Computer Science
Technical Report

Colorado State University

Precise Matching of 3-D Target Models to Multisensor Data [*] [†] [‡]

Mark R. Stevens and J. Ross Beveridge

January 20, 1997

Technical Report CS-96-122

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

[*] This work was sponsored by the Defense Advanced Research Projects Agency (DARPA) Image Understanding Program under grants DAAH04-93-G-422 and DAAH04-95-1-0447, monitored by the U. S. Army Research Office, and the National Science Foundation under grants CDA-9422007 and IRI-9503366

Precise Matching of 3-D Target Models to Multisensor Data *†‡

Mark R. Stevens and J. Ross Beveridge

January 20, 1997

Abstract

This paper presents a 3-D model-based ATR algorithm which operates simultaneously on imagery from three heterogeneous, approximately boresight aligned, sensors. An iterative search matches models to range and optical imagery by repeatedly predicting detectable features, measuring support for these features in the imagery, and adjusting the transformations relating the target to the sensors in order to improve the match. The result is a locally optimal and globally consistent set of 3-D transformations which precisely relate the best matching target features to combined range, IR and color images. Results show the multisensor algorithm recovers 3-D target pose more accurately than does a traditional single-sensor algorithm. Errors in registration between images are also corrected during matching.

1 Introduction

We are developing a new family of Automatic Target Recognition (ATR) algorithms for use with heterogeneous, ground-looking [1] sensors. The intended application domain is Reconnaissance, Surveillance and Target Acquisition (RSTA) from semi-autonomous military scout vehicles. For these RSTA tasks, it is assumed that separate IR, color and range (LADAR) sensors are co-mounted on a single pan-tilt platform.

For the RSTA application, we have developed a multisensor matching capability which registers 3-D target models to imagery from all three types of sensors. To create this new multisensor matching capability, we have combined advances in several key areas:

Multisensor target pose determination and cross-sensor registration. The ATR algorithm operates in a 3-D scene coordinate system within which it manages and adjusts 3-D relationships between sensors and the target. Consequently, matching is able to adjust and refine both the image registration mapping between sensors as well as the 3-D pose, position and orientation, of the target relative to the sensor suite.

On-line target feature prediction. Rather than employ precomputed image-based templates for different target signatures, an algorithm running on graphics accelerated hardware predicts what 3-D features of a given target model should be observable. The prediction algorithm uses the current target pose estimate and time-of-day lighting calculations.

Multisensor match evaluation. A single match error measures the overall quality of a hypothesized geometric relationship between the sensors and the target based upon the features predicted to be detectable. This error function is modular and easily modifiable to exploit new constraints.

*This work was sponsored by the Defense Advanced Research Projects Agency (DARPA) Image Understanding Program under grants DAAH04-93-G-422 and DAAH04-95-1-0447, monitored by the U. S. Army Research Office, and the National Science Foundation under grants CDA-9422007 and IRI-9503366

[1] Ground-looking indicates sensors on the ground plane looking out across terrain.

Robust optimization to find locally optimal multisensor matches. The match evaluation function is non-differentiable and, because new target features are predicted during search, non-stationary [2]. Traditional gradient descent methods are not applicable for minimizing such a function. A new form of local search [25, 27, 36, 2], more specifically a variant on Tabu Search [15], has been developed for this task.

While our work focuses specifically on the RSTA multisensor ATR problem, many of the advances described in this paper can be adapted and applied to other model-based object recognition problems.

The multisensor matching algorithm is embedded within a complete ATR system described briefly in Section 3. Results for the complete system, emphasizing the performance of the multisensor matching component, are presented in Section 7. In short, multisensor matching significantly improves the estimated 3-D target pose relative to estimates generated by a more traditional single-sensor ATR algorithm. This more traditional algorithm is a range boundary probing technique utilized by us both as a benchmark and to perform target type and pose hypothesis generation.

1.1 Nearly Boresight Aligned Sensors

The results presented in Section 7 also demonstrate that modest errors in initial image registration between the sensors can be corrected during matching. Performing sensor registration refinement as part of the ATR process is a novel aspect of our work and it solves what we think is a major problem associated with 3-D model-based recognition from multiple, separate sensors.

Typically, separate co-located sensors will produce imagery with different fields of view and different pixel resolutions. Consequently, the mapping between a pixel in one image to the corresponding pixel or pixels in another can be somewhat involved. In the ideal case of perfect boresight alignment, the mapping may be expressed as a 2D affine transformation between image coordinate systems. For nearly boresight aligned sensors viewing distant objects, the 2D affine mapping is still a good approximation [23].

Presuming that sensors are firmly affixed to a single solid platform, a calibration step can recover the affine mapping between image coordinate systems for different sensors. It might be assumed that once calibrated, the problem of image registration between sensors is solved for all time. However, this a risky and limiting assumption. The stereo group within the Unmanned Ground Vehicle Program has considerable experience with sensors operating on mobile platforms. They have reported that minor day-to-day alignment variations arise due to slight shifts in relative sensor pointing angles [19]. Presumably this is because bouncing around on rough terrain shifts slightly the geometry of the sensor platform. Similar misalignment problems can be expected with other types of co-located sensors.

Such very small shifts in pointing angle introduce what are essentially planar translations between pixels in one image relative to another. Therefore, rather than presume perfect image registration prior to initiating ATR, a more robust approach would utilize the target and sensor geometry together to correct for several pixel translations between sensors as part of the target recognition process. The specific geometric constraints which we use to accomplish this are presented in Section 6.1. A detailed explanation of why sensor translation may be used to compensate for small changes in pointing angles appears in [23].

1.2 Related Work

Model-based object recognition work has long emphasized the importance of aligning 3D object models to features extracted from sensed imagery [7, 30, 18, 22, 2]. While model-based approaches to Automatic Target Recognition have become much more common [14], direct incorporation of alignment into the recognition process is rare [6]. The work reported here is the first such attempt of which we are aware for the case of multiple, heterogeneous, ground-looking sensors.

Many researchers have contributed to the following areas that our work addresses: pose-determination, feature prediction, match evaluation and optimization. Related work for each topic is cited within the sections describing our contributions. On the general topic of sensor fusion, Aggarwal [1] nicely summarizes past work and notes that typically sensor fusion has emphasized single modality sensors, with comparatively little work on different

[2] Non-stationary here indicates that the quality of a state in the search space may depend upon the path taken to arrive at that state. This dependency does not dramatically alter the quality of a state, but it does introduce an added complication for traditional search techniques.

sensor modalities. He goes on to state that relating data from different modalities is more difficult, in part because of issues of sensor alignment and registration. While Aggarwal [32] and others [39] have examples of successful mixed-modality fusion, this is still a young research area.

2 The Fort Carson Range, IR and Color Dataset.

Our algorithms are tested on multisensor images collected at Fort Carson [4] in 1993. The entire collection contains over 30 range, IR and color image triples which are publicly available through our web site [3]. The range data was acquired using a LADAR built by Rathyeon and owned by Alliant TechSystems in Minnesota. The color imagery was collected using a standard 35mm camera and Kodak Ektachrome Elite slide film. The film was subsequently transferred to the Kodak Photo-CD digital format. The IR imagery was acquired using a 3 to 5 micron Amber FLIR. The three sensors were co-located to simulate three nearly boresight aligned sensors operating from a single pan-tilt platform. Additional information about sensor calibration and issues pertaining to alignment may be found in [23].

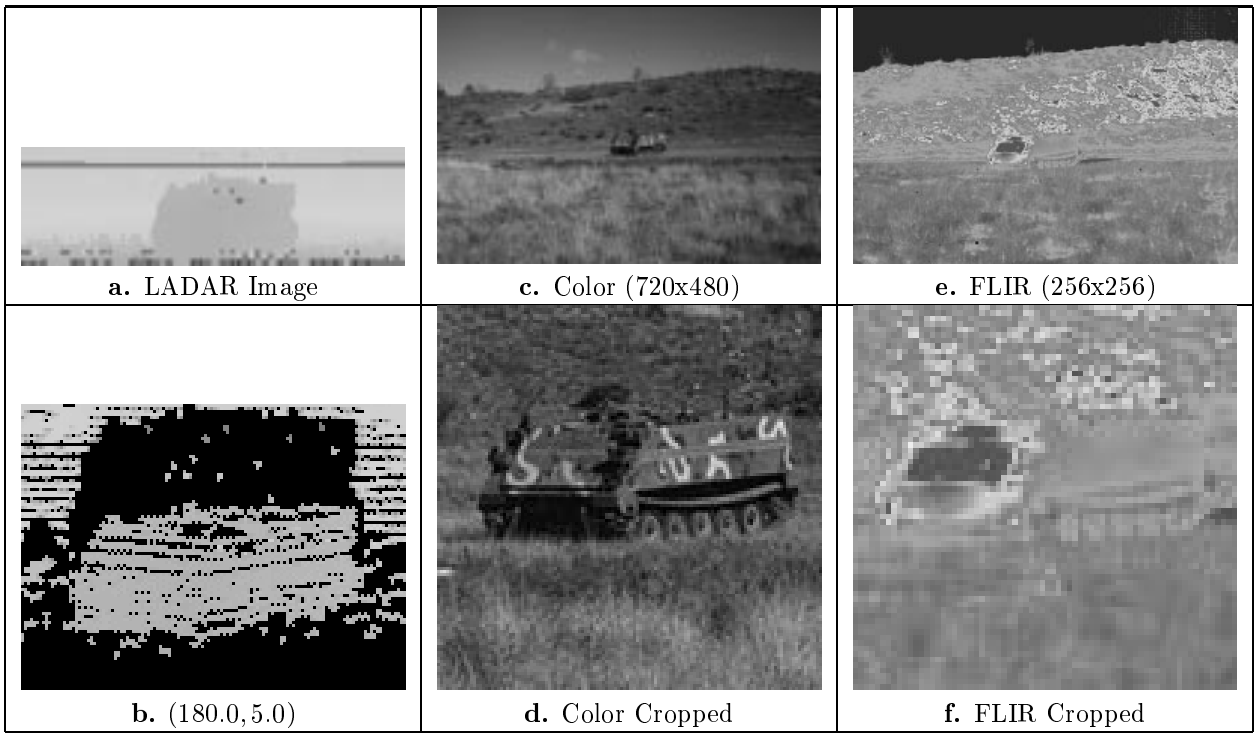
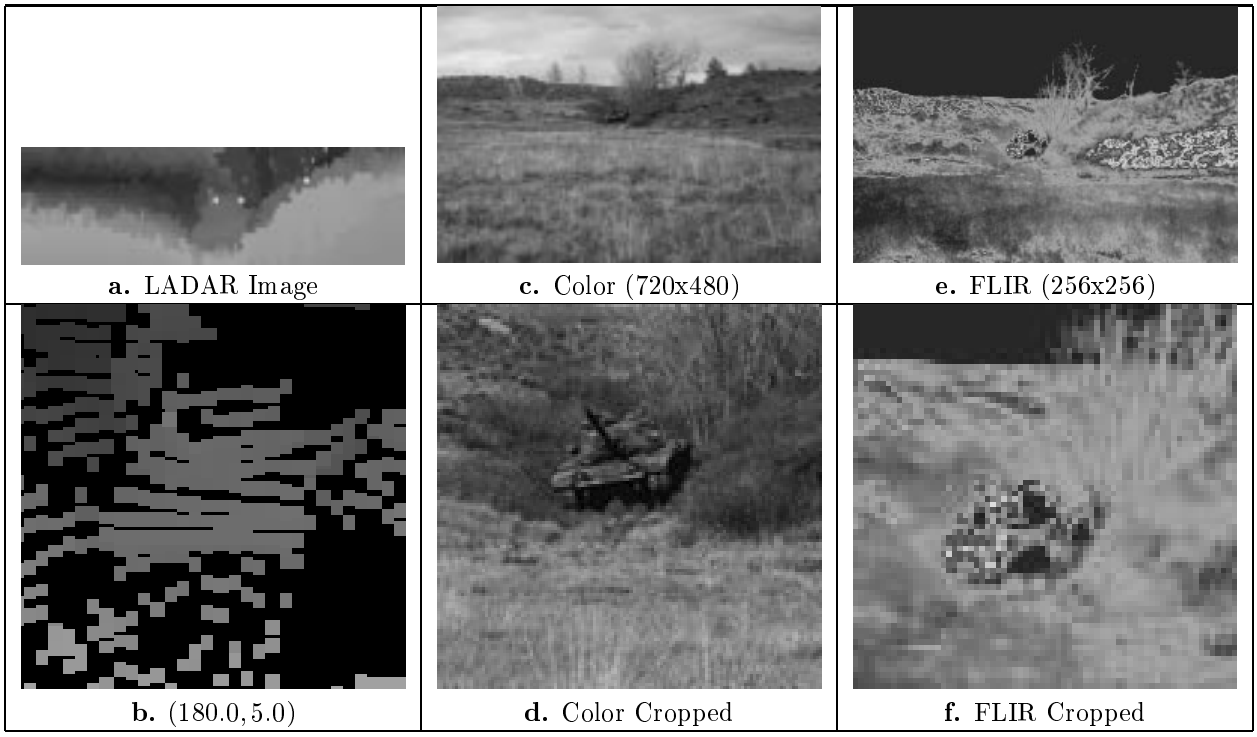


