Unrestricted Recognition of 3-D Objects for Robotics Using Multi-Level Triplet Invariants

Gösta H. Granlund and Anders Moe
Linköping University
Computer Vision Laboratory
581 83 Linköping, Sweden
gegran@isy.liu.se
moe@isy.liu.se

Abstract

A method for unrestricted recognition of 3-D objects has been developed. By unrestricted, we imply that the recognition shall be done independently of object position, scale, orientation and pose, against a structured background. It shall not assume any preceding segmentation and allow a reasonable degree of occlusion.

The method uses a hierarchy of triplet feature invariants, which are at each level defined by a learning procedure. In the feed-back learning procedure, percepts are mapped upon system states. The method uses a learning architecture employing channel information representation.

1 Introduction

The classical model for object identification consists of two steps:

1. Segmentation of object from background
2. Recognition of the segmented object

Figure 1. Classical model for object identification

The implicit assumption for this approach is that it should be possible to determine what pixels belong to a particular object, although the object is not known. This strategy may work in exceptionally simple cases, where there are universally distinguishable features, such as a distinctive color of an entire object, or the object has a known density which makes it stand out from a homogenous background. In most cases of interest, this is an unrealistic assumption. Objects generally do not have homogenous regions or universally distinctive features. Rather they may appear towards a structured background of a similar character, or mixed with other similar objects.

This classical strategy is consequently not usable for any situation of realistic complexity in vision: It is not possible to find with any confidence the pixels which constitute an object before the object has been recognized. This constitutes the fundamental inverse problem of vision. It is necessary to somehow perform a segmentation and a recognition in the same process.

The case considered here is the recognition of a 3-D object given a 2-D projection, such as a camera image. This gives in addition a large variation in the appearances of just a single object. Various approaches to this problem have been explored: [10, 11, 9, 16, 13, 14, 18, 19, 12, 1, 17, 15].

A major problem has been to get a sufficient resolving power from the primitives used, to potentially deal with a large number of objects in several views. This is what the proposed hieararchical structure of primitives is intended to resolve.

Object identification is an inverse problem, in that a hypothesis of structure first has to be made, against which the measurements performed are interpreted. The earlier formulation has certain relevance in that it suggests that recognition implies a two-step process:

1. Postulation of a certain model
2. Performing measurements, and comparing these with a reference, under the assumption of the particular model

Var	Object characterization
ω	Object class
x	Horisontal position of object
y	Vertical position of object
ϕ	Horisontal pose angle of object
θ	Vertical pose angle of object
ψ	Orientation of object in image plane
s	Scale or size of object

Table 1. Parameters of variation.

The main issue is how to select hypothetical models which are descriptive; models which can deal with a structure of reasonable complexity and can be handled efficiently computationally.

In order to use learning for the acquisition of models of sufficiently complex objects, a new structure has been developed using a hierarchy of partially invariant triplet primitives, describing an object in a view-centered fashion [4]. The objective is to assign objects to a class, but it is believed that this requires a simultaneous estimation of some subset of its parameters of variation [3], according to Table 1. These can also be viewed as system states to be estimated.

2 Characteristics of Model Structure

It has for a long time been believed in vision research that a robotics system should have a structure like in Figure 2a. The first part should use the incoming image information to produce a *description* of the image, where different objects are recognized and assigned to the proper categories, together with information about position and other relevant parameters. A second unit would now use this information to produce actions into the physical world, e.g. to implement a robot.

This structure has not worked out very well for a number of reasons, which we will have to omit in this discussion, but reference is made to [3, 4]. It turns out that in fact the order between the parts shall be the *opposite*. See Figure 2b).

The first part of the system is a reactive percept-to-action mapper. After this follows if necessary for the application a part which performs a symbolic processing for categorization and reasoning, in brief what we refer to as AI, and communication.

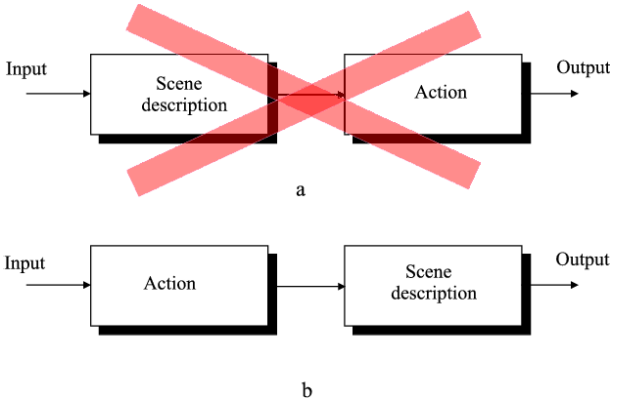

Figure 2. a) Classical robotics model. b) Perception-Action robotics model

The distinctive characteristic of this structure is that percepts are mapped directly onto *actions or system states*, rather than descriptions, as it was the case in Figure 2a). The reason is that this strategy allows the system to learn objects and other aspects of the environment by itself. Descriptions, of which assignment to category is one example, are generated in the symbolic part of the structure, for communication to other systems or for use in symbolic reasoning.

