Global Feature Space Neural Network for Active Object Recognition

Michael A. Sipe and David Casasent
Deptartment of Electrical and Computer Engineering,
Carnegie Mellon University, Pittsburgh, PA 15213
sipe@ieee.org, casasent@ece.cmu.edu

Abstract

We present new test results for our active object recognition algorithms which are based on the feature space trajectory (FST) representation of objects and a neural network processor for computation of distances in global feature space. The algorithms are used to classify and estimate the pose of objects in different stable rest positions and automatically re-position the camera if the class or pose of an object is ambiguous in a given image. Multiple object views are used in determining both the final object class and pose estimate. An FST in eigenspace is used to represent 3-D distorted views of an object. FSTs are constructed using images rendered from solid models. The FSTs are analyzed to determine the camera positions that best resolve ambiguities in class or pose. Real objects are then recognized from intensity images using the FST representations derived from rendered imagery.

I. Introduction

Object recognition involves processing sensor data in order to assign a *class* label (e.g. a part number) from among a limited number of valid possibilities and estimate the *pose* (i.e. position and orientation) of a three dimensional object. We consider *active object recognition* which implies the ability to systematically reposition the sensor to make the object recognition task easier. In our work, we assume that we have the ability to move a CCD camera relative to the object and take additional images to reduce ambiguity in scene interpretation. This involves estimation of a rigid object's class and pose from one view and using this information, and our object representation, to determine where to look next. Consider the problem of estimating the pose of socket 1 in Fig. 1b. The pose of the object is ambiguous because the appearance of the object is identical at 90° and 270°; however, if the viewpoint is rotated to obtain the 0° or 180° views, the pose of the object may be determined by the different hole sizes in the front (Fig. 1a) or back (Fig. 1c) of the socket. Active object recognition has many potential uses in manufacturing including identifying, verifying or sorting parts, on-line defect detection, and vision-guided assembly [1].

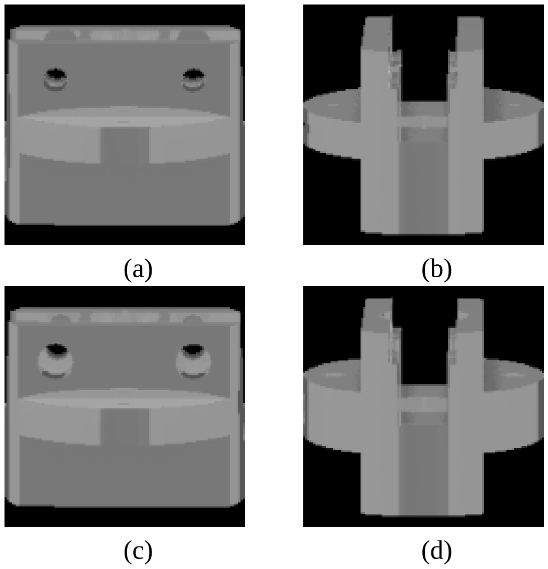


Figure 1: Socket 1 at: (a) 0° (front), (b) 90° (left) or 270° (right), and 180° (back). Socket 2 at 90° or 270°.

Our classification and pose estimation algorithms are based on the Feature Space Trajectory [2] (FST) representation for different distorted views (perspective distortion) of an object. The features are global ones not local geometric primitives such as edges, corners, or surfaces typically used in other methods [3]. In this paper, we consider global features derived from the Karhunen-Loève (KL) transform [4] (also known as principle component analysis) since this reduces the dimensionality of image data using eigenanalysis. KL features are attractive as they compress data and are easily updated when new objects are learned.

Consider an object viewed at a given range and camera depression angle as the aspect angle (rotation of the object about the axis normal to the plane that it rests upon) is changed. Each different object view is a vertex in global feature space. Vertices for adjacent views are connected by line segments to produce an FST. Different objects are represented by distinct FSTs. An input object to be recognized is represented as a test point in feature space.

In order to classify it from its features, the FST processor computes the Euclidean distance in feature space from the input test point to all FSTs known to the system. The class label corresponds to the closest FST. The pose estimate is computed by finding the projection of the test point onto the closest line segment on that FST and interpolating the pose from the known poses of the vertices of the line segment.