Figure 1: Shot20 Imagery from the Fort Carson Data Set

Example triples of range, IR and color imagery are presented in Figures 1, 2 and 3. Figures 1a, 2a and 3a show the range data as a grey-scale image: brighter values are closer. Figures 1b, 2b and 3b show the range data drawn as 3-D rectangular polygons at the depth of the corresponding range measurement. The remaining parts of Figures 1, 2 and 3 show the color and IR imagery: both full images and zoomed to better show the target.

The range visualization technique used in Figures 1b, 2b and 3b requires some explanation. These images were generated for a viewer looking at the 3D polygons from nearly straight on (viewing azimuth angle of 180 degrees) and an elevation of 5 degrees. From this elevation, the viewer looks slightly down upon the range points and can thus begin to discern some of the 3-D structure in the data. This 3-D range rendering is produced by our interactive 3-D visualization system [17, 16, 42]. While modestly useful for still images, the induced 3-D effect becomes more dramatic as the viewpoint changes in the interactive visualization environment.

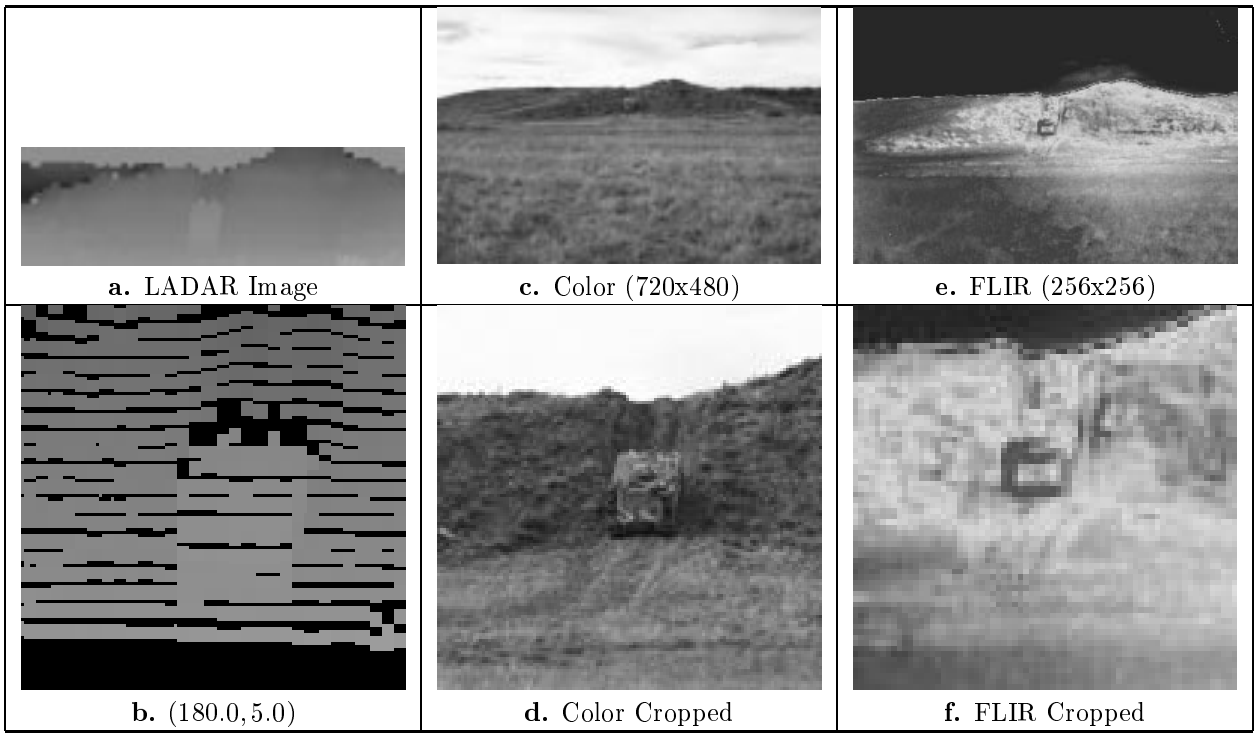
[3] `http://www.cs.colostate.edu/~vision`



a. LADAR Image | **c.** Color (720x480) | **e.** FLIR (256x256)

b. (180.0, 5.0) | **d.** Color Cropped | **f.** FLIR Cropped

Figure 2: Shot26 Imagery from the Fort Carson Data Set



a. LADAR Image | **c.** Color (720x480) | **e.** FLIR (256x256)

b. (180.0, 5.0) | **d.** Color Cropped | **f.** FLIR Cropped

Figure 3: Shot31 Imagery from the Fort Carson Data Set

From the standpoint of ATR, the Fort Carson imagery ranges from relatively simple to difficult. Factors making portions of the dataset challenging include reduced resolution (pixels on target), highly variable IR signatures, terrain obscuration, and finally vehicles at unusual viewing angles. Since our approach exploits color as well as IR and range, it is worth mentioning that the color imagery is highly textured and that vehicle camouflage and terrain are similarly colored.

Shot 20, Figure 1, presents a relatively simple test case, with an M113 viewed out in the open at relatively close range. [4] The IR signature is very weak in this example and IR recognition alone might be problematic. Shot 26, Figure 2, contains an M60 in a gully at 160 meters. Shot 31, Figure 3, shows an M113 coming straight down a steep hill at 135 meters. These latter two examples are more challenging due to the smaller number of pixels on target, the angle at which the vehicle is viewed, and the surrounding terrain.

3 The Complete ATR System

Our multisensor matching algorithm requires queuing in order to function. This queuing is provided by two upstream processes: 1) target detection and 2) target type and pose hypothesis generation. Detection uses color imagery to predict targets based upon the color characteristics of the camouflaged vehicles [8]. The detection information is then passed to a target type and pose hypothesis phase which generates a list of possible target types and orientations. This hypothesis generation algorithm uses boundary template matching in the range imagery [6]. Finally, for each hypothesized target, multisensor matching uses an iterative improvement optimization scheme to develop a best match between target features and the multisensor imagery.

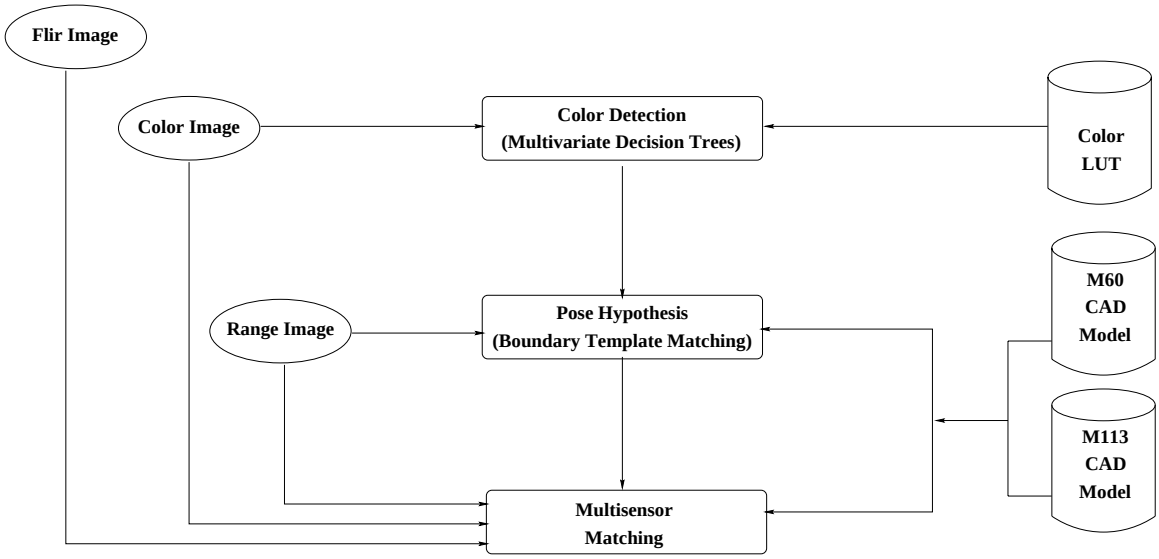


Figure 4: Overview of the ATR Algorithm

Figure 4 provides an overview of our complete ATR system, showing the flow of control through the three phases of the ATR process. Our focus here is on the third multisensor matching phase, and from the standpoint of this paper there is nothing uniquely special about the other algorithms chosen to perform queuing. That said, in order to better understand the functioning of the complete system, all three components are briefly summarized.

3.1 Target Detection

The detection algorithm was developed at the University of Massachusetts [8]. Using training imagery, it learns to discriminate between color values produced by camouflaged vehicles and values produced by background terrain.

[4] Actual Range is 50 meters. However range-to-targets are unusually short to accommodate the short operating range of the older LADAR. Wide angle lenses were used on IR and color to simulate all three sensors working with a better LADAR at 1 to 2 kilometers from the targets.

As long as the training imagery covers different times of day, lighting conditions and target types, the algorithm is able to generalize over these cases. The result of this training is a color lookup table (LUT) indicating, for each possible RGB color pixel value, whether it is more likely to be produced by a target or by the background.

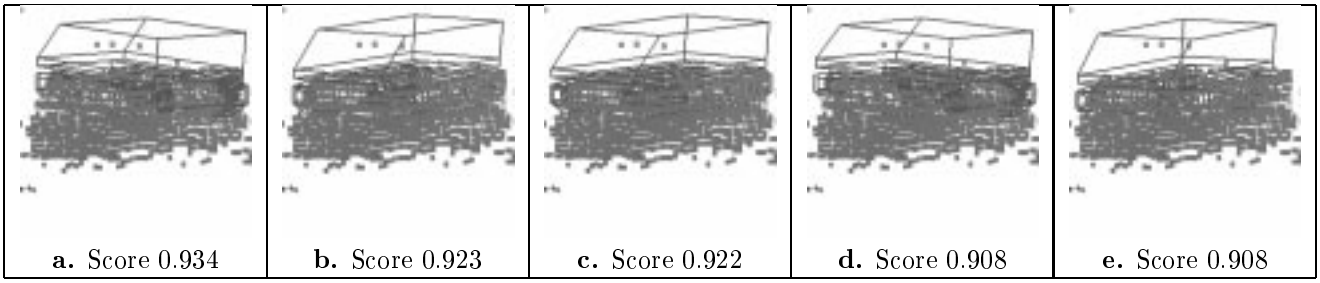
a. Targets **b.** Likelihoods

Figure 5: Result of Detection Algorithm on Shot 20 Color Image

The system then performs real-time color lookup on all the pixels in the image and classifies them as either target or background. A region of interest (ROI) extraction process sums responses over fixed sized windows in the image: one ROI for each local maximum in this summed response image which is over a minimum threshold. As the ROI is being generated, the likelihood for each pixel being a target is retained. Figure 5a shows the ROI of interest provided by the detection phase and Figure 5b shows the associated likelihood map.