An important issue is that learning of an object is not just to identify its category, but to identify its position, pose, orientation, and to learn what action complexes it can be linked to for handling. This is what understanding of an object implies. This information can later be used as contextual parameters.

The model structure developed, has a number of characteristics:

1. Models shall be fragmentable such that a certain model can be a part of a more complex or higher order model. Due to this recursive character, we will simply denote them all models, be it parts or combinations.
2. Learning of models shall proceed from lower levels to higher levels.
3. Acquired lower level models shall be usable as parts of *several* different higher order models.
4. A particular model is only acquired once, and its first occurrence is used as the representation of that model.

2.1 Triplet Models

The basic model or *primitive* consists of a set of 3 point features, $\mathbf{f}_k$, at positions $\mathbf{p}_k$, joined to form a *triplet*. See

Figure 3.

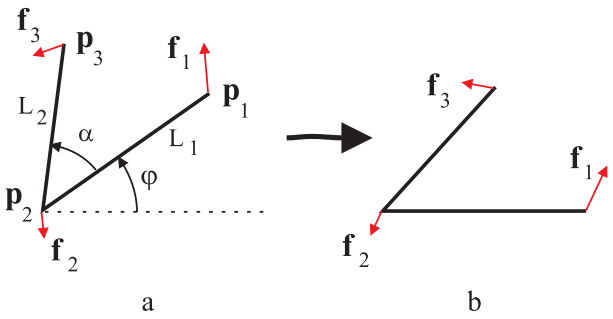

Figure 3. a) Triplet with parameters indicated. b) Triplet rotated to normalized orientation.

Point features are vectors, representing sparse, localized properties of an image such as corners, curvature, centers of homogeneous regions, mid-points of lines, etc. A point feature, $\mathbf{f}_k$, can also represent an entire lower level triplet attached, whereby multi-level triplets are formed. See Figure 4.

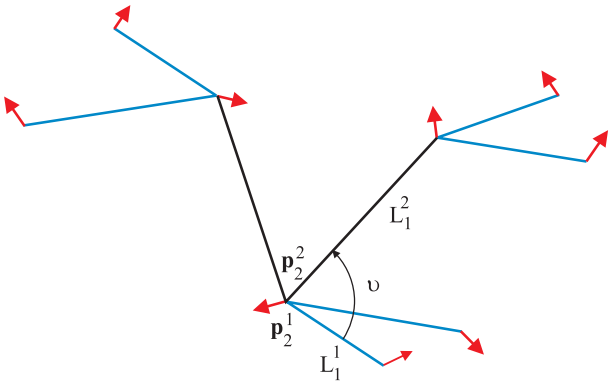

Figure 4. Two levels of triplets, with relational parameters indicated.

There are many ways in which a number of points can be brought into groups. The triplet structure has some attractive properties to ensure a certain degree of uniqueness:

- It allows a unique ordering of the feature points, which is implemented such that the triplet is "right oriented", i.e. that the angle $\alpha < \pi$, as defined in Figure 3.
- The triplet structure allows us to define a scale invariant structure parameter $\gamma = \frac{L_1 - L_2}{L_1 + L_2}$
- The distance between the two feature points not connected by the triplet, must be shorter than the two other distances between feature points, see Figure 3.
- The triplet can be brought into a "normal orientation" by aligning leg L_1 to make $\varphi = 0$, see Figure 3 a) and b).

The preceding properties together with the hierarchical arrangement of triplets make the following parameter variations *trivial*:

- Orientation in the image plane
- Scale
- Object position in x and y

as this reduces the dimensionality of the total system, such that the variations in these parameters can be handled without extending the training space. This implies effectively an *invariance* of the primitives with respect to these parameters.

To decrease the combinatorial complexity and to improve the robustness, additional restrictions, grouping rules and criteria for acceptance of triplets have been devised. Only two will be mentioned here:

- **Spatial grouping range:** We expect primitives to increase in spatial size going towards higher levels. Two point feature vectors $\mathbf{f}_1, \mathbf{f}_2$ with positions $\mathbf{p}_1, \mathbf{p}_2$ can be connected as a part of a triplet if $min_d < |\mathbf{p}_1 - \mathbf{p}_2| < max_d$, where min_d, max_d are the minimal and maximal allowed distance thresholds between the features, respectively. Mechanisms for adaptative generation of these tresholds are not trivial, but outside the scope of this presentation.
- **Object closure criteria:** Tests for homogeneity such as similar density or color inside the triplets, to indicate parts of a common object or region. Tests for texture or conflicting structures may reject the hypothesis of primitive, or constitute an additional descriptive feature.

3 Channel Information Representation

The information representation used for all parameters is a *monopolar channel representation* [5]. The *channel* representation implies a mapping of signals into a higher-dimensional space, in such a way that it introduces locality in the information representation with respect to all dimensions; geometric space as well as property space. This allows a generation of different models for different parts of the input feature space. For the simultaneous representation of two values of a 1-D scalar variable, it may appear as in Figure 5.

A 2-D version of this representation is directly available from wavelets or filter outputs. For a more extensive discussion, see [5].