The FST neural net (NN) processor efficiently computes the distance from an input test point to the piecewise linear approximation of each object's trajectory. The architecture for the FST processor is shown in Fig. 2; it contains neuron planes P_1, P_2, P_3, and P_4 with N_1, N_2, etc. neurons in each plane, respectively. There are N_1 input neurons whose activation levels correspond to the feature values of the input data. The number of P_2 neurons, N_2, is the number of vertices on all FSTs. The P_1 to P_2 weights are the feature vectors corresponding to the different training set object aspect views (FST vertices). The P_2 outputs are the vector inner products of the input and the feature space vectors for all vertices. The P_3 outputs are the distances to the different line segments, and the winner take all (WTA) output denotes the class and pose estimate of the P_1 input. A very efficient neural net algorithm [2] calculates the Euclidean distances (for eight different objects and 16 aspect views per object, it requires only 10,780 on-line operations).

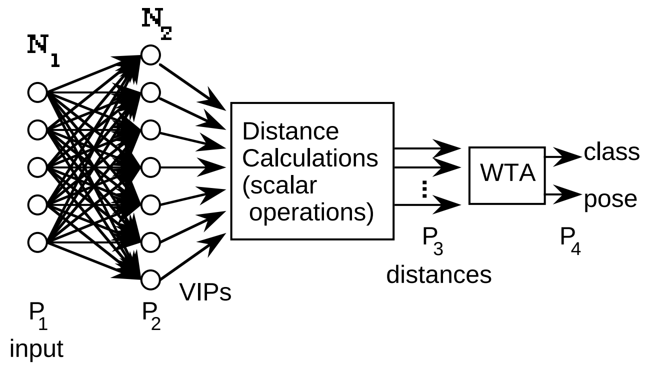


Figure 2: FST neural network architecture.

When object orientation has more than one degree of freedom (e.g., aspect and depression angles), separate FSTs may be generated to cover the degrees of freedom (a separate FST for each depression angle) or the feature space trajectory may be extended to a multidimensional Feature Space Manifold (FSM). In this paper, we use a separate FST to represent each stable resting position of an object.

Prior work [2] addressed the number of aspect views (FST vertices) required to represent an object, selection of which aspect views to use, and an adaptive algorithm to adjust FST vertex positions (P_1 to P_2 weights in Fig. 2). The ability of the FST to utilize a reduced number of object views gives it a major advantage over other distorted object representations: whether the FST is constructed from a prototype object or a computer aided design (CAD) model, a continuum of training views is possible and some means of selecting views is necessary [5,6].

We use images rendered from CAD solid object models as a convenient test vehicle for our algorithms. We can simulate wider variations in motion, lighting, and material properties on a wider array of objects with models than is possible with real images. We also construct FSTs from rendered images and use them to process real image data from actual objects. This increases cost effectiveness since it is not necessary to take a manufacturing system off-line to learn a new part [1].

The FST was originally applied to automatic target recognition of distorted views of different military vehicles in infrared images [7]. There, the emphasis was on classification rather than pose estimation. Prior work showed that the FST gave superior performance compared to other classifiers [7,8], and overcomes problems that other classifiers have including: small training set size, ad hoc parameter selection, poor generalization, selecting the number of hidden layers, estimating distributions etc.

Our prior work [9] described in detail our FST-based probabilistic object representation and our Bayesian methods for estimating pose from a single object view, classification from multiple views, and determining where to look next for more reliable class or pose estimates.

II. Probabilistic Object Representation

The FST NN processor described in Sect. I classifies objects and estimates their pose. In this section, we extend the FST concept to account for ambiguity in the classification and pose estimation process.

There are two fundamental causes for ambiguity and error in classification and pose estimation. The first is lack of information in the image (e.g. it is not possible to unambiguously determine the pose of socket 1 in Fig. 1b). The second is distortions other than pose; such as small variations in illumination, variations in object texture (dirt, rust, material properties, etc.), and sensor noise. Our methods consider both of these factors. In this section, we directly address the issue of non-pose distortions and in Sect. III we make the connection to the lack of information issue.