3.2 Target Type and Pose Hypothesis Generation

Once the ROIs are determined, they are fed into the template matching algorithm which compares stored templates of the different CAD models against the range data [6]. The center of the ROI is converted to a range pixel, and the templates are then applied about that point. A score measuring the percentage of probes matched is used to rank each template according to how well it fits the data. The top five templates provide the pose and vehicle type estimates for the multisensor-matching phase. Figure 6b shows the top five pose estimates and the range data for the ROI shown in Figure 5. Here, higher scores are better.



a. Score 0.934 **b.** Score 0.923 **c.** Score 0.922 **d.** Score 0.908 **e.** Score 0.908

Figure 6: Result of Pose Hypothesis on Shot 20 Range Image

3.3 Multisensor 3-D Target Matching

After the hypotheses have been generated, the information is passed to the final multisensor matching phase. This phase looks at each pose hypothesis and locally refines the model pose for each of the three sensors simultaneously. As the pose is refined, the pixel-to-pixel alignment between sensors is also corrected. Simultaneous refinement of pose and image registration is referred to as **coregistration**.

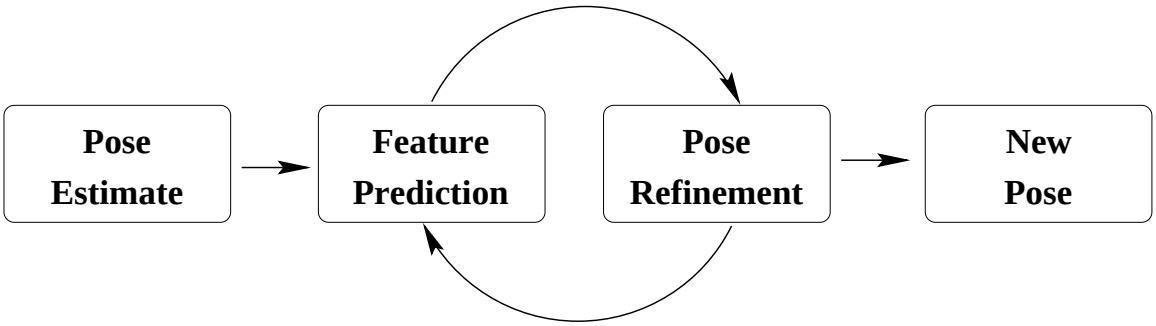


Figure 7: Coregistration Refinement

To optimize, at least locally, the target model to sensor coregistration, the search process executes an iterative generate-and-test loop illustrated in Figure 7. In this loop, the current coregistration estimate is used to predict a set of model features likely to indicate the presence of the target. Then a match metric, a match error in this case, is used to measure how much evidence for these predicted features can be found in the imagery. In our case, this match error is a complex, non-differentiable function. It takes into account whether or not evidence of the vehicle is found in each sensor as well as the strength of that evidence. This match error is computed for a series of possible moves, or states, generated by perturbing the current coregistration estimate. If a better estimate is found it replaces the current estimate. Every time a better estimate is found, the entire process repeats. Search only terminates when no further local improvement is possible.

The following three sections present the key aspects of the multisensor matching algorithm. The prediction of features is discussed in Section 4. The Error term used to guide the search is presented in Section 5. The strategy for generating the search neighborhood and the search strategy itself is discussed in Section 6.

4 Predicting Target Model Features

Prediction selects which model features should be used for matching to the data. Here, 3-D line segments are used for the optical imagery, and visible sampled surfaces for the range imagery. For optical imagery, selection takes into account the physical visibility and expected lighting. Silhouette features are used since they are relatively likely to stand out against the background. However, early work with our color data showed that using only silhouettes leads to ambiguity in the matching. Therefore, features representing internal detail as a function of lighting angle are also utilized.

Highly detailed models of the vehicles in our Fort Carson dataset exist in the CAD model format known as BRL/CAD [43]. Algorithms to reduce the model complexity to a level more closely related to the sensor granularity have already been developed [41, 40]. From these simpler models, features to be used in the matching process are obtained. Currently, we have models for the M113 APC, the M60 tank and a pickup truck. The model shape and contours are loosely based on those developed by Verly [44], who has analyzed the model shape in relation to LADAR data.

4.1 Predicting 3-D Line Segments Which Induce Observable Edges

The silhouette of an object is a valuable recognition cue when dealing with two-dimensional optical imagery [33, 26]. Many systems have been developed to recognize 3-D objects based on their projected 2D silhouettes [45, 28, 46]. More rare are works using 3-D edges directly [12], and then the goal is usually to link 2D image features to 3-D model features. Our method approaches the problem from the other direction: our goal is to work backward from a 2D edge produced in real-time using rendering hardware to predict the original 3-D feature inducing that edge.

Using internal detail, as well as silhouette information, has proven essential. Generally, what is desired is a mechanism for predicting what features are most likely to be measurable. Hoogs has noted that many factors can enter into such predictions, including geometric, temporal, functional, and radiometric factors [21]. Our feature prediction utilizes simple radiometric and temporal context information in order to predict the internal structure

likely to be visible in the optical imagery. Like others [11, 10], we have found these additional features to greatly aid the matching process.

4.1.1 Silhouette Lines

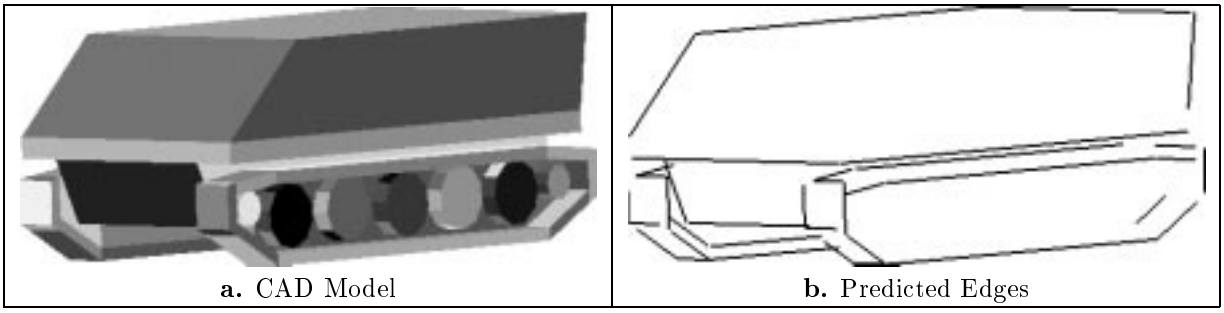
To determine which parts of the CAD model produce the silhouette, a unique color is first assigned to each existing face. This color acts as an index into a hash table of 3-D faces. The model is then rendered from the hypothesized viewing orientation. Rendering is performed on a hardware Z-buffer, and hence can be done very quickly. Running on a Sparc 10 with a ZX accelerator, this process takes roughly 0.3 seconds for a model containing 250 faces. The colors of the resulting pixels indicate which faces are visible. Pixels adjacent to the background color, which is also unique, contribute to the model silhouette. Thus, if the background color appears in a pixel's eight-connected neighborhood, the associated face lies on the silhouette.

Subsequent search determines which specific face boundaries (edges) generate the silhouette. An edge is a possible silhouette edge if only one of the two bounding faces is visible [38]. This step may leave some edges which are actually internal as hypothesized silhouette edges, and it also does not deal with self-occlusion. A clipping algorithm is then used to discover and discard those edges and portions of edges which are not part of the silhouette. The clipping process projects the 3-D model edge endpoints onto the image plane. A line following algorithm then traverses the segment to find the parametric end-points which correspond to the beginning and ending portion of the silhouette edge. Because an orthographic projection is used to render the model, parametric end-point values may be applied directly to the corresponding model edges to produce the resulting 3-D silhouette edges.

4.1.2 Internal Lines

To determine if an edge is likely to cause a significant change in illumination in an image, an estimate of the location of the major light source, the sun in our images, must be made available to the feature prediction algorithm. The sun is modeled as an area light source, and the vector to the sun is calculated using a long/lat estimate, time of day, date, and compass orientation [35]. All of this information is available for our current data set. Once the vector is determined, it provides the direction to the sun for the entire scene, and can be used to predict the internal model edges.

The internal edge prediction is run after the silhouette extraction phase, and therefore all visible faces are known. Each edge of the visible faces is then examined independently, and marked as being a possibly significant internal edge. This list of edges is traversed, and each face which shares that edge is examined. If the dot product of the sun vector with the normal of each face are of the same sign, the edge is removed from this list. This simple test determines edges for which light will be cast onto only one of the visible faces.



a. CAD Model **b.** Predicted Edges

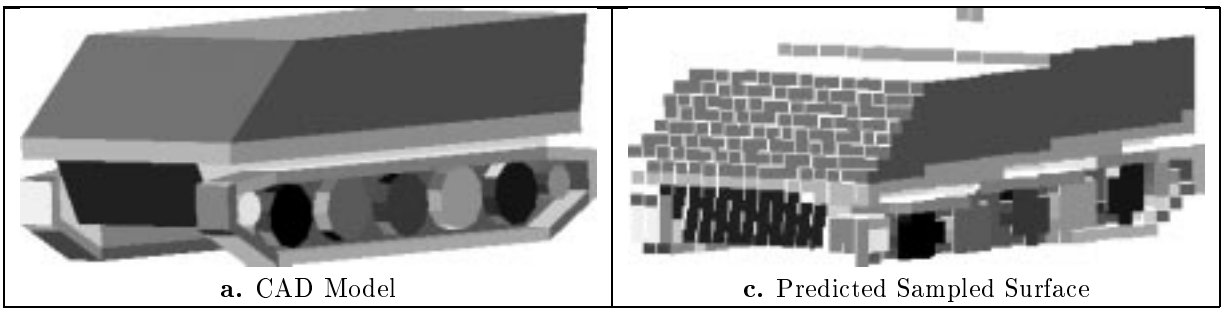
Figure 8: Model Edges Predicted for Matching to the Optical Images for Shot 20

The final pass of the algorithm uses a clipping algorithm similar to that used in obtaining silhouettes: the 3-D edge endpoints are projected onto the image and the parametric endpoint values are determined. The only difference is that our process does not require the edge to lie on the silhouette, and that one of its faces needs to be visible. After both silhouette and internal edges are determined for a given pose hypothesis, shorter lines are discarded using a user specified minimum length threshold. This value was determine empirically, and is currently set to one percent of the maximum vehicle length. Figure 8a shows the CAD model with the face specific coloring

and Figure 8b shows the silhouette and internal edges used in the matching process for the pose hypothesis given in Figure 1.