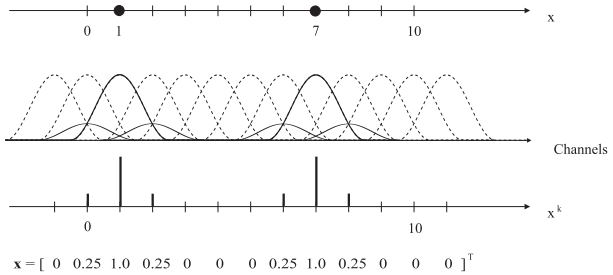

Figure 5. Channel representation as a vector $\mathbf{x}$, of two scalars $x = 1$ and $x = 7$.

The *monopolar* property implies that data only utilizes one polarity, e.g. only positive values, in addition to zero. This allows zero to represent not just another value, such as temperature zero as opposed to other values of the temperature, but to represent *no information*. In our case with local point features, only limited parts of the spatial domain will have non-zero contributions. This provides the basis for a *sparse* representation, which gives improved efficiency in storage and better performance in processing.

The locality allows for a fast convergence in the optimization process to solve for the linkage matrices [5].

4 First Level Triplets

The first level triplet provides the interface between the feature set used and the triplet structure. The image features used in the subsequent example are curvature features [6, 8], but any local interest points or sparse features representable in a single vector can be used. See Figure 6. Curvature is originally represented by a complex number, where the argument gives the direction to the center of curvature. The angle between this complex number and the triplets first leg L_1 (performing orientation normalization) is channel coded to give a feature vector $\mathbf{f}_k$.

A triplet can be characterized in a number of equivalent fashions [7]. The components used are as illustrated in Figure 3.

- $\mathbf{f}_k$: Point feature vectors, $k = 1, 2, 3$, each one coded with h_f channels.
- $\boldsymbol{\alpha}$: Angle between triplet legs 1 and 2, coded with h_α channels.
- $\boldsymbol{\gamma}$: Relative length of triplet legs, $\gamma = \frac{L_1 - L_2}{L_1 + L_2}$, coded with h_γ channels.

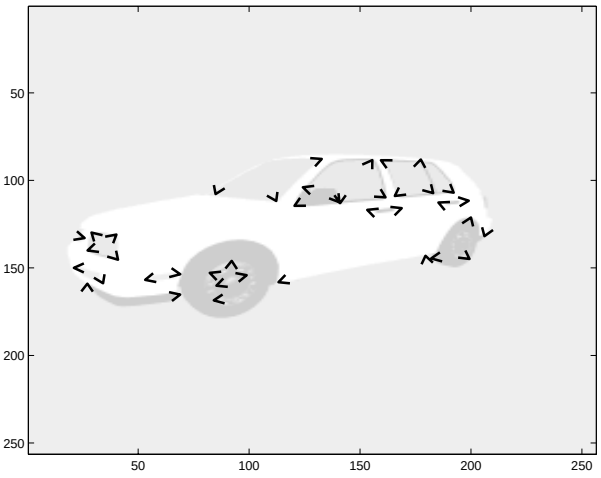

Figure 6. Example of sparse curvature features.

A triplet will be represented by the Kronecker product, $\mathbf{a}$, between the preceding components:

$$\mathbf{a} = \mathbf{f}_1 \otimes \mathbf{f}_2 \otimes \mathbf{f}_3 \otimes \boldsymbol{\alpha} \otimes \boldsymbol{\gamma} \tag{1}$$

We denote $\mathbf{a}$ a *triplet vector*. Feature vectors $\mathbf{f}_k$ as well as the other parameters may typically each be represented by 8 - 12 channels. For a case where $h_f = h_\alpha = h_\gamma = 8$, we obtain for $\mathbf{a}$ an $h_a = (h_f)^3 h_\alpha h_\gamma = 8^5 = 32768$ channels or components. This may seem like a large number, but the sparse character of the representation makes computations fast.

5 Mapping to Dynamic Binding Variables in the Form of System States

From the purely geometric and feature related entities, it is desired to map into variables that are object-related. These are variables which fulfill two requirements:

- They change as a consequence of manipulation of the object, which is essential
- They can be expected to be shared with, or at least coupled to, other primitives at the same level, or at a different level

If primitives are part of the same object, they will be subjected to transformations which may not be identical, but coupled. This allows us to build up more complex models of connected primitives. This is a variety of the classical binding problem [2].

Object states which may be suitable as dynamic binding variables can be found in Table 1. Suitable choices are all or a subset of a state vector $\mathbf{u}$:

$$\mathbf{u} = \begin{bmatrix} \mathbf{u}_{\phi} \\ \mathbf{u}_{\theta} \\ \mathbf{u}_{\varphi} \\ \mathbf{u}_{L_1} \end{bmatrix} = ch \begin{bmatrix} \phi \\ \theta \\ \varphi \\ L_1 \end{bmatrix} \tag{2}$$

where ch stands for channel coding of the scalar variables. Vector components $\mathbf{u}_{\varphi}$ and $\mathbf{u}_{L_1}$ can be viewed as local varieties of ψ and s.