The FST NN classification method accounts for perspective distortions, but not for other sources of distortion. We collectively refer to non-pose distortions as "noise" since they cause undesirable variations in the features observed, and ultimately cause errors in both classification and pose estimation. We have detailed a probabilistic FST description [9] in which the observed feature values are random variables where zero-mean, white Gaussian noise (with standard deviation σ_{ω_i}) is added to the point on the FST determined by the class (ω_i) and aspect angle (θ) of the object. The result can be described by a *probability density* function (PDF) for the observed vector of features $\mathbf{x}$ conditioned on ω_i and θ. We have used this conditional PDF to show that the FST classification procedure discussed in the introduction approximates the minimum probability of error classifier and that the FST pose estimation algorithm approximates the maximum *a posteriori* pose estimate [9].

We have applied Bayesian estimation and hypothesis testing theory to these conditional PDFs in order to derive the new functions for active object recognition [9]. The next section highlights the results.

III. Active Object Recognition

This section highlights the new functions for active object recognition: confidence and uncertainty measures for estimates of the class and pose of an object, the best viewpoints of an object to use for resolving ambiguity (low confidence or high uncertainty), and fusing multiple observations from different viewpoints when estimating the class and pose of an object.

A. Classification

We determine class and classification confidence by computing the *a posteriori* class probability (conditioned on the observation $\mathbf{x}$). The object label chosen is the one with the largest *a posteriori* probability (this is, approximately, the class whose FST is closest to the observation) and the *a posteriori* probability itself is used as the classification confidence. We denote the classification confidence for a particular class ω_i, given an observation $\mathbf{x}$, by $C_{\omega_i}(\mathbf{x})$. Each time we take an observation, we compute C_{ω_i} for the most likely class $\hat{\omega}_i$. We continue to take new observations until the confidence C_{ω_i} is sufficiently high ($C_{\omega_i} > 0.95$).

Now we address the question of *where* to collect additional data, i.e. which new aspect view, or viewpoint, do we choose to resolve class ambiguity ($C_{\omega_i} < 0.95$). Consider the task of discriminating socket 1, shown in Fig. 1a-c, from socket 2 in Fig. 1d. Sockets 1 and 2 are identical except for slight differences in the spacing of the two holes and in the thickness of the central mounting bracket. Discriminating between the two sockets is easiest at 90° or 270° where the thickness of the mounting bracket is most obvious. We select the *best* view to distinguish between different objects algorithmically and *automatically*. We compute the camera motion ($\Delta\Theta$) required for the active object recognition system such that the probability of correct classification is maximized. In the two class case, we *maximize the probability of correct classification approximately by choosing the next viewpoint so as to maximize the distance between the two FSTs*. This is where information in the image ties in: the larger the difference between images, the more distant corresponding points on the FSTs will be.

We denote the best pose of an object of class ω_i to use in distinguishing it from an object of class ω_j as $\Theta_c(i,j)$. The values of $\Theta_c(i,j)$ (computed off-line and stored for each pair of objects) form a matrix which specifies the best camera view for resolving class ambiguity between any pair of objects known to the recognition system. In each iteration of an active object recognition scenario, the system makes an observation. If the classification confidence C_{ω_i} is not sufficiently high after the observation, the system notes the two most likely classes, looks up the best view for distinguishing them, and then drives the camera to that viewpoint using the pose estimate from the current observation.

Although we consider only the two most likely classes at each step, our active object recognition system still resolves cases when more than two objects may be confused. Often there is a single salient view which distinguishes a set of similar parts. Even if this is not the case, moving the sensor to the best viewpoint to discriminate between two objects after each observation tends to discriminate multiple similar objects by a process of elimination.

B. Pose Estimation

We now describe an uncertainty measure $U_{\omega_i}(\mathbf{x})$ for a pose estimate based on an observation $\mathbf{x}$. We use this measure to decide if it is necessary to collect additional data before reporting a pose estimate.

We use the expected value of the pose estimation error magnitude, conditioned on observation $\mathbf{x}$, as our uncertainty measure $U_{\omega_i}(\mathbf{x})$. A smaller value of $U_{\omega_i}(\mathbf{x})$ indicates a more reliable pose estimate. If $U_{\omega_i}(\mathbf{x})$ exceeds 3.5°, we consider the pose estimate to be unreliable and examine the object from another viewpoint.