4.2 Predicting Sampled Surfaces for Range Matching

A 3-D sampled surface is generated in a manner which, in simple terms, simulates the operation of the actual range sensor. The CAD model is transformed into the range sensor's coordinate system using the current estimate of the target position and orientation. Based on the characteristics of the range device, rays are cast into the scene and intersected with the 3-D faces of the CAD model. The results of the rendering step used to extract the silhouette are used here to limit ray intersections to only those faces known to be visible. The closest face intersection is stored as the depth of the current position. By design, noise factors are neglected when generating model features: the intention is to generate a high quality set of model features. Noise is dealt with later when matching the model features to the sensor features. Figure 9a again shows the CAD model in a hypothesized orientation, and Figure 9b shows the predicted sampled surface for matching to the imagery of Figure 1.



a. CAD Model **c.** Predicted Sampled Surface

Figure 9: Model Sample Surface Predicted for Matching to the Range Images for Shot 20

5 Match Error: Relating Predicted Features to Data

A match error function inversely proportional to the quality of a match $\mathcal{M}$ between a set of predicted features and the sensor data may be written as:

$$E_{\mathcal{M}}(\mathcal{F}) > 0 \quad \mathcal{F} \in \Re^{10} \tag{1}$$

where $\mathcal{F}$ represents the geometric relationship between the sensors and the model. In the development below, this is a ten place vector: six values encode the pose of the target relative to the optical sensor (3 rotation and 3 translation), and four values encode the planar translation of each optical image plane relative to the range sensor's image plane (two for each sensor). This set of coupled transformations is further explained in Section 6.1.

The error function is the weighted sum of two components: an error term for the optical data and an error term for the range data:

$$E_{\mathcal{M}}(\mathcal{F}) = (\alpha_{\mathcal{M}})E_{\mathcal{M},\mathcal{O}}(\mathcal{F}) + (1 - \alpha_{\mathcal{M}})E_{\mathcal{M},\mathcal{R}}(\mathcal{F}) \tag{2}$$

where $\alpha_{\mathcal{M}}$ weights the relative importance of the error terms derived from the optical and range imagery. The term $\alpha_{\mathcal{M}}$ was determined empirically, and a constant setting of 0.5 was used for all experiments. The optical image error term $E_{\mathcal{M},\mathcal{O}}$ and range image error term $E_{\mathcal{M},\mathcal{R}}$ are discussed further in the following sections.

The error term for each sensor can alternatively be broken down into an omission error and a fitness error.

$$E_{\mathcal{M},\mathcal{A}}(\mathcal{F}) = \beta_{\mathcal{A}} E_{fit,\mathcal{A}}(\mathcal{F}) + (1 - \beta_{\mathcal{A}})E_{om,\mathcal{A}}(\mathcal{F}) \tag{3}$$

The subscript ($\mathcal{A}$) is replaced below with $\mathcal{O}$ for optical and $\mathcal{R}$ for range. The fitness error $E_{fit,\mathcal{A}}(\mathcal{F})$ represents how well the strongest features (as determined by a threshold) match, and the omission error $E_{om,\mathcal{A}}(\mathcal{F})$ penalizes the match proportional to the fraction of features left unmatched. This happens when no adequate matching features can be found in the sensor data. The term $\beta_{\mathcal{A}}$ was also determined empirically and set to 0.6 for all experiments.

5.1 Optical Image Fitness Error Function

Since we are using two optical images (color and IR), the optical fitness term must be broken down into two constituent components:

$$E_{fit,\mathcal{O}}(\mathcal{F}) = \gamma_{fit} E_{fit,\mathcal{C}}(\mathcal{F}) + (1 - \gamma_{fit}) E_{fit,\mathcal{I}}(\mathcal{F}) \tag{4}$$

where γ_{fit} weights the relative importance of the color and IR error. This term was also determined empirically and set to 0.6 for all experiments. $E_{fit,\mathcal{C}}$ represents the color fitness, and $E_{fit,\mathcal{I}}$ the IR fitness. The fitness terms are formed by examining the estimated gradient magnitude underlying each predicted model line.

Traditional methods for locating objects in optical imagery typically use edge detection algorithms. Local edges [34, 20] may be grouped into larger features such as straight line segments [9, 31]. These linear features, in turn, may be matched to linear features of stored object models [29, 22, 18, 5]. These bottom-up feature extraction algorithms are prone to error [13, 3], and often produce extraneous line segments, fragmented segments, and sometimes over-grouped segments.

Our past work in other problem domains [2] demonstrated that local search, coupled with sound and efficient tests of global alignment, could overcome significant amounts of fragmentation, over-grouping and clutter. However, in this domain we find the features produced by the Burns algorithm [9] are of such poor quality that a top-down rather than a bottom-up approach is preferable. Bottom-up feature extraction is hindered by low resolution, highly textured backgrounds and targets, and finally by similar colors appearing in both camouflage and background.

To overcome these difficulties, the top-down approach first computes the local gradient modulated by a target likelihood derived from the color detection algorithm. This represents a nice example of information being propagated from the detection algorithm onto the multisensor matching algorithm. Next, each predicted model feature is projected into the image plane and a weighted sum for the strength of a given target feature is computed for each predicted model edge. Finally, these sums are converted into a match error term.

5.1.1 Combining Optical Image Gradient and Color Detection Evidence

The gradient magnitude of the image is estimated by convolving all of the pixels in the image with the four different Sobel masks (South, East, SouthWest, SouthEast). These masks are convolved with each image plane (Red, Green and Blue) to produce twelve values per pixel. The maximum of the twelve values is retained as the gradient magnitude for the individual pixel. In order to reduce the effects of the highly cluttered backgrounds, the images are median smoothed before they are convolved. Figure 10a shows the result of convolving Shot 20 with the Sobel masks.

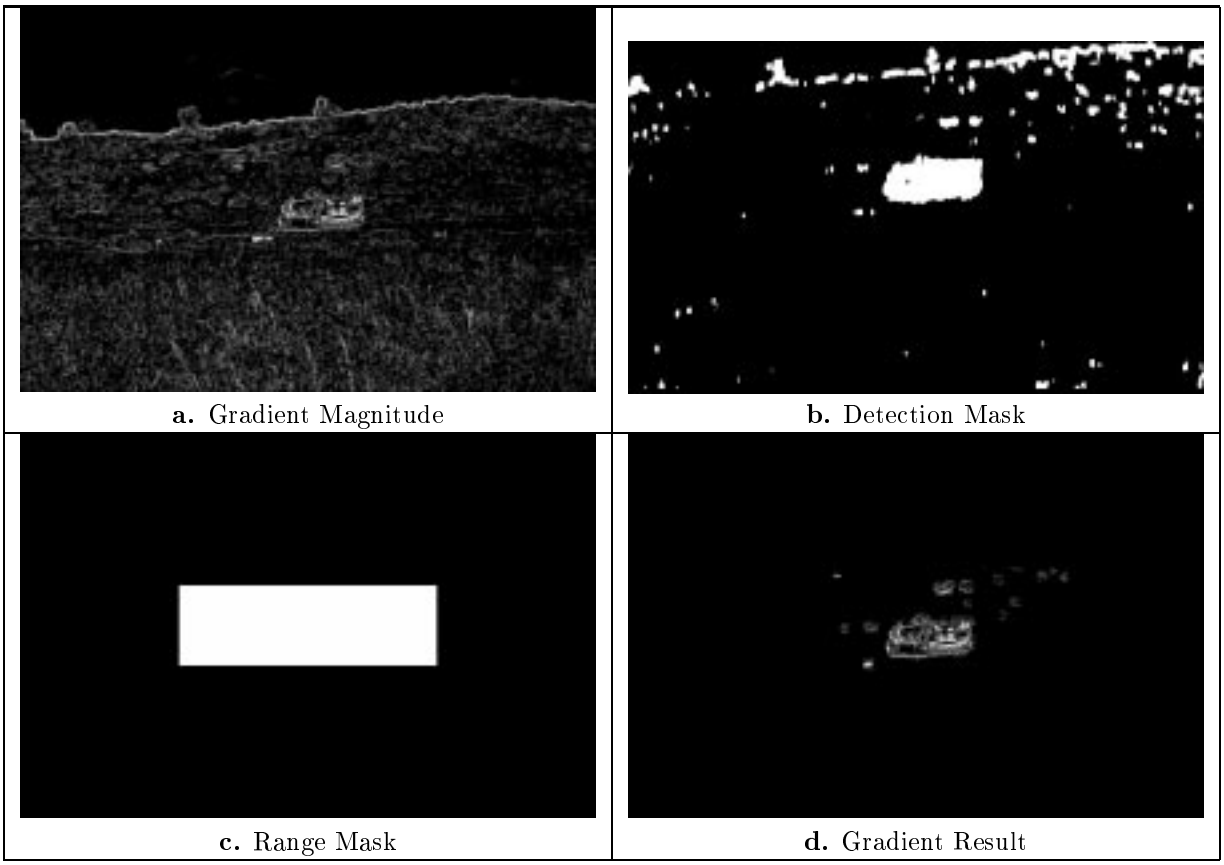
As can be seen from Figure 10a, there are many large regions of high gradient response not associated with the vehicle. In order to reduce the information present which can affect matching performance, the image is convolved with two bit maps. The first bit map is a thresholded version of the detection probabilities shown in Figure 5b. The threshold is selected so any pixel having a probability of target > 0.3 is set to 1, giving the mask shown in Figure 10b.

The next mask used represents whether or not there exists LADAR data for that pixel. Since the multisensor matching algorithm utilizes all three sensor modalities, only the information available in all three can be processed. Figure 10c shows this mask. Because the alignment between the LADAR and color sensors may be in error, the box is increased in size so all possibly relevant color data is included. To be conservative, a fairly large 50 pixel margin is used here.

The result of convolving the image and then masking out the unimportant regions is shown in Figure 10d. The target edge strength as derived in this fashion now pertains mainly to the vehicle. The FLIR image is processed in exactly the same manner, only the detection mask is increased to account for error in the alignment between sensors. These target edge strength measures are next used to estimate the strength of the predicted model lines.

5.1.2 Estimating the Target Edge Strength for a Model Edge

The first step in assessing the strength of each model line is to project it into the optical image. Projecting the 3-D edges into the imagery is possible because both the intrinsic sensor parameters and the approximate pose of the target are known. The parameters for the color sensor have been determined off-line using calibration targets [24]



a. Gradient Magnitude **b.** Detection Mask

c. Range Mask **d.** Gradient Result

Figure 10: Color gradient combined with color target detection output to derived a target edge strength image used in top-down feature matching.

while the IR parameters have been derived from the manufacturer's specifications and interactive adjustment using our visualization tool [42].

The second step is to measure the overall target edge strength under the line segment. A commonly used graphics anti-aliasing technique, known as Pineda Arithmetic [37], is used to determine with subpixel accuracy where the projected line crosses the sensor image. A weighting term is created to scale the edge strength at each pixel, and to obtain the edge strength for a single line. The cumulative strength, $\hat{G}$, for the line segment k is normalized to lie in the range $[0,1]$:

$$\hat{G}(k) = \frac{\sum_{i=X_a}^{X_b} \sum_{j=Y_a}^{Y_b} Gradient(i,j) \cdot w(i,j)}{\sum_{i=X_a}^{X_b} \sum_{j=Y_a}^{Y_b} w(i,j)} \tag{5}$$

where $Gradient(i,j)$ is the target edge strength discussed earlier (normalized to $[0,1]$) for pixel (i,j), and $w(i,j)$ is a weighting term proportional to the distance of the pixel from the true line. The weight $w(i,j)$ is 0 for pixels lying outside the radius of 1.5 pixels from the line segment. The terms X_a, X_b, Y_a, and Y_b represent the pixel bounding box coordinates of line k.

The edge strength, $\hat{G}(k)$ for each line is then converted to an error term. A threshold (v) is used to discard lines with weak responses:

$$\hat{E}(k) = \begin{cases} (1 - \hat{G}(k)) & \hat{G}(k) > v \\ 0 & otherwise \end{cases} \tag{6}$$

the v is initially given a large value so that most lines will match to the data. Because$\hat{G}(k)$ is in the range $[0,1]$, the error term remains normalized to the same range. An annealing schedule is then used as the matching algorithm executes so that over time the system looks for lines with stronger and stronger gradient support.