We can view the triplet vector $\mathbf{a}$ as a function f of the total system state vector $\mathbf{u}$:

$$\mathbf{a} = f(\mathbf{u}) \tag{3}$$

We assume that the system state, $\mathbf{u}$, can be expressed as a linear mapping:

$$\mathbf{u} = \mathbf{C}\mathbf{a} \tag{4}$$

where $\mathbf{C}$ is a linkage or mapping matrix to be determined. In a training procedure, observation pairs $\{\mathbf{a}(n), \mathbf{u}(n)\}$, for a total of N samples, constitute matrices $\mathbf{A}$ and $\mathbf{U}$. Matrix $\mathbf{C}$ is the solution of

$$\mathbf{U} = \mathbf{C}\mathbf{A}. \tag{5}$$

The linkage matrix $\mathbf{C}$ contains hundreds or thousands of different models, each one valid within some subspace of the vector space of $\mathbf{A}$ mapping onto some subspace of the vector space of $\mathbf{U}$. The effect of the fragmentation into the channel representation is to generate separable subspaces for the original, scalar variables. In addition, this leads to a fast optimization.

The details of this solution procedure will be omitted in this presentation, but further details are given in [5].

6 Higher Level Triplets

The generic structure assumes a higher level triplet, which has lower level triplets attached at its nodes, each one described by the vector $\mathbf{f}_k$. Specifically in the higher level triplets, the feature vectors $\mathbf{f}_{\mathbf{k}}^{\kappa}$ are constructed by combining the following channel coded features: $\hat{\mathbf{u}}_{k\boldsymbol{\phi}}^{\kappa-1}$, $\hat{\mathbf{u}}_{k\boldsymbol{\theta}}^{\kappa-1}$, $\hat{\mathbf{u}}_{k\boldsymbol{\varphi}}^{\kappa-1}$, $\hat{\mathbf{u}}_{k\mathbf{L_1}}^{\kappa-1}$, $\boldsymbol{\upsilon}_k^{\kappa}$, $\boldsymbol{\Gamma}_k^{\kappa}$ where κ is the level of the triplet, υ^{κ} is the relative orientation between the level κ triplet and level $\kappa-1$ triplet and $\Gamma^{\kappa} = \frac{L_1^{\kappa} - L_1^{\kappa-1}}{L_1^{\kappa} + L_1^{\kappa-1}}$, see Figure 4. Currently the combination is done in the following way

$$\mathbf{f}_{\mathbf{k}}^{\kappa} = \begin{bmatrix} \hat{\mathbf{u}}_{k\boldsymbol{\phi}}^{\kappa-1} \\ \hat{\mathbf{u}}_{k\boldsymbol{\theta}}^{\kappa-1} \\ \hat{\mathbf{u}}_{k\boldsymbol{\varphi}}^{\kappa-1} \\ \hat{\mathbf{u}}_{k\mathbf{L_1}}^{\kappa-1} \end{bmatrix} \otimes \boldsymbol{\Gamma}_k^{\kappa} \otimes \boldsymbol{\upsilon}_k^{\kappa} \tag{6}$$

but there are other possibilities given the computational complexity accepted.

Feature vectors, $\mathbf{f}_k$, may contain certain components, which are orientation dependent, and will likewise be subjected to the orientation normalization of the triplet.

7 Mapping for Higher Level Triplets

In this case, a triplet vector is generated which is *separate* for each one of the three nodes:

$$\mathbf{a}_k^{\kappa} = \mathbf{f}_k^{\kappa} \otimes \boldsymbol{\alpha}^{\kappa} \otimes \boldsymbol{\gamma}^{\kappa} \qquad k = 1, 2, 3 \tag{7}$$

A training process generates one linkage matrix $\mathbf{C}_k$ for each one of the nodes.

$$\mathbf{U} = \mathbf{C}_k \mathbf{A}_k \qquad k = 1, 2, 3 \tag{8}$$

In the recognition phase, estimates can be computed for the state variables:

$$\hat{\mathbf{u}}_k = \mathbf{C}_k \mathbf{a}_k \qquad k = 1, 2, 3 \tag{9}$$

What this means is that a feature vector $\mathbf{f}_k$ is interpreted under the contextual restriction or modification $\boldsymbol{\alpha} \otimes \boldsymbol{\gamma}$. Given that we deal with measurements upon the same object, there are parameters which should be estimated to the same value, e.g. the pose angles ϕ and θ.

In an ideal case, such state estimates should all be equal:

$$\hat{\mathbf{u}}_{1\boldsymbol{\phi}} = \hat{\mathbf{u}}_{2\boldsymbol{\phi}} = \hat{\mathbf{u}}_{3\boldsymbol{\phi}} \tag{10}$$

$$\hat{\mathbf{u}}_{1\boldsymbol{\theta}} = \hat{\mathbf{u}}_{2\boldsymbol{\theta}} = \hat{\mathbf{u}}_{3\boldsymbol{\theta}} \tag{11}$$

In reality there is noise, which requires a consistency check between the tree statements, and confidence measures can be derived from the similarity of statements. This can be seen as the generic procedure for higher level triplets where the use of consistency checking reduces the complexity in the mapping from each feature vector. For first level triplets, the feature complexity is generally lower, which allows the mapping described earlier.