We now discuss selection of the *best* object view to use in resolving pose ambiguity. Errors in pose estimation are likely to be larger at viewpoints where different parts of the same FST are close in feature space but far apart in aspect angle. In the case of socket 1 object in Fig. 1b, the section of the FST for views near 90° is very close to the section of the FST for views around 270°, since the differentiating object characteristics - the different sized holes in the front (Fig. 1a) and back (Fig. 1c) - are barely visible in these aspect view ranges. Thus, we expect large errors in the pose estimate around 90° and 270°; and smaller errors around 0° and 180°. These observations may be rather obvious to a human; however, the FST is attractive since it provides an *automated* way to achieve such an analysis. To *automatically* find the view of an object that results in pose estimates with the least uncertainty, we use Monte Carlo simulation techniques to find the viewpoint ($\Theta_p(\omega_i)$) which minimizes the expected value of the pose estimation error magnitude. We compute and store $\Theta_b(\omega_i)$ for each object as part of the off-line training process.

C. Multi-Observation Fusion

In active object recognition, we may take more than one observation of an object before assigning a final class label and pose estimate. We use all of the available information (multiple observations and the known relationships between them) to compute the class and pose estimates, the classification confidence, and the pose estimate uncertainty.

We compute the joint PDF for the set of observed feature vectors, conditioned on the class and *final* pose of the object, by using the known camera motion to transform each observation into a common coordinate frame (with respect to the final camera position). We also assume that the additive noise is independent for each observation. We use the resulting joint conditional PDF to derive all of the desired recognition system outputs using all of the available information.

IV. Experiments

The images in Fig. 1a-c were rendered (using ray-tracing [10]) to resemble a real prototype of socket 1 fabricated from metal (we describe the process of rendering images to resemble real images in a prior paper [11]). In our testing, we considered sockets 1 and 2, each in two different rest positions (the second rest position is shown in Fig. 6), and two different bracket parts (not shown) in three different rest positions each. Images of each object, in each rest position were rendered over the 360° aspect angle range in 3° increments. Real and rendered images were processed in a similar manner. First, the object was segmented from the background using simple thresholding. Then the object was scaled uniformly in both dimensions until it filled at least one dimension of the input frame size (128x128 pixels). Finally the energy in each image was normalized to account for global lighting differences. Eighty KL features (retaining 95% of the variability in the rendered training data) were used to extract features from both rendered and real images. Since the training set was large, the eigenspace update method [12] was used. An FST was constructed for each object, in each rest position, (for a total of 10 FSTs) from the rendered aspect views. In order to identify objects in different rest positions, the FSTs were labeled according to a scheme in which the first digit indicates the object's class and the second its rest position.

We used these FSTs - created from CAD models - to recognize real objects from real images captured in the Calibrated Imaging Laboratory (CIL) [13]. In the following subsections, we present sample results from the extensive testing we have performed in the CIL.

A. Classification

In the training process we determine and store, for each object class, the pose Θ_r of the object which best distinguishes that object from each of the other objects. $\Theta_r(\omega_i, \omega_j)$ is the training pose of the object ω_i whose FST vertex is most distant from the FST of a second object ω_j. The graphs in Fig. 3 show the distance from the FST for socket 1 in rest position 1 to socket 1 in rest position 2 (Fig. 3a), and to socket 2 in rest position 1 (Fig. 3b). In both cases, discrimination is expected to be easiest at around 90° or 270° as indicated by the peaks in the graphs in Fig. 3. This is so because the best cue for distinguishing rest position 1 from rest position 2 is that the large slot is on the top in position 1, and on the bottom in position 2; and the thickness of the center mounting bracket best distinguishes socket 1 from socket 2 in the same rest position. Both of these characteristics are most clearly visible at 90° or 270°.