The fitness error is then formed by summing the error terms and adjusting by the line length. The resulting error for each optical sensor is:

$$E_{fit,\mathcal{O}}(\mathcal{F}) = \frac{\sum_{k\in\Omega} \hat{E}(k) \cdot ||k||}{\sum_{k\in\Omega} ||k||} \tag{7}$$

where Ω represents the set of predicted model lines, and $||k||$ the Euclidean length of the line when projected into the image plane. The fitness is calculated in the same fashion for both the color and IR images. The only difference is that for IR, internal target features are not used.

5.2 Range Image Fitness Error Function

The range fitness error represents how well the predicted 3-D sampled surface model points fit the actual range data. The fitness error is related to the Euclidean distance from a model point to its nearest data point. Let (ψ) be the set of data points and (χ) be the set of predicted model points. The distance between a model point $i \in \chi$ and a data point $j \in \psi$ is measured after the model points are transformed into the data coordinate system using the current coregistration estimate $\mathcal{F}$:

$$\overline{D}(i,j)_{i\in\chi, j\in\psi} = ||i-j|| \tag{8}$$

A good measure of fit is the nearest neighbor Euclidean distance for a model point i:

$$\hat{H}(i) = \overline{D}(i,j) \quad : \forall k \in \psi \; \overline{D}(i,j) \leq \overline{D}(i,k) \tag{9}$$

For reasons to be explained shortly, we choose to further restrict attention to only data points in (ψ) which, when projected into the LADAR image plane, lie in the four-connected neighborhood of the projected model point i. Letting ψ_i be this set of data points, the modified nearest neighbor distance may be written as:

$$\hat{H}'(i) = \overline{D}(i,j) \quad : \forall k \in \psi_i \; \overline{D}(i,j) \leq \overline{D}(i,k) \tag{10}$$

There are two reasons to restrict attention to the points ψ_i. First, by restricting the domain choices it makes computing $\hat{H}'(i)$ somewhat less burdensome. Second, and more importantly, it modifies the sense of 'closeness' in a way which can be advantageous. The new definition of closeness causes model points to be paired with data points lying near a common projection ray, and this tends to produce more globally coherent pairings of points.

The fitness error is a function of the nearest neighbor distances:

$$\hat{E}(i) = \begin{cases} \hat{H}'(i) & \hat{H}'(i) < \tau \\ 0 & otherwise \end{cases} \tag{11}$$

The threshold (τ) places an upper bound on the distance between matching features, and is set to discard points considered too far away to match. Similar to the optical threshold, the τ value is also initially set to a large value and then "cooled" over time. The total fitness for the range sensor is then summed over the matched points and normalized to lie in the range $[0,1]$. Normalization takes account of the number of matched points, p, and the maximum allowable distance τ:

$$E_{fit,\mathcal{R}}(\mathcal{F}) = \frac{\sum_{i\in\chi} \hat{E}(i)}{p \cdot \tau} \tag{12}$$

5.3 Omission Error for All Sensors

Omission accounts for weak responses in optical and unmatched points in range. Omission is needed to prevent fixation upon very small numbers of strongly matched features by favoring matches which account for as many

model features as possible. The general form of the omission error is:

$$E_{om,\mathcal{A}}(\mathcal{F}) = \begin{cases} \frac{e^{\alpha \cdot w}-1}{e^{\alpha}-1} & \alpha \neq 0 \\ w & \alpha = 0 \end{cases} \tag{13}$$

where w is the ratio of unmatched model features over the total number of model features. The parameter α introduces a non-linear bias which essentially reduces the penalty for small amounts of omission while increasing the penalty for large amounts of omission. A detailed explanation of this relationship may be found in [2]. The optical omission error term is a linear combination of the IR and color omission terms:

$$E_{om,\mathcal{O}}(\mathcal{F}) = \gamma_{om} E_{om,\mathcal{C}}(\mathcal{F}) + (1-\gamma_{om}) E_{om,\mathcal{I}}(\mathcal{F}) \tag{14}$$

where γ_{om} weights the relative importance of the color and IR error, and was set to 0.6 for all experiments. For both $E_{om,\mathcal{C}}(\mathcal{F})$ and $E_{om,\mathcal{I}}(\mathcal{F})$ the w (equation 13) is the number of unmatched lines over the total number of lines.

For the range data, omission is measured in both directions: model-to-data and data-to-model. Because the nearest neighbor metric allows many model features to match a single range feature, the matching algorithm can be drawn away from the true solution during the first few iterations of the search. Including a term to measure how much range data is omitted from the match corrects this problem. Thus, range omission is given by:

$$E_{om,\mathcal{R}}(\mathcal{F}) = \begin{cases} \frac{1}{2} \cdot \left(\frac{e^{\alpha p}-1}{e^{\alpha}-1} + \frac{e^{\alpha q}-1}{e^{\alpha}-1}\right) & \alpha \neq 0 \\ \frac{p+q}{2} & \alpha = 0 \end{cases} \tag{15}$$

where p is the ratio of unmatched model points over the total number of model points, and q is the number of unmatched data points over the total number of data points within a fixed distance of the model.

6 Local Search to Find Optimal Multisensor Matches

Local search is an iterative generate-and-test procedure which finds a local minima of the match error developed in the previous section. To better explain the process, we begin by formally setting out the space of transformations which relate the target model to the three sensors. We then define the local neighborhoods used by the iterative search algorithm to explore the space of possible transformations. Next the actual search strategy, the ordering of tests and moves through the space, is discussed. Finally, we discuss how the sampling intervals associated with local neighborhoods are automatically scaled according to measurable properties of the specific matching problem.

6.1 Sensor to Target Model Transformations

The relationships between sensors and model are illustrated in Figure 11. There are four distinct 3-D coordinate reference frames: a world reference $\mathcal{W}$ and the three sensor reference frames $\mathcal{C}$ (color), $\mathcal{I}$ (IR), and $\mathcal{R}$ (range). Without coupling the geometry of the sensors, there would be 18 degrees of freedom associated with rotating and translating independently each sensor relative to the world. The constraints imposed between sensors reduce the combined transformation space to 10 degrees of freedom.

The world reference frame is the target model coordinate system with the model centered at the origin and the positive y axis pointing upward. The 3-D transformation of a point $P_{\mathcal{W}}$ in the world coordinates $\mathcal{W}$ to a point $P_{\mathcal{A}}$ in the reference frame of a sensor $\mathcal{A}$ may be defined as:

$$P_{\mathcal{A}} = M_{\mathcal{W},\mathcal{A}} P_{\mathcal{W}} \quad \text{where} \quad M_{\mathcal{W},\mathcal{A}} = T_{\mathcal{W},\mathcal{A}} T_{\mathcal{W}} R_{\mathcal{W}} S_{\mathcal{W},\mathcal{A}} \quad (\mathcal{A} \in \{\mathcal{C},\mathcal{I},\mathcal{R}\}) \tag{16}$$

In general, the scale transformation $S_{\mathcal{W},\mathcal{A}}$ would be used to convert between units of measurement. However, because the current system uses a canonical representation of meters for all of the coordinate systems, $S_{\mathcal{W},\mathcal{A}}$ is set to the identity transform.

The rotation $R_{\mathcal{W}}$ rotates points about the origin of the world (model) coordinate system in order to alter the relative orientation of the object model with respect to the sensors. The same rotation matrix is used for all sensor coordinate systems. The final two transformations, $T_{\mathcal{W}}$ and $T_{\mathcal{W},\mathcal{A}}$, translate the points in the world relative to the sensors.

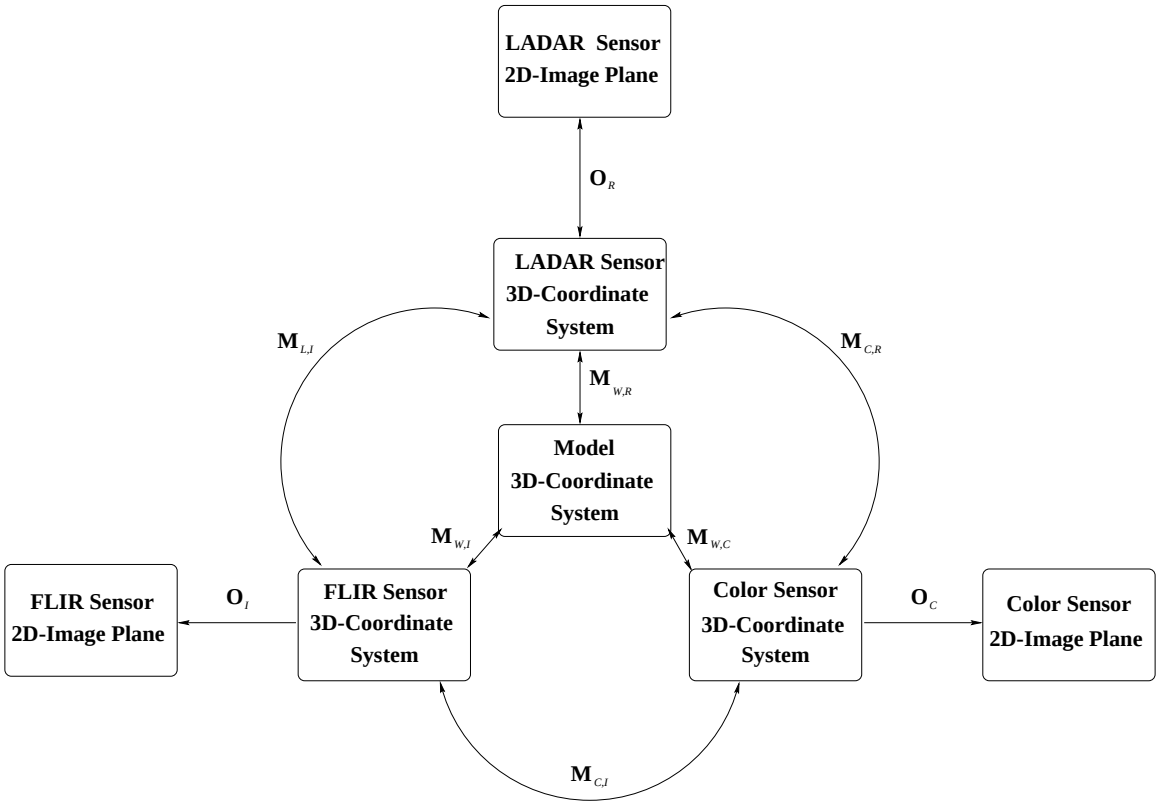


Figure 11: Reference Frames

Local search distinguishes between translation in depth relative to the sensors and translation in a common XY image plane shared by all three sensors. The XY plane translations $T_{\mathcal{W},\mathcal{A}}$ are independent for each sensor. The depth, or Z axis translation ($T_{\mathcal{W}}$) is the same for all three sensors. As already discussed in Section 1.1, the choice of this particular parameterization derives from the assumption that our sensors are co-located and nearly boresight aligned.

