yacov Hel-Or and Michael Werman. Constraint-fusion for interpretation of articulated objects. In *CVPR94*, pages 39–45, 1994.

[21] Richard Hoffman and Anil K. Jain. Segmentation and Classification of Range Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-9(5):608–620, September 1987.

[22] B. K. P. Horn. *Robot Vision.* The MIT Press, 1986.

[23] Berthold K. P. Horn. Closed-form Solution of Absolute Orientation Using Unit Quaternions. *J. Opt. Soc. Am. A*, 4(4):629–642, April 1987.

[24] Daniel P. Huttenlocher, Gregory A. Klanderman, and William J. Rucklidge. Comparing Images Using the Hausdorff Distance. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(9):850–863, September 1993.

[25] X. Y. Jiang and H. Bunke. Fast Segmentation of Range Images into Planar Regions by Scan Line Grouping. Technical Report IAM-92-006, University of Berne, 1992.

[26] Rakesh Kumar. Determination of Camera Location and Orientation. In *Proceedings: Image Understanding Workshop*, pages 870 – 881, Los Altos, CA, June 1989. DARPA, Morgan Kaufmann Publishers, Inc.

[27] Rakesh Kumar. *Model Dependent Inference of 3D Information From a Sequence of 2D Images.* PhD thesis, University of Massechusetts, 1992.

[28] Rakesh Kumar and Allen R. Hanson. Robust methods for estimating pose and a sensitivity analysis. *CVGIP:Image Understanding*, 11, 1994.

[29] David G. Lowe. Fitting Parameterized Three-Dimensional Models to Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(5):441–450, May 1991.

[30] David G. Lowe. Robust Model-based Motion Tracking Through the Integration of Search and Estimation. *International Journal of Computer Vision*, August 1992.

[31] Takeshi Masuda and Naokazu Yokoya. A Robust Method for Registration and Segmentation of Multiple Range Images. *Computer Vision and Image Understanding*, 61(3):295–307, May 1995.

[32] Sateesha G. Nadabar and Anil K. Jain. Fusion of Range and Intensity Images on a Connection Machine (CM-2). *Pattern Recognition*, 28(1):11–26, 1995.

[33] B. Parvin and G. Medioni. Segmentation of Range Images into Planar Surfaces by Split and Merge. pages 415–417, 1986.

[34] Arthur R. Pope. Model-based object recognition: A survey of recent research. Technical Report 94-04, University of British Columbia, January 1994.

[35] Arthur R. Pope and David G. Lowe. Modeling positional uncertainty in object recognition. Technical Report 94-32, University of British Columbia, November 1994.

[36] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C: The Art of Scientific Computing.* Cambridge University Press, 2nd edition, 1992.

[37] Visvanathan Ramesh. *Performance Characterization of Image Understanding Algorithms.* PhD thesis, University of Washington, 1995.

[38] Anthony N. A. Schwickerath and J. Ross Beveridge. Coregistration of Range and Optical Images Using Coplanarity and Orientation Constraints. *CVPR 96*, pages 899 – 906, 1996.

[39] H. W. Sorenson. Least-squares Estimation from Gauss to Kalman. *IEEE Spectrum*, pages 63–68, July 1970.

[40] Mark R. Stevens and J. Ross Beveridge. Interleaving 3D Model Feature Prediction and Matching to Support Multi-Sensor Object Recognition. In *Proceedings: Image Understanding Workshop*, pages 699–706, Los Altos, CA, February 1996. ARPA, Morgan Kaufman.

[41] Mark R. Stevens and J. Ross Beveridge. Optical Linear Feature Detection Based on Model Pose. In *Proceedings: Image Understanding Workshop*, pages 695–697, Los Altos, CA, February 1996. ARPA, Morgan Kaufman.

[42] William M. Wells III. *Statistical Object Recognition.* PhD thesis, Massachusetts Institute of Technology, 1993.

[43] Wayne L. Winston. *Introduction to Mathematical Programming: Applications and Algorithms.* Duxbury Press, 2nd edition, 1995.