8 Removal of Multiple Models

A *model* constitutes a subset of the linkage matrix $\mathbf{C}$, which maps a subset of vectors $\mathbf{a}(n)$ onto a corresponding subset of state vectors $\mathbf{u}(n)$, such that for each sample n_1 there exists at least another sample n_2, such that:

$$\frac{<\mathbf{a}(n_2)|\mathbf{a}(n_1)>}{|\mathbf{a}(n_2)||\mathbf{a}(n_1)|} > a_c \tag{12}$$

and

$$\frac{<\mathbf{u}(n_2)|\mathbf{u}(n_1)>}{|\mathbf{u}(n_2)||\mathbf{u}(n_1)|} > u_c \tag{13}$$

Subsets of samples which fulfill this requirement and form a group of connected samples, form patches in both spaces with a continuous mapping. The sparse and localized representation allows a linkage matrix $\mathbf{C}$, to contain thousands of different models which are each continuous but form discrete patches in both spaces. We have to omit a detailed discussion of this fitting using localized, continuous models.

Every model in the set must be unambiguous in that it maps only onto a single state $\mathbf{u}$ for a given input $\mathbf{a}$. On the other hand, different inputs, $\mathbf{a}$, may map onto the same state $\mathbf{u}$. This can be resolved by removing a later appearing feature vector $\mathbf{a}(n_2)$ and state vector $\mathbf{u}(n_2)$ from the training set, where inequality 12 is satisfied but not inequality 13.

The preceding procedure allows us to use lower level models in the assembly of higher level models for entirely different objects. In this case, the intermediary triplet output variables, will have nothing to do with the actual parameters in the current training, but act as an object identity in a "local language" which is retransformed in the training to the next level triplet.

9 Responses

The number of responses obtained from an object depends on the density of appropriate features and the restriction criteria applied. The strategy is to select the criteria to permit a sufficient number of responses which can cluster to robust estimates of rotation and scale.

Fundamental to the strategy is that several of the hypothetical models may be erroneous, but there should be a sufficient number of selected hypothetical models which are correct. The way they know that they are correct is that they are saying the same thing. In more precise terms this means that outputs cluster.

9.1 Clustering of the responses pose-x ($\hat{\phi}(n)$) and pose-y ($\hat{\theta}(n)$)

If a known object is present, there should be a cluster of estimates around the object's pose-x and pose-y angles. A confidence measure is computed, dependent upon the spread of the cluster. Each cluster with a confidence above some threshold indicates an object with the pose angles given by the cluster position. The position(s) of the object(s) is then estimated by making the same clustering on the positions of the triplets giving the responses in the clusters. This gives the ability to find several objects with the same pose angles in the same image.

If a certain multi-level triplet gives a statement which is consistent with that of most other triplets (near the center of the cluster), we believe that it "belongs" to the object under examination. This triplet can then be used to make other statements about the object such as its class, its orientation or scale. This means that the triplet is trained to map onto these variables.

9.2 Clustering of the responses orientation $\hat{\varphi}(n)$ and length $\hat{L}_1$

The clustering of the orientation and length responses are made separately and only on the responses from highest order triplets remaining after the global clustering of pose-x and pose-y. The orientation estimates of the object are obtained by first calculating the difference between the orientation responses and the corresponding orientations of the triplets in the image $\varphi(n)$.

$$\hat{\psi}(n) = \hat{\varphi}(n) - \varphi(n) \tag{14}$$

To get a unique angle, modulo 2π of $\hat{\psi}(n)$ is calculated. This angle gives the orientation of the triplet compared to the orientation of the triplet during the training, and should consequently be the same for all responses obtained from the object. The position of the cluster formed from all samples, gives the orientation.

The scale is estimated in a similar way by dividing the derived estimate for $\hat{L}_1$ by the triplet length L_1

$$\hat{s}(n) = \frac{\hat{L}_1(n)}{L_1(n)} \tag{15}$$

$\hat{s}(n)$ is channel coded and the scale estimate is obtained with a least squares fit for the estimates close to the cluster

$$\hat{s} = \frac{\sum_n \hat{L}_1(n) L_1(n)}{\sum_n L_1(n)^2} \tag{16}$$

10 Experiments

The recognition structure has been trained on computer generated images of a car. Since the triplet representation is invariant to translation, rotation and scale it is only necessary to train for different pose angles (θ, ϕ), see Figure 7. The pose angles are varied with 5 degree increments between 50-90 degrees for θ and between $0 - 180$ degrees for ϕ, which gives 333 images. With about 20 first level triplets in each image this gives 5781 first level triplets and 27750 second level triplet structures.

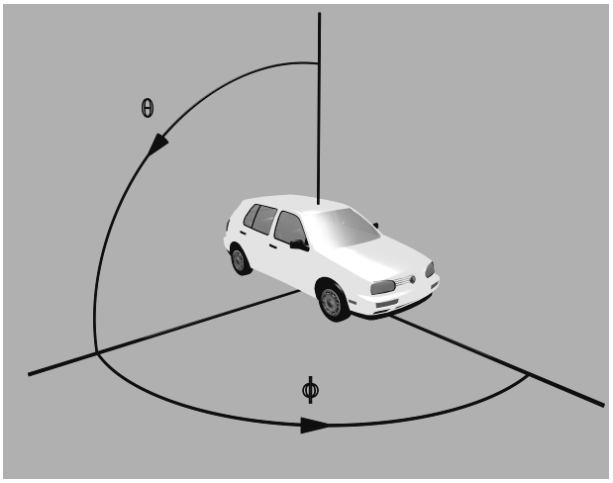

Figure 7. Object training setup. θ, ϕ are the two pose angles.