Given these constraints, the world ($\mathcal{W}$) to sensor ($\mathcal{A}$) transformation may now be written as:

$$M_{\mathcal{W},\mathcal{A}} \;=\; T_{\mathcal{W},\mathcal{A}}\, T_{\mathcal{W}}\, R_{\mathcal{W}} \;=\; \begin{vmatrix} 1 & 0 & 0 & T_{\mathcal{A},x} \\ 0 & 1 & 0 & T_{\mathcal{A},y} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \begin{vmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T_z \\ 0 & 0 & 0 & 1 \end{vmatrix} \begin{vmatrix} r_{xx} & r_{xy} & r_{xz} & 0 \\ r_{yx} & r_{yy} & r_{yz} & 0 \\ r_{zx} & r_{zy} & r_{zz} & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \tag{17}$$

The transformations between sensor reference frames are coupled. Using the constraints specified by Equation 17, the mapping between each sensor frame is constrained to two degrees of freedom and the mapping between any two sensors $\mathcal{A}$ and $\mathcal{B}$ may be written as:

$$M_{\mathcal{A},\mathcal{B}} \;=\; T_{\mathcal{A},\mathcal{B}} \tag{18}$$

This translation between two sensors may alternatively be expressed in terms of the translation of each sensor relative to the world.

$$T_{\mathcal{A},\mathcal{B}} \;=\; T_{\mathcal{W},\mathcal{A}} - T_{\mathcal{W},\mathcal{B}} \;=\; \begin{vmatrix} 1 & 0 & 0 & T_{\mathcal{A},x} - T_{\mathcal{B},x} \\ 0 & 1 & 0 & T_{\mathcal{A},y} - T_{\mathcal{B},y} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \tag{19}$$

The three sensor-to-world transformations are not independent: knowing any two determines the third.

$$\begin{array}{lll} M_{\mathcal{W},\mathcal{C}} \;=\; M_{\mathcal{R},\mathcal{C}} M_{\mathcal{W},\mathcal{R}} & M_{\mathcal{W},\mathcal{I}} \;=\; M_{\mathcal{R},\mathcal{I}} M_{\mathcal{W},\mathcal{R}} & M_{\mathcal{W},\mathcal{R}} \;=\; M_{\mathcal{C},\mathcal{R}} M_{\mathcal{W},\mathcal{C}} \\ M_{\mathcal{W},\mathcal{C}} \;=\; M_{\mathcal{I},\mathcal{C}} M_{\mathcal{W},\mathcal{I}} & M_{\mathcal{W},\mathcal{I}} \;=\; M_{\mathcal{C},\mathcal{I}} M_{\mathcal{W},\mathcal{C}} & M_{\mathcal{W},\mathcal{R}} \;=\; M_{\mathcal{I},\mathcal{R}} M_{\mathcal{W},\mathcal{I}} \end{array} \tag{20}$$

The local search algorithm uses these transformations to construct a neighborhood of possible moves relative to the current coregistration estimate. The match error term indicates which transformations are better than others and allows the generate-and-test search algorithm to refine the initial estimate and move towards a more consistent global coregistration.

6.2 Defining Local Neighborhoods

The search process begins with an estimate from the target type and pose hypothesis phase. This estimate defines a set of homogeneous transformations shown in equation 21. Together, these transformations allow the sensor suite to rotate and translate freely relative to the target and for the sensors to translate relative to each other in a common imaging plane.

$$\mathcal{M}_{pose} \in \Re^4 = \left\{ \begin{array}{rcl} \mathcal{M}_1 & = & R_{\mathcal{W}}(x), \\ \mathcal{M}_2 & = & R_{\mathcal{W}}(y), \\ \mathcal{M}_3 & = & R_{\mathcal{W}}(z), \\ \mathcal{M}_4 & = & T_{\mathcal{W}}(z) \end{array} \right\} \qquad \mathcal{M}_{reg} \in \Re^6 = \left\{ \begin{array}{rcl} \mathcal{M}_5 & = & T_{\mathcal{W},\mathcal{C}}(x), \\ \mathcal{M}_6 & = & T_{\mathcal{W},\mathcal{C}}(y), \\ \mathcal{M}_7 & = & T_{\mathcal{W},\mathcal{I}}(x), \\ \mathcal{M}_8 & = & T_{\mathcal{W},\mathcal{I}}(y), \\ \mathcal{M}_9 & = & T_{\mathcal{W},\mathcal{R}}(x), \\ \mathcal{M}_{10} & = & T_{\mathcal{W},\mathcal{R}}(y) \end{array} \right\} \tag{21}$$

The transformations are divided into two sets: $\mathcal{M}_{pose}$ and $\mathcal{M}_{reg}$. The names derive from an earlier decomposition where the 6 degrees-of-freedom (dof) representing sensor suite pose were expressed in $\mathcal{M}_{pose}$ and the 4 dof representing relative sensor registration were expressed in $\mathcal{M}_{reg}$. This decomposition was dropped in favor of the one shown because the first was found to cause problems in the early phases of the search process. The decomposition in equation 21 encodes sensor translation in the common xy plane independently for each of the three sensors. Hence, $\mathcal{M}_{reg}$ now encodes 6 dof. Correspondingly, $\mathcal{M}_{pose}$ encodes rotation and translation only along the z dimension. These two sets are used independently in two alternating stages of the local search process.

As formulated below, local search repeatedly examines a discrete neighborhood of alternatives defined relative to the current best estimate. The neighborhood is obtained by sampling a predetermined number of points along a bounded interval centered about the current estimate. This is done independently for each of the 10 dimensions of the coregistration space. Thus, a set of offsets are applied to the matrices of equation 21 giving a set of new transformations representing new points in the search space.

$$\mathcal{M}'_{pose} = \left\{ \begin{array}{rcl} \mathcal{M}_1 & = & R_{\mathcal{W}}(x) \cdot (\pm \Delta R_x), \\ \mathcal{M}_2 & = & R_{\mathcal{W}}(y) \cdot (\pm \Delta R_y), \\ \mathcal{M}_3 & = & R_{\mathcal{W}}(z) \cdot (\pm \Delta R_z), \\ \mathcal{M}_4 & = & T_{\mathcal{W}}(z) \pm \Delta T_z \end{array} \right\} \qquad \mathcal{M}'_{reg} = \left\{ \begin{array}{rcl} \mathcal{M}_5 & = & T_{\mathcal{W},\mathcal{C}}(x) \pm \Delta T_x, \\ \mathcal{M}_6 & = & T_{\mathcal{W},\mathcal{C}}(y) \pm \Delta T_y, \\ \mathcal{M}_7 & = & T_{\mathcal{W},\mathcal{I}}(x) \pm \Delta T_x, \\ \mathcal{M}_8 & = & T_{\mathcal{W},\mathcal{I}}(y) \pm \Delta T_y, \\ \mathcal{M}_9 & = & T_{\mathcal{W},\mathcal{R}}(x) \pm \Delta T_x, \\ \mathcal{M}_{10} & = & T_{\mathcal{W},\mathcal{R}}(y) \pm \Delta T_y \end{array} \right\} \tag{22}$$

How these offsets are selected is discussed below.

6.3 Tabu Search: Moving through the Search Space

The simplest way to use these moves to generate new states would be to apply them directly, neglecting any intermediate states. This would generate a discrete neighborhood consisting of only those coregistration values at the bounds of the intervals defined by the matrices in (21). However, this can lead to missed chances for improvement when a better estimate lies within the interval, but not at its upper or lower bound. To protect against such missed opportunities, coregistration values internal to the interval are sampled. This modestly increases the size of the discrete neighborhood, but greatly improves the performance of the algorithm.

For a given delta value in equation 22, the upper and lower bound are replaced by a set of possible values. The specific strategy used to sample the intervals is loosely based upon binary search. The underlying idea is

that the sampling interval should be successively reduced by half as samples approach the current estimate. For translation matrices, this is done by successively multiplying the delta by 0.5 until the spacing drops below an absolute threshold. For example, the ΔT_z in Equation 22 is replaced by the following set of values:

$$\Delta T_z = \left\{ \pm \Delta T_z \left(\frac{1}{2}\right)^0, \pm \Delta T_z \left(\frac{1}{2}\right)^1, \pm \Delta T_z \left(\frac{1}{2}\right)^2, ... \right\} \tag{23}$$

The local search algorithm uses the transformations defined in equation 22, along with the internal sampling strategy, to generate alternative coregistration estimates which are each evaluated using the match error defined in Section 5. As mentioned above, search alternates between considering moves from $\mathcal{M}_{pose}$ and $\mathcal{M}_{reg}$. When considering moves, a greedy strategy is used in which the move yielding the greatest drop in match error is selected.

After every move, new model features are generated using the algorithm described in Section 4. One consequence of feature regeneration is that the match error landscape about the current estimate can change. At times, this change in the landscape causes a move back to the previous state to appear most attractive. In keeping with the underlying concept of Tabu Search [15], our search algorithm keeps a modest history of past states. This history is used to prevent cycling back to previously visited parts of the search space.

While the regeneration of features does complicate the search process, it is critical to the success of our approach. Regeneration allows search to correct for significant errors in the initial coregistration estimate. Entire faces on the model can come into and out of view. Tabu search is therefore essential to prevent cyclical behavior.

The search continues to examine neighborhoods until no further improvement can be made along any dimension. At this point the parameters to the error terms are cooled using the annealing schedule. Once the parameters are completely cooled, the current set of transformations are returned as the final locally optimal coregistration.

6.4 Automatically Adjusting the Neighborhood Sampling Intervals

Obviously, the lower and upper bounds (or step sizes) are extremely important. We are using several simple heuristics to determine the optimal translation and rotation step sizes. After each time the model features are predicted for the range data, a simple moment analysis is done. This moment analysis sets the upper and lower bounds for the search. Once a move to a better state is made and features are re-generated, the moment analysis is performed again.

The moment analysis begins with examination of the predicted model sampled surface points and the range data. The average Euclidean distance in each dimension between the non-omitted model points and the corresponding data points is determined. The average along each dimension provides the translation step sizes (ΔT_x, ΔT_y, ΔT_z). To prevent radical or minute changes, upper and lower bounds are used to cap this value.

To select bounds on the rotations, the model and data points are orthographically projected onto each of the three axis planes. A least squares algorithm fits a line to the non-omitted points of the data and model in each of the three dimensions. The dot product of the data and model lines for each dimension provides the upper and lower rotational bounds(ΔR_x, ΔR_y, ΔR_z). Again to prevent unstable behavior, upper and lower bounds are set on this value.

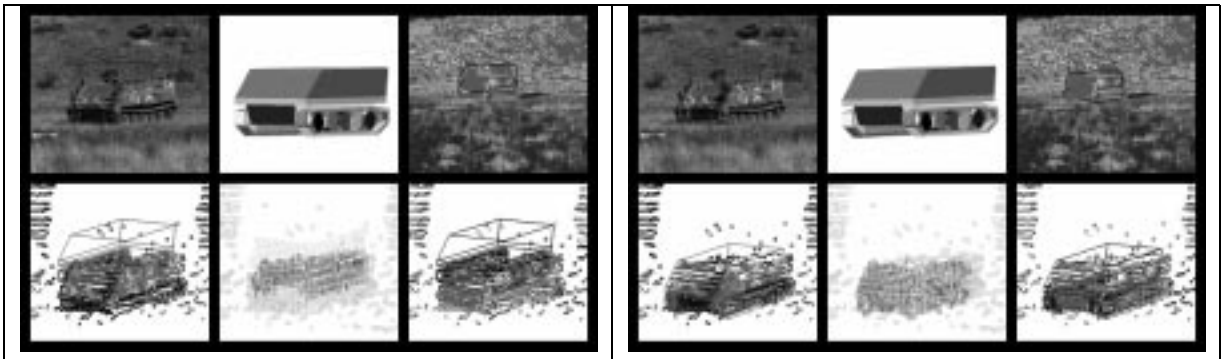
7 Results

The complete ATR system has been applied to fifteen range, color and IR images triples in the Fort Carson data set. Six of those fifteen were used in development of the algorithms and the results on those images are not reported here. For the nine image triples remaining, we present a statistical analysis geared towards assessing how well the multisensor matching algorithm performed. In addition, example results on the three image triples presented in Figures 1, 2 and 3 are displayed to illustrate the results of multisensor matching.