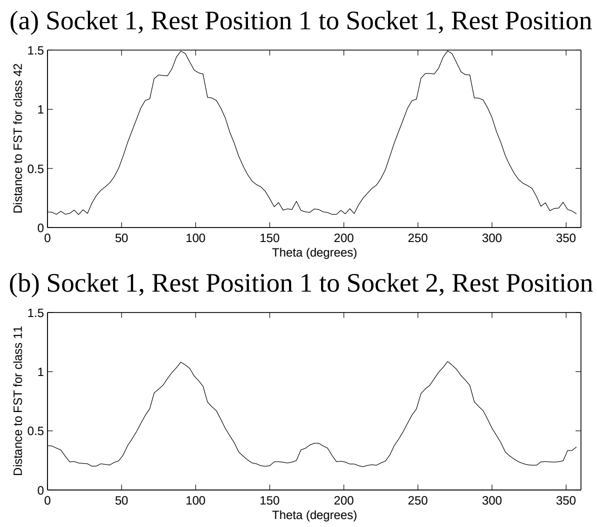
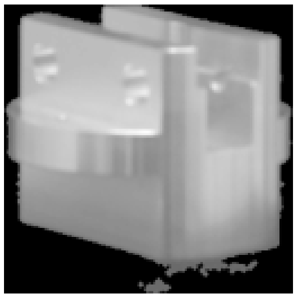


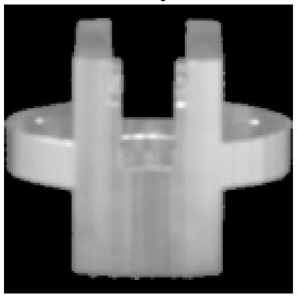
Figure 3: Inter-FST distances for FSTs produced from rendered images.

We now discuss the results of a test in which the real socket 1 was initially viewed in rest position at $\theta = 125^\circ$ (Fig. 4a). At this viewpoint, it is difficult to distinguish between sockets 1 and 2 (or to determine the rest position of either socket). There are also segmentation errors near the bottom of the image in Fig. 4a (observation x_1). For these reasons, our algorithms incorrectly identified socket 1 in rest position 2 (class 12) as the most likely object class (class 22, socket 2 in rest position 2, was the next most likely class), and the θ estimate, $\hat{\theta} = 305.3^\circ$ is off by nearly 180°; however, since $C_{\omega_i}(x_1) = 0.550$ is low, our active object recognition algorithm requests another image of the object at $\Theta_r(\omega_{12}, \omega_{22}) = 90^\circ$. Since the rotation is relative to an erroneous pose estimate, this sensor movement produced a viewpoint near 270° (at $\theta = 269.7^\circ$) instead. Even though this is not the best viewpoint that we identified in the



(a) $\hat{\omega}_i = \omega_{12}, C_{\omega_i}(\mathbf{x}_1) = 0.55, \theta = 125^\circ$, $\hat{\theta} = 305.3^\circ, \mathcal{U}_{\omega_i}(\mathbf{x}_1) = 87.5^\circ$

$\rightarrow$ Rotate by 144.7° $\rightarrow$



(b) $\hat{\omega}_i = \omega_{11}, C_{\omega_i}(\mathbf{x}_1, \mathbf{x}_2) = 1.0, \theta = 269.7^\circ$, $\hat{\theta} = 271^\circ, \mathcal{U}_{\omega_i}(\mathbf{x}_1, \mathbf{x}_2) = 0.7^\circ$

Figure 4: Active recognition scenario for real socket 1 in rest position 1 (class ω_{11}) at $\theta = 125^\circ$.

training process, the correct class and rest position are also obvious from this view. Given the combined information from these two views, we obtained high class confidence ($C_{\omega_i}(\mathbf{x}_1, \mathbf{x}_2) > 0.95$) and accepted the estimates (class $\hat{\omega}_i = \omega_{11}$ is correct and $\hat{\theta}$ is within 1.3° from Fig. 4b). It is interesting to note that *neither observation alone provided a reliable* θ *estimate* (both were in error by nearly 180° and both gave high $\mathcal{U}_{\omega_i}$ of 87.5° for $\mathbf{x}_1$ and 88.8° for $\mathbf{x}_2$ alone). *However, the combination of* both *observations, using the known object rotation, yielded an accurate and reliable* θ *estimate.*