The evaluation images are generated to be different from the training set with 2.5 degrees added to the earlier pose angles. One of the evaluation images together with the obtained pose estimates are shown in Figure 8. The orientation estimates and the scale estimates are shown in Figure 9. One can see that the estimates of the pose angles and the orientation are quite stable while the scale estimates are more noisy. The estimates are then clustered to give the pose, orientation and scale of the object. The resulting estimation errors for this image are $0.1°, 0.4°, 4.0°$ and 2.5% for pose-x, pose-y, orientation and scale respectively. The average estimation errors for these estimates for the evaluation images are given in table 2.

Since the curvature features are not totally scale invariant we will not get as good estimates if we scale the object. Figure 10 shows the car rotated 60 degrees and scaled 20%. The image size is kept constant while scaling the object, so occlusion will occur and affect the estimates as well. Figure 11 shows the orientation and scale estimates. The estimation errors for this image are $1.4°, -2.3°, 8.5°$ and -2.4% for pose-x, pose-y, orientation and scale respectively. The

Estimate	Average error
pose-x	$1.3°$
pose-y	$1.2°$
Orientation	$4.4°$
Scale	4.2%

Table 2. Average error for evaluation images.

Estimate	Average error
pose-x	$1.8°$
pose-y	$1.8°$
Orientation	$6.3°$
Scale	4.9%

Table 3. Average error for evaluation images scaled 10% and rotated 60 degrees.

average estimation errors for the object rotated 60 degrees and scaled 10% are shown in table 3.

In Figure 12 two car objects have been inserted in a natural, structured background. In addition, the illumination has been changed and partial occlusion between the objects occurs. One can see that the background has very little influence on the results. The obtained estimation errors are for the right car $-0.9°, 1.5°, 8.6°, 1.1\%$ and for the left car $-0.6°, 0.8°, 0.1°, 5.5\%$ for the pose-x, pose-y, orientation and scale respectively.

11 Recognition of Object Class

In the preceding presentation, there has for reasons of space, not been much discussion about mapping into object class. This can however be implemented in the same fashion as earlier described for the properties orientation and length.

The crucial thing is the use of the dynamic binding parameters to establish if correct hypothetical models for different levels have been selected. If this is proved to be true, we can use the same triplet models to map onto a suitable class membership variable. While the recognition of multiple objects is not trivial, in that it requires a larger data set with increased risk for confusion, we believe that the crucial mechanisms are those discussed in more detail in the paper. What gives us a benefit with this approach is that lower level models earlier acquired, can be used in the recognition of other objects. This means that learning of objects can be made in an incremental fashion. This in turn means that the complexity of data expands at a rate less than linear, with respect to the number of objects. How much less it expands, is an issue which has to be left out from this presentation.

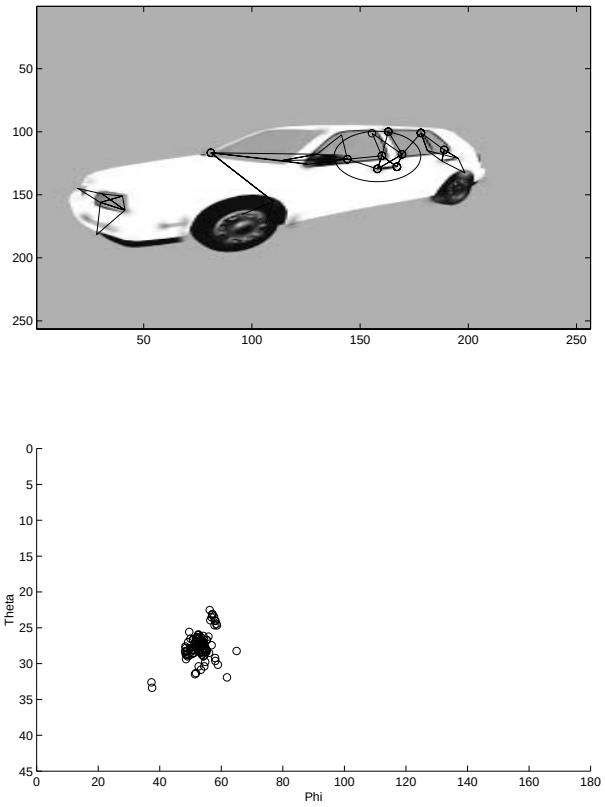

Figure 8. Example of triplets found in a certain pose. The lines are the first level triplets, the small circles indicates the accepted triplets and the large circle is the estimated position of the object. The cluster for the estimated pose parameters is given in the lower part of the figure.