7.1 Results on Three Example Images

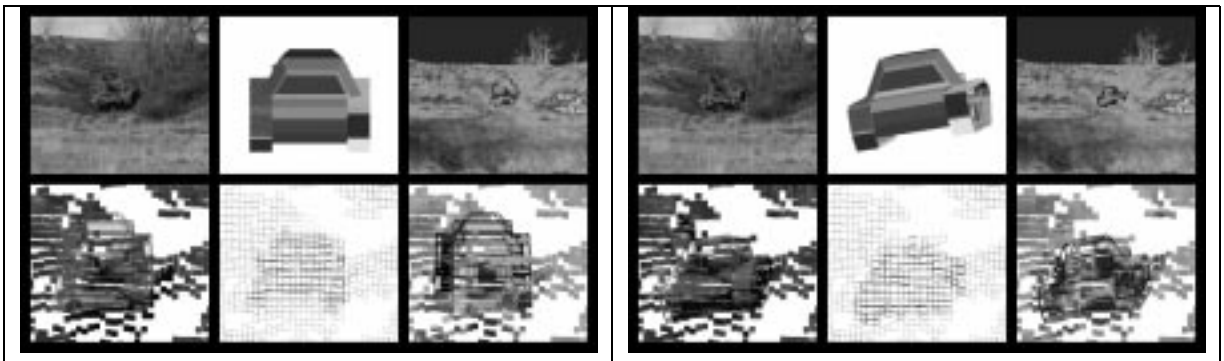
The color detection algorithm successfully detected the targets in the images shown in Figures 1c, 2c and 3c. The pose hypothesis algorithm then provided a sequence of possible target type and pose hypotheses. The multisensor matching algorithm then refined the estimate to correct for pose and alignment errors. The results illustrated

below show the best match found by the multisensor matching algorithm. Recall that the best match is that which minimizes the match error defined in Section 5.



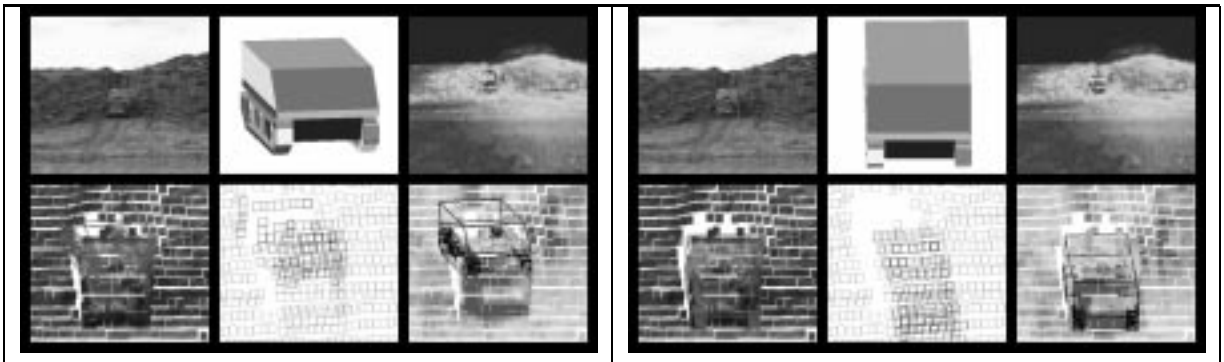
a. Initial Coregistration **b.** Refined Coregistration

Figure 12: Shot20 Multisensor Target Matching Results



a. Initial Coregistration **b.** Refined Coregistration

Figure 13: Shot26 Multisensor Target Matching Results



a. Initial Coregistration **b.** Refined Coregistration

Figure 14: Shot31 Multisensor Target Matching Results

Figures 12a, 13a, and 14a show the initial starting hypothesis for the matching algorithm. Starting from the top left corner of the image and moving clockwise, each image chip represents either different sensor-to-model relationships, or the sensor-to-sensor alignment. The upper left image shows the color image with the predicted model edges drawn in red and blue (red represents a non-omitted model feature). The next image shows the model in the initial orientation, followed by the IR image with the lines in white and black (black is non-omitted).

In the bottom row, the leftmost image shows the wireframe model in relation to the range data. The range data has been texture mapped with the color imagery, which allows the alignment between sensors to be visually assessed. The middle image shows the predicted model features in relation to the range sensor data. The blue boxes are data points and the red and yellow are predicted model points (red is non-omitted). The rightmost chip represents the range data with a IR texture map.

Figures 12b, 13b, and 14b show the resulting pose and alignment after the multisensor matching system has refined these transformations. As can be seen from careful examination of the before and after imagery, the matching algorithm was able to substantially improve upon the model-to-sensor as well as the sensor-to-sensor relationships.

The multisensor matching algorithm took roughly 45 to 90 seconds to converge from the initial to final estimates for Shot31 and Shot26. Shot20 took slightly longer, at 120 seconds, due largely to the greater number of range data points on target. Shot20 required 10 iterations of the local search algorithm and roughly 700 match error evaluations. Shot31 required 15 iterations and roughly $1,500$ match evaluations, and finally Shot26 required 21 iterations and roughly $3,000$ evaluations.

7.2 Statistical Analysis on Fort Carson Data Set

The pose recovery and image registration performance of the algorithm has been analyzed for the nine image triples. For each triple, a ground truth estimate was manually determined. The ground truth established the correct target rotation and translation as well as the alignment between sensors. This ground truth estimate was establish by hand using our visualization system [42]. Due to the coarse sampling of the range data, and the few number of pixels on target in the optical imagery, we expect there to be a slight amount of error in the ground truth [5].

The detection algorithm successfully found each vehicle in the scene, and provided the focus of attention for hypothesis generation algorithm. This algorithm then generated a set of initial target type and pose estimates.

We then set out to answer two questions. The first pertained to the pose of the object in the scene. We were interested in determining how well the multisensor algorithm corrects for rotational and translational errors made by the hypothesis generation algorithm. To illustrate, if the hypothesis is off by 30 degrees relative to the ground truth, can multisensor matching recover a new pose estimate significantly closer to the true target orientation. The second question is how sensitive is the multisensor matching process to initial errors in the alignment between sensors.

To answer these questions, three sets of experiments were run on each image. In the first case, the alignment between sensors, and hence the image registration, was set to the hand generated ground truth. In the second case, this alignment was randomly perturbed in both X and Y by up to 0.5 meters. The third case perturbed the alignment by up to 1.0 meters. Loosely speaking, a 1 meter error with our sensors corresponds to a 5 range pixel mis-registration at 100 meters.

Sensor-to-sensor alignment is used to map color detection results into the range imagery, and to map the initial hypotheses into the three sensors at the start of multisensor matching. Hence, each of the three cases provides a distinct set of inputs for the multisensor matching algorithm. For each case, the top five target-pose hypotheses coming from the range boundary probing algorithm are provided as starting states for the multisensor matching algorithm. Thus, for each of the 9 image triples, the multisensor matching algorithm is run 15 times, yielding a total of 135 runs of the algorithm.

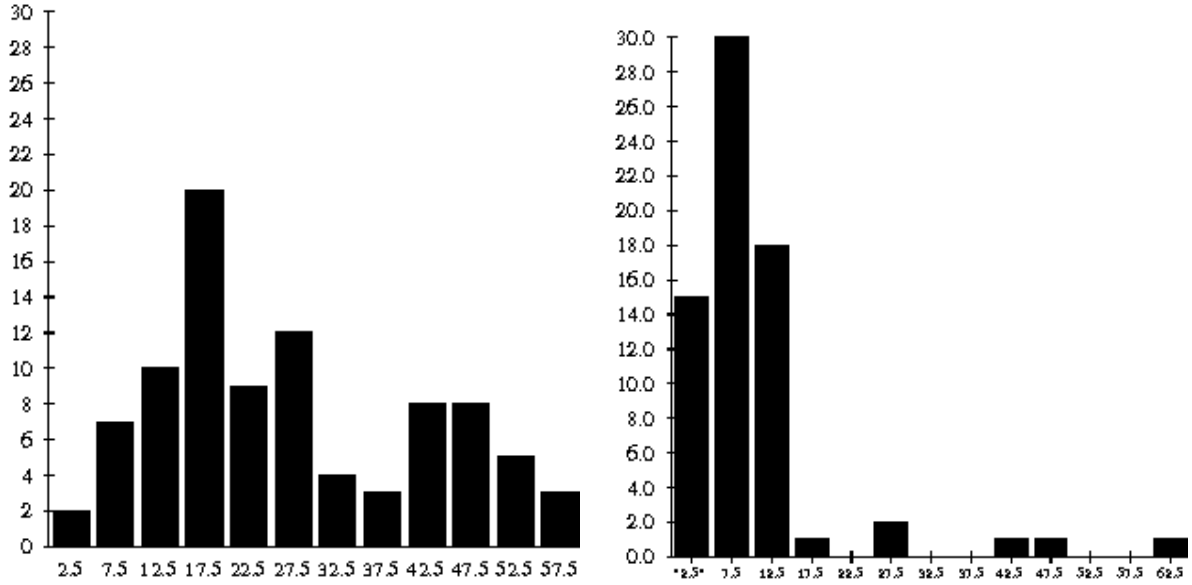
For each run several values are recorded. First, the initial state was compared to the ground truth estimate to give an error measure in rotation, translation and alignment. The multisensor matching system was then run on each of these estimates and the error measures were recomputed to assess if the resulting states were improvements over the original. Statistical analysis was then done to determine how well the system did in terms of correcting for rotation, translation and alignment error.

7.2.1 Rotational Error

The rotational error, measured in degrees, for the initial and final states forms two distinct and disjoint distributions. After examining the initial starting states, it became obvious that the estimates are either within 60° of correct

[5] We estimate this error to be about 0.5° in rotation and up to $0.3m$ in translation and alignment.

or they are off by roughly 180°. This corresponds to confusing the front of the vehicle with the back. The current local search strategy cannot recover correct vehicle pose when front is confused with back. Acknowledging that matching will fail on such grossly incorrect estimates, attention is focused upon the majority of pose estimates which lie within 60° of the true vehicle orientation.



a. Before **b.** After

Figure 15: Comparison of Rotational Error on 9 Images

For those hypotheses within 60° of the true orientation, the resulting distributions for the before and after rotational errors are shown in Figure 15a and 15b. The final rotation distribution has mean 9.4577 ($\sigma = 8.94$) [6] and the initial rotation has a mean of 26.6703 ($\sigma = 15.18$). The mean comparison may not be the best indicator of performance since the distributions are so skewed. The median is a better predictor of the average case, and the median of the initial distribution is 22.36 and the median of the final distribution is 7.44.

Running a normalized t-test on the two distributions shows that there is a statistically significant difference ($t = -9.8527, p \leq 0.0001$) in the two distributions. A comparison of the two means shows the multisensor matching system is able to correct for a large amount of rotational error generated by the pose hypothesis phase.