B. Pose Estimation

The method of Sect. III.B was used to *automatically* establish the best aspect angle to use for pose estimation of each object. These aspect angles are stored by our system so that the active recognition system may be driven to the best viewpoint for estimating the pose of a given object. The estimated expected pose estimation error magnitude for socket 1 in rest position 2 is plotted as a function of pose in Fig. 5. As expected, pose estimation is predicted to be very unreliable near the sides (θ = 90° or 270°) of this object. There are viewpoints where the pose estimates are expected to be better, but estimating the pose of this object is difficult from any viewpoint due to the high degree of (imperfect) rotational symmetry. The object features which distinguish between the front and back of the object are rather subtle, resulting in significant 180° ambiguity in θ even at the best viewpoints. This point is reflected in the large minimum value (16.1°) of the expected error in Fig. 5. Thus, we do not expect that we will be able to estimate θ with acceptable uncertainty $\mathcal{U}_{\omega_i}$ for this object using *any* single observation. However, when multiple observations are combined, we have seen that $\mathcal{U}_{\omega_i}$ can be reduced to acceptable limits and the pose estimation error reduced significantly. From the plot, Θ_B for socket 1 in rest position 2 is 147° (corresponding to the lowest expected error magnitude).

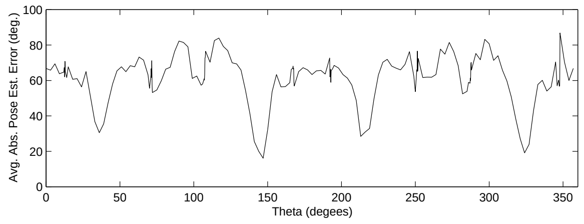
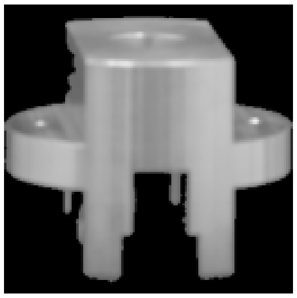
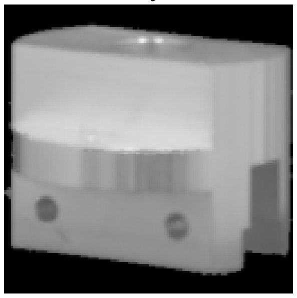


Figure 5: Estimated expected absolute pose estimation error as a function of pose for socket 1 in rest position 2.

Consider the 95° aspect view of the socket 1 object in rest position 2 (ω_{12}) as shown in Fig. 6a. This is not a training view; it is a good view for classification; but, since the holes bored through the front and back of the socket are not visible, it is not possible for the pose estimator to reliably determine if the image is from the right or left side of the object. The object is classified correctly ($\hat{\omega}_i = \omega_{12}$) after the first observation $\mathbf{x}_1$ (Fig. 6a), and the confidence level $C_{\omega_i}(\mathbf{x}_1) = 1.00$ is very high, so the class decision is accepted. The pose estimate $\hat{\theta} = 276.3^\circ$ is off by nearly 180° as anticipated, but the pose estimate uncertainty $\mathcal{U}_{\omega_i}(\mathbf{x}_1) = 88.9^\circ$ is very high, which indicates the need for another observation to improve the pose estimate. Using the initial pose estimate and the correct class, we compute the viewpoint rotation $\Delta\Theta = \Theta_B(\omega_{12}) - \hat{\theta} = 147^\circ - 276.3^\circ = -129.3^\circ$ needed to arrive at the best viewpoint of $\Theta_B(\omega_{12}) = 147^\circ$. Because the pose estimation error is -178.7° after the first observation, the system misses the best viewpoint (147°) by 178.7°. Since the class of the object is known reliably from the first observation, *only the FST for* ω_{12} *is processed to recompute the pose estimate* (this saves processing time). Although not the *best* viewpoint, the 325.7° viewpoint (Fig. 6b) is relatively good (a local minimum of Fig. 5) and yielded an accurate ($\hat{\theta} = 326.6^\circ$) pose estimate, and the low $\mathcal{U}_{\omega_i}(\mathbf{x}_1, \mathbf{x}_2) = 0.6^\circ$ indicates this automatically. Thus, is was not necessary to move again to obtain the best viewpoint Θ_B. *This test demonstrates the ability of our active object recognition system to improve the* θ *estimate by moving the viewpoint.*