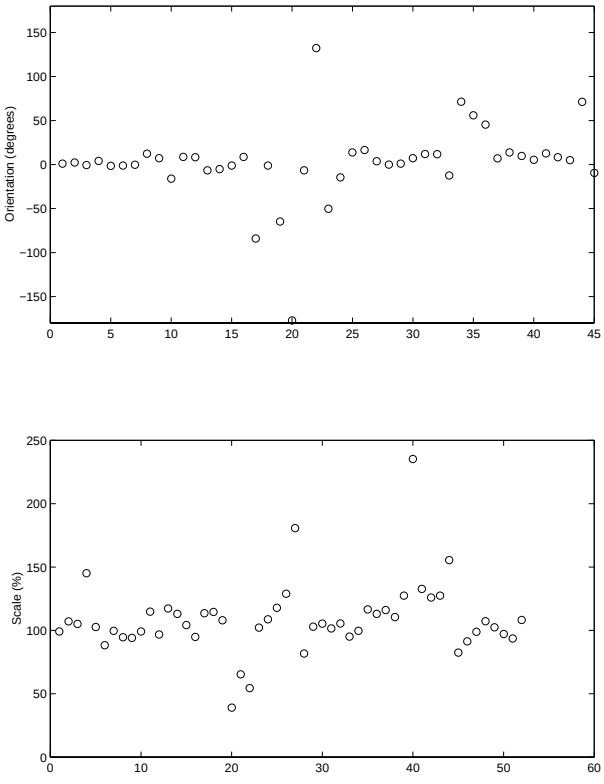

Figure 9. Estimates of the orientation and scale for accepted triplets in Figure 8.

Acknowledgments

The work presented in this report was supported by WITAS, the Wallenberg laboratory on Information Technology and Autonomous Systems, and by EU project VISATEC, which is gratefully acknowledged. The work builds upon several methodologies earlier developed at the Computer Vision Laboratory.

References

[1] P. Besl and R. Jain. Three-dimensional object recognition. *ACM Computing Surveys*, 17:75–145, 1985.

[2] J. Fodor and Z. Pylyshyn. *Connections and Symbols*, chapter Connectionism and cognitive architecture: a critical analysis, pages 3–71. MIT Press, 1988. S. Pinker and J. Mehler, Eds.

[3] G. Granlund. Does Vision Inevitably Have to be Active? In *Proceedings of the 11th Scandinavian Conference on Image Analysis*, Kangerlussuaq, Greenland, June 7–11 1999. SCIA. Also as Technical Report LiTH-ISY-R-2247.

[4] G. H. Granlund. The complexity of vision. *Signal Processing*, 74:101–126, April 1999.

[5] G. H. Granlund. An associative perception-action structure using a localized space variant information representation. *Algebraic Frames for the Perception-Action Cycle (AFPAC)*, pages 29–53, September 2000.

[6] G. H. Granlund and H. Knutsson. *Signal Processing for Computer Vision*. Kluwer Academic Publishers, ISBN 0-7923-9530-1, 1995.

[7] M. Isaksson. Face detection and pose estimation using triplet invariants. Master's thesis, Linköping University, SE-581 83 Linköping, Sweden, 2002. LiTH-ISY-EX-3223.

[8] B. Johansson and G. H. Granlund. Fast selective detection of rotational symmetries using normalized inhibition. *Proc. of the 6th ECCV*, 1:871–887, June 2000.

[9] K.Mikolajczyk and C.Schmid. Indexing based on scale invariant interest points. *International Conference on Computer Vision*, pages 525–531, July 2001.

[10] D. G. Lowe. Object recognition from local scale-invariant features. *International Conference on Computer Vision*, pages 1150–1157, September 1999.

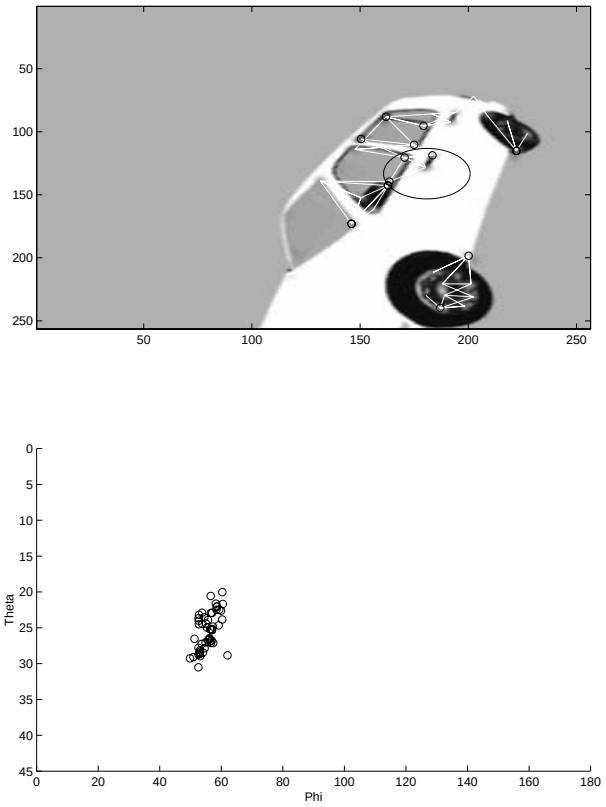

Figure 10. Example of recognition of pose for a rotated, partially occluded object, at a scale different from training. The lines are the first level triplets, the small circles indicates the accepted triplets and the large circle is the estimated position of the object.