7.2.2 Translational Error

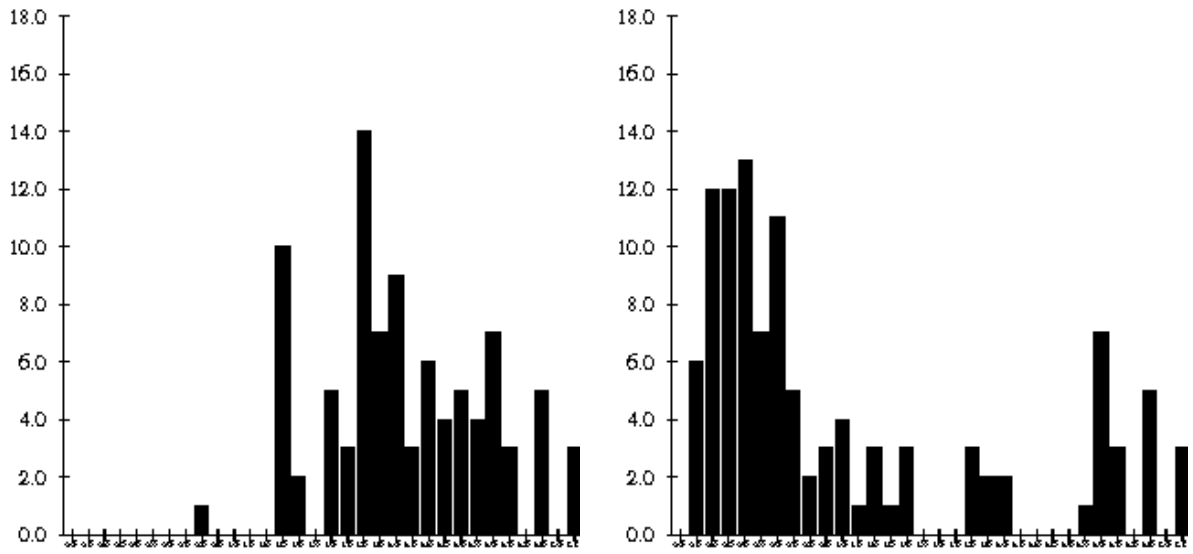
The translational error was compared in much the same manner as the rotational error. The histograms are shown in Figures 16a and 16b. The mean of the final translation distribution is 0.6931 ($\sigma = 0.28$) [7] and the initial translation mean is 2.10 ($\sigma = 0.506$). Again, the two distributions were compared using the t-test, and with a t value of -7.8353 ($p \leq 0.0001$), the distributions are significantly different. The median for the initial distribution is 2.06 and the median for the final distribution is 0.51. Thus the multisensor matching algorithm was able to correct for a large amount of translational error.

7.2.3 Alignment Error

The error in alignment between the sensors was analyzed in a slightly different manner. First the data set was broken down into the three cases described above: no alignment error, slight alignment error (up to 0.5 meters), and large alignment error (up to 1.0 meter). The three different distributions were then compared using the Anova statistic. The Anova compares the means of each group to the mean of the groups combined. The result can tell

[6] All rotational values are given in degrees.

[7] All translational values are given in meters.



a. Before **b.** After

Figure 16: Comparison of Translational Error on 9 Images

us how sensitive the search algorithm is to the alignment error. In other words, it answers the following question: if a single trial result was pulled from the entire set of trials examined, would there be enough information from the alignment error value to determine which class the trial came from?

The Anova returned an F of 0.6173 with 3 degrees of freedom and $p \leq 0.6056$. Thus, the multisensor algorithm performed equally well on all three cases, and consequently, it is not sensitive to modest errors in the initial sensor-to-sensor alignment estimates. Had F been larger it would have meant if we pulled a random trial from the pool of possible trials, the error could be used to determine which case the trial came from.

This is a highly encouraging result, suggesting we have succeeded in developing a multisensor matching capability which will perform robustly even when sensor alignments vary in the field. Further study should assess how target resolution affects the algorithm's ability to deal with errors in alignment, as well as the relationship between alignment error and target resolution.

8 Discussion and Conclusion

The work presented here represents a major step toward our ultimate goal of more reliable target identification through precise 3-D geometric fusion of multisensor data. What we have shown is that the combination of on-line feature prediction, multisensor match evaluation, and robust local optimization can successfully bring 3-D target models into alignment with IR, color and range data. The result is a model-based fusion of heterogeneous sensor data.

For all the results presented in Section 7, a single set of parameters was used. No image specific tuning was performed, and the parameters used were selected empirically from tests on training images not included in our test imagery. While the algorithm does not explicitly model sensor noise, the match error appears to be robust with respect to the numerous sources of clutter and uncertainty present in the Fort Carson dataset. The algorithm performs well for targets at unusual orientations, i.e. not simply for targets rotated about the vertical at a fixed depression angle. It also is handles cases such as the M60 in the gully, where the terrain obscures part of the target and provides significant clutter in the range data.

Currently, we are refining and extending the match evaluation measure. Now that range and optical imagery are precisely aligned, we are working on ways to use occlusion detection in range to explain omitted features in the optical imagery. This is expected to greatly improve performance for occluded targets. Normalizing the match evaluation across targets is also being taken up in order to permit better discrimination between targets.

References

[1] J. K. Aggarwal. Multisensor Fusion for Automatic Scene Interpretation. In Ramesh C. Jain and Anil K. Jain, editors, *Analysis and Interpretation of Range Images*, chapter 8. Springer-Verlag, 1990.

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

[3] J. Ross Beveridge, Joey Griffith, Ralf R. Kohler, Allen R. Hanson, and Edward M. Riseman. Segmenting images using localized histograms and region merging. *International Journal of Computer Vision*, 2(3):311 – 347, January 1989.

[4] 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.

[5] J. Ross Beveridge and Edward M. Riseman. Optimal Geometric Model Matching Under Full 3D Perspective. *Computer Vision and Image Understanding*, 61(3):351 – 364, 1995. (short version in IEEE Second CAD-Based Vision Workshop).

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

[7] R. C. Bolles and R. A. Cain. Recognizing and Locating Partially Visible Objects: The Local-Feature-Focus Method. *International Journal of Robotics Research*, 1(3):57 – 82, 1982.

[8] Shashi Buluswar, Bruce A. Draper, Allen Hanson, and Edward Riseman. Non-parametric Classification of Pixels Under Varying Outdoor Illumination. In *Proceedings: Image Understanding Workshop*, pages 1619–1626, Los Altos, CA, November 1994. ARPA, Morgan Kaufmann.

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

[10] Jin-Long Chen and George C. Stockman. Determining pose of 3d objects with curved surfaces. Technical Report CPS-93-40, Michigan State University, 1994.

[11] Jin-Long Chen, George C. Stockman, and Kashi Rao. Recovering and tracking pose of curved 3d objects from 2d images. In *Proceedings Computer Vision and Pattern Recognition*, pages 233–239, June 1993.

[12] C. H. Chien and J. K. Aggarwal. Shape recognition from single silhouettes. In *International Conference on Computer Vision*, pages 481–490, 1987.

[13] James J. Clark. Authenticating Edges Produced by Zero-Crossing Algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-11(1):43–57, January 1989.

[14] Richard L. Delanoy, Jacques G. Verly, and Dan E. Dudgeon. Machine Intelligent Automatic Recognition of Critical Mobile Targets in Laser Radar Imagery. *The Lincoln Laboratory Journal*, 6(1):161–186, Spring 1993.

[15] F. Glover. Tabu search – part i. *ORSA Journal on Computing*, 1(3):190 – 206, 1989.

[16] M. E. Goss, J. R. Beveridge, M. Stevens, and A. Fuegi. Three-dimensional visualization environment for multisensor data analysis, interpretation, and model-based object recognition. In *IS&T/SPIE Symposium on Electronic Imaging: Science & Technology*, pages 283 – 291, February 1995.

[17] Michael E. Goss, J. Ross Beveridge, Mark Stevens, and Aaron Fuegi. Visualization and Verification of Automatic Target Recognition Results Using Combined Range and Optical Imagery. In *Proceedings: Image Understanding Workshop*, pages 491 – 494, Los Altos, CA, November 1994. ARPA, Morgan Kaufmann.

[18] W. Eric L. Grimson and Daniel P. Huttenlocher. On the Verification of Hypothesized Matches in Model-Based Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(12):1201 – 1213, December 1991.

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

[20] Ellen C. Hildreth. The Detection of Intensity Changes by Computer and Biological Vision Systems. *Computer Vision, Graphics, and Image Processing*, 22:1–27, 1983.

[21] Anthony Hoogs and Douglas Hackett. Model-supported exploitation as a framework for image understanding. In *Proceedings: Image Understanding Workshop*, pages 265–268. ARPA, nov 1994.

[22] Daniel P. Huttenlocher and Shimon Ullman. Recognizing Solid Objects by Alignment with an Image. *International Journal of Computer Vision*, 5(2):195 – 212, November 1990.

[23] J. Ross Beveridge and Bruce A. Draper and Kris Siejko. Progress on Target and Terrain Recognition Research at Colorado State University. In *Proceedings: Image Understanding Workshop*, page (to appear), Los Altos, CA, February 1996. ARPA, Morgan Kaufman.

[24] J. Ross Beveridge and Mark R. Stevens and 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.

[25] B. W. Kernighan and S. Lin. An efficient heuristic procedure for partitioning graphs. *Bell Systems Tech. Journal*, 49:291 – 307, 1972.

[26] J.J. Koenderink. What does occluding contour tell us about solid shape? *Perception*, 13:321–330, 1984.

[27] S. Lin and B. Kernighan. An effective heuristic algorithm for the traveling salesman problem. *Operations Research*, 21:498 – 516, 1973.

[28] Cheng-Hsiung Liu and We-Hsiang Tsai. 3d curved object recognition from multiple 2d camera views. *Computer Vision, Graphics and Image Processing*, 50:177–187, 1990.

[29] David G. Lowe. *Perceptual Organization and Visual Recognition*. Kluwer Academic Publishers, 1985.

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

[31] David G. Lowe and T. O. Binford. The Perceptual Organization of Visual Images: Segmentation as a Basis for Recognition. In *Proceedings Image Understanding Workshop, Stanford*, pages 203 – 209, June 1983.

[32] M. J. Magee, B. A. Boyter, C. H. Chien, and J. K. Aggarwal. Experiments in Intensity Guided Range Sensing Recognition of Three-Dimensional Objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 7(6):629 – 637, November 1985.

[33] David Marr. Analysis of occluding contour. *Proceedings of the Royal Society of London*, B197:441–475, 1977.

[34] David Marr and Ellen C. Hildreth. Theory of Edge Detection. *Proceedings of the Royal Society of London*, B207:187–217, 1980.

[35] G.W. Paltridge and C.M.R Platt. *Radiative Processes in Meteorology and Climatology*. Elsevier Scientific Publishing Company, 1976.

[36] Christos H. Papadimitriou and Kenneth Steiglitz. *Combinatorial Optimization: Algorithms and Complexity*, chapter Local Search, pages 454 – 480. Prentice–Hall, Englewood Cliffs, NJ, 1982.

[37] Juan Pineda. A Parallel Algorithm for Polygon Rasterization. In *Proceedings of Siggraph '88*, pages 17–20, 1988.

[38] W. Brent Seales and Charles R. Dyer. Modeling the Rim Appearance. In *Proceedings of the 3rd International Conference on Computer Vision*, pages 698–701, 1992.

[39] A. Stentz and Y. Goto. The CMU Navigational Architecture. In *Proceedings: Image Understanding Workshop*, pages 440–446, Los Angeles, CA, February 1987. ARPA, Morgan Kaufmann.

[40] Mark R. Stevens. Obtaining 3D Silhouettes and Sampled Surfaces from Solid Models for use in Computer Vision. Master's thesis, Colorado State University, Fort Collins, Colorado, September 1995.

[41] Mark R. Stevens, J. Ross Beveridge, and Michael E. Goss. Reduction of BRL/CAD Models and Their Use in Automatic Target Recognition Algorithms. In *Proceedings: BRL-CAD Symposium*. Army Research Labs, June 1995.

[42] Mark R. Stevens, J. Ross Beveridge, and Mike Goss. Visualization of multi-sensor model-based object recognition. *IEEE Transactions on Visualization and Computer Graphics*, (submitted).

[43] U. S. Army Ballistic Research Laboratory. *BRL-CAD User's Manual*, release 4.0 edition, December 1991.

[44] Jacques G. Verly, Dan E. Dudgeon, and Richard T. Lacoss. Model-Based Automatic Target Recognition System for the UGV/RSTA Ladar: Status at Demo C. In *Proceedings: Image Understanding Workshop*, pages 549–583. ARPA, February 1996.

[45] T.P. Wallace and P.A. Wintz. An efficient three-dimensional aircraft recognition algorithm using normalized Fourier descriptors. *Computer Graphics and Image Processing*, 13:99–126, 1980.

[46] Y. F. Wang, M. J. Magee, and J. K. Aggarwal. Matching three-dimensional objects using silhouettes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:513–518, 1984.