(a) $\hat{\omega}_i = \omega_{12}$, $C_{\omega_i}(\mathbf{x}_1) = 1.00$
$\theta = 95^\circ$, $\hat{\theta} = 276.3^\circ$, $U_{\omega_i}(\mathbf{x}_1) = 88.9^\circ$
$\rightarrow$ Rotate by -129.3° $\rightarrow$



(b) $\hat{\omega}_i = \omega_{12}$, $C_{\omega_i}(\mathbf{x}_1, \mathbf{x}_2) = 1.00$
$\theta = 325.7^\circ$, $\hat{\theta} = 326.6^\circ$, $U_{\omega_i}(\mathbf{x}_1, \mathbf{x}_2) = 0.6^\circ$.

Figure 6: Active recognition scenario for real socket 1 in rest position 2 (class ω_{12}) at $\theta = 95^\circ$.

ACKNOWLEDGEMENTS

The support of this work by grant IRI9530657 from the National Science Foundation is gratefully acknowledged.

REFERENCES

[1] J.S. Sallade and M.L. Philpott. Synthetic template methodology for CAD directed robot vision. *Int. J. Mach. Tools Manufact.*, 37(12):1733–1744, 1997.

[2] D. Casasent and L. Neiberg. Classifier and shift-invariant ATR neural networks. *Neural Networks*, 8(7/8):1117–1130, 1995.

[3] W.E.L. Grimson. *Object recognition by computer: the role of geometric constraints.* MIT Press, Cambridge, Mass., 1990.

[4] C. W. Therrien. *Decision, Estimation, and Classification: An Introduction to Pattern Recognition and Related Topics.* Wiley, New York, 1989.

[5] M. Sipe, D. Casasent, and L. Neiberg. Feature space trajectory representation for active vision. In S. Rogers, editor, *Applications and Science of Artificial Neural Networks III*, volume 3077, pages 254–265. Proc. SPIE, April 1997.

[6] D. Casasent, L. Neiberg, and M. A. Sipe. FST distorted object representation for classification and pose estimation. *Optical Engineering*, 37(3):914–23, March 1998.

[7] L. Neiberg, D. Casasent, R. Fontana, and J. Cade. Feature space trajectory neural net classifier: 8-class distortion-invariant tests. In David Casasent, editor, *Intelligent Robots and Computer Vision XIV: Algorithms, Techniques, Active Vision, and Materials Handling*, volume 2588, pages 540–555. Proc. SPIE, October 1995.

[8] G. Brandstrom, D. W. Ruck, S. K. Rogers, and B. E. Stribling. Space object identification using spatiotemporal pattern recognition. In S. Rogers and D. Ruck, editors, *Applications and Science of Artificial Neural Networks*, volume 2760, pages 475–486. Society of Photo-Optical Instrumentation Engineers, April 1996.

[9] M.A. Sipe and D. Casasent. Global feature space neural network for active computer vision. *Neural Computation and Applications*, 7(3):195–215, 1998.

[10] J. D. Foley, A. van Dam, S. K. Feiner, and J. F. Hughes. *Computer Graphics: Principles and Practice.* Addison-Wesley, Reading, Mass., second edition, 1996.

[11] M. A. Sipe and D. Casasent. Active object recognition using appearance-based representations derived from solid geometric models. In *Symposium on Intelligent Systems and Advanced Manufacturing*, volume 3522. Proc. SPIE, November 1998.

[12] S. Chandrasekaran, B.S. Manjunath, Y.F. Wang, J. Winkeler, and H. Zang. An eigenspace update algorithm for image analysis. *Graphical Models and Image Processing*, 59(5):321–32, September 1997.

[13] R. E. Wilson and S. Shafer. Precision imaging and control for machine vision research at Carnegie Mellon University. Technical Report CMU-CS-92-118, Carnegie Mellon University: School of Computer Science, Pittsburgh, PA, March 1992.