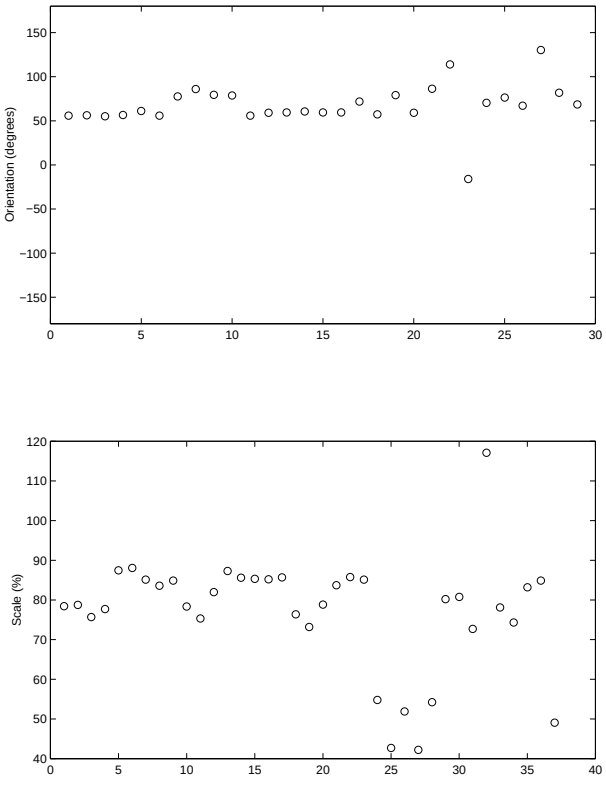

Figure 11. Estimates of the orientation and scale for the accepted triplets, of object in Figure 10.

[11] D. G. Lowe. Local feature view clustering for 3d object recognition. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 682–688, December 2001.

[12] G. Mamic and M. Bennamoun. Representation and recognition of 3d free-form objects. *Digital Signal Processing.*, 12:47–76, 2002.

[13] J. Matas, J. Burianek, and J. Kittler. Object recognition using the invariant pixel-set signature. *Proc British Machine Vision Conference BMVC2000*, 2:606–615, September 2000.

[14] H. Murase and S. K. Nayar. Visual learning and recognition of 3-d objects from appearance. *International Journal of Computer Vision*, 14:5–24, 1995.

[15] B. Schiele and A. Pentland. Probabilistic object recognition and localization. *ICCV*, pages 177–182, 1999.

[16] C. Schmid and R. Mohr. Local greyvalue invariants for image retrieval. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:530–535, May 1997.

[17] C. Schmid, R. Mohr, and C. Bauckhage. Evaluation of interest point detectors. *International Journal of Computer Vision.*, 37(2):151–172, 2000.

[18] I. Weiss and M. Ray. Model-based recognition of 3d objects from single images. *IEEE Transactions on Pattern Analysis and Machine Intelligence.*, 23, February 2001.

[19] C. Yuan and H. Niemann. Neural networks for the recognition and pose estimation of 3d objects from a single 2d perspective view. *Image and Vision Computing.*, 19:585–592, 2001.

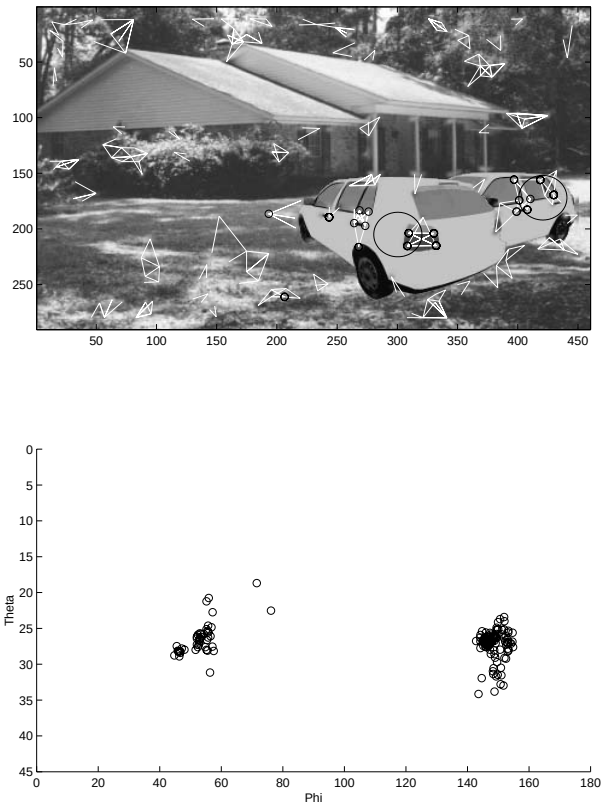

Figure 12. Test of recognition of pose for two partially occluded objects against a structured background. The lines are the first level triplets, the small circles indicates the accepted triplets and the large circle is the estimated position of the object.